NUMERICAL CONTROL PROGRAMMING LANGUAGES

FOR LATHES

NUMERICAL CONTROL PROGRAMMING LANGUAGES FOR LATHES

by

GERALD C. DUNSFORD, B. Sc.

Thesis

Submitted to Faculty of Graduate Studies

in Partial Fulfilment of the Requirements

For the Degree

Master of Engineering

McMaster University

September, 1972

MASTER OF ENGINEERING
(Production)

McMASTER UNIVERSITY
Hamilton, Ontario

TITLE: Numerical Control Programming Languages For Lathes

AUTHOR: Gerald Charles Dunsford, B.Sc. (University of Aston)

SUPERVISOR: Dr. J. Tlusty

NUMBER OF PAGES: ix, 114

PREFACE

In the last decade the concept of Numerical Control has spread very rapidly to the lathe. Numerical Control was developed primarily as a means of manufacturing and checking the complicated shapes required for space age aircraft and missiles. Typical lathe work did not require this sophistication and thus, at the time, was not considered applicable for Numerical Control.

However, with the development of simpler controls, the cost factor dropped and manufacturers began to realize the usefulness of numerically controlled lathes. This, naturally, has led to a boom in programming languages applicable for Numerical Control lathe work.

With all these languages available it is interesting to note their differences and their similarities. It is the purpose of this thesis to expose these differences in an attempt to guide potential users in the choice of a suitable programming language for numerically controlled lathes.

ACKNOWLEDGEMENTS

The author would like to express his appreciation for the assistance and direction of Doctor J. Tlusty throughout the duration of this project.

The assistance of Mr. Dennis Fern and Stelco proved invaluable when dealing with the Compact II language, as did that provided by Mr. Charles Puksta and Mr. Ted Goddard of Jones and Lamson in the area of manual programming.

The author is deeply indebted to his wife, Dianne, and Miss Elaine Spurr for their excellent work in the typing of this manuscript.

TABLE OF CONTENTS

APPENDIX

LIST OF FIGURES

LIST OF TABLES

NUMERICAL CONTROL PROGRAMMING LANGUAGES FOR LATHES

1.0 INTRODUCTION

Numerical Control languages are the means by which man can communicate with a computer in order to ease the burden of providing the tape used to control the actions of a particular machine tool. These tapes can, of course, be produced without the use of a computer. This, however, can be a long and tedious process and is subject to errors.

Lathe numerical control languages can be classified according to many different criteria. They could, for example, be classified according to input, that is fixed format or free format. Another classification could be according to the computer system they are available on e.g. time-sharing, in-house, etc. The classification used in this thesis will be one suggested by Professor J. Peters[8]. The languages are split into three groups according to their automation level. The automation level of a language is defined as the number of operations that language integrates in the program. The operations referred to are such things as determination of tools, choice of cutting conditions etc.

This type of classification also shows the basic development of programming languages for lathes. It shows the trend from a language

whereby the programmer enters in a description of the part, plus the required tool motions, to a language whereby the programmer describes the tool material, the stock description and the finished part description leaving the processor to determine tool motions etc.

Lathe programming languages have come a long way in the past few years. It is now possible, thanks to machining research and group technology to have the computer determine the various cutting parameters, the optimum sequence of cuts, and the choice of machine. By relieving the part programmer of such decisions it is possible to more fully utilize the lathe and also provide for standardization of methods, and thus optimization, from lathe to lathe.

The application of numerical control to turning machines began much later than the development of numerical control drilling and milling machines. The advantages associated with numerical control lathes have now been appreciated and according to a paper by A.E. DeBarr[20] in 1971 the number of lathes is rapidly approaching the number of drilling machines in both U.S.A. and U.K. Another interesting statistic was given in the results of a long range technological forecasting study carried out by Birniehill Institute in 1971. It was claimed that in 1987 50% of all machine tools will be numerically controlled. This is a far cry from the less than 1% now in existance.

Numerically controlled lathes require only a two axis control system per one tool carrier. There may, of course, be two, or even more, tool carrier motions commanded simultaneously. For simple parallel turning only a straight line contour system is required, but most lathes are fitted with continuous path control with interpolation on two axes. This allows for the turning of any shape with an axis of symmetry.

Any number of successive cuts can be taken following any path, with any available feed rate or at a varying rate of feed. Straight lines on each axis, tapers, clockwise or anti-clockwise arcs or circles can also be shaped.

Numerically controlled lathes utilize preset tooling in order to simplify set up and part programming. In most cases throw-away tips are used, intending that the tool holders are not changed. Most lathes are equipped with individual tool off-set dials. These allow for compensation for tool wear, and for exact setting of the tools.

2.0 MANUAL PROGRAMMING[3,4,27]

Manual or Hand Programming is the forerunner of all the numerical control languages in existance. It is used either as the main method of programming a machine or, in conjunction with computer assisted programming, as a means of altering and/or correcting a tape without requiring a new computer run. Thus hand programming is really a must for all programmers.

Hand programming requires the programmer to use the control language to encode the commands. In order to assist the programmer standard coding forms are used, a typical example of which is shown in Figure 1.

As can be seen from the headings on this form the programmer has to include all the appropriate command codes as well as a co-ordinate description of the required tool path.

The tool path is that which describes the cutting tool tip and this can be done using either the tangent point or the tool tip centre. Figure 2 shows an example of programming using both tangent point and tool tip centre. It is advisable in practice to use one system only. The tool tip centre approach is probably a little easier to work with because the tangent point varies when contouring motions are required.

The program uses the co-ordinates of each turning point to determine the direction that the tool will move. A turning point is the intersection of two straight lines, a straight line and a circle, or two circles, i.e.

PART NO. | TAPE NO. | DATE | BY | MACH. SIZE BAR-SHAFT-CHUCKER-COMBI | J&L 2 AXIS BENDIX CONTROL

PART NAME | OPERATION NO. | SPEED RANGE | TAPE PUNCH DATA | SHEET 1 OF

OPER.	BL'K NO.	SPDL CODE	DIAL CODE	MISCELLANEOUS CODES					PREP CODE	DEPARTURES TO ℄	DEPARTURES TO ℄	LOCATORS // TO X	LOCATORS // TO Z	FEED CODE	REF. PT. LOCATION		ZERO: START:	RPM / SFM	IPM / IPR	DIAL DIR
	N---	S--	T--	M--	M--	M--	M--	M--	G--	X±--.----	Z±--.----	I--.----	K--.----	F--	X	Z	DESCRIPTION OF OPERATION			

Figure 1 Standard Coding Form for Use With Manual Programming

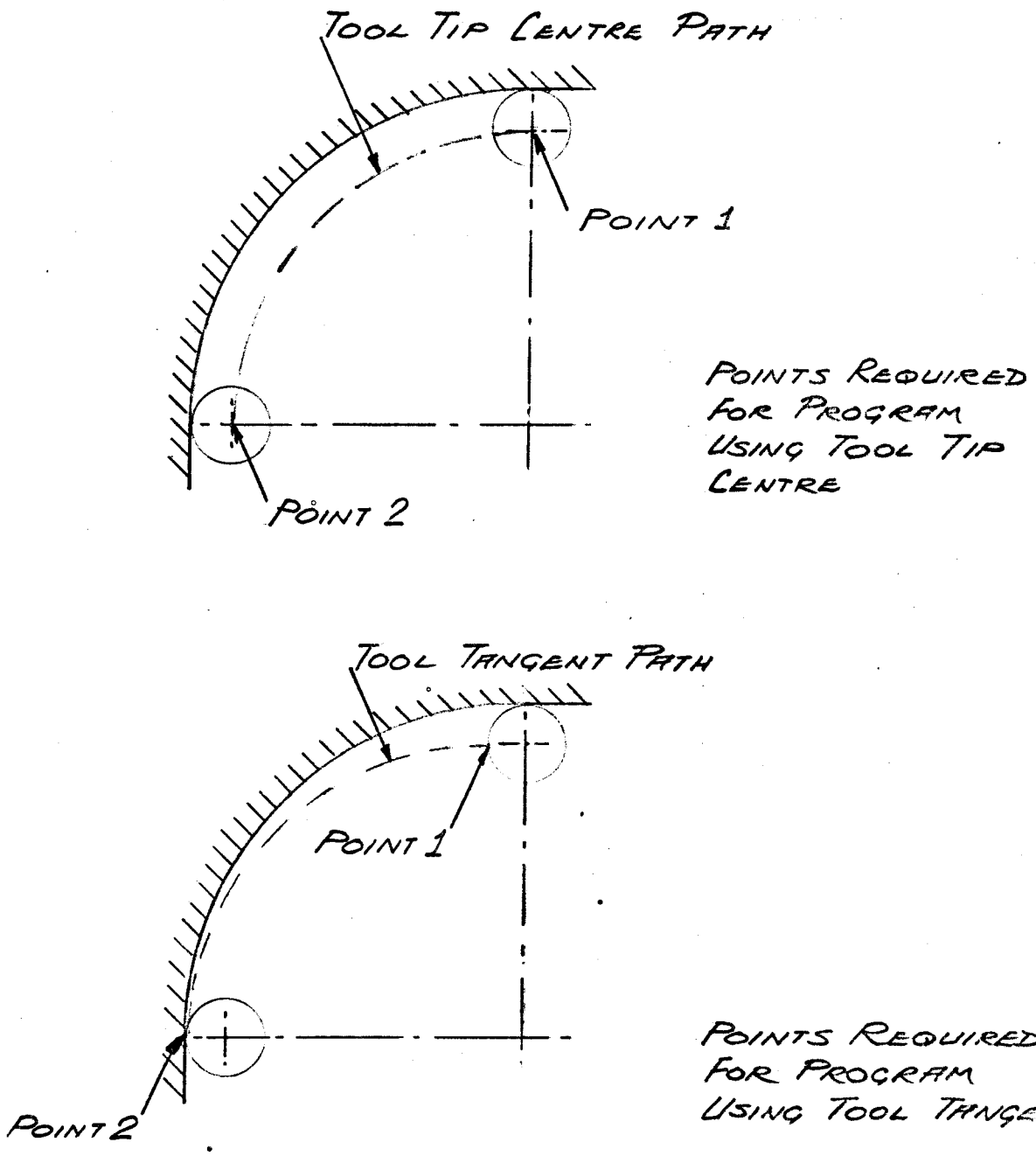


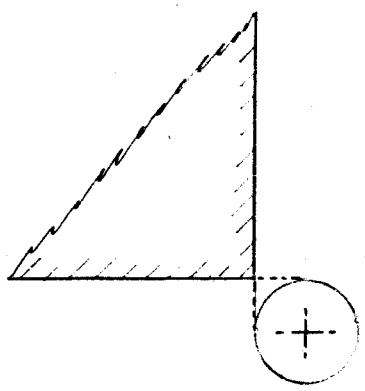
Figure 2 Example of Tangent Point and Tool Tip Centre Programming

it is the point at which the path direction changes. All these co-ordinates have to be determined by the programmer using information from the blueprint and some shop mathematics.

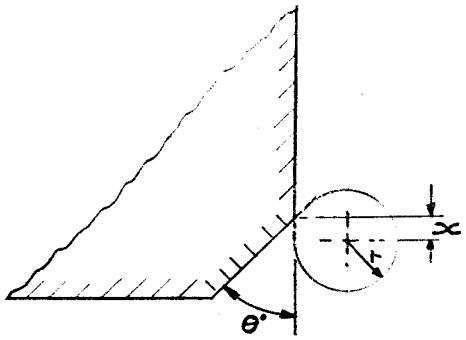
The main problem that occurs for the hand programmer is the determination of the turning point co-ordinates. Figure 3a shows how to locate the turning point if a 90° corner is required. This is a relatively simple case of adding and/or subtracting the tool tip radius to/from the appropriate X or Z axis value.

Figure 3b shows how to locate the turning point when say, a chamfer is required. The third case, Figure 3c, is more involved and shows the location of the turning point when two sloped lines intersect.

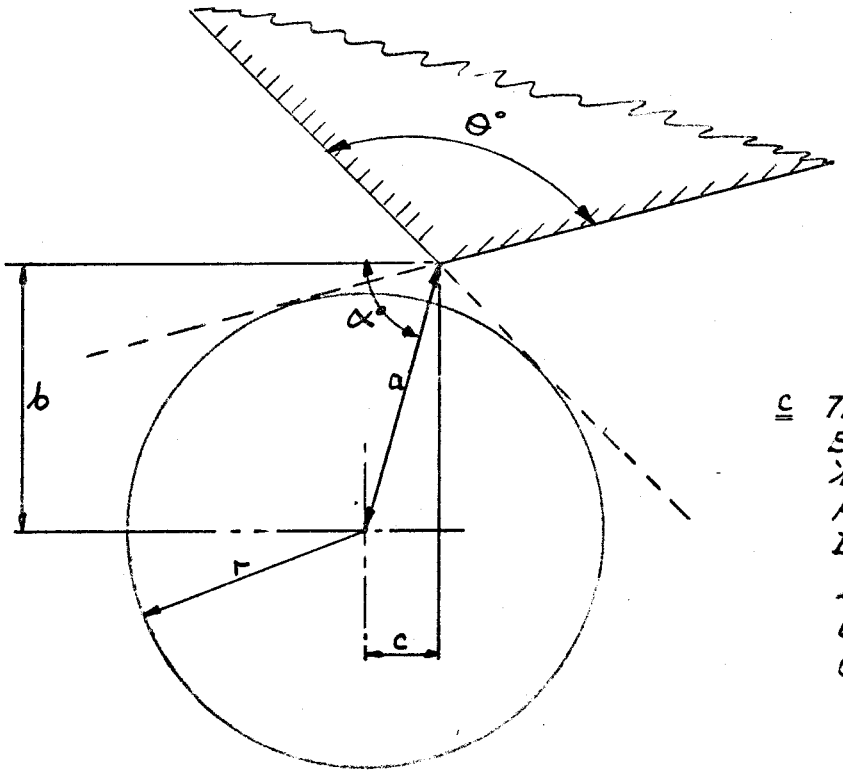
Figures 3a,b,c showed how to determine the co-ordinates of the turning point for straight line intersections. Figures 4a,b show the similar situation dealing with circular intersections. It is worthwhile noting here that when programming a circular arc two extra values are required. These are the 'i' and 'k' values which locate the centre of the arc, 'i' representing the distance from the arc start to the arc centre in the 'X' direction and 'k' the same distance in the 'Z' direction. This is shown in Figure 4c. For any condition a formula can be developed to aid the programmer, or, for that matter, a table can be drawn up covering most situations, thereby eliminating or certainly reducing the computation required.



a THE TOOL TIP CENTRE WILL BE DISPLACED FROM THE WORK BY THE TIP RADIUS



b THE TOOL TIP CENTRE WILL BE DISPLACED FROM THE RECTILINEAR LINE BY THE TIP RADIUS AND FROM THE INTERSECTION BY 'X' WHERE $X = r.\tan\theta/2$



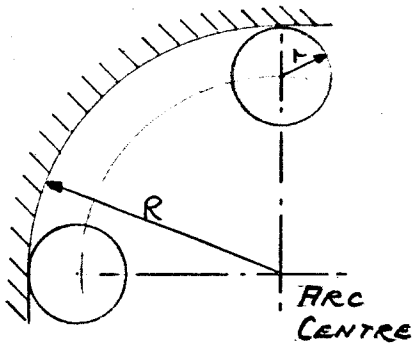
c THE TOOL TIP CENTRE WILL BE DISPLACED BY A DISTANCE 'b' IN THE 'X' DIRECTION AND BY 'C' IN THE 'Z' DIRECTION WHERE

$b = a.\sin\alpha$

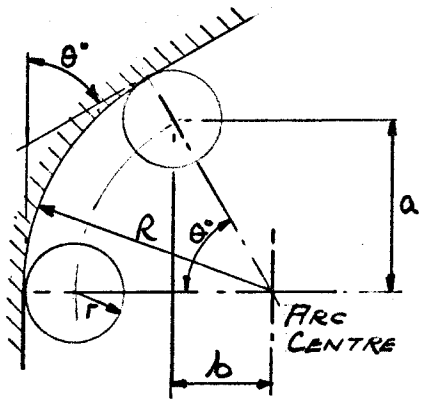
$c = a.\cos\alpha$

$a = r/\sin\theta/2$

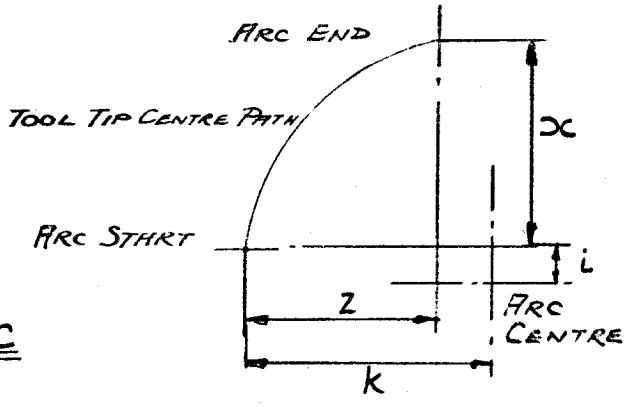
Figure 3 Turning Points for Linear Programming



a THE TOOL TIP CENTRE WILL BE ON CENTRELINE IN ONE DIRECTION AND DISPLACED BY TIP RADIUS IN THE OTHER



b THE TOOL TIP CENTRE WILL BE ON CENTRELINE IN ONE DIRECTION AND DISPLACED IN THE OTHER BY 'a' IN THE X DIRECTION AND 'b' IN THE 'Z' DIRECTION
$a = (R-r)\sin\theta$; $b = (R-r)\cos\theta$



c

Figure 4 Turning Points for Circular Programming

The programmer must have extensive knowledge of the manufacturing process being utilized. Also familiarity with the capabilities and functioning of the machine tool, together with the knowledge of tools available is essential. The programmer is also required to be conversant with metal cutting principles and practice, and with the methods planning function.

This reliance on the programmer for methods planning and cutting values is not conducive with optimization of the machining process. It is because of this and the fact that hand programming can be tedious and error prone that computer assisted programming has taken over in most cases.

3.0 ROLE OF THE COMPUTER IN NUMERICAL CONTROL

Some of the advantages claimed of Numerical Control equipment are greatly reduced lead times, rapid set-up times and superior accuracy and repeatability. The prerequisite for obtaining these benefits is a carefully prepared part program, but too much time spent in preparing the program can offset a number of these advantages however. Thus faster and more accurate ways are required for preparing part programs. The computer is naturally the answer.

The primary contributions of the computer are speed and accuracy in storing data, in making logical decisions, and in handling repetitive operations. The user has, of course, to be able to interact with the computer. To do this a program is written which defines in complete detail exactly what the computer is to do with the input.

Two major elements make up a modern numerical control computer system: the general processor and the post processor. The general processor is normally written in FORTRAN IV to allow for easy interchangeability with most computers. It performs a number of functions, in series, to produce information in a form acceptable to the post processor.

The general processor normally consists of four sections.[45] These are sections for control, input, arithmetic, and editing. The control section serves as the main source of control for the flow of information among the

other sections, plus it assigns the allowable space in the computer storage section.

The input translation provides the interface between the part programmer and the computer. It translates the part program statements into forms more readily handled by the computer. It normally contains diagnostic elements with which it discovers errors in the part program manuscript. If errors are found the processing will usually be stopped after this translation phase.

The arithmetic section transforms the computer coded part program into a series of coordinate points. Then using the motion statements, the shape of the cutting tool, the tolerances specified etc. it will compute the tool locations required to produce the component.

The last section of the general processor is the editing facility which performs auxiliary functions such as transformation of cutter locations from say the part coordinate system to the machine tool coordinate system. It also provides for the printing of the cutter location data which is fed to the post processor.

Thus the general processor converts the part program into a series of coordinate points that will give the required shape. This, however, is not acceptable for a machine tool control system and therefore must be translated into a suitable format. This translation produces blocks of incremental machine tool motion and the conversion unit is known as the post processor.

In general terms the post processor simulates the motions of the machine tool. It takes the output from the general processor and simulates the movement of the specified cutting tool about the part's geometrically defined surfaces. It calculates acceleration and deceleration rates on the basis of feed rate, length of line etc. The final output from the post processor includes all the relative machine command codes, feed rates, coordinate positions etc.

The problem with post processors is that in most cases a different post processor is required for each machine tool. Attempts have been made to standardize post processors but as yet very little has materialized.

Most numerical control languages today are written in FORTRAN IV rather than assembly languages. This is because FORTRAN IV is a common language and with minor alterations can be made to run on most computers. Also it does not demand a great deal of expertise on the part of the software specialist.

There are many systems available for utilizing computers for numerical control programming. The in-house computer is probably the ideal situation but this can present problems with turn around time. A number of companies now offer time sharing facilities for numerical control programming which provide quick service and almost instant access. With most time sharing systems the part program is processed one line at a time and responds with a diagnostic statement if there is

an error in input. This allows the programmer to correct immediately and continue with the input.

A third system that is used is one whereby the part programs are mailed to a processing company. This has a turn over time of around four days but eliminates the need for a terminal and is a reasonable low cost approach for a small company producing a low number of tapes.

The possibilities opened up by the closer association of computers and numerical control machine tools are enormous and there can be no doubt that the future of numerical control is bound up with the increasing use of computers in industry. For example the use of small general purpose computers to replace the control consules on individual machine tools. This is sometimes referred to as CNC (computer numerical control). DNC (Direct numerical control) is a rapidly advancing technique of using a computer to store programs for and control several machine tools.

The computer is obviously a great asset to the numerical control machine tool but one should never forget that it can only do as instructed. It is because of this that the development of part programming languages is towards a simplified input format, with emphasis on removing as many tasks as possible from the part programmer.

4.0 GROUP I PROGRAMMING LANGUAGES

4.1 DEFINITION

The computer program provides the cutter location data using as input data geometric curve definitions of the desired tool path.[8]

4.2 INTRODUCTION

Group I programs normally accept the definition of the curves to be machined and the paths the tool is to follow in an English-like word description. The input format is not rigidly fixed and geometric and path definitions, although generally distinct, can be interspersed.

The English-like nature of the language makes it rather easy to learn and convenient to apply. The more advanced, flexible and English-like is the input description, the more complicated is the computer program which converts the description into detailed commands that direct the machine tool. Thus group I programming languages normally require a large computer to carry out the necessary processing.

4.3 APT[16,29,44,45]

4.3.1 INTRODUCTION

APT (Automatically Programmed Tools) was the first computer assisted Numerical Control programming language to become available to Numerical Control users. The first version, known as APT I, was produced at M.I.T. in 1955. Then, in liaison with the Aerospace Industries Association of America, a second version was developed in 1957 and became known as APT II.

Rapid developments and refinements followed and the present day version, known as APT III, was released in 1961. At this time the Aerospace Industries Association established the APT Long Range Program to continue the development and maintenance of APT by the Illinois Institute of Technology Research Institute and an APT IV version is being developed.[16] This will include some new modules such as a conversational module and a technological module.

APT is basically a free format input language, with APT words assembled in arrangements that suggest English sentence structure, with its subject and predicate. For example each statement consists of a major word (subject), followed by a series of minor words (predicate).

The APT language is universal in that it is applicable to all types of machining operations from simple point-to-point drilling to multi-

axis contour milling. As with most languages a post processor is required for each different type of machine tool.

The APT language contains about 300 words and it is interesting to note that the APT III program respresents over one hundred man-years of development and testing.[44]

4.3.2 PART PROGRAMMING

The basic make up of any part program written in the APT language is two fold. First the geometry of the part is described by defining all the necessary points, lines, surfaces, etc., then the required cutter path motion is described by means of statements containing motion words such as GO TO, GO BACK etc.

The description of the geometry and the tool path make up the main body of the input program. Naturally the program will require identification of the part, machine, post processor, as well as details of speed, feed and cutter diameter.

Geometry statements normally take the form of:

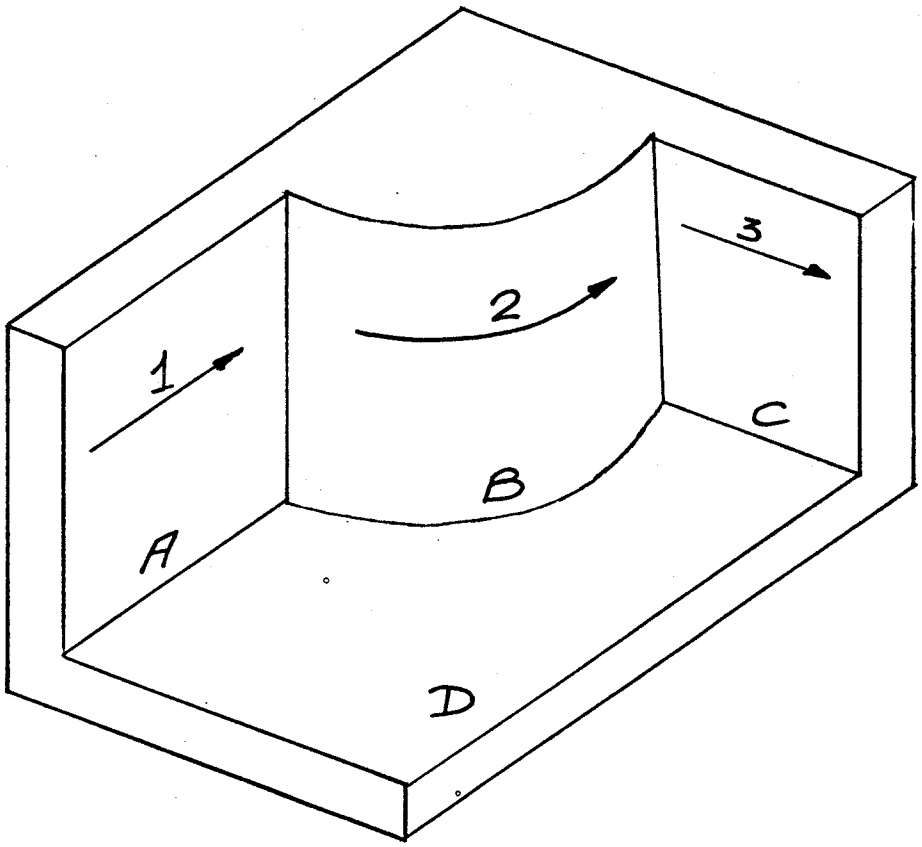
Symbol = Surface type/description data. The symbol represents the appropriate code assigned to the element of geometry, the surface type identifies the type of geometry, and the description data is a set of words, numbers or other previously defined symbols which completely and uniquely describe the desired surface. For example: -

L1 = LINE/PT1, PT2 would define L1 as being a line passing through

two previously defined points, PTl and PT2. Numerous different ways are permissible for defining the various geometric elements. For instance there are thirteen ways to describe a line, twelve to describe a point and eleven to describe a circle. A complete list of all definitions is given in the APT part programming dictionary available from all computer companies who offer APT facilities. This variety of description is very useful for the part programmer and also allows the dimensioning of a blueprint to be flexible.

In order to write motion statements two concepts must be understood by the part programmer. The first is that the tool moves along a pair of intersecting surfaces and the second is that the tool does all the moving i.e. the part remains stationary. In turning both of these concepts present little problem because one of the required intersecting surfaces is always a plane passing through the centre of the workpiece parallel to the lathe bed and also the tool always moves. The two primary surfaces that control tool motion are called the drive surface and the part surface. Each tool motion continues until its motion is checked by a check surface. In most cases the check surface becomes the drive surface for the next motion.

The distinction between the drive surface and the part surface is that the drive surface changes with each tool motion whereas the part surface remains the same for a series of tool motions.[16] Figure 5 shows the concept. In turning the only surface that remains constant throughout a series of motions



MOTION	DRIVE SURFACE	CHECK SURFACE	PART SURFACE
1	A	B	D
2	B	C	D
3	C		D

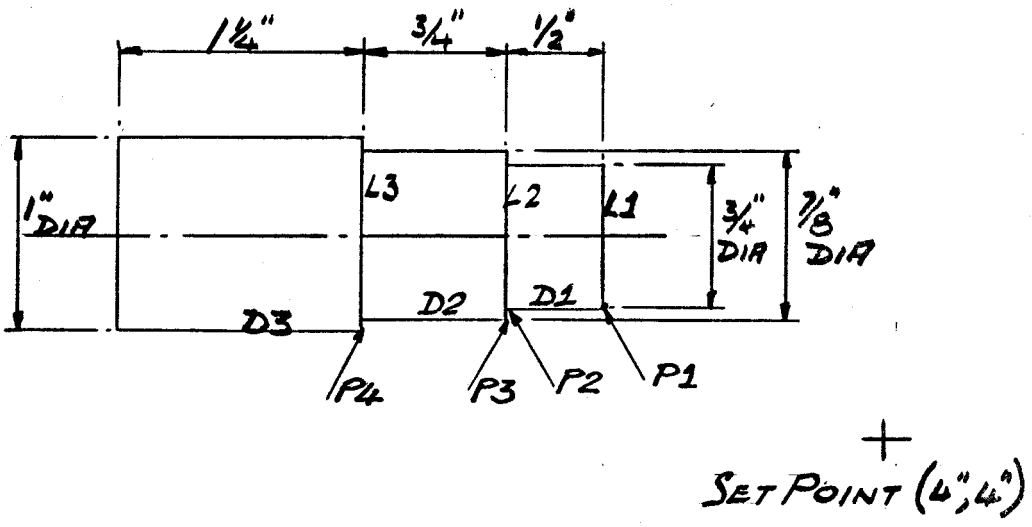
Figure 5 Relationship of Part, Drive and Check Surfaces in APT

is the one mentioned previously, that is a plane passing through the centre of the workpiece parallel to the lathe bed. This plane remains constant for all turning components and, although it must be stated in each program, it in effect makes the APT system two dimensional.

Tool motions adhere to a relatively strict convention geared to the previous direction of motion. This is the reason why the part programmer must always consider the tool to be moving and, in fact, drive the tool around the workpiece perimeter. Figure 6 shows a typical APT program.

APT contains provisions for repetitive programming. It provides three techniques to eliminate the tedious monotony of repeating the same commands over and over again. These techniques are looping, macroes, and copy. The looping and copy features appear to have very limited use in turning, whereas the macro feature can be extremely useful.

A macro is a single statement which refers to a group of part programming statements. It is analogous to a subroutine in a computer language. The macro is written in general terms, i.e. no dimensions, and describes either the geometry of the component or the required tool motions. It is then stored and called into service when required by use of the major word CALL and the identifying symbol of the macro.




```
REMARK DEFINE GEOMETRY
SETPT = POINT/0,0,0
    P1 = POINT/-1.5, 3.625, 0
    P2 = POINT/-2.0, 3.625, 0
    P3 = POINT/-2.0, 3.5625, 0
    P4 = POINT/-2.75, 3.5, 0
    D1 = LINE/P1, P2
    D2 = LINE/P3, PARLEL, D1
    D3 = LINE/P4, PARLEL, D1
    L1 = LINE/P1, PERPTO, D1
    L2 = LINE/P2, P3
    L3 = LINE/P4, PERPTO, D1
    S1 = PLANE/P1, P2, P3

REMARK DEFINE REQUIRED MOTION
CUTTER/0.04
FROM/SETPT
GO/TO,D2,TO,S1,TO,L1
GOLFT/D2,TO,L3
GOLFT/L3,PAST,D3
GOLFT/D3,PAST,L1
GOLFT/L1,TO,D1
GOLFT/D1,TO,L2
GOLFT/L2,PAST,D2
GOTO/SETPT
FINI
```

Figure 6 A Typical APT Program

4.3.3 COMMENTS

APT is probably the most powerful and widely used language in Numerical Control to date. It is available for use with a wide variety of computer systems and an extremely large number of post processors are already in existance. It also provides the base for the majority of other numerical control languages.

One of the problems associated with APT is the fact that it requires a very large computer, one with 256K storage. It is however offered on a number of time sharing systems. Its input is relatively simple but rather lengthy and requires a reasonable working knowledge of trigonometry.

It is not a language that is widely used for programming lathes because of its lengthy input and vast complexity. Using APT for programming lathes has been said to be rather like duck hunting with a cannon.

An example of this complexity can be shown by considering the geometry definitions. A large number of lathe components can be described using diameters and lengths. In APT these dimensions would have to be converted to distances from a set point, then various points defined and finally the required lines passing through these points defined. It is far simpler to be able to take dimensions off a blueprint and program these as input geometry.

APT, of course, has complete flexibility with regard to machine tools. It can be used to program the simplest of jobs on a lathe or a

complicated aerospace component on a five axis milling machine. This is an advantage in that a company with a variety of numerically controlled machine tools requires only one programming language.

4.4 ADAPT/REMAPT[29,42,43]

In order to reduce the size of computer required when using APT, adaptations of APT were developed. ADAPT and REMAPT are two such adaptations developed by IBM and GE respectively. These languages maintain exactly the same part programming language as that used with APT, but require a much smaller computer (32K) and are limited to three axis continuous path programming. Some of the flexibility of APT has been lost in that the number of geometric definitions have been reduced. For example the version of ADAPT available from Honeywell has only nine ways to define a line compared to the thirteen available in APT.[42]

Part programming is exactly the same as for APT and APT post processors can be used with minor modifications.

The ability to use the Macro, copy and looping functions is still available with both languages. ADAPT is offered by IBM, Univac and Honeywell for use with their computer systems, while REMAPT is only available on the Mark II General Electric time sharing service.

5.0 GROUP 11 LANGUAGES

5.1 DEFINITION

The Computer provides the cutter location data using as input the geometric definitions of the raw and finished part. Generally the computer decides about the subdivision of cuts and about the shape of successive passes.[8]

5.2 INTRODUCTION

During the 1960's numerical control equipment and computer manufacturers realized the tremendous lack of programming languages suitable for use with small computers. Also, it was noted that programming in a language such as APT required a great deal of time and a lengthy input. It was to meet these deficiencies that programming languages such as Compact II, Cinturn, and Chips were developed.

The APT programming language is a very comprehensive one, but for a great number of conditions far too detailed. Using APT as the base in most cases, programming languages were developed using a canned cycle approach. Canned cycles are special computer routines designed to generate whole series of operations from a single command. The use of these obviously reduce the input required and also allow the computer rather than the part programmer to decide on how the operations should be carried out. The part programmer is still required to input relevant cutting data such as cutting speed and feed but is no longer required to

describe the tool path.

The extremely short and simplified input of group II languages opens up numerical control machinery for use in such areas as job shop manufacturing i.e., low quantity, multi-component shops.

5.3 COMPACT II[29,46]

5.3.1 INTRODUCTION

COMPACT II is a computer time-sharing numerical control part programming system, and was developed by Manufacturing Data Systems, Incorporated, (MDSI), Ann Arbor, Michigan.[29] It is a common language applicable to all numerical control machine tools and presently has 2, 3, and 4 axes positioning, 2, 3, and 4 axes linear contouring and 2 1/2 axes circular contouring capabilities. A post processor is required, because it is a common language, for each type of machine tool using the system but is supplied by MDSI free of charge.

COMPACT II has basically a free format input, with each statement consisting of one major word and any number of minor words. It contains a number of 'canned cycles' or macros, covering such areas as threading, turning, and taper turning.

COMPACT II is a 'single-pass' processor which causes each statement to be completely processed, including generation of machine tool code, before the computer proceeds to the next statement. It is also equipped with an editing feature that allows the user to correct any programming errors as they occur during the source program's single pass through the computer. It will locate and define the error and suspend processing until the error is corrected.

Once an error free input is obtained, the program is processed by COMPACT 11 and the post-processor and a punched tape obtained.

5.3.2 PART PROGRAMMING

The general input of the part program for COMPACT 11 can be split into four sections. These are:

a) general information,

b) geometrical definitions,

c) description of stock and part boundaries, and

d) machining statements.

The format for the general information is fixed and consists of four statements.[46] These are the:

MACHINE STATEMENT - identifies the machine tool link needed to execute the program.

IDENT STATEMENT - provides identification for both manuscript and the tape.

SET-UP STATEMENT - specifies the initial position of the cutter at the start of the cutting sequence, and

BASE STATEMENT - establishes the origin of the part relative to the co-ordinate system of the machine tool of the start of the program.

The data provided in the IDENT STATEMENT will actually be punched into the leader of tape as visually readable characters. This tends to eliminate the incorrect use of a tape.

The COMPACT 11 language permits the definition of several geometric entities which may be defined in a variety of ways. The geometric types allowable are:

DPT 'N' (define a point),

DLN 'N' (define a line), and

DCIR 'N' (define a circle).

DPT, DLN, and DCIR are major commands and are used in conjunction with the numerous methods available for defining a point, line or circle. The ultimate use of these definitions is in machining operations. Once a geometric entity has been defined it may be used in defining others, or in a machining statement by programming the minor command for the geometric form and its identifying number.

Some geometric definitions may require selectors in order to eliminate any ambiguity. Such selectors are related to the axis co-ordinates of the machine tool and define the next X or Z value location in relation to the previous value.

The stock and part boundaries are then defined. This is done very simply by tracing around the required contour using the previously defined geometric statements. The contour is always defined from right to left i.e. towards the chuck, and each contour change must be stated in the correct sequence.

The machining statements allow the programmer to describe the tooling and the sequence of cuts required to produce the component. When calling up the appropriate tool various parameters are usually defined. These are the tool radius, the cutting speed and feed, the maximum depth of cut, and the amount of stock to be left on for finishing if required.

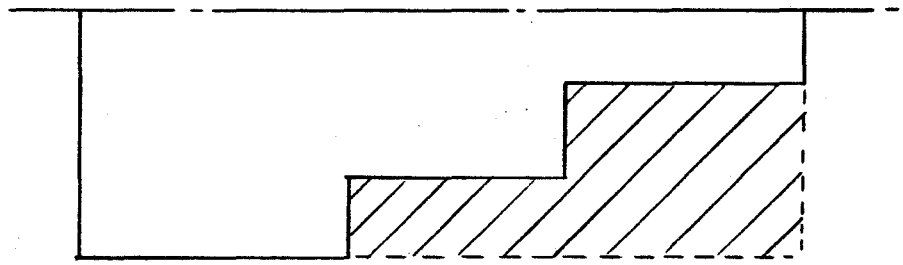
In COMPACT 11 there are two avenues of approach open to the part programmer when defining the machining sequence. The sequence of cuts can either be a series of statements describing every single pass of the tool or use can be made of canned cycle routines.

The canned cycle routines available on COMPACT 11 are facing, turning-external and internal, taper turning-external and internal, and threading-again external and internal. There is also a fast turn cycle that has recently been added.

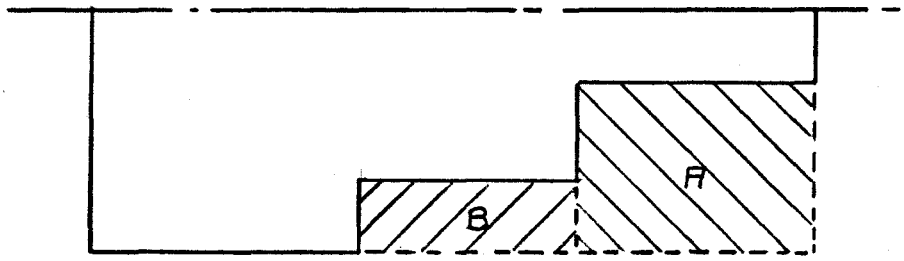
The difference between the fast turn cycle and the normal turn cycle is that fast turn will cut a series of interconnected diameters, whilst the turn cycle will only cut one diameter with each command. Figure 7 shows the difference diagrammatically.

In order to simplify the input COMPACT 11 provides for repetitive programming by use of a STORE command. Using this command it is possible to store previously defined commands and recall them for use as many times as required at a later stage in the program.

A typical COMPACT 11 part program is shown in Figure 8.

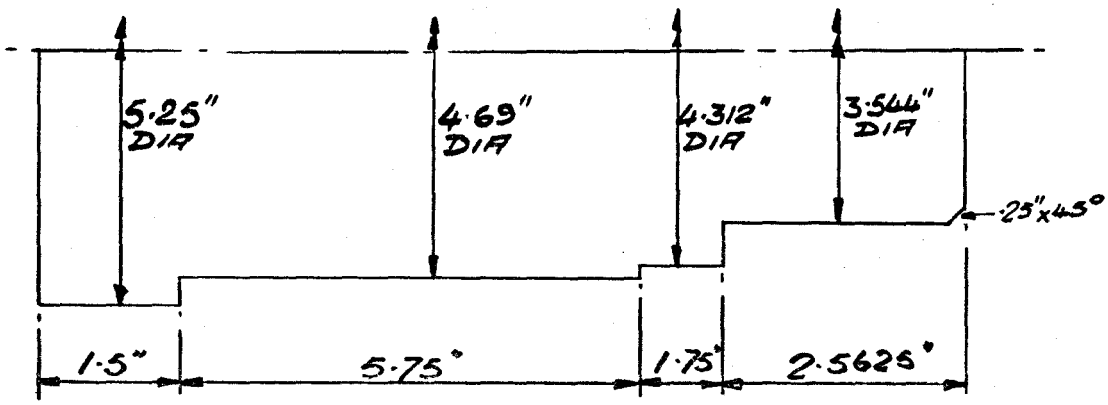


FAST TURN APPROACH. ONE MACHINING COMMAND REQUIRED FOR REMOVAL OF SHADED AREA.



CANNED CYCLE APPROACH. TWO MACHINING COMMANDS REQUIRED, ONE FOR BLOCK 'A' AND ONE FOR BLOCK 'B'

Figure 7 Difference Between the Fast Turn Cycle and the Normal Turn Cycle in COMPACT II




```
MACHIN,CINCITCA4
IDENT,TAPE 2-483 1ST OP.
SETUP, 4X,14.5Z,UNIV,RANGE2,NOG,LNDLT
BASE,XB,ZB
DLN1,14.475ZB
DLN2,14.375ZB
DLN3,3.544D
DLN4,PT(LN3,LN2/.25ZS),45CW
DLN5,LN2/2.562ZS
DLN6,4.312D
DLN7,LN5/1.75ZS
DLN8,4.69D
DLN9,LN7/5.75ZS
DLN10,5.25D
DCIR1,PT(LN5/.093ZL,LN3/.093XL),.093R
DCIR2,PT(LN7/.125ZL,LN6/.125XL),.125R
DCIR3,PT(LN9/.125ZL,LN8/.125XL),.125R
DMB1,LN2;LN10,NOMORE
DPB1,LN4;LN3;CIR1,F180;LN5;LN6;CIR2,F180;LN7;LN8;CIR3,F180;LN9,NOMORE
ATCHG,TOOL4,.031TLR,250FRM,.022IPR
MOVEC,OFFLN1/ZL,OFFLN10/XL
TURNL1,.03STK,.3SDPTH
ATCHG,TOOL6,.031TLR,0STK,350FPM,.008IPR
MOVEC,OFFLN4/XL,OFFLN2/.02ZL
CUT,PARLN4,OFFLN3/XL
CUT,-.2Z,STORE1
MOVE,.3Z,OSTOP,STORE2
FILET90,LN3,LN5,.093R,90CW
CORNR90,LN5,LN6,.02R,CCW
XSTORP1/2
FILET90,LN6,LN7,.125R,90CW
CORNR90,LN7,LN8,.02R,CCW
FILET90,LN8,LN9,.125R,90CW,.015IPR
CORNR90,LN9,LN10,.02R,CCW
END
```

Figure 8 A Typical COMPACT II Program

5.3.3 COMMENTS

COMPACT 11's main asset is that it is available as a time sharing system. This allows for immediate computer accessibility and eliminates the turn-around time normally associated with "in-house" computers or mail service. The special editing subsystem called QED provides an extremely rapid method of debugging the input program.

An interesting routine available called PLOT provides the user of turning machines with a graphic display of the part program and a debugging capability. The plotting originates with the part program and not the CL file. This provides the part programmer with a picture of the geometry prior to the development of the tool path. Editing is immediately available, thus eliminating dry runs of the tape at the machine. The PLOT routing can be displayed using either a plotter or a cathode ray tube.

COMPACT 11 is an easy to use, comprehensive programming language. It is suitable for turning, milling, drilling and boring machines and is not restricted to any one manufacturer's machine tools.

5.4 SPLIT and ACTION 11

SPLIT and ACTION 11 are very similar in format to COMPACT 11 and thus part programming in these languages will not be covered in any detail. In order to appreciate these languages, however, a general description will be included.

5.4.1 SPLIT[7,26,29]

The SPLIT system (Sundstrand Processing Language Internally Translated) is a multi-axis (two to five) positioning programming system with some contouring capabilities.[29] It was developed in 1960 by the Sundstrand Machine Tool Company as a means to communicate with the IBM 360 that Sundstrand had access to. SPLIT became the first program for small computers that could be used to produce numerical control tapes. Next to APT it is the oldest numerical control programming language in existance today.

SPLIT is a proprietary program oriented towards Sundstrand's numerical control machine tools, but it is available for some other numerical control machines. It is written in Fortran IV and will execute on a number of computer configurations. A full-blown SPLIT processor with circular interpolation requires 65K bytes of core, with less core required as features are deleted. SPLIT is a single pass processor and each line is processed completely as it is translated by the computer.

The language is based on common English words which are used to describe the operations to be performed by the machine tool. Part programming in SPLIT is very similar to COMPACT 11, in that each statement consists of a major word followed by a series of minor words. SPLIT includes some canned cycles but these are not suitable for turning machines.

A unique feature of this language is that no post processor is required. A SPLIT processor is individually tailored to each machine and the tape output is actually a result of incorporating the post processing function in the SPLIT processor.

SPLIT is not offered on a time sharing basis and, as previously stated, is primarily for use only with Sundstrand's numerical control machine tools.

5.4.2 ACTION 11[7,29,30]

ACTION 11 was introduced in 1966 for the IBM 360 computer by Numerical Control and Computer Service, Cleveland, Ohio.[29] Recently it has been made available for other computers. Its forerunner ACTION 1 was introduced in 1964 but was limited to defining points only. The capability of defining lines and circles together with their intersections and tangencies was added to get ACTION 11.[30] It will perform limited two axis contouring and point to point operations for milling, drilling, boring, and turning.

In 1969, a new lathe routine was added called AUTO-ACTION. This was really the introduction of canned cycles for rough and finish turning, both external and internal. An AUTO-THREAD routine is now also available for external or internal threading operations.

As with SPLIT, ACTION 11 has no post processor, it incorporates the post processing function directly in the processor. It lends itself to the use of a smaller computer requiring 32K bytes of core.

Time sharing facilities are available but are only accessible via mail service. This, of course, causes a delay in turnaround time but provision has been made to handle rush programs.

Again, as with COMPACT II and SPLIT, the programming language is made up of major and minor words, many of which have exactly the same meaning in all three languages. ACTION II is not limited to any one make of machine and is suitable for programming various machine tools.

5.5 CINTURN[12]

5.5.1 INTRODUCTION

CINTURN, or the CINTURN MACRO system as it sometimes is known, was developed by Cincinnati Milacron's numerical control programming group. It is a collection of APT or ADAPT language subroutines developed for turning and boring. The main objective in the development of the system was to simplify programming by writing and debugging an APT or ADAPT language subroutine for every possible part, feature and machine movement.[12] The rationale was that virtually any turned or bored profile could then be programmed by using all of the required subroutines. The only variables would be dimensional and these would be entered by the programmer.

Some of the macros used in CINTURN were already used in other programming languages. For instance a threading macro is available in both COMPACT II and ACTION II, but there was no macro available

that would, for example, guide a tool through a straight line, into a radius, up a shoulder, into a chamfer, and then into an angle. Such a macro was devised for use with CINTURN and is known as the TFC macro.

5.5.2 PART PROGRAMMING

Programming a lathe machining job by uses of the CINTURN MACROS is a four part procedure. It consists of first stating the dimensions of the finished component and a general start statement setting forth the blank size and maximum depth of cut. Secondly, the part is described by use of the geometry macros, then a routine is called to rough out the geometry and fourthly, a finishing routine is called to bring the part to the desired size.

Essentially this means that there are two types of macros, those that describe part features and those that control machine movements. A typical geometry macro is the GT macro.[12] The G signifies that it is a geometry macro and the T represents a tapered shaft. The macro will be called out be the statement:

CALL/GT, NT, BD, A, L

The CALL/command is equivalent to the major word type of command used in a language such as COMPACT 11 and precedes all statements called for a macro routine. The parameters listed after GT are a series of variables that have to be defined when calling up this macro.

The NT refers to the corner number at which the taper is to begin, A is a decimal value assigned to one-half of the included angle of taper, L is the length of taper and BD is the original bar diameter. There is also a second group of parameters associated with this macro. These indicate certain machining variables such as amount of finishing stock to be left, cutting speed and chip thickness. Values assigned to these variables are average values determined by Cincinnati Milacron, but can be readily changed by the part programmer simply by entering a new value when the appropriate geometry macro is called up.

The geometry macro GT is shown in Figure 9. Also shown are the movements of the roughing and finishing tools in order to produce the required form. A machining macro would be called which would move the tool from a predetermined set point into a blocking routine. This blocking routine would rough out the required diameters one at a time, leaving the correct amount of finishing stock on each. A second machining command would then be called out in order to finish the diameters to size, followed by a third machining macro used for finishing the taper.

An example of a CINTURN program is shown in Figure 10.

5.5.3 COMMENTS

Most languages use English-like terms for program input in order to avoid confusion. CINTURN is unable to use this approach because of the fact that it is based on APT. The macros used are written in APT

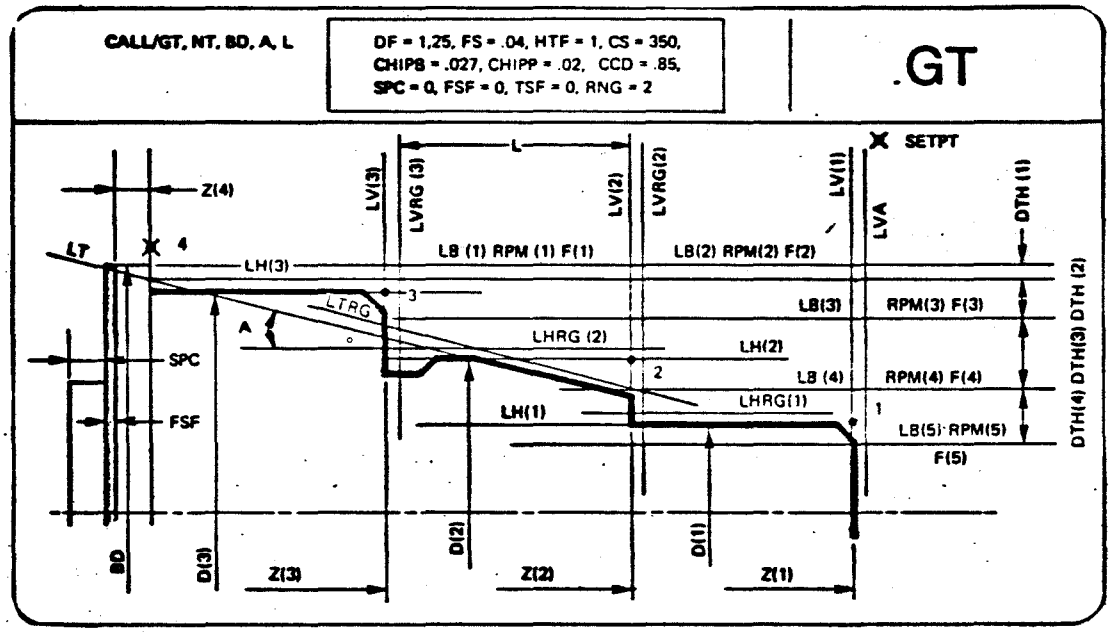
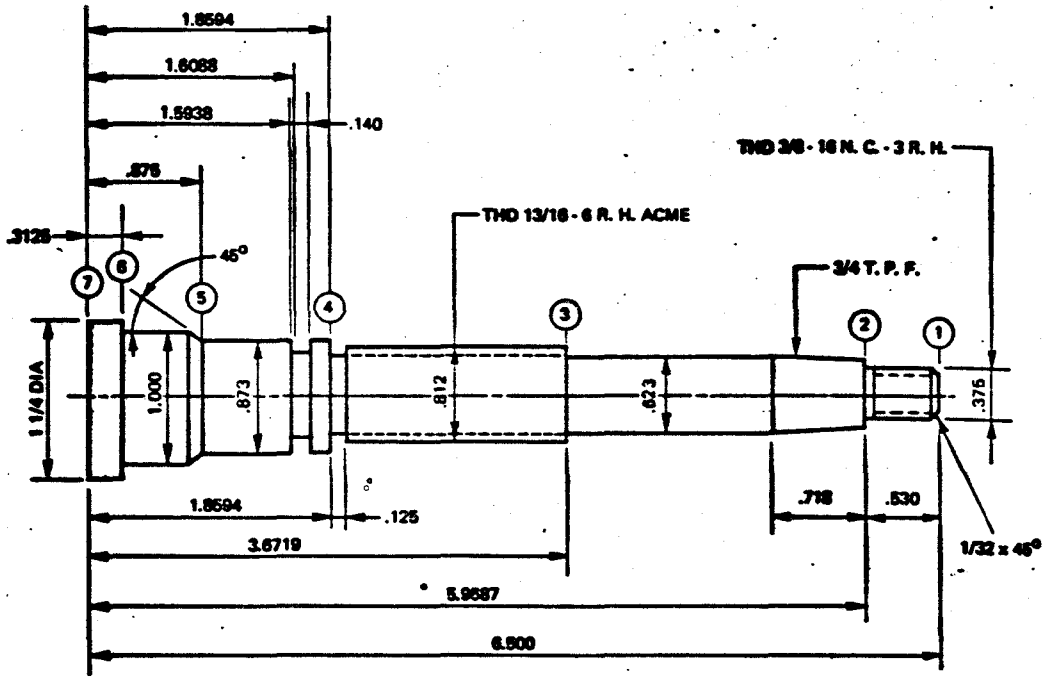


Figure 9 The GT Macro as Used in CINTURN




```
PARTNC  SAMPLE NO. 1
REMARK OD OPERATION FROM BAR STOCK
Z(1) = 6.5000
Z(2) = 5.9687
Z(3) = 3.6719
Z(4) = 1.8594
Z(5) =  .8750
Z(6) =  .3125
Z(7) = 0
D(1) =  .375
D(2) =  .623
D(3) =  .812
D(4) =  .873
D(5) = 1.000
D(6) = 1.250
D(7) = 1.250
CALL/ST, NC = 7, BD = 1.250, DPTH = .200
C(1) = 1/32
CALL/GT, NT = 2, A = 1.83, L = .718, BD = 1.25, CS = 200
CALL/TR
CALL/CHS
CALL/THC, N = 2
CALL/CHH, N = 4, NC = 2
CALL/ANG, N = 5
CALL/CHH, N = 7, NC = 2
CALL/NI, N = 4, NP = 7, Z = 1.5938, DN = .696
CALL/NNI, N = 4, NP = 4, Z = 1.6088, DN = .696
CALL/NNI, N = 3, NP = 4, Z = 1.8594, DN = .623, COR = 1
CALL/THRD, N = 3, NP = 3, L = 1.8125, P = 6, RPMT = 425, NT = 1, AA = 12.5, DTF = .5

CALL/THRD, N = 1, NP = 3, L = .530, P = 16, RPMT = 825, TLF = 7, TLF = 8
CALL/FIN
REWIND/1
FINI
```

Figure 10 A Typical CINTURN Program

and thus any designation used to call a macro must not have any meaning in the APT language. The macros available are for common features only and thus any specialized profile feature requires a new macro to be written. This can be done by Cincinnati Milacron's numerical control programming group or by the individual part programmer. It requires, of course, a working knowledge of the APT language.

CINTURN is only available for use with numerical control lathes and is applicable for most manufacturer's lathes. The system could readily be adapted to other machine tools such as milling and drilling, but as yet nothing is available. The system base, APT, is of course available for these machining operations.

Post processors are required but with the very large number of APT post processors on the market at the present time, this may not present a great problem.

CINTURN can be used on a time-sharing basis as a number of companies already offer the APT system, thus requiring only the addition of the macros.

5.6 CHIPS[22]

5.6.1 INTRODUCTION

CHIPS is a numerical control programming language written by LeBlond for use with the LeBlond Tape-Turn Lathes.[22] CHIPS generates a C.L. file thus requiring a post processor and, therefore, it could be

possible to utilize the CHIPS language with other makes of lathes.

Input data is defined using standard CHIPS data sheets, thus utilizing a fixed field, fixed column format. It is a simplified input with the part programmer defining the rough and finished contours, from right to left, using either absolute or incremental dimensioning. The required machining operations are then defined together with the appropriate tool and tool position. The CHIPS processor will then determine the co-ordinates for the complete tool path including turret indexing if required. Rapid traverse and clearance moves are generated automatically.

Actually two types of machining statements are allowed. The first is the basic machining operation which is the canned-cycle type of operation. Canned cycles are available for internal and external turning, threading, grooving and facing. The second type of machining instruction is known as the elemental motion sequence. This allows the part programmer to generate a unique tool path sequence, one that a canned cycle approach is not suitable for. The programmer directs the tool path motion one move at a time and also controls the clearances, rapid traverse, etc. as they are not automatically determined as with the canned cycle.

5.6.2 PART PROGRAMMING

Input data is declared as belonging to one of the following basic classifications:

a) basic data definitions,

b) part/stock contour definitions,

c) machining operation definitions, and

d) elemental motion sequence definitions.

The input format is relatively free in regard to the order of definitions.

Basic data definitions contain the relevant information with regard to set-up, tooling, threading data, and point location. The set-up instructions are keyed by the word SET-UP and define the initial position of the turret, indentify the machine, state the maximum length of stock plus clearance etc. Tooling data contains the parameters unique to a particular tool in a particular cutting position, whilst the threading data contains the parameters for the closing of a particular thread. The point location is required when using the elemental motion sequence definition. It is used to identify a reference point from which the elemental motion sequence may be started.

The rough and finished contours are defined from right to left, describing each surface element in turn. Each consecutive surface element must be of larger diameter for external or of smaller diameter for internal, than the previous element.

STKOD AND STKID are the commands used for defining the stock boundaries whilst DEFOD AND DEFID are used for the finished contours.

The machining operations make use of the canned cycle routines available and use a command word plus details of the required speed, feed, etc. Canned cycles available allow for roughing and finishing both internal and external contours, drilling, threading internal and external, facing and grooving. With the grooving canned cycles both internal and external plunge grooving operations can be performed as well as a plunge grooving operation into the part face.

Elemental motion sequence is used at the discretion of the part programmer and is used for unique tool path sequences. Here the programmer has to guide the tool into starting position, then through the correct motion sequence and then back to correct position for the next operation.

As previously stated the input order of definitions is relatively unimportant and thus elemental motion sequence statements can be interspersed between canned cycle routines.

5.6.3 COMMENTS

The input program is a relatively short one when compared to APT and is compatable to other similar languages such as COMPACT 11. The use of the standard data sheet is both helpful and necessary. It helps the part programmer to ensure that all the required information

is given and is necessary because of the fixed column format required by the processor. Actually this fixed field, fixed column format eliminates the need for a large number of command words and thus tends to shorten the input.

CHIPS is available for use on an IBM 360 computer and LeBlond also offer a processing service, on a postal basis, for users who have no access to a computer. It is not suitable for machine tools other than lathes and is basically only readily usable for LeBlond numerical control lathes.

6.0 GROUP III PROGRAMMING LANGUAGES

6.1 DEFINITION

The computer integrates into the cutter location programs all technological command data about speeds, feeds, canned cycles, etc. The input data consists of the geometrical definition of the initial and final profile, material used, and the machine required, but the computer has been previously provided with libraries concerning material data, tooling available and machine characteristics.[8]

6.2 INTRODUCTION

The whole manufacturing process from product planning to the workpiece can be regarded as a series of steps in the preparation, handling, and processing of information. For the conventional manufacturing process this means that many repetitive, time-consuming work steps have to be executed by man. The developments tend to have data processing machines, such as numerical control machinery and computers, take over the various steps and further rationalize and automate the whole manufacturing process.

In view of the technical and economic changes that are taking place in respect of the numerically controlled manufacturing process, the use of the computer must not remain restricted to the simplification of the preparation of control tapes, but must simultaneously serve to optimize the machining process. Thus the programmability of the

technology of the machining task is of the utmost importance.

"Computer aided programming systems must provide possible solutions to the technological problems involved in addition to solving the geometrical part of a machining task". This statement was made by Dr. Wolfgang Budde[38] during the presentation of a paper to the Society of Manufacturing Engineers in 1972. This line of thought provides the foundation for programming languages such as EXAPT, GETURN & CUTS, all of which carry out such tasks as:

determination of the optimum cutting sequence,

determination of the required speeds and feeds.

Thus with increasing wages and decreasing computer costs it appears to be imperative that the computer be utilized more fully when dealing with machining tasks.

6.3 TECHNOLOGICAL PROCESSOR[1,2,11,35]

With the advent of Group III programming languages the technological processor has become a very important section of the language. It provides a means of taking the results of years of experimentation and experience and utilizing them for the calculation of suitable cutting conditions for different materials and components. It relieves the part programmer of having to remember or calculate suitable cutting speeds, feeds, etc.

It is extremely important that the calculated cutting conditions are realized in practice and hence the reason for continued investigation into cutting parameters.

In a machining operation many factors have to be considered. The cutting parameters, speed, feed and depth of cut not only have to be calculated but their interrelationship with each other and with other factors such as tool life, chatter etc. must be considered. These relationships are available as mathematical formulae, which readily allow the adoption of the computer to make technological descisions.

Essentially then the output required from this portion of the processor is a cutting speed, a feed and a depth of cut. The various factors which influence the choice of values for these cutting parameters are the forces encountered in the cutting process, the chip form and the tool life. These are influenced by the geometry and dimensions of the tool, the properties of the workpiece, the chip cross section and the machine

tool.[11]

When rough machining the main objective is usually to remove the maximum amount of material in the minimum time. To do this the cutting speed, feed or depth of cut is increased. An increase in one of these parameters, however, requires a decrease in another if the same tool life is needed. Cutting speed has the greatest influence on tool life, followed by feed and depth of cut respectively. Thus in order to preserve tool life and get maximum metal removal the feed and depth of cut are increased. Depth of cut, however, is restricted by more factors than the feed e.g. chatter restrictions, tool width, required diameter of workpiece etc. Thus it is usual to increase the feed to a maximum when roughing and keep the depth of cut to feed ratio a constant.

Another factor for choosing to maximize feed is that the specific cutting force decreases as the feed increases. The main cutting force is equal to the specific cutting force times the product of the depth of cut and feed. Thus, with the depth of cut to feed ratio held constant, the main cutting force will increase less than proportionatly to the chip cross-section. This of course means that with the maximum possible feed the volume of metal removed/unit of power will be a maximum.

In finish machining the governing criterion is usually the required surface finish. The tool profile is reproduced on the work surface in the form of feed marks. Equations have been derived[1] for the surface

finish depending upon the feed and the tool profile.

According to Chisholm[1], and Ansell and Taylor the surface finish decreases as the cutting speed increases, until the surface finish equals the ideal finish. Further increases in speed do not alter this surface finish. In general a continuous chip with no built up edge is required for a good surface finish and higher speeds help to eliminate the built up edge and provide steady cutting conditions.

Thus determining the cutting speed and feed for finishing operations presents a slightly different problem to calculating the same parameters for roughing.

For reasons outlined previously in rough machining the feed is normally maximized. In order to do this the maximum allowable cutting force must first be determined. Three factors affect this value, these are workpiece geometry, the machine tool and the cutting tool. The maximum allowable cutting force must be such that it does not violate:

i) the maximum force that can be withstood by the workpiece,

ii) the maximum force that can be generated by the maximum permissible torque of the machine,

iii) the maximum force that can be withstood by the tool.

In other words the maximum allowable cutting force is the minimum value of the above three forces.[11]

Once the maximum cutting force is known then it is used to calculate

the maximum feed. This is done by utilizing the known relationship between the maximum allowable cutting force, the specific cutting force, the feed and the depth of cut.

If a constant depth of cut to feed ratio is used then it becomes a simple matter to calculate the depth of cut given the feed. Depth of cut, however, has two restricting factors. These are chatter and the length of cutting edge of the appropriate tool.

The length of cutting edge is obtained from the tool file whilst a mathematical model can be used to determine if the calculated depth of cut will produce chatter. If this is the case the depth of cut is modified to a suitable value and the feed recalculated.

The final parameter, cutting speed, is controlled by power available and/or tool life. From the point of view of power available it is a relatively simple matter to calculate the maximum allowable cutting speed, utilizing the already determined maximum cutting force. But the cutting speed value must satisfy the requirements of tool life and therefore a value can be calculated using the previously determined feed and depth of cut with some adaptation of the Taylor equation for tool life. The cutting speed chosen will obviously be the smaller of the two calculated.

A typical algorithm for the automatic determination of cutting conditions is shown in Figure 11. This was produced by Takeyama[2] at

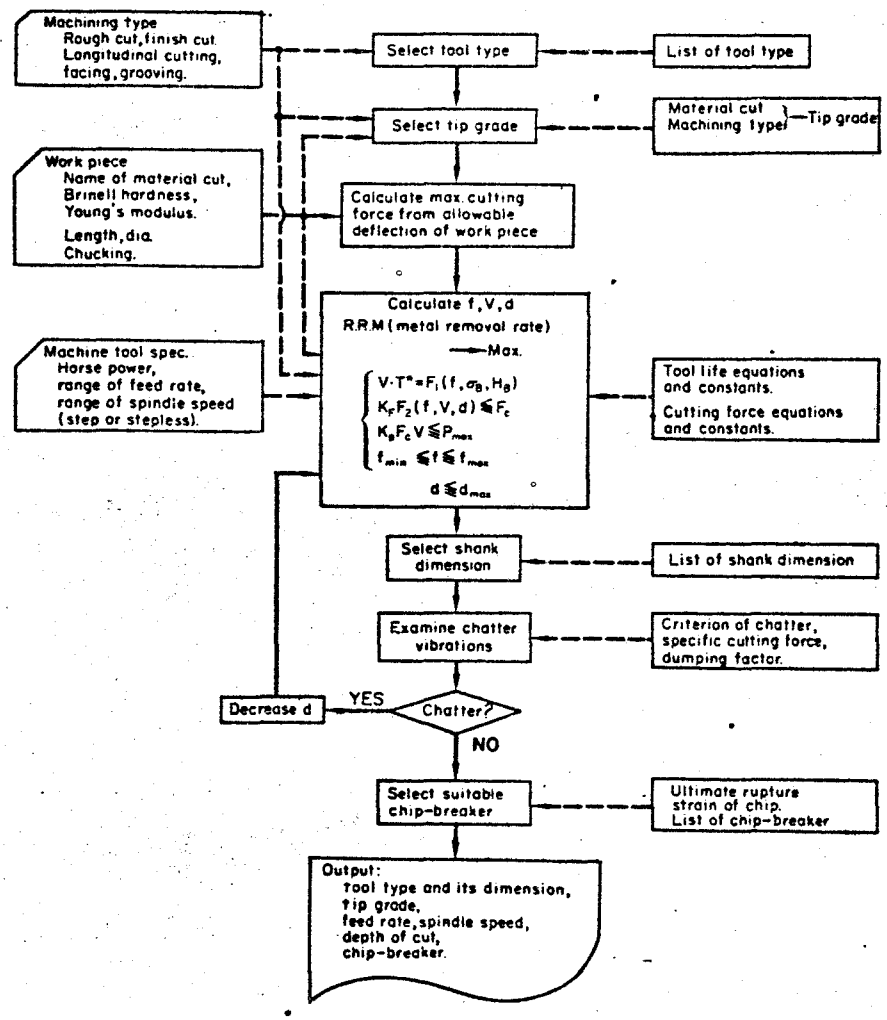


Figure 11 Algorithm for the Automatic Determination of Cutting Conditions

the Government Mechanical Laboratory in Tokyo and used data collected since 1956 on such aspects as tool life, surface roughness, accuracy, etc.

The input information consists of the type of operation, type of work material and its mechanical properties, dimensions of work material and type of work clamping, machine capacity, feed steps, and speed steps. These inputs are shown on the left side of the flow diagram. Actually if this system was incorporated into a numerical control programming language, the majority of these inputs would be in the data file. For instance the mechanical properties of the workpiece are included in the material file in GETURN, whilst the machine capacity, etc. is included in the machine data file.

The contents of the data file are shown on the right of the flow diagram and are the type of tool, the tool tip suited to each work material and type of operation, tool life equation for each material cut and operation, formulae used for cutting force and power, dimension list of tool shanks, mathematical model of chatter vibration, data of rupture strain of work material, and list of chip breakers.

Initially a suitable tool type and tip grade is chosen by referring to the input information and the stored file. Speed, feed and depth of cut are then determined from the tool life equation beginning with the maximum machine feed, for example, under the constraints of permissible work deflection, maximum machine power, permissible feed, etc. In the

course of this process the speed, feed, and depth of cut are also determined so that the metal removal rate may be maximum in a rough cut under the aforementioned conditions. The shank dimension of the tool is then determined.

A mathematical model for chatter is used to ascertain if chatter will occur under the parameters already obtained. If the parameters are such that chatter will occur then the depth of cut is reduced to such a level to eliminate the chatter. If the depth of cut reduction is larger than a specified value, say 5%, the speed and feed are reduced by means of a feed back loop in the algorithm.

Finally, the type of chip breaker suited to the determined conditions is chosen. Again, provision is made for a feed back loop if the feed parameter proves to be too small for the chip breakers.

The computer routine was developed in order to minimize the capacity of the computer to be used, and as stated previously a number of the mathematical models used were based on extensive data collected over a great number of years.

Tests carried out proved to be highly successful when comparing the results obtained through the algorithm with those obtained from actual studies.

6.4 GETURN/MITURN[8,10,11,14,15,39]

6.4.1 INTRODUCTION

The MITURN programming system (Metal Institute TURNing Program) was developed by the Certrum voor Metaalberwerking of the Metaalinstituut, TNO of the Netherlands. It was developed using the same philosophy as the AUTOPROG language. AUTOPROG was developed in Czechoslovakia under the direction of Doctor Kolog during the period 1966-1969[8]. The philosophy was that approximately eight programs per day were required to keep a numerically controlled lathe running on a two shift basis. Thus simplified input part programming was a necessity for economic usage of numerical control lathes. The General Electric Company has now taken over Miturn and offers it on the General Electric MARK II Time-Sharing Network as GETURN.

GETURN is a relatively new, self-contained programming system specifically designed for lathes.[10] It is in fact a comprehensive production system composed of a number of subsystems which together make an important contribution to the optimum utilization of numerically controlled lathes. It is equipped with a complete metal cutting technological capability and thus relieves the part programmer of the necessity of determining cutting conditions, as well as the need to devise the geometry of each cut.

The following operations are carried out automatically by the

GETURN system:

i) the optimum sequence of operations is determined,

ii) the optimum depth of cut, feed and cutting speed are determined per cut,

iii) the programmed surface quality is reached,

iv) the best tool for each operation is selected from the stock of tools,

v) the operating instructions for the machine are formulated,

vi) the expected machine time for the programmed workpiece is specified,

vii) the tape is punched for the machine.

In order to carry out all these functions GETURN requires a group of technology input files. These files consist of a Tool File, a Methods File, a Material File and a Machine File. These files stay constant for long periods of time and only need updating in the event of changes in machining practices. Figure 12, shows the general input and output, whilst Figure 13, shows the relationship between the part program and the technological files.

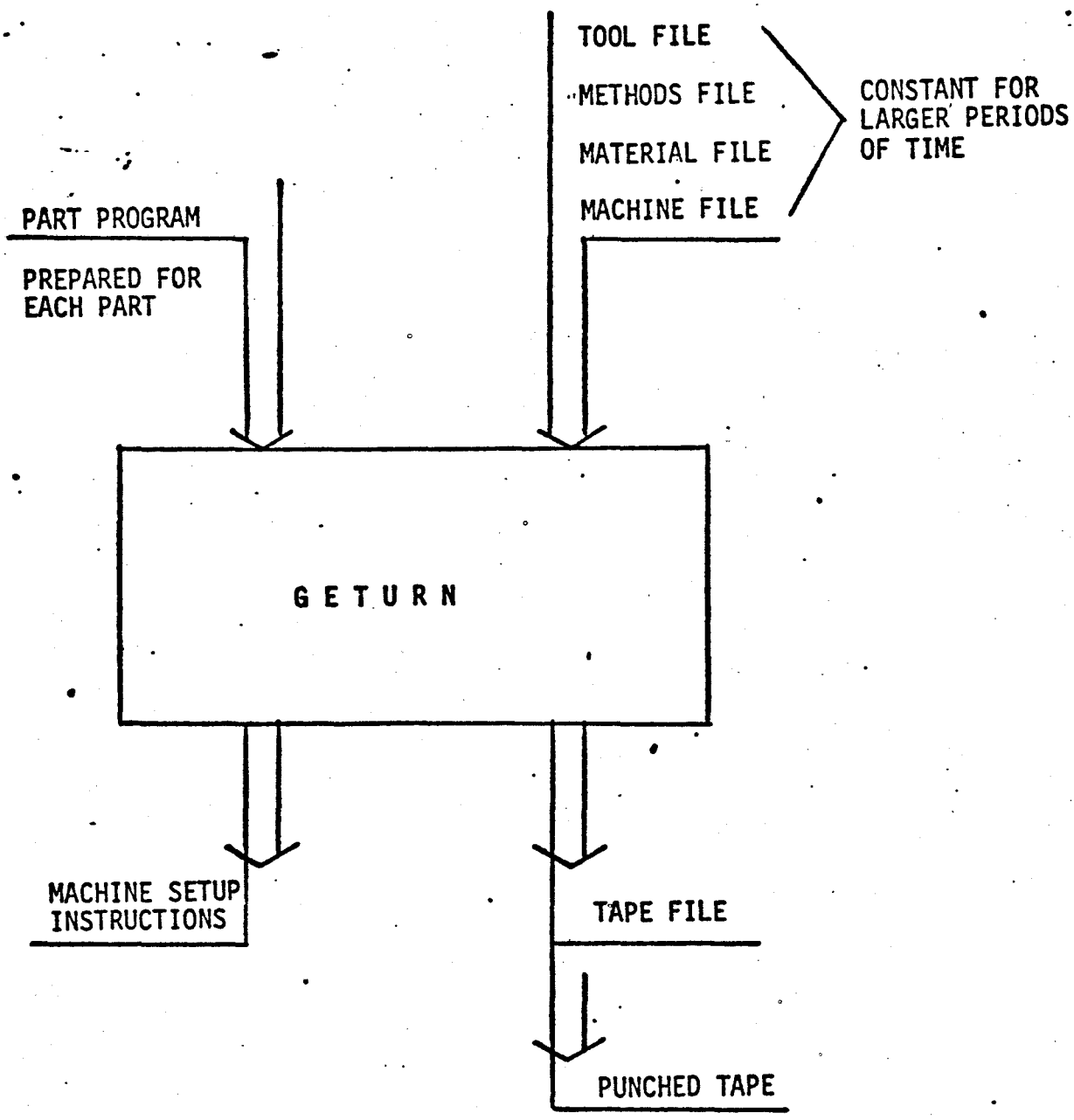


Figure 12 Relationship Between the General Input and Output of the GETURN Language

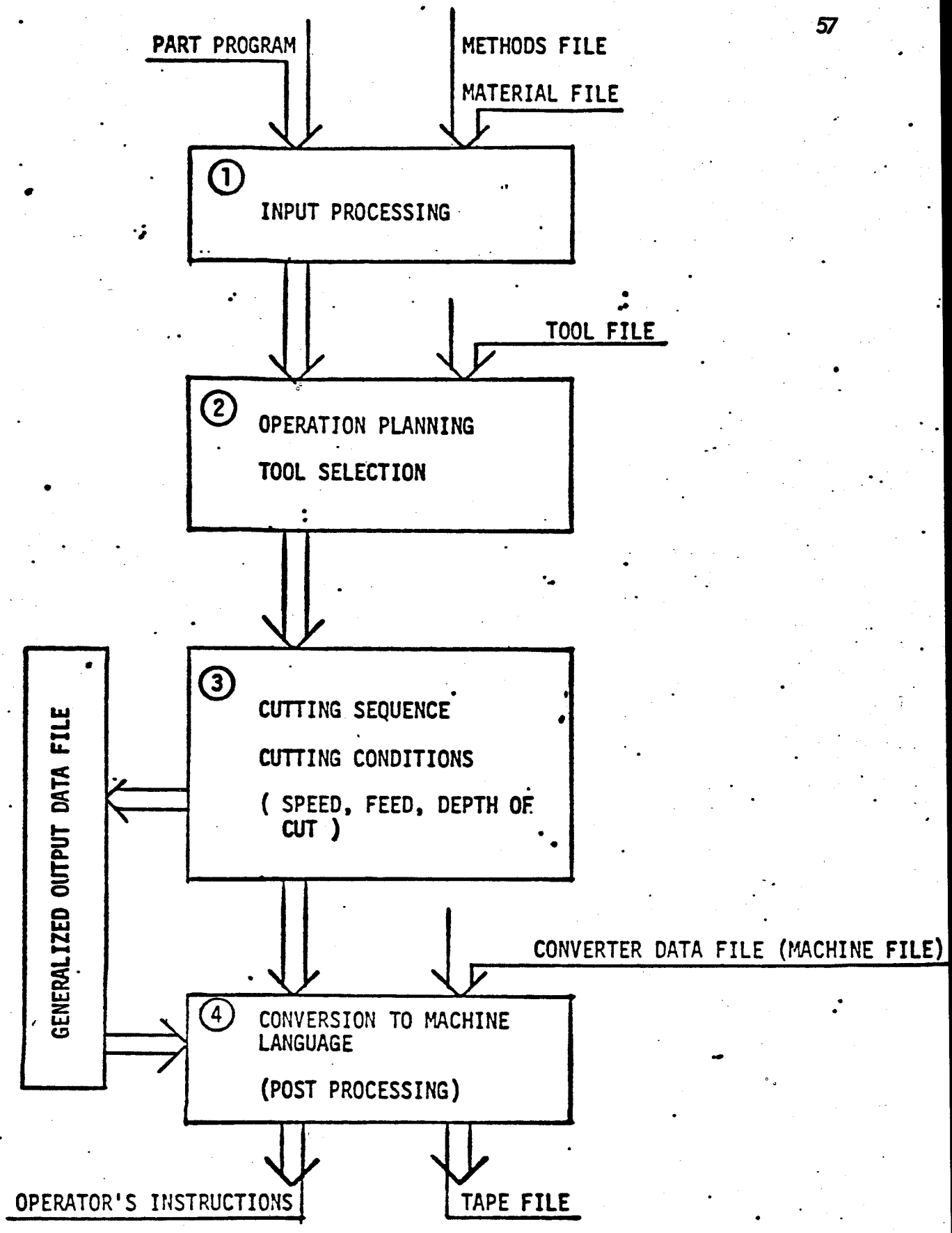


Figure 13 Relationship Between the Part Program and the Technological Files in GETURN

6.4.2 PART PROGRAMMING

The part program consists of the following:

i) heading with the programmer's and program's identifications,

ii) set-up specifications with important basic parameters of the machine used, overall workpiece size and configuration, clamping method, major positioning parameters and machinability,

iii) workpiece specification,

iv) comments for production.

As GETURN is only offered on a time sharing basis, Section (I) normally includes general identification of the users Machine File, Tool File etc. and is of relatively standard form.

The set-up specifications consist of two ordered sets of numbers which denote values of certain important basic parameters. The first set describes machine and workpiece related conditions, while the second set is concerned with the geometry of the set-up.

The first set consists of a series of six numbers which represent:[39]

1) Machine Number: Each lathe covered by GETURN is given a number which is entered in the Machine Catalogue.

2) Operation Code: This is a two digit number with the first digit specifying the work holding method and the second digit the desired operations.

e.g. Operation Code 10 would mean that the workpiece is held in the chuck without any other support and that both roughing and

finishing were required.

3) Material Record No.: This refers to the workpiece material of which the technological parameters are supplied in the Material File under the same number.

4) Relative Machinability Rating: This number is used as a multiplier of the cutting speeds given in the Material File. It is a value between 0 and 1.

5) Depth to Feed Ratio: The desired ratio of the depth of cut to the feed per revolution. This value will override the value given in the Material File unless set to zero.

6) Overall Set-Up Rigidity Coefficient: This value has a proportional effect on the maximum chip cross section permitted. It ranges from 0.1 to 1.0.

The second set of numbers are shown in Figure 14:

L1) Distance from spindle nose to the end of chuck.

L2) Required safety distance. This is the minimum that will separate the tool tip from the work holding device in the axial direction.

L3) Distance from work holding device to right hand end of blank.

L4) Total length of external machined contour.

L5) Total length of internal machined contour.

L6) Safety distance around workpiece. This distance provides an envelope around the current shape of the workpiece within which

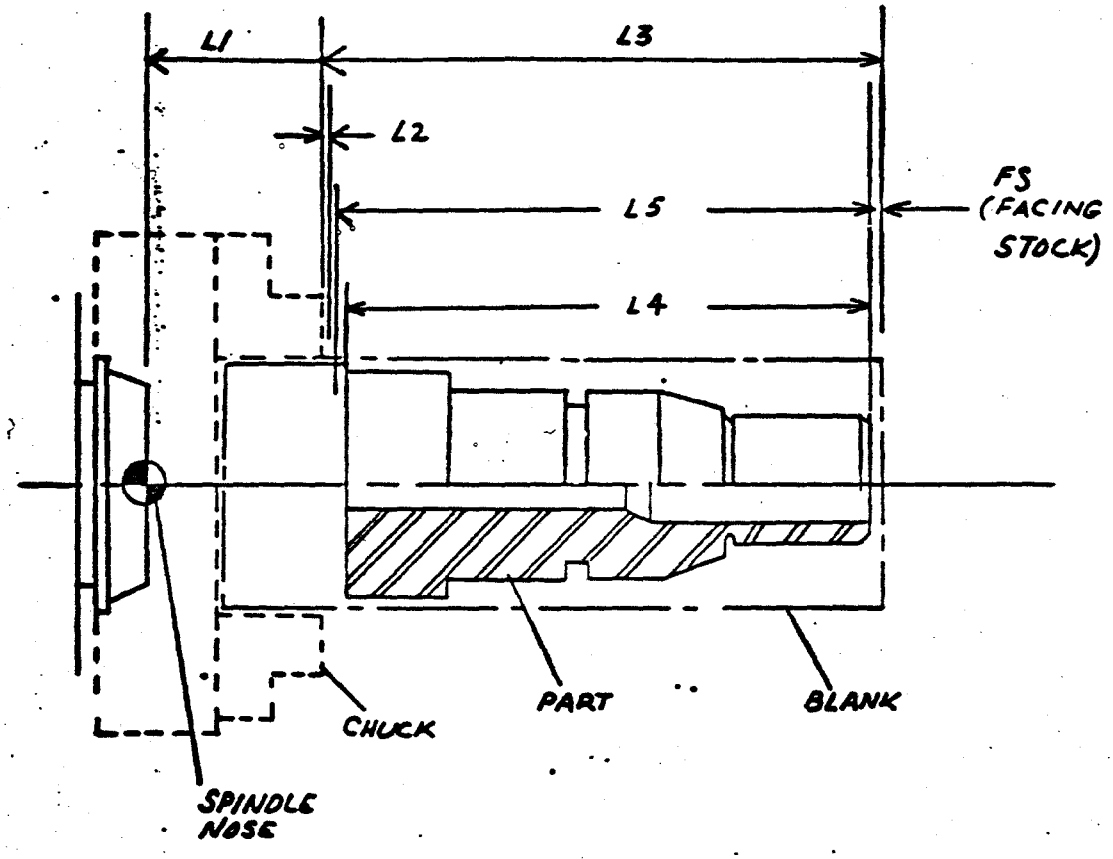


Figure 14 Required Set-up Distances for the GETURN Language

the tool will move only at metal cutting feed.

When using a steady rest an additional dimension has to be defined.

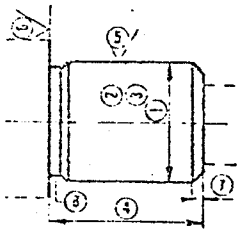
L7) Distance from work holding device to centre of steady rest. In this case the safety distance L2 must include the steady rest.

For section (iii) in order to describe the workpiece fully the internal and external contours of it are separately divided into longitudinal body segments, called elements, and each element is fully described by a string of parameters which can be taken directly from the blueprint. Figure 15 shows details of all the elements presently available. As can be seen they are: cylinder, taper, circular arc and threaded cylinder. Radial grooves are also available, but are not considered longitudinal elements. They can be superimposed on any cylinder or threaded cylinder.

Part programming consists of listing the elements consecutively from the tail end of the workpiece to the headstock end i.e. from right to left. Consecutive diameters from right to left must be non-decreasing for the external contour and non-increasing for the internal contour, with the exception, of course, of grooves.

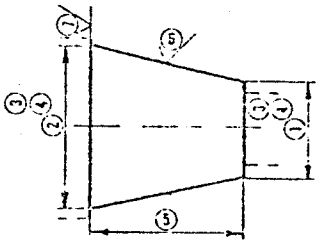
In programming the part, first the blank size and then the finished contour is described. The order of the descriptive elements is also rigid and is as follows:

First the number of external elements is listed, followed by the first



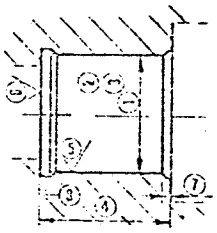
1 2 3 4 5 6 7 8

cylinder (1)



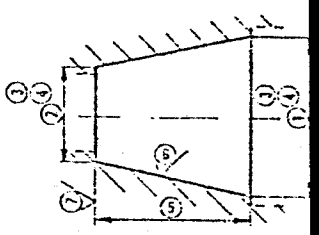
1 2 3 4 5 6 7

taper (3)



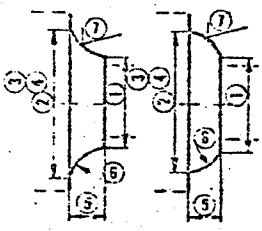
1 2 3 4 5 6 7 8

cylinder (1)



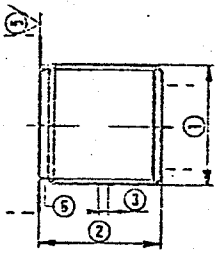
1 2 3 4 5 6 7

taper (3)



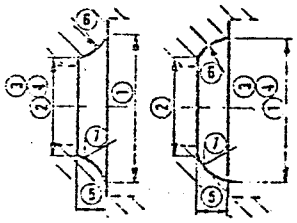
1 2 3 4 5 6 7

radius (4)



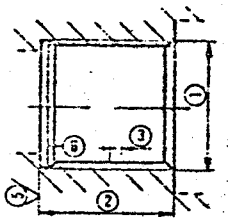
1 2 3 4 5 6

threadcyl. (5)



1 2 3 4 5 6 7

radius (4)



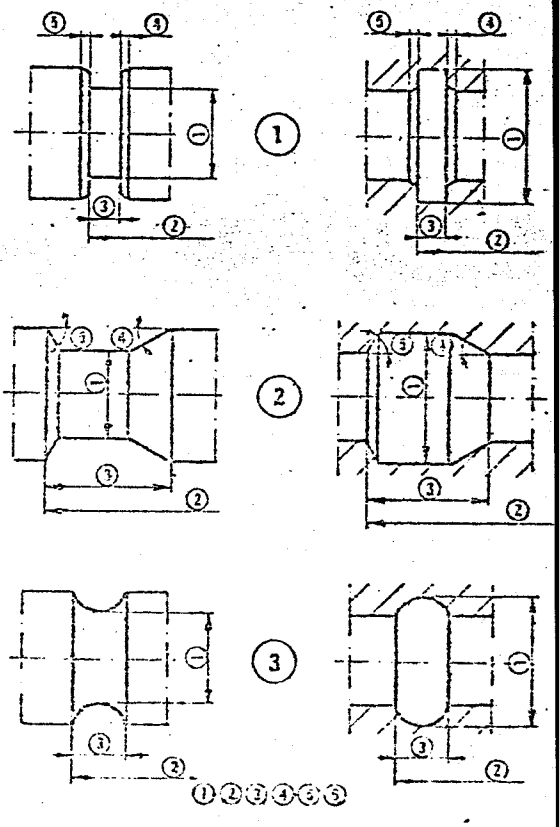
1 2 3 4 5 6

threadcyl. (5)

□ parameters for rough elements
○ parameters for machined elements

□ parameters for rough elements
○ parameters for machined elements

superimposed elements



1 2 3 4 5 6

Figure 15 Details of the Elements Available in GETURN

element code together with the appropriate parameter string. When all the external elements are defined, the number of external grooves followed by parameter strings defining each groove is listed. The whole procedure is then repeated for all internal contours.

Figure 16 shows a typical component and the programmed description of both the initial blank and the required finished workpiece. As previously stated the initial blank is described first starting with the external contour.

Line 210 shows the number of external elements on the initial blank.

In this case it is three: a taper, an arc and a cylinder.

Line 220 describes the types of elements.

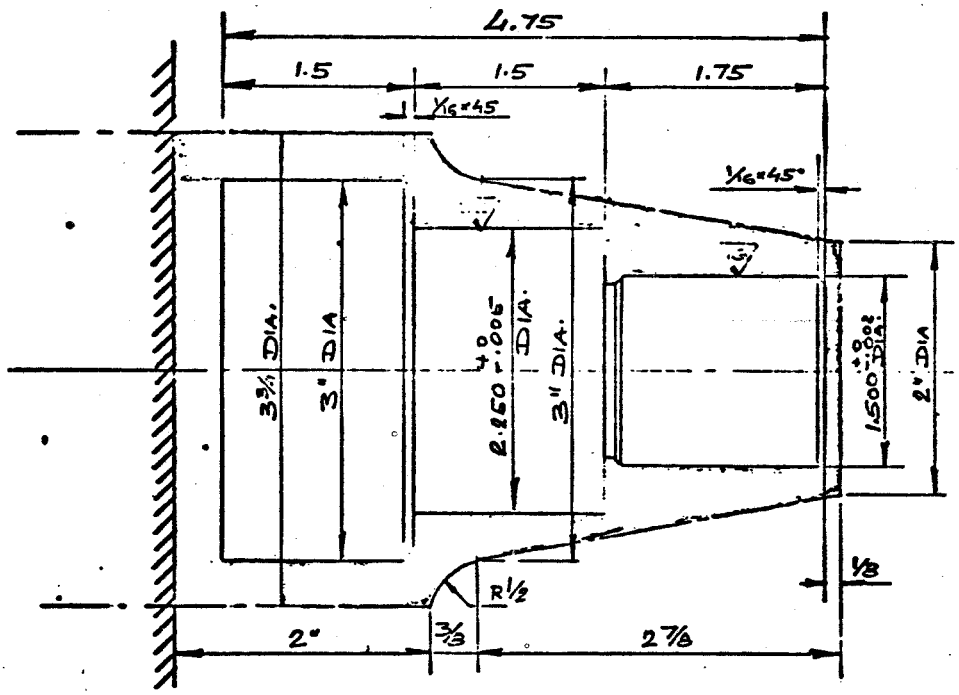
i.e. 3 denotes a taper

4 denotes an arc

1 denotes a cylinder

It is worthwhile mentioning once again that the elements must be listed from right to left.

Line 230-250 list the parameters of each element. Here it should be noted that measurements for each element are taken from the end of the previous element. This is the case for all elements except grooves. e.g. Consider Line 240 which is the list of parameters for the arc. The third parameter denotes the length of the arc and is 0.375", which is the




```
190*BLANK DESCRIPTION*****************************************
200*EXTERNAL CONTOUR
210 3
220 3 4 1
230 TPR 2 3 2.875
240 ARC 3 3.75 .375 .5
250 CYL 3.75 2
300*INTERNAL CONTOUR
310 0
400*MACHINED PART DESCRIPTION*********************************
410*EXTERNAL CONTOUR
420 3
430 1 1 1
440 CYL 1.5 0 -.002 1.75 16 16 .063 -1
450 CYL 2.25 0 -.005 1.5 32 0 0 0
460 CYL 3 0 0 1.5 0 0 .063 0
500*EXT.GROOVES
510 NUMBER OF GROOVES
520 0
600*INTERNAL CONTOUR
610 0
700*INT.GROOVES
800*FACE DESCRIPTION
810 FACE .125 0
820*COMMENTS***************************************************
```

Figure 16 A Typical GETURN Program

length as measured from the end of the taper which was the previous element.

Line 300-310 describe the internal contour of the blank. As there is not internal form, a zero is programmed.

The initial blank has now been completely described and thus the required finished workpiece follows, again starting with the external contours.

Line 420 denotes three external elements.

Line 430 states that all three are cylinders.

Line 440-460 contain the required string of parameters describing each cylinder. Here it is interesting to note the difference between the parameters defined at Line 250 and those at Line 440. At Line 250 only two parameters were defined, the diameter and length of the cylinder, whereas at Line 440 eight parameters were defined. The reason for this is that the blank size is considered to be "rough" and tolerances and surface finish values are not required to describe it. Thus the extra parameters required for the finished cylinder describe the tolerance bondaries on the diameter, the surface finish on the diameter and shoulder, the chamfer and the undercut.

Line 500-510 describe the external grooves. Here a zero is programmed although the blueprint shows an undercut at the end of the 1.500" diameter cylinder. This is because the undercut is of standard dimensions and is defined at the same time the cylinder is defined.

Line 600 and onwards proceed to describe the internal contour and face dimensions.

The fourth section of the parts program is a general one for comments. It allows the part programmer to add any special instructions required and to pass them on to the lathe set-up man and/or operator.

6.4.3 TECHNOLOGY FILES

The tool file is a collection of tools that are available for a particular lathe, and are admissible i.e. they conform to certain geometrical restrictions. The tools to be used with any part program are automatically selected by GETURN, with selection being based on type numbers and within the same type on tool parameter values.[39]

Each tool is defined by the tool number, the tool name and eighteen tool parameters. The tool number is a four digit number, with the first two digits reflecting the tool type (e.g. drill, boring bar etc.) and the second two the serial number which determines the tool's position in the list with respect to other tools of the same type. The tool's relative position is important in ensuring that the best tool is selected

for any operation. Consider, for instance, a finish boring operation and a number of boring bars could enter the rough hole and have the reach to perform the operation. GETURN will select the sturdiest one, i.e. the one with largest diameter, and if there is more than one with this diameter it will then select the shortest one. Thus in order to assist the GETURN program finish boring bars should be ordered, in the tool file, first in descending order according to diameter. Then if more than one bar has the same diameter they should be listed in ascending order according to length. The diameter and the length would be known as the selection sensitive parameters for finish boring bars.

The Methods File contains fifty constants, the values of which influence the execution of machining operations. The effect of these constants is independent of the workpiece and tool tip materials. There are two standard methods files available for GETURN, one for metric and the other for inch system machine tools. If either of these files is not suitable then provision has been made for the user to enter his own file or to use a standard file with a number of modifications.

The depth of cut for roughing operations is automatically determined by considering the programmed set-up rigidity coefficient, the rigidity of the workpiece, the available spindle torque and is limited by the maximum allowable cutting force on the tool and the maximum depth of cut. For finishing operations the depth of cut will equal the finishing

stock.

The feed for roughing operations is chosen so as to implement the desired depth/feed ratio. For finishing operations the feed will be such as to provide the programmed surface finish quality required. Cutting speed is chosen from the Material File.

The machining sequence is an ordered execution of the operations required to complete a part. It is dependent upon the Operations Code, certain Methods File constants and by the absence or presence of internal/external machining operations.

The normal machining sequence is as follows:

1) End facing
2) Drilling
3) External roughing
4) Internal roughing
5) External finishing
6) External grooving
7) External threading
8) Internal finishing
9) Internal grooving
10) Internal threading

The parameters which influence depth of cut, feed and speed are supplied in the Material File. There are two standard material files

available for GETURN, one for metric machines, the other for inch system machines. These files can be used as is but there is provision for a user to modify any parameter or create his own files.

The standard Material File represents two classes of materials: ductile and brittle. The parameters for the "ductile class" are based on SAE. 1112, cold drawn free machining steel, and those for the "brittle class" on soft grey iron. The effect of these parameters can be modified through three entries of the set-up specification: relative machinability rating, the depth to feed ratio and the overall rigidity coefficient.

The relative machinability rating is used as a multiplier to all speed values of the Material File. For example if the rating entered is 0.5 then certain parameters will be automatically halved.

The depth to feed ratio can be modified by entering the required value in the set-up specifications. This will override the value in the Material File.

The overall rigidity coefficient is used as a multiplier to the depth of cut and feed computed by GETURN.

Complete description of these files and their make up are well documented in the GETURN part programming manual.

6.4.4 COMMENTS

From a part programmer's point of view GETURN is an excellent programming language. It has an extremely simplified input format and

relieves the part programmer of a great deal of technological decisions.

It is designed only for lathe work, which means that companies using GETURN would require another language for programming any other machine tool. It is, however, adaptable to any make of lathe. Post processors are required but General Electric does offer a standard lathe post processor called GELATH.[23] This generalized post processor is based on the concept that most numerical control turning machines have features that are common to one another. Any differences can be compensated for by additional entries.

The fact that any element description requires fixed parameters restricts the dimensioning of a blueprint. In order to be compatible with the simplified input format, components should be dimensioned in the same way as the input required for any element.

6.5 EXAPT[6,17,36,38,40]

6.5.1 INTRODUCTION

EXAPT was developed by the Technical Institutes of Aachen, Berlin and Stuttgart, in Germany.[17] Work first started on the EXAPT programming language in 1965 when it was found that no language, at that time, covered any of the technological tasks required of a programmer. The starting point was the programming language APT and the name EXAPT indicates the relations of the structure and computer machining of the EXAPT system to the APT programming system. It was also noted that

APT requires a complicated input, even for the simplest components, and thus only some of the geometrical defining possibilities of APT were adopted for EXAPT.

Expansions were made in EXAPT, in relation to APT, and it was made possible to include the technology for different machining problems into the automatic production of punched tapes.

The complete EXAPT programming system consists of three languages. Of these languages EXAPT I deals with the programming of numerical control machine tools with point-to-point and straight path control and thus is used for drilling and simple milling operations. EXAPT 2 provides for programming turning operations, dealing with the programming of machine tools with straight path and contour control, with circular and linear interpolation. EXAPT 3 is a further extension of EXAPT I and allows for the programming of contour milling.[17]

As this report is dealing with languages for turning operations only EXAPT 2 will be discussed in any detail.

At the present time EXAPT 2, along with determining the tool path, also calculates the cutting values and the cut distribution. It will not automatically select the tool but this refinement is presently being developed.

At this stage it is worthwhile mentioning the EXAPT association. The previously mentioned Technical Institutes collaborated with a large

number of industrial companies to found the Association for the Advancement of the EXAPT Programming System in 1967. The basic tasks of the Association are to introduce the EXAPT programming system, to preserve the uniformity of the system, and to expand the system according to the user's experience and technical progress. Because of EXAPT's similarity to APT close contact is maintained between the EXAPT Association and the APT Administration.

6.5.2 PART PROGRAMMING (EXAPT 2)

A block diagram indicating the program structure for EXAPT 2 is shown in Figure 17. The part program consists basically of three sections, initial instruction, defining instructions and machining instructions. The initial instructions define the part number, the machine number which denotes the post processor to be used, the material and the chucking details.

The second section of the part program can be subdivided into three further sections. These are the geometric definition of the blank, the geometric definition of the finished workpiece, and the technological definitions.

The contours of the blank and finished part are respectively described in two statements:

i) that of geometric definitions according to APT and

ii) that of the contour-describing instructions for connection under the use of the previously defined elements.

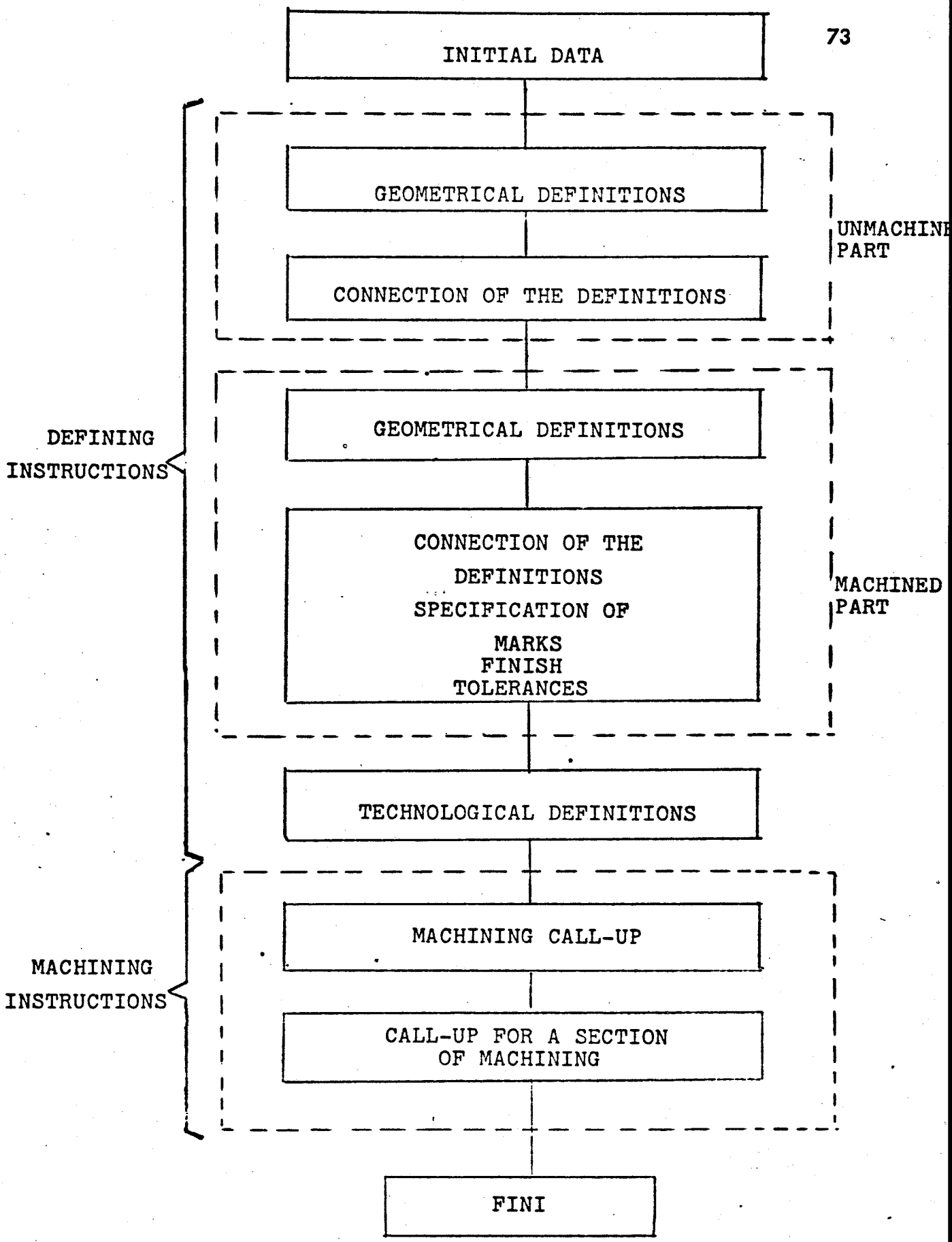


Figure 17 The Program Structure for EXAPT 2

The APT definitions contain points, straight lines, and circles which are appropriate for the description of the turning part. The selection of these definitions was based on a study of turning part drawings of different manufacturing processes. Some simplified definitions have been developed however, in order to be able to describe such geometry as axis-parallel straight lines which occur very frequently in turning. Figure 18 illustrates the use of these simplified definitions for the description of plane surfaces and diameters. The connecting instructions are required in order to determine the shape of the contour.

The blank description is introduced through the instruction BLANCO and the finished part description with PARTCO. Both are terminated by the instruction TERMCO. Due to rotation symmetry, turning parts are unambiguously defined when one half of the geometrical cross section is described. The description is effected only in the X,Y plane, with the X axis taken as the axis of rotation. The contour description consists of geometrical elements linked together. The linkage is effected through statements in which a geometrical element is always attached to a part of the contour.

The first linking statement must fix a starting point for which an optional contour point can be chosen. Then starting from this point and going in a clockwise direction the full contour of the component is described. Through a determination of the same type of description for

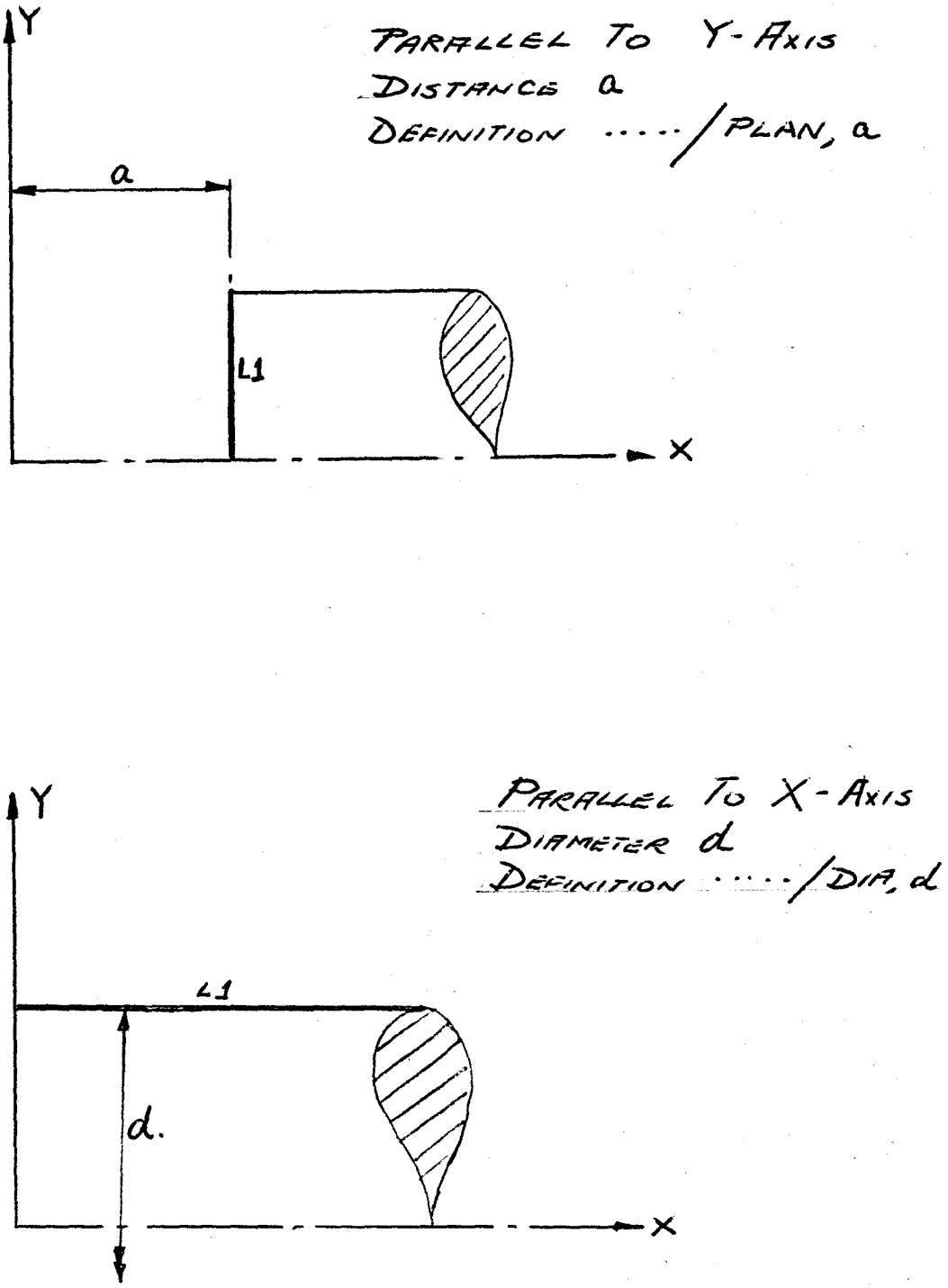


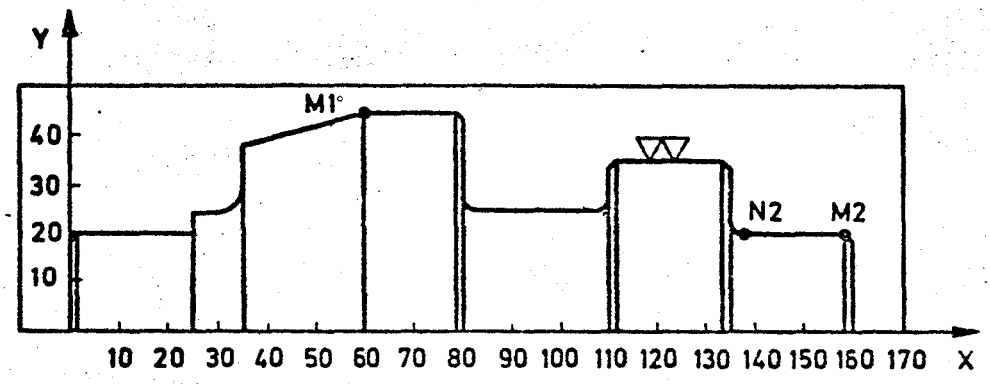
Figure 18 Illustration of the Use of Simplified Definitions in EXAPT 2

blank and finished part, the disposition of the volume to be removed is clearly established.

The contour description provides the basis for the processing of the machining stages in the computer in order to determine the sequence of cuts necessary to obtain the contour of the finished part. When technological data is to be automatically determined, it is necessary to select points on the contour of the finished part. These points, or marks as they are sometimes called, define a region which can be finished by a single tool.

Figure 19 shows an example of the descriptions of the rough and finish contours required for that component. It can be seen that the structure of the connecting instructions is very similar to that required in APT. An extra feature is the use of additional modifiers which make possible the description of phases and curvature radius as well as the determination of the required finish and tolerances.

The technological definitions could also be called machining definitions and their structure corresponds to the structure of all EXAPT instructions, a major word followed by a series of modifiers. The basic type of machining is determined by the major word and a list of admissible machining definitions is given in Figure 20. Figure 20a lists all the definitions available for internal machining, whilst Figure 20b lists all those available for external work.



ROHTEILBESCHREIBUNG

```
CONTUR/BLANCO
BEGIN  /-10, 0, YLARGE, PLAN, -10
     RGT/ DIA, 100
     RGT/ PLAN, 170
     RGT/ DIA, 0
TERMCO
SURFIN /ROUGH
```

FERTIGTEILBESCHREIBUNG

```
CONTUR/PARTCO
BEGIN  /0,0, YLARGE, PLAN, 0, BEVEL, 1
     RGT/ DIA, 40, ROUND, 2
     LFT / PLAN, 25
     RGT/ DIA, 50, ROUND, 5
     LFT / PLAN, 35
     RGT/ (LINE/(POINT/60,45), ATANGL, 15)
M1, FWD/ DIA, 90, BEVEL, 1
     RGT/ PLAN, 80, ROUND, 2
     LFT / DIA, 50, ROUND, 2
     LFT / PLAN, 110, BEVEL, 1
     RGT/ DIA, 70, FIN
     RGT/ (LINE/(POINT/134,35), ATANGL, -45)
     RGT/ PLAN, 135, ROUND, 2
N2, M2, LFT / DIA, 40, BEVEL, 1
     RGT/ PLAN, 160
     RGT/ DIA, 0
TERMCO
```

Figure 19 Description of a Rough and Finish Contour in the EXAPT 2 Language

Major words \ Modifiers			SO	DIAMET d	DEPTH r	TOOL e,f	FEED s	SPEED v	SPIRET g	NOREV	DIA&EV i	ANG&EV m
Single Machining Operations	Centre drill	CDRILL	●	○	○	●	○	○	—	—	—	—
	Drill	DRILL	●	●	●	●	○	○	○	○	—	—
	Ream	REAM	●	●	●	●	○	○	○	—	—	—
	Spiral sink	SISINK	●	●	●	●	○	○	○	—	—	—
	Sink	SINK	●	●	●	●	○	○	○	—	—	—
	Tap	TAP	●	●	●	●	—	○	○	—	—	—
	Counter sink	COSINK	●	○	—	●	○	○	○	—	○	○
	Bore	BORE	●	●	●	●	●	●	○	—	—	—

Figure 20A

● Modifier must be given

○ Modifier can be given

Major Words \ Modifiers			SO	LONG	CROSS	ATANGL	TOCL e,f	SETANG	DEPTH r	FEED s	SPEED v	ROUGH	FIN	FINE	TAT p	PITCH h	OSETNO no	SPIRET g
Single Machining Operations	Turn	TURN	●	(1)	(1)	(1)	●	●	○	○	○	(2)	(2)	(2)	—	—	○	○
	Contour turn	CONT	●	(1)	(1)	(1)	●	●	○	○	○	(2)	(2)	(2)	—	—	○	○
	Recess	GROOV	●	(1)	(1)	(1)	●	●	○	○	○	(2)	(2)	(2)	—	—	○	○
	Thread	THREAD	●	(1)	(1)	(1)	●	●	○	○	●	(2)	(2)	(2)	●	●	○	○

Those modifiers characterised by 1, exclude each other mutually.
The same applies for modifiers characterised by 2.

Figure 20B

(2) ● Modifier must be given

○ Modifier can be given

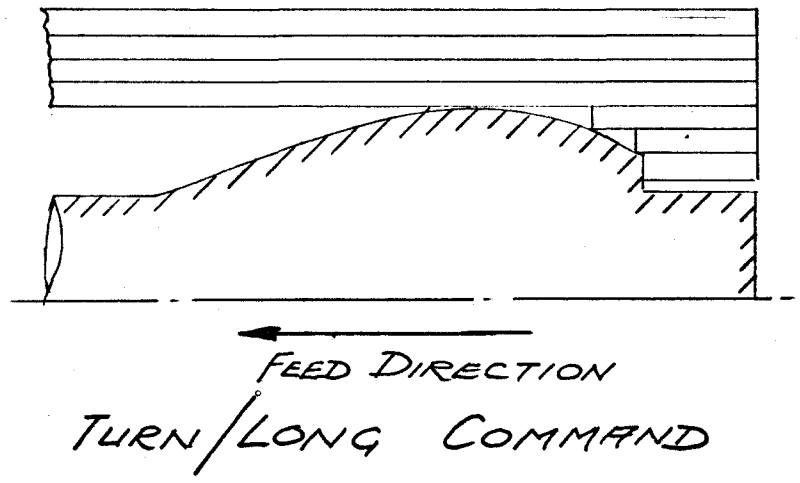
Figure 20 List of Admissible Machining Definitions Available in EXAPT 2

As can be noted certain modifiers must be given with each major word. The statement LONG, CROSS, or ATANGL gives the direction of feed of the tool. This direction must be stated for TURN, GROOV, and THREAD, but is not necessary for CONT. LONG is in a direction parallel to the axis of rotation, CROSS is perpendicular to it, and ATANGL gives an angular direction of feed[6].

The difference between the TURN and CONT commands should be noted. For example with a command of TURN/LONG, a series of cuts parallel to the X-axis will be obtained, but the tool cannot traverse, recess or groove. With CONT/LONG the tool is able to traverse, recess and groove. Figure 21 shows the difference between the two commands.

Feed, speed and depth of cut can all be given or can be automatically determined. At this stage the required tool must also be called up, using a tool identity number and a magazine number.

Each machining definition is signified by code number in order to allow it to be called up during the third section of the part program, the machining instructions. The call-up begins with the major word WORK, followed by the code number given to the appropriate machining definition. The machining positions at which the called up work operation is to be carried out is then stated immediately following the WORK statement. The major word CUT is used for the machining position call up, together with the marks, which were defined during the contour



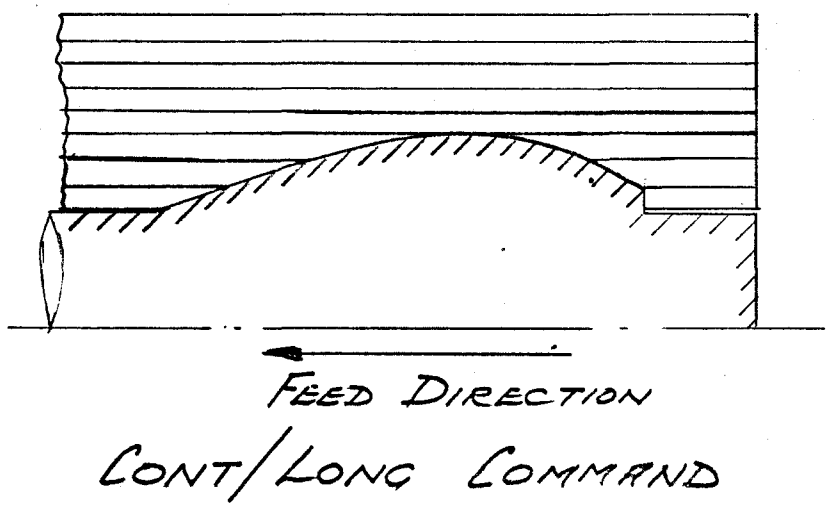


Figure 21 Illustration of the Difference Between the TURN and CONT Commands in EXAPT 2

descriptions, which will denote the length to be cut.

The sequence of the WORK instruction with the corresponding machining position call-ups determines the work sequence for a clamping. The different clamping positions are described by the following instructions: CLAMP/A, INVERS. The value "A" indicates the position of the clamping plane in the workpiece coordinate system. The modifier INVERS means that the workpiece is clamped while turned 180° from the finished part description.

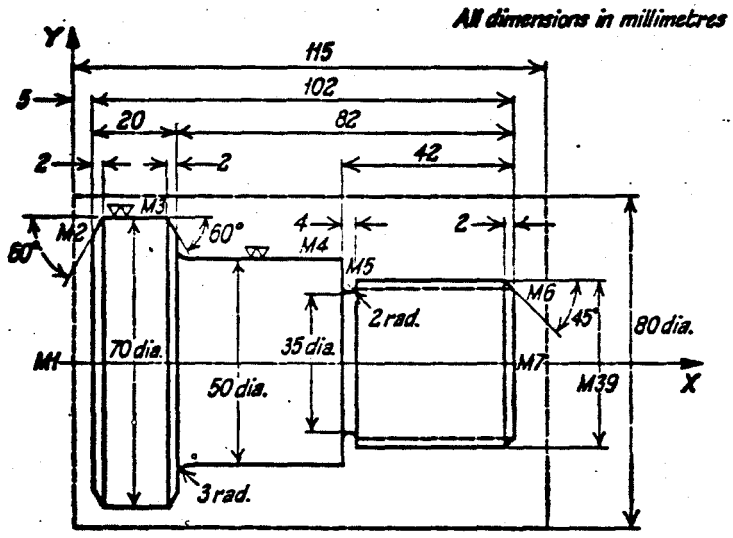
Figure 22 shows the EXAPT 2 part program for the component shown.

6.5.3 CUTTING VALUE DETERMINATION

As stated earlier speed, feed and depth of cut can be automatically determined by the EXAPT program. In order for the technological processor of the EXAPT program to be able to calculate these values, a number of input files are required. These are a tool file, a material file, and a machine tool file.

The tool file contains all the necessary data regarding operating conditions and dimensions of the tool. The data stored in the material file contains all limiting values and characteristic data for a material relative to the material of the cutting edge. The machine tool file is similar and contains such limiting values as maximum power available, and spindle speeds.

Thus in the program for automatically determining the cutting value,




```
Initial data
                  PARTNO/STUD, DRAWING No. 6                                  1
                  MACHIN/NCDM 1                                               2
                  PART/MATERL,3                                               3
                  CHUCK/12,90                                                 4
                  CLPRNT                                                      5

Unmachined part
                  CONTUR/BLANCO                                               6
                  BEGIN/0,0,YLARGE,PLAN,0                                     7
                  RGT/DIA,80                                                  8
                  RGT/PLAN,115                                                9
                  RGT/DIA,0                                                  10
                  TERMCO                                                     11

Machined part
                  SURFIN/ROUGH                                               12
                  CONTUR/PARTCO                                              13
geometric definitions
                  M1,  BEGIN/5,0,YLARGE,PLAN,5                               14
                       L1 = LINE/7,(70/2),ATANGL,60                          15
connection of definit - ions
                  M2,  RGT/L1                                                16
                       RGT/DIA,70,PIN                                        17
                       L2 = LINE/23,(70/2),ATANGL,-60                        18
specification of marks and tolerances
                  M3,  RGT/L2                                                19
                       RGT/PLAN,25,ROUND,3                                   20
                       LFT/DIA,50,PIN                                        21
                  M4,  RGT/PLAN,65                                           22
                       C1 = CIRCLE/(65+2),(35/2+2),2                         23
                  M5,  LFT/C1                                                24
                       LFT/PLAN,(65+4)                                       25
                       RGT/DIA,39,BEVEL,2                                    26
                  M6,  RGT/PLAN,107                                          27
                  M7,  RGT/DIA,0                                             28
                       TERMCO                                                29

additional data
                  CLDIST/1                                                   30
                  OVSIZE/FIN,0.8                                             31
                  COOLNT/ON                                                  32

technological definitions
                  A1 = TURN/SO,CROSS,TOOL,111,1,SETANG,-90,ROUGH             33
                  A2 = CONT/SO,TOOL,222,2,SETANG,-90,ROUGH                   34
                  A3 = CONT/SO,TOOL,333,3,SETANG,-90,FIN                     35
                  A4 = GROOV/SO,CROSS,TOOL,444,4,SETANG,-90,ROUGH            36
                  A5 = THREAD/SO,LONG,TOOL,555,5,SETANG,-90,SPEED,
                       6,TAT,1,PITCH,3,2,ROUGH                               38

clamp statement
machining call-up
call-up for a section of machining
                  .
clamp statement
machining call-up
call-up for a section of machining
                  .
                  .
                  .
                  .
                  CLAMP/115,INVERS                                           39
                  FDSTOP/PLAN,35                                             40
                  WORK/A1                                                    41
                  CUT/M1,TO,M2                                               42
                  WORK/A2,A3                                                 43
                  CUT/M2,TO,M3                                               44
                  FDSTOP/NOMORE                                              45
                  CLAMP/5                                                    46
                  WORK/A1                                                    47
                  CUT/M6,TO,M7                                               48
                  WORK/A2                                                    49
                  CUT/M6,RE,M3                                               50
                  WORK/A3                                                    51
                  CUT/M4,RE,M3                                               52
                  WORK/A4                                                    53
                  CUT/M4,TO,M5                                               54
                  WORK/A5                                                    55
                  CUT/M6,RE,M5                                               56

end of program
                  FINI                                                       57
```

Figure 22 A Typical EXAPT 2 Program

the cutting conditions and the operational conditions of the tool are adjusted to one another by a series of mathematical equations. The results for each single cut are calculated with the aid of the corresponding data from the material, tool and machine tool files.

The depth of cut is determined through the cut distribution, which is based on the maximum effective cutting edge length in the case of several passes.[6]

The feed is calculated from the depth of cut with the aid of a chip form criterion. To obtain this criterion the chip width, the chip thickness and the resulting width to thickness ratio are required. In the case of roughing, the feed is limited by the maximum permissible chip thickness and the maximum permissible cutting power of the tool as well as the maximum torque of the machine. In the case of finish turning, the quality of surface finish required is the factor limiting the feed.

The value of the cutting speed is calculated using the following equation when tool life is the determination criterion:[6]

$$\text{Cutting Speed} = K \cdot A^F \cdot S^E \cdot T^G \cdot VB^H$$

Where A = depth of cut

K = tool life constant

S = feed

T = tool life

VB = width of wear mark

E,F,G,H = constants

As stated previously the three cutting values can all be entered by the programmer during the machining definitions.

6.5.4 PROCESSOR FLOW CHART

The flow chart of the processor is shown in Figure 23. The contour table is compiled in the geometrical processor. The technological processor extends the contour description corresponding to the description of the chuck. Subsequently, the segment to be machined is determined, the range to be machined is corrected relative to the tool geometry, and finally the balanced and complicated cut distributions for TURN or CONT are calculated.

The KOLLIS program is an interesting innovation, in that it determines whether the area to be cut can actually be machined by a tool with the given geometry. Consider the component in Figure 24. It is required to machine from M1 to M2, using a tool shaped as shown. In theory either of two commands would suffice, that is TURN/LONG----or CONT/LONG----. If TURN/LONG---- was used the final profile obtained would be that shown by the solid line(a). This is because the TURN statement will not allow the tool to change direction to form the groove. If CONT/LONG---- was used then the area below the dot-dash line cannot be machined because a collision with the secondary cutting edge would occur. Thus the correct way to machine from M1 to M2 would be to TURN/LONG---- followed by a GROOV/CROSS----.

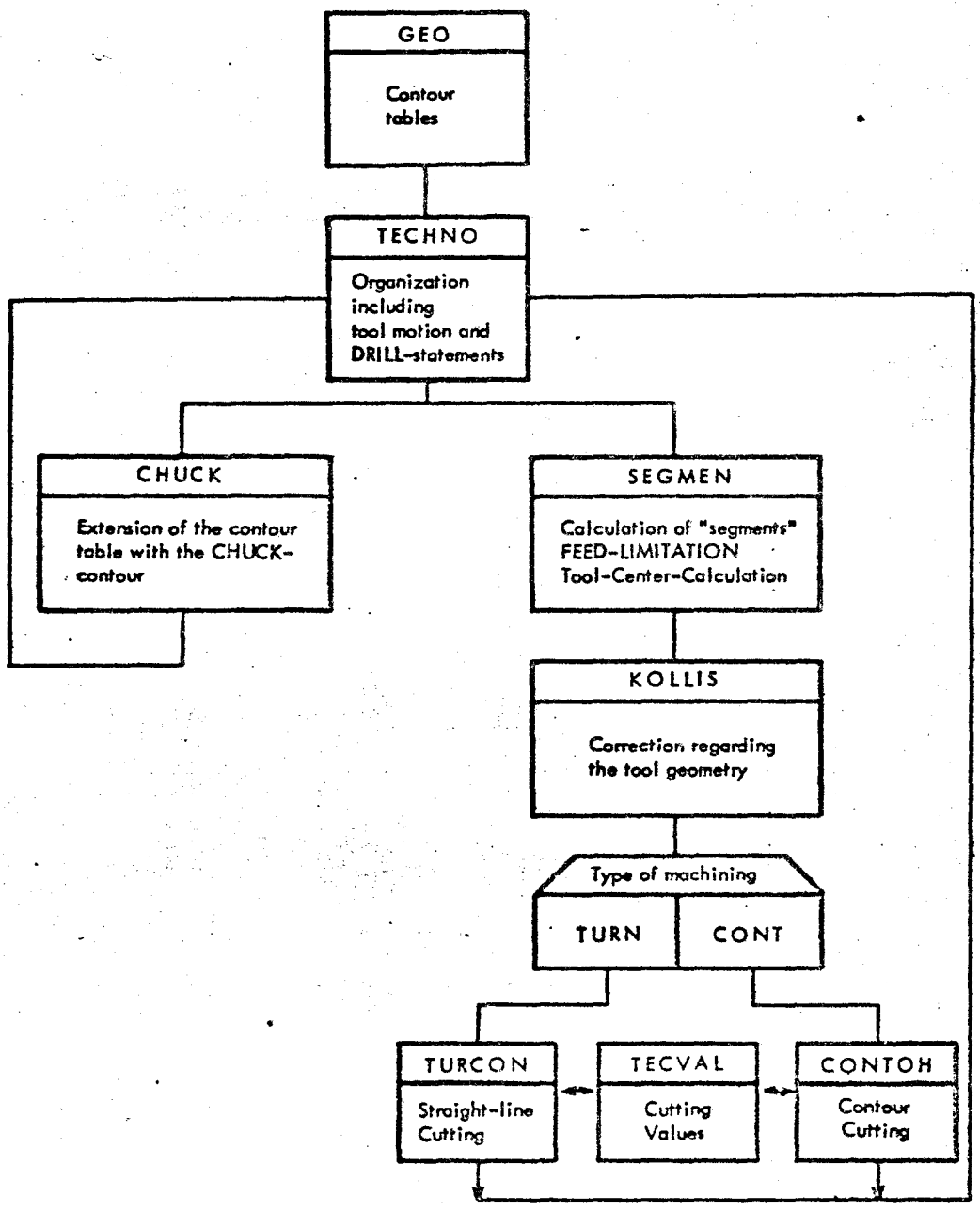


Figure 23 The Processor Flow Chart of the EXAPT 2 System

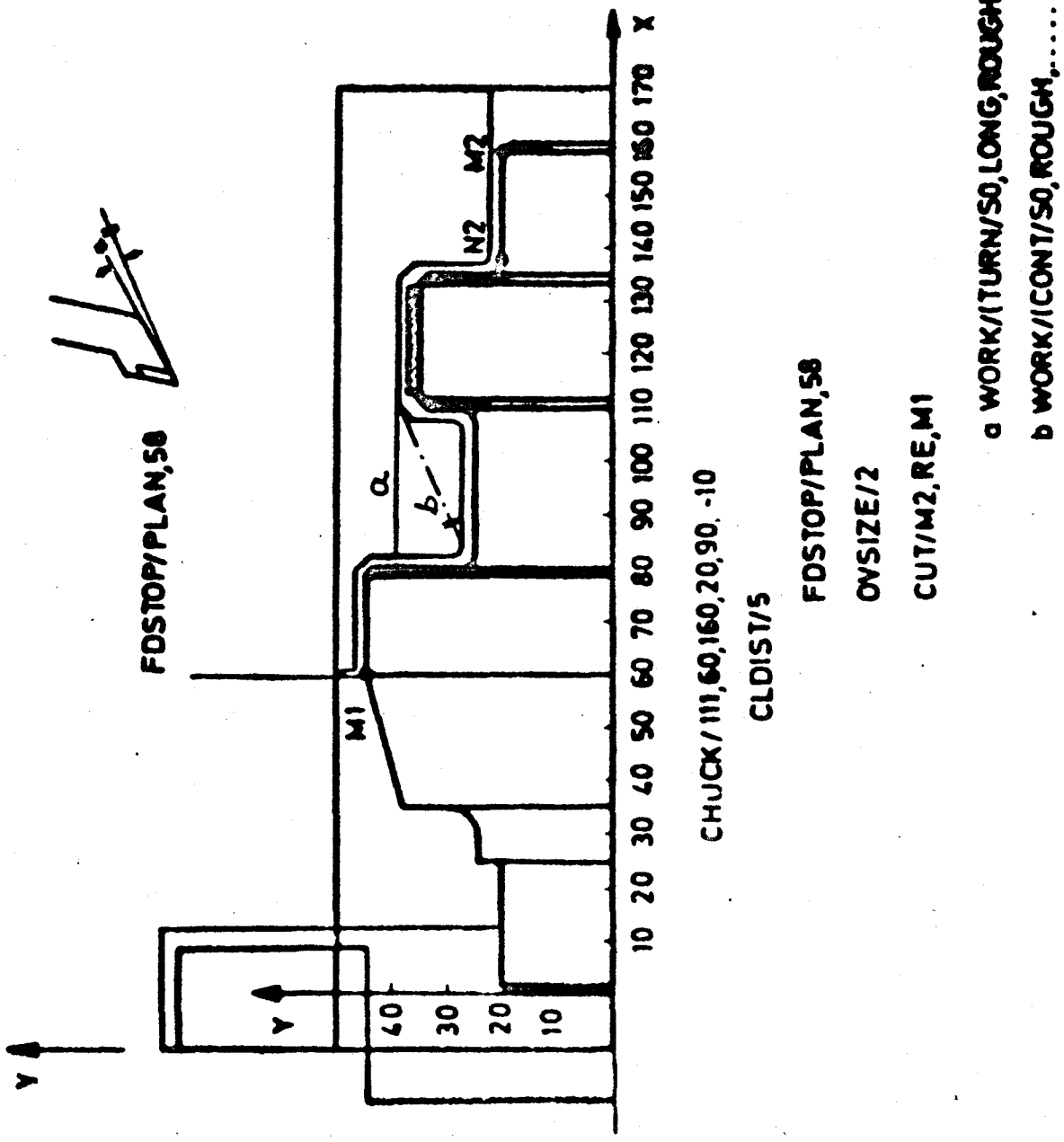


Figure 24 Illustration of the Kollis Program in EXAPT 2

6.5.5 COMMENTS

The EXAPT family of numerical control programming systems is a very unique one. To the author's knowledge it is the only system that offers solutions to the technological decisions facing a part programmer for various types of manufacturing processes. The programming language for drilling, milling and turning is a unified one, which is an advantage for part programmers who are required to program various types of machines. It is also advantageous in terms of post processors. At present machine tool users often desire from the manufacturer the necessary post processor for the programming language they are interested in. A unified language, such as EXAPT, would put an end to the recurring expenses of post processors for the manufacturer.

In order for EXAPT not to be dependent on any computer manufacturer, translater programs were written in FORTRAN IV, which makes it possible to transfer the programs in a relatively short time from one computer to another.

EXAPT 2 is a relatively easy language to learn to program in, especially if a programmer has had previous experience with APT. One minor problem that could lead to confusion is that the contour definitions are defined using X and Y coordinates. On the North American continent the Y axis referred to is known as the Z axis.

6.6 TNC LANGUAGE[47]

6.6.1 INTRODUCTION

The TNC Language is a programming language presently being developed by Jones and Lamson for use with the Jones and Lamson TNC lathes. It combines the programming functions of control, geometry, and machinability into a single system of inter-connected computer program modules which act as a single program. The fact that the control aspect is built in eliminates the need for a distinct post processor.

The TNC Language examines the total relationship of the input commands to the specific machining situation, then uses this in formation to optimize the machining commands and to generate many of the tool motions automatically.[47] It also selects the appropriate tools, specifies the set-up tooling position and the corresponding set dimensions.

Built into the system is a machinability file, a tool geometry file, and a machine specification file. These files are an integral part of the system and do not have to be created by the user. Provisions have been made, however, for overriding these built-in files and thus the user can include special tools or substitute his own machining technology. The machinability file is based on the accumulated technology and experience of Jones and Lamson over the years. It contains machining data on more than 800 different materials. The tool file is based on the standard Jones and Lamson carbide tools and tool holders. It contains the tool geometry

information on around 350 tools and holders. The machine specification file describes the dimensional characteristics of the Jones and Lamson TNC lathes.

The system is designed for use with the GE-635 Mark II time sharing computer but can be readily adapted to fit other time sharing systems or in-house computers because all the supporting computer programs are written in FORTRAN IV.

6.6.2 PART PROGRAMMING

Very little information on part programming in TNC is available as Jones and Lamson is still finalizing the development. The TNC language consists of words of four or less characters which are the exact or slightly abbreviated form of the words normally used in lathe terminology to describe lathe cuts, slide motions, and miscellaneous machine functions. Each word in the language is a command to the system and has associated with it one or more parameters which explicitly define the command. It appears that each contour required needs three types of commands in order to produce that contour. The control is defined first, followed by the geometry of the contour and then the machinability commands. This process is repeated for each contour.

The control commands index the turret, turn spindle on, turn coolant on etc. The geometry command describes the type of contour and its dimensions.

The machinability commands activate the machinability files and machinability logic in order to compute the speed, feed etc. Each machinability command contains a word command describing the type of machining operation required and a series of parameters describing the start and finish points.

Figure 25 shows an example of a program.

6.6.3 COMMENTS

The TNC language will cover various types of Jones and Lamson TNC lathes including bar machines and chuckers. It will also be adapted to suit the 4 axis machine and production centre machine that Jones and Lamson offer.

A collision check is incorporated in the TNC language by means of an algorithm which senses the shape of a user defined clearance envelope surrounding the workpiece. A warning message is printed out if the turret doing the cutting causes the opposite turret to move inside the clearance envelope.

The fact that the language can be used either on an in-house computer or on a time-sharing basis is an advantage but this could be offset by the fact that it is limited to Jones and Lamson TNC lathes.

6.7 CUTS[25]

6.7.1 INTRODUCTION

CUTS is the Warner and Swasey Computer Utilized Turning System

```
10 PART NO. 71765   OP. 2  PART NAME-LOGIC TEST PIECE
20 OPERATION DESCRIPTION-DRILL, R&F BORE, REAM,TAP,
30 R&F FACE END, AND R&F OUTSIDE PROFILE
40 (CONT.)
 50 CONTROL- BENDIX 800
60 7-5-71 MATERIAL-SAE 3120  MACHINE-312 COMBI
70 COMBI,312,B,0.,40.,9.5,S19,200.,10.,2000.,33.,2000.,9.,2.5
80 1.,0.,DRLL,1.,0.,0.,7.,0.,0.,5.
90 6.,0.,REAM,1.,0.,0.,7.,0.,0.,
100 3.,0.,TAP,1.,0.,0.,5.,0.,0.,5.
110 4.,0.,-411.,1.,.047,0.,0.,14727.,0.,0.
120 5.,0.,470.,1.,.047,0.,5.,0.,0.,5.
130 2.,0.,-411.,1.,.047,0.,0..14727..0..0.
140 1.,422.,0.,0.,.047,0.,0.
150 2.,422.,0.,0.,.047,0.,0.,0.,0.,0.
160 STRT,0.,7.18,270.
170 FACE,0.,1.25,0.
180 ANGL,135.,1.0E3,-5.
190 TURN,12.375,-4.,0.
200 ZZZ,90.,0.,0.
210 CSTP,14.,7.3,300.
220 INDH,1.,1.
230 INDR,1.
240 SPIN,FWRD
250 HEX
260 COOL,ON
 270 DRLL,1.375,7.280,2.1,0.,0.,0.,1.,1.,4B
280 HEX1-DRILL 1.375 HOLE 4.5 DEEP
290 1.,DRLL,1.375,5.18,0.,0.,0.,0.,0.,0.
300 ZZZ
310 INDH,5.,1.
320 CUT,.020,330.
330 STRT,-1.75,7.28,270.,1.,100.,0.,0.,0.,0.
340 BORE,1.75,-.780,0.,0.,0.,0.,2.,0.,0.
350 ZZZ,90.,0.,0.,0.,0.,0.,0.,0.,0.
360 HEX5-R . BORE 1.75 HOLE
370 2.,RBS7,-1.75,.78,0.,0.,0.,.160,0.,0.
380 ZZZ
390 INDH,4.,1.
400 CUT,.050,410.
410 STRT,-2.575,7.0,180.,1.0,0.,0.,0.,0.,0.
420 FACE,2.575,.50,0.,0.,0.,0.,3.,0.,0.
430 ZZZ,180.,0.,0.,0.,0.,0.,0.,0.,0.
440 HEX4-R. FACE END,OVERRIDE BOTH FEED AND SPEED
450 3.,RFCE,-2.375,.50,300.,.015,0.,.130,0.,0.
460 ZZZ
470 INDH,5.,1.
480 CUT,0.,490.
490 STRT,-1.75,7.15,270.,1.0,0.,0.,0.,0.,0.
500 BORE,1.75,-.65,0.,0.,0.,0.,4.,2.,2.
510 FACE,1.75,-.250,0.,0.,0.,0.,5.,2.,2.
520 ZZZ,180.,0.,0.,0.,0.,0.,0.,0.,0.
530 HEX6-F. BORE 1.75 DIA
540 4.,FBS7,-1.75,.65,0.,0.,0...020,0.,5.
542 HEX6-FACE OUT TO .50 DEPTH
545 5.,FBS7,-1.75,.25,0.,0.,0.,.020,0.,5.
550 ZZZ
```

Figure 25 A Typical TNC Program

and was developed by Warner and Swasey for use with their numerically controlled turret lathes. It is now available for programming the Warner and Swasey 2-axis turret lathes and is designed for use with the IBM 360 computer model 30 with 65K storage.[25] CUTS includes methods determination and tool engineering and thus requires only a simplified input description of the workpiece from the programmer. The CUTS program has the ability to decide:

i) the number of cuts to be taken on each surface,

ii) how the surface is to be produced i.e. by turning, facing or contouring,

iii) how much stock each cut will remove,

iv) in what sequence the cuts are to be made,

v) what tools are required to make the part,

vi) the position of the turret each tool will occupy,

vii) feeds and speeds for each cut,

viii) a punched tape,

ix) general operator instructions.

CUTS will handle almost every type of standard cutting operation. Included are: turning, boring, drilling, internal and external facing, necking, grooving, chamfering, threading, tapers, and radii extending through a full 90° arc. Basically the program only excludes behind shoulder operations and complex contours.

The system operates in conjunction with a permanent tooling system, which uses a few simple and basic tool holders. Thus the number of variables connected with tooling with which the program must cope is drastically reduced.

6.7.2 PART PROGRAMMING

Part programming in CUTS is relatively simple in that a brief description of the workpiece, giving rough and finish sizes, material specification, finish requirements and a few other descriptions, is written in two steps. Step one identifies the part, the work material, and the holding information, whilst step two describes each surface to be machined. The surface description is done using normal shop terms and decimal dimensions, and can normally be taken directly from a blueprint.

The CUTS program then analyzes this information, with the help of information drawn from the appropriate tool file, material file etc. and then decides how the part should be machined.[25] The output is given in two forms, one, a printout, understandable to the operator; the second, a tape, understandable to the machine. In order to simplify the input and aid the part programmer Warner and Swasey have developed two standard forms for use with the CUTS program. The first form is known as a Header Sheet and contains the information relating to the part description, the material, the machine to be used, various parameters

required for cutter location, the length of the part, and details of the end facing operation. The post processor is also called up using the code PPIDEN followed by the correct post processor name.

The second form is known as the Surface Description Sheet and, as the title suggests, provides the layout for the description of the required contour. Each contour is described, starting from right to left as the workpiece sits in the chuck, by means of a word describing it, followed by a series of decimal dimensions. The word descriptions are exactly the same, in most cases, as those used to describe the contour in the English language. For example the word for a chamfer is CHAMFER. The only word description that is abbreviated is diameter which reduces to DIAM. The input is fixed format and each contour requires a number of parameters, its length and its angle, whereas a diameter requires three or four parameters. The parameters required for a diameter are the upper and lower boundaries of the finished diameter, the initial stock size, and the required surface finish. The surface finish is an optional parameter and does not have to be specified.

The contour description is described in this way first externally, then internally. The CUT-PAST command ensures that the final boring pass will clear the end of the component and not leave any step due to blank size variations. All input programs must finish with the command ENDPART which signifies to the computer the finish of the part

description.

Figures 26, 27 and 28 show a component and the required input part program for that component.

6.7.3 COMMENTS

The main fact that stands out with CUTS is that it is only available for Warner and Swasey turret lathes, thus is a disadvantage for in order to use CUTS and keep a constant programming language a machine tool user must virtually forego his freedom of choice concerning machine tool selection.

The input part program is simplified especially when using the standard coding forms designed by Warner and Swasey, and in most cases requires the part programmer to transfer information from the blueprint to the coding forms.

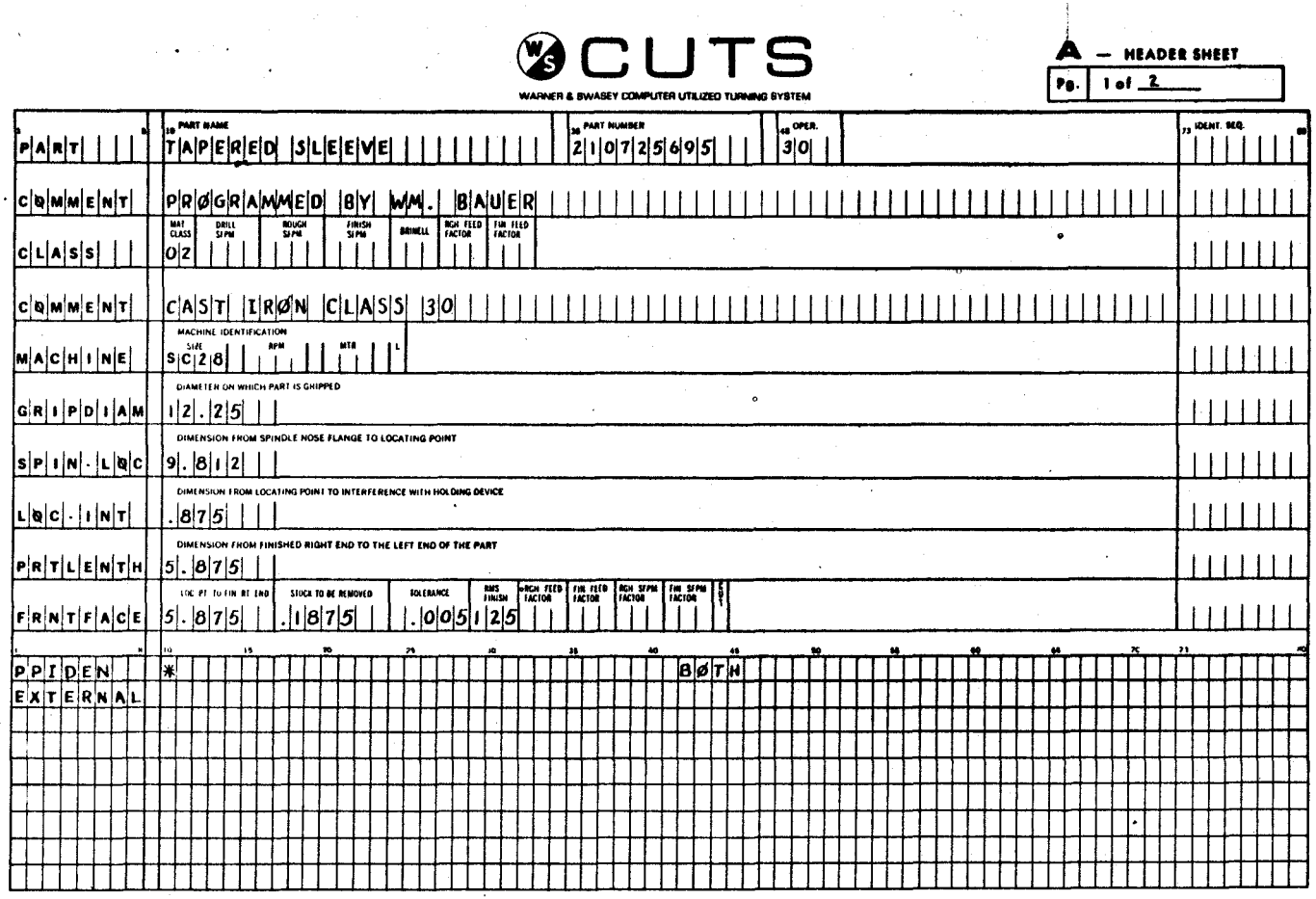
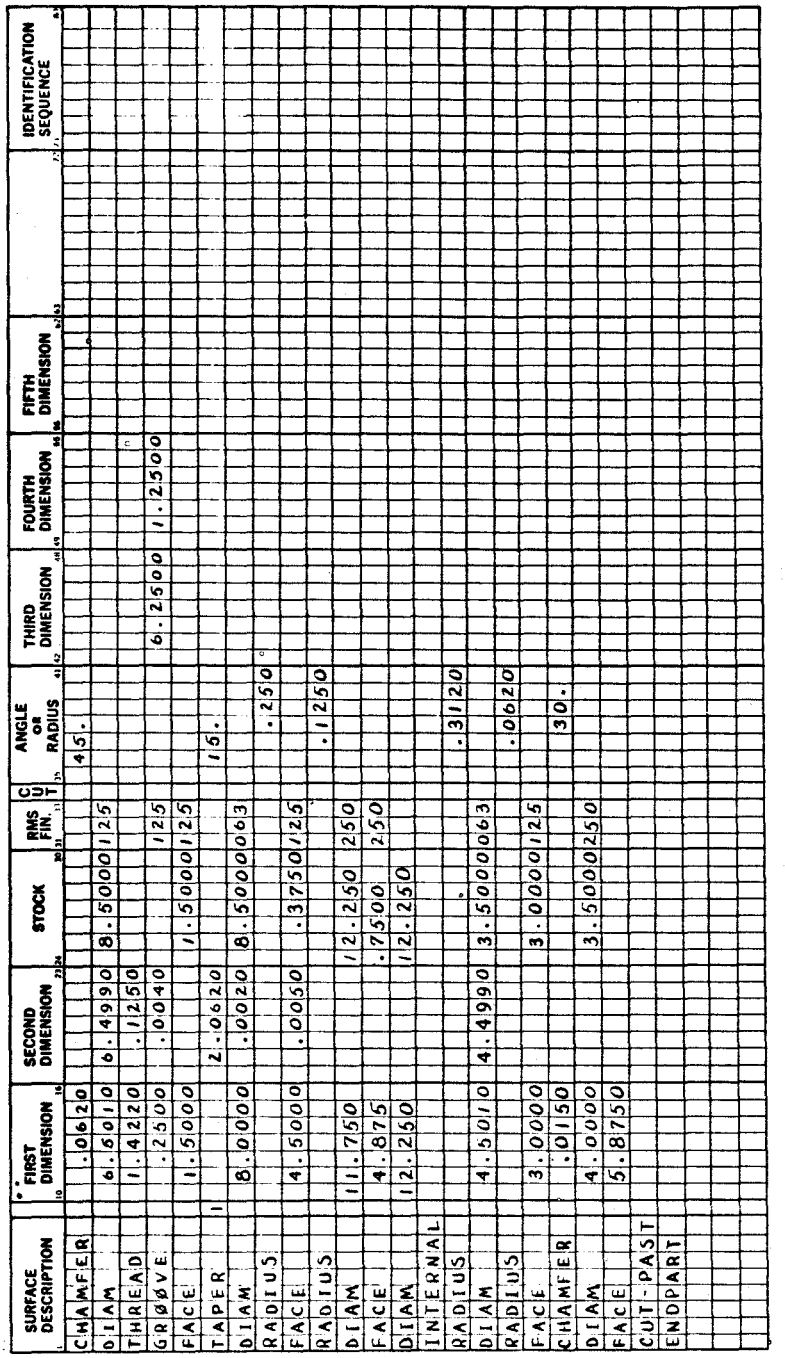


Figure 26 The Header Sheet for the CUTS Program for the Component Shown in Figure 28



CUTS

WARNER & SWASEY COMPUTER UTILIZED TURNING SYSTEM

B – SURFACE DESCRIPTION

Pg 2 of 2

SURFACE DESCRIPTION	FIRST DIMENSION	SECOND DIMENSION	STOCK	RMS FIN.	CUT	ANGLE OR RADIUS	THIRD DIMENSION	FOURTH DIMENSION	FIFTH DIMENSION	IDENTIFICATION SEQUENCE
CHAMFER	.0620					45.				
DIAM	6.6010	6.4990	8.5000	125						
THREAD	1.4220	.1250								
GROOVE	.2500	.0040		125			6.2500	1.2500		
FACE	1.5000		1.5000	125						
TAPER	11	2.0620				15.				
DIAM	8.0000	.0020	8.5000	0063						
RADIUS						.250				
FACE	4.5000	.0050	.3750	125						
RADIUS						.1250				
DIAM	11.750		12.250	250						
FACE	4.875		.7500	250						
DIAM	12.250		12.250							
INTERNAL										
RADIUS						.3120				
DIAM	4.5010	4.4990	3.5000	0063						
RADIUS						.0620				
FACE	3.0000		3.0000	125						
CHAMFER	.0150					30.				
DIAM	4.0000		3.5000	250						
FACE	5.8750									
CUT-PAST										
ENDPART										

Figure 27 The Surface Description Sheet for the CUTS Program for the Component Shown in Figure 28

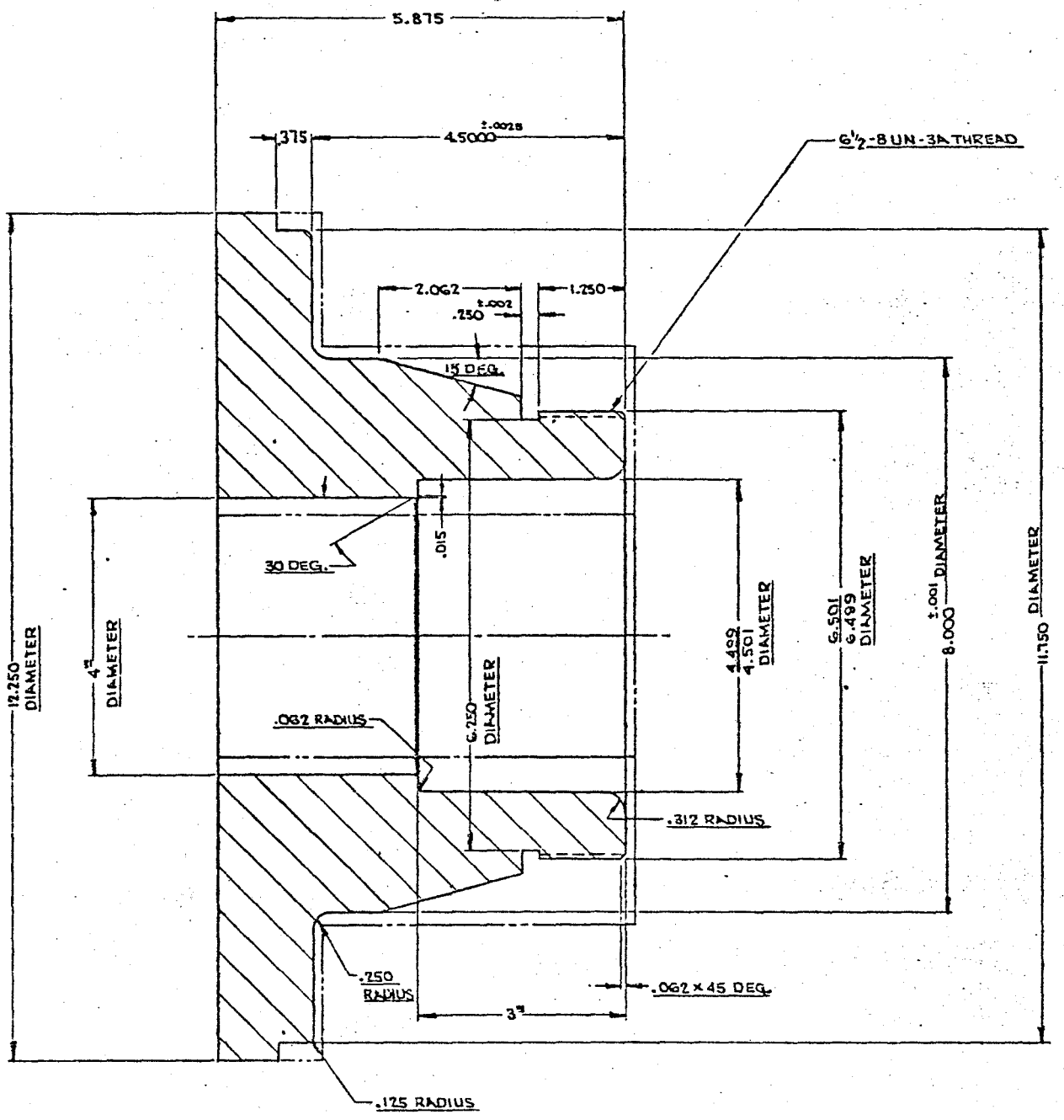


Figure 28 Component Used for the CUTS Program

7.0 DISCUSSION

Manual programming or computer assisted programming? This represents the first decision that one has to make when deciding upon a programming system. The basic criteria for this decision are the volume, variety and complexity of the work load. The volume and variety control the number of different tapes required during a given time interval, whilst the complexity determines the number of calculations to be made. Although most turning jobs are not extremely complex, a manual programmer can still be involved in long tedious and error prone calculations. Also the manual programmer is required to know the control system and peculiarities of each numerical control machine requiring programming.

Manual programming is probably only advantageous when the number of different control tapes is low and the lead time is great. It can, however, have a demoralizing effect on the part programmer because of the tedious nature of the work.

The prime advantages of computer assisted programming over manual programming are drastic reductions in lead time leading to a lower inventory, reduction of indirect labour costs because of the decrease in manual calculation required, and the improvement of program reliability. Obviously as the complexity of components increases, the need for computer assisted programming becomes imperative.

Choosing a suitable programming language is not just a simple matter of matching a language to a machine. A number of other factors should be considered. For instance computer accessibility is an important factor, as is the type of numerical control equipment to be purchased in the future or the availability of post processors.

Computer accessibility has a great deal of influence on the choice of a suitable language. The decision between in-house computer usage or some time-sharing mode must take into consideration which languages are available in which mode. Of course other factors must enter into the decision making process. With an in-house computer the numerical control programmer does not normally have direct access to the computer but has to "wait in line". This waiting is eliminated in a time sharing situation, together with the high initial investment cost of a computer or the high rental cost. For a company only using computer facilities for numerical control programming some method of time-sharing is probably most economical.

Progra mming languages can be split into two areas, general purpose languages and special purpose languages. General purpose languages are those which can be used for various types of machine tools e.g. mills, lathes etc. whilst special purpose languages can only be used with one type of machine tool. They can be further split into proprietary or non proprietary languages. If a language is selected without considering future needs then it is possible for a company to

have a large number of different programming languages being used and thus putting added pressure on the part programmer. A similar situation can develop if a proprietary language is selected. Thus a general purpose, non-proprietary language such as APT or EXAPT holds distinct advantages for the numerical control user with a variety of machine tools. The availability of a post processor is also an important criterion when choosing a language. Some programming languages such as ACTION II and SPLIT do not require a post processor but it must be remembered with these languages that a different general processor is required for each machine tool. Others such as GETURN have a standardized post processor available that can be adapted very simply for use with most lathes. Although numerous post processors have been written, especially for APT, there still remains problems with their availability. Some of the problems associated with post processors will be eliminated if standardization of control systems and the post processor design could be achieved. It is doubtful if the problem will ever be completely eliminated however, because of the diversity of machine tools.

Table I gives a comparison of the languages discussed with regard to time-sharing facilities, general purpose program etc. All the special purpose languages are written for lathes in this case.

Thus before selecting a programming language a number of general questions should be answered. These are:

LANGUAGE	IN-HOUSE COMPUTER	TIME SHARING	GENERAL PURPOSE P*	GENERAL PURPOSE NP*	SPECIAL PURPOSE P*	SPECIAL PURPOSE NP*	POST PROCESSOR REQUIRED
APT	*	*		*			*
ADAPT	*	*		*			*
REMAPT		*		*			*
COMPACT II		*		*			*
SPLIT	*		*				
ACTION II	*	*		*			
CINTURN	*					*	*
CHIPS	*	*			*		*
GETURN		*				*	*
EXAPT 2	*			*			*
TNC	*	*			*		*
CUTS	*				*		*

P* = Proprietary Program

NP*=Non-Proprietary Program

Table 1 A Comparison of the Various Programming Languages

i) What is the best method of computer accessibility?

ii) If an in-house system is used who will implement and maintain the language?

iii) General purpose or special purpose language?

iv) Is a post processor available or will a new one have to be written?

All the languages discussed have been with relationship to turning and thus they will be compared on this basis. The areas of comparison are input of information, the output obtained and the ability of the program to handle the required operations.

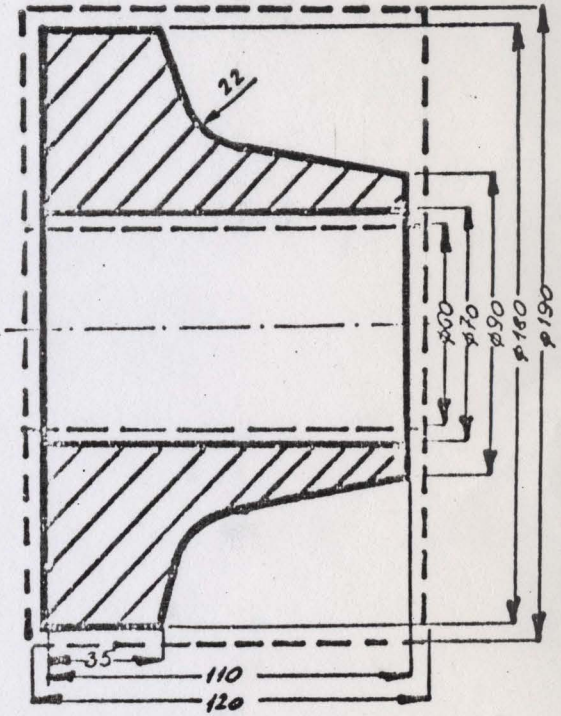
Obviously with the upward spiral of wages and the downward trend of computer costs the emphasis is on simplified input. Fixed format inputs provide a simpler method of part programming than do free format. Writing a part program becomes virtually filling in numbers in the appropriate columns of a standard coding sheet. Figure 29 shows a comparision of a program written in both a free format and a fixed format language. To pay for this simplicity, however, fixed format languages lose in flexibility. They can, normally, only describe a contour in a set way and thus adoption of a fixed format language could require redimensioning of all blueprints. Certainly it will require rigid standardization if no calculation is to be done by the part programmer. Most free format languages retain this flexibility, but then one could argue as to how much flexibility is required for turning.

```
PARTNO/DEMONSTRATIONSWERKSTUEK
        MACHIN/HEYNF,GE 100S,666.0,303.0,250.0,410.0,35.0
        PPFUN/SPINDL,2,AUSGAB,2,REIHEN,1,WZLIST,1,ZEILEN,1
REMARK/ROHTEILBESCHREIBUNG
        CONTUR/BLANCO
        BEGIN/-5,(60/2),YLARGE,PLAN,-5
        RGT/DIA,190
        RGT/PLAN,115
        RGT/DIA,60
        TERMCO
REMARK/FERTIGTEILBESCHREIBUNG
        SURFIN/FIN
        CONTUR/PARTCO
M1,     BEGIN/0,(70/2),YLARGE,PLAN,0.
M2,     RGT/DIA,180
L1    =LINE/(POINT/35,(180/2)),ATAGL,-75
L2    =LINE/(POINT/110,(90/2)),ATANGL,10
C1    =CIRCLE/XLARGE,L1,YLARGE,L2,RADIUS,22
M3,     RGT/L1
        FWD/C1
        FWD/L2
M4,     RGT/PLAN,110
M5,     RGT/DIA,70
        TERMCO
REMARK/TECHNOLOGISCHE AUSSAGEN
        PART/MATERL,203
        CUTLOC/BEHIND
        CLDIST/2
        OVSIZE/FIN,0.5
REMARK/BEARBEITUNGSDEFINITIONEN,WERKZEUGE
A1    =TURN/SO,CROSS,TOOL,114205,1,SETANG,270,ROUGH
A2    =TURN/SO,LONG, TOOL,124201,2,SETANG,270,ROUGH
A3    =CONT/SO,      TOOL,124201,3,SETANG,270,ROUGH,1,FIN
A4    =TURN/SO,LONG, TOOL,223201,7,SETANG,180,ROUGH
A5    =CONT/SO,      TOOL,223201,5,SETANG,180,OSETNO,2,FIN
REMARK/BEARBEITUNGSSTELLENAUFRUFE, 1.AUFSPANNUNG
        CHUCK/40000,310.0
        CLAMP/115,INVERS
        WORK/A1
        CUT/M2,RE,M1
        WORK/A2,A3
        CUT/M2,T=,M3
```

APT-LIKE

```
J.NAME  8/24/70
DEMONSTRATIONSWERKSTUEK
MM
2 10 193 5 125 110 125 3
1 0.8 10 1
1
1
190 120
1
1
60 120
4
5 4 3 1
90 108 0 0 50 0 0
110 140 0 0 17.5 22 0
140 180 0 0 7.5 0 0
180 0 0 35 0 0 0 0
0
1
1
70 0 0 110 0 0 0 0
```

MITURN



1:2,5

Figure 29 Comparison of Free Format and Fixed Format Inputs

The use of macros or canned cycles is an excellent innovation. Most programming languages in the Groups II and III utilize macros, in fact one, CINTURN, is based purely and simply on macros. These have the effect of reducing the required input by eliminating repetitive programming. Together with the ability to use pre-programmed macros it is important that the programmer is able to drive the tool over the surface. This is similar to entering the tool path as in APT. The reason for this is to enable the machine operator to utilize the tool offsets to correct any machining deviations.

Table II shows the various macros available in each language and also if it has a free or fixed format input.

It is really only the Group III languages that give anything extra in the way of output. They provide technological details on optimizing the cutting conditions, the required tooling, and the required position of the tooling. In order to fully utilize and optimize the numerically controlled machine this information is invaluable, especially if one is dealing with a large number of different components and materials. The problem is, of course, ensuring that the theoretical results work in practice. This can only be solved by continued research into metal cutting and tool materials. It should be remembered when adopting a Group III language that it may require considerable time and effort to be spent compiling the necessary files. The technological output can

LANGUAGE	FREE FORMAT	FIXED FORMAT	TURN INT.& EXT.	BORE INT.& EXT.	THREAD INT.& EXT.	GROOVE INT.& EXT.	TAPER TURN INT.& EXT.
APT	*						
ADAPT	*						
REMAPT	*						
COMPACT II	*		*	*	*		*
SPLIT	*						
ACTION II	*		*	*	*		
CINTURN	*		*	*	*	*	*
CHIPS		*	*	*	*	*	*
GETURN		*	*	*	*	*	*
EXAPT 2	*		*	*	*	*	*
TNC	*		*	*	*	*	*
CUTS		*	*	*	*	*	*

Table 11 The Various Macros Available with Each Language

only be as good as the information contained in the prepared files.

In order to ensure that the language selected will be able to program all the required components, one must remember to project into the future in an attempt to envisage new designs. The only two problem areas in turning would appear to be complex contours and behind the shoulder machining. Complex contours can be programmed utilizing free format languages but could possibly require some involved mathematical calculations on the part of the programmer. Behind the shoulder machining is not usually required because most components requiring such machining are normally machined in two operations.

8.0 CONCLUSIONS

The selection of the correct programming language is an extremely important function in the economic use of a numerical control machine tool. There is no hard and fast rule as to what language is the most suitable but the trend appears to be two fold. One way is towards the standardization of the language base e.g. APT and a building block approach based on this language. The other way is towards the specialized languages such as GETURN which simplify part programming but are restricted in application.

It would be convenient, but highly unlikely at the present, if all the associations, computer companies, and machine tool companies could agree on some method of standardization of programming languages.

APPENDIX 1
REFERENCES

REFERENCES

1. Amerago E.J.A. Brown R.H. — "The Machining of Metals" Prentice Hall 1968

2. Takeyama H., Honda T., Sekiguchi H., Inoue K., Takada K. — "Study on Automatic Programming for Numerical Control. Automatic Determination of Cutting Conditions in Longitudinal Rough Turning" Cirp Vol. 19 No. 4, Page 713, July 1971

3. Roberts A.D. Prentice R.C. — "Programming for Numerical Control Machines" McGraw Hill 1968

4. Wilson F.W. — "Numerical Control in Manufacturing" McGraw Hill 1963

5. Childs J.J. — "Principles of Numerical Control" Industrial Press 1965

6. Rechziegel D. — "The Distinctive Features of Exapt" Frontiers in Manufacturing Technology Vol. IV, Page 145. Institute of Science and Technology, University of Michigan 1969

7. McCarroll J.D. — "Computer-Aided Part Programming for Numerical Control" Institute of Science and Technology, University of Michigan 1969

8. Peters J. — "Programming Languages for N.C. Machine Tools" Technical Paper

9. Capellano P.G. Parodi M. — "Specially Oriented Languages for N.C." Technical Paper

10. Kerry T.M. — "N.C. Lathe Programming with Geturn" Technical Paper MS72-150, Society of Manufacturing Engineers, Michigan 1972

11.	Wessels J.G.	"The Computerization of Technological Process in General, with Special Reference to the Geturn Program for Numerically Controlled Lathes" Technical Paper MS72-151, Society of Manufacturing Engineers, Michigan 1972
12.	Dallas D.B.	"A Job Shop Discovers the Cinturn Macros" Manufacturing Engineering & Management Vol. 67 No. 6, Page 14, December, 1971
13.	Ahuja D.V.	"Numerical Control Graphics" Paper Presented at Third International Seminar on Optimization of Manufacturing Systems June, 1971
14.	Schilperoort B.A.	"Group Technology and Production Preparation for Conventional and Numerically Controlled Operations" Technical Paper MS72-193, Society of Manufacturing Engineers, Michigan 1972
15.	Anon.	"Mitum, A Computer-Aided Production Planning System for Numerically Controlled Lathes" Technical Paper
16.	IIT Research Institute	"APT Part Programming" McGraw Hill 1967
17.	Opitz H., Simon W., Spur G., Stute G.	"The Programming of Numerically Controlled Machine Tools" Excerpt from Frontiers in Manufacturing Technology, Institute of Science and Technology, Michigan September, 1967
18.	Vlahos C.J.	"Fundamentals of Numerical Control" Chilton Book Company 1968
19.	Rasch F.O., Rolstadas A.	"Selection of Optimum Feed and Speed in Finish Turning" Cirp Vol. 19 No. 4, Page 787, July, 1971

20. DeBarr A.E. "The Development of Numerical Control" The Production Engineer, Page 372, September, 1971

21. Dyke R.M. "Numerical Control" Prentice Hall 1967

22. Anon. "The LeBlond Cycle/360 Tape-Turn Lathe Programming System" LeBlond Incorporated

23. Anon. "Gelath Post Processor" Reference Manual. General Electric Information Services December, 1969

24. Anon. "Numerical Control Programming" Users Guide. General Electric Information Services December, 1968

25. Anon. "Cuts Programming" The Warner and Swasey Company 1970

26. Reichhold C. "Split----A Machine Dependent Programming Language" N/C World September, 1969

27. Anon. "TNC Combi Programming" Jones and Lamson 1970

28. Anon. "N/C Part Programming Systems for Jones and Lamson TNC Turning Equipment" Jones and Lamson

29. Beran D.C. "Current Status of N/C Programming Languages in the U.S.A." Technical Paper MS72-162 Society of Manufacturing Engineers Michigan 1972

30. Reichhold C. "Action----A Common Sense Programming Language" N/C World, Page 30, October, 1969

31. Robinson F. Miles J. — "Future Trends in Numerical Control" The Production Engineer, Page 449, September, 1968

32. Tipton H. — "Information and the Control of Machine Tools" The Production Engineer, Page 443, September, 1969

33. Rohs H.G. — "Numerical Control and its Application to Turning Operations" N/C Information No. 5 V.D.F.

34. Davies B. — "The Future of the Manufacturing System" The Production Engineer, Page 233, June, 1971

35. Braithwaite G.R. Hague A.G. — "A Statistical Analysis of Machining Variables" The Production Engineer, Page 389, July, 1970

36. Weill R. — "General Review of Numerical Control Languages in Western Europe" Technical Paper MS72-191, Society of Manufacturing Engineers, Michigan 1972

37. Bjorke D.O. — "Manufacturing System Software" Technical Paper MS72-187, Society of Manufacturing Engineers, Michigan, 1972

38. Budde W. — "Including Machining Technology and N/C Languages" Technical Paper MS72-192, Society of Manufacturing Engineers, Michigan 1972

39. Kerry T. — "The Geturn Part Programming Manual" General Electric Information Services January, 1972

40. Leslie W.H.P. — "Numerical Control Users Handbook" McGraw Hill 1970

41. DeBarr A.E. "Machining Data" The Production Engineer, Page 504, December, 1971

42. Anon. "Numerical Control Handbook" Honeywell 1969

43. Anon. "Remapt Part Programming" Reference Manual. General Electric Information Services October, 1968

44. Anon. "APT Part Programming Dictionary" Honeywell November, 1971

45. Anon. "APT Part Programming Encyclopedia" Honeywell October, 1971

46. Anon. "Compact Part Programming Manual" Manufacturing Data Systems Inc. 1971

47. Anon. "TNC Language. A Modular Programming System for Jones and Lamson TNC Lathes" Jones and Lamson.