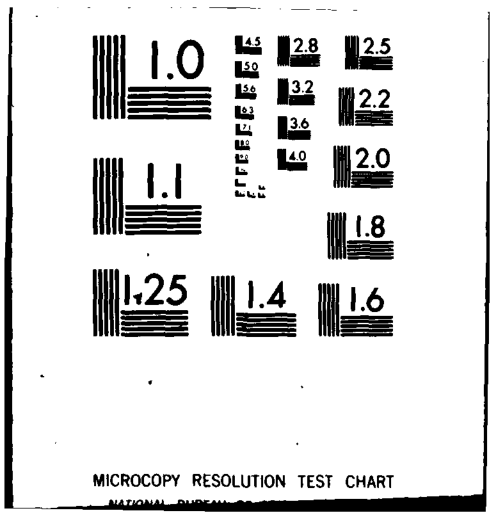
AD-A081 683 ADVANCED COMPUTER SYSTEMS INC DAYTON OH F/G 13/9
NUMERICAL CONTROL LATHE LANGUAGE STUDY.(U)
OCT 79 P D SENKIW, J HARRINGTON, J J CHILDS DAAB07-77-C-0185
UNCLASSIFIED CORADCOM-77-0185-F NL

1 of 2
AD
A081683



MICROCOPY RESOLUTION TEST CHART

NATIONAL BUREAU

4

RESEARCH AND DEVELOPMENT TECHNICAL REPORT
CORADCOM-77-0185-F

NUMERICAL CONTROL LATHE LANGUAGE STUDY

Peter D. Senkiw
James J. Childs
Joseph Harrington, Jr.
ADVANCED COMPUTER SYSTEMS, INC.
3131 South Dixie Drive
Dayton, OH 45439

David K. Ruppe
US ARMY CORADCOM

November 1979

Final Report for Period 17 Sep 77-30 Sep 79

A

CORADCOM
US ARMY COMMUNICATIONS RESEARCH & DEVELOPMENT COMMAND
FORT MONMOUTH, NEW JERSEY 07703

80 3 3 035

RESEARCH AND DEVELOPMENT TECHNICAL REPORT
CORADCOM- 77-0185-F

NUMERICAL CONTROL LATHE LANGUAGE STUDY

Peter D. Senkiw
James J. Childs
Joseph Harrington, Jr.
ADVANCED COMPUTER SYSTEMS, INC.
3131 South Dixie Drive
Dayton, OH 45439

David K. Ruppe
US ARMY CORADCOM

November 1979

Final Report for Period 17 Sep 77-30 Sep 79

A

CORADCOM
US ARMY COMMUNICATIONS RESEARCH & DEVELOPMENT COMMAND
FORT MONMOUTH, NEW JERSEY 07703

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19 REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER 18 CORADCOM-77-0185-F	2. GOVT ACCESSION NO.
	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) 6 NUMERICAL CONTROL LATHE LANGUAGE STUDY.	5. ~~TYPE OF REPORT & PERIOD COVERED~~ 9 Final rept. 17 Sep 77 ~~to~~ 30 Sep 79,
	6. ~~PERFORMING ORG. REPORT NUMBER~~
7. AUTHOR(s) 10 Peter D. Senkiw, Joseph Harrington, Jr., James J. Childs, David K. Ruppe ~~US Army CORADCOM~~	8. CONTRACT OR GRANT NUMBER(s) 15 DAA B07-77-C-0185 new
9. PERFORMING ORGANIZATION NAME AND ADDRESS Advanced Computer Systems, Inc. 3131 South Dixie Dr Dayton, OH 45439 New	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS MMET 2779679
11. CONTROLLING OFFICE NAME AND ADDRESS US Army Communications R&D Command DRDCO-SO-AM Ft. Monmouth, NJ 07703	12. REPORT DATE 11 Oct~~ober~~ 1979
	13. ~~NUMBER OF PAGES~~
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) 12 162	15. SECURITY CLASS. (of this report) UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Numerical Control, Programming Languages, Manufacturing Methods, Computer Aided Manufacturing.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An examination of fifteen numerically controlled lathe programming systems was conducted to characterize them qualitatively and quantitatively. The report presents a description of each of the fifteen voluntary participants' systems. The report: describes the non-technical characteristics of each system -- the business and operational characteristics such as hardware and software sources and costs, documentation, training, vendor support and maintenance; tabulates the capabilities of the languages for description of the geometrical configure-ations of the part being programmed, and the variety of the geometrical formats

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

411 643

accepted by each system as manuscript statements; discusses the use of macros to simplify the writing of programs to perform the common operations of all lathe work -- automatic roughing, finishing along a profile, threading, grooving and necking, drilling, boring, reaming and tapping; presents a brief discussion of the distinguishing characteristics of each system; describes the preparation of ten test parts for use in demonstrating the capabilities of the fifteen systems; describes the capabilities demonstrated by the fifteen systems to program the ten test parts; the amount of time required to write the program, and to debug it; it shows the success in processing and postprocessing the program, and the verification of the output tape. The evaluation was limited to NC lathe works; it did not cover milling machine, machining center, or punch press work. The systems evaluated are:

Cinturn II	Cincinnati Milacron, Inc.
QUICK-PATH	Digital Systems Corp.
Genesis	Encode, Inc.
GETURN	General Electric Co.
Ingersoll Lathe Prog.	Ingersoll Milling Machine Co.
COMPACT II	Manufacturing Data Systems, Inc.
TOOLPATH	Manufacturing Software and Services
APT for Lathes	McDonnell Douglas Automation Co.
GTL/T	Olivetti Corp. of America
APTURN	Structural Dynamics Research Corp.
VNC	Threshold Technology, Inc.
UNIAPT	United Computing Corporation
UCCAPT	University Computing Co.
PROMPT	Weber NC Systems
WESTURN	Westinghouse Electric Corp.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF TABLES AND MATRICES

FOREWORD

One of the key problems facing a user of numerical control equipment is the selection of a computer-assisted part programming system. There are many such systems offered today; they present a wide range of capabilities, limitations, and costs. A user new to numerical control, or even one just moving from manual to computer aided programming, has to match his situation and his needs to the characteristics of available systems. This is not a simple task. In an effort to assist those who must make a choice, this report presents for comparison, information on fifteen lathe programming systems.

This evaluation is a companion volume to the 1974 study "Numerical Control Language Evaluation,"* prepared under the aegis of the Numerical Control Society, by the authors of the current report. That earlier volume was prepared between October 1972 and March 1974 for the U.S. Army Electronics Command, Fort Monmouth, N. J., and by their direction it was limited to programming systems only for milling, drilling and boring machines and machining centers. Furthermore, by direction of the sponsor, only seven programming languages were studied. The report received wide circulation and has proved most useful, both in the Armed Services and in civilian industry. Many have expressed a desire for similar information on other machine tool applications, and on a wider variety of systems.

In 1977 the Electronics Command asked for a study of lathe programming languages, a request which has culminated in this report. It should be noted that this study is not merely a duplication of the earlier report, transposed into the lathe field. And for good reasons:

- Lathes and lathe work are quite different from machining center and milling work. Hence the languages, processors, and postprocessors are distinctly different, and produce a different set of decision parameters with which to deal.
- The lathe study deals with many more systems than did the 1974 report. Therefore the study method, the evaluation, and the data presentation are different.
- The whole concept of computer aided part programming has changed in the intervening five years, and its use is more widespread. While there are still many parts manually programmed, most of them are found in point-to-point and other simple work, such as for punches, drills, and milling machines.
- Computer technology has leapt forward in the past five years. The mini-computer is no longer uncommon, and microprocessors are appearing everywhere. The structure of the computers and the relationship of the programmer to his computer have changed.
- The rate of change, both in computer technology and in the science of NC programming, is very rapid.
- Finally, the experience of preparing the 1974 report has enabled us to prepare this report more efficiently.

* Research and Development Technical Report ECOM-0058-F

There is no one simple, universal answer to the question — "What is the best NC lathe programming system?" There probably is an answer to the question — "What is the best NC lathe programming system for the XYZ Corp. in the years 1979 to 198x?" But that one best choice depends on two environments — the internal demand and the external supply situation:

- Internally, what is the entire picture of the XYZ Corp. insofar as not only NC lathe work, but all NC machine tool work is concerned: what kinds of parts, how many of each, what NC tools are available, what computers are available, etc., etc.?
- Externally, what NC lathe and/or milling programming systems are available?

ONLY the XYZ Corp. can answer the first of these questions. This study can **ONLY** provide input to the second of these questions. It therefore cannot conceivably prescribe the one best choice. That decision lies with the individual user-to-be.

What this report does is to present an objective analysis of a large selection of systems, setting forth their several capabilities and limitations. It compares the claimed and the demonstrated performance of each. It discusses the significant criteria which may be considered in comparing one system with another.

CHAPTER I

INTRODUCTION

It is the purpose of this report to evaluate the capabilities of fifteen of the current languages and processors used in programming NC lathes, in order that a potential user may make the best choice of a system to meet his unique requirements. As a reference point, the history of NC lathe programming is presented, and contrasted with programming for milling machines, and the respective roles of languages, processors, and lathes are outlined.

A. Purpose

Industry has recognized the importance of a facility for preparing the input control data for numerically controlled machine tools, ever since the initial demonstration of such tools at the Servomechanism Laboratories of the Massachusetts Institute of Technology in 1952. It was only natural that workers turned to the computer for assistance in performing the many long geometrical calculations involved, and in processing the data as it moved through the various stages of preparation. The US Air Force sponsored the early efforts on NC programming methods, just as it had sponsored the development of the NC machine tools.

While the NC tool control-tape data can be derived manually from the drawing of the part to be produced, the process becomes incredibly complex and intolerably tedious even for moderately complicated parts. In fact, manual part programming is economical only on very simple work pieces — point-to-point work and two-dimensional straight line cutting. Computer aided programming is cost effective even for these parts, providing the trained programmer and the computer are present and available. If a plant has a mix of part designs, some of which require computer aided programming, it usually does all its programming on the computer.

Computer aided programming systems have proliferated because of their desirable characteristics — speed, versatility, reduction of human errors, and compatibility with the rest of the NC data processing. Each system has a language by which the manufacturing engineer communicates with the computer, and a program stored in the computer which converts the input (source) data into an output (control) tape. (When the control data gets to the machine tool, it is still further processed before being used.)

The creator of each system is free to select a syntax, vocabulary, and software technology to best suit his needs. He implements his system on a specific computer configuration, and he can select which portions of the processing (e.g., interpolation) are to be done in the computer and which in the controller at the NC machine. He may elect to use a dedicated computer, which he can afford to monopolize, or to use part of the time of a much larger, faster machine and share its cost. He takes as his target the entire marketplace and the full gamut of part complexity, or he can aim at a sub-set of the market which seems to him more lucrative.

Given these many options, innovative programmers, aggressive entrepreneurs, and the infinite variety of the circumstances which occur, it is not surprising that a great many NC languages have been developed and used. Some have survived the tests of time and the marketplace. Some were short lived. Some are broadly applicable, others are narrowly focussed on the needs of a specific type of operation, or a specific company's machinery. Some relied on a computer technology or a machine tool technology which has been superseded. Most of the current languages are vigorously merchandised by their proponents, each on its own merits.

It is the purpose of this report to evaluate the capabilities of fifteen of the current languages and processors used in programming NC lathes, in order that a potential user may make the best choice to meet his unique requirements. The resultant substantial economies will be based upon relevant factors impartially assessed.

Further, the evaluation indicates the trade-offs between languages for the benefit of those managers who must use other than their preferred language due to scheduling requirements, unavailability of computer facilities, limited man-power or skills, unavailable production equipment, or the form of existing engineering data. The evaluation thus makes economies possible by providing a basis for a "best alternate choice" in difficult engineering situations.

Since this evaluation is an operational comparison, direct tangible savings can accrue by providing to both industry and government, in a definitive, easily available reference format, information helpful in selecting the language most appropriate for the specific requirements of a given numerical control manufacturing environment.

B. Background

Numerical Control is defined as the method of control of production machinery by instructions pre-recorded in symbolic form. A numerically controlled machine tool is one on which servo devices and electronic control circuitry have replaced the hand controls, so that the motions of the cutting element and the operation of the auxiliary functions will respond to coded instructions on punched paper tape. The advance preparation of the coded instructions is called programming. In the case of simple parts, the coded instructions can be manually calculated and punched into the tape, but more complex parts require assistance from a computer to make the dimensional calculations. Computer aided part programming requires a vehicle of communication from the programmer to the computer; such media are called Numerical Control Programming Languages, or — more briefly — NC languages.

NC machine tools were first developed by the Servomechanism Laboratory of the Massachusetts Institute of Technology, under contract with the Air Materiel Command of the USAF. The first model tool was demonstrated in 1952, and it became immediately apparent that some form of computer assist would be necessary to prepare tapes if the NC system was to be practicable. An NC language was conceived by Mr. Douglas T. Ross in late 1952, and called APT, standing for Automatically Programmed Tools. It was first sponsored by the Air Materiel Command, but subsequent sponsorship and development was carried on by a committee of the Aircraft Industries Association. In the early 1960's the sponsorship was extended beyond the AIA, and a committee called APT Long Range Program (ALRP) Committee assumed responsibility. Project work was carried out by the Illinois Institute of Technology Research Institute (IITRI) under contract with the ALRP Committee.

In 1972 ALRP incorporated itself under the name Computer Aided Manufacturing — International, Inc. (CAM-I), and established an office in Arlington, Texas, to continue APT projects and to extend its sphere of interest into CAM.

APT was conceived as a generalized approach to the conversion of English-like statements describing the machining of a part into the machine operating codes suitable for control of the servomechanisms and control logic systems of an NC machine tool. No limits are placed on the complexity of the part design, and as a result APT was from the beginning a large program requiring a large computer. As large computers are not universally available, many organizations designed languages which were simpler to use or were more limited in their scope and hence could operate on smaller computers. Some were limited to a specific machine tool, or to tools of one machine tool manufacturer, or to one class of tools (e.g. lathes). Some were sub-sets of APT, others were unrelated to the APT syntax or grammar. At least forty such languages have been announced, although not all of them have survived.

In the late 1950's and early 1960's many NC tools were built with relatively limited capabilities compared to the original machines used in the aerospace industries. Many of the new machines were point-to-point devices, some could mill in straight lines, and a few could do continuous (curved) path cutting in the X Y plane. However, a large fraction of the total *number* of parts programmed fell within these limits. Consequently programming languages less extensive than APT were quite appropriate. As time passed these simple machines were enhanced to permit more complicated machining tasks — e.g., by the addition of rotary tables. Concurrently the programming languages evolved to keep pace, enhancing their powers by adding technical improvements — e.g., the ability to select the plane in which 2-D contouring was to be done, or the ability to rotate the work piece about two axes to position it for three axis machining, thus achieving five axis end results.

NC lathes were later in appearing in the trade than NC milling machines, possibly because the early thrust in NC was concentrated upon the needs of the aircraft industry for three, four, and five axis milling. Lathe work is essentially two axis machining. As the simpler versions of APT referred to above evolved — for example, systems suitable for two axis point-to-point or straight line milling — it was an easy step to apply these systems to lathe work also. As machine tools, NC lathes presented no more unusual problems insofar as the NC control systems are concerned than did two axis milling machine tables.

NC punches also appeared in significant numbers about the same time (1960). Here too, the first work required two-axis point-to-point controls, followed later by two-axis contour cuts made with a nibbling punch.

The APT language was not designed with lathe work in mind, and as a result the application of APT to turning work was cumbersome. The simpler languages were more readily adaptable, and as a result early lathe programming relied largely upon them.

APT has now been satisfactorily adapted to lathe work. As presented in American National Standards Institute (ANSI) Standard X3.37-1977, APT now has a nucleus language, with four modular features having syntax designed for a special application of the nucleus, such as Point-to-point, Drafting, 2D Complex Surfaces, and Input/Output. The 2D Complex Surfaces feature in turn has three feature modules: 3D Geometry, Lathe, and Extended 2D Geometry. The lathe module was introduced in Revision 2, proposed in February 1978.*

Another sub-committee of the ANSI/X3 committee is preparing a standard for languages other than APT, including COMPACT II. Tentative standards are expected to be submitted to ANSI in July, 1979.

C. Turning Contrasted to Milling

What is it that makes turning and milling basically different arts, and hence calls for different characteristics of the programming technologies?

Turning and milling are alike in that both are metal cutting technologies in which a sharp edged cutting tool moves across the surface of a part, shaping it by removing chips. But in other respects turning and milling are unlike, almost opposites.

- In milling, the cutting tool rotates rapidly and the work piece moves slowly relative to the tool to feed new areas to the cutting point.
- In turning, the work piece rotates rapidly while the cutting tool is fed slowly relative to it, to act upon new areas of the work piece.

Simple as is that concept, its significance to metal cutting technology and hence to NC programming technology is tremendous. It is appropriate therefore to study systems for NC lathe programming separately from other NC tool programming studies.

Parts designed to be produced on a lathe have one characteristic in common: they have one axis of symmetry. This need not be the case in milling machine work.

- In tools with rotating cutters (mills, drills, boring bars, saws, reamers, etc.) the cutting surface of the tools may be directed to any and all areas of the work piece, and be moved in two, three or even five axis motions, and hence produce a very wide variety of piece part configurations (all within the dimensional limitations and capabilities of the machine tool, of course). About 50% of all chip-making metal cutting machine tools fall within this classification.
- In tools with rotating work pieces (lathes), when the cutting point is engaged with the work it produces a cut all points of which are equidistant from the axis of rotation. As it is moved in two dimensions, it produces surfaces which, if cross-sectioned perpendicular to the axis, will always be circles. About 30% of all chip-making metal cutting machine tools fall within this classification.**

Because of the symmetry of surfaces produced in lathes, it follows that all sections which would be revealed by a cutting plane passing through the axis of symmetry will reveal an identical intersection profile with the surface of the part. (See Figure 1.) This intersection or profile is a line

* Obtainable from Computer and Business Equipment Manufacturers Assoc., 1828 L Street NW, Washington, D.C. 20036

**The remaining 20% of metal cutting machines comprise punches, broaches, band and hack saws, shapers, planers, and shears. Non-chip making but metal cutting tools include spark erosion, chemical milling, abrasive machining and flame and laser powered cutting tools.

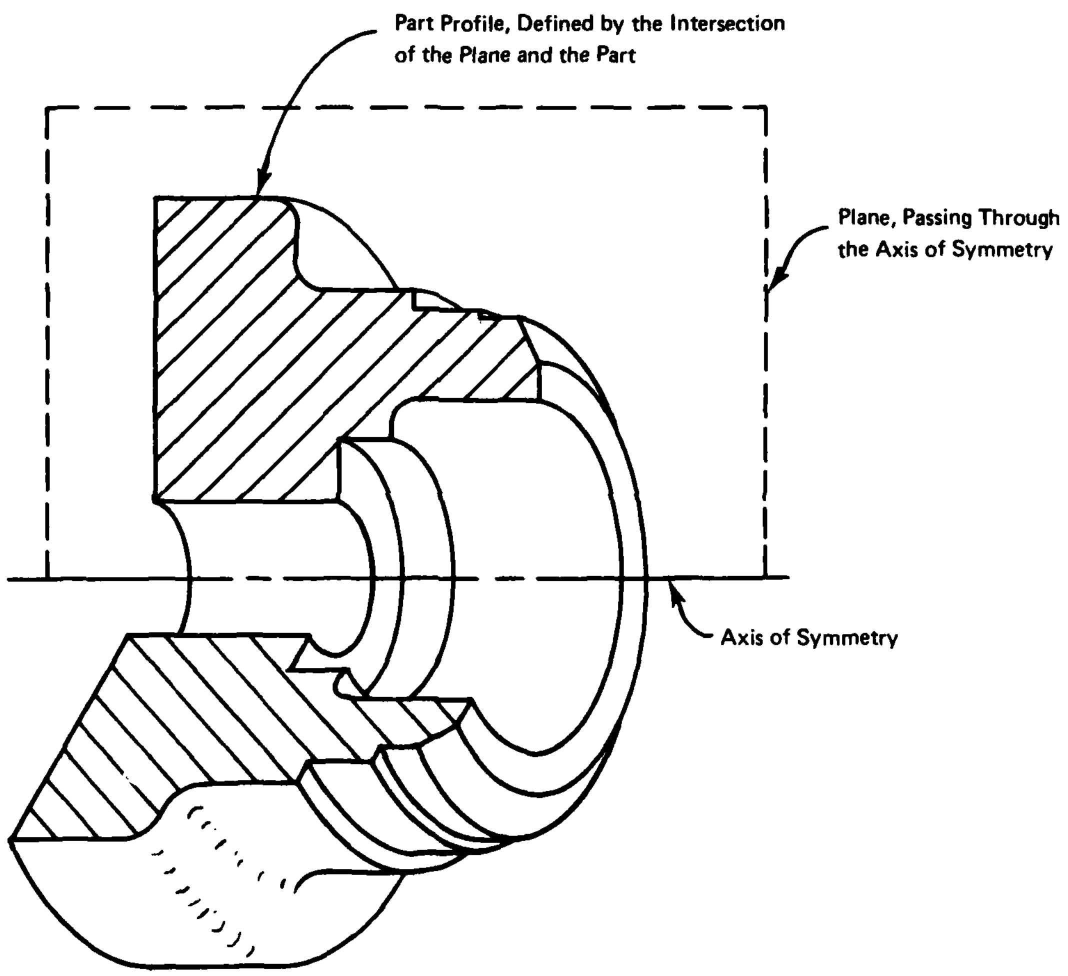

Part Profile, Defined by the Intersection of the Plane and the Part
Plane, Passing Through the Axis of Symmetry
Axis of Symmetry

FIGURE 1

The part profile is a connected series of straight and curvilinear line segments, lying in a plane. When the profile in its plane rotates about an axis lying in the plane, the profile sweeps out a surface in space known as a "surface of revolution."

composed of straight and curvilinear segments connected end to end. Mathematicians would say that such a line, when rotated about an axis would sweep through space and generate a "surface of revolution." Hence the entire surface to be produced in a lathe may be defined by defining a two-dimensional curve.

Not all two-dimensional curves may be used to generate surfaces of lathe parts. The curve must not cross the axis; it must not cross itself; and it must be continuous, among other restrictions.

As a result of this two-dimensional definition of lathe-created surfaces, NC lathe programming is a simpler technology than NC programming for machining centers. Also, automatic tool changing had been routine on lathes long before Wallace Brainard invented the automatic tool changer for machining centers; both chuckers and shaft lathes use rotating turrets for tool changing.

On the other hand, lathe work presents some problems not found in milling and drilling machines. The spindle speed of these latter machines will be the same RPM wherever the cutter engages the work piece. But a facing tool on a lathe may move from the axis to the outer edge of the work well away form the axis. This theoretically calls for a variable spindle speed during a single cut, something not found in milling work.

D. NC Languages, Processors, and Machines

The role of NC languages in manufacturing can be described by the following orientation: When the manufacturing engineer determines that a particular part is to be operated upon by an NC machine, it becomes necessary to convert the detail drawing into an NC tape, or its equivalent. Both the human mind and the digital computer play a part in this conversion, and the NC language is the medium through which the engineer transmits his intentions to the computer.

The logic sequence may take this form:

1. Comprehend the part design as specified in the drawing and the associated data.
2. Conceive the sequence of cuts which will produce the desired finished product from the proposed input materials.
3. Select an available NC machine capable of such an operation, and note its characteristics and limitations.
4. Select specific cutting tools from those available, and note their characteristics and dimensions.
5. Determine how and just where the input material is to be held on the machine tool table or chuck.
6. Determine the sequence of cutting operations which are to be made.
7. Determine the available feeds and speeds for the cuts which most closely approximate the optimum cutting technology for the given materials of the part and the cutters.

This substantially completes the creative engineering portion of the task. What remains is to convert this conceptualized operation into instructions comprehensible to the machine. The engineer then communicates his desires to the computer; he will:

8. Reduce the concept to a written record. The resulting manuscript must be in a language which both the engineer and the computer can understand and manipulate.
9. Transmit the manuscript to the computer. Usually this is accomplished by keyboard input, either through punched cards or tapes, or directly from the keyboard into the computer.*

The computer then executes the following steps:

10. Converts the "English-like" input statements which describes the desired end results into the machine tool commands in the code which the specific machine tool can accept and execute. This may include extensive numerical calculations to convert the described part geometry into cutter positions.
11. Outputs the final product as a series of commands for the machine tool. Usually this takes the form of a punched tape, but may be recorded as the "tape image" in some form of computer memory device.

This study is concerned with steps 8 and 10 above:

Reduce the conceptualized operation to a written record (step 8), and

Convert the input statements into machine tool commands (step 10).

The associated steps 9 and 11 are reasonably straightforward data handling procedures. It is clear that we are dealing with two different things: The NC language used in step 8 and the computer program used in step 10. We will refer to them as "the LANGUAGE" and "the PROCESSOR," and to their totality as "the SYSTEM."

Implicitly involved in this logic sequence are also the part design as considered in step 1, which forms the input to the system, and the NC machine tools, which use the output. Both ends of the sequence impose constraints upon our evaluation process. Therefore both must be considered in the evaluation process.

It should be noted that the product mix in each factory is unique: it may require NC only on lathes, or only on machining centers, or (what is more likely) on some proportion of both types of tools. The NC Programming Department will have to be capable of servicing whatever tools are installed.

There is no one language or system which is superior to all others in all situations. The part designs which are to be programmed, the NC tools which are to be employed, and the ability of the parts programmers must all be considered. The better they are matched, the more satisfactory will be the operation.

* In some very simple machining tasks the assistance of a computer is not needed. The manuscript is written in a code directly usable by the machine tool, and the typewriter/punch produces the machine tape. We are not concerned with this procedure.

CHAPTER II

HOW THIS REPORT WAS PREPARED

A description of the methods used in preparing this report, and the reasons for selecting those methods, will help the user to understand and benefit from the report.

A paramount objective of this report is impartiality in the comparison of the NC lathe programming systems. It is inevitable that any such comparison will show differences between the systems; indeed, that is the exact objective of the study. These differences must be most carefully stated. The methods therefore have been selected so that criteria are as objective, factual, significant, and supportable as possible. Subjective criteria have been kept to a minimum. All characterizations of a system have been checked for accuracy with its proponent.

Basically the method has been to collect and compare data relating first, to the claimed capabilities of a system, and second, to the demonstrated capabilities. The claimed technical capabilities are taken from documentation and test patterns furnished by the proponent. The demonstrated capabilities report the performance of the system in actual operation on a series of test parts. Also reported is the background information on each system, such as its name and description, the proponent's identification, availability of the language, processor and post-processor, history and extent of use of the system, costs, hardware requirements, and support services offered.

It will be immediately apparent that there are very clear differences between the systems, part of which stems from the fact that in some cases they were designed to be different. NC lathe work is an extremely broad and varied field, and the proponents have each selected that part of it, (or all of it) which seems most attractive to them. They have tailored their systems to these specific market domains. However, within one domain, several systems may be competing, and it is there that the differences between them, both claimed and demonstrated, become significant to the users in that domain.

A. Selection of Languages and Systems

The Communications Research and Development Command, U.S. Army, CORADCOM, announced the start of this evaluation and requested that candidate lathe programming languages should be nominated. The announcement appeared in metal working trade journals, and was repeated at a DOD meeting in Memphis, Tenn., on Nov. 16-18, 1977. The invitation was also repeated at various meetings of interested professional societies. The contractor prepared a list of approximately 29 systems, including all companies that were even remotely reported to have a lathe system available. Invitations to participate were mailed by the contractor to all. It was agreed, however, that no one should be pressured to participate.

Certain acceptability criteria were used: A candidate system should be commercially available; currently supported by the proponent; if proprietary, it should have users; it need not be in current use at a government installation.

CORADCOM reserved the right to add to or delete from the list proposed by the contractor, but has not felt it necessary to exercise this option.

Some of the invitees simply declined to participate. Others were unable to do so for reasons unrelated to the merits of their system. Some systems had been withdrawn from the market. Two proponents enrolled but were forced to withdraw later because for business reasons they felt that they could not devote the necessary effort to do a good job, and felt that no participation was better than a poor job. After lengthy negotiations, fifteen proponents and systems were enrolled; each agreed to participate and to cooperate with the contractor to the necessary extent. This included furnishing documentation, information, and the demonstration of their system on test patterns and test parts. We are grateful to all these people.

The list follows, arranged in alphabetical order of proponents' corporate names. All tabulations and matrices in this report will follow this sequence.

B. Collection of Information

The method of information collection can influence both the responses and the ease of utilization. NC lathe programming is a broad and varied field, and some care was required in phrasing questions so that they would conform to the requirements of both impartiality and applicability.

As mentioned above, the information collected could be divided into three general categories:

1. Business oriented: hardware, software, and operational characteristics.
2. Claimed technical capabilities.
3. Demonstrated capabilities.

All of the data collected have been recorded in the frame of reference of a potential user, rather than that of the vendor.

Questionnaires

Operational characteristics quickly divided the systems into two groups — those designed for local processing and those designed for remote processing. "Local processing" means in-house processing. The user company owns (or has on lease) the CPU of the necessary computer, plus the peripherals and the software. This does not preclude the possibility that the processing involves a long-line telephone link between one plant site and another. "Remote processing" means that the user company does not own or lease the CPU involved; another corporate entity performs the processing on their equipment as a service function. Remote processing usually involves long-line telephone links.

TABLE 1

LIST OF PARTICIPANTS

Serial Number	Name of Participant and Address	Name of System
1	Cincinnati Milacron, Inc., Machine Tool Group 4701 Marburg Avenue, Cincinnati, Ohio 45209	Cinturn II
2	Digital Systems Corporation 317 Monroeville Mall, Monroeville, Pennsylvania 15146	QUICK-PATH
3	Encode, Inc. Perkins Way Dexter Industrial Green, Newburyport, Massachusetts 01950	Genesis
4	General Electric Co., Information Services Business Division 401 N. Washington Street, Rockville, Maryland 20850	GETURN
5	Ingersoll Milling Machine Company Rockford, Illinois 61101	Ingersoll Lathe Program
6	Manufacturing Data Systems, Incorporated 4521 Plymouth Road, Ann Arbor, Michigan 48105	COMPACT II
7	Manufacturing Software & Services 6761 Bramble Avenue, Cincinnati, Ohio 45227	TOOLPATH
8	McDonnell Douglas Automation Company P.O. Box 516, St. Louis, Missouri 63166	APT for lathes
9	Olivetti Corporation of America 500 Park Avenue, New York, New York 10022	GTL/T
10	Structural Dynamics Research Corporation 5729 Dragon Way, Cincinnati, Ohio 45227	APTURN
11	Threshold Technology, Inc. 1829 Underwood Boulevard, Delran, New Jersey 08075	VNC
12	United Computing Corporation 22500 S. Avalon Boulevard, Carson, California 90745	UNIAPT
13	University Computing Company 1930 Hi Line Drive, Dallas, Texas 75247	UCCAPT
14	Weber N/C Systems 11611 West North Avenue, Milwaukee, Wisconsin 53226	PROMPT
15	Westinghouse Electric Corporation, Industry Systems Division 200 Beta Drive, Pittsburgh, Pennsylvania 15238	WESTURN

The requirements for hardware and software, and the bases for the accrual of costs of operation, are so different between these two groups of users that separate questionnaires were prepared to collect business oriented information. In a few cases, one system could be used both ways, and so two sets of questions were asked. Questionnaires were filled out by all the participating proponents. In so far as possible, the information has been summarized in matrix form in Chapter IV. Other information from the questionnaires which was not amenable to matrix presentation has been discussed in Chapter VII, Distinguishing Characteristics: unique features of the equipment, processor, system architecture, programming manual evaluation, and costs.

Documentation

Each proponent was asked to submit copies of his programmer's manual, supported in some cases by sales brochures. These documents were the source of data on claimed technical capabilities. It was here that some of the most striking differences between the systems appeared. Some of the manuals were complete, lucid, and effective teaching devices. Others were incomplete or virtually missing, confusing, and defied interpretation. It appears that a good manual is the result of a series of trial drafts, experience in use, revision, and republication.

The writing of a programmer's manual is a very large and difficult task, and perfection calls for professional editorial skills. There seemed to be an evolutionary pattern in the manuals. First editions seemed to be a transcript of the system designer's notebook — directed to the most straightforward programming tasks. Second editions showed the impact of experience with the diversity of NC lathe work; more and more variations had been provided. Later editions tended to be completely restructured, with evidence of professional editing and illustrating. The most mature versions covered so many variations, options and special situations, and had been so formalized, that they read like a legal or mathematical textbook.

Because the programmer's manual becomes the daily companion of the programmer, it is important that it be so written as to teach the user correct procedure, both in writing programs and in operating the computer or terminal. It must also do so in a simple, lucid, effective manner. It must be indexed so that a question leads directly and easily to the correct answer.

We have inevitably made comparative evaluations on this basis of the documentation submitted to us. Fancy binders, glossy paper and multicolored pictures were not taken into consideration. Logical structure, good indexes, clear diagrams, simple language and completeness were considered. This is the one *subjective* evaluation we have made. Because it is not quantifiable, we have used only three grades — Good, Adequate, and Inadequate. (See Chapter VII, and Chart 3.)

It should be noted that documentation is not a static affair. Several of the systems involved have published revised manuals during the course of this study. It is presumed that this will continue to be the case in any strongly continuing system.

Test Patterns

In the course of studying the programming manuals, the contractor became keenly aware of the differences in the approaches to writing manuscript statements. While each system followed a self-consistent syntax and vocabulary, presented in a sequence which fitted its processing concept, the fifteen sets of manuals were not easily comparable. It was therefore decided to make up a series of patterns, and ask each proponent to show how they would write their input statements for each.

Thirty-three test patterns were developed, together with a set of cutting technology values to be used when a statement format might require such information. A set of patterns was sent to each proponent, who returned them with the statements appropriate to their languages and processors.

A discrepancy immediately appeared, arising from the differences in the fundamental structures of the systems. To illustrate:

> Pattern 1 asked for the part profile definition statement for a profile consisting solely of straight lines. Pattern 7 asked for the statement for a roughing cut to the same profile. Pattern 15 asked for the statement for a finishing cut along the same profile.

In some APT-based languages, this elicited three different responses. In other languages, there is no separation of profile definition and tool movement. In some others, "statements" per se are not used. Consequently the responses were not always comparable.

Nevertheless, the completed test patterns shed considerable light on the workings of the system, the number and complexity of the source code inputs, and in some cases supplied information which could not be found spelled out in the programmer's manuals. In addition, the test patterns showed capabilities in handling the several lathe operations such as roughing, threading, grooving, etc., which many systems can handle as sub-routines or macros. Thus all but a few of these special features could be eliminated from the final test part programming exercise.

C. Claimed Technical Capabilities

In order to compare the relative technical capabilities of the systems, it was necessary to do two things: First, completely understand each system, just as if we were users-to-be. Second, characterize the system in a uniformly structured summary, so that all systems could be compared.

The fifteen systems were divided up between the three principal investigators, and each system was carefully studied. For each, the questionnaire responses, the programming manuals, and the test patterns were examined. A set of facts was abstracted, covering:

- A description of the system — program structure, computer support, lathe work sub-sets, etc.
- Part description capability — axis nomenclature, available geometric definitions, fillets and chamfers, etc.
- Special lathe routines or macros — roughing, finishing, threading, grooving, drilling and tapping.
- Tool control procedures and technology of metal cutting.

The summary was reviewed with the proponent to verify our understanding of the system: corrections were made as needed. This served to assure the proponents that the investigator had understood the system.

The three investigators met repeatedly to compare findings, and to structure the evolving form of the report. As the work progressed, the points of similarity and the points of difference in the systems emerged. It is necessary that these be presented uniformly and impartially, and yet display all the special features which had been designed into some of the systems.

The fifteen systems are indeed widely divergent in their objectives, their languages, and their procedures. This we believe to be a desirable situation and worthy of note, as almost any sort of local programming need may thereby be served by at least one of the systems.

What was learned has been incorporated in four of the chapters which follow in this report:

Chapter IV	Business and Operational Information
Chapter V	Basic Geometric Capabilities
Chapter VI	Lathe Programming Modules
Chapter VII	Distinguishing Characteristics of the Systems

The first three of these four chapters presents information in matrix format for easy comprehension and comparison of the systems. The last of these four chapters presents in narrative form those characteristics which distinguish one system from another; this latter body of information would not fit into the highly structured format of the matrices of the prior chapters.

D. Demonstrated Capabilities

"The proof of the pudding is in the eating of it." Having comprehended the claimed capabilities of the systems, the contractor next turned to demonstrations of the systems in action. Ten test parts were designed and each proponent was asked to program these parts and verify the tapes produced.

The parts are based upon common industrial and governmental lathe part designs. Several hundred actual part drawings were collected, examined and classified. From these drawings, ten test parts were synthesized for this study. Whereas the previous evaluation study, ECOM 0058-F, involved programming by both system proponents and users, this contract asks only that test parts be programmed by the system proponents. We expected, and found, that they assigned their most expert programmers to the task. This has minimized the human element of programming, and emphasized the effect of language structure and processor design. Further to minimize the role in the programming of the individual man, each test programmer was supplied with detailed method sheets, tool descriptions and layouts, and illustrations of the operations.

The ten test parts may be briefly characterized as follows:

Part	Name	Description
1	Housing	Turning, boring, chamfering, facing and simple threading. All profile segments are straight lines or circles.
2	Punch	Outside turning and tapers, blended by circular arcs.
3	Cap	Boring and turning of face profiles, defined by tangent circles or circles and tapers.
4	Case	Tabulated curve definition of turned surface.

5	Reflector	Mathematically defined curve on face.
6	Connector	Undercuts, internal and external.
7	Body	Grooving, four types.
8	Fitting	Threading: straight, tapered, and multi-start.
9	Scroll	Threading on face.
10	Post	Family of parts programming exercise.

Drawings of the ten test parts, and a complete set of method sheets, tool layouts, and illustrations of the operations for Part 1, are included in Chapter IX, page 128. The rationale for part design is also discussed there, together with the method of testing and the test results. We thank the government's manufacturing agencies and the industrial concerns who submitted sample part drawings from which portions were excerpted to create the test parts.

E. Presentation of Findings

It must be obvious to even the casual reader that the findings of such a study as this cannot be reduced to a single number — a sort of "figure of merit" — to be derived for each system studied. There are many measurement parameters, and the relative weight given to each must be determined by the user after examining his own unique needs.

Under such circumstances, the matrix of detailed findings seems to be the best, if not the only way to present the results of the study. The reader will therefore find many matrices in this report. With each there is an explanation of what aspect of lathe programming is being addressed, how the various parameters were selected, and the significance of the entries. The reader should study with care these explanatory sections before trying to extract information from the matrices. From time to time even the matrix format failed to reduce all the data to an orderly array of rows and columns. In these cases, footnotes have been added as necessary.

The matrix format also contributes significantly to the easy use of the findings, as will be discussed in the next section.

CHAPTER III

HOW TO USE THIS REPORT

Selecting an NC lathe programming language is a two part process. A systematic, comprehensive and accurate analysis of the user's unique requirements is half of the selection task. This Chapter outlines a method to help a potential user define his requirements, and match them with the characteristics of available systems. The other half of the task is the characterization of the available systems; it is the subject of the Chapters that follow this one.

It is the purpose of this report to assist a potential user of lathe programming systems to make the best choice to meet his unique requirements. The choice depends on two environments — the internal demand and the external supply. These two environments should be carefully assessed, and the best possible match found.

While there is a limited number of commercially available answers to the supply side of the question, there is only one internal, or demand situation, for each potential user — his own. We can and do list information concerning the available systems, but because of the almost infinite variety of demand situations, we can only suggest how each user can make his own internal assessment. We suggest a method which will help to make the task thorough, comprehensive, and honest. Above all the self analysis must accurately portray the facts as they are; let the chips fall where they may.

The Manager of Manufacturing Engineering or one of his people will probably make the evaluation, for three reasons: He should have, or have access to, the necessary past, current and future data which will be needed. He should be well equipped to appreciate the significance of the many factors to be considered. And his department will be the one on which the programming task will fall, and hence he will be responsible for its effective execution.

The need to make a selection of an NC lathe programming language may arise in several ways:

- The plant may have no NC equipment and no programming system, but is preparing to make its first venture into the field.
- The plant may have NC lathe equipment which has been programmed manually and feel that a computer aided system is in order.
- The plant may have NC machining centers, with a manual or a computer aided programming system in operation and fully occupied; and it may be in the process of adding enough NC lathes to justify a computer aided lathe programming system *separate from* the system for the machining centers.

- The plant may have an NC programming system installed and serving its lathe department, but be planning a significant change in the number or complexity of its lathe parts, or a significant change in the volume of programming work.
- The plant may have an installed NC programming system which is not satisfactory for any of several reasons, and replacement is contemplated.

A programming system should be selected to serve a specific plant for a period of time stretching several years into the future. It should be selected to deal with the spectrum of part designs used, or *to be used* in the plant, processed by computers installed or now accessible, or *to be installed or accessed*, in preparation for part manufacture on NC machine tools available or *planned for future purchase*.

There is a reasonably straightforward procedure to be followed in assessing the internal, or demand, environment. In other words, to answer the question, "What will we need for an NC lathe programming system?" The following sections of this chapter describe this methodology.

A. Internal Analysis

Product Mix

The first step in the analysis is the determination of the product mix — the ratio of parts turned on a lathe, to those shaped in milling, drilling, or boring machines, to those made in a sheet metal punch, to all the other parts. Some parts require operations on both mills and lathes, but usually one characteristic predominates and should govern the parts classification.

The product mix is easily determined in plants in which some system of classification and coding (group technology) is in use. Lacking that facility, a sample group of drawings may be very quickly sorted into four stacks — lathe parts, milled parts, punched sheet metal parts, and others. The latter category covers purchased components, hardware, piping, springs, etc. When in doubt, the operations sheet for the part will be helpful.

The next sort of the first three piles of drawings segregates the parts to which NC technology is economically applicable. For example, a part now made on an automatic screw machine or a multiple spindle chucker is probably not a candidate for NC turning by reason of the implied volume of production. On the other hand, a part now made on a tracer lathe may be an excellent candidate. Current practice, as listed on the operation sheet is an indicator of past practice, but not necessarily of the desired future practice. The operation sheet would certainly not be a guide in plants with no NC lathes.

This step in the analysis need not be carried out with mathematical precision. It is merely intended to show whether NC candidate parts are mostly lathe, mostly mill, or a mixture, and whether or not these ratios may be expected to hold in the future. In many instances it may be a ratio of which the Manager of Manufacturing is amply well aware without a survey.

It will be assumed that, for the purpose of this discussion, a significant number of NC lathe candidate parts are found.

The end result should be a simple matrix, expressed in terms of the percent of total part *designs. (Not manufacturing volume.)*

CHAPTER III

HOW TO USE THIS REPORT

Selecting an NC lathe programming language is a two part process. A systematic, comprehensive and accurate analysis of the user's unique requirements is half of the selection task. This Chapter outlines a method to help a potential user define his requirements, and match them with the characteristics of available systems. The other half of the task is the characterization of the available systems; it is the subject of the Chapters that follow this one.

It is the purpose of this report to assist a potential user of lathe programming systems to make the best choice to meet his unique requirements. The choice depends on two environments — the internal demand and the external supply. These two environments should be carefully assessed, and the best possible match found.

While there is a limited number of commercially available answers to the supply side of the question, there is only one internal, or demand situation, for each potential user — his own. We can and do list information concerning the available systems, but because of the almost infinite variety of demand situations, we can only suggest how each user can make his own internal assessment. We suggest a method which will help to make the task thorough, comprehensive, and honest. Above all the self analysis must accurately portray the facts as they are; let the chips fall where they may.

The Manager of Manufacturing Engineering or one of his people will probably make the evaluation, for three reasons: He should have, or have access to, the necessary past, current and future data which will be needed. He should be well equipped to appreciate the significance of the many factors to be considered. And his department will be the one on which the programming task will fall, and hence he will be responsible for its effective execution.

The need to make a selection of an NC lathe programming language may arise in several ways:

- The plant may have no NC equipment and no programming system, but is preparing to make its first venture into the field.
- The plant may have NC lathe equipment which has been programmed manually and feel that a computer aided system is in order.
- The plant may have NC machining centers, with a manual or a computer aided programming system in operation and fully occupied; and it may be in the process of adding enough NC lathes to justify a computer aided lathe programming system *separate from* the system for the machining centers.

PRODUCT MIX

Type	% of Total	% Amenable to NC
Lathe parts Mill/drill/bore parts Sheet metal punched parts Other parts		

Volume Forecast

The second step in the analysis is to obtain a five year forecast of volumes of NC work. There are two points of interest in volume. *First* is in manufacturing volume, which determines the overall scale of plant activity. Also, the larger the usage of parts, the more likely there are to be changes in design — the so-called Engineering Change Notices. Even minor changes may require re-programming to some extent. Further, large and continued NC production volume justifies substantial effort in program optimization. *Second* is in the level of developmental activity. The number of new products or new models determines the volume of new part programming to be performed.

These volume forecasts should cover the foreseeable future, which realistically means about five years. The Marketing Division may be called upon for probable production volume trends; *the Research and Development Division may be called upon for probable new product in*troduction rates. Normally this type of information is regularly supplied to the Manager of Manufacturing for his long range work planning.

The five year (approximate) figure is selected because most plants regularly make such forecasts of market volume. New products intended to reach the market within five years are probably already in planning if not in advanced stages of design. Acquisitions of NC tools or computer equipment, and of skilled programmers, is also a multi-year task, so that if new technology is ordered immediately it will not be fully operational until well into the five year period. Another term may be substituted for five years if it is better adapted to local forecasting procedures.

The end result of the volume forecast, combined with the product mix forecast, will give an indication of the number of parts which must be programmed per year, or per month, and the trend of that number over the forecast period.

Volume Forecast

	5 years ago	Last year	This year	5 years ahead
New part designs New parts to be programmed Old parts to be reprogrammed				

It is now necessary to convert the number of parts to be programmed into approximate hours of programming time. Take a representative sample of your lathe parts, and examine a print of each. Sort them into several categories of programming difficulty, expressed in hours to get a usable tape. Ten categories should be ample, say from one hour to ten hours work. Programmer's time requirements may be estimated for the several categories for any one of the fifteen systems surveyed in this report by reference to the time studies discussed in Chapter X. Note that the same part may require a different amount of time to program in the different types of systems. Note also that these times are those of *expert programmers*, and adjust accordingly. You can now determine the percentage of parts which are one hour jobs, two hour jobs, etc. From this will come an estimated total of programming hours. It may be checked against recent actual time usage, *if* you are using a similar system. You now have a profile of the number of parts to be done versus their complexity, and by multiplication, the total number of programming hours per year to be expected, for any selected programming system.

Equipment Forecast

The third step in the analysis is to list the NC equipment in your plant, and the known planned acquisitions. This includes both NC machine tools and their adjuncts, and computers and their peripherals. This listing should cover a five year forward period, although the result of your assessment may justify the addition of more or different computers, or more machine tools.

Include in the assessment the age of the tools. Tools lose productivity as they age, and while they may be overhauled or retrofitted to restore the level of efficiency, there is an inevitable cumulative degradation of output. Major maintenance overhauls may take a machine out of production for weeks or even months. The same loss of effectiveness is true in aging computers. What you need to know is the productivity available in tools and computers. There is no point in programming parts if there are no tools to make them. The Plant Engineering Department may be called upon for these long range plans.

On the other side of the coin is the possibility of acquiring new machine tools with capabilities for work for which the programming system is unprepared. For example, a new lathe may be a four axis machine, capable of moving two tools in two axes each, simultaneously. A very special processor and post processor is required to handle such a lathe.

The end result of the equipment forecast, combined with the part programming forecast, will give an indication of when the computer system will reach its capacity and will have to be augmented. Or, alternatively, when the volume of remote processing of programs will justify having an in-house computing facility.

Part Character Forecast

The final step in the internal analysis is consideration of the evolution of part design complexity. It is common to find that as NC manufacture continues, more sophisticated parts are designed for NC, and a greater number of parts in a product are intentionally designed to take advantage of NC production. The result is a steady rise in the programming load, even though the volume forecast may show no growth.

To meet this trend, NC lathes are being improved steadily, with design features such as the simultaneous four axis feature mentioned above. These changes in turn call for more sophisticated programming systems.

In Chapter VI there are listed five lathe routines — automatic roughing, profiling, threading, grooving, and drilling — which comprise a very large fraction of all NC lathe work. Each is discussed in a section, concluding with a matrix listing the range of variations commonly encountered. The matrix questions may be copied as work sheets. Take a generous and representative sample of your lathe parts, and examine a print of each. Make a tick mark opposite each characteristic found on each print, and segregate all prints containing machined surfaces *not* covered by these matrices. Total the tick marks to get a profile of your lathe parts.

- Note those characteristics in the list with zero tick marks
- Note those characteristics with tick marks in less than 5% of the parts.
- Note those characteristics with tick marks in over 5% of the parts.

Next take the prints containing machining requirements which *could not* be covered by the five types of macros. Make a list of the types of these surfaces, and count the occurrences of each type. Copies of the surface definitions matrices found in Chapter V will be helpful.

Confer with the Product Design Department to determine which of the macro characteristics with less than 5% frequency, and which of the surface definition characteristics not covered by macros are:

- essential, and must be accommodated,
- could be eliminated in future designs at little effort or inconvenience.

Also inquire what the trend in the next five years in each of these categories will be. For example, tapered threads are common in oil country applications, but are rare elsewhere. If your plant does not make heavy pipe joints or oil country machinery, there is no point in having programming capability for such threads. By eliminating that feature from a shopping list, some systems might be considered which would otherwise be excluded from consideration.

Also inquire whether the elimination of some particular characteristic would change the requirements for NC tools, or conversely, if new NC tools to be purchased will come equipped with machining capabilities not covered by a candidate programming system.

You now have a part characteristic profile. It may indicate a need to modify some of the estimates of the volume of programming, the schedule for new computer acquisitions, and the skills needed for programming. Indeed, any such analysis as that outlined above will inevitably require one or more reiterations before it can be said to be finished.

B. Programmers, and Programming Languages

NC programming languages are what are known as "application oriented" computer languages. They are both man-readable and machine-readable, but more importantly, they must be man-written. Their vocabulary, syntax, and structure is designed to describe the function they are to deal with, rather than the way the computer is to deal with it. Because of the special characteristics of lathe work (see Chapter I, page 6), a less extensive and therefore simpler programming language may be used for lathe work than for milling work.

A special language will contain no more words than are necessary to describe the operations for which it was designed. The simpler the operations to be programmed, the simpler and smaller the processor will be. And the simpler and smaller the processor, the smaller the computer required; or, the faster a program will run in a given computer. This is particularly true in systems dedicated to lathe work.

It should be noted here that some systems can operate with two or three levels of a language. Simple lathe parts may be programmed using a subset of the language of limited scope; more complex parts may be programmed using a larger subset of the language, with a more complex syntax; the most complicated parts may require the full capabilities of the parent language. The system's processor will accept input on any level, and the levels can usually be mixed together in one program. If, for example, the lower level language will describe all of a part except one feature, the programmer can describe that one feature by inserting statements written in the higher level of the language.

There are attractive advantages in a language consciously devised with limited capacity, and several of the systems in this study have such languages. There are also some disadvantages, as will be seen. The survey of part characteristics mentioned above will show which language limitations can be comfortably accepted. For example, does the plant ever cut spiral grooves on a face surface? variable pitch threads? hyperbolically curved surfaces? If not, then there is no need for such descriptors in the language or algorithms in the processor. Note, however, that it may be difficult to enhance a limited capability language should the plant's part characteristics become more complex, unless, as mentioned above, there is a higher level of the same language to which the programmer can escalate. Not all languages have this feature.

The total product mix will also affect the type of language selected. The NC programming department will have to program whatever parts are designed. Generally it is desirable to have as few different NC systems as possible in use in any one plant at any one time. This is to avoid confusion, reduce computer hardware and software, and increase the flexibility of part programmers' work assignments. There are three possibilities:

- One system, capable of handling all NC tools in the plant.
- Two systems, one for lathes and one for machining centers, etc.
- One computerized system for the dominant type of tool installed, lathes or mills, plus manual programming for the few remaining tools if the computerized system will not handle them too.

The choice may be clearly dictated by the product mix, or may have to be resolved by an economic balance.

Writing NC manuscripts requires much greater precision in syntax and grammar than does writing text material in English. Most languages require strict adherence to the rules, or the computer will either reject the input or (worse) misinterpret it and produce a useless tape. Nevertheless, programmers become very nimble in writing in their chosen language.

If one asks a programmer to write in two languages — to become bilingual, so to speak — it puts a greater burden on the man and decreases his efficiency. The more closely the languages resemble one another, the greater the chances of stumbling into an error in the detailed differences. NC processors are unforgiving of errors! It is advisable not to ask one man to switch back and forth between two programming systems. If a plant has two systems, and enough work to keep staffs for both busy, well and good. But balancing the work load may not be easy.

A single language in a plant is desirable.

The ability of a programmer to produce efficiently will depend on two skill factors:

- How much experience he has had in writing programs in a given language, and
- His native ability to visualize three dimensional machining processes from a two dimensional drawing.

Time will enhance the first skill, but conceptualization is a skill one either has or does not have. Fortunately the characteristics of the work in a plant varies from simple to complex, just as the capabilities of men vary from marginally useful to brilliantly capable. If the volume of work justifies only one programmer, he must be capable of handling any part that is designed in the plant. But if the volume of work is large, efficient use may be made of the usual spread of talent available.

C. Selection Procedure

Having completed the internal analysis describing:

- product mix: lathe/mill/punch/other parts,
- volume forecast: new parts programming and support reprogramming,
- equipment forecast: computers and NC tools,
- part characteristic forecast: complexity trend,
- and having assessed the options in language choices,
- full range vs. limited application,
- one, two, or more languages,
- programmer skill levels available.

you are now in a position to match this internal (demand) environment with the external (supply) environment.

The first question is which systems have the technical capabilities to match your five-year characteristic profile. (See page 23.) As suggested previously, some rarely used part characteristics may be taken out of the profile if their transfer to an expert manual programmer, or to a service organization, would yield an economic advantage. Match the work sheets as previously filled out (see page 22) to the matrices in Chapter VI and then in Chapter V to find which systems are "possibles"; i.e., have the necessary technical capabilities.

The next step is to estimate the cost of the "possible" alternatives. The programming volume forecast (see page 21) will tell roughly how many programmers will be needed, using the test part times in Chapter X as a guide for each of the possible systems. This in turn suggests the number of terminals that will be needed for remote processing or local shared time-processing, or the number of stand-alone units needed for dedicated processing. One- or two-shift operation may be considered. Using the check lists of cost elements at the end of this chapter, and the data in Chapter VII, a *preliminary* cost for acquiring and using each of the different "possibles" may be estimated and the several costs compared.

It is probable that this exercise will give a clear indication of what kind of a system is needed, and which system should be considered further. At least the list of "possibles" should be cut down to two or three.

At this point, interviews with the system vendors are in order. While the data in this report are as accurate as they could be made at the cut-off time for printing (June, 1979) they may well have changed before they are put to use by a potential user. NC programming systems are in a constant state of improvement and enhancement. All the characteristics and capabilities cited in this report *should be checked* at the time of selection, and if necessary, changes should be made in your work sheets. The same caution applies doubly to cost figures. Prices are subject to constant change, and there is no guarantee that the prices cited in this report will be in effect at any future time. *All costs should be checked* with the vendors for the specific configuration and usage you contemplate.

It is suggested that benchmark tests be conducted, using a few typical parts and one or two typical parts of maximum complexity, drawn from your own sample of parts. Ask each vendor to write a source program and prepare a tape for each, giving you the time to program the parts, and the cost of the tapes (if done on a time sharing system). Compare the performance of the various vendors.

Selection of a programming system is a relatively long range commitment, and you will be entering a close association with the system vendor. Examine his reputation by checking with other users. Ask for his list of users, and visit one or two of them. See what kind of parts they are programming, and what their experience has been. Ask the vendor how long he has had his system running successfully, and how many people he has to offer service and consultation. Examine the quality of his documentation, and his training. What maintenance and/or back up service does he provide? How are post processors provided to you? In short, decide whether or not you can live with comfort and security in affiliation with this vendor.

With this type of selection study, the mysteries of choosing a system will be somewhat cleared away, and the hazards of selecting a system will be reduced. The ultimate choice will stand partly on objective, and partly on subjective factors. And the ultimate choice MUST be made by the potential user.

TABLE 2

CHECK LIST OF COST ELEMENTS

A. LOCAL PROCESSING SYSTEMS

Host computer:	purchase or lease if used only for NC programming
Host computer:	use charges if host is operated by another division
Host computer:	variations in charges for time of day and turnaround time
Terminals:	purchase or lease
Terminals:	connection cost if terminal is remote from host
Plotter(s):	purchase or lease
Additional memory units and memory controller	
Printers, punches, readers, hard copy units:	purchase or lease
Supplies:	cards, tape, paper
Maintenance of hardware:	terms, and warranty period
Software, operating and communication systems:	initial and service charges
Software, processors of all sorts:	initial terms, maintenance, service
Documentation costs	
Training costs per pupil; special courses	
Consulting services, cost and arrangements	
Programmers' salaries plus fringes	
Supervision and overhead	
Office space, environmental controls	
Stabilized power supply	
Telephone charges for calls to vendor for advice and help	
Hardware maintenance technician costs	

B. REMOTE PROCESSING SYSTEMS

Connect time charges to host computer	
CPU use charges	
Variations in the above with time of day	
Storage charges for postprocessors, etc.	
License fee for system	
Telephone tolls to remote site	
Terminals:	purchase or lease
Plotters:	purchase or lease
Printers, punches, readers, hard copy units:	purchase or lease

TABLE 2 (Continued)

B. REMOTE PROCESSING SYSTEMS (Cont'd.)

Maintenance of terminals and peripherals	
Documentation costs	
Training cost per pupil; special courses	
Consulting services, cost and arrangements	
Postprocessor costs and maintenance charges	
Programmers' salaries plus fringes	
Supervision and overhead	
Office space, environmental controls	
Supplies:	cards, tape, paper
Security for proprietary information	

NOTE: Hardware may be purchased, leased, or rented, through the vendor or through a third party.

CHAPTER IV

BUSINESS AND OPERATIONAL CHARACTERISTICS OF THE SYSTEMS

Two types of information concerning NC lathe programming systems are of equal interest to a potential user. This chapter discusses the non-technical characteristics of the systems studied — the business and operational characteristics such as hardware and software sources and costs, documentation, training, vendor support, and maintenance.

In selecting a programming system for NC lathe work, a potential user will explore two almost equally important sets of factors. One deals with the technical capabilities of the system — what it can do. The other deals with the business and operational characteristics of the system — how does it work, with what hardware/software/human skills, and at what costs.

This chapter deals with the business and operational characteristics. The data herein were collected from the system vendors by a rather extensive questionnaire which all the vendors answered. Two versions were used, one for remote processing systems and one for local processing systems. Some vendors offer both types of systems, and hence filled out two questionnaires.

Some questions were answered by terse, straightforward factual answers, while others brought forth rather extended dissertations which respondents felt necessary to properly characterize their systems. Other respondents answered only some of the questions, apparently finding the rest not applicable. Inevitably, the answers were oriented to the respondent's system and phrased in his nomenclature. Nevertheless, much of the mass of information could be boiled down and collated in matrix form, and is presented in following pages. That which did not lend itself to this format has been discussed in Chapter VII on Distinguishing Characteristics.

Some definitions of terms used are in order at this point.

Local processing means in-house processing. The user company owns or has on lease the CPU of the necessary computer equipment, plus the peripherals and the software. The term does not preclude the possibility that the processing may involve a long-line telephone link between one plant site and another within the user company.

Remote processing means that the user company does not own or lease the computer CPU involved. Another corporate entity performs the processing on their computer equipment as a service function. Remote processing almost inevitably implies long-line telecommunications links.

Processor means the software which 1) translates the applications-oriented source data, written in the system language (e.g. APT) into computation-oriented statements written in a language understood by the computer (e.g. FORTRAN, COBOL); and 2) performs the geometric calculations required and generates the cutter location (CL) file data necessary to control a cutting tool relative to the work piece.

Postprocessor means the software which translates the CL file data into machine tool control commands for a specific make and model of lathe and make and model of controller. It outputs the control data as a perforated tape, or its equivalent image in a computer memory.

Preprocessor means the software which translates input data in whatever form it is available into a format acceptable to the system processor. Input data may be either written manuscripts, computer images from a CAD system, or voice commands.

Batch means a mode of operation in which a complete unit of input such as a part program is committed to the computer for processing at one time. The task enters a queue of other tasks; when it is taken up and executed it is returned to the user as a unitary output. Processing may be voluntarily delayed by the user to some low-cost time period such as night time, with delivery the following morning.

Time-shared means a mode of operation in which several users are concurrently delivering to the system individual instruction statements. These mini-tasks enter a queue and are taken up in sequence, executed, and returned to the user. Because the speed of the CPU is so great relative to the time required for transmission, the user perceives the response as an immediate reply, in spite of the fact that his work has taken its turn in a queue with the work of many other users.

Dedicated means the right to the sole use of a computer by the programmer. Whether he commits his input one statement at a time or as a batch, there is no waiting in a queue.

Conversational means a mode of operation (time-shared or dedicated) in which the programmer enters his program one statement at a time, and sees the processed output before he enters the next statement. The computer may issue error statements or diagnostics if necessary instead of a processed reply, so that corrections may be made before proceeding to the next statement in the program.

Prompted means a mode of operation in which the computer asks the programmer for the next input statement, frequently in a format which makes the programmer's reply merely a matter of entering parameters rather than conventional syntactical statements.

None of the systems required special environmental conditions. Power requirements were those normally found in offices — 115v AC 60 Hz, with voltage stabilizers or dedicated power service requested in some cases. If third party equipment is used, its environmental conditions govern.

Abbreviations commonly used in the matrix are:

ASR33	Teletype terminal	HP	Hewlett-Packard
CRT	Cathode ray terminal	Tek,	Tektronix Corp.
DEC	Digital Equipment Corp.	TI	Texas Instrument Corp.
DG	Data General Corp.	XDS	Xerox Data Systems
GE	General Electric Co.	IBM	International Business Machines

TABLE 3
BUSINESS AND OPERATIONAL INFORMATION
ALL 15 SYSTEMS
General Information About the 15 Companies and System Characterization
Documentation Evaluation

	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Computing	University Computing	Weber N/C	Westinghouse
General Information															
How long has the company been involved in NC programming service or product business?	24	7	4	12	8	10	7	18	10	5	3	9	12	2	10
How many employees are assigned to *these activities?*	3	3	20	many	4	482	20	100	50	3	6	40	45	11	12
How many NC programming service or product customers have you?	105	35	200+	200	2	2261	200	150	100 U.S. 700 Worldwide	15	15	250 Worldwide	275	23	100
System Characterization															
For LOCAL processing:															
Batch:	√		√		√	√	√		√		√	√	√	√	
Time-shared:		√				√	√						√		
For REMOTE processing:															
Batch:	√			√			√	√					√		√
Time-shared:	√			√		√	√	√		√			√		
Documentation Evaluation See Page 000	Good	Inadequate	Adequate	Adequate	Adequate	Good	Good	Adequate	Good	Adequate	Adequate	Good	Adequate	Adequate	Adequate

TABLE 4A

BUSINESS AND OPERATIONAL INFORMATION FOR LOCAL PROCESSING SYSTEMS
SYSTEMS 1 to 7
Basic Equipment, Optional Peripherals, Maintenance and Terms

		Cincinnati	Digital	Encode	Ingersoll	MDSI	MS&S Mini-computer
Basic Equipment							
Host Computer (A = alternatives)		A IBM 370 A UNIVAC 1108 A Honeyw 6000	PDP 8A 16K	Encode 16K A Nova 3/12 32K	IBM 370/145 or 148 VS/1	A TI980B 32K A DG/230 48K A DG Nova 3/12 32K A PDP 11/60 32K	DG Eclipse S/130 48K
Terminal			ASR 33	A ASR33 A Dasher 30 cps	IBM	A ASR 33 A MDSI ST1	–
Other Components of Basic System			Paper Tape Punch/Reader	DG Disk	–	Line Printer, Reader/Punch, Card Reader	LeBlond DPS 4000 I/O Controller 10 Meg. Disk
Optional Peripherals							
Plotters		As Appropriate To Host System	A Tek. 4006 A Tek. 4662 A H.P. 7200A A H.P. 7203A	A DG CRT A Tek. 4662	On-Line Plotter A Tek. A Calcomp	A HP7203A-114 A HP 7221A-114 A Tek. 4014-1 A Dec. VT 52	CRT Digital Graphic Plotter
Additional Memory			–			RK06 Disk 256K	Floppy Disks (1 to 8) Core Expansion
Printers, Punches, Readers			Roytron 50 cps	300 cps Reader 75 cps Punch	–	Line Printer Reader/Punch Card Reader	Line Printer Mag Tape Reader Card Read/Punch
Maintenance	Availability	See Vendor	From Vendor	From D.G.	By IBM	From Vendor	From National Service Org.
	Cost	See Vendor	Est.: $200/Yr.	Est.: $130-$170/mo.	See IBM	Est.: $35-$500/mo.	Annual: $860/mo. On Call: $42/hr.
Terms	Purchase	See Vendor	From DEC	Via Distributors	See IBM	From Vendor	From MS&S
	Lease	See Vendor	Via Third Party	Via Distributors	See IBM	Via Third Party	Via Third Party

TABLE 4B
BUSINESS AND OPERATIONAL INFORMATION FOR LOCAL PROCESSING SYSTEMS
SYSTEMS 7 to 15
Basic Equipment, Optional Peripherals, Maintenance and Terms

	MS&S Maxi-computer	Olivetti	Threshold	United Computing	University Computing	Weber
Basic Equipment						
Host Computer (A = alternatives)	A: IBM 360 A: IBM 370 DOS/OS/VS	Olivetti P6060 8K	DG Nova 3/12 32K	A: "Turnkey" DG Nova 3/12 and Eclipse G/230 A: DEC–PDP8S -PDP11s, System 10	DEC PDP8 64K 10 Meg. Disk	HP9835A
Terminal	–	Include in Above	Threshold Voice Preprocessor with Visual Display	HP2100A HP2116A TI980B UNIVAC 90/30 Varion 620*		Included in Above
Other Components of Basic System	DOS: 128K Core OS: 200K Core Disk Drive Printer Card Read/Punch	24K Memory Print/Plotter Tape Punch	10 Megabyte Disk CRT 22" Plotter Reader/Punch Line Printer A: ASR–33	Hi-Speed Paper Tape Reader & Mylar Tape Punch	Tape Read/Punch Line Printer	HP9872 A (Plot) HP9884 (Tape Punch)
Optional Peripherals						
Plotters	IBM Compatible CRT, Plotters	CRT Plotters	Flat Bed or Continuous	Calcomp Plotter TEK 4014 Hard Copy Units	CRT Plotters	–
Additional Memory	IBM Compatible Disks	Extra Disks Extra Core	–	–	Floppy Disk MPG. Tape	HP9885 M Flex. Disk Drive
Printers, Punches, Readers	IBM Compatible Card-Tape Converters	Hi-Speed Printer Card Reader CAS	High-Speed Printers, Punch	Line Printers Card Readers	Card Read/Punch	HP2631 Impact Printer HP9883 A Tape Reader
Maintenance Availability	By IBM	By Olivetti's 100 Service Offices	1	By DG.	By Vendor	By H.P.
Cost	See IBM	Av. – $1000/yr.		Varies with System	See Vendor	Av. - $106/mo.
Terms Purchase	By IBM or	See Olivetti		Turnkey, from United	See Vendor	–
Lease	Thru Third Party	See Olivetti			See Vendor	Lease Only
				*Also Burroughs B1700, Westingh. 2500		

TABLE 4C

BUSINESS AND OPERATIONAL INFORMATION FOR LOCAL PROCESSING SYSTEMS

Operating System and Application Software, and Documentation

	Cincinnati	Digital	Encode	Ingersoll	MDSI	MS&S Mini-comp.	MS&S Maxi-comp.	Olivetti	Threshold	United Computing	University Computing	Weber N/C
Operating System Software provided to the user:												
Software Identification	CINTURN II	Digital Floppy Disk: DEC 05/8	GDOS	IBM	RT-11; RSX-11M; TI 980; DG: RDOS IBM: OS/DOS etc.	RDOS DPS-4000	IBM OS/DOS	Basic	RDOS	UDS	Vendor's Std.	Prompt (Basic)
Who provides this software?	Cincinnati	DEC	Encode	IBM	Vendor	DG	*IBM*	*Olivetti*	*Vendor*	United	Vendor	Vendor
Who supports this software?	Cincinnati Milacron	DEC	Encode	IBM	MDSI: TI & DG Nova	MS&S for DPS-4000	MS&S, only for Std. IBM	Olivetti	Vendor	United	Vendor	Vendor
Application Software provided to the user:												
Preprocessors? In what languge?	Yes	Dev. to order	Assembler	No	No	No	Yes	Considering	Yes, Fortran	Yes, UPL	Yes, Fortran	–
Processors? In what language?	Yes, Fortran	PDP8 Ass'b.	Assembler	Yes	Assembler	Yes, Fortran	IBM Ass'b.	Ext. Basic	Yes, Fortran	Yes, UPL	Yes, Fortran	Yes, Basic
Postprocessors? In what language?	Yes	PDP8 Ass'b.	Assembler	Yes	Assembler	Yes, Fortran	IBM Ass'b. or Fortran	Ext. Basic	Yes, Fortran	Yes, UPL	Yes, Fortran	Yes, Basic
Is your company the developer?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	UCC & Vendor	Yes
Is your company responsible for maintenance?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	UCC & Vendor	Yes
Is there a procedure for prompt handling of failure reports?	Phone in; Follow-up	Documented bugs are patched	Phone in; Document	If Needed	Phone-in; Corrected	Phone in; 24-Hour Response	Phone in; 24-Hour Response	Phone in; Document	Phone in	Phone or Mail	APAR System	–
Do you maintain an in-house, full time staff of software developers?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Documentation provided to the user:												
Part programmer's manuals?	2/Syst. N.C.	2/Syst. N.C.	Yes N.C.	Yes $14.60	Yes	1/Syst. N.C.	2/Syst. N.C.	1/Syst. N.C.	2/Syst. N.C.	Yes	2 Syst.	Yes
Cost for extra manuals	$35	$25	–	$14.60	$30	$100	$45	$35	$15	$30	–	–
System manuals?	From Vendor	Yes	Yes	No	Yes	Yes	From IBM	Yes	By License @ $15,000	Yes	From Vendor	Yes
Do they include diagnostics?	See Vendor	Yes	Available	Provided	Yes	Yes	Provided	Yes	Yes	Yes	From Vendor	Yes
Operations manuals?	From Vendor	Yes	Yes	Yes	Yes	Yes	From IBM	Yes	Yes	Yes	From Vendor	Yes
Postprocessor SOURCE code;	Yes	No	No	Yes	No	No	Available	Yes	No		No	–
Cost for documentation (other than part programmer's manuals)	N.C.	N.C.	N.C.	Op. Man. $3.70	N.C.	–	–	N.C.	–	Varies	–	N.C.

TABLE 4D

BUSINESS AND OPERATIONAL INFORMATION FOR LOCAL PROCESSING SYSTEMS

Customer Training Courses and Consulting Services Offered

	Cincinnati Milacron	Digital	Encode	Ingersoll	MDSI	MS&S	Olivetti	Threshold	United Computing	University Computing	Weber N/C
Customer Training Courses Offered											
How often are training courses given?	Quarterly	2/Month	1/Month	As Necessary	26/Year	2/Month	1 or 2/Mo.	Before Installation	As Necessary	20/Year	At Time of Installation
How long are the courses?	5 Days	2 Days	5 Days	"	1 Week *	1 Week	5 Days	1 Week	Varies	1 Week	3 Days
Where are courses given?	Cincinnati Ohio	Monroeville Pa.	Newburyp't Mass.	Rockford Ill. or On-Site	Ann Arbor Los Angeles Houston or On-Site	Cincinnati Los Angeles	Tarrytown N.Y.	Delran, N.J. or On-Site	Carson	Dallas and 8 Regional Offices	On-Site
What is the cost per student?	$300	1st 2—N.C. More at $200	N.C.	Will Quote	Computer Use Only Av. $230	1st 4 N.C. More at $250	1st 2 N.C. More at $500	N.C. at Delran On-Site $240/Day Plus Exp.	Varies $280—$520	$300/wk. (Includes Use of Computer	N.C.
Are special courses given?	No	No	Advanced	Could be	Advanced 3 Days, 2/Mo. in 24 Cities	Advanced	Postprocessor Writing	Systems Design	Postprocessor and Others on Request	Intermed. Apt Adv. Apt Lathe Prog.	As Necessary
What is the cost for special courses?	n.a.	n.a.	$100	Will Quote	As Above Plus 2 Day* System Training	1st 2 N.C. More at $250 On-Site, $1000 Plus Exp.	1st 2 N.C. More at $500	$240/Day at Delran $320/Day Plus Exp. On-Site	$280—$520 Per Student	$300/wk.	—
Consulting Services Offered											
Is consulting support available on:											
Part programming?	Yes	Yes	Yes	—	Yes	Yes	Yes	Yes	Yes	Yes	Not Usually
Tooling?	Yes	No	Yes	—	No	Yes	No	No	No	No	"
System operation?	Yes	Yes	Yes	—	Yes	Yes	Yes	Yes	Yes	No	"
Other Subjects?	—	Preprocessor	—	—	Links		Post-processor	—	Yes	As Req'd	—
What is the cost of a consultant?	—	$200/Day	N.C.	Will Quote	N.C. Except for Corp. Appl. Engineer	N.C.	N.C.	Negotiated	Will Quote	N.C.	—

TABLE 5A

BUSINESS AND OPERATIONAL INFORMATION FOR REMOTE PROCESSING SYSTEMS

Geographic Areas Served and Equipment Requirements

	Cincinnati Milacron	General Electric	MDSI	MS&S	McAuto	SDRC	University Computing	Westinghouse
Areas Served	For remote processing, Cincinnati uses networks of: General Electric, or University Computing, or Westinghouse	North America, Great Britain, Europe, Japan, Australia, and Far East	North America, Great Britain, Japan	North America	Continental United States	North America	"International"	Continental Unit United States, Europe by private phone
Regional Center Locations		Rockville, Md. Brook Park, O., Amsterdam, Holland	Many in U.S. France Japan	Cleveland, O. Pittsburgh, Pa.	St. Louis, Mo. Long Beach, Ca.	Maryland, Ohio, Amsterdam, Holland	Dallas, Texas	Pittsburgh, Pa.
Dial-up Locations Served								
"All Major Cities"		✓	✓			✓	✓	✓
"Anywhere"			31 Lines in U.K. and Europe		✓			
Terminal at User Site	ASR 33	ASR 33 Terminet 300,30 IBM 2741, 3767	ASR 33 Bell 113A MDSI STI	LeBlond DPS 4000	ASR 33 Terminet, etc.	ASR 33 Terminet etc.	ASR 33 and All Popular Terminals	ASR 33 ASR 35
Terminal Service by Supplier?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Terminal Back-up: User's Responsibility?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Host Computer	See note above	Worldwide Net of Honeywell 6000 and IBM 370	XDS 940's	IBM 370/155, or 158, or 165	CDC 175	Honeywell 6080 or Use G.E. Co. Network	UNIVAC 1108	IBM 370/165
Host System Service by System Vendor?		Yes	Yes	Yes	Yes	Yes	Yes	Yes
Host Computer Back-up:		✓	✓		✓	✓		
Automatic switch to another unit							✓	✓
Own staff maintenance								

TABLE 5B

BUSINESS AND OPERATIONAL INFORMATION FOR REMOTE PROCESSING SYSTEMS

Documentation and Postprocessors

	Cincinnati Milacron	General Electric	MDSI	MS&S	McAuto	SDRC	University Computing	Westinghouse
Documentation Provided to the User								
Part Programmers' Manuals	Yes	Yes	Yes	Yes, 2/Site	Yes	Yes	Yes	Yes
Systems Manuals	Supplied	Yes	Yes	No Source	No	Yes	Yes	For Postprocessor
Operations Manuals	by the	Yes	Yes	Yes, 2/Site	No	Yes	Yes	Yes
Postprocessor Source Code	Time-Share	Usually	See Page 000	No Source	Yes	No	Yes	Yes
Other Documentation	System	Various Others Available	Yes	Class Manuals Postprocessor Manuals	No	No	No	Westurn, CAM, FMILL
Cost of Documentation	1/Student – N.C. $35/Extra Copy	See Publication List Prices	N.C.		Included in System Validation Cost – Approx. $100	N.C.	Prices Available	N.C.
Postprocessor Sources, Costs, and Maintenance			See Page 000					
Does your staff develop postprocessors?	Yes	Yes for U.S. Customers	Yes	Yes	Yes	Yes	Yes	Yes
Who else develops postprocessors?	–	G.E. Service Div. for Other Countries	–	User May be Supplied Std. Apt/Adapt	–	–	A Few by Machine Tool Builders	Some by Machine Tool Builders
Does your staff support postprocessors?	Yes	Yes	Yes	Yes	Yes	Yes	Yes, for UCC's	Yes
Any other support?	–	As Above	–	–	–	–	Vendor Supports His Own	–
How are postprocessors updated and maintained?	Fixed at User's Request	Updating G.E.'s Library; also Storing User's Customized P.P.	Fixed at User's Request	?	McAuto Fixes	As Necessary	APAR System Upon Report from User	–
What are the costs for postprocessors?	N.C.		N.C.	N.C.	Based Upon Usage. Can be No Charge for Heavy User	N.C.	$100–$300	Fee
What are the costs for maintenance?	N.C.		$12/mo. for Maintenance and Storage	N.C.		N.C.	N.C.	N.C. if Westinghouse Westinghouse P.P.
Are there other costs?	–			Storage Charge		Storage Charge	Storage $10/Mo.	N.C.

TABLE 5C
BUSINESS AND OPERATIONAL INFORMATION FOR REMOTE PROCESSING SYSTEMS
Customer Training and Consultation Services

	Cincinnati Milacron	General Electric	MDSI	MS&S	McAuto	SDRC	University Computing	Westinghouse
Customer Training Courses Offered								
How often are training courses given?	Quarterly	Quarterly	26/Year	2/Month	As Needed	As Needed	20/Year	Basic Apt. 6/Yr. Adv. Apt. 2/Yr. Westurn 2/Yr.
How long are the courses?	1 Week	1 Week	1 Week	1 Week	2 Weeks	1 Week	1 Week	Apt's 1 Week Westurn 3 Days
Where are the courses given?	Cincinnati, O.	Rockville, Md.	Ann Arbor, Mich. Los Angeles, Calif. Houston, Texas or On-Site	Cincinnati, O. Los Angeles, Calif.	St. Louis, Mo., and Customer Site	Cincinnati, O.	Dallas, Texas and 8 Regional Offices	Pittsburgh, Pa.
What is the cost per student?	$300	$50/Day	Computer Use Only. Av. $230	1st 4 N.C. More at $250	Time and Materials	N.C.	$300	Apt's $375 Westurn $225
Are special courses given?	No	Post Processors 1/Yr. at Rockville	Adv. N.C 3 Days 2/Month 24 Cities	Adv. Post Proc. Use and/or Development	No	No	Int. Apt. Adv. Apt. Lathe Program	Rarely, on Informal Basis
What are the costs for special courses?	n.a.	$50/Day	On-Site $1000. + Exp. + Computer		n.a.	n.a.	Varies	n.a.
Consulting Services Offered								
Is consulting support available on:								
Part programming?	Yes	Yes, By Phone	Yes	Yes	Yes	Yes	Yes	Not Usually
Tooling?	Yes	No	No	Yes	–	No	Yes	Not Usually
System operation?	Yes	Yes	Yes	Yes	–	Yes	Yes	Not Usually
Other subjects?	No	No	Links	Cost Reduction File Mgm't. N.C. Application	–	–	As Required	Not Usually
What is the cost of a consultant?	Phone N.C. On-Site $40/Hr. + Exp.	On-Site $26 to $50/Hr. + Exp.	N.C. for Field Service Engineer. Fee for Corp. Appl. Engineer	N.C.	Time and Materials	Quoted by Job	N.C.	n.a.

CHAPTER V

BASIC GEOMETRICAL CAPABILITIES

Numerical control manuscript preparation requires the programmer to enter the geometrical configuration of the part being programmed. The systems vary in their ability to handle certain geometries, and also in the formats acceptable as manuscript statements to describe those geometries. A comparison of these capabilities will help the user to select a system to match his requirements.

When a programmer sits down to write an NC lathe program, he must in some way communicate to the system the shape and dimensions of the part to be made. Usually this information comes to the programmer in the form of a set of documents including a detailed drawing of the finished part, and a specification or drawing of the input raw material. The drawings will contain a formalized pictorial representation of the shape, plus dimensions (and tolerances) of the surfaces to be machined.

Each of the systems studied in this report provides a means for the programmer to communicate shapes and sizes to his processor. The systems differ, however, in the complexity of the geometrical surfaces which they were designed to handle. A very large fraction of all lathe-produced surfaces fall into five categories:

- The first three are all defined by straight lines in the part profile:
  - — Cylindrical surfaces, sometimes called "diameters" because of their distinguishing dimensions;
  - — Conical surfaces, also called tapers;
  - — Plane surfaces, called faces, lying in a plane perpendicular to the axis,
- The fourth and fifth are defined by circular arcs in the part profile:
  - — Spherical surfaces, with center on the lathe axis;
  - — Toroidal surfaces, which includes circular blends, fillets and round corners.

Some system designers have consciously limited their systems to these few surfaces, gaining simplicity in the software while foregoing the ability to handle those parts which contain surfaces not listed above.

Because of the essentially two-dimensional nature of lathe part characterization, the other geometrical elements encountered (besides straight lines and circles) will be other curves:

— Mathematically defined by an equation, such as the conic sections:

— Non-mathematically defined, commonly known as tabulated curves because they are defined by tabulating the coordinates of several points on the curve.

Some system designers have provided the necessary software to handle one or both of these families of curves.

When a programmer examines a part drawing he will find that the elementary segments of the profile have been located and dimensioned in a way that was convenient to the detailer, or was a natural result of the way that segment's shape and size were determined by its function. For example, the tool holder socket in the end of the spindle in a milling machine is determined by the standard contour of the tool holders. As a result, a given surface may be dimensioned on the drawing in several different ways: A taper may be defined

— b᾿ 'ts length and the diameters at either end, or
— by its length, the diameter at one end, and the angle of taper, or
— by one diameter and z dimension, and tangency to a spherical or toroidal surface some distance away.

However, the processor in the computer will require that a surface be defined in one fixed manner, called the "canonical equation." Systems designers have saved the programmers from having to convert the drawing dimensions to the canonical form by arranging their system so it will accept any form of dimensioning reasonably to be expected on a drawing; the processor makes the conversion calculations

Some typical canonical equations for lines, circles, and other curves follow. They are not necessarily the forms used in any specific system; there are several similar formats. Some systems can define a new element by inserting coefficients in the canonical format, or by recalling the canonical form of some previously defined element and changing the coefficients. See "Canonical Definitions" in Table 9 for this capability.

Typical Canonical Formats

Point	A, B
Line	$Ax + By + C = 0$
Conic, or second order curve	$Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0$
Circle	$Ax^2 + Cy^2 + F = 0$

None of the fifteen systems contained canonicals for third order curves.

Typical Algebraic Formats

Line	$y = mx + b$	
Circle	$x^2/r^2 + y^2/r^2 = 1$	when
Parabola	$y^2 = px$]	symmetrical
Ellipse	$x^2/a^2 + y^2/b^2 = 1$	about the
Hyperbola	$x^2/a^2 - y^2/b^2 = 1$	origin

Each system will have its own way of writing the manuscript entries for any given one of the acceptable definitions. The fact that these formats differ from one another is a characteristic of the differences in the syntax of the languages, not in the acceptability of that particular kind of a definition of a point, line, or circle. From a user's point of view, it is only necessary that there be *a* way to enter the point, line, circle, or other curve's geometry which he expects to encounter in his product designs.

On the other hand, if his designers place dimensions on drawings in such a way that none of the definitions applicable in a given language exactly fits the situation, then the user will see if a combination of two definitions will serve the purpose. For example (refer to Tables 6A and 8A):

> Suppose the programmer needs to define a circle as concentric with a previously defined circle whose center location he does not know, such as one defined as passing through three points. (See Circles, 6.) He can first define a point as the center of the known circle (see Points, 4), and then define his desired circle as having its center at that point and having the desired new radius. (See Circles, 1.)

Such definitions are called nested definitions; if his system will accept nested definitions, he can identify an element in one statement without resorting to mathematical calculations. See "Nested Definitions" on Table 9 for this capability. No definitions achieved by nesting have been included in the matrices, for obvious reasons.

The systems differ in the variety of alternate formats they will accept for points, lines, circles, and other curves. Some system designers have deliberately limited their systems to a relatively few very common formats, thus gaining simplicity in their software while possibly requiring the programmer to make a few calculations on the side.

Certain definitions are ambiguous unless a selector is added. For example, a line through a point and tangent to a circle could have two possible locations. (See Lines, 7.) A selector is added to specify which line is desired. Such symbols as XL, XS (for X Large, X Small), ZL, ZS, IN, or OUT may be used as selectors. An "S" in the matrix indicates this requirement. When two or more selectors are required to avoid ambiguity (see Circles, 9), the matrix entry is "SS..".

The matrices contain the most frequently encountered formats for definitions; some of the simpler languages do not even use all these choices. There are of course many more esoteric and unusual possibilities, such as:

Points:	At the intersection of a conic and a tabulated curve.
	At the intersection of two conics.
Circles:	With a given radius, tangent to two conics.
	With a given radius, tangent to two tabulated curves.

Such formats will not be found in the matrices; if a potential user expects to encounter the need for them, he should check with the vendors on his list of candidate systems.

We have also omitted definitions derived from other definitions by rotation or translation of the coordinate system, or both. While such procedures can be very valuable, we have construed them to be matrix algebra manipulations, rather than definitions of geometry. Similarly, the

Olivetti and Weber systems which use "directed geometry" consider lines and circles as having a specific direction of motion along the line or circle. Reversing the direction changes the property of the element for programming purposes, but does not locate it in a different position. We have omitted these reversals from the matrices.

All of the foregoing is related to geometric definitions following the classical geometric/trigonometric approach, in which points may define lines and circles, and the intersection of lines and circles may define points. One system, PROMPT, uses a quite different approach. It thinks of the two dimensional part boundaries as paths along which a cutting point moves. In PROMPT, the path's segments must be either a straight line or a circular arc. Circles are defined by their center coordinates and radius — dimensions usually available from the detail drawing. (If not, the programmer is provided with a mathematical macro or sub-routine so that he can do a little trigonometry to get the dimensions.) Circles are connected either by straight lines or other circles. Points in a boundary, such as the intersection of two straight lines, are treated as circles of zero radius.

Arcs and lines are strung together to make a boundary. To describe it, the programmer divides it into "composites" of two to five elements. He describes the parameters of the composite in a sequence of dimensions suitable to each composite type. The last element of one composite becomes the first element of the next.

The PROMPT programmer has no need for a multiplicity of alternate ways to convert drawing dimensions into point locations and line and circle definitions. He has but one definition for a point location, a straight line, or a circle's center. As a result, the matrix format (Tables 6, 7, 8, and 9) are not compatible with the PROMPT system, and vice versa, and the absence of check marks in the PROMPT column should not be misconstrued. The geometrical capabilities of the various languages will be apparent in the matrices on the following pages. There are four sections, one each for Points, Lines, Circles, and Other Curves. The latter page also includes some miscellaneous indicia.

TABLE 6 A

DEFINITIONS OF POINTS ACCEPTED BY THE VARIOUS SYSTEMS

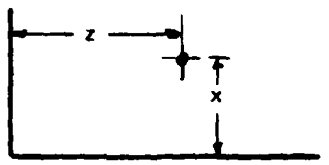
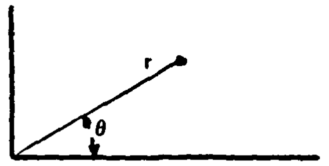
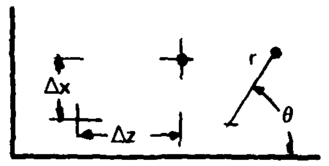
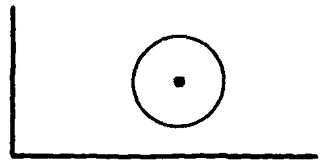
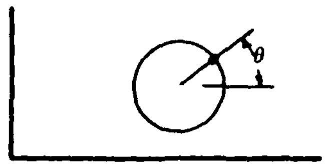
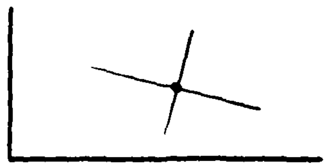
		Location determined by:	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
1		Cartesian coordinates x,z or x,y	X	X	X		X	X	X	X	X	X	X	X	X	X	X
2		Polar coordinates r, θ	X	X			X	X	X	X	X		X	X	X		X
3		Offset from a given point Δx, or Δz, or both; or r, θ	X	X		See Chapter VII, page 84	X	X	X	X	X	X	X	X	X	See Chapter VII, page 106	X
4		Center of a known circle	X	X			X	X	X	X	X	X	X	X	X		X
5		Angular location on a known circle	X				X	X	X	X		X	X	X	X		X
6		Intersection of two lines	X	X	X		X	X	X	X	X	X	X	X	X		X

TABLE 6 B

DEFINITION OF POINTS
ACCEPTED BY THE VARIOUS SYSTEMS

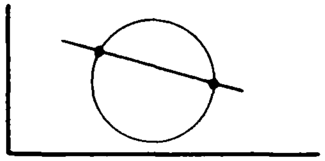
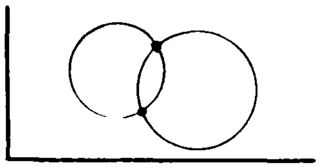
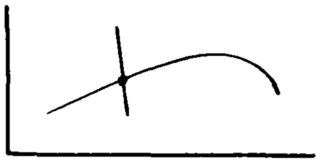
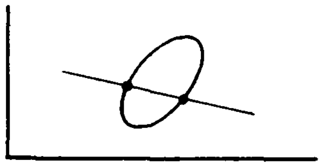
		Location determined by:	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
7		Intersection of a line and a circle	X S	X	X S		X S	X S	X S	X S	X S	X S	X S	X S	X S		X S
8		Intersection of two circles	X S	X S	X S		X S	X S	X S	X S	X S	X S	X S	X S	X S		X S
9		Intersection of a line and a tabulated curve	X							X			X	X	X		X
10		Intersection of a line and a conic	X S						X S	X S			X	X S	X S		X S

TABLE 7 A

DEFINITION OF LINES ACCEPTED BY THE VARIOUS SYSTEMS

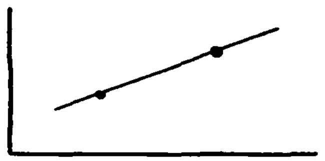
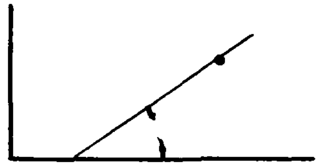
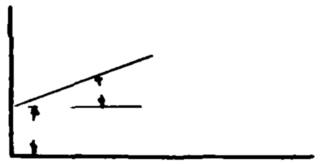
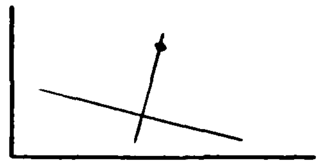
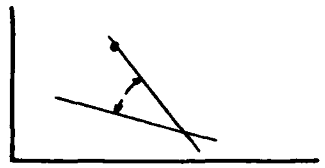
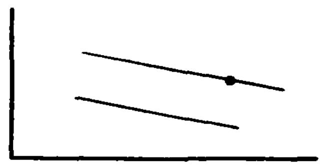
		Location determined by:	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
1		Two points	X	X	X		X	X	X	X	X	X	X	X	X	X	X
2		One point, and an angle with the Z axis	X	X	X		X	X	X	X	X	X	X	X	X		X
3		X axis intercept, and the slope of the line	X	X	Axis Only	See Chapter VII, page 84	X		X	X		X	X	X	X	See Chapter VII, page 106	X
4		One point, and perpendicular to another line, or to an axis	X	X	X		X	X	X	X	X	X		X	X		X
5		One point, and a given angle to another line, or to an axis	X	X	X		X	X	X	X	X	X	X	X	X		X
6		One point, and parallel to another line, or to an axis	X	X	X		X	X	X	X	X		X	X	X		X

TABLE 7 B

DEFINITION OF LINES
(cont.)

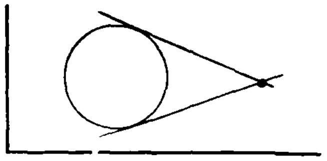
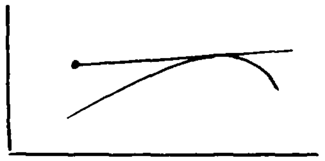
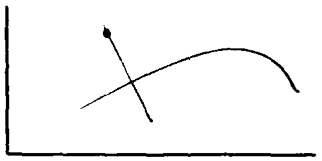
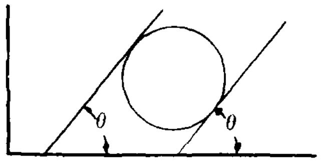
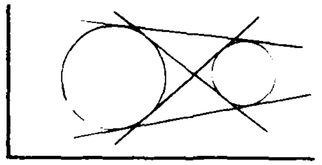
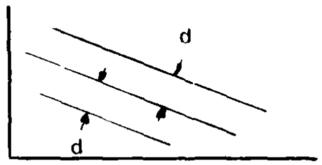
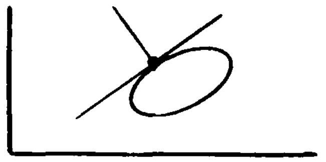
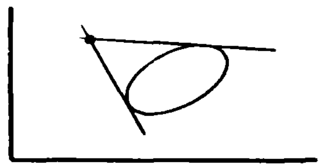
		Location determined by:	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	.MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
7		One point, and tangent to a circle	X S	X	X S		X S	X S	X S	X S	X	X S	X S	X S	X S	X	X S
8		One point, and tangent to a tabulated curve	X						X 2	X				X	X		X
9		One point, and perpendicular to a tabulated curve	X						X 2	X				X	X		X
10		Tangent to a given circle, at an angle to the Z axis		X S	X S		X S	X S	X S		X	X S	X S				
11		Tangent to two circles	X SS	X S	X SS		X SS	X SS	X SS	X SS	X	X SS	X SS	X SS	X SS	X	X SS
12		Parallel to another line, at a given distance	X S	X S	X S		X S	X S 1	X S	X S	X	X	X S	X S	X S		X S

TABLE 7 C

DEFINITIONS OF LINES
(cont.)

		Location determined by:	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
13		Tangent or perpendicular to a conic, at a point on the conic							X						X		
14		Tangent to a conic, and through a given point							X S						X S		

Footnotes:

1 Distance may be measured parallel to X axis, parallel to Z axis, or perpendicular to the line.

2. The point must be on the tabulated curve.

TABLE 8 A

DEFINITION OF CIRCLES
ACCEPTED BY THE VARIOUS SYSTEMS

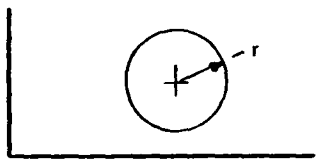
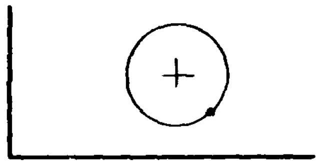
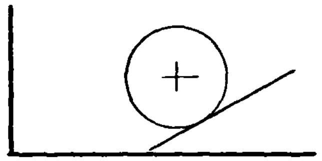
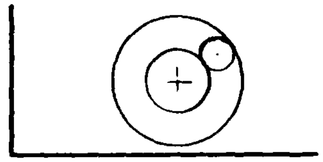
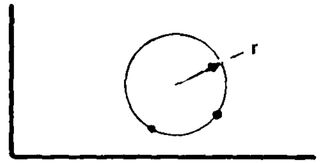
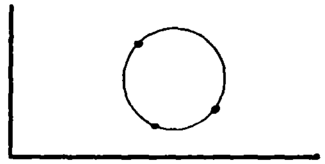
		Location determined by:	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
1		Center at a point, given radius	X	X	X	See Chapter VII, page 84	X	X	X	X	X	X	X	X	X	X	X
2		Center at a point, and circumference passing through another point	X	X	X		X	X	X	X		X		X	X	See Chapter VII, page 106	X
3		Center at a point, and tangent to a given line	X	X			X	X	X	X	X	X	X	X	X		X
4		Center at a point, and tangent to a given circle	X S	X S			X S	X S	X S	X S	X S	X S		X S	X S		X S
5		Through two given points, with a given radius		X							X						X S
6		Through three given points	X	X			X	X	X	X	X	X	X	X	X		X

TABLE 8 B

DEFINITION OF CIRCLES
(cont.)

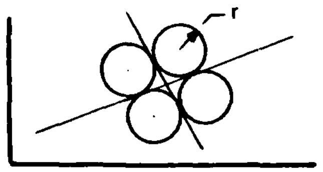
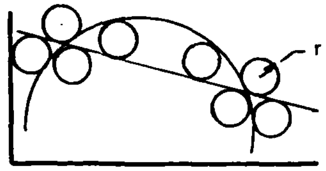
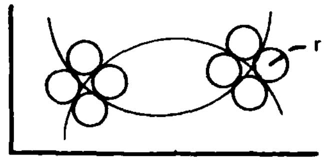
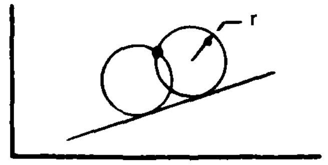
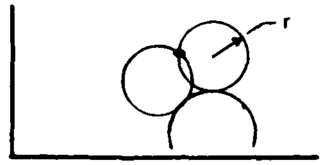
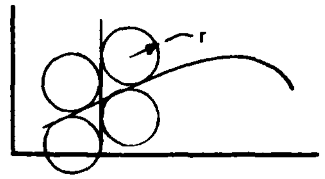
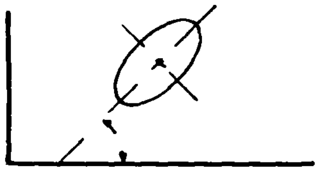
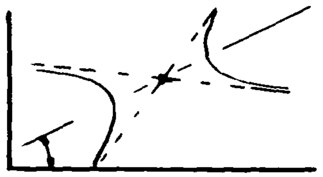
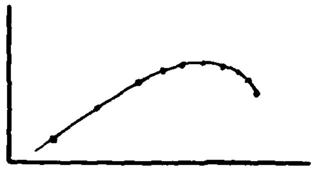
		Location determined by:	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
7		Given radius, tangent to two given lines	X SS	X SS			X SS	X SS	X SS	X SS	X	X SS	X SS	X SS	X SS		X SS
8		Given radius, tangent to a line and a circle	X SSS	X SSS			X SSS	X SS	X SSS	X SSS	X		X S	X SSS	X SSS		X SSS
9		Given radius, tangent to two circles	X SSS	X SSS			X SSS	X SS	X SSS	X SSS	X		X S	X SSS	X SSS	X	X SSS
10		Given radius, through a point and tangent to a line	X S	X S			X S		X S	X S	X	X S		X S	X S		X S
11		Given radius, through a point and tangent to a circle		X S							X S	X S	X S				
12		Given radius, tangent to a line and to a tabulated curve	X SS							X SS			X SS	X SS	X SS		X SS

TABLE 9

DEFINITION OF OTHER CURVES ACCEPTED BY THE VARIOUS SYSTEMS

			Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
		Second Order Mathematical Curves															
1		Ellipse, defined by center point, semi-major axis, semi-minor axis, angle with Z axis	X						X	X			X	X	X		X
2		Hyperbola, defined by center point, half-transverse axis, half conjugate axis, angle with Z axis	X						X	X				X	X		X
3	$Ax^2+Bxy+Cy^2+Dx+Ey+F=0$	General conic, whose canonical parameters satisfy the equation	X					X	X	X			X	X	X		X
4		Canonical redefinition, by changing the parameters of a previously defined general conic	X	X				X	X	X			X	X	X		X
		Non-Mathematically Defined Curves															
5		A tabulated curve, described by a table of coordinates of points lying on the curve	X						X	X		X	X	X	X		X
6		Nested definitions capability	X	X	X		X	X	X	X			X	X	X		X

CHAPTER VI

LATHE PROGRAMMING ROUTINES

Much of lathe programming can be expedited by using generalized sub-routines for the common turning operations. The user should be familiar with these operations and with the simplified programming methods offered by the various systems.

Certain patterns of tool movement in lathe work repeat themselves so often that they present an ideal opportunity to use a generalized sub-routine, or canned cycle, or macro in the program. To make use of such a routine it is only necessary to identify it and to supply the basic dimensional parameters. The software then inserts the correct distances, feeds and speeds in the correct sequences, and the generalized pattern becomes the specialized pattern for the particular part being programmed. The software then generates the equivalent of a complete series of manuscript statements, and processes them as usual.

This feature greatly simplifies program writing; it shortens the source code input to the processor; it insures a proper sequence of motions, and it eliminates programmer errors to a large extent. Fortunately the technique is applicable to a large portion of the lathe programming task. It is commonly available for automatic roughing, finishing, threading, grooving, drilling and boring. The routines are compatible with the more conventional tool motion statements, so that when a part presents a need for a cut not available as a sub-routine, it can be programmed directly in the conventional manner in the same input manuscript. On the other hand, the systems designers of some of the less sophisticated programming systems have consciously elected to limit their system to those lathe operations covered by these routines, plus some rather simple tool motion statements, or by user-written macros.

The systems studied in this report vary as to what routines they offer, as to the versatility of these routines, and as to the extent to which a programmer may insert therein his own preferred metal cutting technology or shop practice. Their relative capabilities will be apparent in the matrices which follow in the individual sections.

In general, each routine includes the following basic elements:

- Selection of the required cutting tool by turret location,
- Movement of the tool at rapid traverse speed to the proximity of the work,
- Deceleration and approach at feed speed, when appropriate,
- Performance of the cutting sequence,
- Departure in a safe path from the proximity of the work, and
- Return of the tool at rapid traverse speed to the home position of the turret.

Intervention of the programmer may be necessary to assure safe approach and departure paths, particularly on internal boring or threading, or threading close to a shoulder, etc.

In the sections that follow, we present a general discussion of the various common routines, their variables and their variations, and then present a matrix of evaluations of the systems in the study. It is impossible in some instances to phrase questions which adequately explore all the differences without getting into too much complexity. It has seemed better in such cases to add footnotes to bring out these detailed points, or to discuss the subject at more length in Chapter VII.

Letters inserted in the matrices that follow have a significance indicated by an underscore in the questions asked. For example: "Y" or "N" in the matrix may be explained by "Yes, No" in the question. For another example, "N," "L," "R," or "E" in the matrix may refer to "No, to the Left, to the Right, Either" in the question line. When Both or Choice is offered in the question, it is assumed that the programmer must indicate the desired option. An entry "na" means that the question is not applicable to that particular system. Footnotes are grouped at the end of each table, sometimes on a separate page.

A. Automatic Roughing

Lathe work normally begins with an oversized work piece, which is *first* reduced to a "nearly-finished" shape by roughing cuts which remove the bulk of the unwanted material, and *second* brought to a "final-finished" shape by a variety of finishing, grooving, threading and similar operations. This section is directed to those first operations called roughing.

"Roughing" is the planned sequence of cuts necessary to remove all the material between the initial stock profile and the finished-part-profile-plus-stock-allowance. It is usually accomplished in a systematic series of straight-line passes. The depth of lathe roughing cuts is limited to what a single point tool can remove in one pass, so that multiple passes are normally required in the removal of excess stock.

Lathe work usually begins with either a piece of bar stock, or with a casting, forging, or weldment. The initial stock size is easy to define when working from bar stock; usually an outside diameter and a length will be adequate. (See Figure 2.) Initial size definition is more complex if the work piece is a casting, forging, or weldment, for the very reason that they can be pre-formed to nearer net size, which while it means less metal to remove, means more complex initial surfaces to define. (See Figure 3.)

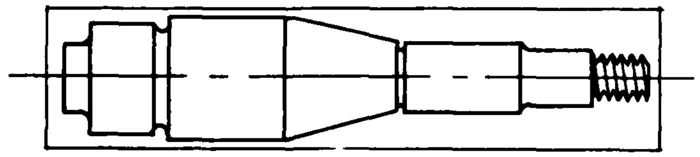


FIGURE 2 PART CUT FROM BAR STOCK

TABLE 10

SIMPLIFIED PROGRAMMING MEANS

Special lathe programming routines which simplify and expedite the preparation of part program manuscripts.

General Availability	Cincinnati	Digital	Encode	General Electric	Ingersoll	M D S I	M S & S	McAuto	Olivetti	S D R C	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
Does the system provide simplified programming means for the following lathe operations:															
Automatic roughing from raw stock size to near finished part size?	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Finishing cuts along a prescribed profile?	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Threading with single point cutting tools?	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Threading with taps and dies?	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y	Y	N	Y
Plunge cuts for grooving and necking?	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Drilling, boring, and reaming cuts?	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Does the system provide capability for programming lathe tool motions other than the above macros?	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

For more detailed examinations of the capabilities of the above macros, see the sections which follow, each with its more detailed matrix.

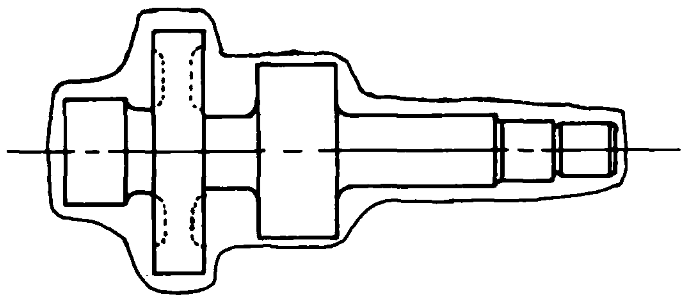


FIGURE 3 PART CUT FROM A CASTING

It is common practice to carry the roughing cuts to a point close to the finished part shape, leaving a small amount of material to be removed with a finishing tool, running at an appropriate speed and feed to yield the required surface finish and tolerances. (See Figure 4.) Not all surfaces on a part will require the same stock allowance for the finish cuts; indeed, some surfaces may not get a finish cut. Nevertheless, the "finished part profile" is the usual reference for the inner boundary of the roughing operation. Stock allowance over the finished profile, if not uniform, is specified surface by surface.

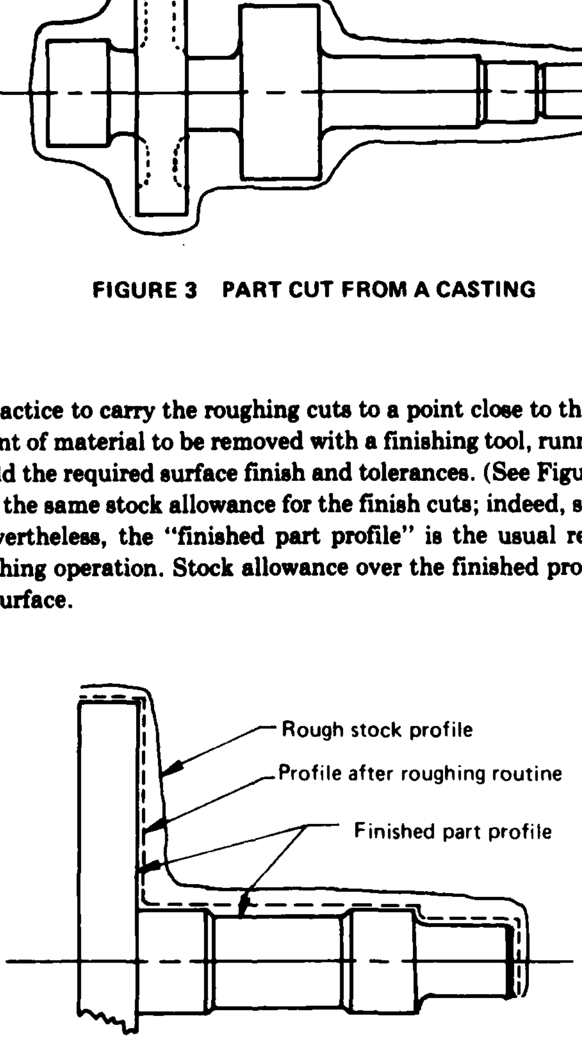


FIGURE 4 ROUGH STOCK AND FINISHED PART PROFILES

Depth of Cut

The amount of material that can be removed in a single pass is a function of the part material, the cutting point, the holding fixture, and the design of the lathe. It is not always advisable to remove the maximum possible amount. The depth of the initial cut into a casting or forging may be specified in order to insure a) getting under any hard skin, and b) continuity of cutting even though the actual profile may not be exactly what is expected, or may not be perfectly round or concentric.

TABLE 10

SIMPLIFIED PROGRAMMING MEANS

Special lathe programming routines which simplify and expedite the preparation of part program manuscripts.

General Availability	Cincinnati	Digital	Encode	General Electric	Ingersoll	M D S I	M S & S	McAuto	Olivetti	S D R C	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
Does the system provide simplified programming means for the following lathe operations:															
Automatic roughing from raw stock size to near finished part size?	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Finishing cuts along a prescribed profile?	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Threading with single point cutting tools?	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Threading with taps and dies?	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y	Y	N	Y
Plunge cuts for grooving and necking?	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Drilling, boring, and reaming cuts?	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Does the system provide capability for programming lathe tool motions other than the above macros?	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

For more detailed examinations of the capabilities of the above macros, see the sections which follow, each with its more detailed matrix.

After the first cut, the programmer may wish to take a series of equal depth cuts; if the calculated number of passes contains a fraction, an extra cut may be taken and the depth of each cut may be reduced to make all cuts of equal depth.

It may also be desirable to specify a minimum depth of cut which may be acceptably taken with a roughing cutter. Or, if after a certain number of roughing cuts, less than this minimum amount remains, it may be desirable to leave the rest for the finishing tool, or to distribute it over the preceding cuts by increasing each slightly and thereby save a pass across the work.

Direction of Cut

Roughing cuts may be parallel to the axis, perpendicular to the axis, at an angle to the axis, or some combination of the above. Not infrequently both perpendicular and parallel passes may be made on the same piece, first one set and then the other set. (See Figure 5.) Parallel passes are usually made from tail stock toward the head stock, or in the direction of increasing part diameter. The same is true of slightly angled passes. Perpendicular passes may be programmed inwardly or outwardly.

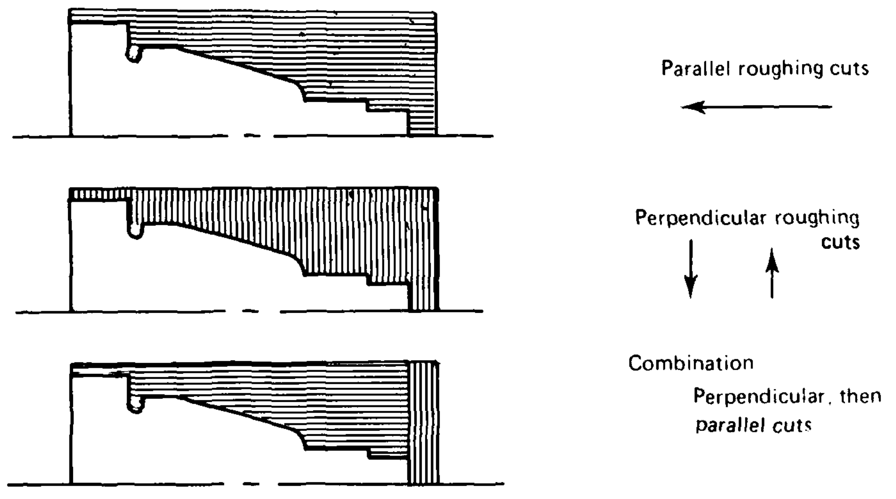

Parallel roughing cuts
Perpendicular roughing cuts
Combination
Perpendicular, then parallel cuts

FIGURE 5 DIRECTION OF ROUGHING CUTS

Pullout from Roughing Passes

When a parallel pass terminates at a face, the tool feed stops and the tool may pull out at right angles to the axis. When a parallel pass terminates at any other surface, pullout may be either a) perpendicular to the axis, b) at an angle to the axis, or c) parallel to the profile.

Option (c) leaves a uniform stock allowance, but requires a contouring cut; options (a) and (b) leave a serrated surface, but do not require contour control. (See Figure 6.)

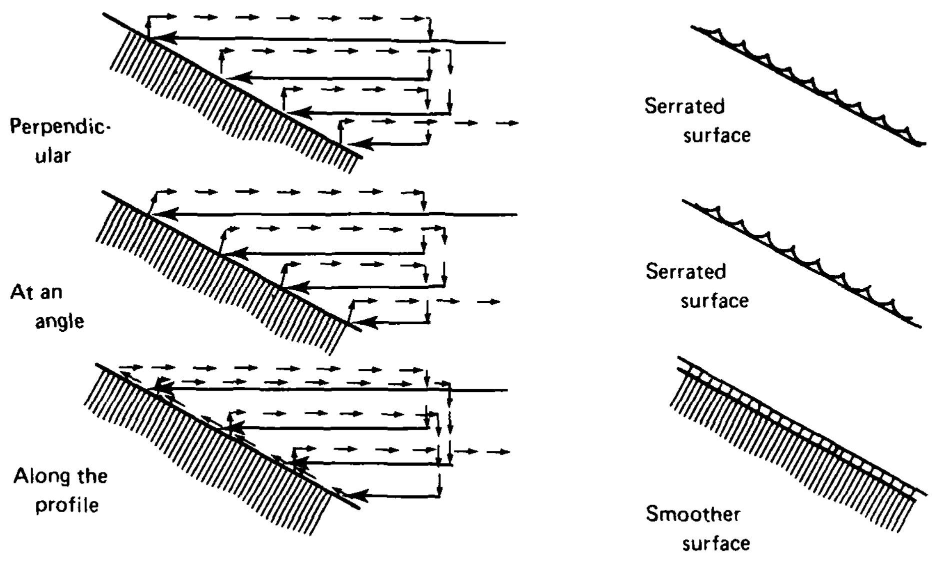


FIGURE 6 DIRECTION OF ROUGHING TOOL PULLOUT CONTROLS SURFACE TEXTURE

The Final Roughing Pass

Preliminary roughing passes are usually straight line motions (except for pull-out as noted above). A final contoured roughing pass may be programmed following parallel to the finished part surface but leaving the necessary stock allowances for finishing. It removes residual roughing tool marks.

Not all the features of the final finished part profile are reflected in the final roughing pass. (See Figure 7.)

1. Grooves, necks, etc., are usually omitted as beyond the capability of the roughing tool.
2. Recesses or undercuts are omitted, to be cut by a) the finishing tool or b) separate roughing routines. See below.
3. Fillets and chamfers may be left for the finishing tool pass.

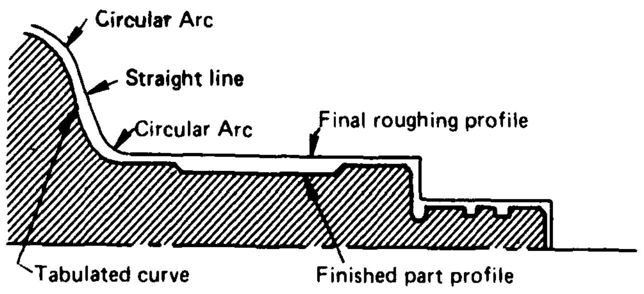


FIGURE 7 FINAL ROUGHING PROFILE AND FINISHED PART PROFILE

Internal Roughing

The same options in the removal of rough stock apply to internal as to external surfaces, except that internal roughing is usually not programmed perpendicular to the axis. (See Fig. 8.)

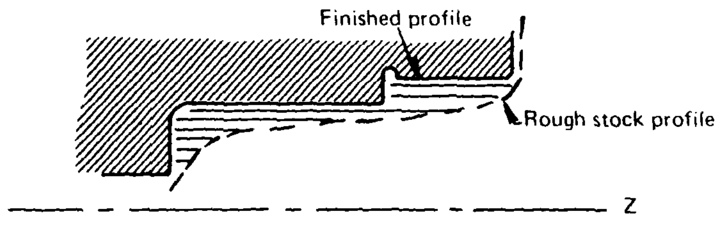


FIGURE 8 INTERNAL ROUGHING

All that has been said above concerns procedures that may be programmed move by move, using ordinary program writing techniques. However, most NC lathe programming systems offer automatic roughing routines which will program the entire process of excess stock removal, given a few simple instructions:

1. The rough stock profile
2.a) The finished part profile plus stock allowances, or
 b) The final roughing pass profile
3. The applicable metal cutting technology

Thereafter the system generates the entire sequence of commands necessary to select the roughing tool, approach the work, make the cuts, and withdraw safely.

The systems vary as to how much the programmer may modify the automatically generated sequence, or the cutting technology, etc. They also vary as to the extent to which they can be caused to optimize the sequence and procedure, in order to reduce cutting time in the lathe.

The final roughing profile may be defined independently of the finished part profile. This makes it possible to use simpler profile segment definitions. For example, a tabulated-curve surface in the finished part may be replaced by straight lines and circular arcs for the final *roughing* profile. In that case, the final *finished* profile must be separately defined. (See the left section of Figure 7.)

A single lathe part may require more than one automatic roughing routine. For example, a part may require an external roughing sequence and an internal roughing sequence. Or the external shape may be treated as two or more separate areas, each to be roughed by a separate routine.

Undercuts or Recesses

A part design may include a slight undercut on a portion of a cylindrical surface, such as a grind relief, or a recess cut in a face surface. Undercuts are defined as follows:

When a tool is performing turning, boring or facing cuts it ordinarily progresses consecutively in one Z direction and in one X direction, alternating or combining these moves as appropriate. If to traverse a profile, one of these progressions has to reverse direction, the surface is said to be undercut.

Most systems which offer automatic roughing routines will not rough undercuts in the automatic roughing cycle. A separate series of cuts are required.

When the undercut is shallow — of the order of magnitude of a finishing cut depth — the undercut may be ignored in roughing and stock removal left to the finishing tool. (See Figure 7.) When the undercut is deeper, a separate roughing routine is written to remove the material in the undercut area below the final boundary of the main roughing routine. (See Figure 9.) Finishing of this undercut area may be included in the finishing routine for the entire part, or done separately.

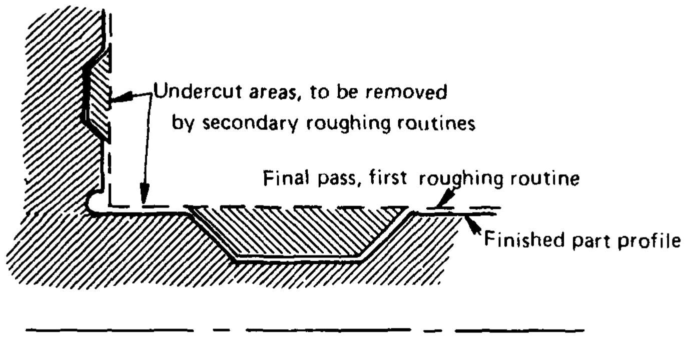

Undercut areas, to be removed
by secondary roughing routines
Final pass, first roughing routine
Finished part profile

FIGURE 9 UNDERCUTS AND RECESSES

Tool geometry for roughing and finishing undercuts is determined by the shape of the area to be cleaned out.

B. Finishing

A finish cut in lathe work leaves the machined surface at the design dimension (within the design tolerance) and with the specified surface smoothness. The term usually applies to internal, external and face surfaces, plus all fillets, blends, round corners or chamfers. A part made in a lathe need not receive a finish cut on all its surfaces; if desired, the original stock surface may be left untouched, or a rough cut surface may be left without a finishing cut.

Finish cuts may be taken on the initial stock profile if the amount to be removed is small. More generally, finish cuts follow roughing cuts which have removed most of the unwanted material from the initial stock profile. See the section on Automatic Roughing. If the finished part profile has been defined in connection with an automatic roughing routine, it need not be redefined. However, finish cuts may only be taken if some stock has been left on the surface by the roughing tool for that purpose; if no stock allowance was left on some element of the surface, it is presumed that that element will be skipped in the finishing operation.

Finish cuts begin at a starting point and traverse the part profile, surface element by surface element, to an ending point. Between those two points, the surface is defined by a series of line segments connected end to end. Fillets and chamfers or round corners may be included in the

TABLE 11A

SIMPLIFIED PROGRAMMING MEANS

Automatic Roughing

	Cincinnati	Digital	Encode	General Electric	Ingersoll	M D S I	M S & S	McAuto	Olivetti	S D R C	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
How many different boundaries may be used in one part program? No limit, a number X	N	100	[2]	N	[3]	20	[1]	N	2	N	N	N	N	N	N
How many line segments may be used in one boundary definition? No limit, a number X	N	100	[2]	N	[3]	255	[1]	N	50	N	[4]	N	N	N	N
Are undercuts and recesses: Separately roughed, left to the Finishing cutter, Choice, Integral with roughing cut?	I		S	C	C	C	C	C	C	F	S	C	C	S	C
Does the system automatically make a clean-up profile roughing pass? Yes, No	Y	N	Y	N	N	N	N	N	Y	N	N	N	N	N	Y
Initial or raw stock boundary definition: What kinds of line segments may be used in the definition?															
Straight lines Yes, No	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Circles Yes, No	Y	N	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	N
Other curves Yes, No	N	N	N	N	N	N	N	N	N	N	N	Y	[8]	N	N
Inner boundary of automatic roughing area: What kinds of line segments may be used in the definition?															
Straight lines Yes, No	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Circles Yes, No	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Other curves Yes, No	N	Y	N	N	N	N	N	Y	N	Y	N	Y	[8]	N	N
Can a blend radius be substituted for a circular line segment definition statement? Yes, No	Y	N	Y	Y	N	Y	Y	N	Y	N	Y	N	N	Y	Y
Can the stock allowance left for the finishing be specified as: all surfaces the Same, Differently x and z surfaces, each surface Independently?	I	I	[5]	I	I	S/D	I	S	I	I	S	D	S	I	D

Table is continued on the next page.
Footnotes on the second page following.

TABLE 11B

SIMPLIFIED PROGRAMMING MEANS

Automatic Roughing, Continued

	Cincinnati	Digital	Encode	General Electric	Ingersoll	M D S I	M S & S	McAuto	Olivetti	S D R C	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
Will the system make parallel roughing cuts — No, to the Left, to the Right, Either?	E	E	L	L	E	E	E	E	E	E	E	E	E	E	E
Will the system make perpendicular roughing cuts — No, In, Out, Either?	E	E	N	E	E	E	E	E	E	E	E	E	E	E	E
Will the system make roughing cuts on an angle — No, to the Right, to the Left, Either?	N	N	N	N	N	N	N	N	N	N	N	E	N	N	N
Does the System or the Programmer choose between parallel and perpendicular?	P	P	na	P	P	P	P	P	P	P	P	P	P	S	P
Can both be used on one part? No, yes in a Fixed sequence, yes in Optional sequence	O	O	na	O	O	O	O	O	O	O	O	O	O	O	O
How is the direction (L/R, In/Out) of cut determined: Automatically, Specified?	S	S	na	A	S	A	S	S	S	S	S	S	S	S	S
How is the first-cut depth determined: Specified, Automatically calculated, Choice?	C	C	S	A	A	S	S	S	S	S	C	S	S	S	S
How is the number of roughing cut passes determined: Specified, Automatically calculated, Choice?	C	A	A	A	A	C	A	S	A	C	C	A	A	A	S
Can a minimum depth of cut be specified? Yes, No	Y	N	N	N	N	Y	Y	Y	N	N	Y	Y	Y	N	Y
Will the system optimize the distribution of depth of cut over the several cuts? Yes, No	Y	N	N	Y	Y	Y	Y	N	(6)	Y	Y	Y	Y	Y	N
Will the system change spindle speeds automatically as roughing proceeds? Yes, No	Y	N	(5)	Y	Y	Y	(7)	Y	Y	Y	Y	Y	Y	Y	Y
Can specific speed rates be prescribed for certain surfaces? Yes, No	Y	Y	(5)	Y	Y	Y	(7)	Y	Y	Y	Y	Y	Y	Y	Y
Is the pull out from cuts: Perpendicular to the axis, along the part Boundary, at an intermediate Angle?	A/P	B	P	(9)	B/P	A/B	A	A	A	A	PBA	A	A	A	A/B

Footnotes on the next page

FOOTNOTES TO TABLES 11A, 11B, AND 12

(1) System provides for 1496 point definitions from which to compile all the boundary definitions, including circles. Only 30 circles are permitted.

(2) Limit is 25 in 16-bit computers, 40 in 32-bit computers.

(3) Total segments for all boundaries = 75.

(4) Limit 3 segments plus 2 blends, or 2 segments plus 1 circle and 1 blend.

(5) This feature may be achieved by breaking up the boundary into several smaller boundaries.

(6) Automatically takes a minimum number of cuts, all of the same depth.

(7) System can interrupt a roughing sequence to change speeds or feeds without a tool change statement.

(8) Mathematically defined and tabulated curves may be handled by inserting standard APT statements.

(9) System permits the user to override the highly structured technology incorporated in the processor. This applies to all entries in this column in this section of the report.

series, even though they are defined in a different manner from the other elements. Line segments may be straight lines, circles, or other curves (mathematically defined or tabulated).

A single part may require more than one finishing cut sequence. For example, a part may require an external finishing cut with one tool and an internal finishing cut with another tool. Or the external finish may be done in two separate sequences, for example one for cylindrical surfaces and one for faces, with the same or different tools.

Not all features of the ultimate finished part are reflected in the surfaces made by finishing tool sequences:

1) Grooves, necks, etc., are usually omitted as beyond the capability of the finishing tool.
2) Surfaces to be threaded usually are roughed to size but usually receive no finishing cut before threading.

Undercuts or recesses may be finished either:

a) without prior rough undercutting, if shallow, or
b) after a separate undercut roughing sequence, if deep.

Undercut finishing may be done in the proper sequence in an overall finishing cut, or in a separate mini-sequence if a tool change is required by the shape of the undercut. See the section on Undercuts included in Automatic Roughing Routines (above).

Depth of Cut

A single finishing cut pass is normal, so that the finish cut depth is determined by the stock allowance left by the roughing cuts. Speeds and feeds are calculated to produce the specified surface finish and dimensional tolerance required.

Direction of Cut

The finishing cuts proceed from the specified start point toward the specified finish point, following the finish part profile in a contouring mode. By breaking the total finish contour into segments, direction of cut on any portion can be controlled.

Finishing as a Part of Automatic Roughing

Because the finished part profile is the usual reference used as the inner boundary of an automatic roughing routine, many systems use the same finish part profile in both the automatic roughing and in the finishing contour procedures. It may be thought of as a pattern which may be called in both routines.

The system may be so designed that the finish cut must be generated by a separate finish statement. Obviously, if the finished profile includes a tabulated curve which has been replaced in the automatic roughing routine calculations by an approximation of straight lines and circles to simplify those calculations, then the true finished profile pattern must be specified for the control of the finish cut contour.

TABLE 12

SIMPLIFIED PROGRAMMING MEANS

Cutting Along a Profile

	Cincinnati	Digital	Encode	General Electric	Ingersoll	M D S I	M S & S	McAuto	Olivetti	S D R C	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
Can this capability be used for Finishing, Roughing, or Both?	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B
How many different boundaries may be used in one part program? No limit, a number X	N	100	(2)	N	3	20	(1)	N	2	N	N	N	N	N	2
How many line segments may be used in one boundary? No limit, a number X	N	100	(2)	N	3	255	(1)	N	50	N	N	54	N	N	40
What kinds of line segments may be used in the description? Straight lines Yes, No	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Circles Yes, No	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Other curves Yes, No	Y	Y	Y	N	N	Y	Y	Y	N	Y	Y	Y	(8)	N	(8)
Fillets Yes, No	Y	N	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y
Chamfers Yes, No	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Round corners Yes, No	Y	N	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y
May certain segments be skipped by the finishing tool: Unable, use Skip statement, Break up the profile	S	B	B	S	B	S	B	B	B	U	B	B	B	B	B
Will the system adapt spindle speeds to the diameter at which it is cutting? Automatically, Specified	A	S	S	A	A	S	S	A	A	A	S/A	A	A	A	S
Does the system check for interference of the back angle of the tool? Yes, No	N	N	N	Y	Y	N	N	N	Y	N	N	N	N	N	N
Does the system provide means for automatically breaking all sharp corners? Yes, No	Y	N	N	N	N	Y	N	N	N	N	N	N	N	Y	Y

Footnotes on Page Roughing 8

It is conceivable that a part will require a roughing cut, followed by a finishing cut, but that an automatic roughing routine is unnecessary or inappropriate; a simple end-to-end roughing cut, followed by an end-to-end finishing cut, is all that is required. In this event, both cuts are thought of as contouring cuts, in which a tool point is driven along the connected line segments of the contour, with the appropriate stock allowance (or no allowance). The roughing cut control technology is then identical to that for finishing, described above.

It is also possible that a part will require grinding after turning, in order to achieve the desired tolerance. In that case some stock must be left by the turning operation for removal in the grinding operation. The part dimension as shown on the drawing, plus this grinding stock allowance, is the dimension used for the finished *turning* profile.

C. Threading

Threaded parts may be produced on lathes; the part rotates and the cutting tool does not. Screw threads may be required on either internal or external surfaces; they may be cut into cylindrical or into tapered surfaces. Thread-like forms may also be required for use other than as fastenings: for lead screws; for materials feed screws, either cylindrical or conical; or for scroll cam slots such as found in three-jaw chucks, cut in a face perpendicular to the axis.

Threading tools can always approach the start point of the thread by moving parallel to the axis, without intercepting any portion of the work piece. If this were not so, a nut could not be assembled to the thread. (See Figure 10.) However, clearance for the threading tool to run out of the cut at the other end of the thread is another question. Good design will provide a clearance space between the end of the thread and the next shoulder, with a slight groove between them, for the cutter run-out. This is not always possible, and special provision must be made in such cases for retracting the cutter from the thread surface. (See Figure 10.)

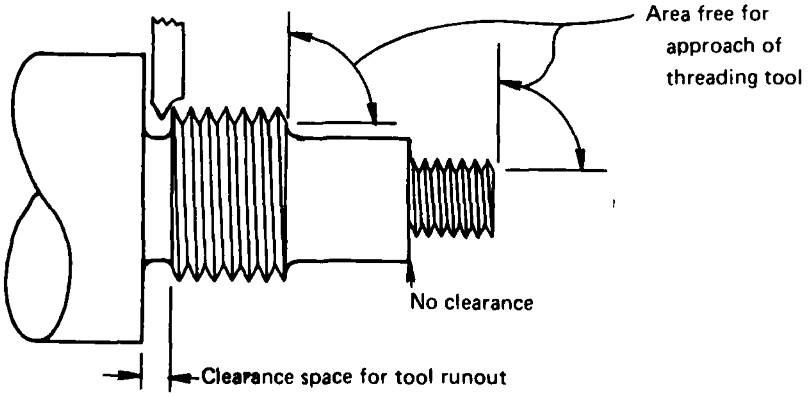

Area free for approach of threading tool
No clearance
Clearance space for tool runout

FIGURE 10 CLEARANCE AROUND THREADS

Threads may be cut either by multi-point tools (taps and dies) or by single point lathe tools. The threading procedures followed for the two types of tools are quite different, and so therefore are the programming techniques. They are in fact different enough so that they should be considered separately.

Single Point Threading

Threads may be cut in a lathe on any surface of a part — external or internal, cylindrical, conical, or end face — with single point cutting tools. They may be given any desired tooth profile by properly shaping the single cutting point and guiding it to the correct depth of cut. Multiple passes of the cutting point are generally required to produce the desired depth of cut, surface finish, and dimensional accuracy. A single point cutting tool may be used to produce threads on any diameter, of any length, right or left handed, single or multi-start, and of any profile — V, Acme, buttress, etc.

During the threading operation, the lead screw moves the cutting point axially during each revolution of the work piece, by an amount called the lead. The cutting point is driven along the length of the threaded portion a number of times, cutting a little deeper each time until the desired finish depth is reached. After each pass the tool must retract to a safe distance above the work, move rapidly back to the start point, and reapproach the work. The manufacturing engineer has a wide variety of options in programming this sequence and pattern of motions.

Depth of Cut. In some materials it is desirable that the initial cut exceed some minimum amount, so the "first cut depth" may be prescribed. Similarly, a small final cut may be desired, so a "finish cut depth" may be prescribed. The number of intervening passes is a function of the metal cutting technology; the more cuts, the longer the expected tool life, and the longer the threading operation time. The radial increments for the tool setting after the first pass can be handled in two ways:

1. Make all radial increments equal, which results in increasing chip removal volume by successive passes.
2. Make the radial increments smaller as the work progresses, thus approximating equal chip removal volume on each pass. (See Figure 11.)

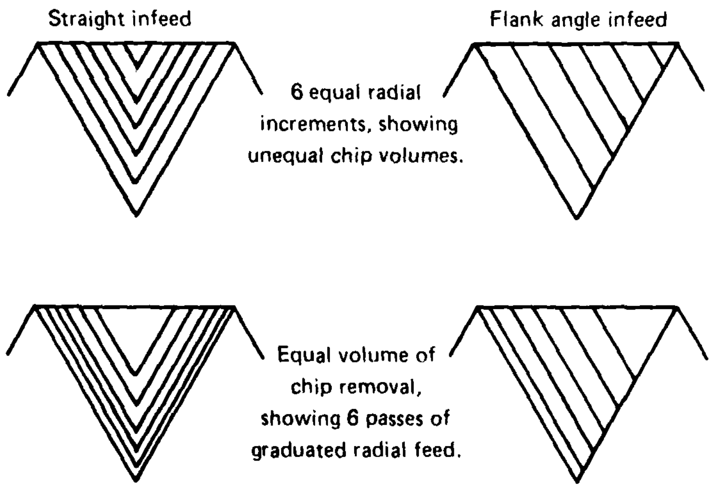

Straight infeed
Flank angle infeed
6 equal radial increments, showing unequal chip volumes.
Equal volume of chip removal, showing 6 passes of graduated radial feed.

FIGURE 11 THREADING CUT INCREMENTS

Spring passes. For fine finish and close tolerance work, one or two very light cuts may follow the "finish cut" pass. The very light tool pressure virtually eliminates spring in the cutting tool/lathe/ work piece system, thus increasing accuracy.

Gaging passes. The programmer may require the work to be stopped for dimensional check by the operator using a micrometer, before the work is adjudged to be complete.

Angle of approach to the work. On successive passes, the starting point of the tool tip takes a series of positions along a line:

1. Perpendicular to the work piece surface, thus balancing chip load on both sides of the cutting point. (See Figure 12.)

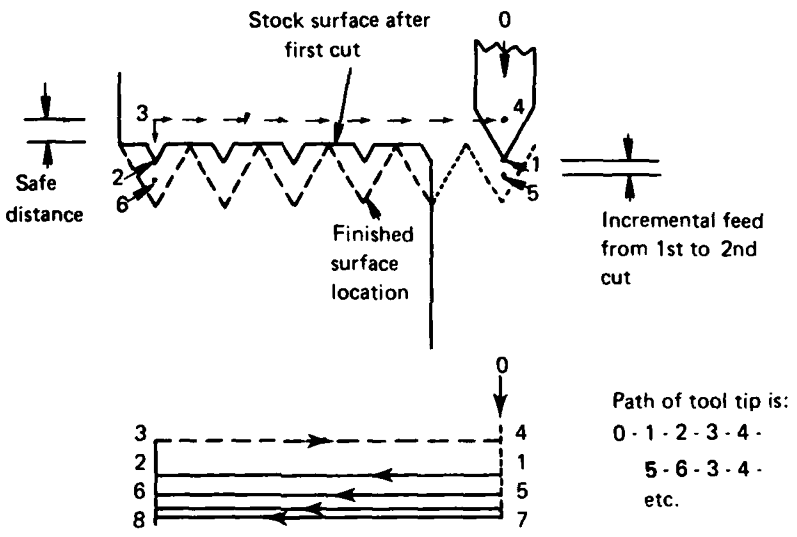

Stock surface after first cut
0
3
4
1
5
2
6
Safe distance
Finished surface location
Incremental feed from 1st to 2nd cut
0
3
2
6
8
4
1
5
7
Path of tool tip is:
0 - 1 - 2 - 3 - 4 -
5 - 6 - 3 - 4 -
etc.

FIGURE 12 STRAIGHT APPROACH AND RETRACT

2. Nearly parallel to one flank of the tooth profile being generated, thus putting all cutting on one edge of the tool point. Usually the path is 1/2° to 1° less than the flank angle. (See Figure 13.)

3. At some intermediate angle.

Angle of retraction from the work. When there is a clearance space between the end of the thread and the next shoulder on the work piece, the tool tip may be retracted along a line parallel to the tooth profile, and in so doing enter this clearance space. When there is no clearance space, the tool must retract along a line perpendicular to the axis. (See Figures 10 and 12.)

Lead and pitch. During the threading operation, the lead screw moves the cutting tool axially by an amount called the "lead," during each revolution of the work piece. The pitch is the axial distance from the point on one thread profile to the point on the next profile. A "single"

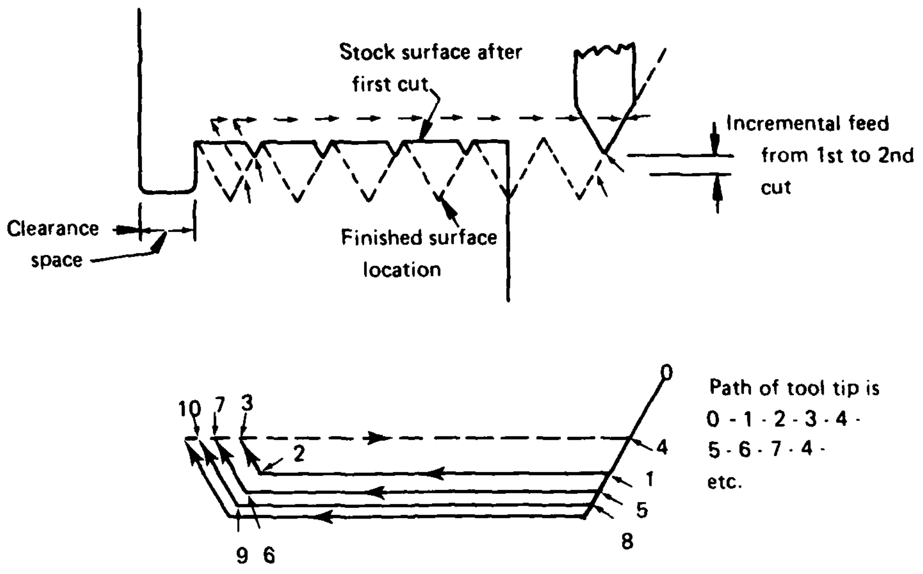

Stock surface after first cut
Incremental feed from 1st to 2nd cut
Clearance space
Finished surface location
Path of tool tip is 0 - 1 - 2 - 3 - 4 - 5 - 6 - 7 - 4 - etc.
10 7 3
2
4
1
5
8
9 6
0

FIGURE 13 FLANK ANGLE APPROACH AND RETRACT

thread is a thread in which the lead is equal to the pitch. In a "two-start" or "double" thread, the lead equals two times the pitch; in a three-start" or "triple" thread, three times the pitch, etc.

Multi-start threads. Cutting multi-start threads with a single point cutting tool may be done in either of two sequences:

1. Complete the cutting of one start to finished depth; then go back and cut the next start from top to finished depth, etc.
2. Set the cutting point for the first cut depth; cut one pass from each start; reset the tool to the next depth increment, and cut a pass from each start; repeat until all passes are cut to full depth.

Right and left hand threads. Conventionally, with the work rotating so that the top comes toward the operator, right hand threads are cut by moving the tool from tail stock toward the head stock. Left hand threads may be cut by reversing the direction of tool feed, or by reversing the spindle. The latter calls for a different tool configuration.

For the NC programmer, single point threading requires him to select and specify the technology he wishes to use.

1. The correct tool is called.
2. The tool is moved rapidly to close proximity to the work.
3. The tool approaches the work along a line at a prescribed angle, to the first cut depth.

4. The tool chases along the length of the thread to the retract line.
5. The tool retracts to a clearance plane along a line at a prescribed angle.
6. The tool is moved rapidly back to the approach line used in step 3. Steps 3, 4, 5, and 6 are then repeated at a succession of increasing tool penetrations to the final depth.
7. The tool is moved rapidly back to the tool change position.

Each of the foregoing may be written as a separate statement, or the entire operation may be capable of being called in one or two statements. (See Figures 12 and 13.)

Multi-point Threading

Internal screw threads in drilled holes may be produced by taps, while external screw threads on turned diameters or tapers may be produced by threading dies. A single pass of the tap or die produces a complete threading operation. Both types of tools have three or more cutting elements to engage the work piece. Each cutting element has a series of sharp cutting points, spaced apart by the pitch length of the thread and shaped to produce the desired thread tooth profile. A special tap or die is required for each combination of nominal thread diameter and pitch, right or left hand, and for each tooth form.

The tap or die feed and the spindle rotation must be correlated by the lead of the thread. Naturally, both tool feed and spindle rotation must be stopped in phase at the correct depth, and both must be reversed in phase to back the tap out of the threaded hole it has just produced, or to disengage the die from the work piece it has just threaded.

Taps and dies may be solid, or may be built as collapsing taps and self-opening die heads. When using the solid taps or dies, the work piece is rotated and the tool is advanced until the desired length of thread has been produced. Then the spindle must be stopped and reversed, thus backing the tap or die out of engagement with the work.

The collapsing taps and self-opening dies are shells containing inserted cutting elements called chasers. When the desired length of thread has been generated, the chasers are retracted radially from engagement with the work, and therefore the tool head may be withdrawn without having to stop and reverse the spindle. Retraction of the chaser is automatically triggered by adjustable stops.

Collapsing taps and quick opening dies are tools normally used in high production machine tools such as multiple spindle chuckers or bar lathes, where the seconds saved by not having to stop and reverse the spindle can mount up to significant savings. Such uses on NC lathes are so unusual that we omit these capabilities from the matrix.

For the NC programmer, tapping and die threading are fairly simple tasks to program:

1. The correct tool is called.
2. The tool is moved rapidly to close proximity with the work.
3. The tool is fed until the desired thread length is cut.
4. The spindle and tool feed are stopped.
5. The spindle and tool feed are reversed until the tool clears the work.
6. The tool is moved rapidly back to the tool change position.

TABLE 13A

SIMPLIFIED PROGRAMMING MEANS

Threading

	Cincinnati	Digital	Encode	General Electric	Ingersoll	M D S I	M S & S	McAuto	Olivetti	S D R C	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
Will the system program:															
Threads on cylindrical surfaces? Internal, External, Both, Neither	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B
Threads on tapered surfaces? Internal, External, Both, Neither	B	B	B	B	N	B	B	B	B	B	B	B	B	B	B
Threads on an external face? Yes, No	Y	Y	Y	N	N	Y	Y	Y	Y	Y	Y	Y	Y	N	Y
Variable pitch threads? Yes, No	Y	N	N	N	N	Y	Y	N	N	N	N	Y	Y	N	N
Multi-start threads? No, method 1 (see page 6), method 2, Choice	2	C	2	C	2	2	C	1	2	N	N	2	2	2	2
Single point cutting, Multi-point cutting, Both	B	S	B	S	B	B	B	B	B	B	S	B	B	S	B
For *single point* threading tools:															
How is the first cut depth determined? Specified, Automatically generated, Choice	A	C	S	A	A	S	C	S	A	S	C	S	S	C	S
How is the final cut depth determined? Specified, Automatically generated, Choice	A	C	S	A	A	S	C	S	A	S	C	S	S	C	S
How are depth increments made: all Equal, Graduated depth, equal Volume of chips, Choice	G	C	C	E	G	C	G	C	V	C	C	C	C	C	G
How are the number of passes determined? Specified, Automatically calculated, Choice	C	S	A	C	A	C	S	S	S	S	C	A	A	C	A
How are spring passes handled? Specify depth and number, No special provision	S	S	S	N	S	S	S	S	S	S	S	S	S	S	S
How are gaging passes handled? Specify depth and number, No special provision	S	S	N	S	S	S	S	S	N	S	S	S	S	S	S

Continued on next page

TABLE 13B

SIMPLIFIED PROGRAMMING MEANS

Threading, Continued

	Cincinnati	Digital	Encode	General Electric	Ingersoll	M D S I	M S & S	McAuto	Olivetti	S D R C	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
For *single point* threading tools, continued:															
What are the options on angle of approach? Perp. to the axis, Flank angle, Other, Choice	C	C	C	C	F	C	C	C	F	C	C	C	C	C	C
What are the options on angle of retraction? Perp. to the axis, Flank angle, Other, Choice	C	C	C	C	F	C	C		C	C	C	C	C	C	C
How is the left hand threading done: reverse the Direction of cut, reverse the Spindle, Choice	C	C	C	C	S	C	C	C	C	C	C	C	C	C	C
For *multi-point* threading tools:															
Will the system program tapping? Yes, No	Y	Y	Y	[1]	Y	Y	Y	Y	Y	Y	N	Y	Y	N	Y
Will the system program die cutting of external threads? Yes, No	Y	Y	Y	[1]	Y	Y	Y	Y	Y	Y	N	Y	Y	N	N

[1] Multi-point threading may require an inserted routine.

Not infrequently, a canned cycle or a macro will create this entire sequence from one statement in the program manuscript.

D. Grooves and Necks

A groove in a turned part is a narrow and relatively deep recess in the part surface, made by a plunge cut or cuts. Common usage applies the term "groove" to a re-entry cut wholly within a cylindrical, conical, or face surface (Figure 14), and the term "neck" to a re-entry cut at the junction of two surfaces, such as a face and a cylinder (Figure 17). Grooves may be employed for holding O-rings, piston rings, snap rings as well as many other design functions. Necks may be used to increase the radius of the corner between two intersecting surfaces where a fillet might interfere with a square-edged component (Figure 17-3). Also necks serve as a clearance space for grinding or threading tools to run out into (Figure 18) and for many other design purposes.

A plunge cut tool may have parallel sides (Figure 14-1), or slightly relieved sides (Figure 14-2), or it may have a tip configuration of any desired shape (Figure 15-2).

Grooves are basically straight-sided, flat bottomed recesses, but are infinitely variable in minor detail.

- Bottom corners will of course have the radius of the tool nose corners (sharp corners being avoided as stress raisers). A radius larger than the tool nose radius at the bottom corner of the groove calls for a contouring motion, and hence a groove width greater than the tool width (Figure 14-4).
- Upper corners may be chamfered (Figure 14-5) or rounded (Figure 14-6); the corner arc is usually tangent to the side wall of the groove and to the host surface.
- Side walls may be perpendicular to the host surface (Figure 14-1 and 2) or slightly angled (Figure 14-7). For perpendicular side walls a straight sided tool may be used (Figures 14-1 and 2). For angled side walls, a compound tool motion or a form tool is required.
- A semicircular bottom may be made with a round nosed tool of the correct radius (Figure 14-8).
- Grooves of any desired profile may be made using a specially shaped tool (a "form" tool) and making a plunge cut (Figures 14-9 and 10).

Grooves may be left with the surface finish produced by the grooving tool, or the groove may be made with a roughing cut to the approximate size, followed by a finishing cut on side walls, bottom, or both. The same tool may be used both for roughing and for finishing with appropriate feed rates. Rounded or chamfered corners are usually made by a contouring motion of the basic grooving tool. Wide grooves may be made with a narrower tool by repeated plunge cuts, each overlapping the prior by a small amount (at least twice the radius of the tool nose corners). (See Figure 14-3.)

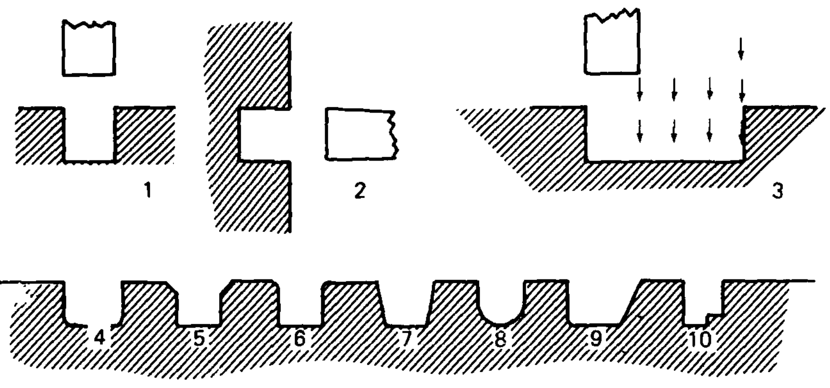


FIGURE 14 PLUNGE CUT SHAPES

Grooves may be required on any surface — cylinder, taper, or face; internal or external; but rarely on a surface whose profile is a circular arc or other curve. Usually one corner of the basic rectangular groove profile is used as the reference point for locating the groove relative to the rest of the part design (see Figure 15-1) and one corner of a square nosed tool point, or the center of a round nosed tool point, is used as the reference point in locating the tool relative to the turret. (See Figure 15-2.) Groove dimensions may be conveniently given with reference to the host surface, the width and depth being the basic parameters. If dimensioned relative to the part axes, the reference corner and the bottom dimension are the basic parameters. (See Figure 15-3.)

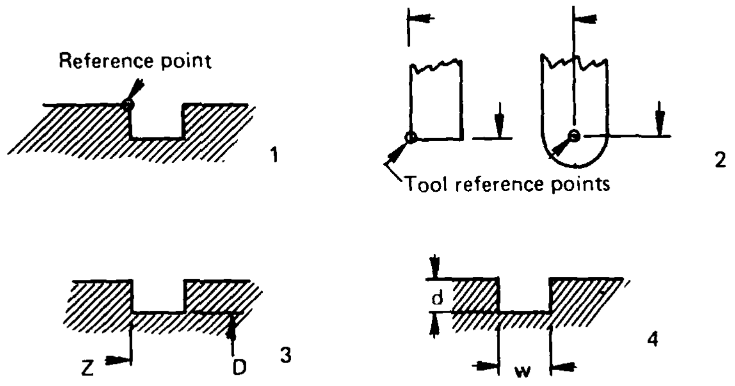
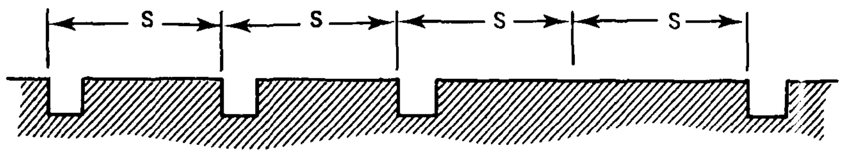


FIGURE 15 PLUNGE CUT REFERENCE POINTS AND DIMENSIONING

Grooves may be repeated at several points in a host surface at spaced intervals, and an NC system may have the capability of machining such a series by a reiteration of the instructions for the first groove of the series. Additionally, some systems have the capability to omit one or more designated grooves from the regularly spaced series. (See Figure 16.)



" 5 groves spaced s inches apart, omit #4 "

FIGURE 16 MULTIPLE GROOVES

Necks at the junction of a face and a cylinder, for example, may be cut with one side a prolongation of the cylinder (Figure 17-1), or one side a prolongation of the face (Figure 17-2), or may be plunged at an angle to both (Figure 17-3). Specially shaped necking tools are required. Necks meant to provide clearance for threading or grinding may or may not undercut the adjoining face, but frequently have a fillet tangent to the face (Figure 18-2) and an angled side adjoining the threading area. Round nosed finishing tools with a proper back angle may be used, in which case a plunge cut is not made, but instead the neck is created as if it were a shallow undercut or recess.

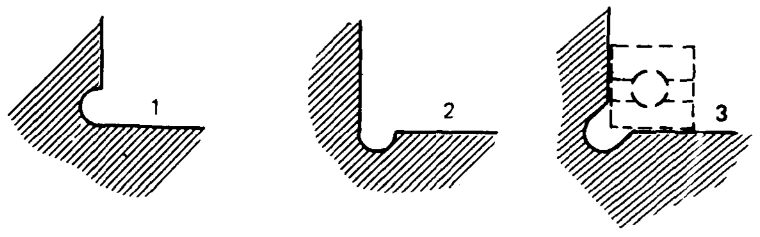


FIGURE 17 NECK CUT SHAPES

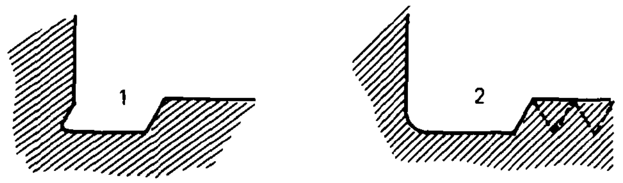


FIGURE 18 TOOL RUNOUT GROOVES

NC lathe programming systems may place certain restrictions on groove or neck designs in order to simplify the macros used in the programming. Among such arbitrary constraints may be the following:

— The groove shape must be symmetrical about its centerline.

— Top and bottom corner radii together must not exceed the depth of the groove.

TABLE 14

SIMPLIFIED PROGRAMMING MEANS

Plunge Cuts for Grooving

	Cincinnati	Digital	Encode	General Electric	Ingersoll	M D S I	M S & S	McAuto	Olivetti	S D R C	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
Will the system program:															
Contoured radius at bottom corner of groove? Yes, No	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	N	Y
Contoured radius at top corner of groove? Yes, No	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	N	Y
Chamfer at top corner of groove? Yes, No	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	N	Y	Y	N	Y
Angled sides of groove (compound motion)? Yes, No	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	N	Y
Will the system handle combinations of the above? No, Some, All	A	S	A	A	A	A	S	A	S	A	S	A	A	N	A
Does the system provide for:															
Plunge cuts only, plunge and Finish cuts on sides and/or bottom of groove?	F	F	F	F	F	F	F	F	P	F	P	F	F	P	F
Multiple plunges for grooves wider than the tool? Yes, No	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Will the system program repeat grooves at uniformly spaced intervals? Yes, No	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Will the system omit designated grooves in a series of grooves? Yes, No	N	N	Y	N	N	Y	N	Y	N	N	N	N	N	N	N
Will the system make necks at the junction of two surfaces by plunge cuts: parallel to the Z axis, to the X axis, to Both the X and Z axes, at an Angle, all Three, None of the three?	T	T	T	T	T	N	N	T	B	X	T	T	T	N	T

— Neck tools may have a radius on one bottom corner only.

— Deep plunges on the first pass may be broken by dwells, to clear chips.

— A dwell may be called for at the bottom of a groove to even the diameter.

— Finish cuts on the bottom of wide grooves are always to be cut in one direction.

— Series of grooves are cut in sequence in one direction.

Not all the above constraints may be present in one system. They may of course be evaded if necessary by abandoning the macros and simply writing a set of tool control statements in the basic programming language. If a system does not provide for round corners, a form tool may be used with the appropriate corner formation, and a simple plunge cut made, thus achieving the same end result.

E. Drilling, Boring, Reaming

Lathe work held in a chuck may be operated upon by tools such as drills, boring bars, end mills, reamers and taps. These tools are held in the turret without rotation and presented to the rotating part along the axis of rotation. The tools are advanced into the work and then retracted in a relatively simple tool movement pattern.

Drills and end mills may be used to create round straight-sided holes in a piece of stock. For shallow holes they may be fed at the correct rate of penetration and to the desired depth, and then retracted and returned to the home position without stopping the spindle. The size of the hole produced is controlled by the tool diameter.

Boring bars may be used to enlarge holes already in existence. The size of the finished hole is independent of the diameter of the boring bar. Better roundness, more accurate hole diameters, and smoother finishes may be achieved by use of the boring tool as compared to a drill. Also the holes need not be straight-sided, but may be contoured, stepped, tapered, or otherwise shaped.

Reamers following drills will also produce straight, smoother, rounder, and more accurately dimensioned holes than will drills alone.

There are few control problems in this type of routine. Rapid approach of the tool from the home position is stopped a short distance from the expected work surface, and speed and feeds are governed by the tool and the material characteristics.

Deeper holes (over two or three diameters) may be drilled by a pecking cycle. The tool advances into the work — about two diameters for example — retracts from the hole to clear the chip accumulation, rapids to a point close to its deepest penetration, and then feeds for another distance. The pecking cycle is repeated until the proper depth is reached, or the work is fully pierced. When a peck drill cycle is used, the parameters of first penetration and subsequent incremental penetration may be fixed by the cycle routine, or may be controllable by the programmer.

Table 15 compares the routines of this type available in the various systems.

TABLE 15

SIMPLIFIED PROGRAMMING MEANS

Drilling, Boring, and Reaming

	Cincinnati	Digital	Encode	General Electric	Ingersoll	M D S I	M S & S	McAuto	Olivetti	S D R C	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
Will the system program peck drilling cycles? Yes, No	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y
Are the first and later peck cycle depths — Fixed, Controllable?	C	C	F	C	C	C	C	C	F	C	na	C	C	C	C

CHAPTER VII

DISTINGUISHING CHARACTERISTICS

The fifteen systems covered by this study differ from one another by certain characteristics. While some of these characteristics are apparent from an examination of the matrices in the previous three chapters, others by their nature require separate mention. This chapter contains a section on each language to collect and identify these distinguishing characteristics.

People who design and develop NC programming languages and processors incorporate in their systems such capabilities and characteristics as seem to them most desirable to meet the needs of the users in that portion of the total market place which they hope to serve. These differences between the systems are the results of two factors: 1) conscious design decisions, and 2) compromises inevitable in the design of any system.

As mentioned before (see page 3) each system is directed at some portion of the market, selected by the proponent as an attractive business opportunity, and as compatible with the system's technical structure. The fact that the proponents have selected different targets is an interesting tribute to independent initiative in the business world. It certainly contributes to the variety of choice offered to a potential user.

Not only have the proponents selected varying targets, but their success in meeting the demands of those separate market areas varied. Any system design must be the result of many compromises during the developmental stage, between what would be desirable and what can be afforded. New systems suffer most from the residuals of these inevitable compromises; older systems have had time to work around the constraining factors.

Also, as mentioned before (see page 17) it is not possible to reduce the findings of this study to a single "figure of merit." This is particularly true in certain evaluations of a subjective nature, such as the evaluation of the various programmers manuals (see page 14). It is equally true in such concrete questions as cost of a system. Most systems, like most makes of automobiles, come in several levels of elegance. We find costs quoted for a "basic system," an "enlarged system," and a "full system." And each level is subject to the addition of a variety of optional peripherals. This report can only cite prices for combinations quoted by the vendor. Note that these prices are subject to change since our receipt of the data.

A section follows for each system, following the alphabetical order shown in Table 1, page 13.

A. Cinturn II

Cincinnati Milacron, Inc.'s Machine Tool Group of Cincinnati, Ohio offers the Cinturn II system for the programming of lathe parts. It is usable only with Cincinnati Milacron lathes, and is admittedly used as a sales aid for machine tools in the form of a special service to its customers. It is available as a remote system accessed through one of several network services (General Electric, University Computing, and Westinghouse), or the software can be purchased for use on an in-house computer such as the IBM 370, the Univac 1108, or the Honeywell 6000 series.

General Description

The Cinturn II system is highly structured and automatic. There are subroutines for carrying out almost any conceivable lathe operation. By virtue of the machine sequencing files that have been pre-arranged, one short statement can control a goodly number of parameters and operations. For example, the statement "CALL/RT" will handle a complete roughing operation. The selection of the tool, the tool's motions, depths of cut, feeds and speeds, and other machining functions are automatically determined, based upon a previous statement describing the specific machine tool and the material to be cut. Any of these pre-arranged parameters may be overridden by the part programmer. While the override would increase the detail required of the programmer, it does allow for flexibility when required. Indicative of the extent of routines available is the automatic changing of a tool for a new one when the preprocessor calculates that the old tool has become dull.

Comments on the Processor

In order to facilitate the automatic operation of Cinturn II it is necessary that detailed tooling files and machine tool characteristics be pre-recorded. This has been done by Cincinnati Milacron, although the programmer may establish his own files without too much difficulty by following instructions in the part programmer's manual.

The simpler geometric forms involving lines and tangent circles may be described by noting the corner points of the geometric form. A corner point is described as the distance from the origin along the Z axis, and the diameter. Other more complex geometric forms involving lines intersecting arcs, circles tangent to circles, splined curves and mathematically defined curves are handled by using conventional APT geometry statements. This is possible since Cinturn II is an APT based system, and APT statements may be intermixed with Cinturn II statements.

Actually, Cinturn II may be considered as a preprocessor which converts Cinturn II statements into basic APT statements, which are suitable for the APT processor.

Programming Manual Evaluation

Cincinnati has produced a good manual. It is highly detailed, which may cause some confusion on first review. However after some careful study and an understanding of the many symbols involved, the approach is more clearly appreciated and understood. There are a goodly number of examples which are clearly defined and illustrated.

Prices Quoted

For local processing: Software:

Cinturn II system for IBM computers	$5950.00
Cinturn II system for Univac and Honeywell Computers	6950.00
Postprocessor for 2-axis lathe, IBM	3000.00
Postprocessor for 2-axis lathe, Univac and Honeywell	3500.00
Postprocessor for 4-axis lathe with full merge capability, IBM	6990.00
Postprocessor for 4-axis lathe with full merge capability, Univac and Honeywell	7490.00

For local processing: Hardware: obtain from vendor.

For remote processing: Service is obtained through a service network such as General Electric, University Computing, or Westinghouse. See prices quoted in later portions of this chapter.

B. Quick-Path

Quick-Path is a numerical control tape preparation system which is offered by Digital Systems Corporation, Monroeville, Pennsylvania. It is designed to prepare tapes for two or three axis machines. The system resides on a mini-computer system composed of a Digital Equipment Corporation (DEC) PDP 8 with 16K words of memory, an ASR-33 I/O device, and the associated Quick-Path software. Additional peripherals such as plotters, disk storage, high-speed tape read/punch, high-speed printers, high-level programming languages (i.e., BASIC, Fortran) are offered as options.

General Description

The Quick-Path software is interactive and therefore provides the part programmer with immediate response by processing input data on a statement basis. Appropriate error messages are initiated by the system at the time that the user inputs invalid data. Quick-Path provides a method for the definition of part geometry and tool path.

The geometry feature permits the programmer to define part boundaries with lines, points, and circles by using numerical assignments. Each geometry element has a unique number which the programmer uses in the statement definitions. Modifiers are available and permit the selection of the position of the geometry (i.e., "+" indicates the circle above the line, "−" indicates the circle below the line).

The motion of the tool around part boundaries is controlled by the path input. Tool motion is specified in an absolute coordinate system and by specifying the sequence of geometry elements that define the path, the cutter is directed along the boundaries through the use of terminators.

Comments on the Processor

The Quick-Path processor is highly structured and requires the part programmer to utilize numeric codes when specifying geometry or tool path data. All conceivable geometries which can be accommodated by Quick-Path have been reduced to numeric representations. For this reason the processing of the part programming data is comparatively fast in spite of the system being resident on a relatively small and slower mini-computer. The same technique (i.e., numeric representation) is utilized by the postprocessors.

Programming Manual Evaluation

Digital provides a part programmer's manual; however, this manual is quite simplistic and serves primarily as a reference manual. Digital Systems Corporation prefers that their customers attend the formal training which is provided and recognizes the limitations of the manual.

The manual contains a brief overview of the system, codes, examples, and self testing sections.

Prices Quoted

Standard Unit	**Total**
1 PDP8A model computer with 16K words of core memory	
1 33ASR teletype with paper tape reader/punch and interface	
1 30" high walnut top computer stand	
1 set Quick-Path language software and manuals	
1 machine tool postprocessor	
	$14,975

Plotter Option	**Total**
1 Tektronix model 4006 graphic display terminal (alternate Hewlett Packard hardcopy graphic plotter with 15 x 10" plotting surface)	
1 interface and cable to plotter	
1 additional training and warranty module for plotter	$ 4,495
High-speed paper tape punch	
1 Facit model 4070 tape punch 75 cps mylar	
1 parallel interface to punch	
	$ 2,800
High-speed paper tape reader	$ 1,300

C. GENESIS

Encode, Inc. of Newburyport, Mass. offers GENESIS as a general purpose local processing NC programming system. It operates on a dedicated stand-alone mini-computer and an ASR 33 Teletype terminal. GENESIS will program for machining centers as well as for lathes. We discuss particularly those portions of GENESIS applicable to lathe work.

General Description

The GENESIS language resembles ADAPT in that the geometry is first defined, element by element, and then tool motions are called for step by step. GENESIS also resembles COMPACT II in that the parametric values may be entered in a statement in any sequence, providing they are labeled — e.g. "Z 1.250."

GENESIS operates in either Supervisory Mode or Run Mode. In the first, the programmer enters his instructions into the text memory, and gets a diagnostic check. In the Run Mode, the processor executes the program and delivers a machine control tape. A program may be written in Supervisory Mode, stored on a disk, and then run in Run Mode at a later time if more convenient. Postprocessors are not handled as separate computer passes. In the Supervisory Mode the operation is conversational. The processor will interrupt with diagnostics if necessary. The operator can enter statements via the keyboard or the tape reader, edit, add or delete material. In the Run Mode, the program as entered is processed through in batch mode to the machine control output stage. This batch operation includes the postprocessing.

Comments on the Equipment

GENESIS runs on a stand-alone mini-computer of either 16K or 32K memory capacity, using an ASR 33 Teletype terminal for communications between the operator and the mini-computer. Tape readers and punches may be added. A floppy disk unit may be attached for storing programs and postprocessors. A Tektronix Model 4662 plotter may be added. The system can be converted to work with Hewlett Packard Plotters.

Comments on the Processor

GENESIS has macros for roughing, finishing, undercuts, threading, grooving, drilling and tapping. A series of cuts such as used in grooving can be repeated by a special statement "DOLINE," and specific repetitions may be omitted from the repeating pattern by a statement "OMIT."

All parts are programmed using only points, lines, and circles to define the geometry, and in only a small variety of alternate formats — four for points, seven for lines, and two for circles. Profiles may be defined, using a series of CUT and/or CONTOUR instructions made in the same manner as if the boundary were being contoured; no tool action takes place. Thereafter the profile may be called as a boundary for roughing, or as a pattern for finishing cuts.

Programming Manual Evaluation

The GENESIS Lathe Programming Manual and its twin, the Milling Programming Manual, are both well written documents. Both are 120 page, loose leaf bound, printed on one side of good grade paper in clear easily read type. There are many good diagrams and sample programs. The Lathe Manual covers GENESIS language rules, supervisory commands, writing a part program, and geometry definitions and references. Tool motion instructions and part programming instructions are complete and lucid, and are specifically tailored to lathe programs. There is a summary of part programming instructions, arranged in alphabetical order, which serves as an index.

Prices Quoted

Equipment plus the Master Program:

System I — Basic System	$13,400.00
Teletype 10 cps reader/punch	
16K computer, Nova 3/4 core memory	
One master program	
System II	17,100.00
Teletype 10 cps reader/punch	
16K computer, Nova 3/4 core memory	
300 cps high speed reader	
Two master programs	
System III	22,900.00
Teletype 10 cps reader/punch	
32K computer, Nova 3/12 MOS memory, battery back-up	
Diskette system for master program and part program storage	
Two master programs	
System IV	25,750.00
Teletype 10 cps reader/punch	
32K computer Nova 3/12	
Dual Diskettes as above	
Two master programs	

All systems include cabinet, manuals, programming schools, and a 60 day warranty.

Software:

The processor is included in the system price
Postprocessors range between $1300 and $1850 per master program.

Optional extra equipment and accessories:

Tektronix 4662 Graphic Plotter, including software	$6,650.00
High Speed reader/punch with tape reel 300 cps read, 75 cps punch	3,725.00
Dasher 30 cps high speed terminal with reader/punch 300 cps high speed reader, 75 cps punch	5,485.00
Hewlett Packard Plotters, conversion to Encode system Model 7202A $1,800 Model 7203A	2,150.00

D. GETURN

General Electric Company's Information Services Business Division, Rockville, Maryland, offers GETURN for the programming of lathe parts. It was developed with the cooperation of the TNO Metaalinstituut of the Netherlands, and has been operational in Europe under the name MITURN since 1970. It is available on the General Electric Mark III Foreground (time-sharing) Network, a worldwide network using Honeywell 6000 Series equipment.

General Description

GETURN describes the outline of the rough blank and then the outline of the finished part in terms of elements. Each element — well defined longitudinal segments of the work piece, such as a cylinder, taper, circular arc, or thread — is described by numerical data specifying its dimensions and surface qualities. Elements may be external or internal. Fillets, chamfers, round corners and undercuts take standardized proportions, and their dimensions are included in the parametric string of the host surface element. Threads and grooves of all sorts are treated as superimposed elements. Any other part profiles not included in the above categories may be machined by inserting routines written in a language which resembles APT but is NOT a subset of APT. It has its own rules of syntax and grammar.

The General Electric Company feels that the GETURN system is not comparable to other systems using criteria established with these other systems in mind. Accordingly, there is no response from General Electric in Tables 6 — 9 (Geometric Capabilities) and in Table 16A — J (Test Patterns).

GETURN asks the programmer to describe the part set-up and shape. It then refers to four built-in files to find the user's preferred technological data — a tool file, a methods file, a materials file, and a machine file, GETURN then analyzes the operational requirements of the part, selects a sequence of operations, selects the best available tool, and determines the feeds, speeds, and depth of cut. A generalized postprocessor, GENCO uses data from the machine file to do the postprocessing. The user is responsible for creating and up-dating these various files.

The geometry of a part is described in terms which are not compatible with the conventional definitions of points, lines, circles, etc., although the basic elements of GETURN (cylinder, taper, and circular arcs) are defined by dimensions related to the part axes. As a consequence, the matrices of geometrical definitions (Tables 6, 7, 8, and 9) are not pertinent. This fact is noted in the matrices.

The machining routines are completely under the control of the user, and are so made as to reflect his particular production methods and preferences. The entries in Tables 10 — 15 reflect this adaptability.

Comments on the MARK III Network

The user communicates with the host computer by any one of a variety of terminals varying in speed from 6 characters per second (Telex) to 480 characters per second. Access to the host computer is available by dialing a local telephone number, day or night, and for as long as the connection is needed.

The Foreground Network is an interactive service suitable for entering the program, correcting errors found by the diagnostics, and debugging. Processing may be done at one of three different priorities — immediately, deferred (start within 3 hours), or overnight (start within 24 hours). The lower the priority, the lower the cost. In addition to the Foreground Network, a Background Network is also available for extended computer activities.

Output may be received on teleprinters, CRTs, plotters, and tape punches. A wide variety of commercially available devices is acceptable.

Program Manual Evaluation

The GETURN manual is an adequate document. It covers part programming, and the use of the four files. It has instructions for creating the files and the postprocessor, and for the implementation of a system installation. It contains over 225 pages, typeset and printed on both sides of good grade 8½ x 11 inch paper, perfect bound with a paper cover. It has many clear illustrations and sample part programs. There is no index, but the extensive Table of Contents is an adequate guide for finding information.

Prices Quoted

MARK III charges are composed of several factors. Because of the variety of usage modes, access channels, and transmission speeds, only a portion of the total rate schedules can be shown here. "Metro Access" means access via a special telephone line in one of ten or more metropolitan areas. "National Access" means access via any other route. Of the several options/plans for each combination, only one is quoted here.

1. One time charge for initiation of service		$100.00
2. Monthly charge minimum approximately		105.00
3. Foreground Network Service		
	National	**Metro**
50-300 Baud Service		
Terminal connect time	$ 8.50/hr	$ 6.00/hr
I/O characters	.30/M	.24/M
1200 Baud Service		
Terminal connect time	35.00/hr	20.00/hr
I/O characters	.17/M	.10/M
In data Entry Mode		
50-300 Baud Service		
Terminal connect time	1.50/hr	.75/hr
I/O characters	.10/M	.05/M
1200 Baud Service		
Terminal connect time	N/A	2.00/hr
I/O characters	N/A	.10/M
Continuous access line	$300.00/mo	
I/O characters, prime time	.20/M	
I/O characters, non-prime time	.05/M	

	National	Metro
Computer resource units (CRU)	.12 each	
Program storage units	.80 each	
Data storage units	.22 each	
Priorities for Independent Runs		
Immediate	Standard CRU x 1.0	
Deferred	Standard CRU x .6	
Overnight	Standard CRU x .4	

E. Ingersoll Lathe Language

The Ingersoll Milling Machine Company of Rockford, Illinois developed for their own use an NC lathe programming language. It is available through the IBM Corp. as an "Installed User Program." It operates on an IBM 370/148 and 145, which it shares with other technical and accounting users. Access is gained through a number of IBM keyboard/CRT terminals scattered throughout the NC programming areas.

Ingersoll also has an NC drilling and boring language and a three or five axis milling language. They are all closely related and share many software features. They also work in conjunction with a CADAM system used for part design work and for some NC control work. We will concentrate on the lathe language as an independent capability.

General Description

The Ingersoll language is based on APT, but considerable departure from APT has been made when necessary to achieve turned-part programming efficiency. The programmer's manuscript is keyboarded on one of the terminals, and transmitted to the computer for batch processing at either rush, normal, or overnight priorities. Errors when found are noted by the computer, and when the programmer calls for the results, the computer displays them on the CRT so that the programmer may edit or correct them. When all errors have been eliminated, the computer is again requested to process the program to the CL file level. The CL tape file is APT-compatible, suitable for individual machine tool postprocessors.

The Ingersoll language was specifically designed to include those part programming features — and only those features — necessary to handle Ingersoll's product designs. They do not, for example, handle tabulated or mathematical curves. The language was also designed to reflect the manufacturing technology practiced by Ingersoll's plant. It therefore incorporates in its tool control and postprocessor sections many automatic decisions normally requiring programmer initiative and action. Furthermore, because the lathes and tooling to be used are known in detail, the system can make extensive checks for collision of the tool in use, the turret and other tools, the work piece, chuck and jaws, and any other known obstructions.

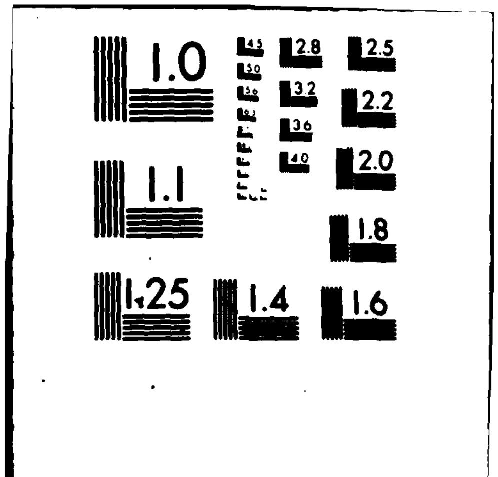
As a result, tapes may be run directly in production without getting tool path plots or making dry runs at the NC machine, an important time saving when production lots sizes average between one and three parts.

Comments on the Processor

The programmer describes the tool layout, the rough stock, the finished part, and the areas of material to be removed. The computer generates the tool motions necessary. It may make several tapes from one program if, for example, the part is to be turned on one end, removed and rechucked, and then turned on the other end. Different lathes may even be involved. Collision checks are made for each set up.

Speeds and feeds are selected from a table for the specific metal to be cut, the surface finish desired, and the tolerance. This is done automatically by the processor.

The language has a statement known as "define area" — DAREA n. This permits the programmer — in effect — to draw a boundary around some area on the two-dimensional part



MICROCOPY RESOLUTION TEST CHART

drawing. He may then command "rough turn area n," and the processor will generate a sequence of cuts to rough turn that stock from the part. This resembles the conventional automatic roughing procedures of other languages. The same macro used for DAREA is also used for describing areas off limits to the tools or turrets.

The Ingersoll lathe input statements are written in free format. Fortran math or logic may be inserted in a program. Part geometry is defined in the fourth quadrant.

Undercuts or recesses are termed "grooves," as well as the more conventional formed tool plunge cuts. The sides may be either straight or angled. The two sides of a groove may be at equal or unequal angles, either more or less or exactly 90°. In a similar manner, flanges may be defined; they are raised above a host surface, rather than depressed below it.

Family of parts programming may be done using variables for dimensions on the prototype, and then defining the values for a specific part. Alternatively, with this type of batch processor, programs which have been stored may be recalled at any time. The programmer keys in the request and the manuscript of a similar part previously programmed may be displayed on the CRT. He then edits in the new part's dimensional data, leaving the rest of the manuscript intact; gives the resulting program a new part number and name, and processes it in the usual manner.

Comments on the Equipment

The Ingersoll processor runs on an IBM/370-148, which manages the queueing and priorities as well as some of the processing, and on an IBM/370-145 which does processing. One IBM 3330 disk spindle is required to hold the programs and data, in addition to the disk requirements for the operating system.

A Tektronix 4631 plotter may be used, or a plotting machine such as a CalComp.

Programming Manual Evaluation

The Ingersoll lathe programmer's manual is a well written document, designed to lead the programmer, whether novice or expert, to produce good programs. It contains 206 pages of 8½" x 11" good grade white paper, bound in a loose leaf form. It is clearly printed and easily readable. Liberal use has been made of diagrams and sample programs. It is well indexed. The manual includes the part definition procedures and the manufacturing technology features.

Availability and Prices Quoted

The language and processor are available through the IBM corp. Monthly charge . . . . $655.00. The monthly charge is waived after the first 12 consecutive monthly payments.

The computer and its peripherals are available from IBM corp.; for terms apply to IBM.

F. COMPACT II

Manufacturing Data Systems, Inc., of Ann Arbor, Michigan, offers COMPACT II, either as remote or local processing systems. It runs either interactively or in batch mode. In interactive mode, on-line error correction is provided.

General Description

COMPACT II is a general purpose NC programming language and processor. In addition to turning, it is also in widespread use for milling, drilling, boring and machining center applications, as well as for punching, flame cutting and EDM. An optional graphics capability is available. A sub-system called FasTurn is of interest in rough stock removal in lathe work. We discuss particularly those portions of COMPACT II applicable to lathe work.

In lathe work, COMPACT II will handle single turret machines, single-slide double turret machines, and double-slide double turret machines. It will also support combination machines which function both as lathes, with the spindle rotating, and as machining centers with the spindle chuck becoming a controlled C axis. Double turret lathes may be "2 + 2" axis machines, or both turrets may engage the work simultaneously, but in independent cuts. In this case, two separately written programs are merged in properly timed relationship on a single control tape.

Comments on the Processor

Unlike processors for the APT-like languages, which have a main processor outputting a CL data file, and a subsequent postprocessor to convert the CL data into machine control tape, the COMPACT II processor outputs machine tool control data directly. It is therefore necessary to inform the system of the machine tool and controller to be used, in the initialization statements. In operation, each statement in a program is processed, and then immediately passed through a software "link" which is equivalent to a postprocessor. The resultant machine tool control data is placed in the output file before the next source statement is read into the processor. Therefore, the CL file and separate postprocessor programs, conventional in APT-type languages, do not exist in the COMPACT II system.

Links are written by MDSI for each machine tool and control combination, and updated whenever the machine tool or control is modified. When a family of machine tools exist which differ only in parametric detail, in some instances, a "family link" may be written, and the programmer specifies the specific machine to be used by inserting appropriate parameters in the set up statement.

Like APT statements, COMPACT II statements have a major word, and may also have a group of minor words. Unlike APT, the minor words need not be in a fixed sequence.

COMPACT II Lathe Routines

There is a special metal removal routine in COMPACT II called FasTurn, which simplifies the writing of statements for the roughing removal of material. FasTurn requires the definition of the raw material boundary and the finished part boundary, and will automatically generate all commands to remove the material between the two. The programmer has extensive latitude in selecting the methodology of the cutting sequence. FasTurn includes the finishing cut and undercuts, but not threading and grooving which are separate routines. Boundaries for roughing must be composed of straight lines and circular elements, but for the finishing cut, any boundary may be used, even including tabulated curves and mathematically defined curves.

Programming Manual Evaluation

The COMPACT II Programming Manual is an outstanding document. It contains over 500 8½ x 11 inch pages, perfect bound. It is printed on good quality paper in easily readable type with black, gray, red and pink letters, each color indicating a different type of data required in the mandatory and optional components of the programming statements. There are many clear diagrams. There is a large index and marginal tab marks on the pages to help in locating each chapter.

The volume covers the programming of the complete COMPACT II *system capabilities*, which includes lathe work, mill work, punching and plotting. There are special chapters on FasTurn, threading and drilling on lathes. There is an extensive dictionary.

COMPACT II has been in use for ten years, and the manual in its present form has been revised and updated in each of the last five years, showing evidence of growth of the system. With the extension of the capabilities to cover every programming problem encountered in ten years, the variety of programming statements has multiplied. The text summarizes these multiple options in generalized statements.

Prices Quoted

Remote Processing Equipment	Rent/Lease	Buy
ASR-33 Terminal (Basic)	$ 62/Mo.	$ 2,000
Bell 113A Data Set (Basic)	20/Mo.	n.a.
MDSI ST1 (Basic or optional)	151/Mo. (3 yr)	8,800
Hewlett-Packard 7221A Plotter (Opt.)	289/Mo.	4,700
Remote Processing Rate Structure		
Processing costs, computer core unit		$ 0.45/CCU
Connect line time		12.00/hour
Storage costs, mass storage unit = 1024 characters		0.05/day/MSU
Link storage and maintenance		12.00/mo./link

Local Processing Equipment and Approximate Costs:

Basis System

Computer makes and models	Lease	Buy
TI980B, 32K words	$558/Mo.(5 yr)	$25,400
Data General/230, 48K words	947/Mo.(5 yr)	43,070
Data Gen. Nova 3/12, 32K words	284/Mo.(5 yr)	12,800
Digital Equip. PDP11/34, 32K words	625/Mo.(5 yr)	28,420
Peripheral Hardware		
ASR-33 Terminal		2,000
Optional Hardware		
HP-7203A-114 Plotter	138/Mo.(3 yr)	4,200
HP-7221A-114 Plotter	155/Mo.(3 yr)	4,700
Tektronix 4014-1 CRT		11,600
Digital Equip. VT52 CRT		2,000
MDSI ST1 Intelligent Terminal		8,800

	Lease	Buy
Basic and Optional Hardware		
Line Printer	178/Mo. (5 yr)	8,125
Reader/Punch	184/Mo.(5 yr)	8,375
Multi-user (sale price hardware dependent)	220/Mo.(5 yr)	800-10,000
Card Reader	110/Mo.(5 yr)	5,000
Multi-plexer	50/Mo.(5 yr)	2,300
Local Processing Software		
Operating Systems		
Type	**Lease**	**Buy**
MDSI COMPACT II Series I	$111/Mo.(5 yr)	$ 5,000
MDSI COMPACT II Series II (Interactive with editor)	$506/Mo.(5 yr)	23,000
MDSI COMPACT II Series II (Batch processing)	310/Mo.(5 yr)	15,500
Editor	50/Mo.(5 yr)	2,500
Postprocessors (Links)		
2 Axis Point-to-point		$2500-3000
2 Axis Contouring		2500-3500
3 and 4 Axis Lathes		5000-6500

G. TOOLPATH

Manufacturing Software and Services, a Division of LeBlond, Inc., offers TOOLPATH, either as a remote or a local processing system; in either case, TOOLPATH is usable in batch or interactive modes.

General Description

The TOOLPATH language and processor is a generalized numerical control programming system capable of producing control tapes for many classifications of NC machine tools. Two and three axis machines with two axis continuous path contouring may be programmed. Programs for turning, milling, drilling, punching, flame cutting, etc., may be produced. We have examined those portions of TOOLPATH particularly applicable to lathe work.

TOOLPATH is an ADAPT-like language. It can handle three axis NC programming in an APT-like manner. While many of the capabilities of TOOLPATH are not applicable to lathe work, it should be noted that the users of TOOLPATH can apply the same processors and procedures used in lathe work to parts other than lathe work. The full TOOLPATH language and the sub-sets for lathe work described below may be thought of as dialects of one language, each adapted to a special class of programming. The programmer selects the dialect most appropriate to what he is doing at the moment. The dialects may be intermixed in any one program, as the processor hears all of them as part of a single language.

Comments on the TOOLPATH Processor

The timesharing processor is designed to offer NC programming on time sharing networks, with high accessibility and fast turnaround, conversational input and output, editing at source language and tape data levels, graphic tool path plotting, and diagnostics. It is applicable to all NC programming tasks.

Two types of in-house, or local, processors are offered: One runs on IBM 360 or 370 systems, which will operate in either the batch or the interactive mode. The other is called MICROAPT, and runs in a mini-computer of Data General, the Eclipse S/130. Both systems use APT or ADAPT postprocessors, and provide diagnostic statements when an error is detected.

Lathe Sub-sets

There is a sub-set of TOOLPATH for simple lathe work programming which uses only point definitions and locations. The tool is moved *to* a point, creating linear motions; or *around* a point, creating circular motions. No lines or circles need be defined. It is called Two-Axis (Lathe) Positioning Motions.

There is another and larger sub-set of TOOLPATH for lathe part programming which requires contouring — i.e., producing a number of discrete motions along a connected sequence of surfaces made up of line and arc segments in the profile which defines the path of the tool. It is called Two-Axis (Lathe) Contour Motions.

In the simpler (Two-Axis Positioning) sub-set, axes may be rotated individually and independently, in order to obtain a slanted reference axis. The tool may then be driven parallel to this new axis orientation to produce tapers and chamfers.

Programming Manual Evaluation

The TOOLPATH Programming Manual is a good document. It contains 268 8½" x 11" pages in loose leaf form. It is offset from typewritten manuscript on good quality matte paper with black ink. There is an initial summary section containing a vocabulary/index, synonyms, and all major definitions. There is also a Post Processor Part Programming Dictionary, which defines and describes a number of routines, and gives the error list used in diagnostics.

The volume covers the programming of the complete TOOLPATH system, including the special sub-set languages and conventions for lathe programming. Diagrams are inserted at points as needed to explain the text. The volume is written in a carefully worded manner, much as one would expect to find in a text book on English grammar, which it in many ways resembles. Sample programs are given. Some of the reproductions of computer printouts are difficult to read.

The manual is written as a reference work rather than as an instruction book. A person familiar with NC programming, particularly with APT language programming, would have no trouble in using the manual. A novice should use one of the training course booklets which are available.

Prices Quoted

In-house system using IBM computer:	IBM 360/370, DOS/OS/VS	
Hardware: See IBM for computer		
Plus, for DOS: 128 k core memory (min.), disk drive		
for OS: 200 k core memory (min.), disk drive		
Software: TOOLPATH processor	for DOS	$6,000.00
	for OS	$9,500.00
Postprocessors for 2-axis lathe:		
Generalized postprocessor in object code with 1 machine data sub-routine		$3,500.00
Machine data sub-routine additions to basic generalized postprocessor, per lathe		$1,500.00 to $2,000.00
In-house system using Data General mini-computer S/130 plus 48 k word memory		
Hardware: See Data General for computer		
Plus: 10 megabyte mass storage disk		
Plus: LeBlond DPS 4000 microprocessor I/O unit		
Basic MICROAPT hardware and software system		$60,000.00
Software: Included in the base price		
Remote Time Sharing Service "LTTP"		
Hardware: For time sharing, LeBlond DPS-4000 basic		$ 7,650.00
For RJE operation, LeBlond DPS-4000 basic with floppy disk, 2780 bisynch option		$11,050.00

Rate Structure:

Processing turnaround time and rate per CPU:

Immediate execution	$0.11	4 hour execution	$0.065
Class R	0.11	8 hour execution	0.055
1 hour execution	0.085	24 hour execution	0.035
2 hour execution	0.075	Weekend execution	0.03

Charges for connect time:

Terminal Speed	Mon.-Fri. 8:00 am to 6:00 pm	Non-prime time
10 cps (110 baud)	$14.00/hour	$6.50/hour
30 cps (300 baud)	16.00/hour	8.40/hour

Storage charges:

Disk track storage (156 lines per track)	$ 0.09/track/day
Postprocessor storage surcharge for access to existing postprocessor	$20.00/month

Minimum usage charge:	$50.00/month

H. APT for Lathes

McDonnell Douglas Automation Co. of St. Louis, Missouri, offers a lathe programming language "APT for Lathes." It is a remote processing system running either in batch or time-sharing mode. The language is based on the APT language, but uses statements which are abbreviated by the use of synonyms.

General Description

APT for Lathes is an APT based system, and any of the APT vocabulary words or statements may be intermixed with the system's abbreviated format. An extensive table of synonyms and abbreviations for geometric features has shortened the length of normal APT geometry statements. A string feature, not yet released, is expected to shorten the conventional APT motion statements.

Regarding the abbreviation of geometry statements: there are three categories — APT, Mini APT, and Micro APT. For example, an APT definition of a point, such as PTA = POINT/CENTER C21, would be reduced in Mini APT to PTA = C21, and still further reduced in Micro APT to A = (C21). Lines perpendicular to the centerline may be described as FA,10 for example if the Z coordinate was 10 inches from the origin; or the description might be TN,1 if the line were parallel to and 1 inch from the centerline.

Comments on the Processor

The processor handles three basic machining operations: roughing, finishing, and threading. The roughing statement is

ROP/F_p,F_c,F_r,clearance,PASSES, n,d,n,d,.....

in which F_p is the positioning feed rate, F_c is the cutting feed rate, and F_r is the retract feed rate. Clearance denotes the distance the cutter moves back from the workpiece when retracting. The letter n indicates the number of passes at depth d. The roughing statement would have to be followed by normal APT motion statements that are ended by a TOP (terminate operation) statement.

Finishing statements follow the standard APT format. There is a threading statement which includes the normal variables, such as pitch, positioning and retract feed rates, number and depth of passes, and the major diameter. There are no special provisions for grooving, nor are there any special tool orientation statements.

Prices Quoted

Cyber service time-sharing:

$ 0.18/MRU
10.00/hour connect time
0.04/indirect storage unit/week
3.00/direct storage unit/week

IBM service, batch:

$ 6.75/MRU
1.00/SRU

IBM service, time-sharing:

$ 7.50/MRU-TSO
10.00/hour connect time
0.04/day/track for storage

I. GTL/T

Datamat Programming Systems, Inc., with offices in New York City offers the Olivetti Corp. *of America's GTL/T system as a local processor on a stand-alone, dedicated desk top mini-*computer for programming NC lathe work.

General Description

The GTL/T system is specifically designed for lathe work. Olivetti also offers a similar system, GTL/3, specifically designed for machining center work. While the two systems use essentially the same geometric statements, they have unique "technological statements" which define the motion of the tool and activation of specialized sub-routines, and control the machine tool's functions. Neither of these two systems can be considered to be a sub-set of the other.

GTL/T is similar to ADAPT in its language construction, but it is unusual in that it uses "oriented geometry," in which lines and circles have a sense of direction, as do angles. All geometry is defined in terms of points, straight lines and circles.

The sense of direction of a line is positive in the direction from a first point to a second point, or in the sense of direction of a parallel line or axis from which the line in question is located. The sense of direction of a circle is positive in the counterclockwise direction. The opposite sense of either may be indicated by prefixing a " −" before the element's symbol or the circle's radius. Angles are positive when they measure the counterclockwise rotation of the +Z axis to bring it parallel to, and in the same direction as, the referenced element.

The use of oriented geometry resolves many ambiguities which would otherwise call for the use of a selector in the definition statement. For example, in ADAPT language, the definition of a line as passing through a point and tangent to a circle offers two possibilities, and a selector has to be added to tell which one was meant. In GTL/T it is assumed that the elements join one another with the same sense of direction at the point of junction, so that a bug crawling along the profile would find each successive line segment to have the same direction in which he was progressing. If necessary, a negative sign may be affixed to the element symbol to create this continuity. The use of oriented geometry also simplifies the writing of tool control statements. It is also a convention that the sequence of segments follows along the minor arc of any circle encountered.

Comments on the Equipment

GTL/T runs on a mini-computer system known as Olivetti P6060 Personal Mini-Computer. It is a central processing unit, full alphanumeric keyboard, a printer/plotter, line display, dual floppy disk units, and a tape punch; and optionally, a printer, 10 megabyte memory disk, and interfaces. Plotters and tape readers may be attached. It will work as a time-sharing terminal. The CL file image may be plotted on the printer. The system will issue diagnostics to the user if it detects an error in input, syntax, or wrong or missing statements.

Comments on the Processor

GTL/T has macros for roughing, finishing, threading, grooving, drilling and tapping.

Segments of previously defined elements may be strung together, at points of intersection or tangency, to form a continuous profile of a part. Profiles must begin at a point and end at a point, and are called out in a counterclockwise direction. When necessary because of the complexity of a part, it may be programmed as two or more separate programs through the CL file stage; they may then be strung together and postprocessed in a single operation to the machine control tape output stage.

The definitions of profiles are separated from the control of a tool along the profile.

A revision to the processor due in January 1980, will enhance the system and decrease processing time by 70%.

The floppy disks which contain the postprocessors also contain a statement of the rules under which each was written. It is available to the user on the printer on call, in case of trouble.

Programming Manual Evaluation

The GTL/T programming manual is a good document. It contains 250 pages, 8½" x 11" loose leaf bound. It is offset from typewritten manuscript, using several type fonts, on good quality matte paper with black ink. There are about 150 diagrams in the text, illustrating each point. There is an appendix on installation; on program error messages; and on advisory or information messages issued by the system during its interactive operation.

The manual is written in a clear, tutorial manner which is easy to follow. Manual revisions are issued as the processor is revised and enhanced. The current volume shows careful editing for readability and easy application. Sections of sample programs are shown to illustrate each element of the programming system as it is explained.

Prices Quoted

Equipment and Approximate Costs: Basic System:

Olivetti P6060 and two floppy disk drives	$ 6,600.00
32 K byte additional user memory	3,600.00
80 character per second printer/plotter	1,850.00
Olivetti paper tape punch	1,595.00
Alphanumeric display	no charge
Optional Extra Equipment:	
Paper tape reader	1,295.00
Asynchronous line control for time sharing and connection to compatible EIA RS232C peripheral units	500.00
Central memory extensions, 8K	900.00
Moving head disk unit	11,000.00
High speed impact printers	3.800.00+
Software:	
Processor	3,000.00
Postprocessors, each	1,700.00

J. APTURN

The APTURN processor is the product of Structural Dynamics Research Corporation (SDRC), Cincinnati, Ohio. APTURN is a generalized lathe processor designed to prepare tapes for numerically controlled lathes. It is segmented into modules and is available to the customer on a time-share basis. The APTURN system was offered through the United States Steel Corporation time-shared network; however, the system has been recently converted and is offered through the General Electric Mark III computer and communications network.

General Description

APTURN is designed to be used as a time-shared system. The statements used are shop-oriented and easily recognizable by machine tool part programmers. APTURN is written in Fortran and is extremely modular for ease of support and modification by the SDRC systems support personnel. As the name implies, the APTURN system has been specifically designed for lathes and therefore provides a variety of capabilities involved in lathe operations (i.e., facing, turning, drilling, threading, etc.). The system remotely resembles the APT System in that a similar vocabulary structure exists and the same general format is used. The APTURN System is modular, uses English-like descriptors, provides excellent diagnostics, and contains a wide range of features.

Comments on the Processor

The APTURN processor is segmented into two major classifications and within each, the part programmer has the option of writing statements in a variety of formats. The two major classifications are operators and sub-operators. Typical operator statements are roughing, finishing, facing, drilling, etc., and typical sub-operator statements are part boundary description, forging, facing point, plunging point, etc. The system provides the part programmer with the ability to perform arithmetic calculations within the statements and thereby reduce programming time. An additional feature of the System is its capability to link-edit to a calculation subroutine which is used as an on-line calculator to solve trigonometric and mathematical problems. The solutions are then used within the main-line part program.

Programming Manual Evaluation

The APTURN part programming manual provided by SDRC is excellent. It provides a system overview and a complete detailed review of each and every capability offered by APTURN. Diagrams and examples are used extensively throughout. Error diagnostics are complete and offer a complete explanation.

Additional instruction is provided by SDRC regarding the use of the APTURN System and use of the remote terminals.

Prices Quoted

Prices when using the General Electric Mark III network:

Rates per CRC	$ 0.13
plus charge for characters transmitted	0.12/M
Connect time, input	1.50/hr
Connect time, output	12.00/hr
plus charge for characters returned	0.10/M
Storage	80.00/unit
Monthly invoice, minimum	20.00
Variation for deferred response	
Immediate	100% of above
Within 3 hours	60% of above
Within 8 hours	40% of above

NOTE: The test parts of Chapter X were run on the U.S. Steel network.

Prices when using the U.S. Steel network:

Rates per ARU	$ 0.60
Connect time	14.00/hr
Storage per unit of 1920 characters	0.80/unit

K. VNC-200

VNC-200 is a general purpose numerical control system developed and offered by Threshold Technology, Incorporated of Delran, New Jersey. The system consists of a processor, the VNC-200 (Voice Numerical Control-200) and a variety of postprocessors for point-to-point and continuous path operations. The software resides on a Data General Nova 3/12 mini-computer. The basic system consists of a 32K memory, 10 megabyte disk, CRT, paper tape reader/punch, and a variety of support hardware. The system is offered as a voice data entry system or keyboard data entry system, at the option of the buyer. Additional peripherals are offered as systems options including plotters, high-speed printers, high-speed tape read/punch, etc.

General Description

The VNC-200 System is unique in that the developers, Threshold Technology, Inc., have designed a system which offers the user two distinct modes of data entry. Aside from the more industry-acceptable keyboard entry method, the VNC-200 System responds to voice data commands. The part programmer speaks into a microphone which is connected to the Data General mini-computer by way of voice preprocessor and associated electronic units. The voice command is "decoded" by the System and translated to numerical control vocabulary statements used by the appropriate postprocessors. Threshold Technology offers a variety of postprocessors each tailored for a specific machine tool/controller combination and each residing on the system disk. The VNC-200 software consists of complete source code or tape editing capabilities as well as tooling and mathematical libraries, family-of-parts storage, system macro functions, and plot routines.

The VNC-200 System is a prompt-type interactive system which queries the part programmer, sequences him through the logical steps of data preparation, and advises him of errors on a real-time basis.

The programmers voice is introduced to the system through a "training" technique consisting of repeating the system vocabulary and storing the "voice pattern." There is no restriction as to the number of part programmers who are able to use the same system since the system is able to distinguish between each programmer and recall the appropriate information.

Comments on the Processor

The VNC-200 processor contains a vocabulary of eighty-nine (89) unique words. They include the digits from zero (0) through nine (9) and tool and motion statements such as TURN, THREAD, SLOPE, TERMINATE, LINE THROUGH CIRCLE, etc.

Included in the supervisor commands is the command for the mathematical library which provides the math algorithms. They include thirteen (13) distinct algorithms and their variations (i.e., calculation of X, Y coordinates based upon two intersecting lines when the data given is four points, two points, and a point and angle given, two points and two angles given, etc.).

Programming Manual Evaluation

Threshold Technology, Incorporated has developed and offers its customers an excellent operations manual. It is concise and complete. It provides a brief overview of the VNC-200 philosophy and a review of the voice and keyboard vocabulary as well as the supervisory statements. The majority of the manual reviews each and every statement, provides both input command format and visual display read-out, and provides the reader with an in-depth explanation of how to use the VNC-200 System.

Threshold Technology, Inc., provides customer instruction at its own facility and at the customer site on the use of the VNC-200 System.

Prices Quoted

Voice mode entry (the system used for test parts):	
VNC-200 HCP22 Includes CRT Console, High Speed Reader/Punch, and High Speed Line Printer	$67,570.00
Basic voice system:	
VNC-200 Includes ASR 33 Teletypewriter	48,910.00
Keyboard entry system:	
VNC-200K Includes ASR 33 Teletypewriter	38,660.00
Upgrade of VNC-200K to Voice	11,500.00

L. UNIAPT

United Computing Corporation of Carson, California, now a part of the McDonnell Automation Company, offers UNIAPT, an NC programming system which has all the capabilities of the APT language, but operates on a mini-computer. A lathe module "ULM" extends the basic APT language to facilitate the definition of turned parts geometry and the metal removing objectives. It is fully compatible with the main UNIAPT language, and they may be intermixed. We have directed our evaluation principally to the lathe module and the parent system.

UNIAPT is a local processing, batch mode system.

General Description

The UNIAPT lathe module ULM is an adjunct to the basic APT vocabulary, and not a set of macros or preprocessors designed simply to ease the burden of lathe programming. The several lathe module statements have all the powers and advantages of the APT syntax structure. ULM statements available are: SHAPE, for defining the blank and finished shapes of the part; LATHE/ROUGH, for generating the multiple cut statements necessary to remove excess stock; LATHE/FINISH, for generating the finish cuts after roughing; LATHE/GROOVE, for cutting grooves; LATHE/DRILL, for center line drilling; LATHE/THREAD, for designating a complete threading sequence with one command; and LATHE/DRAFT, to cause a plot of the blank and finished part to be made. Each statement includes as many parameters as necessary to define the task. Any part design features beyond the powers of these lathe module statements may be programmed using the standard UNIAPT language.

UNIAPT has been in field service since 1969.

Comments on the Processor

The UNIAPT processor gains its capability for handling the full APT structure in a mini-computer by virtue of the internal design of the processor. It operates in two passes instead of the usual four pass procedure for APT in large computers. The first pass consists of a single program. The second pass is comprised of a number of programs, which are called from disk as they are needed. Pass two programs overlay one another. Both the processors and the postprocessors are written in a proprietary machine-independent language UPL, so that it is relatively easy to implement the system on any new computer.

In addition to the full APT capability for 2 to 5 axis machines, there are the following extension modules: Lathe; five axis continuous path; ruled surface definition; sculptured surface definition; CL file generator; and mill. UNIAPT may be extended into an interactive graphic system UNIGRAPHICS to do CAD and CAM work.

Comments on the Equipment

UNIAPT is sold as a turnkey system on the Data General Corp. Nova 3/12 and the Eclipse S/230 computers. It is also sold for use on a user-supplied computer, and is currently implemented on over 20 other mini-computer systems. A UNIAPT system is intended for the sole use of the NC Programming Department, but the computer need not be dedicated to one programmer; several programmers can work concurrently on the system.

Programming Manual Evaluation

The UNIAPT programmer's manual is a good document. It consists of 226 pages that are described under a detailed table of contents and, what is most significant, there is an alphabetical index and glossary. The alphabetical index distinguishes it from many other manuals that do not have them. The manual is in loose leaf form to allow for update. It is clearly written and well referenced, and consistent in quality with United's well designed UNIAPT manual.

Prices Quoted

"Turnkey" installations

Nova 3/12	$50,000- 70,000
Eclipse S/230	82,000-120,000

The precise cost is a function of how many programmers the system is prepared to support. The cost includes the computer, input/output devices, software, installation, operator training, programmer training, and one year of software maintenance.

Peripherals: card readers and line printers	$ 6,000- 18,000
Optional Hardware: processing and editing stations	4,000- 25,000
plotters	6,000- 33,000

Customer-supplied computer installations

UNIAPT basic software	$19,000- 30,000
UNIAPT postprocessors	3,500- 8,500
Extension modules	1,200- 6,000
Plotters and interactive CRT postprocessors, approximately	4,000

Hardware and software may be leased.

M. UCC-APT

University Computing Company of Dallas, Texas, offers UCC-APT for NC programming, with enhancements for the NC lathe programming routines of automatic roughing, finishing, and threading. UCC-APT is a classical APT language and processor; their version has been under development since 1966, and embodies many proprietary enhancements and improvements. It operates as a remote or local processor, running in batch or time-sharing mode. Our tests and evaluations are based insofar as possible on the special lathe routines, supplemented as necessary by the standard APT statements.

General Description

UCC-APT is a complete NC programming system, capable of handling point-to-point and continuous path programming for machining centers, mills, drills, punches and lathes. It can describe not only the usual geometric elements, but also cylinders, cones, spheres, tabulated surfaces and ruled surfaces. It will control four and five axis machines.

The supersets of APT used for NC lathe programming use only portions of this total capability, but obey the same rules of syntax and grammar. Hence they may be intermixed with the APT statements to describe any lathe part design. The three supersets simplify the preparation of programs from information to be found on the part drawing, by eliminating the necessity of writing geometric definitions and motion instructions.

The LATHSQ (Lathe Sequence, pronounced lathe-seek) statement can be used to program a sequence of lathe moves, in which a cutting tool nose is moved along a boundary of a part. This boundary may be used to produce a cutting operation, or to define one side of an area to be removed by an automatic roughing routine. The THRDSQ (pronounced thread-seek) statement can be used to program an entire threading operation by means of a single statement. The UTURN statement can be used with two boundary definitions to remove all the material between the start boundary and the end boundary.

The boundaries defined by LATHSQ may consist of straight line segments and circular arc segments. If any other segments are required, the full UCC-APT language is available to describe their shape and to write the motion statements. Similarly, if grooving, drilling, or tapping are required, they are programmed using the full UCC-APT system.

Comments on the Equipment

The user requires a terminal with keyboard, printer, punch, plotter, etc., with telephonic access to the UCC network for remote processing, or local connection to his own computer for local processing. The remote host computers are Univac 1108s. For local processing, the user selects the same or equivalent computer systems.

Program Manual Evaluation

The UCC-APT programming manual is an adequate document, well written and easy to understand. It covers the entire capability of the system, of which the three lathe supersets are a part. It is phototypeset (interestingly, on UCC's own word processing system, UTYPE), and printed on both sides of 8 ½ x 11 inch white paper. There are 309 pages, loose-leaf bound, of which 62 are devoted to the lathe programming capability. There are many good diagrams and sample programs, an index and a large table of contents.

Prices Quoted

I. UCC 1108 AND ASSOCIATED RESOURCES

A. RATES PER CPU SECOND

Description	Level	Prime Time (8AM - 6PM)	Non-Prime (6 PM - 8 AM)
Super Express Plus	1	$.55	$.46
Super Express	A	.46	.38
Express	D	.40	.33
Standard	F	.34	.28

NOTES:

1. Non-Prime Time Rate also applies to work processed between 8 AM Saturday and 8 AM Monday.
2. An additional charge computed at 10% of applicable CPU Rate is made when the "MXD" option is used.
3. An additional charge computed at 25% of applicable CPU Rate is made when the Large Core option is used.
4. Minimum charge per run is $2.00 when using SUPER EXPRESS PLUS priority.

B. PERIPHERAL RATES (CARDS AND PAPER NOT INCLUDED EXCEPT WHERE NOTED)

Card Reader	$ 3.00	Per 1,000 Cards
Card Punch	5.50	Per 1,000 Cards
Line Printer	.08	Per Page

NOTE: All Peripheral Rates apply only to UCC owned and operated devices.

C. STORAGE RATES

On-Line Mass Storage $.20 Per Block (7,168 36-bit Words) Per Day

On-Line Mass Storage Discount

Monthly Storage Charge	Discount
$ 0 - $ 500	0%
$ 500 - $ 1,000	20% on Amount Over $500
$ 1,000 - $ 2,000	$100 + 30% on Amount Over $1000
$ 2,000 - $ 3,000	$400 + 40% on Amount Over $2000
Over $3,000	$800 + 50% on Amount Over $3000

D. COMMUNICATION PORT CHARGE

Dedicated Port	$300.00	Per Month
Dial-up Port	20.00	Per Connect Hour

E. TAPE MOUNT CHARGE $2.00 Per Tape Mount

II. FASBAC AND ASSOCIATED RESOURCES

A. CONNECT RATES AND COMPUTER RESOURCE UNITS (CRU)

10 - 15 CPS	10.50 Per Hour
30 CPS	13.50 Per Hour
CRU	1.35 Per 1,000

NOTE: The maximum Hourly FASBAC Rate is $27.50 per hour measured monthly.

B. FASBAC WATS SERVICE

$10.00 Per Connect Hour

C. FASNET SERVICE

$5.00 Per Connect Hour

D. ON LINE MASS STORAGE

On Demand	$1.25	Per Page Per Month
Scheduled in Advance	.75	Per Page Per Month

NOTES:

1 One page equals 2048 ASCII Characters.

2 Scheduled storage must be requested in writing prior to the beginning of the calendar month for which storage is required and is scheduled only in 100 page quantities

E. PERIPHERAL RATES (INCLUDES STANDARD RIBBON AND SINGLE PART PAPER)

Upper and Lower Case Line Printer $.10 Per Page

N. PROMPT

Weber NC Systems, Inc., of Milwaukee, Wisconsin, offers PROMPT as a general-purpose local processing NC programming system. It operates on a dedicated stand-alone mini-computer, with plotter and printer attached. PROMPT will handle milling, drilling, flame cutting, and plotting machines as well as lathes. We have examined the lathe programming portion in particular.

General Characteristics

The PROMPT language and its programming system bear no resemblance to the APT or COMPACT families of languages. Its architecture was designed without reliance on any prior language architecture. PROMPT might be characterized as shop oriented.

The system gets its name from the displayed statements and questions by which it leads the programmer step by step through the programming function. Because these questions are stored in and retrieved from the processor, they need not be written in a coded format; they can be displayed in normal English language. In this sense, therefore, PROMPT does not use a "language." The question displayed at any point in time is dependent upon the answers previously given; the menu of questions would resemble a decision tree. Usually a choice of two or more responses is offered, and only a single key stroke is required to make the response. Or, a numerical value may be required such as a speed rate or dimension. Thus lengthy typing is avoided, and syntax errors are not a problem because the system will accept only one of the correct optional answers.

PROMPT uses "oriented geometry" in which lines and circles have a sense of direction, as do angles. All geometry is defined in terms of lines and circles. In point-to-point work, points are located by their coordinates. In contouring, points are conceived as circles of zero radius. Circles are defined by their center point and radius, and lines may be defined as connecting two points. Lines are positive in a direction from the first point to the second point. Circles are positive in the clockwise direction. Angular directions are thought of as a compass course, with 0° at North or +X, and 90° at East or +Z, etc.

Boundaries of a part are described by groups of two, three, four, or five of the line or circle elements which define the boundary. These groups all fall into one of seven "types," four of which are shown in Figure 19. The last element of one archetype becomes the first element of the next. The sequence is repeated until a boundary has been completely described. Thus a variety of alternate formats in which a line or circle may be defined in APT-like languages is not used in PROMPT. As a consequence, the matrix of geometrical definitions given in Chapter V are not pertinent. This fact is so noted in Chapter V and in the matrices.

Coordinates are expressed in terms of absolute diameters and incremental lengths, rather than in X, Z or X, Y.

Comments on the Equipment

PROMPT runs on a Hewlett-Packard mini-computer model 9835A, which has a full alphanumeric keyboard, and a CRT capable of displaying a series of questions or statements. A floppy disc unit HP9885M contains the processor and postprocessor, and a mathematical problem solver called CALPROMPT. It also has room for storage. A small tape cassette, integral with the

PROMPT MASTER COMPOSITES

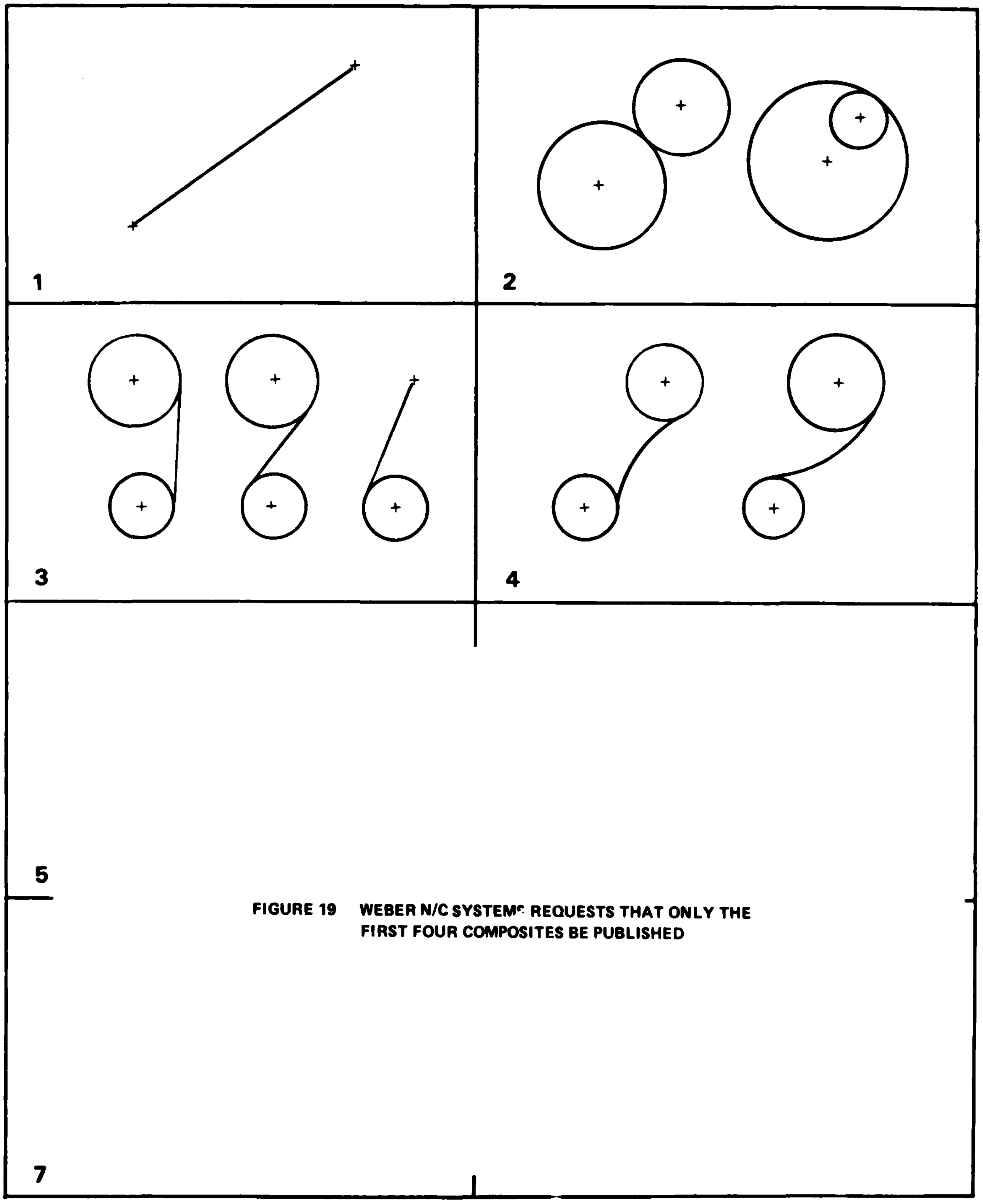


FIGURE 19 WEBER N/C SYSTEMS REQUESTS THAT ONLY THE FIRST FOUR COMPOSITES BE PUBLISHED

mini-computer, stores finished contour definitions, points, and the source and object records. Hewlett-Packard plotter 9872A and line printer 2631A are attached. A tape punch is also attached.

Programs may be written, stored, recalled, edited, and output as hard copy or perforated tape.

Comments on the Processor

PROMPT programs are written directly onto the keyboard without a prior paper manuscript. The sequence of events in the production of a finished control tape is as follows:

— The (lathe) part boundary is described, using a series of boundary types. Any syntax or format errors in this process are rejected by the processor, which will not proceed until they are corrected.

— The processor computes and then plots the finished part profile. Any errors in shape or dimension are perceived and corrected.

— The processor computes and plots the tool tip center path.

— The processor submits a list of questions concerning the first tool to be used. The programmer answers, defining the type of cut and tool he desires, with technology information. When satisfied, the processor computes and plots the tool motion.

— The above is repeated for each tool and cut to be made. Any error in tool motion, such as tool/part interference, is immediately perceived and may be corrected.

— When tool motions are all defined, the processor produces a tape and a hard copy just as fast as the punch or the printer can work. Plots may be made to any scale and offset desired.

As many profiles may be defined as necessary to describe the part. They may be defined in any order and in either direction. PROMPT has macros for rough stock removal, cutting along a profile, threading, grooving and drilling. There is as yet no macro for tapping or die threading, although one will be written if requested by a system user.

Programming Manual Evaluation

In place of the usual manual for programmers, PROMPT has an interactive teaching facility, using the mini-computer as a teacher. The program leads a student step by step, showing first how to work the keyboard, then how to describe a part and call for its production. The cassette may be played over as many times as needed. It is reported to be a quick and effective instruction method.

In addition, PROMPT has a 30 page mimeographed notebook, 8½ x 11, with instructions for programming. It is in process of revision.

Because of the tutorial nature of the system, the programming manuals are characterized as adequate.

Prices Quoted

All hardware is Hewlett-Packard. Each system includes all interface cables.

System I:		$19,095.00
9835A	Computer with 64K bytes memory (includes video terminal)	
9872A	Plotter	
9884	Tape Punch	
System II:		$22,295.00
Consists of everything in System I plus the following:		
2631	Serial Impact Printer	
System III:		$26,045.00
Consists of everything in System II plus the following:		
9885M	Flexible Disk Drive	
Any of the following options may be added at the additional price shown:		
16098A	Printer Stand	$ 275.00
9883A	Tape Reader	4,110.00
Software "PROMPTURN"		
Base price		$ 8,500.00
Price if added to existing PROMPT system		6,000.00
Postprocessors		1,750.00

O. WESTURN

Westinghouse's Industry Systems Division, of Pittsburgh, Pennsylvania, offers WESTURN, the Westinghouse Turning Program developed to provide a shorthand method of programming lathes. Programs are processed in batch mode on the Westinghouse Tele-Computer Network. The Westinghouse general NC programming system uses IBM APT based language. WESTURN is a sub-set, tailored specifically to the programming of lathe parts. We have confined our evaluation to this WESTURN sub-set.

General Description

WESTURN is a "superset of the APT language," developed to provide the briefest possible means of describing a lathe operation. However, it does not prevent the programmer from selecting the sequence of operations and tooling. WESTURN uses a series of preprocessors, one for each of the most commonly used lathe routines. Each preprocessor operates ahead of the processor, generating APT input statements from the input macros. The APT statements then run through a conventional APT processor and postprocessor to generate the machine control tape. The preprocessor has been designed to minimize processing time, and to provide clear diagnostics.

WESTURN accepts geometric input limited to diameters (cylinders), and faces; chamfers, and fillets or round corners of any radius may be interspersed. Diameters may be tapered, threaded, or have recesses, undercuts, necks, and grind reliefs. If other profiles are required by the part design, the programmer may intermix longhand APT statements with the WESTURN macros. Over 50 macros are available. A macro definition follows a set form for each type of macro, and may have a dozen or more parameters to be filled in. That single statement — for example, "CALL/MACRO ....," in a program — may generate many lines of conventional APT input statements. Each macro selects the proper tools, and generates the proper feeds and speeds and depth of cut; the programmer may override these parameters if he wishes. The macros are strung together like building blocks to construct a complete program.

Comments on the Equipment

The equipment is conventional. The user has a keyboard terminal, connected by telephone line to the Westinghouse remote processing system host computer, an IBM 370/165 system. Programs are returned by wire to the user in the form of machine tapes produced locally on a tape punch, and a hard copy produced on a line printer. A plotter may be attached. Program editing is conversational.

Programming Manual Evaluation

The WESTURN manual is an adequate document. It provides a reference to individual macros for producing a part program. When used in conjunction with the APT program manual it provides all the necessary information to produce a part program for processing by the IBM APT system implemented on the Westinghouse Tele-Computer Network.

It is printed by offset on one side of 8½ x 11 inch paper, in a tab-indexed loose-leaf binder. Diagrams are provided for each macro, showing the dimensions and other information covered in the defined parameters. Ten sample part programs are given. There is no adequate general index in this volume.

Prices Quoted

Function		Charges
IBM 370/165 Computer Processing		
Batch and Remote Batch Job Processing		
Submitted via High Speed Terminal [3]		$1300.00 per Computer Resource Unit [1]
Submitted via CMS Terminal [2]		$1560.00 per Computer Resource Unit [1]
Input/Output and Editing via CMS [2]		$ 440.00 per Terminal Service Unit [8]
Data Communications Ports		
High Speed Terminals [3]		
Voice grade shared usage		$14.00 per hour
CMS Connect Time 10 and 30 CPS		
via Westinghouse Network		$15.40 per hour
via User's Direct Dial		$ 4.40 per hour
Input-Output Processing [4]		
Printing	– 1,000 lines	$2.45
Punching	– 1,000 cards	$10.50/1000 cards
Reading	– 1,000 cards	$2.45
Plotting	– 1,000 pen movements	$0.0715
Paper tape punching	– 1,000 characters	$1.375
Online Disk Storage		
CMS Interactive Files		$1.00 *per cylinder day* [9]
Batch and Remote Batch Files		$0.050 per track per day [5]
CMS Output Files		$2.20 per 1000 block days [10]
Offline Disk Storage		
Disk Storage [6]		$30.00 per pack per month
Magnetic Tape Storage		$ 2.00 per tape per month
Shared Disk (Panvalet)		*$ 0.05 per block month* [10]
Operator Actions		
Disk pack mounting		$2.40 per disk pack mounted
Tape mounting		$1.50 per tape mounted
Auxiliary Services		
Card Punching/Verification		$10.00 per hour
Card Collating		$20.00 per hour
Card Sorting		$20.00 per hour
Card Reproducing		$20.00 per hour
Card Interpreting		$20.00 per hour
Mailing/Express Service		$1.50 per package plus postage or transportation
Monthly Usage Discount [7]		
Discountable Volume		**Invoiced Amount**
0 – $2400 00		Full Amount
$2400.00 – $3000 00		$2400.00
Above – $3000.00		80% of Discountable Volume

[1] *The basic unit of measure* is a Computer Resource Unit (CRU), which is calculated according to the following algorithm:

$$CRU = \frac{1}{3600}(CP + 0.005 \times N_T + 0.010 \times N_D)\left(1 + \frac{K}{600}\right)$$

where

CP = Central processor time in seconds as recorded by the IBM operating system.

N_T = Number of tape input and/or output operations. Each execution of the EXCP (Execute Channel Program) instruction is *considered to be an input-output operation*

N_D = Number of non tape input and or output operations. Each execution of the EXCP (Execute Channel Program) instruction is considered to be an input/output operation

K = *Thousands of bytes of magnetic core* memory requested.

The Computer Resource Unit will be subject to a surcharge factor for certain proprietary applications. There will be a minimum charge of 0.002 CRU's per run

[2] For remote terminal operations, the CMS system requires a model 33 ASR or model 35 ASR teletype terminal with dial-up telephone communications or compatible equipment

[3] Requires remote batch terminals such as IBM 2780, IBM 1130, IBM 360, UCC COPE series, UNIVAC 9000 series, Westinghouse 2550. User provides the line

[4] *Charges apply to input-output processing per*formed at the Westinghouse terminal in Pittsburgh, Pennsylvania. Input output processing is not charged for work done on the customer's terminal

[5] One track can contain up to 11760 8 bit characters. There is a minimum charge of 19 tracks whenever batch online disk storage is reserved. There is no minimum for CMS

[6] One disk pack provides approximately 23 million characters of storage. IBM 2314 compatible

[7] Discountable volume includes normal computer resource unit and terminal service unit charges, operator actions, input output processing, online disk storage, mountable storage, auxiliary service. The discount is not applicable to charges made for data communications ports, terminal equipment or man . me

[8] The Terminal Service Unit is calculated according to the following *algorithm*

$$TSU = \frac{1}{3600}(4.4\,CP + 0.018\,N_t)$$

where

CP = Central processor time in seconds as recorded by the IBM operating system

N_t = Number of input and or output operations. Each execution of the EXCP (Execute Channel Program) instruction is considered to be an *input output operation*

There will be a minimum charge of 0.005 TSU *per* CMS session

[9] A cylinder of CMS storage can contain up to 572,070 characters (8 bits)

[10] A block is approximately 5 lines of printed output

CHAPTER VIII

TEST PATTERNS

Before parts were designed for benchmark tests, it was desirable to see how the various systems wrote statements for a wide variety of commonplace lathe operations. A set of patterns was distributed to the system proponents and they responded with a set of statements appropriate to each pattern.

In order to determine and verify specific capabilities of the systems, and to obtain illustrations, proponents were asked to write the statements pertaining to a series of geometric patterns and basic lathe operations. The geometric patterns varied in complexity from straight lines parallel to the X and Z axes, to mathematically defined curves, such as a parabola. The basic lathe operations covered roughing, finishing, threading, grooving, drilling, and tapping.

The patterns were designed to illustrate a broad spectrum of turning requirements. Patterns #1 through #6 describe geometry forms; patterns #7 through #14 cover roughing operations using the geometric forms described in patterns #1 through #6; patterns #15 through #20 cover finishing operations, again using the geometry that was described in the original six patterns; threading is covered in patterns #21 through #27; and grooving, drilling, tapping and tool orientation statements are covered in patterns #28 through #33.

Specific guidelines were set forth for each of the patterns, such as the speeds, feeds, depth of cuts, material, tool nose radius, and the finish required, when such information might be required to write a complete statement.

The patterns have been reduced in size and are shown in Table 16, together with a description of the operations and checkmarks to show the response of the systems to each pattern.

"X" indicates that the proponent responded to the pattern.

"A" indicates that the response used statements that are predominantly APT statements.

"PA" indicates that the response used statements that are partially APT statements.

The latter is significant since six of the fifteen systems in the evaluation are APT based, and because the Army, together with most of the Department of Defense, is APT oriented. The proponents of the APT based systems have prepared special programs — generally either preprocessors or macros, that supplement their APT processors. It is therefore possible to intermix APT statements with those that have been developed especially for lathe part programming. Also, in those instances where part geometries are relatively complex, such as those defined by mathematically defined curves or tabulated curves, APT statements may be used for writing most, or all, of a lathe part program.

In summary, all of the systems can handle simple roughing, finishing and tapping operations with processors, preprocessors, or macros that have been developed especially for lathe operations. In some instances where the geometry is a bit more complex, involving lines that are tangent to, or that intersect circles, APT statements have been resorted to in parts of the program (MS&S, McAuto, and University Computing).

All of the systems are capable of handling simple threading operations. However, two systems could not handle threading on a face (Ingersoll and Weber N/C). Grooving does not present a problem to any of the systems, nor does drilling and tapping. All of the APT based systems (Cincinnati, MS&S, McAuto, United, University, and Westinghouse) are able to handle the more complex mathematically defined and tabulated curves, as are some of the other non-APT based systems (Encode, MDSI, and Threshold). Weber, as noted on page 106, does not describe geometry in a manner compatible with this exercise, and did not respond to the patterns. Threshold did not respond to patterns 21 through 33.

TABLE 16 A
TEST PATTERNS

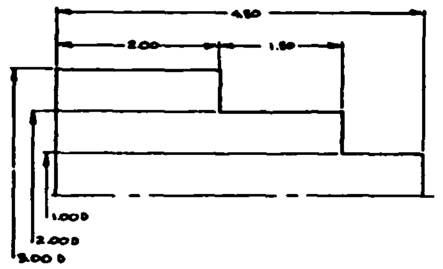
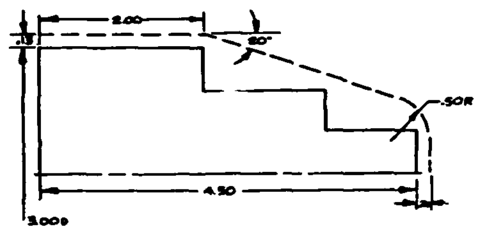
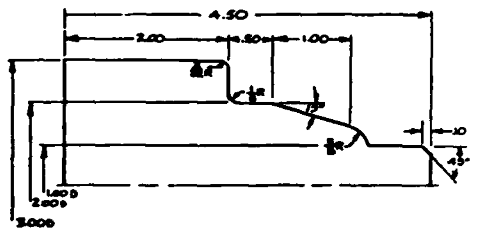
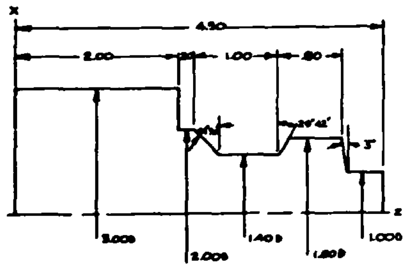
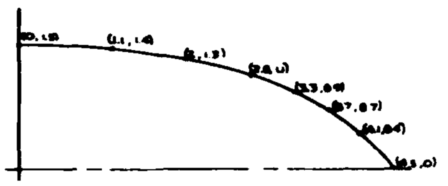
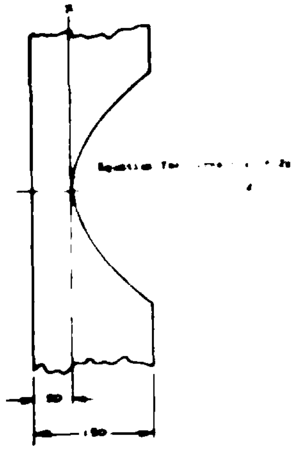
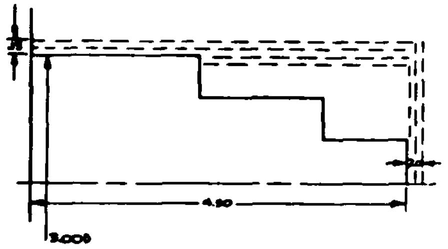
	Defining Geometry	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
	1) Straight lines parallel and perpendicular to the center line – for the finish form	X	X	X		X	X	X	X	X	X	X		X		X
	2) Geometry of rough forging or casting – taper line tangent to circle	X	X	X	See Note, page 84	X	X	X	X	X	X	X	X	A	See Note, page 106	X
	3) Circles, tangent lines, taper line, line intersecting circle	X	X	X		X	X	PA	PA	X	X	X	X	PA		X
	4) Undercut on the finish form	X	X	X		X	X	X	X	X	X	X	X	X		X

TABLE 16 B
TEST PATTERNS

	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
Defining Geometry (cont.)															
5) Splined curve described by a series of coordinates	A	X	X		n a	X	A	A	X	X	X	A	A		A
6) Formula related curve. In this case a parabola defined by the equation $x^2 = 2z$.	A	n a	X	See Note, page 84	n a	X	A	A	n a	n a	X	A	A	See Note, page 106	A
Roughing Operations															
7) Roughing to a geometry pattern having straight lines that are parallel and perpendicular to the Z axis. Facing and turning cuts	X	X	X		X	X	X	X	X	X	X	X	X		X

TABLE 16 C

TEST PATTERNS

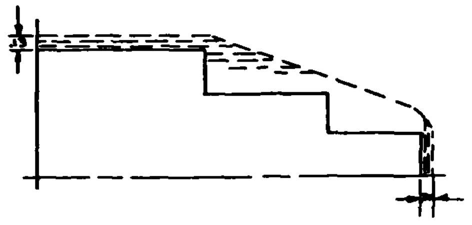
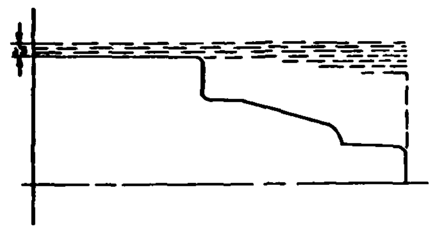
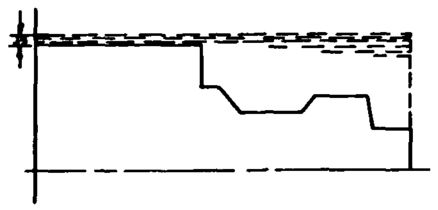
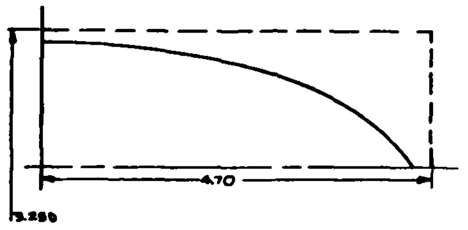
	Roughing Operations (cont.)	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
	8) Roughing to a pattern having straight lines that are parallel and perpendicular to the Z axis. Raw material formed.	X	X	X		X	X	X	X	X	X	X	X	X		X
	9) Roughing to a pattern having straight lines and circular arcs	X	X	X	See Note, page 84	X	X	X	X	X	X	X	X	X	See Note, page 106	X
	10) Roughing involving straight lines and an undercut	X	X	X		X	X	X	X	X	X	X	X	X		X
	11) Roughing to a splined curve	A	n a	X		n a	X	PA	PA	n a	X	X	X	PA		A

TABLE 16 D
TEST PATTERNS

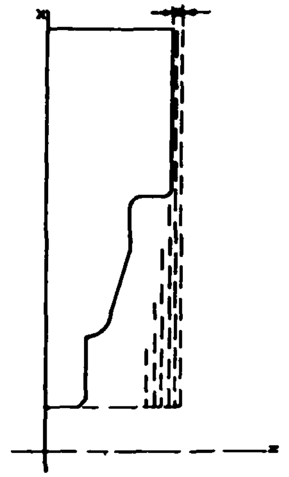
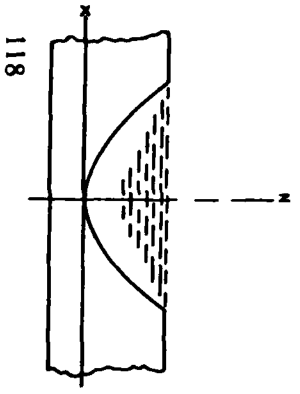
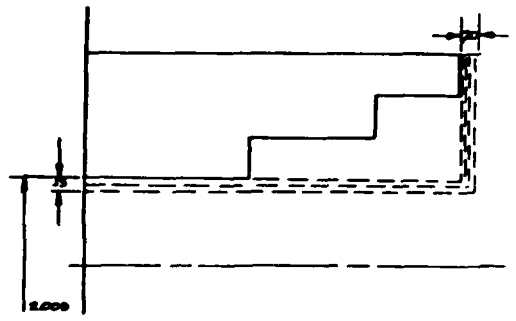
	Roughing Operations (cont.)	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
	12) Roughing on a face	X	X	n a	See Note, page 84	X	X	PA	A	X	X	X	X	PA	See Note, page 106	A
	13) Roughing to a formula defined curve	A	n a	n a	See Note, page 84	n a	X	PA	A	n a	n a	A	PA	PA	See Note, page 106	A
	14) Rough boring and facing	X	X	X	See Note, page 84	X	X	X	A	X	X	X	X	X	See Note, page 106	X

TABLE 16 E

TEST PATTERNS

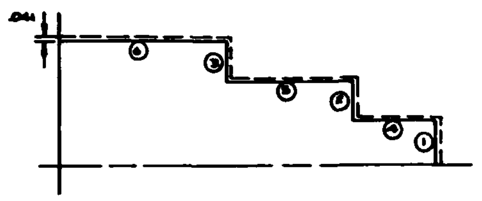
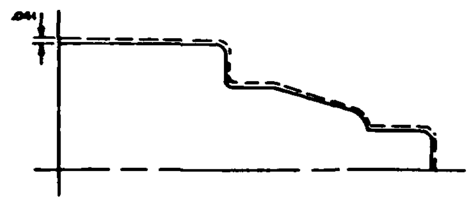
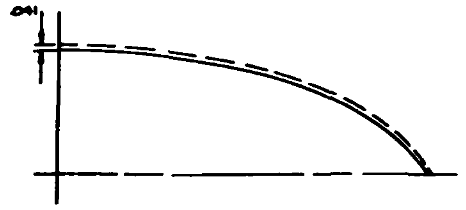
	Finishing Operations	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
	15) Finish random selected surfaces	X	X	X	See Note, page 84	X	X	X	A	X	X	X	X	X	See Note, page 106	X
	16) Finish contour surface	X	X	X		X	X	X	A	X	X	X	X	X		X
	17) Finish spline surface	A	n a	X		n a	X	A	A	n a	X	X	X	A		A

TABLE 16 F
TEST PATTERNS

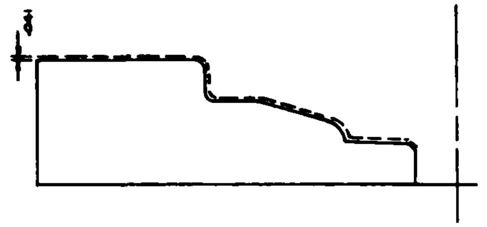
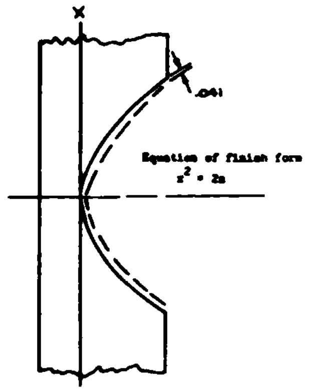
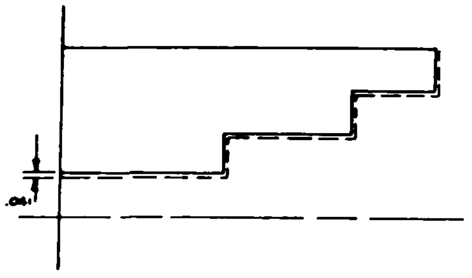
	Finishing Operations (cont.)	Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
	18) Finishing on a face	X	X	X	See Note, page 84	X	X	PA	A	X	X	X	X	X	See Note, page 106	X
	19) Finishing to a formula defined curve	A	n a	n a	See Note, page 84	n a	X	A	A	n a	n a	X	X	A	See Note, page 106	A
	20) Finish on the ID	X	X	X	See Note, page 84	X	X	X	A	X	X	X	X	X	See Note, page 106	X

TABLE 16 G

TEST PATTERNS

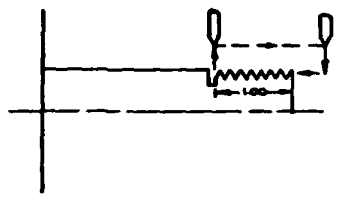
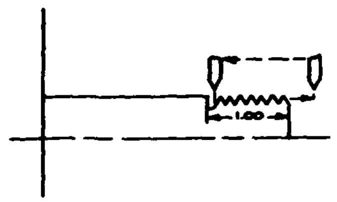
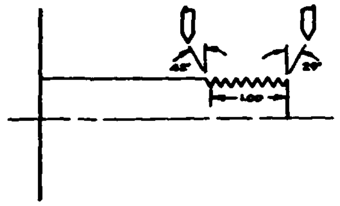
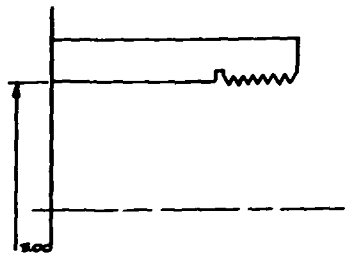
	Threading	**Cincinnati**	**Digital**	**Encode**	**General Electric**	**Ingersoll**	**MDSI**	**MS&S**	**McAuto**	**Olivetti**	**SDRC**	**Threshold**	**United Comp.**	**University C.**	**Weber N/C**	**Westinghouse**
	21) Straight in-feed and pull-out. Thread right to left	X	X	X	See Note, page 84	X	X	X	X	X	X	See Note, page 114	X	X	See Note, page 106	X
	22) Straight in-feed and pull-out. Thread left to right	X	X	X		X	X	X	X	X	X		X	X		X
	23) Compound in-feed and pull-out in lead	X	X	X		X	X	X	n a	X	X		X	X		X
	24) Threading on the ID	X	X	X		X	X	X	X	X	X		X	X		X

TABLE 16 H
TEST PATTERNS

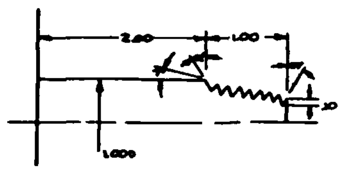
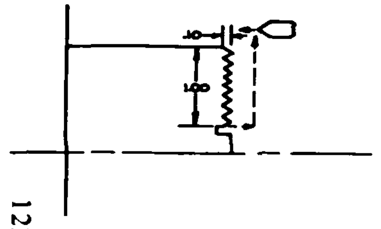
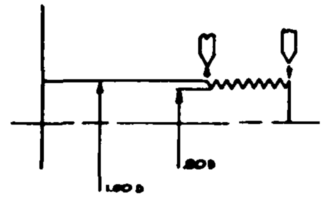
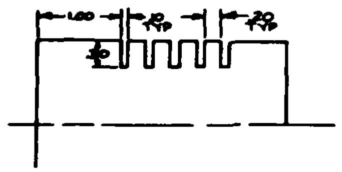
		Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
	Threading (cont.)															
	25) Threading on a taper	X	X	X	See Note, page 84	n a	X	X	X	X	X	See Note, page 114	X	X	See Note, page 106	X
	26) Threading on a face	A	X	n a		n a	X	X	A	X	X		X	X		n a
	27) Multiple threading	X	X	X		n a	X	X	X	X	X		X	X		X
	Grooving															
	28) Grooves evenly spaced, perpendicular to the Z axis	X	X	X		X	X	PA	X	X	X		X	X		X

TABLE 16 I

TEST PATTERNS

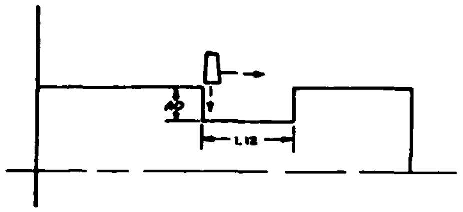
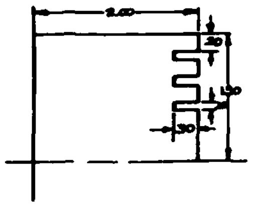
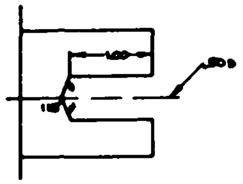
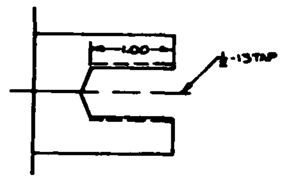
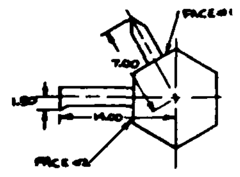
		Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
	Grooving (cont.)															
	29) Grooving for an undercut	X	X	X	See Note, page 84	X	X	X	X	X	X	See Note, page 114	X	X	See Note, page 106	X
	30) Grooving on the face	X	X	X		X	X	PA	X	X	X		X	X		X
	Drilling															
	31) Drilling	X	X	X		X	X	PA	X	X	X		A	X		X

TABLE 16 J
TEST PATTERNS

		Cincinnati	Digital	Encode	General Electric	Ingersoll	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United Comp.	University C.	Weber N/C	Westinghouse
	Tapping															
	32) Tapping operation	X	X	X	See Note, page 84	X	X	X	X	X	X	See Note, page 114	X	A	See Note, page 106	X
	Tool Orientation															
	33) Describing the location and orientation of tools	X	n a	X		X	X	A	X	X	X		X	X		X

CHAPTER IX

BENCHMARK TESTING

Benchmark tests were carried out to permit each of the fifteen systems to demonstrate its capabilities for NC lathe programming. Ten test parts were carefully designed, and a test procedure was devised to insure impartiality in the conduct of the tests. One of the contractor's investigators conducted each test at the proponent's site, observing and recording the programming times required. The programs were verified.

There is a distinction between the characteristics built into a programming system by its designer, and the performance of the system when in use in a specific user's environment. The preceding Chapters IV through VII have categorized and compared the complete set of capabilities claimed for each of the fifteen systems. It is not possible to put each of these capabilities to the test in a report of this magnitude. The most important have been selected and tested. This and following chapters describe the tests which each proponent was asked to perform.

A. Design of the Test Parts

In the previous study covering milling machines and machining centers, ECOM 0058-F, ten test parts were designed and the proponents of each of the seven languages covered in that report wrote programs for them insofar as they were able. In addition, the same test parts were programmed by users of the systems. The results of the user programs were not as statistically significant as might be desired, since, in a few cases, the sample size was relatively small.

A similar procedure was agreed upon for this evaluation, but with some modifications. Once again the test parts were based upon common industrial and governmental lathe part designs, several hundred of which were collected, examined, and classified. From these drawings, the ten test parts for this study were synthesized. Whereas the previous evaluation had involved test programming by both proponents and users, this contract asks only that test parts be programmed by the proponents. We expected, and found, that they assigned their most expert programmers to the task, which has minimized the human element of programming and emphasized the effect of language structure and processor design. Further to minimize the role of the programmer, each test part design was accompanied by detailed method sheets, tool descriptions and layouts, and illustrations of the successive operations.

The contractor was faced with the dilemma of balancing the number of test parts with the complexity of part design. While it is desirable to minimize the number of test parts by piling a goodly number of test tasks on each part, our previous experience showed that if this were done, one system might successfully do — say — five out of the six tasks, another four out of the six, and another a different five out of the six, with the result that no valid comparisons would be possible on the overall test times.

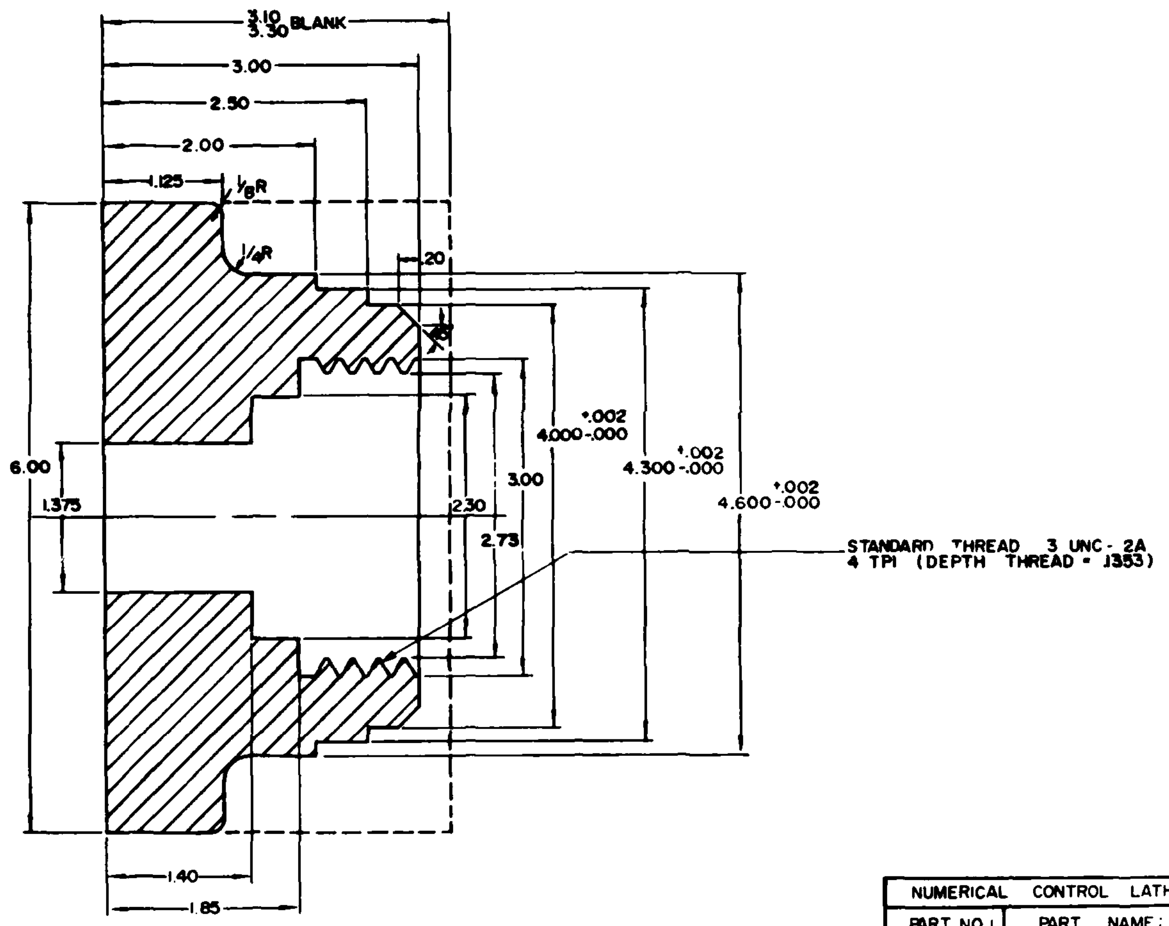
It was concluded, therefore, that in this evaluation if there were to be more than one test task per part, then all the tasks on a given part must be consistent as to their level of complexity, and that any system that could do one of the tasks should be expected to do them all. However, due to the variation in the capabilities of the fifteen systems, it is almost impossible to meet this objective in the more complex programming tasks. There appeared no way to make consistently graduated test parts, on any one of which any one of the fifteen systems would be expected to do all, or — in effect — none of the tasks.

Therefore the first three test parts are designed as complete parts with graduated complexity; all the tasks on any one should be within the capability of a given system, and the programming times should be comparable. The remaining seven parts require the demonstration of a specific system capability. Each system then will either meet the challenge or have to pass it up. For those that meet the challenge, the test results will be comparable.

Initially a set of test parts was designed and then programmed at six government installations (Army, Navy, and Air Force). This served to check the part design drawings and method sheets, and to give the contractor an initial test run in a non-competitive environment. Testing procedures were improved, and minor corrections were made in the test part drawings. The cooperation of the government installations in this matter is acknowledged with thanks.

To further put the proponents' systems on an equal footing and to avoid the burden of writing a postprocessor, it was determined that all proponents had postprocessors in existence for the Warner & Swasey 1-SC or 2-SC, or for the Cincinnati Milacron Cinturn, lathes. Operation sheets and tool layouts were therefore prepared for all ten test parts for both Warner & Swasey and Cinturn lathes.

The drawings for all ten test parts are reproduced in Figure 20, and Figures 22 through 30 on the following pages. A sample set of the operation sheets, tooling descriptions and layouts, and illustrations of the operations, is reproduced as Figure 21 A-F. The sample relates to Test Part 1.



NUMERICAL CONTROL LATHE LANGUAGE EVALUATION	
PART NO. 1	PART NAME: HOUSING
PREPARED BY: ADVANCED COMPUTER SYSTEMS, INC. (CONTRACTOR)	
SPONSOR: U.S. ARMY, CORADCOM	DATE:

FIGURE 20

SHEET 1 OF 3

OPERATION SHEET

NUMERICAL CONTROL LATHE LANGUAGE EVALUATION

SPONSOR U. S. ARMY — CORADCOM

PART NO. 1 PART NAME Housing MATERIAL Steel SAE 1020

OP. NO.	OPERATION	TOOLING	REMARKS
1	Face part	0° Facing Tool	Speed (rough cuts) = 300 sfm
	Max. depth of rough cuts = .100	Tool Set A	Feed (rough cuts) = .020 ipr
	Depth of finish cut = .020		Speed (finish cut) = 400 sfm
			Feed (finish cut) = .010 ipr
2	Turn OD	5° Turning & Facing Tool	Speed (rough cuts) = 300 sfm
	Max. depth of rough cuts = .100	Tool Set B	Feed (rough cuts) = .020 ipr
	Depths of each of two finish cuts		Speed (finish cuts) = 400 sfm
	= .020		Feed (finish cuts) = .015 ipr
	See NOTE under REMARKS column		NOTE: If programming language has
			capability, incorporate an optional
			stop at the end of the first finish
			cut to allow operator to make tool
			offset adjustment.

SHEET 2 OF 3

OPERATION SHEET

NUMERICAL CONTROL LATHE LANGUAGE EVALUATION

SPONSOR U. S. ARMY — CORADCOM

PART NO. 1 PART NAME Housing MATERIAL Steel SAE 1020

OP. NO.	OPERATION	TOOLING	REMARKS
3	Drill through center	1 3/8 HSS Drill	Tip of tool to extend .5" from
		Tool Set C	back of part.
			Speed = 235 rpm
			Feed = .020 ipr
4	Bore ID	0^{o} Boring Tool	Speed (rough cuts) = 300 sfm
	Maximum depth of rough cuts = .060	Tool Set D	Feed (rough cuts) = .030 ipr
	Finish pass = .010		Speed (finish cut = 350 sfm
			Feed (finish cut) = .015 ipr
5	Cut groove	ID Grooving Tool	Speed = 300 sfm
		Tool Set E	Feed = .003 ipr
			NOTE: Program minimum of two
			revolutions of dwell at bottom
			of groove

SHEET 3 OF 3

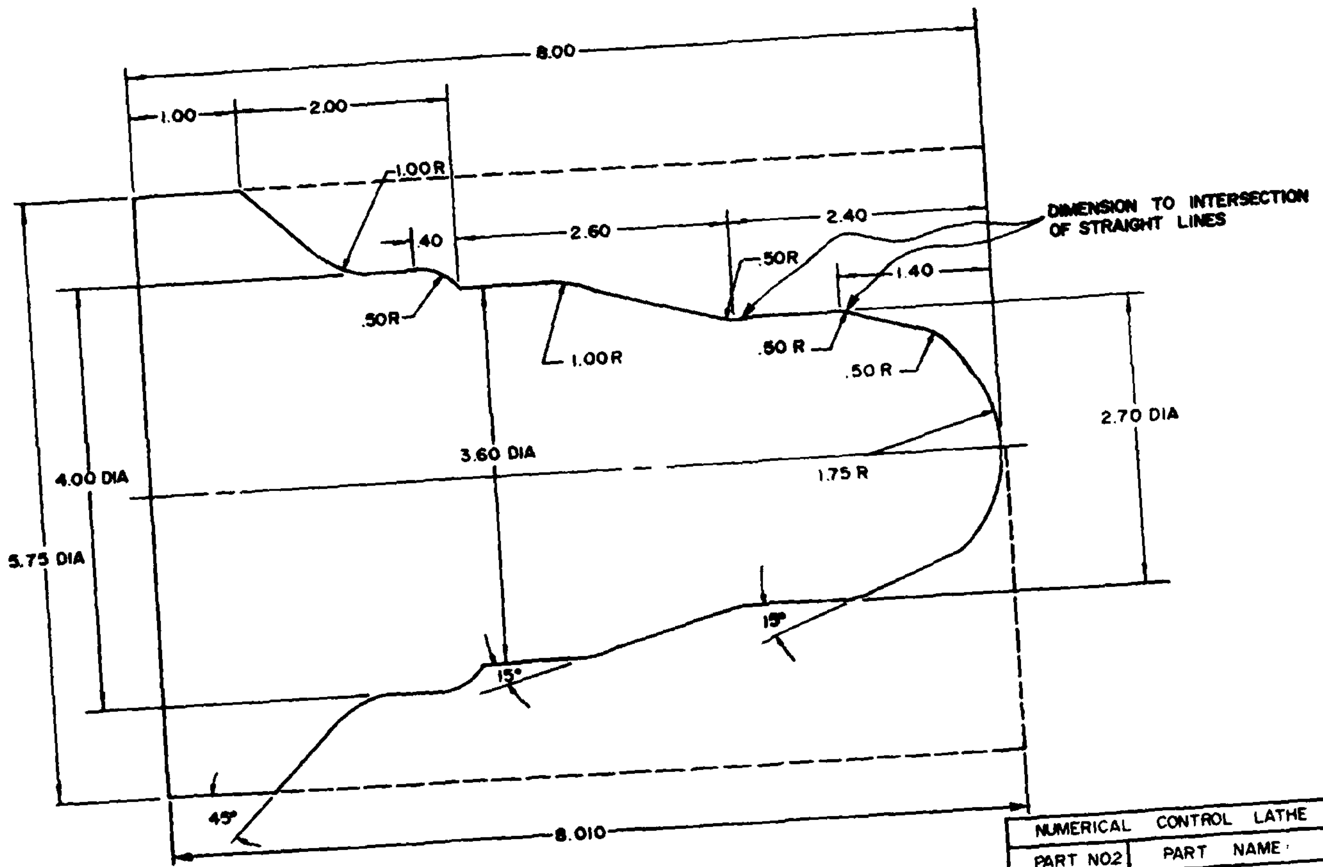
OPERATION SHEET

NUMERICAL CONTROL LATHE LANGUAGE EVALUATION

SPONSOR U. S. ARMY — CORADCOM

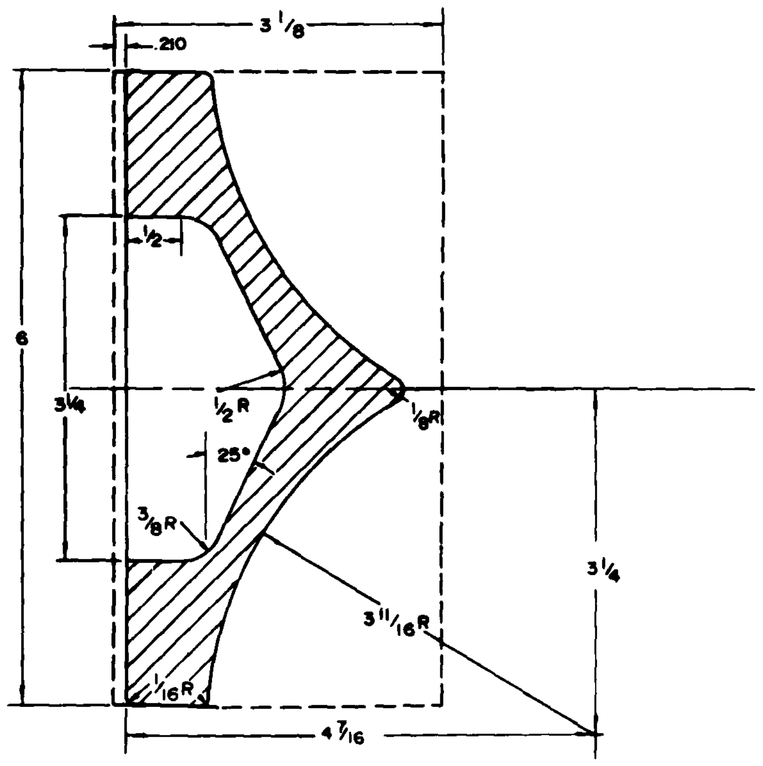
PART NO. 1 PART NAME Housing MATERIAL Steel SAE 1020

OP. NO.	OPERATION	TOOLING	REMARKS
6	Cut ID thread	Threading Tool	
	Max. depth of rough cuts = .006	Tool Set F	
	Max. depth of two finish cuts = .001		
	One clean-up pass at full depth		



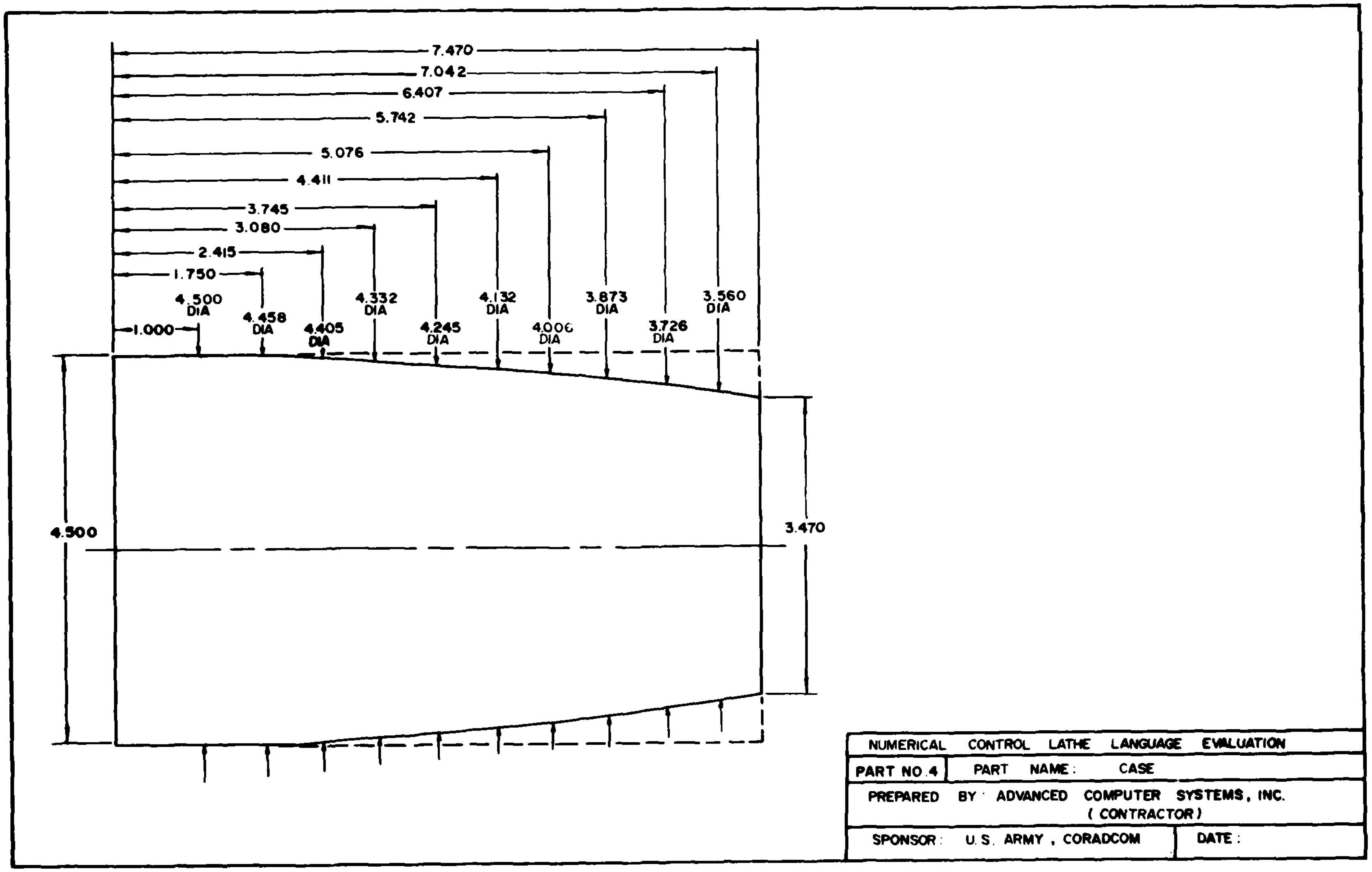
NUMERICAL CONTROL LATHE LANGUAGE EVALUATION		
PART NO2	PART NAME: PUNCH	
PREPARED BY: ADVANCED COMPUTER SYSTEMS, INC. (CONTRACTOR)		
SPONSOR: U.S. ARMY, CORADCOM		DATE:

FIGURE 22



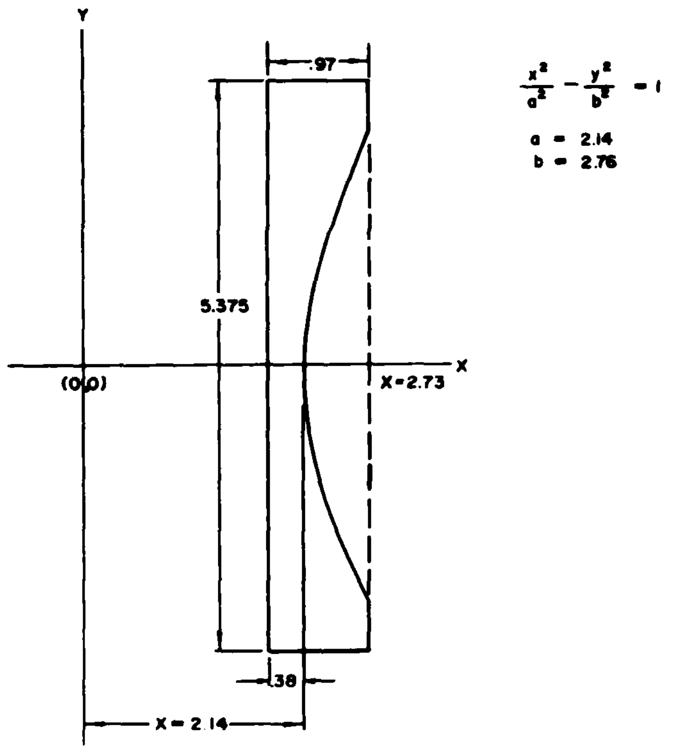
NUMERICAL CONTROL LATHE LANGUAGE EVALUATION	
PART NO 3	PART NAME CAF
PREPARED BY: ADVANCED COMPUTER SYSTEMS, INC. (CONTRACTOR)	
SPONSOR: U.S. ARMY, CORADCOM	DATE:

FIGURE 23



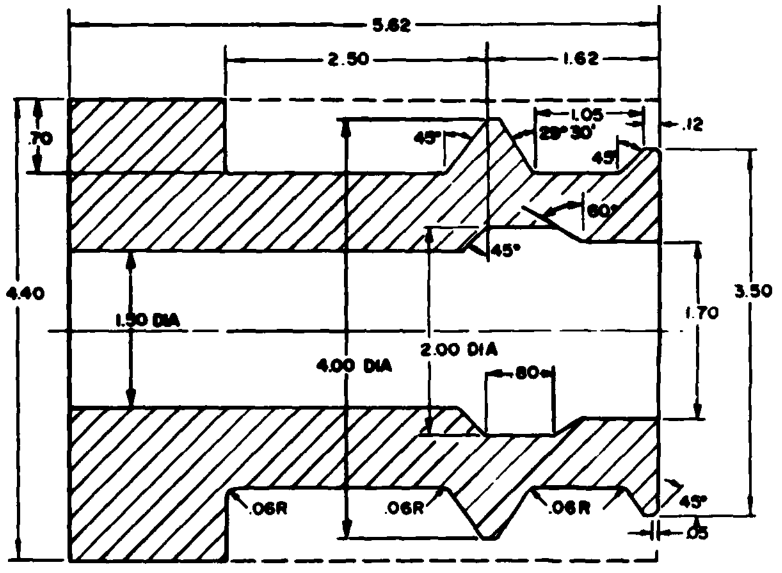
NUMERICAL CONTROL LATHE LANGUAGE EVALUATION	
PART NO.4	PART NAME: CASE
PREPARED BY: ADVANCED COMPUTER SYSTEMS, INC. (CONTRACTOR)	
SPONSOR: U.S. ARMY, CORADCOM	DATE:

FIGURE 24



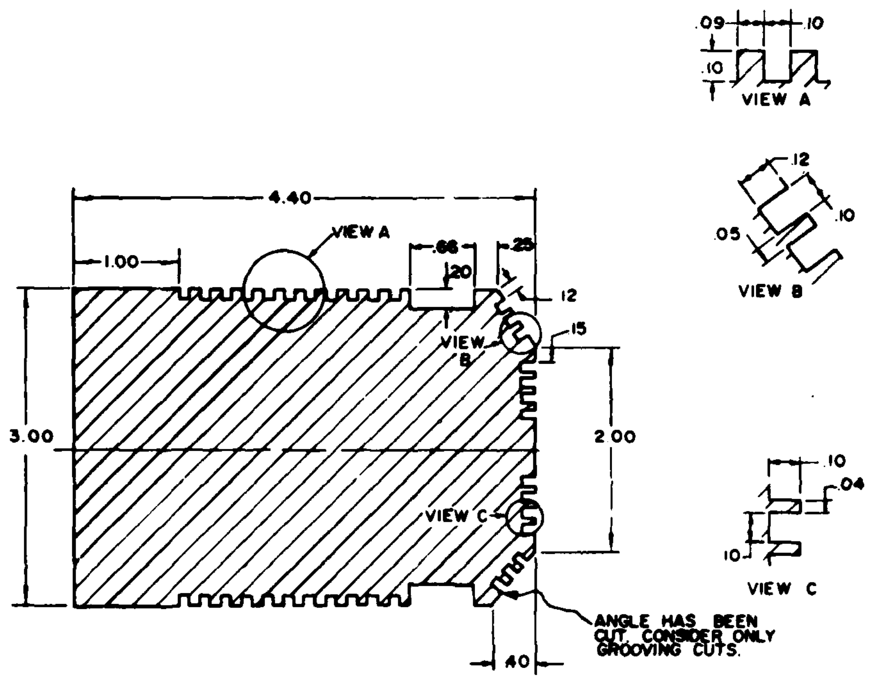
NUMERICAL CONTROL LATHE LANGUAGE EVALUATION	
PART NO.5	PART NAME: REFLECTOR
PREPARED BY: ADVANCED COMPUTER SYSTEMS, INC. (CONTRACTOR)	
SPONSOR: U.S. ARMY, CORADCOM	DATE :

FIGURE 25



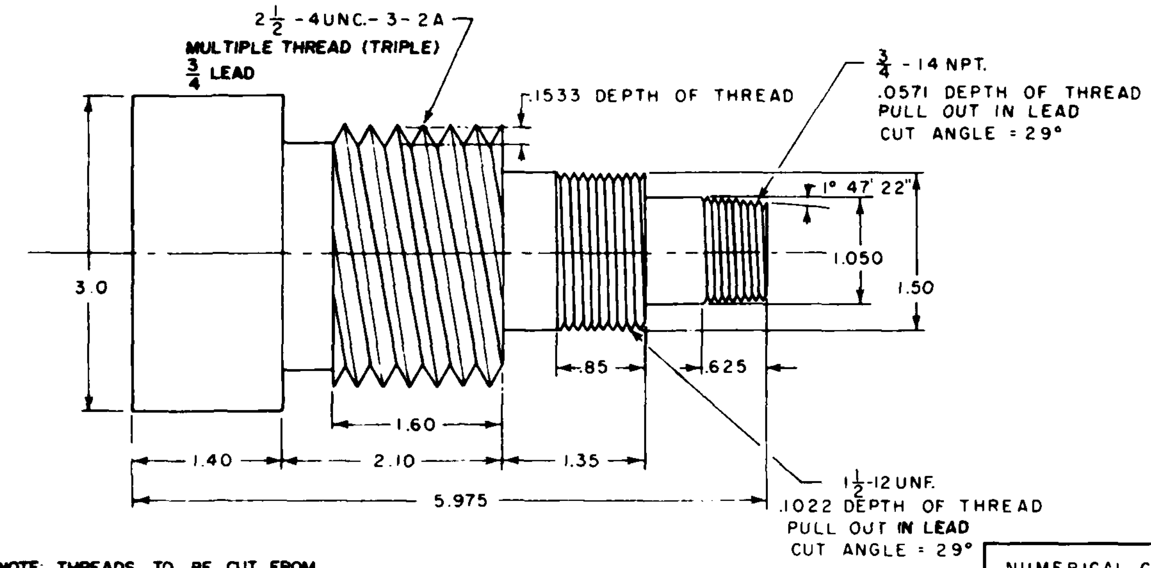
NUMERICAL CONTROL LATHE LANGUAGE EVALUATION		
PART NO. 6	PART NAME: CONNECTOR	
PREPARED BY: ADVANCED COMPUTER SYSTEMS, INC. (CONTRACTOR)		
SPONSOR: U.S. ARMY, CORADCOM		DATE:

FIGURE 26



NUMERICAL CONTROL LATHE LANGUAGE EVALUATION	
PART NO.7	PART NAME: BODY
PREPARED BY: ADVANCED COMPUTER SYSTEMS, INC. (CONTRACTOR)	
SPONSOR U.S. ARMY, CORADCOM	DATE:

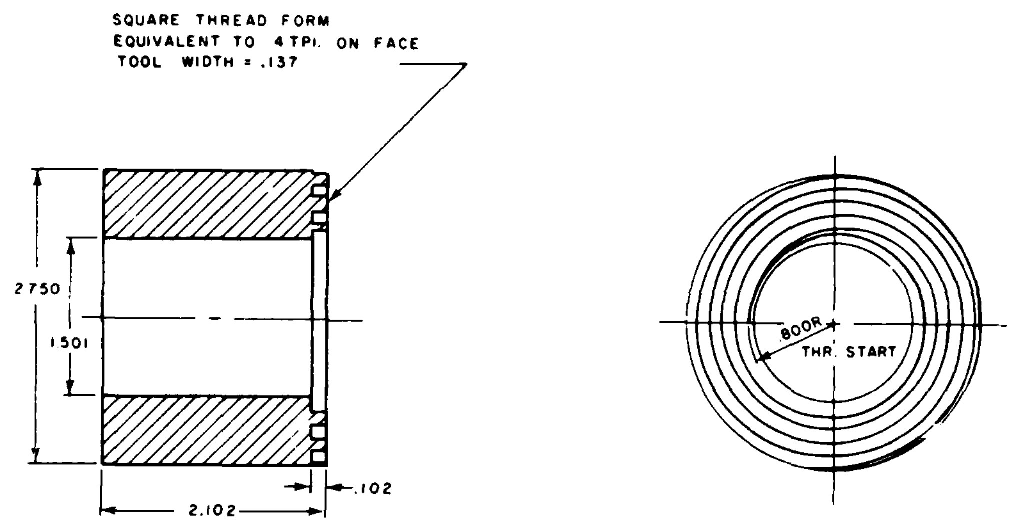
FIGURE 27



NOTE: THREADS TO BE CUT FROM
SEMI-FINISHED PART PROGRAM
ONLY THREAD CUTTING

NUMERICAL CONTROL LATHE LANGUAGE EVALUATION	
PART NO 8	PART NAME FITTING
PREPARED BY: ADVANCED COMPUTER SYSTEMS INC. (CONTRACTOR)	
SPONSORED BY: U.S. ARMY CORADCOM	DATE:

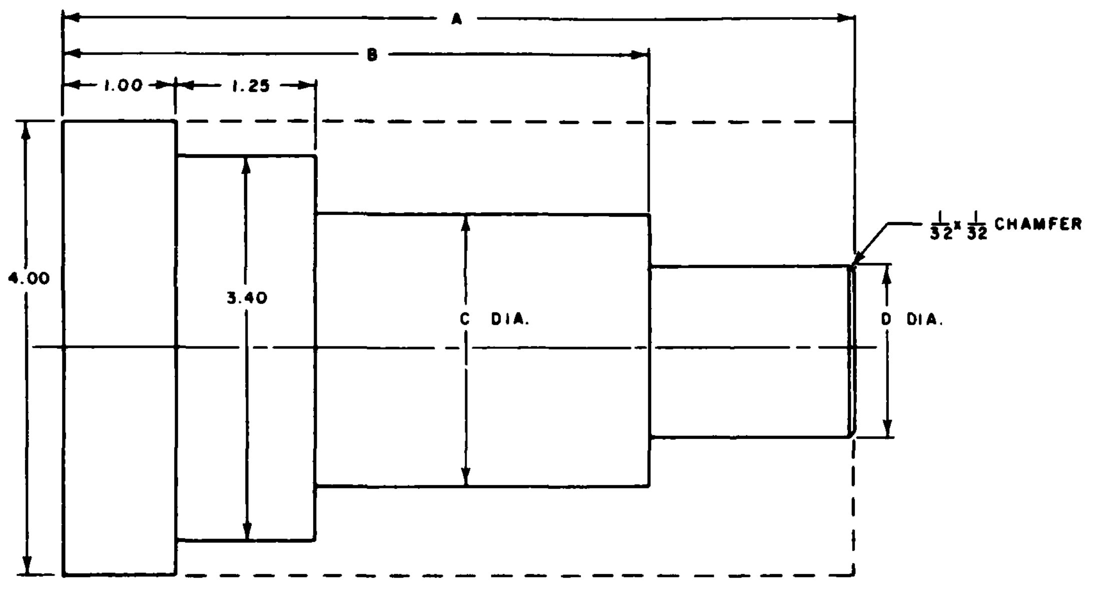
FIGURE 28



NOTE: HOLE PREDRILLED & FACE BORED TO 1.60 DIA
PROGRAM ONLY THREAD ON FACE

NUMERICAL CONTROL LATHE LANGUAGE EVALUATION	
PART NO: 9	PART NAME: SCROLL
PREPARED BY: ADVANCED COMPUTER SYSTEMS INC. (CONTRACTOR)	
SPONSORED BY: U.S ARMY CORADCOM	DATE:

FIGURE 29



"FAMILY OF PARTS" TABLE

PART NO.	A	B	C	D
11738-1	5.60	4.10	2.05	1.35
11738-2	5.85	4.30	2.12	1.30
→ 11738-3	6.15	4.55	2.20	1.34
11738-4	6.30	4.75	2.25	1.38
→ 11738-5	6.80	5.00	2.30	1.45
11738-6	7.00	5.20	2.40	1.50

NUMERICAL CONTROL LATHE LANGUAGE EVALUATION	
PART NO: 10	PART NAME: POST
PREPARED BY: ADVANCED COMPUTER SYSTEMS INC. (CONTRACTOR)	
SPONSORED BY: U.S. ARMY CORADCOM	DATE:

FIGURE 30

B. Conduct of the Tests

Conduct of the tests was completely impartial. No proponent saw the test part designs until his programmer sat down to work. The contractor observed the tests, and kept track of the elapsed times. It is appropriate to comment on what was being timed. Ordinarily the programming task is divided into four parts:

- the time the programmer takes to study the drawing, operation sheet, tool layout and illustration of the operation;
- the time to write the source program and commit it to the computer;
- the rewrite and debug time; and
- the time the computer takes to process the source program into an output tape.

With regard to the first three of these four parts, there are two factors which enter into the time spent by the programmer. One is his personal skill level, and the other involves the characteristics of the particular system in use. It was assumed that the proponents put their very best programmer on the job. Obviously, he would not be available to a user, but his programming time can be taken as an indication of the minimum time for any similar part on his particular system. Thus the time difference between the systems may be credited to the systems' characteristics rather than to the programmers' skills.

In order to treat all systems equitably, the first two tasks described above have been combined and the total time for preparing a tape was recorded in three categories, namely:

1. The time to review the material (print, method sheets, etc.) plus the time to write the part program.
2. The time to debug and rewrite the program.
3. The time to process the program — either the measured time on a dedicated local computer or the time equivalent charged by a remote time-share system — and produce the tape.

Most systems have the programmer prepare and review the part program on a manuscript paper, which may then be keyed into the system by a clerk/typist, or by the programmer himself. In general, programmers are relatively slow typists, and prefer to turn over the copying work to speedier keyboarders. Because the keyboarding is merely a data transmission function, and does not contribute to the programming function, keyboard time when it was separately recorded has been excluded from the measurement of the system performance, and not considered a factor for comparison.

Regarding item 2 above, syntax and other program input errors generally caused an error message and required a correction before the program could be processed successfully. In a few systems with a conversational mode of operation, the correction was immediate and generally rapid, and would be considered as part of the part programming time, category 1 above. With most systems, the corrections were made following the program input, and could be timed. Errors in dimensions or in motion statements were immediately apparent with a few systems where on-line plotting was available. Here too the correction could be accomplished in a relatively short time. Other systems did not detect these types of errors until the program was committed to the processor; a correction and a rerun was necessary. Timing of error correction was not easily comparable in all systems.

Regarding item 3 above, timing the computer may or may not be significant, depending on how the system is operating. For stand-alone systems with a dedicated mini-computer, the time to compute is of interest because the system cannot do *anything else while it has a part program* in process. Parts with long computational sequences can be the limiting constraint on how many part programs a system can handle over a given time period such as a day. The speed of computation depends directly on the computer equipment available; a bare mini may take minutes where the same mini with two peripheral disks and a high speed printer may take a fraction of a minute for the same task. Therefore the precise computer equipment used was recorded, and must be taken into account in any comparative evaluation.

For time share systems, the time to process a program determines the cost of the service. Processing includes transmission to the remote computer, time in the computer's central processing unit (CPU) and return transmission to the programmer. Charges may be made for the "connect time" during which a user is employing one of the input/output channels of the computer, and calculation time during which the CPU is employed in processing the user's program. While CPU time on large computers is expensive relative to the mini-computer, CPU times are short, and the user is not charged for the big CPU when he is not using it. He may however pay for program storage, such as for his postprocessors, and for minimum usage.

For a batch remote mode of operation, the user may elect to defer use of the CPU to a low cost time period, such as nights and week ends, and accept the longer turnaround time for completion of his program. Several scales of charges are usually available, depending on how long a delay is tolerable. While the time in the CPU will be the same, the cost will vary. If the program is urgently needed, the batch mode user will enter the "immediate priority" queue, and the CPU will do the processing when the program gets to the head of that queue, and at a premium charge.

For interactive remote modes of operation, the system accepts statements from the user, processes them and responds, and then turns to serving other users while the return transmission, the user's reaction and next transmission takes place. The next statement of the user input then awaits in queue (usually a fraction of a second) for the attention of the CPU. At rush hours, usually between 2:00 and 5:00 PM, when a system is usually loaded with users contending for attention, the response time may be slowed.

A prospective user of NC lathe programming systems will be interested in the cost per program. Two quite different cost comparisons are necessary.

For the dedicated computers, the capital equipment cost can be converted to a cost per hour by well known accounting methods. Knowing how long a dedicated computer will be tied up with a given program will permit the calculation of a cost per program. These times were easily acquired in the tests and are reported in the next chapter.

For the time shared computer systems, it was not so easy. The system will bill the user for the connect time and for CPU time, and the cost per program can be calculated. However, there are so many variables in computing charges that the data collected was not properly comparable.

Therefore, proven source programs from Test Parts 1,2,6,7, and 8 prepared by the proponents of shared time systems were submitted to users of the respective systems. These users simply entered the source programs into the system as if they were their own, using the highest priority, and reported back the time and charges for the tape, made to them by the system. These actual costs may be scaled back if desired to convert to lower priority running conditions. These costs are listed in the next chapter, and serve as one factor in the comparison of the various systems. We thank the system users whose cooperation made this comparison possible.

C. Verification

Each test part output program was verified by the proponent by plotting or an equivalent procedure. Verification time was noted. In addition, the contractor re-verified selected tapes by plotting on a Tektronix Tape Verifier at ARRADCOM (Picatinny Arsenal, Dover, N. J.)

CHAPTER X

TEST RESULTS

Performance times of each system are shown for each test part completed by the system. Processing and tape production costs for time shared remote systems are tabulated insofar as available. To help the reader to comprehend the significance of over 150 data items, a comparative analysis has been made: the test parts are grouped into four categories. The average time per part is shown for each system for each category, charted from the lowest to the highest average time. The relative performance of APT vs. non-APT systems, and of remote vs. stand-alone computer operations are also shown. Cost comparison procedures are outlined.

As pointed out in Chapter II (Page 17), the findings of such a study as this cannot be reduced to a single number — a figure of merit — for each system studied. There are many parameters by which a system is judged and selected. This chapter presents the times and in some cases the costs necessary to program the test parts.

A. Total Time to Program

Table 17 consists of three charts showing, in both graphic and numeric form, the number of minutes each of the proponents took to program those of the ten test parts which he could do. Although the performance time was recorded in several categories (see page 140) the figures shown in Table 17 are the total time — review of the print, operation sheets and tool layout, plus the time to write the part program, plus the time to debug and verify the program. The contractor feels this to be the most representative figure for the programming time performance of a system.

A few test part program times were very much longer than the rest. To preserve a readable scale for the other data, these long times have been graphed with a broken bar, to indicate that they are not to be compared visually to the adjacent bars. Each bar on the chart has noted below it the total time figure it represents. Where a system did not have the capability to program a part, the bar is absent and an asterisk notes this fact.

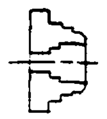
For reasons we could not determine, there were occasions during the tests upon which a system failed to program a test part, although it might reasonably have been expected to do so. Success at a later date could not be counted, as the ground rules required that the programmer had not previously seen the drawings and operation sheets when he began to work. Examples are Westinghouse on part #2 and United on parts #3 and #6. Similarly, Digital's five and a half hour struggle with part #1 is inconsistent with their performance on other parts and is indicative of some unusual situation. Nevertheless, the records must stand as recorded.

The system consistently showing the fastest programming time is Cincinnati Milacron. The system is highly automatic and compact, but most significantly is restricted to programming for Cincinnati Milacron lathes only, in its present form. See page 78. It is an APT based system and therefore could use APT statements in programming some of the more complex parts, such as parts #4 and #5.

TABLE 17A

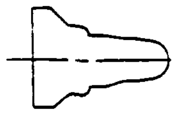
TOTAL TIME TO PROGRAM

PART #1 Housing



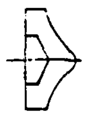
Common type of part involving facing, straight turning, boring, chamfering, simple threading

PART #2 Punch



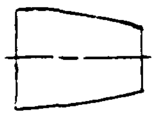
Geometry oriented part. Tangent and intersecting lines and circles

PART #3 Cap

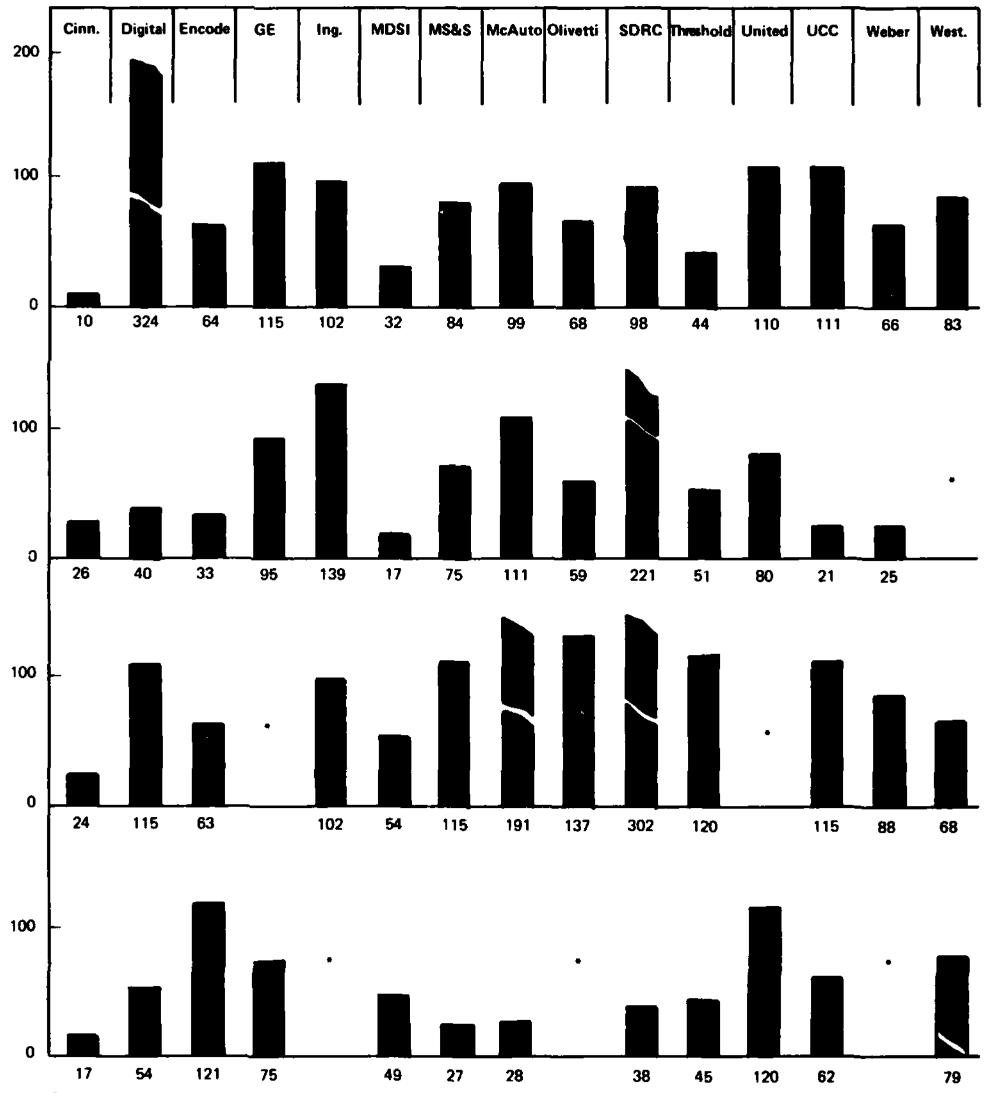


Contour on a face with tangent circles

PART #4 Case



Splined curve described by points



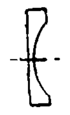
	Cinn.	Digital	Encode	GE	Ing.	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United	UCC	Weber	West.
PART #1	10	324	64	115	102	32	84	99	68	98	44	110	111	66	83
PART #2	26	40	33	95	139	17	75	111	59	221	51	80	21	25	•
PART #3	24	115	63	•	102	54	115	191	137	302	120	•	115	88	68
PART #4	17	54	121	75	•	49	27	28	•	38	45	120	62	•	79

•Did not do during tests.

TABLE 17B

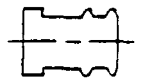
TOTAL TIME TO PROGRAM

PART #5
Reflector



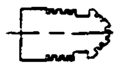
Formula defined curve on a face (hyperbola)

PART #6
Connector



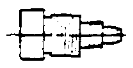
Undercuts; internal and external

PART #7
Body

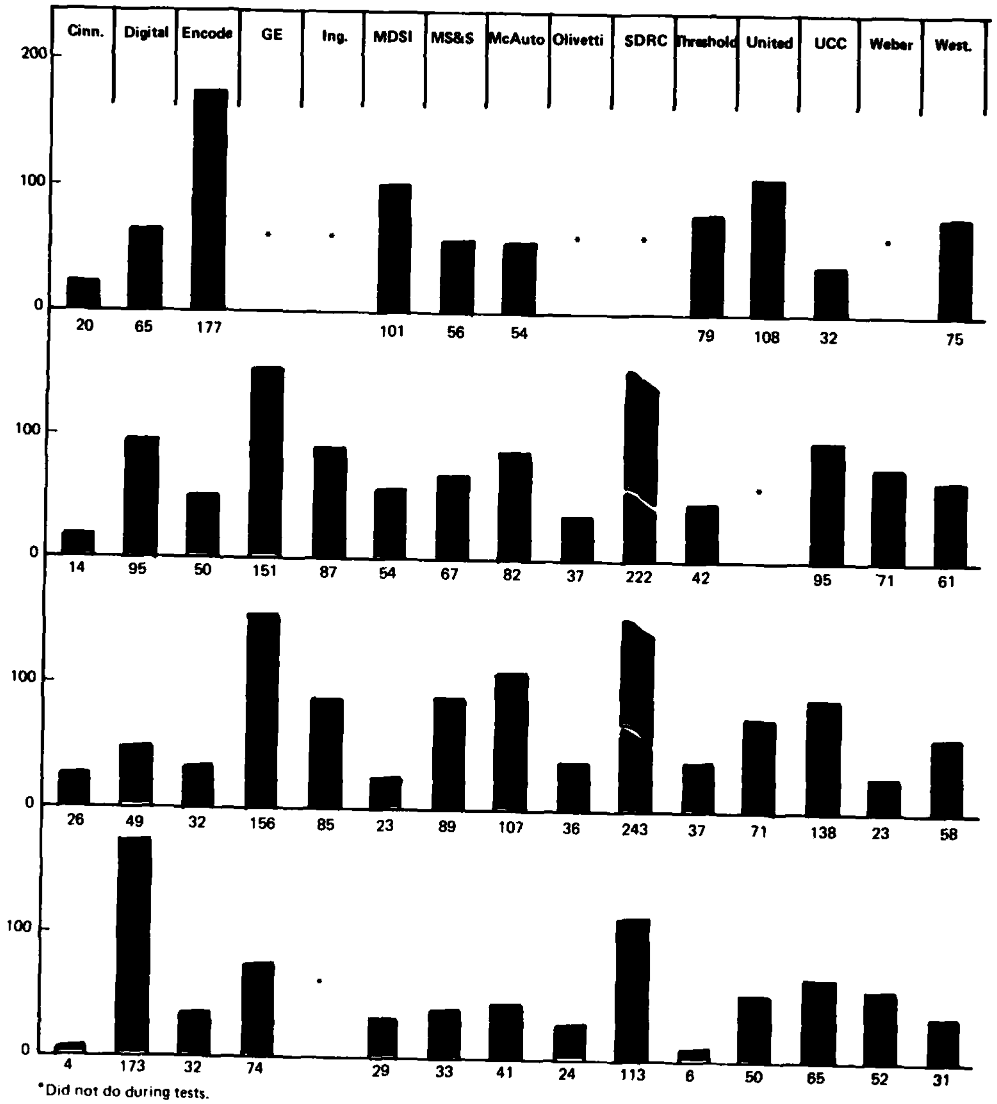


Grooving, four types

PART #8
Fitting



Threading; straight, tapered, multi-start

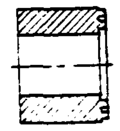


*Did not do during tests.

TABLE 17C

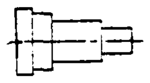
TOTAL TIME TO PROGRAM

PART #9
Scroll

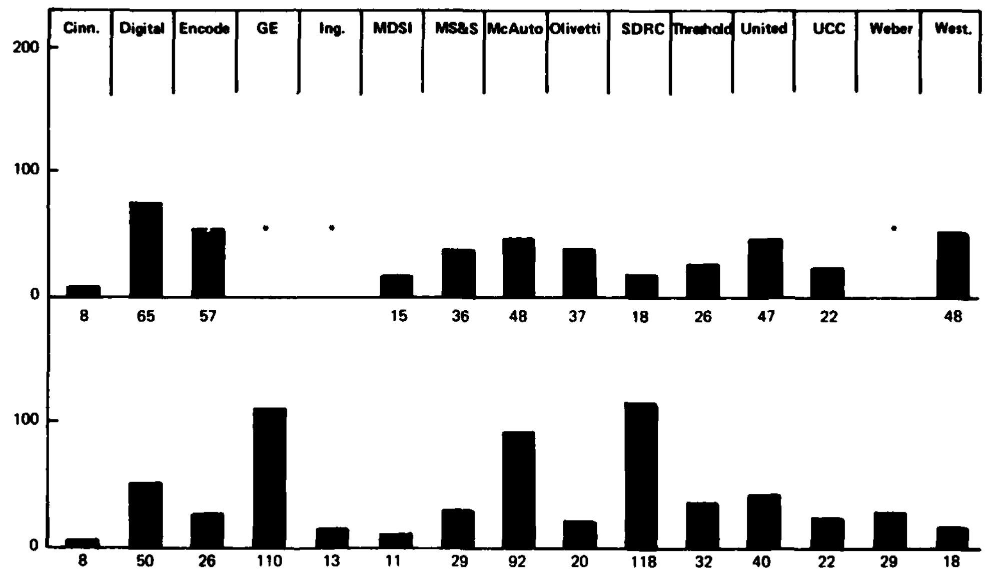


Threading on a face

PART #10
Post



Family of parts programming



	Cinn.	Digital	Encode	GE	Ing.	MDSI	MS&S	McAuto	Olivetti	SDRC	Threshold	United	UCC	Weber	West.
Threading on a face	8	65	57	*	*	15	36	48	37	18	26	47	22	*	48
Family of parts programming	8	50	26	110	13	11	29	92	20	118	32	40	22	29	18

*Did not do during tests.

A number of the systems were unable to handle all of the more complex parts — notably General Electric, Ingersoll, Olivetti, SDRC, and Weber. Parts #4 and #5, which fall in this category, were handled by a spline program or a formula routine respectively by the proponents. An exception was Digital, which passed circular arcs through three-point segments along the curves. These circular arcs were then handled by circular interpolation in the control system. It should be noted that the APT based systems were generally better able to handle these complex curves.

B. Comparative Performances

Since the Army, and indeed most of the Department of Defense, is heavily oriented to the use of APT, it is appropriate to compare the performance of APT based systems with non-APT based systems.

To make such a comparison meaningfully, the ten test parts, which represent the full gamut of lathe part designs, were grouped into four categories of complexity.

- Parts #1, #6, and #10 have profiles composed of straight lines.
- Parts #2 and #3 have profiles involving both straight lines and circular arcs.
- Parts #7, #8, and #9 are threaded and grooved parts.
- Parts #4 and #5 have profiles defined by tabulated or mathematical curves.

Table 18 has been designed to show the comparative performances of the systems in each category. The total time for the two or three parts in the category was determined, and the bar chart constructed, beginning at the left with the lowest time and progressing to the right with the longest time. When in some cases a system did not program all the parts in a category, no bar was plotted but the omissions are noted in the right margin. The median of the average times is shown by a horizontal line.

Time measurement during the tests was no more precise than the nearest minute. As each of the total time records consists of three or four parts, the overall tolerance on times should be regarded as no less than plus or minus two minutes. In the discussion of Tables 18 and 19 below, note that this tolerance could cause two systems to exchange positions in the rank ordering. For example:

— in category 1, Threshold and Olivetti, or Westinghouse and Weber;
— in category 2, MS&S and Olivetti;
— in category 3, MDSI and Threshold, or MS&S and United;
— in category 4, McAuto and MS&S, or MDSI and Westinghouse, or Digital and Threshold.

To differentiate between the APT and the non-APT systems, bars for the APT based systems are dark color and the bars for the non-APT based systems are light colored. Table 18 shows that the non-APT based systems generally performed better than did the APT based systems. Of those systems performing better than the median,

5 of the 7 in Category 1 (simple straight line parts)
4 of the 6 in Category 2 (circular arc parts)
4 of the 6 in Category 3 (threaded parts)

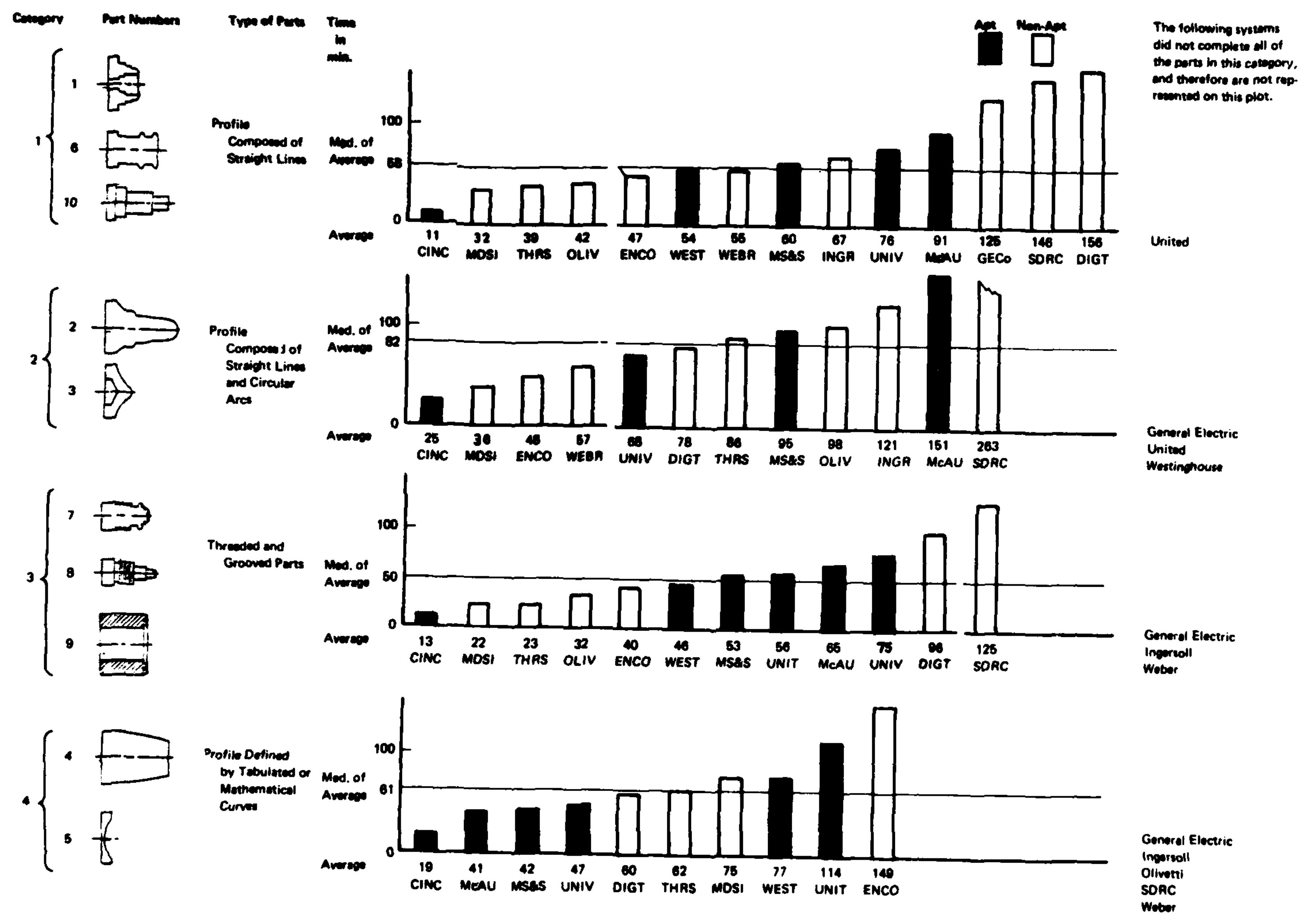


TABLE 18
COMPARATIVE PERFORMANCE, APT VS. NON-APT SYSTEMS

are all non-APT languages and systems. The reverse is true of the parts in Category 4, the more complex parts, where only one of the five systems better than the median is a non-APT based system.

This is not an indictment of the APT concept, as witness the exceptionally strong performance of the Cincinnati system, which is APT based. It does indicate that the three-dimensional orientation of APT is cumbersome for the handling of the two-dimensional requirements of lathe part programming and the detailed description of loops and macros is inefficient, *unless* the language and the processor of the APT system have been sufficiently modified for the purpose.

On the other hand, an APT based system can handle a broader range of requirements such as the more complex geometry exemplified in Category 4. MDSI, Threshold, and Encode — all non-APT based systems — consistently did well in Categories 1, 2, and 3 but took longer to program the parts in Category 4 than four of the APT based systems. As pointed out in Chapter III, page 25, organizations who have three axis machines such as machining centers and mills as well as lathes will consider the requirements of all types of parts when making a language selection.

Another consideration is the relative performances of remote processor systems vs. stand-alone local computer systems. Table 19 contains the same data as does Table 18, except that the remote systems data are plotted with solid-shade bars, and the stand-alone computer systems data are plotted with cross hatched bars. Of those systems performing better than the median in each category,

4 of the 7 in Category 1	(simple straight line parts),
3 of the 6 in Category 2	(straight line and circular arcs parts),
4 of the 7 in Category 3	(grooved or threaded parts), but only
1 of the 5 in Category 4	(tabulated or mathematical curves)

are systems with stand-alone computers. It should be noted that Cincinnati, MDSI, MS&S, and University Computing offer both remote and stand-alone versions of their systems. The designation used in Table 20 for these companies is the type of system used in the benchmark tests.

Note that from midday to late afternoon, both East Coast and West Coast organizations are actively programming, and remote processing systems experience longer queues at their central processing computer locations, and hence show slower processing times.

C. Cost to Program

As noted in the previous chapter (page 141), the cost to process a program on a remote system is determined by many variables, including:

- The use of central computer time
- The length of connect time to the I/O units of the system
- Whether the telephone link was made by toll or local call, and from where, and at what time of day
- The speed of response requested — for example, Rush, Immediate, 3-hour, or 24-hour
- Plotting requested

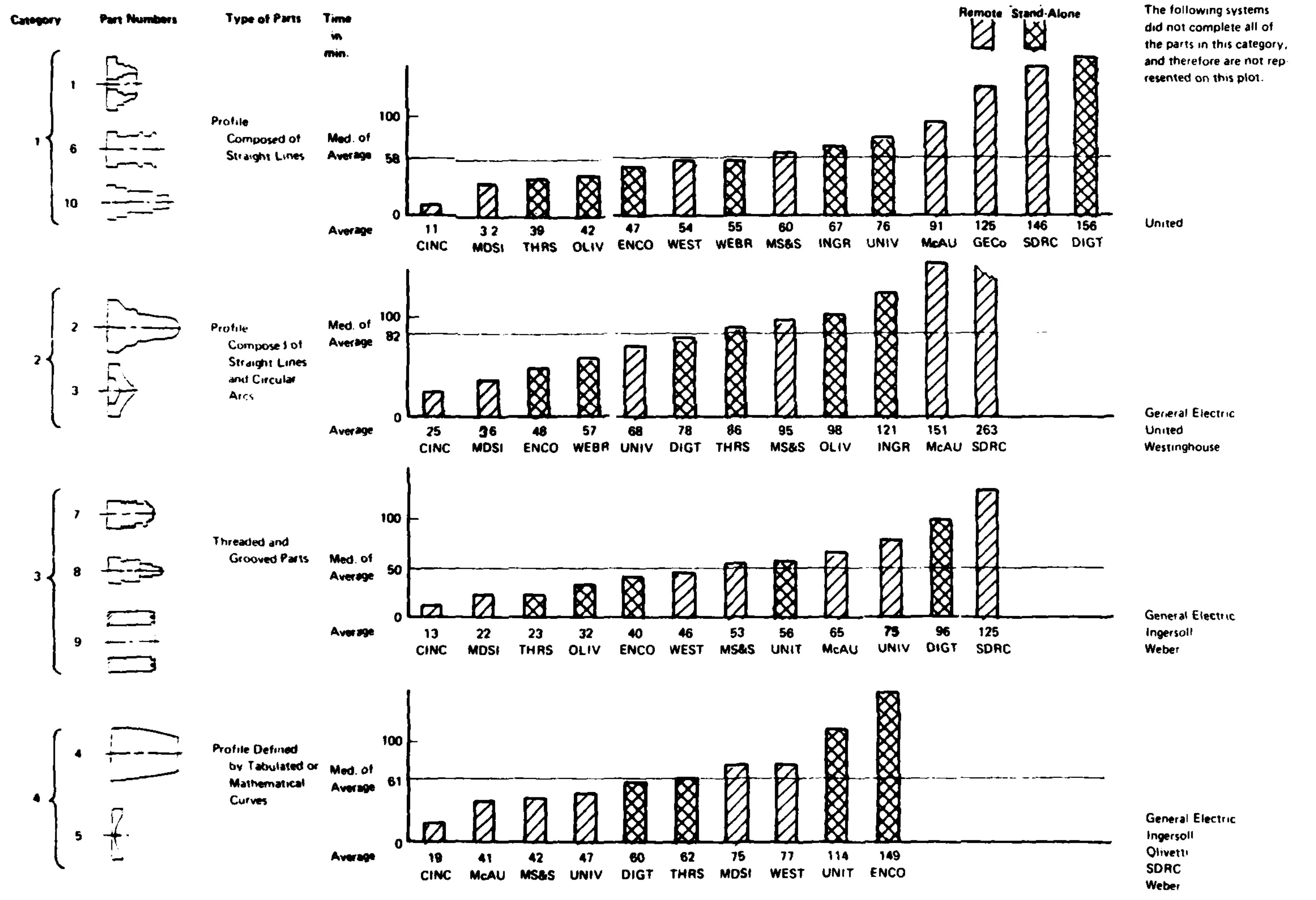


TABLE 19

COMPARATIVE PERFORMANCE, REMOTE VS. STAND-ALONE OPERATION

and possibly also the following in some cases:

- File storage charges
- Data transmission rate requested
- Card or tape punching requested
- Operator assistance required to mount tapes or disks
- Mail or express service to return tapes, plots, or printouts

In addition, a user may receive quantity use discounts and/or monthly minimum usage charges, which would be allocated pro rata to each program processed.

It was not possible to determine how large the cost accrued would have been for each part program during the benchmark tests, for those systems which ran programs on a remote system. Therefore, each proponent was asked to suggest a user who could run the program. The contractor asked this user to assist. Five proven source programs (Parts #1, #2, #6, #7, and #8) which the proponent had written, processed and verified were handed to the user. He was asked to submit them and report the charges from the system for the processing and the tape punching. All participants were asked to specify identical conditions — highest priority available, no plot. Their cooperation was gratifying.

The results are presented in Table 20.

While all cost differentials cannot be explained, the contractor feels that there may be reasonable explanations in some cases, or at least contributing factors that can be identified. The closeness of Cincinnati and Westinghouse may be explained by the fact that both companies use the same computer system, Westinghouse supplying the service for both. The fact that their costs are relatively high may be attributed to their highly automatic systems, especially Cincinnati's. The Cincinnati concept, in which a highly condensed input generates a rather lengthy APT source program that must in turn be processed, while highly efficient from the programmer's standpoint (see Table 17), is relatively costly from the data processing standpoint. On the other hand, the McAuto program, which uses a goodly number of basic APT statements, requires more part programming time but less data processing time.

It must be remembered that all programs reported in Table 20 were requested at prime time and at the highest priority. This would mean almost instant turn-around in all cases. As pointed out in earlier portions of this report (see Chapter VII) most remote processing systems offer a scale of discounts when longer turn-around times are acceptable. These discounts may range to as much as 75% of the charges for the host computer's processing time. Also it should be noted that quantity discounts may prevail for large quantity users.

No two systems have the same rate structure, let alone the same rate levels. It must be remembered that the entries in Table 20 represent single samples for each entry; had we chosen other users to run the tests, the exact charges would probably have been slightly different. Nevertheless, they are indicative of the relative processing costs of the systems.

TABLE 20

COST TO RUN PROGRAMS ON REMOTE SYSTEMS

System	Part #1	Part #2	Part #6	Part #7	Part #8	Total
Cincinnati	$53.85	$25.33	$36.43	$30.10	$102.95	$244.92
General Electric	32.50	17.31	29.43	10.52	32.39	121.65
MSDI	49.18	31.44	28.80	19.02	36.88	165.32
MS&S	88.10	57.13	49.19	36.45	74.04	304.91
McAuto	10.24	9.76	4.40*	5.93	13.57	43.70
SDRC	13.05	27.79	61.42**	57.12	18.23	177.61
UCC	28.76	25.58	21.37	15.95	28.90	120.56
Westinghouse	50.51	37.63	43.44	35.40	69.22	236.20

*The user test site could not run this part program due to the fact that the capability had not yet been installed on his system. The figure shown is that of the proponent's run, and is being used in order to arrive at a total figure. It is believed to be comparable.

**The user test site could not run this program, for an undetermined reason. The figure shown is that of the proponent's run, and is being used in order to arrive at a total figure. It is believed to be comparable.

To these costs, one would have to add the cost of software license or ownership, and the hardware including terminals, plotters, printers, punches, etc., and also the time of the programming department personnel, to calculate the total cost of remote processing programs.

For comparison, a local processor would use the cost of his software license or ownership, and the hardware costs including the computer, terminal, plotter, printer, punches, memory devices, etc. The investment in capital equipment, and the annual fees can be converted to an hourly figure by well known accounting procedures. To this hourly cost for the local hardware and software, one would again add the time of the programming department personnel in order to calculate the total cost of locally processed programs.

Check lists of costs are given in Table 2, page 27.

Some systems' designs emphasize very fast programming; others emphasize fast processing capabilities. Neither characteristic can be considered in isolation, as noted above. However, the mathematics of the relationship is dependent on the user's unique local conditions, and each reader who intends to use this data to assist in selecting a lathe part programming system must perform this exercise himself.

For example, assume that the part programmer's salary is $20,000 per year (an average figure taken from a number of "Positions Open" offerings in recent trade journals), and that his direct time carries a 100% overhead. Then for a 2000 hour work year, programming time costs $20.00 per hour. If we multiply the total time to program the five parts (number 1, 2, 6, 7, and 8) taken from Table 17 by $20.00 we arrive at the programmer's cost. To this we can add the total processing cost for the same parts, taken from Table 20.

This exercise has been performed *as an illustration only* and the results plotted in Figure 31. Please remember that:

- This applies only to the eight systems offering remote time-share service. (Quite another kind of a calculation would be required for stand-alone systems.)
- That portion of the costs attributed to data processing involved proven and debugged tapes.

D. Conclusions

It is the purpose of this report to evaluate fifteen of the current systems used in programming NC lathe parts, *in order that a potential user may make the best choice* to meet his own special requirements. We re-emphasize here the points made several times in the body of the report:

- There is no one language or system which is superior to all others in all situations.
- The capability of a system cannot be reduced to a single "figure of merit."
- Selection of a system should seek to achieve the best fit between the capabilities of the system chosen and the foreseeable requirements of the user.
- Selection of a system should be made with regard to the requirements for all parts to be programmed, including parts to be made not only on lathes, but also on mills and machining centers.

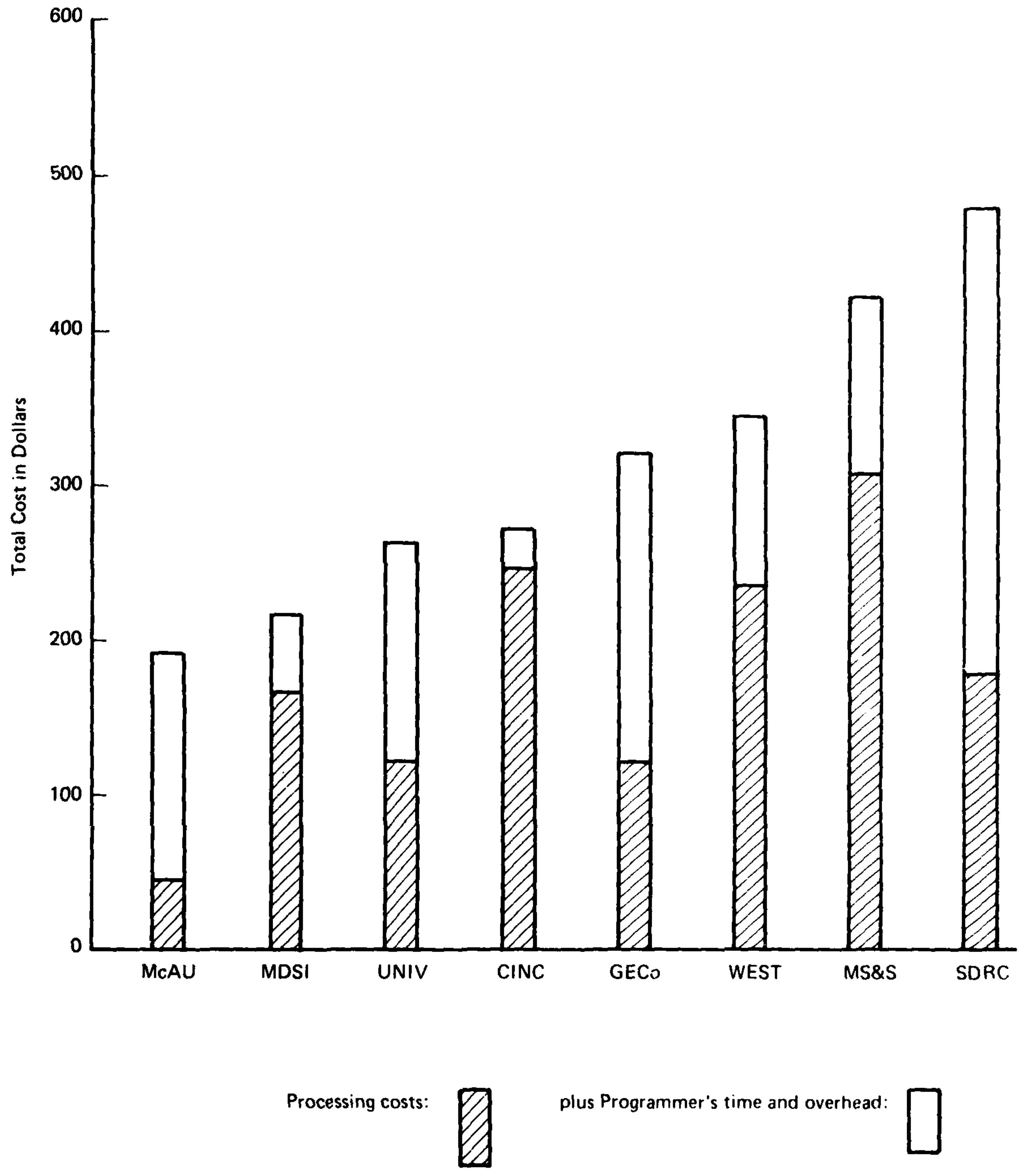


FIGURE 31 TIME SHARE SYSTEMS' COSTS FOR PARTS 1, 2, 6, 7, AND 8

The reader is strongly urged to re-read Chapter III, "How to Use This Report."

Chapter III recommends that the potential user determine, by an internal analysis: the product mix of his plant; the character of part designs amenable to NC programming; the future volume forecast for new part and revised part programming tasks; and a listing of the projected NC tools and the computers to be available. Recognizing that a single language in a plant is desirable, the potential user then selects a small number (two or three) systems which most closely fit his requirements. These systems should be investigated by contacts with the vendors, and by test demonstrations on the user's own parts.

Selection of these few systems is made by matching the internal requirements with the available systems. The characteristics of the fifteen systems have been spelled out in great detail in Chapters IV through VII. Table 2 gives a check list of the many cost elements to be considered, and the many intangible parameters entering into a final decision on a system and its vendor.

Insofar as possible, repetition of comments in this report has been eliminated; it is therefore important to read the entire report carefully.

While each potential user must make his own evaluation, there are some generalizations which may be made concerning the art of NC lathe programming.

Over the years, simpler systems have been developed to meet simpler geometry requirements, and in general lathe programming presents relatively simple requirements. If one considers all part programming, he finds there are relatively few very complex parts to program and many simple parts. If this array were visualized as a triangle, with complex parts at the upper tip, and the simplest parts along the base, and with the width of the triangle at any level representative of the number of parts, then most lathe part geometry would fall midway or lower in the spectrum.

In this study, systems designed specifically for lathe work performed very well. Some of them have intentionally foregone the programming of parts in the complex tip of the triangle, and have concentrated on the truncated lower portion, which represents the volume market. Frequently we found these systems operating on stand-alone mini-computers.

If you have a predominance of simple lathe parts, a stand-alone system might be considered. If you have complex part designs (those in the apex of the triangle), the APT based languages should be carefully considered. Both mini-computer and large main frame computer versions of APT based languages are available. The larger systems, which can handle the complex parts, have other desirable capabilities such as extended editing and cataloging powers.

If you have a small number of parts to program, then a remote time sharing system should be considered. On the other hand, as the number of parts to be considered for programming becomes larger, a break-even point is passed and a local processing system should be investigated. If a transition from a remote to a local system is being made, and the local system being considered uses the same language as the remote system, the trauma of conversion will be reduced. Part of the conversion consideration is economic, and part of it is based on intangible factors such as the convenience of trouble shooting service and ready software offered by the time sharing systems. By initially selecting one of the four systems that offer both remote and local processing systems, the user preserves his option to change from one to the other with minimum disturbance.

Selection of a part programming system is a long range commitment, and should not be undertaken lightly or too quickly. By careful use of this report, the selection process may be aided. The ultimate choice will stand partly on objective, and partly on subjective factors. And the ultimate choice **MUST** be made by the potential user.