UNITED ARAB REPUBLIC

THE INSTITUTE OF NATIONAL PLANNING

التخطيط القومي

Memo. No. 884

NUMERICAL CONTROL SERIES

PROGRAMMING FOR NC

By

ALWALID ELSHAFEI

OPERATIONS RESEARCH GROUP

April, 1969

CAIRO
3, MOHAMED MAZHAR - ZAMALEK

WHAT IS MEANT BY PROGRAMMING?

Programming is the process of writing the group of instructions telling the numerically controlled m/c where to move the cutter in order to form a part. This involves the following:

1) Defining the part to be cut
2) Describing the sequence of m/c'g operations needed to obtain the final shape
3) Defining the location of the tool relative to the w/p during each of the m/c'g operations
4) Describing the sequence of events which occur as the location of the tool changes.

From this definition, we might expect programming not to be difficult since it seems merely to involve defining the surface of the part to be cut, the sequence of cutting operations and necessary tools. Actually, programming is not this simple.

The complexities of programming derive from the necessity of defining precisely where the tool is to be at any given instant and the great number of mathematical calculations required to determine these points. The degree of complexity spans from the simplest algebra to the most rigorous solid and analytical geometry. The time required for their manual execution is commensurate with the degree of effort required. And this can be both speeded and simplified with computer assistance.

Another difficulty then occurs in combining this information about the tool movements with machining instructions in a format which a particular combination of machine tool and control system will understand. Computer programming will automatically take care of this.

A third complexity arises from dimensional considerations. Since the cutting tool has a radius, the path of the tool centerline (not the part surface) must be defined for the m/c. This added wrinkle may make it difficult to program even relatively simple two dimensional shapes. In most cases the tool-offset calculations require only trigonometry,but the computations are so tedious and time-consuming that the required manpower is costly and errors are frequent. The alternative is to let a computer make the tool-offset calculations.

MANUAL PROGRAMMING:

Manual programming is widely used in industry,and the method contains the fundamental elements upon which the computer programming methods are based. A general knowledge of manual programming also helps in understanding what is required in computer programming.

BASIC REQUIREMENTS IN PROGRAMMERS:

1- Machining Knowledge:

Satisfactory knowledge of different m/c'g operations, jigs and fixtures and chip cutting is necessary.

2- A Mind With Logical Thought Processes

3- Good Visualization:

- Ability to visualize the kind of fixture to hold the part
- Ability to sculpture a three-dimensional part out of a two-dimensional drawing.

4- Attitude & Maturity:

A programmer should be eager to learn new things. And he must be mature enough not to get discouraged when his first program is not a success.

5- Good Knowledge of Trigonometry and Arithmetic:

Despite assistance he may get from a computer,a programmer must have a satisfactory knowledge of trigonometry and arithmetic.

6- Background In Computer Programming:

At least,the programmer must be able to describe in machine language all the steps needed to produce the m/c'd component.

The programmer must have several types of information to program a part manually. In addition to complete information on the part geometry,he must be familiar with the machine tool and control unit to be used. The tape format,for example,must be tailored to the specific requirements of the control unit.* Format encompasses the following considerations:

Word Address or Tab Sequential: The word address format uses an alphabetic character to identify exactly what each numerical word refers to. This format allows the words to be placed in any order within a block. The tab sequential system,on the other hand, relies on the sequence of the word within the block rather than an alphabetic character to identify the quantity specified.

Maximum and Minimum Number of Digits: The number of numerical digits used to represent a given piece of data is usually specified explicitly for each type of machine control. Preparatory functions (called G-address) and miscellaneous functions (M-address) are usually expressed by two digits. Dimensional data may be expressed by five, six,or seven digits,preceded by a minus sign when necessary to denote negative quantities.

Maximum Block Length: The number of characters permitted in a block of data is sometimes restricted. This is primarily because of limitations of tape-preparation equipment rather than machine control restrictions.

Function Coding: Miscellaneous function coding and corresponding specific functions (M-address) are usually of an on-off nature for use on the machine tool itself. Examples are coolannt on or off, flood or mist,and spindle on or off,clockwise or counter clockwise. Preparatory functions (G-address) are primarily used internal to the machine-control unit to direct a machine tool through a given machining operation or for manipulating an axis. Examples are deceleration or acceleration and fixed cyclic actions of the spindle or head.

---

* Rfer to Apendix A & Apendix B.

Spindle Speed: Spindle speeds are usually programmed with an S-address prefix and a number of digits to indicate the desired speed.

Feed Rate: Feed rate codes are prefixed by the letter F and, like spindle speed,magnitude is specified by a variety of codes.

In addition to tape format,other programming parameters must be considered. These include the type of dimensioning system to be used and other factors dictated by limitations of the control unit.

Absolute or Incremental System: The dimensional information must have a format acceptable to the control unit. This data will be in either absolute or incremental form. Absolute measurements are made from a specific zero reference. Incremental dimensions are measured from the preceding point in a sequence of points.

Linear or Circular Interpolation: When a cutter is programmed around an arc or circle, the complexity of the programmer's job is determined by whether the control unit has circular or linear interpolation. With linear interpolation an arc is generated by a series of straight lines. The "smoothness" of the arc is determined by the number of straight-line cuts programmed.* Each cut requires a block of tape and a separate calculation by the manual programmer. An arc of fine resolution would obviously take a long time to program with only linear interpolation. If the controller can perform circular interpolation, the manual programmer need specify only the start and the end of the arc-all within one block of tape. The controller, through a form of on-line data processing, generates sufficient information to produce the arc in a continuous curve. This interpolation suggests the kind of assistance offered by a computer. (In this case the controller can be considered a form of computer.)

---

* Refer To Fig. 10.

Pluse weight: All dimensional words must be exactly divisible by the command pulse weight (resolution) of the system. Resolution is defined as the smallest increment of machine slide motion caused by a single command pulse. An error accumulation results if dimensional words are not divisible by the system pulse weight.

Acceleration and Deceleration: Acceleration and deceleration of the tool must be taken into account to prevent overshoot and undershoot errors. For example, the tool must decelerate prior to reaching a corner, so that the corner can be machined within desired tolerances. The tool should decelerate only enough to produce the cut within tolerance; if it is slowed too much, machining time is unnecessarily increased. The manual part programmer usually relies on graphs or empirical equations to determine the coorect deceleration and acceleration.

Tape-Reader Limitations: Tape readers commonly supplied with modern numerical-control systems are high-speed photoelectric devices capable of reading at rates up to 500 characters per second. High reading rates are necessary to maintain continuous machine-tool motion; otherwise dwell marks may be generated on the part in contouring operations. In other words, the reading device of the controller must be capable of reading data blocks faster than the control system can process the data. Dwell marks result when a short machine-tool slide motion is programmed at a high feed rate for a single data block. This allows the machine tool to complete the move before the following data block has been read into the control system. To prevent possible dwell in a contouring cut, the programmer must make sure that the interpolation time of each data block is not shorter than the tape-reader read time for the succeeding block of data.

The best way to get an idea of the steps involved in programming is to follow the programming operations for an actual part, Fig. 1.* <u>The operations are classified in three major categories:</u>

Selecting the Cutting Tool: The shape of the cutting tool determines the position of the cutter path necessary to produce the part. Assume, for example, that the programmer has selected a ¾ in diam cylindrical cutting tool. The cutter path then must be offset from the part by the 3/8 in tool radius. Direction of cutter travel around the part also must be-decided by the part programmer. This decision is based on machining factors such as surface finish and cutter type.

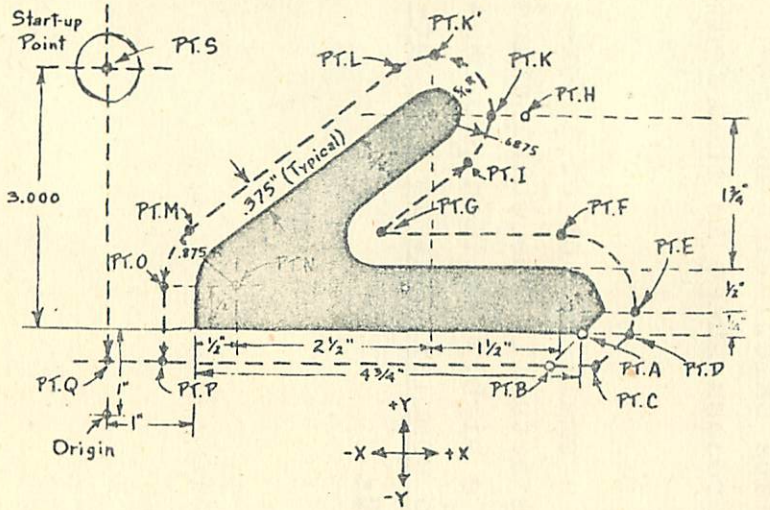
Calculating the Cutter Path: An origin is selected and designated as the zero reference point. A start-up point (S) is selected with coordinates related to the origin. Solid dots in Fig. 1 represent the termination of a tool movement, while small circles represent points which are required for use in the mathematical calculations. As shown in Fig. 1 Point S has coordinates of $x = 0.0000$ and $y = + 4.000$.

The first tool motion is from PTS (Point S) to PTQ, and requires an incremental move from PTS of -3.3750 in the y direction. This move results in coordinate values for Q of $x = 0.0000$ and $y = + 0.6250$.

---

* From an article by Robert Chandler; IITRI, ILL., USA.

manual programming



manual programming

N	G	X	Y	Z	I	J	K	F	S	T	M	REMARKS
		REWIND STOP CODE										
001	08	XO	Y-27450	Z-16250				F514	717	01	03	PT.S to
002	09		Y-6300					F2222			08	PT.Q
003		X59053	YO					F50				PT.Q to PT.C
004		X3750	Y3750					F565				PT.C to PT.D
005	17											X-Y PLANE
006	03	X3447	Y2500		I7803	J2500		F491				PT.D to PT.E
007	03	X-8750	Y8750		I7803			F491				PT.E to PT.F
008		X-20000	YO					F150				PT.F to
009		X-2332						F428				PT.G
010		X11565	Y8333					F214				PT.G to PT.I
011	03	X2642	Y5417		I4233	J5417		F669				PT.I to PT.K
012	03	X-6875	Y6875		I6875			F669				PT.K to PT.K'
013	03	X-4233	Y-1458			J6875		F669				PT.K' to PT.L
014		X-26154	Y-18522					F100				PT.L to PT.M
015	03	X-3363	Y-6895		I5387	J6895		F491				PT.M to PT.O
016		XO	Y-8750					F342				PT.O to PT.P
017	08	X-5625	Y30000	Z16250				F466				PT.P to
018	09	X625	Y3750					F3731			30	PT.S

FIG. 1

The tool path from PTQ to PTC requires the calculation of PTC. Using trigonometry, the programmer determines that an incremental move in x of + 5.9053 is required to move to PTC. In a like manner incremental moves in x and y are calculated to move the tool center to PT's D, E, F, G, I, K, K', L, M, O, and finally to P. Many intermediate calculations between points E and F, I and K, K and L, and M and O would also be required if the control unit is capable of only linear interpolation. These calculations are straightforward, but nevertheless are repetitious, time consuming, and susceptible to error.

Machine Parameters: The programmer now utilizes graphs and formulas to calculate acceleration and deceleration distances, feed rates, spindle speeds, and other such considerations. This information is combined with the coordinate calculation and entered on a process sheet. Data on the process sheet is then fed into the control unit on tape or cards.

Although there appears to be only a small amount of information on the process sheet, the data is actually the culmination of a considerable effort on the part of a highly skilled programmer. Manual programming is, in fact, so complex that it is practical only for point-to-point operations.

ANOTHER APPROACH FOR MANUAL PROGRAMMING:

Suppose we want to produce the pattern, shown below, of the equally spaced holes on a multi-spindle drilling machine*. How could the coordinates be manually specified and the drilling operations be described?

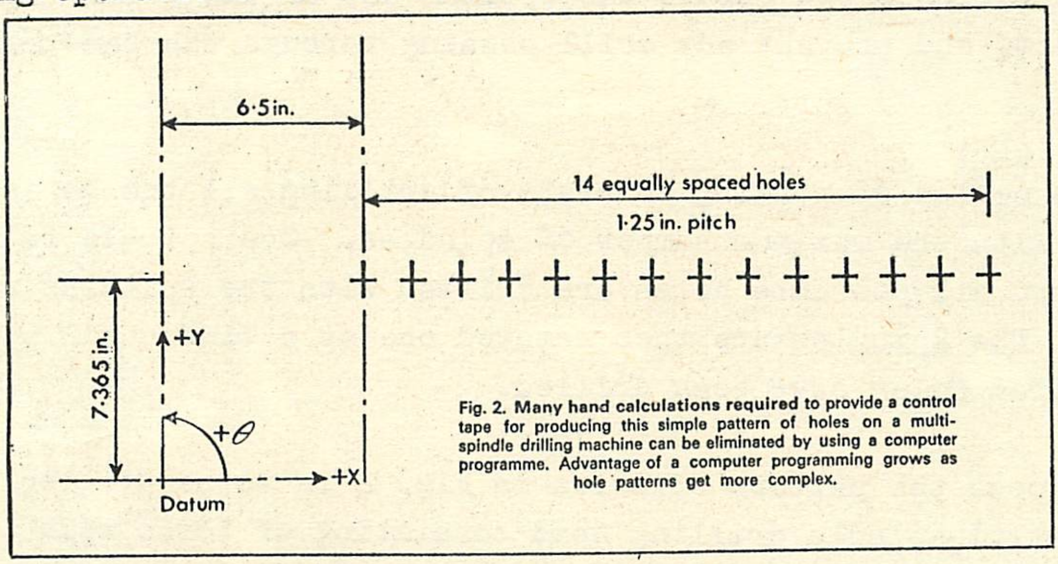


Fig. 2. Many hand calculations required to provide a control tape for producing this simple pattern of holes on a multi-spindle drilling machine can be eliminated by using a computer programme. Advantage of a computer programming grows as hole patterns get more complex.

1 - SPECIFYING COORDINATES:

The coordinates can be specified through the following table:

L	6.5	7.365	1.25	0	14
Type of pattern	X co-ord of hole A	Y-co-ord of hole A	Dist- between holes	Angle to X axis	No. of holes

Thus, in one line of six characters, 14 holes have been uniquely described. From the description a computer can calculate the relative X and Y coordinates and, once the machine type has been indicated, produce the machine tool control tape in any particular code that may be required. Translation from inch to metric units can also be accommodated during these calculations.

---

* From an article by G.F. Crudgington.

With a tape controlled multi-spindle drilling machine, the planning engineer or part programmer must not only calculate the coordinates for the required holes but also the coordinates which are to be used to control the relative spindle positions and their selective exclusion to ensure the optimum use of the multi-spindle arrangement, and prevent any drill passing through the same hole twice.

The method of working consists of drilling all the holes possible with the maximum number of spindles. One spindle is then removed and all possible holes are drilled with the spindles remaining. The spindles are then removed one at a time until all the required holes have been drilled.

Suppose the pattern of holes in Fig. 2 is to be drilled using a multi-spindle drilling head consisting of three spindles arranged in a straight line parallel to the row of holes shown.

The distance between spindles is assumed to be 3.75 in. and the reference point is at the centre spindle. The order of spindle removal is to be spindle 1 and then spindle 3.

Using manual programming the first requirement is to decide which of the holes can be drilled using three spindles, then which can be drilled using two and finally one spindle. Working from left to right, the best drilling pattern is as follows:

1, 4 & 7 2, 5 & 8 3, 6 & 9	3 spindles
10, & 13 11, & 14	2 spindles
12	1 spindle

The X and Y coordinates of the reference points for the centre sindle for each of these operations must then be calculated. These are as follows:

X		Y
3.75)	3 spindles	7.365
5.0)		7.365
6.25)		7.365
15.0)	2 spindles	7.365
16.25)		7.365
13.75)	1 spindle	7.365

These coordinates would then have to be re-arranged into a form suitable for the preparation of the punched tape and, at the same time, the necessary auxiliary and word address functions to control the spindle of the machine inserted. This could be as follows:

R

X 037500	X 162500 B
X 050000 B	X 137500 B
X 062500 B	Y 073650 B
X 150000 B	

Where R = alignment

B = move to next position

Z = stop at datum

This data is then punched on to the tape using the correct code.

This example illustrates that, even for a simple line, the process is fairly complex. As such problems become larger the engineer's job becomes both difficult and laborious, the following two examples will lead to better undertanding of the arising difficultes.

DIFFICULTY OF MANUAL PROGRAMMING

EXAMPLE 1:

Suppose we have a part such as a printed circuit board, a micromodule board, or a rocket nozzle in which there are many holes, perhaps all of the same size and equally spaced (Fig. 3).

NOT DIFFICULT, you say? PERHAPS NOT, but what it lacks in difficulty, it makes up in tedious repetition. Our manual programmer may not find it a very challenging part. So much so that after programming the first few holes, his mind is likely to wander far enough that the first part will be scrapped unless the programme is checked.

COULD THE COMPUTER HELP? BUT, OF COURSE! A few simple statements defining the pattern, the number of holes in each row, the number of rows and the computer takes over. In less time than it takes to write the name of the part on our work sheet, the answers are calculated and arranged neatly, ready to use.

It is the computer's task to bring order from confusion, to bridge the communications gap between man and machine by accepting man's language and converting it into the language of the machine, and at the same time relieving man from the multitudinous mathematical manipulations. Let us assume the preceding pattern of holes is located in the plate shown in Fig. 4. and take it to be

EXAMPLE 2:

Our programmer is not only confronted with the relatively simple (but tedious) task of resolving each hole location into its axis coordinates trigonometrically (which could have been made more difficult if the angle were 57deg., for example); but the drawing also states that there are 12 equally spaced holes in an elliptical pattern.

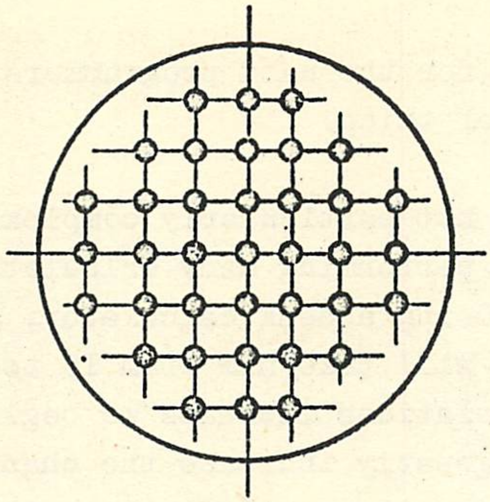


FIG. 3

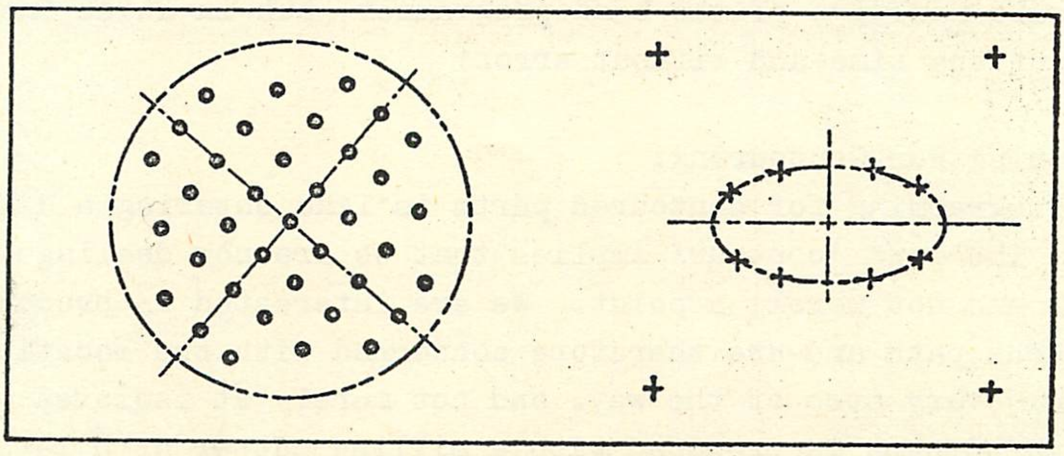


FIG. 4

DIFFICULT? YES, for the hand programmers, but the computer thrives on this sort of thing.

This example is not particularly complex, but our programmer will find himself performing many calculations to determine point coordinates. Using a desk calculator, slide rule, or merely pad and pencil, these will take him from 15 to 30 sec. each, and as the number of calculations increase we begin to add additional programming time and greatly increase the chance for human error. The computer can perform these calculations in one millionth of the time and continuously guarantee error-free work. It does not get tired, or confused, or distracted, or careless. It does what it is told-every time-perfectly.

Look at the drawing once more. The computer is not concerned with the angle at which the matrix of holes is orientated. It will first calculate the coordinates of the holes in the reference shown, just as our hand programmer would do, and would then re-calculate their positions using the necessary trigonometry to put them in the proper reference for the machine's axes. The computer performs the same basic function of the hand programmer, but in about one millionth of the time-and without error!

Programming For Contouring:

Programming for contoured parts is like entering a different world. The word 'contour' implies that we are now dealing with a surface and not merely a point. We are interested in producing a continuous path and are therefore concerned with the location of the tool every step of the way. and not merely at isolated points. A contoured part is produced with a milling cutter or a lathe tool. It is not enough to programme the surface of the work as described on the engineering drawing. No, indeed. The cutters have radii, and therefore, their centres must be offset from the work surface.

It is not the work surface that must be programmed, but the cutter centre path that must be calculated and transformed into machine instructions. This will be the path the centre of the tool must follow. Programming it may or may not be a simple task. It certainly is going to require a considerable knowledge of mathematics, and countless numbers of individual calculations.

If ours is a simple part described in terms of straight lines and circles, we could probably programme it by hand. After all, the offset of a circle is simply another circle, and control builders do supply control systems with circular interpolation. But, what if the surface is neither straight nor circular? Suppose it is a developed surface, not bounded by arcs of circles, what then? Or suppose it is defined by circular paths, but our control system is capable of linear interpolation only.

HOW CAN WE SOLVE THIS?

In this latter, much simpler case, the arc is segmented into chords which do not depart from the true circle by more than a specified tolerance (Fig. 5).

Having calculated the coordinates of these individual segments (no mean task in itself), we must recalculate their equivalent offset. If the surface is not circular, a similar procedure must be followed, but the mathematics becomes considerably more complex.

So far, we have assumed only plane surfaces. Contoured parts are often not plane surfaces. Three-dimensional surfaces will bring gigantic problems of the hand programmer. For this sort of work we are probably using expensive 5-axis machines. It would be

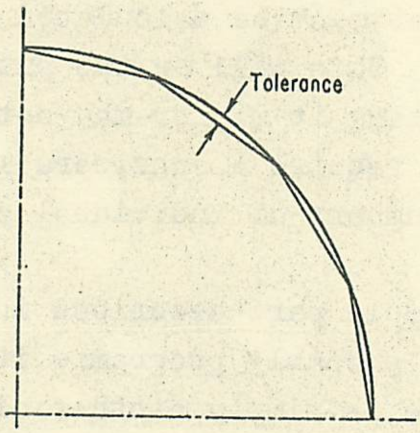


Fig.5. Tolerances have to be calculated when contouring with linear interpolation.

embarrassing to us if the effectiveness of these tremendously capable machines were solely limited by the inability of the hand programmers to write programmes for them.

A CONTOURING EXAMPLE

For purposės of example, let us consider a valve seat (Fig.6) which does not involve any circles except for a fillet or two and which is to be produced on a lathe. It's a very simple part, far less complex than some of the more exotic three-dimensional examples.

Let's assume the bottom surface is to be the reference plane. The part would be machined in two setups, one to establish the reference surface, and the second to complete the part. We would

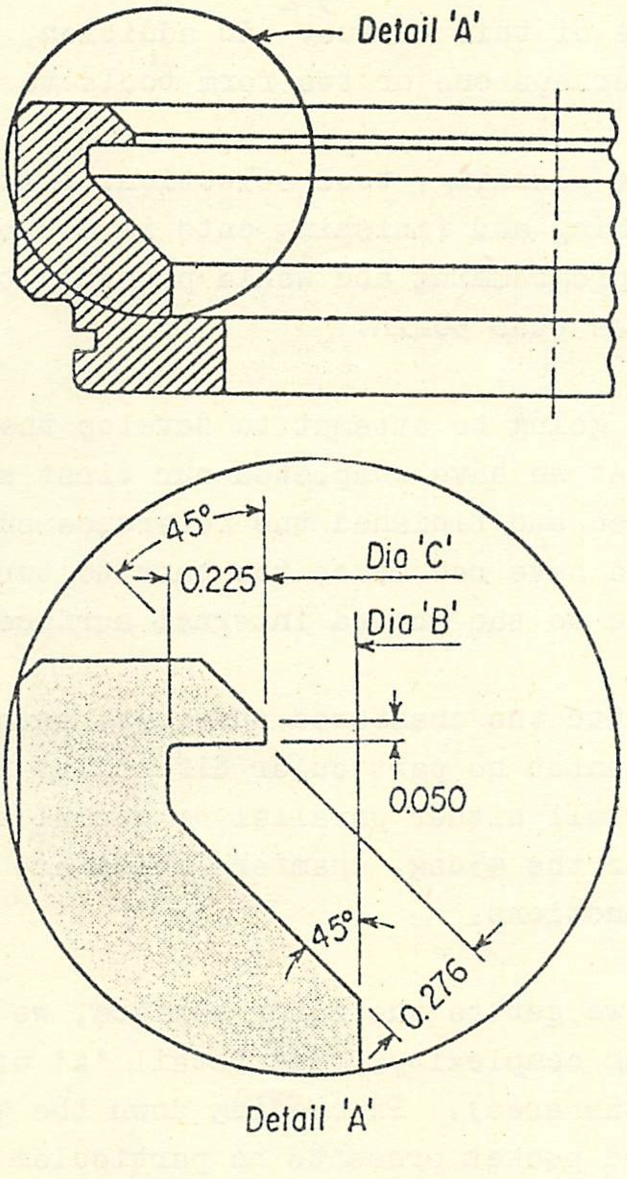


FIG. 6

probably choose tooling with a $\frac{1}{32}$ in. radius tip, as the print specifies fillets of this radius. In addition, we would need a boring bar and perhaps one or two form tools to do the grooves.

The methods planning, tool selection, fixturing, and determination of roughing and finishing cuts is a separate operation from the actual programming and would probably take the experienced planner less than 30min.

We are not going to attempt to develop the complete programme. Let us assume that we have completed our first setup, roughed the inside bore, faced and finished the reference surface, the groove and shoulder, and have rechucked the part to turn the outside, face the top, and do the angled internal surfaces.

Starting with the chamfered shoulder, and proceeding up the outside, we encounter no particular difficulty in programming as the surfaces are all either parallel or normal to the axis of rotation except for the 45deg. chamfers which are handled with simple trigonometric functions.

But, when we get to the inner surface, we face a problem of more than passing complexity, (See Detail 'A' of Fig.6 for an enlarged view of the area). Proceeding down the 45deg. slope and into the filleted pocket presents no particular problem, but getting back out on the second 45deg. slope does.

Next, you will notice that the second $\frac{1}{32}$in. fillet is tangent to the inner wall, which is parallel to the axis of rotation of the part, and to the 45deg. slope, but its centre is not shown. Neither is the point of tangency between the fillet and the slope given. To locate these points so that the cutter path can be programmed will require some rather extensive construction.

DIFFICULT? Not particularly; but detailed and tedious. I estimate it would take approximately 15 to 30hr. to programme this part by hand not including the time it would require to punch the tape. Let's have a look at how the computer can help.

COMPUTER PROGRAMMING:

Before starting our talk about computer programming, we would better bear in mind the following remarks:

1 - The computer cannot think, but it can do whatever we tell it to, and it is programmed by the smartest brains in the business.

2 - The computer can do no more than what a man can do; but it is infinitely faster and does not get tired.

3 - The computer never makes a mistake unless he is programmed incoorectly. Then it will stop and tell us about it. More than likely, he will tell us exactly where the mistake is.

Instructing the computer

The computer cannot be given its instructions in English. English has far too many words in its vocabulary. Some of the words are too big, and worst of all, some words have more than one meaning. So, in NC languages we limit the vocabulary to a few hundred useful words or so, shorten all words to no more than six characters, and give each word only one meaning. The result is a Pidgin-English-like language that both we and the computer can understand.

We can describe our part in the NC language and in seconds the computer has done all those mathmatical calculations and prepared, by means of the postprocessor a tape for our machine tool coded in the format that the tool will accept and understand.

The computer does more than that, too. For example, if you have a long departure to make, it knows how much your machine can take in one step, and breaks it down into shorter segments if need be. Same thing about spindle speeds; if you call for a spindle speed that the machine can't produce the computer finds one that will work. So it is with feed-rate, too. The computer won't let you go too fast if the machine can't go as fast as you would like to. These are only a few of the things that the computer knows about your machine and control system through its postprocessor.

NEED FOR COMPUTER PROGRAMMING

Namerically controlled m/c tools are highly efficient, versatile, automatic production devices. Without the aid of computers, however, their capability quickly exceeds the ability of the man to provide operating directions. Combined with computers, they become efficient extensions of man's creative abilities. The complexity of manual programming, beside the labourious procedure, has led to the development of computer assisted programming techniques. The value of computer-assisted programming is illustrated by the following example: A company producing a device containing numerous complex parts found that one year's output required billions of calculations to program the numerically controlled production machines. To do this manually would have required hundreds of mathematicians. But with a good computer programming system only a few programmers were needed.

The computer not only is valuable for producing complex parts, but also greatly speeds the preparation of data for relatively simple point-to-point applications. Users who have switched from manual to computer programming have found that they can program parts faster, cheaper, and more accurately with the computer.

The end result of a program for the NC production of a part must be the same whether the part is programmed manually of by computer, that is, a tape or cards must be produced in a particular format to operate a specific machine with a specified control system.

The ideal system would allow man to communicate with an NC machine in his own language to tell the machine precisely what it is to do. The practical compromise to this presently impractical ideal is the development of part-programming languages. These languages are comprised of a limited vocabulary of words (a form of pidgin English usually formed by phonetic contractions of multiple words) allowing no ambiguity. Through these languages, the programmer constructs a part program to tell the computer what he wants the machine tool to do. The computer, directed by the computer program and postprocessor, translates this information into the instructions necessary to operate the machine tool.

JUSTIFICATION OF COMPUTER PROGRAMMING:

It is most unfortunate that the numerically controlled m/c tools do not understand the spoken word, nor they look at a drawing and produce the part without any further instructions.

Since they do not have this capability, programmers must use a very rigid code and follow a format which the machine tool will accept.

Certain conclusions about manual programming are self-evident: it is slow, it is expensive and it is definitely limited in its capability. This is true for the small shop and for the large one.

For the small manufacturer with simple point-to-point machines, and possibly, simple parts on centre lathes, any other technique may not seem worth the effort. However, a good case can be made for investigating the use of computer as an aid to part programming. Even with simple machines, an early investment in this direction might be justified, and certainly a very handsome return can be foretold when additional or more complex equipment is added. Certainly, the capabilities of the programming group will be severely taxed if computer aid is not considered.

Much fear and confusion has been generated in the area of programming, particularly when computer-aided part programming is mentioned. Fear born of ignorance is unnecessary.

We should not refuse to consider a new technology simply because we do not fully comprehend its innermost workings. It has already proven its worth for others. Therefore, let us examine the requirements of programming for NC machines and determine what advantages, if any, the computer might offer for us.

Elements of Computer Programming:

Elements of computer programming were explained in the first article of this series. To complete illustration here, they will be again briefly surveyed (Refer to Fig 7).

The procedure goes as follows:

1. The programmer prepares a manuscript, called a part program, in the language accepted by the computer. This manuscript describes the shape of the part, the desired tool path, and machine tool functions required to produce the part.

2. The manuscript information is punched on cards according to a specified code. The punching is done through a conventional typewriter key-board, and the stack of cards is called the program deck. This deck is the input to a computer-software system called a processor.

3. The processor contains instructions which tell a general-purpose computer how to process the part program. The processor accepts the part-program instructions and prepares a computer for calculation of the coordinate points required to guide the tool. Additional instructions, such as feed rates, spindle speeds, and coolant flow, are accepted and passed on to a postprocessor.

4. The postprocessor provides the computer with instructions pertaining to the particular machine-tool control system under consideration. Input to the postprocessor includes information such as cutter path coordinates and programmer instructions for machine tool functions. The postprocessor removes a great deal of responsibility from the programmer because it performs numerous functions which, in manual programming, are the job of the programmer himself. For example, the postprocessor:

* Adheres to tape format.
* Monitors significant digit requirements for the various codes.
* Controls data block length.
* Handles programming sequence within a block of data.
* Provides data in absolute or incremental form depending on the requirements of the control system.
* Codes spindle speeds and feed rates.

* Handles linear or circular interpolation.
* Conforms to pulse weight requirements.
* Makes acceleration and deceleration compensations.
* Considers tape-reader limitations.

In short, the postprocessor allows a human to program a part without requiring him to have extensive knowledge of the machine tool and control system to be used.

5. The general-purpose computer processes the input data (part program) according to instructions contained in the processor (which generates geometry and computes the center line of the cutter path) and the postprocessor (which processes the output of a processor for a specific combination of machine tool and control system).

6. The output of the general-purpose computer contains all the commands, in the proper format, to produce a part on a given combination of machine tool and controller.1/

7. Either the general purpose computer istelf, or off-line conversion equipment is used to convert the output to the form required for the machine tool.2/ Perforated tape is the output form used in most control systems. The computer or conversion equipment also prints a manuscript for use by the programmer and the machine operator. These manuscripts provide a way to determine whether the tape is correct.

1/ See Appendix A

2/ See Appendix B

8. The tape is now ready for a numerically controlled device, which may be a drafting machine, a simple drill press, a two-dimensional contouring machine, or a complex multi-axis milling machine.

How the Computer Talks to the Tape

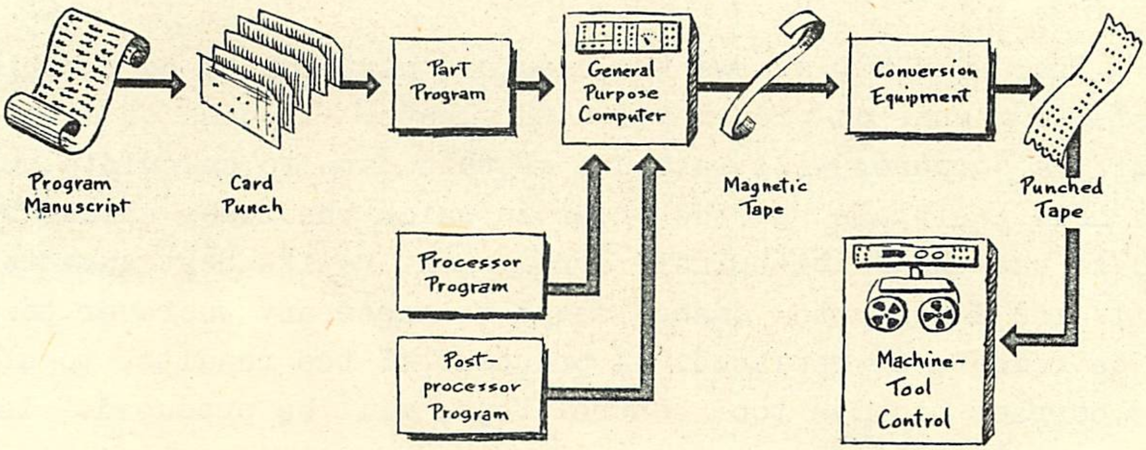


FIG.7

Example of Computer Programming:

Let us take the two previously handled examples dealt with using manual programming and let us see the difference on using computer programming. In order to allow for better demonstration, each example will be dealt with using a different programming system. The MULTIAID program (See Appendix C), which is an ICL programme, is used for the first example; while using APT (Automatic Programmed Tools), which is an IBM program, for the second.

DRILLING THE EQUI-SPACED HOLES PATTERN (See FIG. 2)

The ICL have produced the MULTIAID program in conjunction with G. & T. Weir Ltd., and J. Jameson Ltd. It is a development of the AID program.

The AID program depends upon describing simple groups of uniformly spaced holes. More complex patterns of holes can be considered as being a number of such groups, so that they can be dealt with in a series of short descriptions.

The MULTIAID allows the required patterns to be described in a simple manner on preprinted data sheets. In case of multi-spindle drill, a computer will make use of this data to calculate not only the hole positions but the order in which the holes should be drilled to use the multi-spindle arrangement to its best advantage. It will also insert tool change stops as necessary wherever tools are to be removed or replaced. A printout of the results, together with the correct machine tool control tape, will be produced. In the case of machines using tapes that cannot be produced by the computer or normal tape preparation equipment, such as the 4in. wide plastic variety used by older Airmec units, the output can be arranged in the exact form required by the punch operator.

The printout contains a list of the coordinates of all the reference points together with a description of the operation being carried out at each position. Furthermore, the printout contains instructions to the machine operator as to which tools are to be removed and when to carry out these instructions (Fig. 8). In addition, a graph plot of the positions to which the multi-spindle head moves will be produced, which will provide an approximate check of the accuracy of the data preparation.

POINT	TURRET	NO. OF DRILLS	X	Y	AUX CHANGE
ALL DRILLS READY FOR OPERATION NUMBER 1					
1	1	3	20.0000	17.7500	B
2	1	3	20.0000	17.5000	B
3	1	3	20.0000	17.2500	B
4	1	3	20.0000	17.0000	B
5	1	3	20.0000	14.7500	B
6	1	3	20.0000	14.5000	B
7	1	3	20.0000	14.2500	B
8	1	3	20.0000	14.0000	B
9	1	3	20.0000	11.7500	B
10	1	3	20.0000	11.5000	B
11	1	3	20.0000	11.2500	B
12	1	3	20.0000	11.0000	B
13	1	3	20.0000	8.7500	B
14	1	3	20.0000	8.5000	B
15	1	3	20.0000	8.2500	B
16	1	3	20.0000	8.0000	B
17	1	3	20.0000	5.7500	B
18	1	3	20.0000	5.5000	B
19	1	3	20.0000	5.2500	B
20	1	3	20.0000	5.0000	B
21	1	3	20.0000	2.7500	B
22	1	3	20.0000	2.5000	B
23	1	3	20.0000	2.2500	B
24	1	3	20.0000	2.0000	B
REMOVE NEXT SPINDLE					
25	1	2	20.0000	20.7500	B
26	1	2	20.0000	20.5000	B
27	1	2	20.0000	20.2500	B
28	1	2	20.0000	20.0000	B
REMOVE NEXT SPINDLE					
29	1	1	10.0000	1.0000	B
30	1	1	10.0000	3.0000	B
31	1	1	10.0000	5.0000	B
32	1	1	10.0000	7.0000	B
33	1	1	10.0000	9.0000	B
34	1	1	10.0000	11.0000	B
35	1	1	10.0000	13.0000	B
36	1	1	10.0000	15.0000	B
37	1	1	10.0000	17.0000	B
38	1	1	10.0000	19.0000	B
39	1	1	20.4481	21.6060	B
40	1	1	20.4481	21.3560	B
41	1	1	20.4481	21.1060	B
CHANGE ALL DRILLS READY FOR OPERATION NUMBER 2					
42	2	3	30.0000	2.0000	B
43	2	3	30.0000	2.5000	B
44	2	3	30.0000	5.0000	B
45	2	3	30.0000	5.5000	B
46	2	3	30.0000	8.0000	B
47	2	3	30.0000	8.5000	B
48	2	3	30.0000	11.0000	B
49	2	3	30.0000	11.5000	B
50	2	3	40.0000	14.0000	B
51	2	3	40.0000	11.0000	B
52	2	3	40.0000	8.0000	B
53	2	3	40.0000	5.0000	B
54	2	3	40.0000	2.0000	B
REMOVE NEXT SPINDLE					
55	2	2	30.0000	14.0000	B

FIG. 8

A further problem confronting operating numerically controlled machines is the estimation of production times for rate fixing, so that quotations can be made. With a multi-spindle drilling machine the problem is complicated by the number of machine movements involved. MULTIAID automatically deals with this problem and, in addition to the coordinates, the movement time, complete with a breakdown of the machine movements, is calculated and these are printed out after the reference point coordinates.

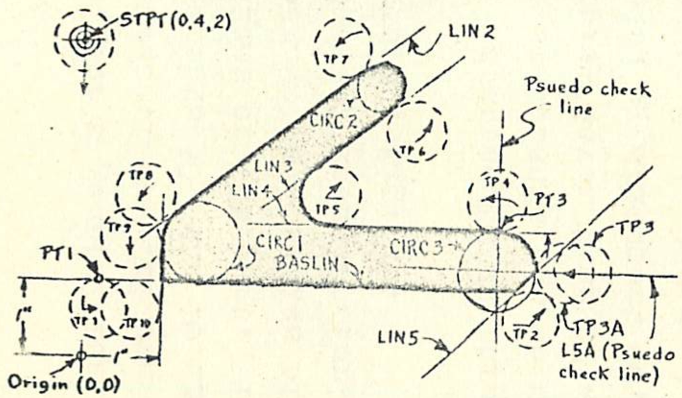
ANOTHER EXAMPLE: (FIG.9)

Consider the part (in Fig. 9) that was used to illustrate manual programming. Now suppose the same part is to be programmed by computer in the APT language. As shown in Fig. 9, the part can be described completely by a series of points, lines, and circles. For identification, the programmer arbitrarily assigns names to all elements (LIN1, LIN2, CIR1, etc.).

The programmer also indicates appropriate instructions for diretion of cutter travel, feed rate, and spindle speed. These parameters are dictated by considerations such as tooling to be used, material to be cut, and surface finish desired. The programmer also sets machine auxiliary functions such as coolant settings. This information is expressed in the computer language according to specified formats.

The complete program in APT is shown in Fig. 9. Statements are numbered in multiples of ten to permit addition of statements that may have been omitted in error. Note that no particular machine tool or control system is called for in this program; it is suitable for any machine and controller combination capable of the motions described.

computer programming



computer programming

```
PARTNO    REVISION 2, TEMPLATE 4614 N/J                        10
REMARK  = PART MACHINED IN 1/2 INCH ALUMINUM                   20
          INTOL/.0005                                          30
          OUTTOL/.0005                                         40
          CUTTER/.75,.125                                      50
STPT    = POINT/0,4,2                                          60
PT1     = POINT/.25,1                                          70
PT2     = POINT/5.750,1                                        80
BASLIN  = LINE/PT1,PT2                                         90
CIRC1   = CIRCLE/CENTER, (POINT/1.5,1.5), RADIUS,.5            100
LIN1    = LINE/(POINT/1,1),LEFT,TANTO,CIRC1                    110
CIRC2   = CIRCLE/4,3.5,.3125                                   120
LIN2    = LINE/LEFT,TANTO,CIRC1,LEFT,TANTO,CIRC2               130
CIRC3   = CIRCLE/5.5,1.25,.5                                   140
LIN4    = LINE/(POINT/4,1.75),LEFT,TANTO,CIRC3                 150
LIN3    = LINE/(POINT/INTOF,LIN4,(LINE/PARLEL,
          LIN2, XLARGE, .625)), RIGHT, TANTO, CIRC2            160
LIN5    = LINE/PT2,ATANGL,45                                   170
          SPINDL/ON,CLW                                        180
          FEDRAT/140                                           190
          COOLNT/ON                                            200
          FROM/STPT                                            210
          GODLTA/0,0,-1.625                                    220
PSURF   = PLANE/0,0,1,.375                                     230
          PSIS/PSURF                                           240
          INDIRP/PT1                                           250
          GO/PAST,BASLIN                                       260
          FEEDRAT/30                                           270
          TLRGT,GOLFT/BASLIN,PAST,LIN5                         280
          GOLFT/LIN5,ON,(L5A = LINE/0,1.25,9,1.25)             290
          TLON,GOLFT/L5A,TO,CIRC3                              300
          TLRGT,GORGT/CIRC3,ON,(LINE/5.50,0,5.50,9)            310
          GOFWD/LIN4,TO,LIN3                                   320
          GORGT/LIN3,TANTO,CIRC2                               330
          GOFWD/CIRC2,TANTO,LIN2                               340
          GOFWD/LIN2,TANTO,CIRC1                               350
          GOFWD/CIRC1                                          360
          GOFWD/LN1,PAST,BASLIN                                370
          COOLNT/OFF                                           380
          FEDRAT/140                                           390
          GOTO/STPT                                            400
          STOP                                                 410
          FINI                                                 420
```

FIG. 9

For a general familiarity with NC computer programming, several features of the program in Fig. 8 are worth noting. For example, statements 30 and 10 indicate the tolerance desired on the contoured portions of the part. The APT system calculates the tool path around a contour as a series of straight-line cuts. The number of straight-line cuts necessary for a given contour is dictated by the tolerance specified, Fig. 10.

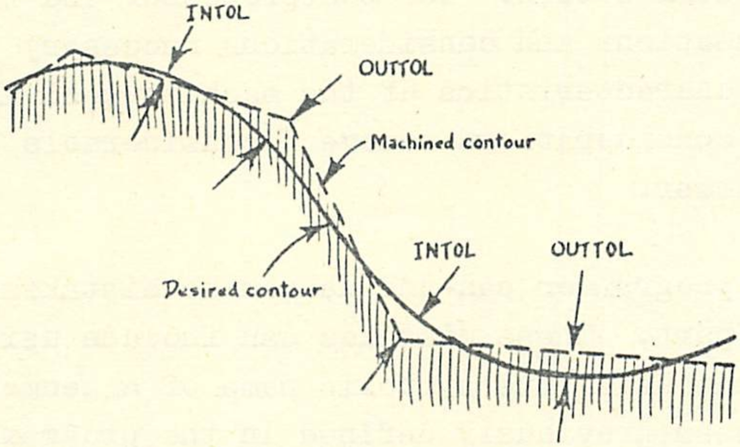
Statements 60 through 160 are definitions of geometry. Most such statements can be made in a number of ways. There are, for example, seven "legal" ways to define a point. The most basic, illustrated by statement 60, defines STPT (starting point) as xyz coordinates of 0, 4, and 2, A typical circle definition (there are 10 different ways to define a circle in APT) is shown in statement 120, where CIRC2 is defined in the xy plane with x and y coordinates of 4 and 3.5 and a radius of 0.3125. Statements 180 through 200 indicate spindle direction, spindle speed, feed rate, and coolant settings.

The remainder of the program consists mostly of motion statements that describe the desired motion of the tool around the part. The programmer uses commands such as GOFWD (go forward), GOLFT (go left), and GORGT (go right), as if he were riding on the cutting tool

The output of the program is a completely general solution, consisting of a numerical description of the path of the cutter centerline for the given cutter diameter and for the tolerance specified All computations are made by the computer.

Now only a single card must be added to the APT program to produce the control tape. This card tells the computer which particular post-processor program to use for the specific APT solution.

Contours from Straight Lines



All curves and contours in the APT program are generated by series of small straight-line cuts. The degree to which the straight cuts approximate a curve is established by specifying the magnitude of INTOL and OUTTOL in the computer program.

FIG. 10

The whole solution-processing and postprocessing - is performed during one computer access. The postprocessor and the computer make all compensations and considerations necessary to satisfy the particular characteristics of the machine tool and control system. These considerations remove a considerable burden from the part programmer.

The part programmer can-and does-make mistakes, particularly with a complex part. These mistakes can include using an illegal definition format using the symbolic name of a geometric element which has not been previously defined in the program, calling for the intersection of surfaces which do not intersect, and so forth. These mistakes and any other errors are found by the computer. The computing process usually stops at the error, and the computer makes a statement, called a diagnostic, that tells what kind of error has been made, and in which statement the error was first made.

It is meaningless to make quantitative comparisons of the time from blueprint to control tape between sophisticated computer programming and manual programming. The reason is that the only comparisons for which data are available are for parts which are trivial solutions for a sophisticated programming system.

PROGRAMMING DISCIPLINE:

Programming a part is the responsibility of one man, and he stays with it right through first piece inspection by the quality control department. While he does not maintain 'control' of the part in production, the programmer still continues his assocation with maintenance of that particular part programme through all subequent engineering changes, manufacturing modifications, and any problems that may arise even after initial debugging of the programme. The programmer gets occasional calls on parts he programmed four or five years ago.

The programmer writes his programme manuscripts in the NC parts programming language, because, after punching on the cards, they are processed on the computer.

The NC programme is contained in a disc file in the computer installation, together with the specific postprocessors for all of the varied NC machine tools in the plant. (The postprocessor adapts the part programme to the specific machine tool and control system combination to be used on the part.)

The part programmer, does not have to write the computer programme. He merely indicates on his part manuscript which computer programme and postprocessor he wants to use. This information is punched on a card, and the computer processing unit does the rest.

At the same time that the machining tape is being punched by the computer installation, a duplicate of the input manuscript is automatically produced along with the additional machining data calculated by the computer. This 'playback' document provides the part programmer with both a recheck against the original manuscript and a 'roadmap' as he supervises the block-by-block machining of the first workpiece.

The part programmer has a manual key punch unit available to him so that he can, at this checking stage, correct any erroneous cards in the part programme deck.

It is important to realise that a world of difference exists between an NC parts programmer and a computer programmer. The parts programmer programmes machine tools-in this case, with the aid of a computer. The computer programmer programmes computers-for example, the computer programmer writes the basic NC programme and postprocessors.

DRAFTING SPECIFICATIONS FOR NC PROGRAMMING:

The job of the programmer is greatly simplified if the designer provides information in a specific manner. To begin with, the designer should be familiar with the particular language to be used to reproduce the part. This knowledge will help him to anticipate the information that the programmer needs to produce a tape. But regardless of the language used or whether the tape is to be prepared manually or by computer, there are at least five basic conventions that smooth the flow of information from designer to machine tool:

(1) Dimension the part from a specific Cartesian coordinate system. More than one origin can be used for a part. Any number of subcoordinate systems may be used as long as these systems are located with respect to the major coordinate system.

(2) Place the origin, if possible, at a convenient location on the part itself (rather than outside the part) to simplify inspection.

(3) Specify tolerances precisely. The comment, "match in assembly," has become virtually unnecessary for parts to be fabricated through numerical control because of the inherent accuracy and repeatability of NC. But remember that even with NC, tolerances should not be tighter than required.

(4) Define contours either mathematically or by a table of coordinate values. Avoid such descriptions as "blend smoothly;"

(5) Identify dimensions likely to be changed during the course of engineering test and development programs. Likewise, indicate any dimensions likely to be changed in future redesigns of the part. Specify such dimensions and contours in a manner that facilitates change (as, for example, by specifying contours through tabular data). The programmer, in turn, structures his program to accommodate these possible changes.

FUTURE TRENDS IN COMPUTER PROGRAMMING

One of the more interesting aspects of numerical control includes the development trends in computer languages, and the possible effects of those trends on design. Perhaps the area of language development that will have the most profound effect on the designer is the work with systems such as the AED (Automated Engineering Design) Jr. linguistic processor at MIT. The AED, Jr. system essentially allows the user to devise his own computer language using his own vocabulary and performing only those particular functions which he desires. This is done by direct man-to-machine communication (on-line). The development of systems such as this are necessary to build the practical computer-aided design systems which will eventually make automatic manufacturing possible.

Closer to reality are computer languages devised for scope-to-tape programming. These systems allow the programmer to specify cutter paths merely by tracing (with a lightpen) the outlines of the part on a cathode-ray tube hooked on-line with a computer. Plans are also being made for convetsational versions of APT.

A number of organizations are making feasibility studies on scope-to-tape programming. Limited tests have already shown the great promise of this technique. For example, a simple test part (a two-dimensional template) comprised of straight lines, circular arcs, one parabolic arc, and one drilled hole was programmed using a scope-to-tape system. Total time from start to control tape was one day. By comparison, the same part was programmed in APT by an experienced programmer. Three computer runs were necessary to get through the APT system without programming errors. Total time from start to control tape was four days.

Tied closely to computer-assisted programming are the concepts of computer-aided design. It is only a matter of time before the two concepts are merged to form a system that combines predesign calculations, design of the part, and generation of a control tape for the NC production of the part-all in one short session with a computer. It is not unreasonable to expect that ultimately the designer, planner, and programmer will be combined in one person who will be the only human link between a design concept and the finished part.

APPENDIX A

The work involved in producing a control tape for a small number of holes in a simple workpiece probably does not justify the use of a computer. But the user still has to buy equipment to punch tapes and the choice of tape punching equipment is becoming difficult as the EIA and the ISO standards of tape coding fight for predominance, and as tape formats within each code are many and varied. Tape preparation is a major chore, however, when the component has a thousand holes, and manual efforts are time consuming, costly and error-ridden. Even working out the coordinates for a control tape to position 20 equally spaced holes lying in a circular pitch is a time consuming business.

The work is not finished even when the coordinates of the holes have been calculated, since the auxiliary machine functions have to have their control signals inserted where necessary, and a control tape has to be prepared and the work varies with each combination of machine and control system. The lines quoted below illustrate this problem; the top line and remaining two lines contain instructions to drill the same hole in the same component, but they refer to two different control system/machine combinations.

061252106375

N001 G81 X061252 Y106375 Z7625 F415 R8625 S675 T07 M03 M13.

However, since the compiling of these instructions is purely one of processing known data according to certain rules, a computer can be used to overcome all these problems, thus simplifying the planner's task. The preparation of taped control instructions for straight line and contour machining or for a machining centre benefits even more from the use of a computer.

APPENDIX B

Purchasers of tape-controlled machines are taking an especially careful look at the postprocessor available with any numerically controlled equipment they plan to purchase. The postprocessor, generally considered only an increasingly important factor in determining over-all and long term NC operating costs.

Before the development of the Fortran IV code language, only certain large IBM and Univac computers could handle such large-scale general-purpose NC programmes as APT and ADAPT used in producing control system punched tapes. Relatively few manufacturers had these large computers. Other companies had to lease time and were dependent either on those few companies running programmes as a side-line operation or on computer company service bureaux.

Many plants with a smaller or different make of computer solved the problem by writing a special programme keyed to their own unit. But a special programme involves routines that, though often extremely ingenious, are nevertheless expensive in the long run. These special programmes usually result in circuitous instead of straightforward procedures and are limited in scope.

APPENDIX C

Programmes now available include MULTIAID, the point-to-point programme for multi-spindle drilling machines. AID the point-to-point programme for single spindle and turret drilling machies, MILMAP the programme developed specifically for an automatic toolchanger of the Mil-waukee-Matic type, and PMT2 a mesh-of-points programme for resolving tool paths on curved surfaces. Post processors have been produced to enable these programmes to be used for producing tapes for a number of control system/ machine tool combinations and the range will gradually be extended.

Jobs tackled by MULTIAID so far vary from heat exchanger tube plates to a grading sieve. The programme is applicable at present to multi-spindle machines having the spindles arranged parallel or perpendicular to one of the machine's axes. Any number of spindles can be accommodated and the distance between spindles may either be variable or fixed. Post processors exist which enable tapes to be produced for Jameson, Kearney & Trecker and Archdale machines fitted with EMI B100, Ferranti Series 1, Autoset, and AEI Aximatic systems. Further developments of the programme will cater for other machines, and will enable it to deal with other spindle configurations such as triangular pitch arrangements.