GE Time-Sharing Service

Information Systems
Time-Sharing Service

NUMERICAL CONTROL PROGRAMMING WITH A TIME-SHARING COMPUTER

. . . a new way to go from engineering drawing to N/C control tape in hours, instead of days, at a cost anyone can afford.

E. L. McCleary
Manager — Marketing
Information Service Department
General Electric Company

Presented at
Fifth Annual Meeting and Technical Conference
Numerical Control Society, Inc.
Philadelphia, Pennsylvania

April 4, 1968

GENERAL ELECTRIC

NUMERICAL CONTROL PROGRAMMING WITH A TIME-SHARING COMPUTER

E. L. McCleary
Manager — Marketing
Information Service Department
General Electric Company

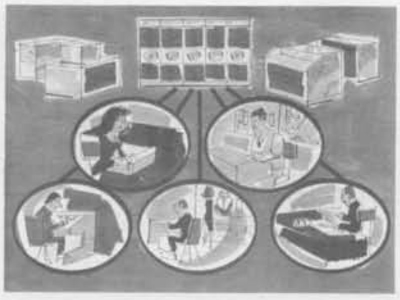
Good afternoon, gentlemen. When I say it's a pleasure to be here with you, you know it's not just a speaker's courtesy. I've known many of you for many years and this seems a little like old home week. But aside from all those friendships I'm having such fun renewing, I've got another reason to be so pleased to come before you today—what I've got to tell you about is, in my opinion, one of the most exciting things that's happened in the numerical control business since somebody first discovered you could do parts programming with the help of a computer. And its effect on *your* operations may well be even more revolutionary.

What I'm going to tell you about is a new capability of the General Electric Time-Sharing Service that will make it possible for your parts programmers to go from engineering drawing to ready-to-run tape in *hours*, not days . . . let them use their time for *productive work*, not for struggling with mind-boggling math problems . . . and enable *you* to get more and better work not only out of *them*, but out of your numerical control machines, thus improving your company's return on investment—and keeping your company's *customers* happier, too.

Now you all know what Time-Sharing is: one of our customers once called it "bringing the mountain to Mohammed," and that's a pretty good analogy. What Time-Sharing does is bring the power of a big computer within the physical (and financial) reach of everyone —as close as your telephone, and nearly as easy to use.

For a few dollars a month, a man can have a terminal right beside his desk that puts him instantly in direct contact with a million-dollar computer that may be many miles away. All it costs him is approximately $85 a month for the terminal, a few cents a second for the computer time he actually uses, ten dollars an hour for terminal time, and the phone bill—which is usually not a problem, because General Electric is now offering local service in *nearly every* major U.S. city.

Why so inexpensive? Because you're actually sharing the computer's time with many other simultaneous users—though neither you nor they are aware of each other — and as Dean Tribus of Dartmouth's School of Engineering pointed out recently, Time-Sharing is cost-sharing. (Slide 1)



1.

2.

Let's hear what a few of our more than 50,000 users think of General Electric's Time-Sharing Service.

J. E. Hart (Slide 2), manager of heavy machinery estimating for Cleveland Crane and Engineering, says:

"With this service, savings on production costs are fantastic, and there's no huge capital outlay required."

Warren K. Brown (Slide 3) is a

3.

structural design engineer for the Hall Ski Lift Company, in Watertown, N. Y. He says:

"It permits us to economically finalize design layouts in hours instead of weeks. Computations on the computer cost one-tenth of what they would if done manually."

Listen to James Edmiston (Slide 4), product engineer of the Star-

4.

Cutter Company, a metal-cutting and machine-making firm:

"It took four seconds to get an answer on a gear design problem which would have taken all day to obtain with a calculator. As a result, more problems are solved in far fewer man-hours."

Many of our customers consider the money they save a secondary benefit. Many of their companies already have big computers, but still they want to tie into our Time-Sharing service.

Why? Because big batch computers are fine for payroll accounting and other such massive, repetitive jobs, but for individual, one-time, problem-solving kinds of work, Time-Sharing offers a great many advantages.

Parts programming for numerical control machines is that kind

5.

of work. Let me show you what I mean.

To program anything that requires more math than he can do on his fingers, a parts programmer would like to use a computer. (Slide 5)

But if he does, he must make out (Slide 6) a coding sheet, get that punched on cards, check the cards for errors, get them to a computer center, and wait—sometimes several days—for his program to come back. (Slide 7)

And then when it finally does come back . . . well, how often is it right the first time? So then he has to go through the same thing all over again. (Slide 8)

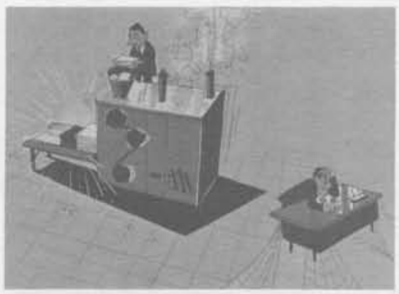
Is it any wonder that, despite advances like APT and ADAPT,

6.

many parts programmers still elect to do some of their less complex programs manually—time-consuming and error-prone though that method may be—rather than face the frustrations of using a computer?

But this is where Time-Sharing comes in.

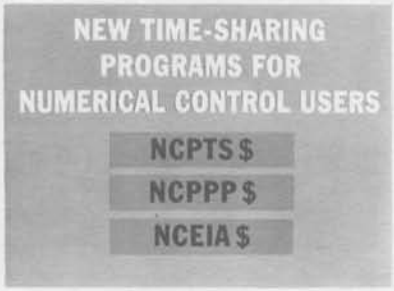
Now there's a way for him to gain all of the advantages of using a computer—the time it saves him by doing all the tedious math, the



7.

errors it saves him from making—without losing the things that appeal to him most about manual programming—the ability to keep control of a project from beginning to end, the chance to stay with it 'til it's done and not lose his train of thought by switching from job to job while waiting, the ability to see his mistakes and correct them on the spot.

8.



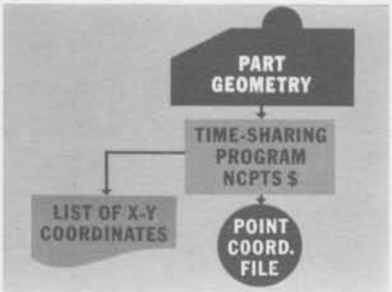
9.

Here's how it works.

We've added to our library of programs for Time-Sharing customers three new ones (Slide 9) called NCPTS$, NCPPP$, and NCEIA$. Used severally or in sequence, they are the answer to a parts programmer's prayer.

Here's what they do:

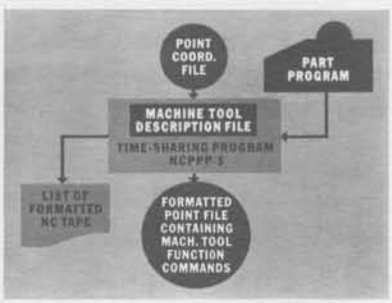
NCPTS$ takes the geometry—your parts programmer's description of the piece to be machined, in terms of points, lines, and circles—



10.

and sets up a point coordinate file. (Slide 10) It does math in seconds that it would've taken him hours to do by hand, and gives him a list of X-Y coordinates.

NCPPP$ (that stands for numerical control point-to-point processor) takes three inputs — that point coordinate file, plus a description of the particular machine tool and control combination your



11.

man is programming for, and a part program file describing the necessary machining sequence. (Slide 11) (The machine tool description file is usually prepared in advance and stored in the computer 'til needed — I'll talk more about it in a minute.) Anyway, NCPPP$ takes these three inputs and produces a list of instructions in the exact format of the control tape needed to machine the part he wants to machine on the equipment he wants to use.

Finally, after he's checked the print-out for errors, he can use NCEIA$ to take that data and punch a master control tape for him, in EIA code, right at his terminal. (Slide 12)

So—those are the new programs we've added to our Time-Sharing library. Using them, your parts programmer can go, as I said, from engineering drawing to finished part in a couple of hours or less, and do it without ever leaving his desk. He can stick with this one job until it's completely done. No distractions, no delays, no need to make work while waiting for cards to be punched, or work to come back from a batch computer somewhere.

What are the drawbacks, then? Is it hard to learn to do? Does he have to learn some whole new language in order to use the system? Not if he already speaks English. The symbols he uses are very similar to what he's already using, even if he's programming manually, and the words he needs are English-like.

In fact, he doesn't even have to be any better than a biblical system typist. (You all know what the biblical system is, don't you? Seek and ye shall find?)

(There isn't even the psychological problem of introducing a computer into your operations. The terminal just acts like a smart typewriter!)

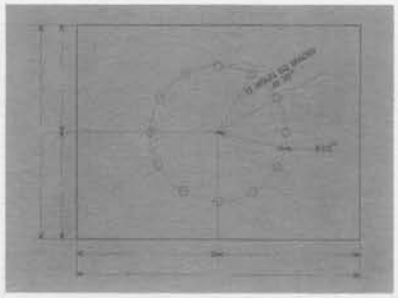
To show you how easy and familiar it all is, let me take you

12.

step-by-step through an actual program:

Let's do something simple. Let's drill a dozen holes in a bolt-hole circle. The drawing would look like this. (Slide 13)

First, your parts programmer would identify the geometry—the location of the center of the circle, the radius, and the angle between the points (that is, the spacing between the holes)–and pick a starting point and a stopping point.


13.

Now he's ready to use the computer. He turns to the terminal and calls it up, just like you'd call your Aunt Minnie on the phone. (Slide 14) (The computer even used to say "HELLO," but now we get right down to business.)

First, the computer asks him who he is. He types in the user number we've assigned to him, and the computer asks him what lan-

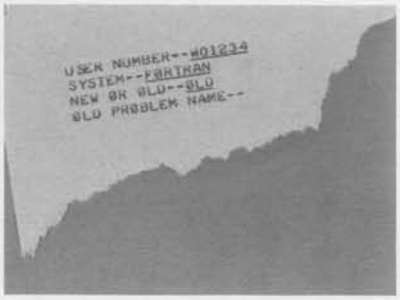

14.

15.

guage he wants to use. For these NC programs, he responds "FORTRAN." (Slide 15)

(Now that doesn't mean he has to go to school and learn how to program in FORTRAN. All he has to know for *his* purposes are a few simple symbols, most likely already familiar to him, as you'll see in a minute or two.)

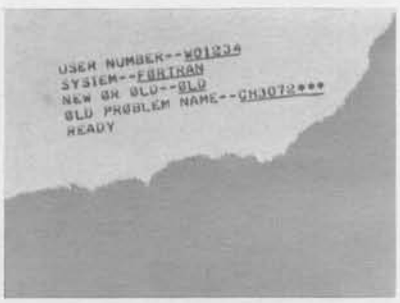
Next, the computer asks him if he wants to enter a new program, or work with one it already has on file. Since these NC programs are



16.

in the program library, he types in "OLD." (Slide 16)

Which old program do you want to do first, the computer asks, and now your parts programmer has to do a couple of routine mechanical things. He has to set up a place in the computer to store the data he's about to develop, and he has to tell the computer how to use that data.



17.

So, first he types in a file number—let's say CH3072, which simply means "let me have a file space with room for 3072 characters." That ought to be plenty for what we want to do. (Slide 17)

O.K., says the computer. Now what?

Now he has to give that particular file a name, so he can go back in and get what'll be stored there. Let's call it MYFI, my first file. (Slide 18)

When the computer acknowledges this, he tells it to save what he puts in that file, and it says it will.

So far, so good; right? Even high school kids could do it. Matter of fact, they *do*, as a routine exercise, in a growing number of schools.

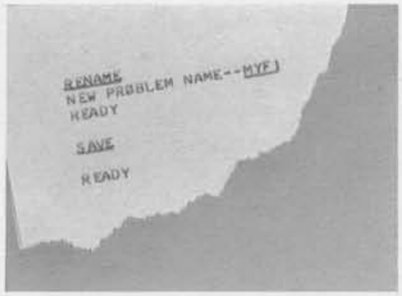
Now your man is ready to work out his problem and this will be a new program, in the computer's eyes. Let's call it MYJOB. (Slide 19)

Your parts programmer now writes a couple of instructions, identifying the file he just set up, and telling the computer to fill it with a point coordinate file he's about to develop, using the first of the three programs we'll walk through today—NCPTS$. Let's go, he says. (Slide 20)

O.K., says the computer. This is your program name, the time and the date. I'm going to use NCPTS$ to figure out what you want to know, and I have a little geometry program I'll use as part of that. What sort of a pattern do you want to make? (That's what "input code" means.) (Slide 21)

Here your programmer enters one of a number of symbols which describe the pattern of this first cut or series of holes.

For example, he might type PXY, which means define a point, given the X and Y coordinates. Or PDP — given the distance and angle from a known point, find the unknown point. (Slide 22) There



18.

are dozens of these little shorthand symbols which enable your programmer to find almost any point or path from whatever known values he starts out with — lines, angles, radii, whatever. All of these symbols are described in a USER'S GUIDE we give him.

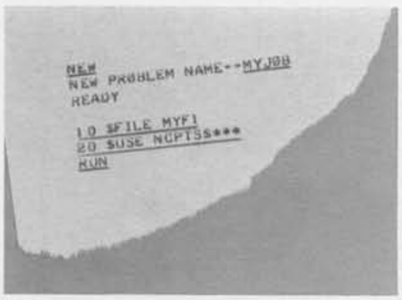
For our purposes today, he will use PAC—points around a circle—which means "given a known circle (center point and radius), a starting angle, a stopping angle, and the angle between the points,

19.

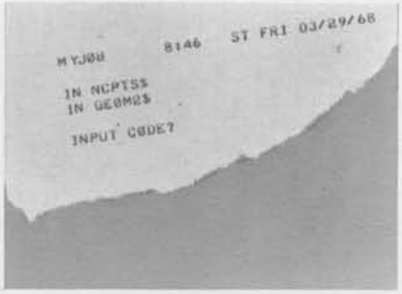
find successive points on the circumference of the circle." (Slide 23)

The computer then asks him to supply these values, which he does, and it then supplies him—in seconds—with the X and Y coordinates for each of the holes he wants to drill.

WPF, he types, which means write (enter) the point coordinates into the file I set up. How many



20.

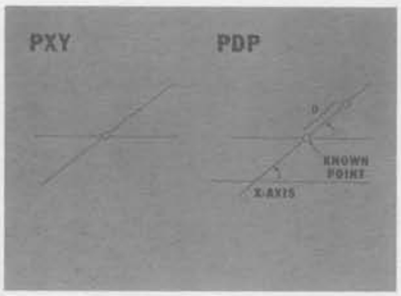


21.

are there, asks the computer? He tells it, and it says done—what's the next step?

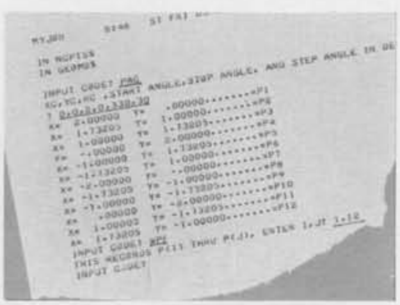
"End," he types, because that's all he wants to do today, but depending on the complexity of the piece, he could go on and on entering instructions and values to configure any pattern of operations that can be performed on a two-axis machine. (Slide 24)

Two things I should mention here—first, it actually takes longer for me to tell you about this than



22.

it does for a man to do it, and second, I've chosen an extremely simple example here to show you how the system works. You and I both know good manual parts programmers who could work out this particular problem almost as fast. However, using Time-Sharing, it would take the merest fraction of a second longer to work out the coordinates for 88 holes around a



23.

circle, or 362, and to do that manually would be another story.

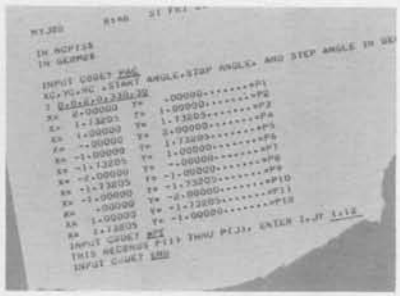
So we're not really attempting to show you the total capabilities of the system here today — merely how it works.

O.K. Now that he has the geometry done, your parts programmer is ready to go on to the next step—developing the actual machining instructions for your particular tool.

You'll remember that I said the Time-Sharing program he'll use to do that—NCPPP$—requires three inputs: that point coordinate file, a machine tool description file, and a part program. (Slide 25)

Let's talk about the machine tool description file first.

This is something your man does just one time, for each tool you own, and stores in the system to be called out as needed—like a

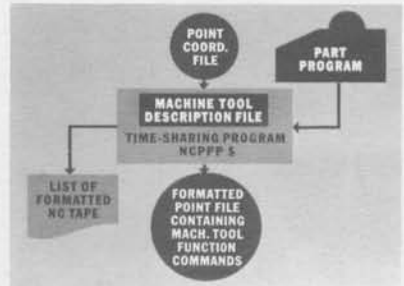


24.

post processor. (Slide 26) It describes that particular machine tool/controller combination, taking into account the various options available and specifying the proper format of a block of data for the control tape.

The information needed to do this is contained in the programming manual supplied with each tool by the manufacturer. It shouldn't take more than a couple of hours to do. Your programmer does it just once and then it's there as long as you have that tool.

To make it even easier, we've set up a machine tool description file for a typical tool, which in most cases will only have to be modified to fit his particular machine. And,



25.

a GE application specialist is available to assist your programmer with his first couple of files.

Obviously, this means that to add a new tool to the system, all you have to do is set up another file. It also means that you don't have to spend several thousand dollars on a post processor.

O.K., so much for the machine tool description file. Now let's move on to the parts program.

This is another new problem in the computer's eyes — that is,

26.

something not already stored in its memory — and we'll call it "my program." (Slide 27)

Your man now proceeds to describe the sequence of operations necessary to machine the part in question on his particular machine. For our purposes today, we've made a number of assumptions. We're programming for a 3-axis

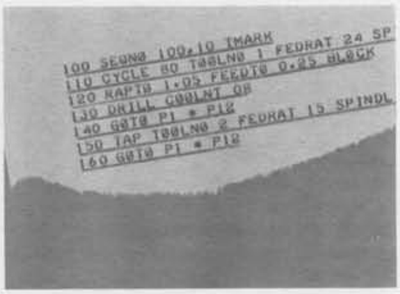
27.

28.

machine tool with a Mark Century positioning control (little plug there). It has a tool changer and several canned cycles.

The first step is mechanical: he tells it he'll start his sequence numbers at 100 and proceed at intervals of 10, and to put an identification mark on the tape. (Slide 28)

Now he tells it that the drill we want is in tool position number one, he tells it to use feed code



29.

number 24, spindle code number 32, and to rapid-traverse down to within .05 inches from the workpiece, which we're saying is an inch thick. Next, he tells it the hole should be three-quarters of an inch deep, and to put all that data in a block on the tape. Now, he says, we'll instruct the tool to drill and to turn on the coolant.

Finally, he tells the computer to

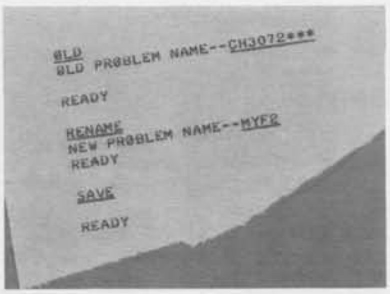
30.

tell the machine to repeat the sequence for each of the twelve holes for which he has coordinates.

He could quit at that point and have a complete operation, but let's say he wants to *tap* the holes, too. All he does is put in another instruction telling it so, and changing the tool and feed and spindle speeds. (Slide 29) He then tells it to repeat that operation for each hole.

He could've entered instructions to ream or to do any number of other operations just as easily, but let's say he's through now — so he says so and tells the computer to save all that. O.K., it says. Now what? (Slide 30)

And that's all there is to the parts program. Now he has all the inputs he needs for the Time-Sharing N/C point-to-point processor program.

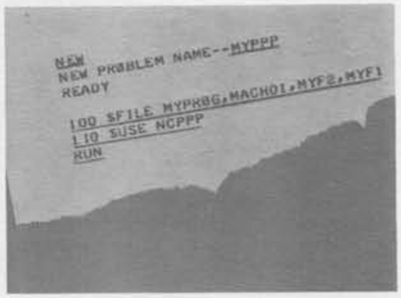


31.

Next he sets up another empty file to receive the output, just as he did before — let's call it "my second file" — tells the computer to save what goes in there, and he's ready to go. (Slide 31)

This is a new program, so he tells the computer so, and names it MYPPP—"my point-to-point processor." O.K., says the computer, and at this point your man simply identifies for the computer his parts program, his machine tool description file (we've called that "machine zero-one"), tells it which file to put the new data in, and where the point coordinate file is stored. Then he says use NCPPP, and start running. (Slide 32)

And the computer does, translating the data he's supplied into a

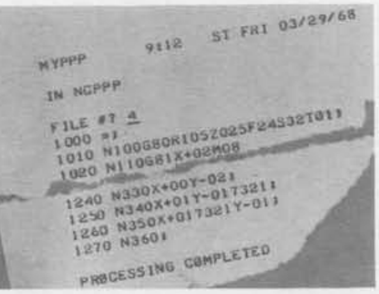


32.

complete list of instructions and coordinates in the proper control tape format for his machine. (We've cut that off, obviously.) (Slide 33)

Now he can take this information and use the third of our new N/C Time-Sharing programs — NCEIA — to punch his tape for him.

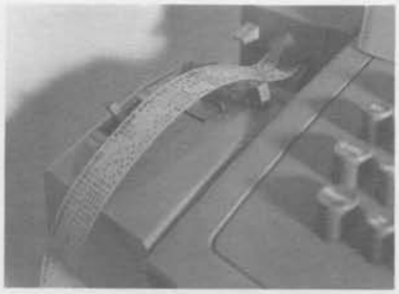
Here's how he does that. (Slide 34) It's a new program and we'll call it "my punch." All he has to



33.

do is type out four instructions, telling the computer where the formatted list of machining instructions he's just developed is stored (in his second file, remember?), telling it to call up a subroutine named Punch (1), telling it to use NCEIA, and to only punch the information actually needed for the machining—that's what "Plot" means. When it says

34.



35.

it's ready, he says run, pushes a button to turn on the punch, and out comes the tape. (Slide 35)

And there you are — from a drawing to punched tape in literally less time than it takes to talk about it. (In fact, the computer time it took to run this whole program cost a grand total of five dollars and eighty cents.)

And a problem much more complex would cost very little more to run.

36.

That, gentlemen, is how General Electric's Time-Sharing Service can be used to make life easier for your numerical control parts programmers, to make them more efficient, more accurate, and more productive.

Of course, as I said, we haven't begun to describe the true capabilities of the programs. There are dozens of special features that add to their value and utility. For example, there's a feature in the N/C point - to - point processor called "translate," which means if your man wanted to drill another dozen holes five inches to the right, he'd simply give the computer a single instruction to do just that, and it would take it from there. There are lots of things like that that can truly save a parts programmer hours of time.

37.

There are other benefits too, related to the flexibility of Time-Sharing, that we haven't discussed. Suppose you get a last-minute engineering change (I'm told that happens now and then!), or your parts programmer makes a logic error the computer can't detect— he writes "zig" when he should've zagged — and the error doesn't show up until the first part is actually being machined.

Well, with Time-Sharing it's actually a relatively simple matter to get to a terminal, call out the program listing, correct the mistake, and repunch the tape. The ability to react as quickly as that could save you the time and expense of tearing down a set-up some day.

O.K. All of this is good theory, but if I were you I'd be wondering right about now if anybody's tried it *out*, out where they use real bullets.

The answer is *yes.*

Just beginning to use the General Electric Time-Sharing Service for N/C programming is the Columbus Metals Products Company in New Jersey. (Slide 36) They make custom-designed steel cabinets for electrical equipment.

Another new user is the Waldron-Hartig Division of Midland Ross in New Brunswick, New Jersey, makers of machinery for the paper, plastics and textile industries. (Slide 37) N/C programmer Joe Vivona tells us it used to take him about three hours to set up one point-to-point program manually, but with time-sharing, he's been able to cut that down to 10 or 12 minutes.

38.

Out in the Buffalo, New York, suburb of Elma, at the Numerical Cutting Company (Slide 38), Ed Brown's people have been using Time-Sharing for parts programming for four or five months now. A custom subcontract parts manufacturer, Ed's got ten N/C machines and does things like two-axis contouring with a point-to-point system. (Slide 39)

Used to be that his people did all their programming by hand,

39.

except for some "specials" they bought outside. (Slide 40) Now using the GE Service, they are able to prepare a tape in an hour and a half for a particular contouring operation which would've required more than 1400 individual calculations and 45 man-hours if done manually.

How well has Time-Sharing worked out for Ed Brown? (Slide

40.

41.

41) Just *ask* him. He's attending the meeting. As a matter of fact, they're getting to be pretty sophisticated time-sharing users. As long as they have the terminal in place, they think they'll start using it for production analysis and marketing studies as well. No reason why not.

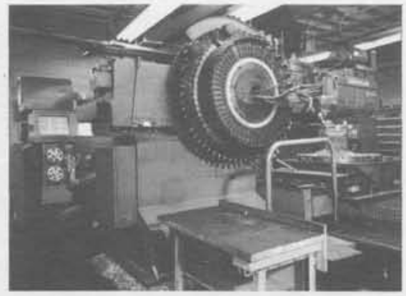
In Cleveland, the Quality Gage and Manufacturing Company does all kinds of tape-controlled drilling and milling, also on a job-shop basis. (Slide 42) There, too, parts

42.

programming used to be done by hand, using a desk calculator.

Now they're using Time-Sharing and plant manager "Buzz" Buzzelli (Slide 43) called it "Terrific — a heck of a time-saver!", citing one contouring program the computer ran in 4 minutes that would've taken a good two days the old way just to run, and another day or two to check.

43.



44.

Also in Cleveland is the NuMac Corporation, specialists in tape-controlled machining of parts for the aerospace industry. (Slide 44) They had a computer in-house (Slide 45), but switched to Time-Sharing for N/C programming a few months ago. The reason? Engineering manager Lloyd Kagley (Slide 46) says it was a matter of incremental versus fixed cost. With Time-Sharing, they pay only for the computer time they actually use.

In Akron, the Akron Equipment Company makes engraved tire molds for rubber and tire companies all over the world. (Slide 47) They have half-a-dozen N/C machines and have been using Time-Sharing for tape preparation assistance for nearly a year now.

And, of course, a number of GE plants have been using these Time-Sharing programs for help in making N/C tapes, too. (Slide 48) Among them is the Lamp Equipment Operation. (Slide 49) They use their N/C machines to make cams, which are in turn used in lamp-making. Phil Grim (Slide 50), who heads up manufacturing engineering out there, turned to Time-Sharing for this application despite the fact that they have access to huge computer complexes serving the whole GE Lamp Division. Why? Because Time-Sharing is so much faster. They can generate tapes in a couple of hours or less now, instead of waiting a couple of days for them to come back, as before.

45.

46.

So . . . although this N/C programming capability is a new feature of our Time-Sharing service, it is completely field-tested,

47.

and there is a good and growing number of very satisfied users.

This new capability is really just a natural extension of GE's total interest in the numerical control field, combined with our desire to make the advantages of Time-Sharing service available to more and more people with problems to solve. General Electric Time-

48.

49.

Sharing Service users now number more than 50,000 in this country and overseas, and to keep up with this accelerating demand we're installing new systems and opening new centers at a rapid rate.

This duality of experience—and of commercial interest—on our part guarantees our dedication to success in this field, and makes further progress a natural, too.

The N/C programs we've talked about this afternoon represent breakthroughs a long time in the planning—and the beginning of a whole series of exciting innovations you'll be seeing in N/C Time-Sharing service.

50.

GENERAL ELECTRIC

122058
5-68 (4M)

PRINTED IN U.S.A.