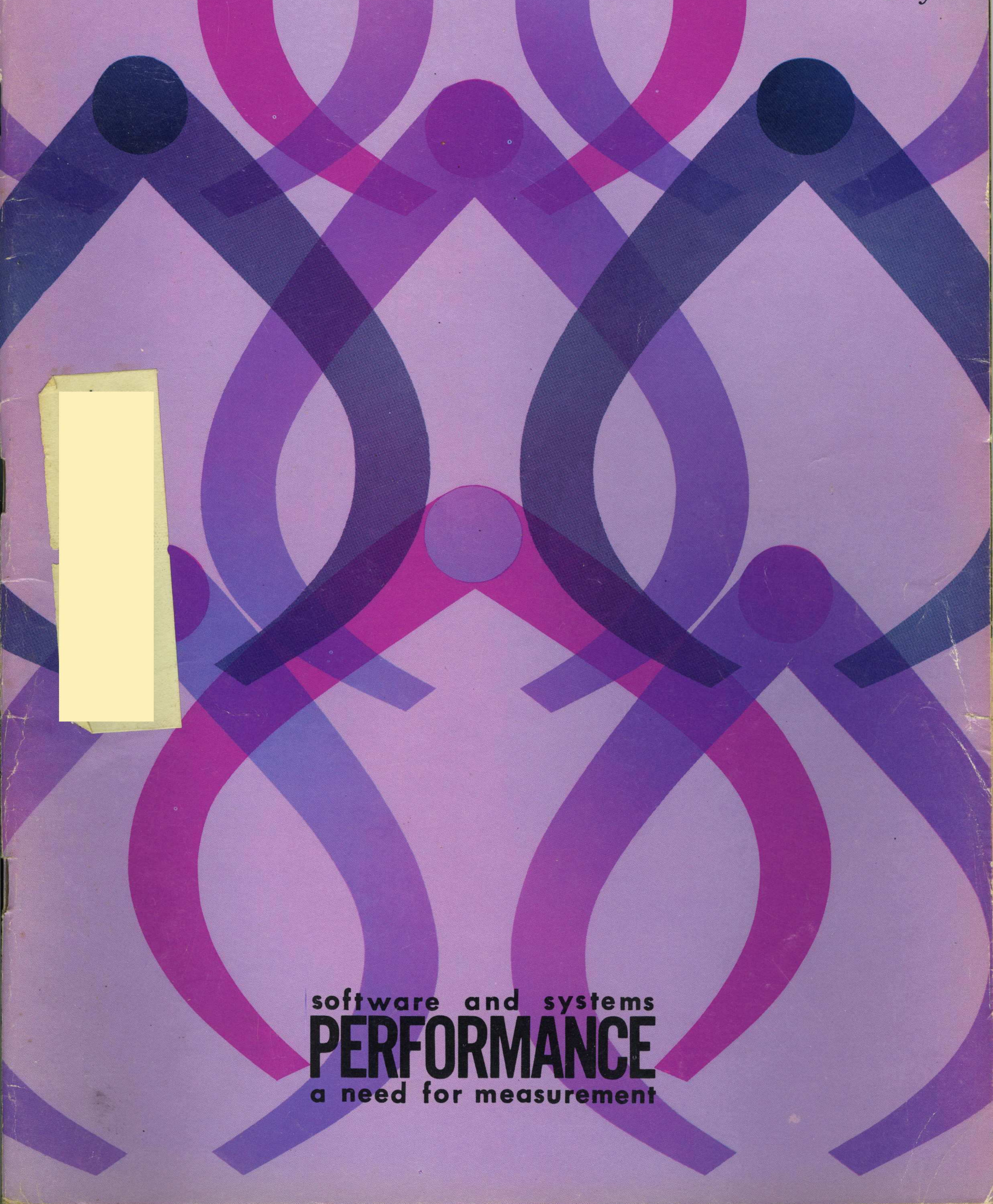
# DATAMATION 64®

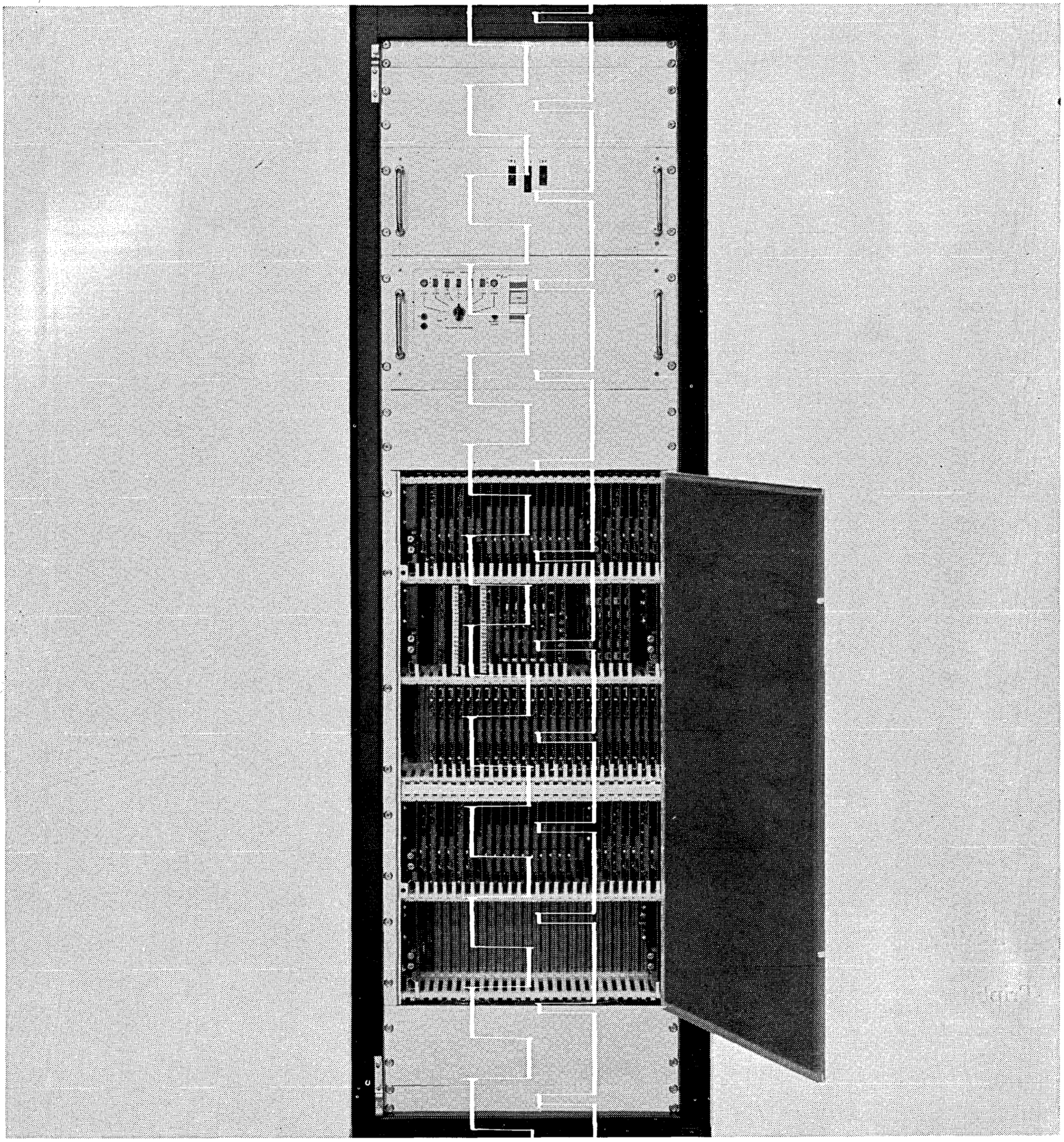
July



software and systems

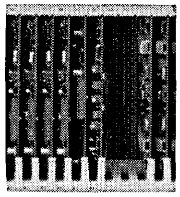
# PERFORMANCE

a need for measurement



**Special introductory offer: 1.8 microseconds for the price of a 6.**

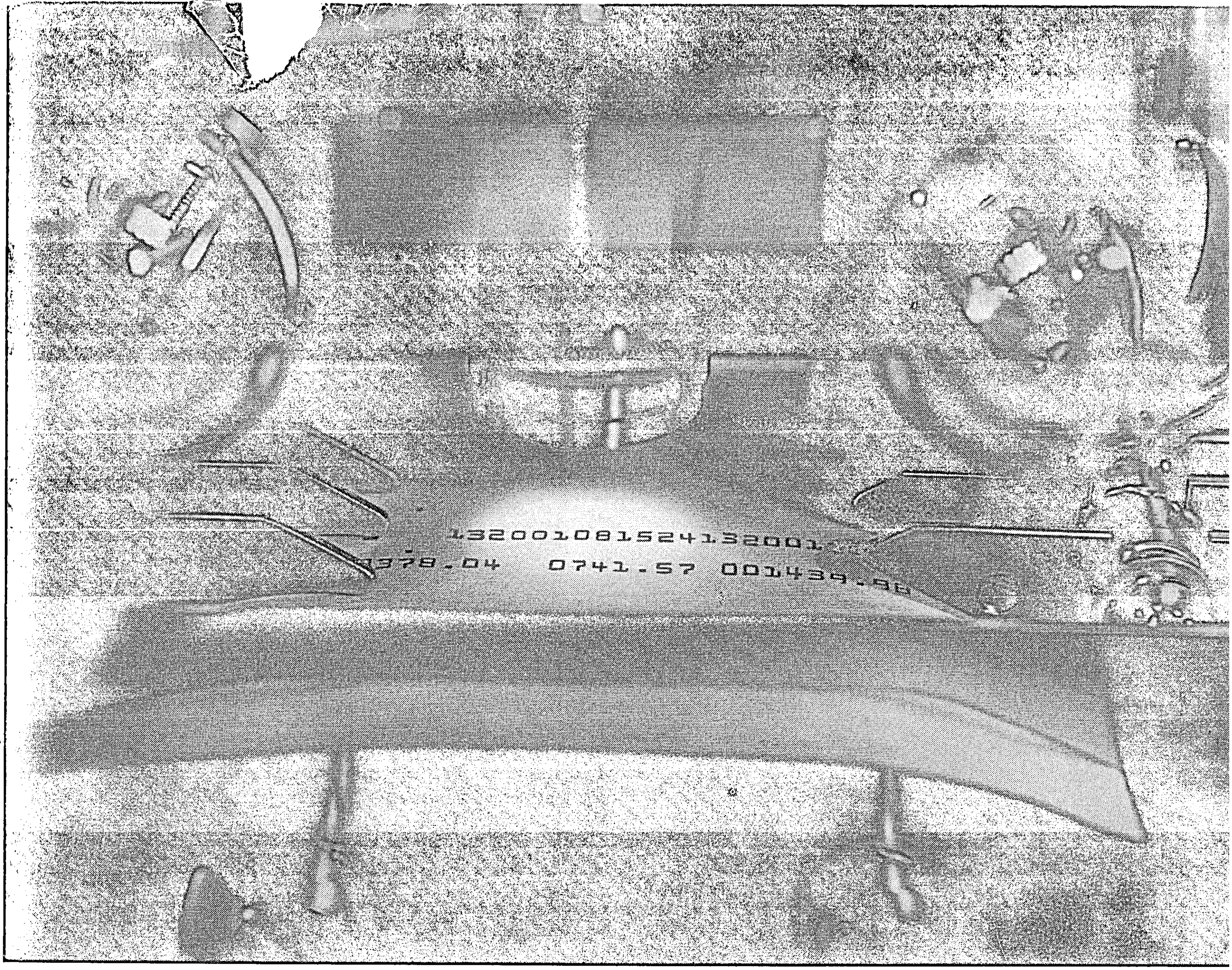
Introducing the Ampex RZ, up to 16,384 words of core memory with a complete cycle time of only 1.8 microseconds. And a price tag lower than many 4 and 6 microsecond memories. The latest in the highly successful Ampex R series (RVQ, RQA, RQL, RVS), the RZ also offers a longer word length: 8 to 56 bits. With no redundancy in drive circuitry. Access time is faster: 650 nanoseconds. And the RZ has a wide environmental range: +10°C to +45°C. The RZ is fast, big and flexible enough to be used as a main-frame memory, an auxiliary memory or for high-speed bulk storage.



AMPEX

And because of its relatively low-cost, it's well worth your consideration, whether you're looking for a high-performance system, a/d conversion, or a way of updating your present computer. Suggestion: write Ampex for full details. Why Ampex? Because we can offer you core memories in a wide variety of storage capacities and speeds. With cycle times from 24 to 1 microseconds in capacities up to 1 million bits or more. Ampex Computer Products Division, Culver City, California. Term leasing and financing. Sales and service engineers throughout the world.

CIRCLE 1 ON READER CARD




# What's a TV camera doing in a computer?

Tripling data re-entry speed.

**I'm a businessman, not an astronaut.**

Videoscan is designed for businessmen—profit-conscious businessmen. This unique, TV-based optical character reader permits you to feed source documents directly to a computer, without first converting to punched holes in cards or paper tape.

**I still don't get the TV connection . . .**

Videoscan's sensing device is the famous RCA-developed Vidicon tube. It reads characters passing before it like the beam-sweep on a TV screen.

**I'm impressed . . . but what does this buy me in data processing power?**

The ability to read live data directly into an RCA computer at speeds up to 1500 characters per second, or 90,000 documents an hour. Utility bills, premium notices, coupons, tickets . . . Videoscan can read them faster, more economically, more reliably than any system ever developed.

**All because of a TV-camera?**

Plus RCA electronics—a high-speed transport system . . . and the powerful performance of RCA computers.

**I'm interested. What channel do I dial?**

Dial your local RCA EDP office, immediately—for full details and a system evaluation. Or just write.

RCA ELECTRONIC DATA PROCESSING, CHERRY HILL, NEW JERSEY.

RCA® The Most Trusted Name in Electronics

CIRCLE 4 ON READER CARD

# WHEN SHOULD YOU SEND AND

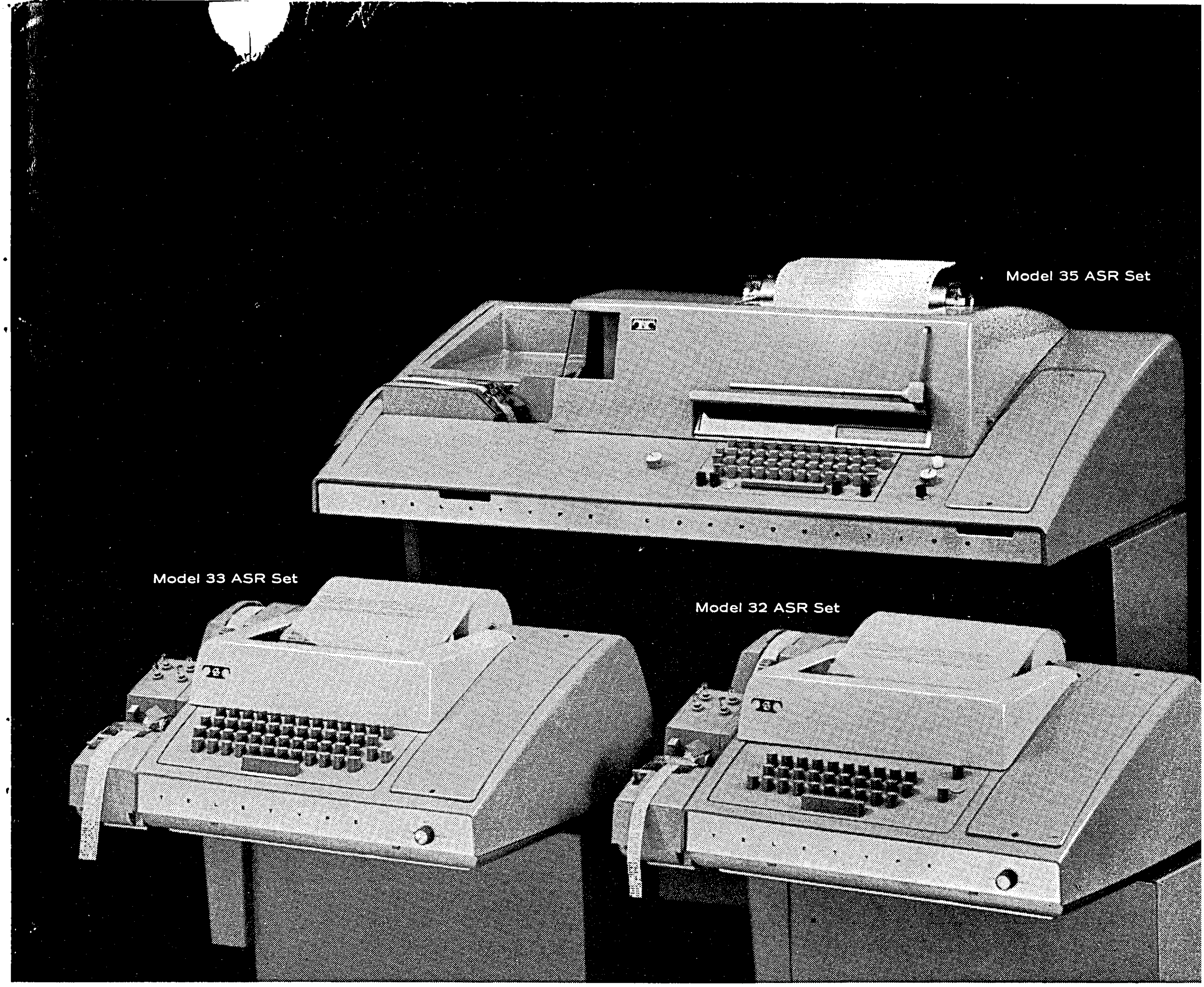
Many firms are stalled behind a communications barrier caused by increased paperwork. Thus, simplifying and speeding up communications will help make operations more efficient and profitable. Firms, using Teletype page printers to handle routine communications, could speed things up considerably by adding the versatility of automatic transmission with punched paper tape.

*Three advantages of automatic send-receive sets.* Operators can punch messages and data off-line providing errorless tape for later transmission. Messages and data can be collected from many sources at different times and transmitted on-line more economically at maximum speed in one continuous message. Punched tape makes it possible to store messages or basic data that can be used again and even combined with variable data to save retyping.

*Three models to choose from.* There are three different Teletype automatic send-receive sets to satisfy your communication needs. The Models 32 and 33 are the most economical to use where traffic ranges from normal to light. The Model 35 is designed for handling a much larger volume of messages and data, as well as offering increased versatility for on-line and off-line communications.

*Communicates with business machines.* Actually, punched paper tape prepared on a Teletype automatic send-receive set can do much to automate communication procedures. These ASR sets "speak" the same language as many business machines and computers because they operate on an 8-level code, which conforms to the newly approved American Standard Code for information interchange.

*How are Teletype ASR sets used?* Teletype ASR sets speed up order processing by reducing errors and duplication of paperwork between departments; improve warehousing, purchasing procedures, and distribution methods; and link business machines with existing channels of communication. For example:


Model 35 ASR Set
Model 33 ASR Set
Model 32 ASR Set

# DATAMATION 64

*the automatic handling of information*

*volume 10 number* 7





EDITORIAL OFFICES

1830 WEST OLYMPIC BLVD.
LOS ANGELES, CALIF. 90006

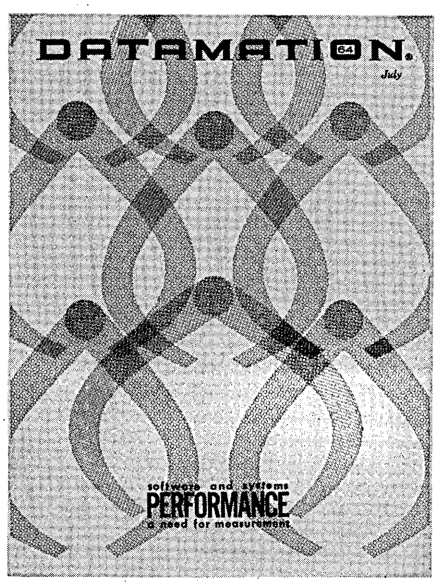

## Feature Articles



## Departments




THIS ISSUE — 46,952 COPIES


## Cover

An industry which talks glibly about nanosecond cycle times, "real-time" and "throughput" still lacks the fundamental tools for measuring the power of the machines and systems it proudly produces. Two articles this month highlight the need for measurement in two critical arenas: system performance and software. Representing the basic theme of this month's issue is the cover, designed by Art Director Cleve Boutell.


Circulation audited by Business Publications Audit — BPA

Member, National Business Publications — NBP

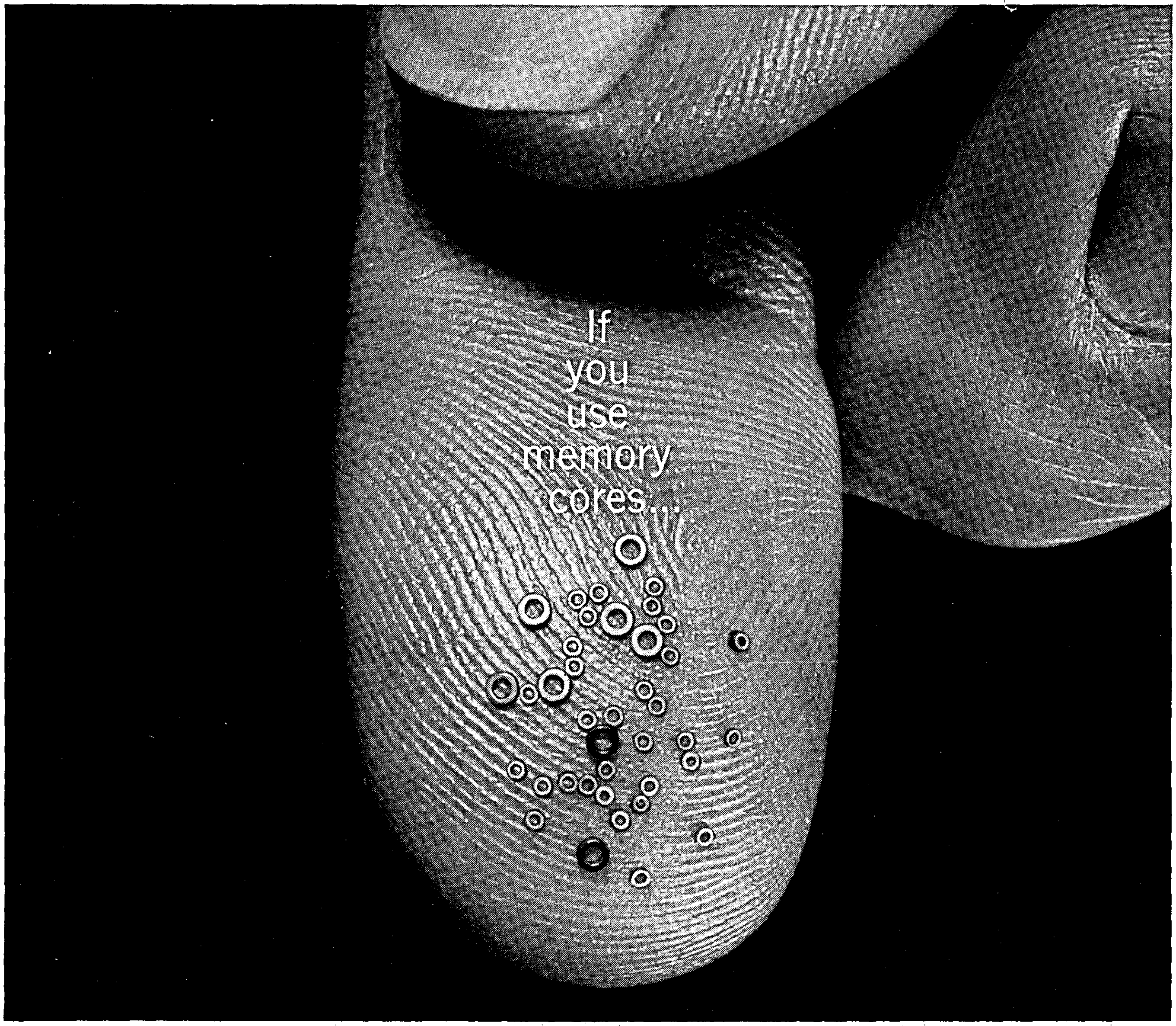
DATAMATION is circulated without charge by name and title to individuals who qualify by our standards and are employed by manufacturers or users of automatic information-handling equipment in all branches of business, industry, government and the military. Available to others by subscription at the rate of $15.00 annually; single issues (when available) $1.50. Foreign subscriptions are on a paid basis only at a rate of $25.00 annually. No subscription agency is authorized by us to solicit or take orders for subscriptions.

Microfilm copies of **DATAMATION** may be obtained from University Microfilms, Inc., 313 No. First St., Ann Arbor, Michigan.

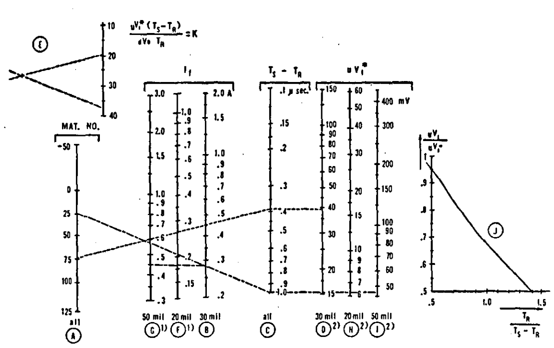
DATAMATION is published monthly on or about the tenth day of every month by F. D. Thompson Publications, Inc., Frank D. Thompson, president. Executive, Circulation and Advertising offices, 141 East 44th Street, New York, New York 10017 (MUrray Hill 7-5180). Editorial offices, 1830 W. Olympic Blvd., Los Angeles, California 90006. Published at Chicago, Ill. Controlled Circulation paid at Columbus, O. Form 3579 to be sent to F. D. Thompson Publications, Inc., 205 W. Wacker Dr., Chicago, Ill. 60606.



Printed by Beslow Associates, Inc.




**THIS WILL CHANGE YOUR WAY OF THINKING!**



Another breakthrough. ■ A new IGC process automates the manufacture of FERRAMIC® memory cores for coincident current memories. ■ Makes possible all magnetic and physical property combinations within nature's boundaries. ■ Maintains core thresholds to within ±2%. ■ Permits sampling of any core within two weeks. ■ Send for IGC's Memory Core Nomograph — **in minutes,** select your core size, switching speed and drive current. ■ We told you it would change your way of thinking. ■ Now write to Indiana General Corporation, Electronics Division / Memory Products, Keasbey, New Jersey.

**INDIANA GENERAL**

CIRCLE 48 ON READER CARD

# DATAMATION calendar

● The Engineering Extension of UCLA, Los Angeles, will conduct a course in Reliability and Industrial Statistics. Session I will be held August 3-14; session II will be held August 17-28.

● A special program on Language Data Processing will be held August 10-21 at the Harvard Summer School, Cambridge, Mass. Topics include survey of mathematical linguistics, formal languages and associated machines, syntactic analysis of natural languages, automatic analysis of content and language processing applications.

● The third annual meeting of the Users of Automatic Information Display Equipment will be held Aug. 12-14 at the International Hotel, Los Angeles. UAIDE is an organization of S-C 4020 users.

● The Western Electronic Show and Convention will be held in Los Angeles, Aug. 25-28. Exhibits will be displayed in the L.A. Sports Arena and Hollywood Park racetrack.

● The Fourth International Analogue Computation Meetings will be held September 14-18, 1964, at the Technical College, Brighton, England. Sponsor is the British Computer Society, under the sponsorship of the International Association for Analogue Computation.

● The Northwest Computing Assn. will hold its seventh annual conference, Sept. 17-18. Co-sponsored by the Univ. of Washington Computing Center, the meeting will be held at the Univ. of Washington, Seattle.

● A seminar on the management of government contract research will be held Sept. 28-Oct. 2 at the Executive House in Wash., D.C. The institute is sponsored by American Univ.

● Papers on dp systems for state and local government are being accepted for the conference which will be held Sept. 30-Oct. 2 in New York City. This event is co-sponsored by New York Univ. and System Development Corp.


# "...Let every man keep a close watch!"

Lieutenant James Allen and his small detachment of soldiers cautiously portaged through the dense woods around the perilous but beautiful St. Croix falls. The date was July 28, 1832 and the Chippewa and the Sioux were at war. Allen's diary candidly refers to "inhospitable Indian country." Wandering white men in those days often failed in health and hairline if they failed to keep "a close watch."

Near this same spot, 131 years later, very careful men are designing and building core memories at Amery, Wisconsin.

With Fabri-Tek's manufacturing facility stringing over two million cores a week, quality assurance means the health of a reputation — a reputation that is based on "a close watch!" Every single core is fully inspected three times before it is finally shipped as a part of a memory plant, stack or system.

The Fabri-Tek Quality Assurance group reports directly to management with no middlemen to absorb the shock of a hard-hitting quality program.

Is it possible that you're in the digital systems business and haven't discovered Fabri-Tek? These people who are concerned with careful quality have:

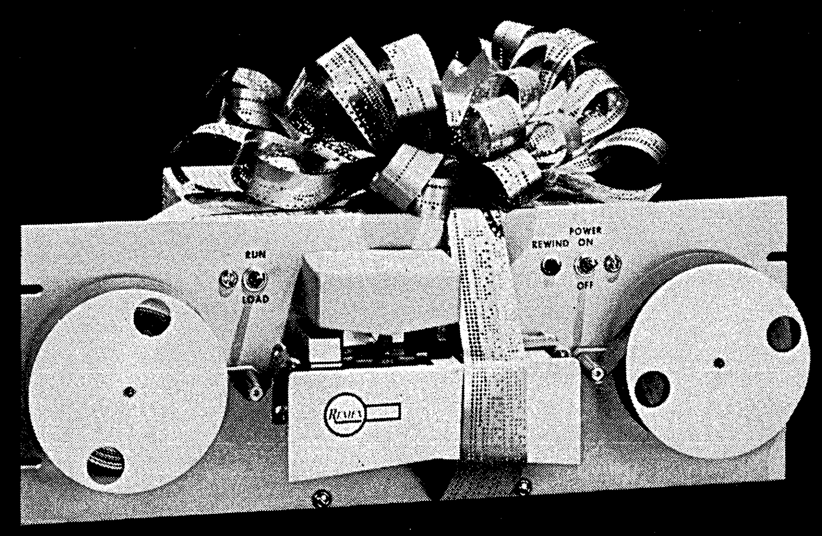
Airborne Instruments Laboratory
Dynatronics
Ferranti-Packard
Hamilton Standard
Navcor
Potter Instrument

For the asking, we'll be happy to send you some very interesting material about core memories and memory systems. Write FABRI-TEK, Incorporated, Amery, Wisconsin.

FABRI-TEK

EXECUTIVE OFFICES: FOSHAY TOWER, MINNEAPOLIS 2, MINNESOTA

CIRCLE 11 ON READER CARD






# letters

### non-decision plaudits

*Sir:*

David Bourland Jr. is to be congratulated for his very fundamental formulation ("Non-Decision Theory," May, p. 52) of an old subject which until now has eluded mathematical analysis.

The article elicited the following comment from a high-level supervisor in a client's organization: "This paper confirms my belief that administration exists because of uncertainty—and if decisions were immediately forthcoming there would be no uncertainty."

It would now appear that Parkinson's Law is derivable from non-decision theory. The proof is left for Mr. Bourland. (Please do not print my name lest the resulting number of milli-decisions diverge and overcome the strong law).

NAME WITHHELD

### cybernation & enigma

*Sir:*

Plaudits are in order for your most lucid treatise in the May issue (p. 83) entitled "Cybernation Featured in Triple Revolution Paper." As the article implies, the subject is enigmatic in nature, running the gamut "creation of the universe" through "the effects of frost on ferrite cores." We were led inexplicably to the true essence of the problem, your article a beacon of guiding light in the tortuous darkness of ignorance. (We really do know the word cryogenic).

JUDITH GREEN
DOUGLAS C. HARRISON
*Newton Centre, Massachusetts*

**We're always suspicious of people who live in towns that spell "center" c-e-n-t-r-e. So take tongue from check, dear programmers, if that's where it is. That beacon you saw was meant to spotlight, not illuminate, the topic of cybernation. And the story was written to arouse sufficient interest so that you'd write for a copy—and read it. We recommend it especially to those in the computre field.**

Computer
$21,800

## The PB250 Computer as a Digital System Control Unit gives you:

- High speed at lowest cost
- High reliability—more than 150 in use
- Stored program flexibility
- 51 scientific/systems commands
- Standard peripheral equipment for reduced systems engineering cost

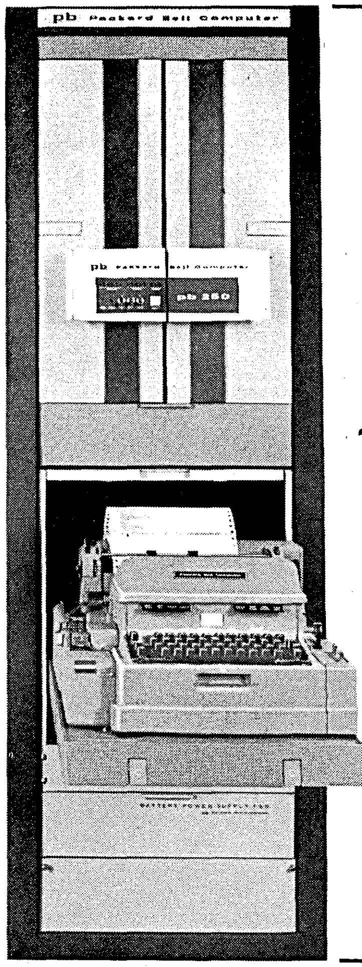
Now you can take advantage of the speed and power of a rack-mounted 2 mc solid-state computer for $21,800. A 22-bit word means high accuracy for all your applications, and 51 commands give control and computing capacity superior to computers costing up to twice as much.

PB250 instructions include add, subtract, multiply, divide and square root. Add and subtract are executed in 12 microseconds; multiply, divide and square root in 252 microseconds. These and many other powerful logic and arithmetic functions make the PB250 capable of performing at high speed on complex control and computation system problems.

To match the speed, power and versatility of this small computer with custom-designed equipment will cost three to five times as much as the PB250 and not be as reliable. Consider the flexibility of the PB250. Last-minute changes before system delivery, or revised or added requirements after system installation, are handled quickly and economically by changes in PB250's stored program.

For systems operation, the PB250 provides up to 29 control input and 32 control output lines for sensing, signalling and controlling. An optional high-speed buffer provides two-way communication between the PB250 and asynchronous system elements at up to 85,000 wps. Memory expands from 1,040 words to 15,888.

You can easily expand your system with standard PB250 peripherals: magnetic tape, paper tape readers & punches, card readers, digital graph recorders, buffers, battery power supplies, memory modules, A/D and D/A converters. A full software library includes algebraic compilers, interpreters and many useful subroutines.



***A few current systems applications:***

*Digital-analog simulation of space vehicle control and guidance systems*

*Control of digital system for recovery of satellite doppler data*

*Real-time tracking and control of radar antennas*

*Rapid on-line monitoring in data acquisition system*

*Automatic checkout of space vehicles*

*Real-time satellite tracking, antenna control and visual display*

*On-line performance analysis of steam cycles*

*Computer-controlled astronomical telescope system*

*Control of nylon filament manufacturing process*

*Real-time, on-line simulation of inertial navigation systems*

*Controlling data acquisition system for shock-tube wind tunnel*

*Controlling mobile data acquisition facility*

*Write for Data File C-26-7.*

Complete
Data System
$45,500

## The CDS250 – a packaged data system ready for use

The CDS250 includes the PB250 with 2320 words of memory and Flexowriter, plus a 64-channel multiplexer, 64 kc 12-bit A/D converter, buffer and control unit . . . all neatly mounted in a single rack that can be rolled from job to job. Up to 64 external devices (128 optional) and 32,000 input channels may be addressed directly. The CDS250 is delivered ready to plug in to transducers or test equipment or to serve as a prepackaged subsystem within a larger complex. All standard PB250 peripheral gear and a 15-bit A/D converter are optional.

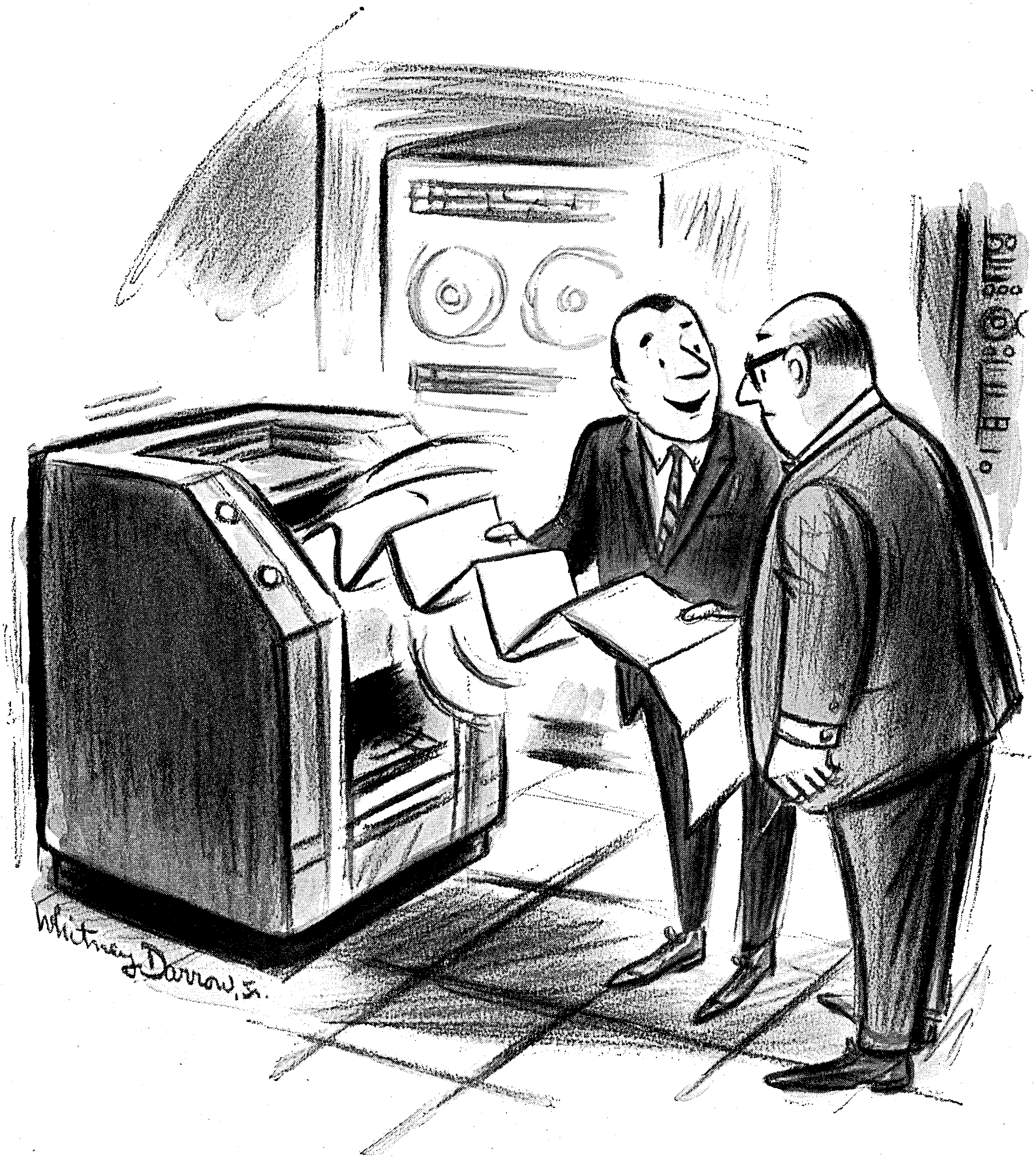
Inputs to the CDS250 may originate from production test equipment, satellite and missile check stands, laboratory and process analytical instruments, tensile testers, etc. Output includes IBM-gapped magnetic tape, paper tape, reduced and logged data, alarms, displays, plotters, or signals for system or process control.

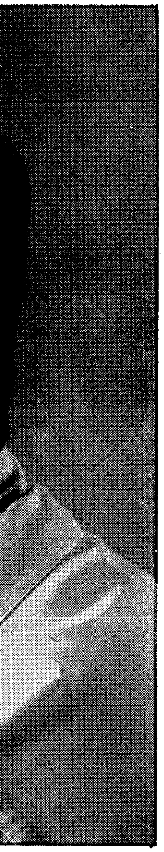
Offered at no charge: programming package for system operation, two-week programming course, and three-week maintenance course.

**Packard Bell Computer**

2700 South Fairview Street • Santa Ana, California • Phone: (714) 546-7160

CIRCLE 13 ON READER CARD


Whitney Darrow, Jr.



ess...





DATAMATION

# BUSINESS & SCIENCE

## UNIVAC'S ANSWER TO 360: HOT-RODDED 1107

Univac hopes that its hopped-up 1107 – the 1108, scheduled for announcement in early July – will help it cash in on the transition problem facing 7090/94 users stepping up to the 360. Reportedly sporting a 375-nanosecond cycle time, the 1108 should overcome the processing sluggishness of the 1107, which already out-compiles the 94. At Boeing, Seattle, the 1107 offers a 6-7 FORTRAN compiling edge over the 94. In a linear programming test, the 1107 reportedly took 2 hrs, 8 minutes to do a job which required five hours on the 94.

## COURTSHIP OF C-E-I-R, CDC SEEMS ENDED

The rumors concerning negotiations between Control Data and CEIR were true. But, for the second time in recent weeks, the talks have been broken off... presumably for good this time. The last round of talks in Minneapolis followed by only a week a formal statement from CEIR that ''tentative talks'' had been concluded. This time CEIR has notified its troops that the talks were over, and the final CDC offer rejected. It ends the delightful speculation on what Control Data would have done with a slug of service bureaus loaded with IBM hardware...and a $45-million order for new 360 systems.

Now CEIR can turn its attention again to seeking acquisitions. With some cash burning a hole in its corporate pocket, the Washington, D.C., firm says it's interested only in something it can easily digest... somebody with under $10 million gross.

## AUSTRALIAN ACADEMICS INDICATE INDEPENDENCE

All kinds of dreams were smashed in Australia recently when the five universities seeking computers went their separate ways. Monash Univ. will get a large 3200 in cooperation with Control Data's data centres organization; National U. is getting an IBM model 50; New South Wales will probably do likewise; Western Australia will probably go the PDP-6 route; a fifth school is accepting a gift IBM machine, we hear. The selection of incompatibles ends hopes the universities could tie into the big CSIRO network.

## 465L A FAILURE? WELL...NO

In answer to an ugly question raised in these pages three months ago, it appears that 465L has not been written off as a failure. DOD officials admit the

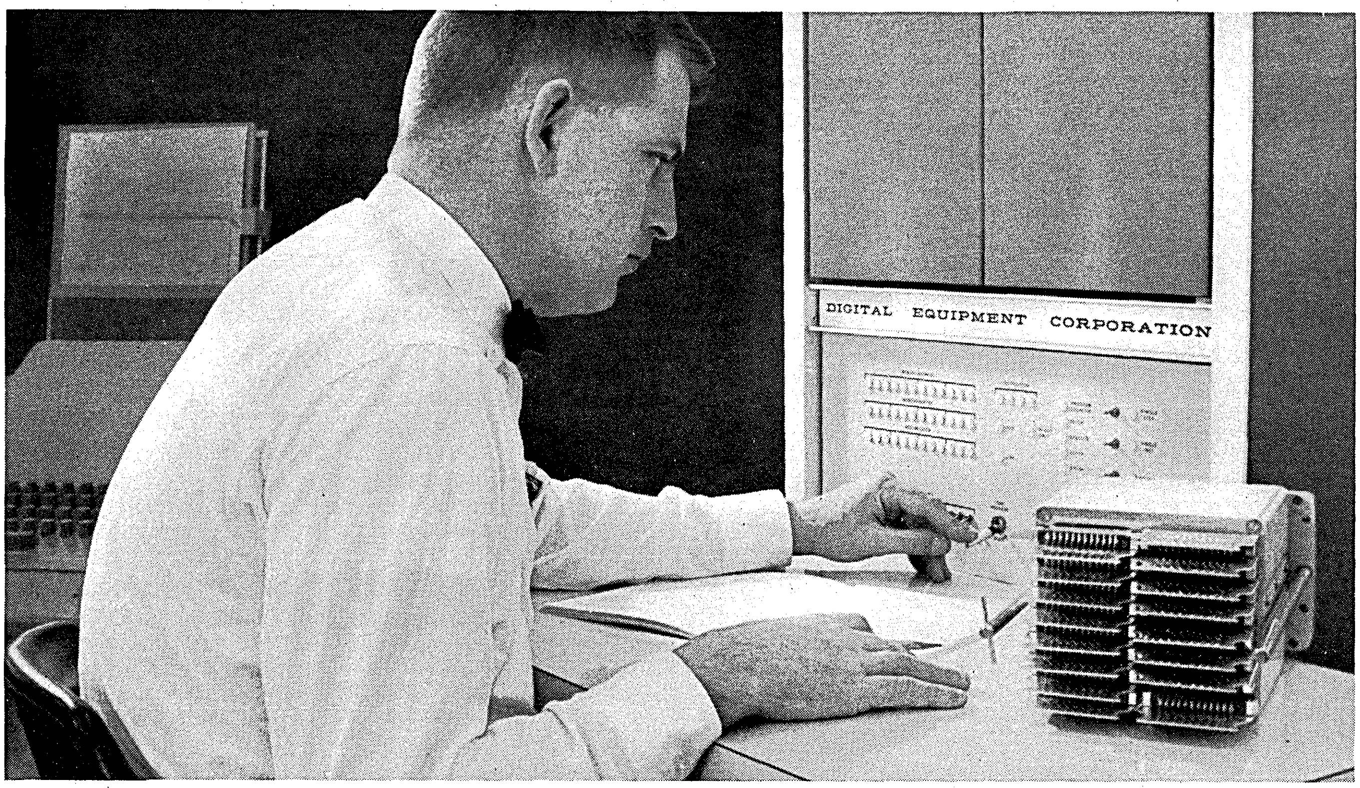
system will not live up to original specifications, which were evidently outside the pale of ''meetability.'' But the big, big Strategic Air Command system is now being installed despite its share of developmental problems, and will be operating early next year.

**CDC MAKES SHARP CLASS DISTINCTION**

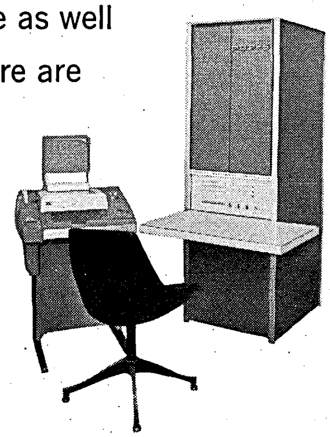
Taking issue with *Datamation's* interpretation that the 6600 market life would be shortened by the 360, Control Data says that its machine and the mod 70 (top of the announced line) aren't even in the same class. Says CDC, the 6600 has a 7-12 speed ratio advantage over the mod 70, and they expect a cost/performance ratio of ''better than 7-1.'' for ''typical installations.'' They also point to advantages in one-usec core and high-speed channels for multiprocessing. Now it's IBM's turn.

**RUMORS AND RAW RANDOM DATA**

One peddler of mag tape drives estimates there are maybe some 225,000 units installed...Inside scoop is that Ivan Sutherland of MIT will succeed JCR Licklider at the DOD Advanced Research Projects Agency post which has funded a great deal of time-sharing research (including Project MAC). Sutherland has done extensive on-line research, so it appears that the Licklider time-sharing legacy at DOD will be safe... Quote-of-the-month from new ACM prexy George Forsythe, answering a question from the floor concerning the possibility of running FORTRAN algorithms in the *ACM Communications*: ''The trouble with FORTRAN — there are too many, and it's always changing. ALGOL is just a better language for stating algorithms''...Control Data claims it will have the world's biggest data center at the old Bendix plant in L.A. The complex will include 32 ,34, 36 and 66 hundreds...That big, multi-bank, on-line system for the Savings Banks Trust Co., NYC (April, p. 17), is supposed to go to IBM. Consultant Diebold is said to have recommended a 360 ...AFIPS chairman Don Madden is leaving IBM after a little over a year there...U.S. flowchart standards progress has slowed temporarily as X3 recommends that they conform to international standards, requiring some five changes...Almost ready for publication as proposed standards: basic and full FORTRAN, a glossary (now called a vocabulary), and optic character recognition...Says one software consultant of life since the 360: ''I've never had it so good''...The Wadsworth VA hospital in Los Angeles is time-sharing the Q-32 computer at System Development Corp. as part of the Automated Hospital Information system study. So far, it's only for communication of doctors' orders for lab tests, and transmission of results back to the ward...Didja hear what astronaut Wally Schirra said of his thoughts just before lift-off for his six-orbit flight: ''All that gear beneath me was built by the lowest bidder.''




**You can own the PDP-5 computer for what a core memory alone used to cost: $27,000.** This general purpose computer is ideal for scientific computations, system and process control, and data collection and reduction. Don't let the low price fool you. The PDP-5 has features and capabilities many higher priced computers wish they had. A six-microsecond memory cycle and fully parallel processing, for example, permit a 55,555 additions-per-second computation rate as well as real-time data compilation and processing or recording. Then there are the 4096-word random access magnetic core memory (or 1024-word memory for $3,000 less), 12-bit word length, and 24-bit arithmetic. Add to these a two-megacycle bit input via built-in data channel, program interrupt for input-output devices, and a complete software package—including FORTRAN. Price includes keyboard-printer, paper tape reader, and punch. Impressed? Then let's get together soon and talk computers.



digital
EQUIPMENT
CORPORATION
MAYNARD, MASSACHUSETTS

Los Angeles • Palo Alto • Park Ridge, Ill. • Washington, D. C. • Parsippany, N. J. • Pittsburgh, Pa. • Ottawa, Canada • Munich, Germany • Sydney, Australia

CIRCLE 18 ON READER CARD

# WASHINGTON REPORT

**BANKS & DP: LONG, LONG TRAIL**

H.R. 9548, the proposal submitted by Rep. Multer of New York to prevent banks from offering dp services to non-bank users, has come in for some slight alteration, although the main thrust of the bill is unchanged. The language has been touched up so as not to make illegal customary bank services for clients, such as check reconcilement and other routine tabulations. As amended, the bill is soon expected to go before the full House Banking and Currency committee.

**MORE CASH FOR DP TRAINING**

Breakup up legislative logjam created by the civil rights filibuster likely means a substantial chunk of federal cash will soon be forthcoming for dp training courses in the nation's high schools and universities. Senatorial consideration of funds to implement the Vocational Education Act of 1963, known as the Perkins Act, has been stymied by the marathon oratory, but now favorable action by the solons is considered highly probable in the near future.

The appropriations measure, as it passed the House, would allot $118.5 million in fiscal '65 for an expanded vocational training program to be administered through the states, who would get the federal cash on a matching basis for approved training curriculums. This new money is in addition to funds now supplied under other measures for vocational training. The Perkins Act also authorized $177.5 million for expanded vocational training in fiscal '66, and $225 million for each fiscal year thereafter.

Dp and computer skills come into the act because the bill establishes as a criterion for courses that successful graduates be able to get employment in the areas in which they have been trained. A healthy change.

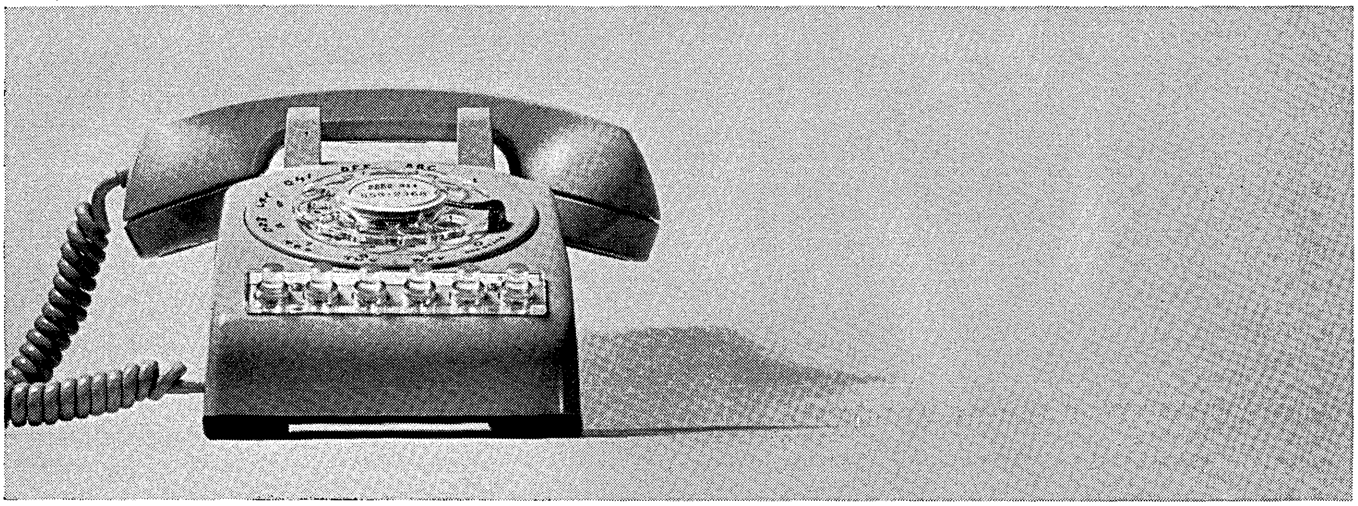
When funds are made available, the Health, Education & Welfare Dept.'s Technical Education section is ready to embark on a greatly expanded dp training program. Currently, from the almost $60 million now available for vocational training, about $15 million is alloted to technical education. Of this, about one-third has gone into the establishment of pilot dp training programs in 40 cooperating states and for related projects.

Should the same formula be observed in alloting funds under the Perkins Act, this would mean an additional $10 million for dp training in fiscal '65, $15 million in '66, and $20 million thereafter. The bill also provides for the use of funds for new building construction and for research grants on new training programs, which again could very well involve dp skills.

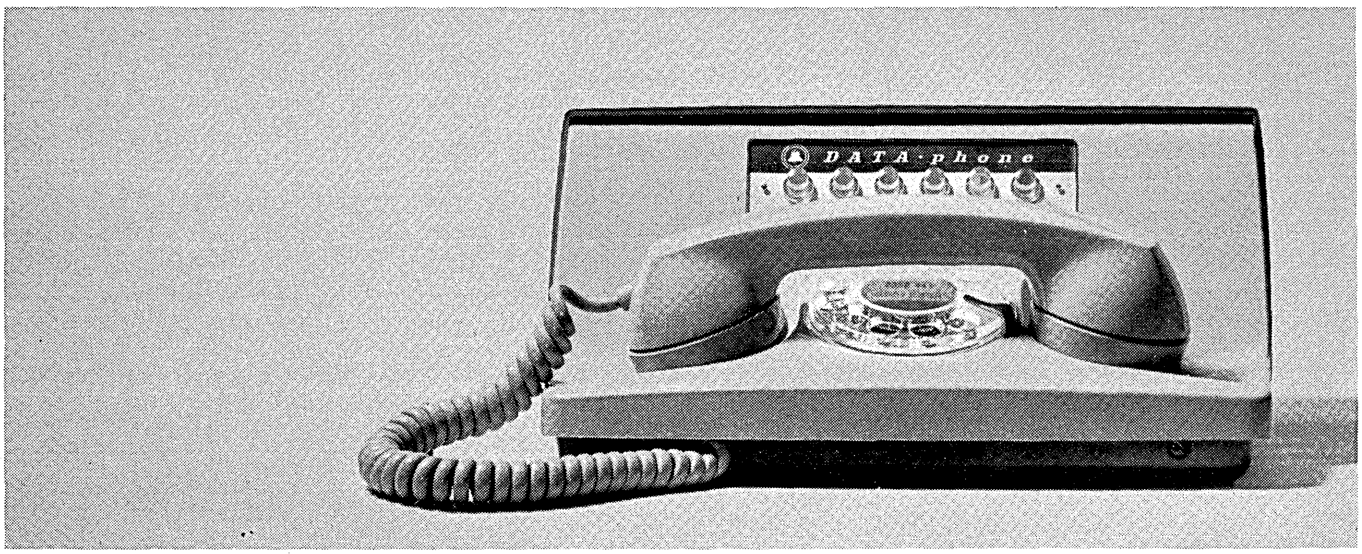
**NO WINDFALL, NO PRATT FALL, SAYS DOD**

The Defense Dept., which switched from a lease to a purchase policy for computers only after several years of guerrilla resistance in the bureaucratic jungles of Washington, was first surprised, then irked at a charge in Jack Anderson's Washington Merry-Go-Round

*Continued on page 87*



# What's the connection?




Telephones are for *people talk* . . . for making appointments, planning meetings, discussing problems, inquiring, informing, announcing, explaining.

DATA-PHONE data sets are for *machine talk* . . . for transmitting inventories, payrolls, sales figures, receivables and other business data at speeds of 100 to 2700 words per minute.

Combine the two services . . . and you'll have an *integrated* information-handling system that will help you save time, control costs, reduce paperwork and sharpen service to customers. Profitably.

Our Communications Consultant is ready and able to plan and set up such a system for you. Just call your Bell Telephone Business Office and ask for his services.

**Bell Telephone System**

Serving you

CIRCLE 19 ON READER CARD

# EDITOR'S READOUT

## THE HOUSE OF SCIENCE

The house of science is a splendid and many-hedroned thing, a sparkling ivory edifice without windows in which learned men ponder abstruse and esoteric imponderables.

According to the minstrels of science, it is from this all-encompassing dome that progress flows. This despite the fact that science has no specific goal, no tidy target. It is the sublime (if evasive) truth, the nagging gnat of curiosity which lures and drives men on, seeking knowledge for its own sake.

That's in the front wing. 'Way in the back, or maybe underground, men whose white coats are tattle-tale gray and just a little frayed think about other sorts of things. If their work is not *too* impractical, it's called Research.

In an annex, grubby people worry about petty, practical problems. They are shlonks and toads and their fingernails are dirty.

We're just kidding, of course. The editor of a magazine named *Datamation* would have to be out of his skull to knock science. We're all for it, really. That and progress and motherhood.

But we *do* think there is a tendency to think of science as exclusively devoted to the abstract and the mysterious. 'Tain't so. Science includes also some very basic and practical matters . . . measurement, for instance. How far is it from here to there? How much does this stuff weigh? And, more important, how do we know? Scientific pros like Archimedes and Galileo realized the importance of measurement.

The young and teensy province of computerdom must realize it too. In its haste to catch up with all the work awaiting its magical powers, computing has overlooked or bypassed some rather critical matters of measurement.

We still don't know exactly how to tell just how effective or efficient a computer system is. Who knows how to evaluate software? Or how to measure programmer aptitude . . . or productivity? Even plain old hardware evaluation —formerly a simple matter of comparing cycle and access times—is a tricky, sticky thing.

So . . . while we encourage and support the kind of modern metaphysics and far-out thinking that is ordinarily thought of as representing "true" scientific endeavor, let's not forget the fundamental importance of plain, down-to-earth problems like measurement. Even if we know where we're going, it would be nice to be able to know how to tell when we've gotten there.

# MEASURING PERFORMANCE

by ROBERT L. PATRICK

In man's brief but hectic history, many a debate has been held by participants spouting opinion as if it were fact. The even briefer and more hectic history of information processing is no exception; the pages of this very magazine have carried some of these airy arguments. The casual reader has been hard put to separate fact from fantasy. In the early days of the field, all was exploration, and considered opinion, of necessity, held sway. As the field matures, opinion must give way to analysis and analysis must be supported by measurement.

Recent new equipment announcements change all of our sacred comparative measures: cycle time, logical organization, word length, channel control; management is concerned about rising costs; the federal government is contemplating drastic steps to bring its costs under control; many new techniques are being discussed (e.g., time-sharing) in new research areas for which no accepted measures exist; and some top management is feeling its ignorance and insisting on education and indoctrination such as that offered by the Univ. of California extension series and the Dept. of Defense Computer Institute. Clearly, the time for computer professionals to measure their own performance is at hand.

The essence of any measurement scheme is a series of written definitions. Fig. 1 shows a useful matrix for classifying computer operations by the type of environment in which the computer is installed and the response dictated by the application. The five categories are:

**I. Integrated Operations.** The computer is built into some larger system and must be considered as a component of this larger system. Thus, it must carefully mesh with its environment.

**II. Personal Computation.** The user of the computing equipment is present at the time the computer is producing the solution. The usual case involves the actual operation of the machinery by the person with the problem being solved.

**III. Computation Service.** The services of a computer are made available to many users, each for a short period of time. Usually the user prepares his run for the machine, submits it for computation, and returns later for the results. The machinery is run by professional operators.

**IV. Research and Development.** The computer is augmented by some special computer hardware or modifications to standard computer hardware for unique use in some research activity.

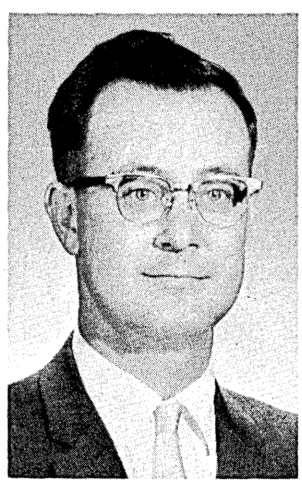
**V. MILSPEC.** Special purpose hardware required for the unique needs of the military or specialized versions of commercial equipment produced to meet more exacting requirements are especially packaged to make them amenable to more adverse environment not usually encountered in the commercial field.

While some of the above categories overlap, they are useful since a great majority of our computing installations fall into one and only one of these environments. In addition, another complementary set of classifications can be devised for computer applications. These depend upon the time allowed for solution once the solution is requested *and* the input data is available. The four classes are:

**A. Real-time.** Whenever the computer is associated with a dynamic, on-going process, and the subsequent control of that process is heavily (or totally) dependent upon the computer's output, the process is said to be real-time. Thus, an answer is required in the instant following the availability of the input data. In the usual case the computer is electrically connected into this system in such a manner that any unplanned delay in the computer solution results in adverse consequences, possibly of catastrophic proportions.

**B. Priority.** Frequently a computer is used in support of a project or test where the computer solution is required before the remainder of the work force can continue. In these cases the computer is not electrically connected to a process, but the input data are prepared by semi-automatic methods and presented for computer processing. If a priority run is significantly delayed, the consequences, while not catastrophic, are accompanied by an unusually high (relative) cost. Thus, it is desirable to complete priority runs as soon as possible in order to minimize the possibility that delays will cause excessive cost.

**C. Scheduled.** This type of application is characterized by a fixed completion time. A job may be deferred at the discretion of the facility manager as long as the completion schedule is fulfilled. Such allowable delays are usually of



*A computer consultant in Northridge, Calif., Mr. Patrick has been concerned with throughput and its measurement at both user and manufacturer levels, as well as with research firms and military development contractors. He was recently acting director of the newly-formed Defense Dept. Computer Institute during its initial months, and is editorial adviser to* **Datamation.**

the magnitude of four to 24 hours.

**D. Historical.** Runs of the historical type are required for some future reporting purpose and, in general, do not involve critical time constraints. Delays of 12 to 72 hours can usually be tolerated provided the final completion schedule is met.

As with the categories described earlier, the classes of applications may overlap. Furthermore, some computer installations may operate in more than one mode. For instance, they may operate in a real-time environment for a few critical hours out of the day while the remainder of the day is spent processing applications whose completion has been scheduled.

The classification chart in Fig. 1 is useful for framing discussions. SHARE members principally fall into classification IIIB (user absent, professional operator; priority applications). The commercial portion of the field occupies the majority of classifications IIIC and IIID (scheduled or historical; user absent, professional operator). The prolonged debate about time-sharing consists of the laudatory efforts of my research colleagues to transfer a technique which has been proven in the laboratory (classification IVB, priority, research and development) into the domain of the large scientific and engineering user, category IIIB.

then running through the shop. A 7090 and three 1401's were carrying the load. Two programming systems were in operation. Runs were batched. The mode of operation was tape to tape. A priority system existed which called for top priority jobs to be run as batches of one.

When the accounting records were scrutinized, it was found that 90% of the jobs (by count) ran less than 12 minutes, and 80% of the jobs (by count) ran less than six minutes. These jobs accounted for approximately two-thirds of the usage, while a few very important long jobs accounted for the other third of the machine time used.

As the study proceeded, it was found that efficiency was easier to talk about than it was to measure. The computer is an interesting device since it is either running or isn't. Idle time was rather easy to measure, but the efficiency of use when the computer was actually running was much more difficult to contemplate.

As the study progressed, management added two more constraints. These were:

5. Minimize the changes to existing programs.
6. Consider only announced existing hardware.

As the study attempted to understand present operations, the turnaround time problem was attacked first.* It was determined that turnaround time on the remaining work was extremely sensitive to the number of priority jobs run. Thus, in an emergency, the throughput of the facility (number of jobs per unit time) decreased markedly when faced with a heavy load of priority tasks. The solution to this seemed to be to replace the existing serial media used for storing the input stream (tape) with a cyclic media (disc) so that jobs could be rescheduled after they were submitted to the machine without loss of throughput. Fig. 2 shows the relation

Fig. 1

**COMPUTER APPLICATION CLASSIFICATION CHART**

| Category / Class | I Integrated Operations | II Personal Computation | III Computation Service | IV Research and Development | V MILSPEC |
|---|---|---|---|---|---|
| A — Real-time | | | | | |
| B — Priority | | | | | |
| C — Scheduled | | | | | |
| D — Historical | | | | | |

Given the classification chart shown in Fig. 1, the objectives of an installation may be determined. For example, the objective function for categories IIIC and IIID is minimum cost within the schedule constraint. These installations, and they are the great majority of the computers now installed in the United States, are keyed to the tempo of the business world. Turnaround time is not a strong consideration as long as the schedule can be met.

On the other hand, classification IIIB is an environment where the time required for solution may be as important as or more important than the cost for that solution. Over the past 18 months, I have been associated with a computer modernization effort for a class IIIB installation. This has overtones sufficiently interesting that it warrants some detailed discussion for the benefit of facility managers who are contemplating similar steps.

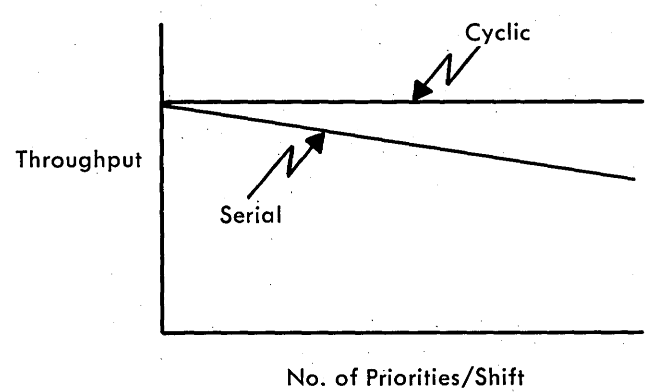
Due to a change in the physical source of the work, management convened a study group and charged them with the following four objectives:

1. Maintain the existing efficiency of the shop.
2. Achieve a console ability so we may serve remote users by wire line.
3. Do not increase the total cost of the facility.
4. Decrease the turnaround time for the remaining work.

The study crew set out to measure the work load

Fig. 2

**RELATION OF THROUGHPUT TO PRIORITY FOR TWO INPUT STORAGE MEDIA**




* Turnaround time has been the subject of many discussions in barrooms and the trade press. A marketing manager for a major computer manufacturer recently stated, "The programmers all talk about turnaround time, but management seems unwilling to pay to reduce it." In one specific case, I can attest to that statement.

of throughput to number of priorities per shift for the two types of storage media.

Being bound by a total cost constraint, we had to release existing equipment in order to obtain new devices. As the situation developed, it appeared that replacing the three peripheral processors with one larger machine would reduce the cost and concentrate our processing capacity in one mainframe where we could exploit it more fully. Several configurations were given preliminary inspection: shared discs, private discs, both disc and drum, shared drum, and electronic coupling of the two processors. During these studies, it became painfully apparent that the performance of the configurations being studied was *extremely* sensitive to the volume of input/output being handled, the periodicity of this input/output, and the volume of programming systems required to prepare a job for execution.

### the hardware monitor

Just as our need for detailed data became acute, the manufacturer produced a measuring instrument and offered to assist in its installation on the computer. This device is called a "hardware monitor." It connects to a 7090 so that channel usage, channel overlap, I/O device usage and unadulterated CPU processing time (no I/O in operation) can be tallied. An experiment was designed so that the load on the batch operated shop could be accurately described and measured. The hardware monitor was obtained, a crystal controlled millisecond clock was procured, and a computer controlled camera was set up to automatically record the data without stopping the computer. Over a four-day period, 500 jobs were run through the instrumented machine and 3,500 film frames were exposed. Data was gathered by job phase (compile, assembly, loading, and execution). It was then aggregated for the final report.

It was found that 90% of the available time was being profitably used, and that only 10% idle was accumulated over the course of the four day test. It was found that many hours of the day were spent in the presence of a significant backlog so that the shop operated as a computer-limited installation. In this case, processing efficiency determines turnaround time. When the data from this experiment, together with volumes of cards read, lines printed, etc. for the same jobs were made available to the study team, the various competing configurations could be evaluated for both cost and performance. It was found that an average job could be characterized and, based on these fundamental measurements, its execution time could be broken down into the following functions: system I/O, primary input, useful computing, unit record, printing in process, final printing, programmer I/O - buffered, programmer I/O-unbuffered.

Having actual measures for the above functions for the average job, we could then evaluate configurations by predicting the performance of the average job on the contending configuration. Once this had been achieved, we could divide by rental to obtain a measure of cost effectiveness.

The configuration finally chosen was a 7094 coupled to a 7040. A 1301 disc supplied the file storage. Ten magnetic tapes supplied historical input and output ability, one card reader carried the input load, and three printers processed the output. Two 1014 remote input stations were provided for operator communication. It was decided that this configuration should be installed and operated successfully before the remote users were served.

The software required was significant. Performance was hard to achieve. In one recently measured case, a job which we thought typical required 122,000 words of system program to be transmitted between the two machines before execution could proceed. It appears that near optimum scheduling of the disc arm and allocation of space is required to approximate the efficiency of a free-standing batch operated machine.

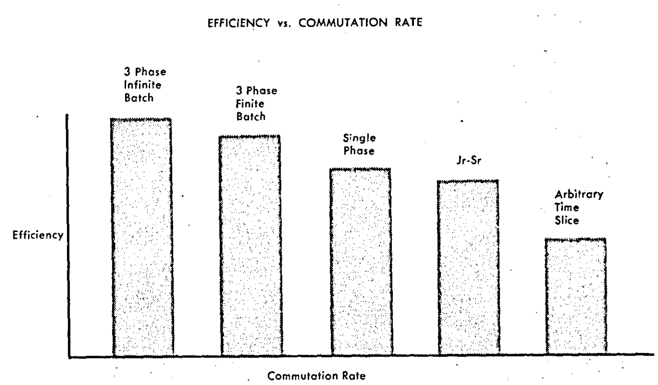
### improving the disc scheduling

At one time, the disc I/O was handled in less than optimum fashion and a collection of nine benchmark jobs ran in 15% longer time than on the batch operated equipment. Since the shop is compute limited, this loss of machine efficiency was directly reflected in longer turnaround time and a higher cost per solution. A rescue effort was immediately launched to improve the disc scheduling and regain the lost performance increment.

Since so many installations are now contemplating equipment modernization programs, a few conclusions are in order. Without clearly stated objectives, the modernization effort will be haphazard and opinionated. The clear statement of management's objectives allowed the study to proceed logically. Intellectual arguments were terminated rationally. In the case illustrated above, the contemplated configurations caused major adjustments in the operational flavor of the shop, the distribution of functions between the two processors, the automation of functions which had heretofore been manual, and the response of the shop to change. Actual measurements of the load, broken down into its primary functions, were invaluable in achieving a configuration whose cost/performance matched the desired goals.

One measure of efficiency is time/job minus overhead/job both divided by time/job. Overhead is primarily non-productive system time. The frequency of system access can be expressed as a commutation rate. Investigations to date seem to indicate that we can tentatively class operational modes on an efficiency scale as a function of commutation rate as follows: three-phase with an infinite batch size, three-phase with a reasonable batch size (20 jobs), one-phase, the junior/senior configuration described above, and time-sharing with arbitrary time slices. Fig. 3 shows this empirical plot.

Fig. 3




The three-phase mode leads the list since the systems are referenced only once and each job proceeds to its final conclusion. The three-phase with reasonable batch size is less efficient (but gives better turnaround). The systems are referenced once per batch. The one-phase system has many emotional attractions but the system must be fetched following each job which requires the major portion of the available primary memory. In the case of multi-faceted jobs and current operating systems, the full system is fetched each time and some portions several times within a job (122,000 words worth!). The junior/senior configuration is slightly less efficient than the properly programmed single-phase configuration since the senior processor must communicate its desires to the junior, and unscheduled requests frequently cause the senior processor to wait for the junior processor to respond. Time-sharing, and particularly the arbitrary time slice method of operation, seems to be particularly inefficient since the computer program in residence in primary memory is arbitrarily interrupted, placed in secondary store, and replaced by the next program awaiting operation whenever the time period is exhausted. The time spent in this activity is, of course, unproductive overhead to the system and must be retrieved in some other way. I have been told that this mode of operation results in a greatly reduced number of runs per checked out solution, although I do not understand how this can be measured.

In conclusion, it looks as if the batch mode will be with us for some years to come due to the efficiency it offers. However, if some method can be devised for offering better turnaround to those who require it while exploiting the "tolerance for delay" of the other users to gain efficiency, then the console will creep into general use for classification IIIB. It is encouraging that some of this work is accompanied by built-in measurement schemes.* Based on the information we measure and gather, management may then choose a combination of turnaround and efficiency commensurate with the dollars they are willing to pay. ■

* Rosenberg, Arthur M., "Computer-Usage Accounting for Generalized Time-Sharing Systems," *Communications of the ACM*, Vol. 7, No. 5, May 1964, pp 304-308.

**evaluation techniques**

# MEASUREMENT OF SOFTWARE CHARACTERISTICS

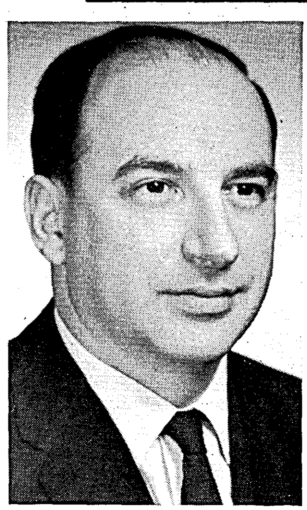
by ASCHER OPLER

Electronic data processing machines are sold on the basis of their total performance. Since manufacturer-distributed automatic programming and operating systems (software) are used almost exclusively today, the user should evaluate a combined hardware-software system. The user or prospective user of computing equipment must carefully consider the *combined* effect before doing any of the following:

1. Make comparisons among several prospective computers.
2. Consider the effect of a replacement computer.
3. Consider the effect of replacement software for an existing computer.
4. Plan an acceptance test for the total system delivered by the manufacturer.
5. Estimate running time for production programs written in a source language such as COBOL or FORTRAN.
6. Estimate the time required for compilations, assemblies and checkout runs.

From the viewpoint of the supplier of both the hardware and software (the computer manufacturer), the problems are related. He requires early and highly specific knowledge of the total performance of his system. This knowledge must meet a variety of needs: marketing plan, sales support, proposal preparation, performance specifications, etc.

Independent software producers who prepare their products under contract to manufacturers face problems that, within their own frame of reference, are even more



***Mr. Opler is vice president and director of Programming Systems for Computer Usage Co. Inc., New York City. He has developed numerous computer applications and automatic programming systems since entering the field in 1947 with Dow Chemical Co., and presently is a member of the ACM Council representing the Greater New York region.***

serious. Frequently, they are called upon to design systems at a very early stage in order to permit delivery of software with the first hardware deliveries. (The same problem, of course, exists within the manufacturer's own software division). Since the buyer, in this case the manufacturer, has set objectives for the software performance, the independent contractor must be able to plan his effort so that, at an early date, quantitative specifications for the finished software can be set, and when the package is delivered, these specifications must be met. Associated problems of quality control and acceptance testing are of growing concern to both independent suppliers and manufacturers.

Incredibly, specification and measurement of software are seldom performed. A whole company, a whole industry or even a whole military operation may be completely dependent on their computing equipment. They may have carefully specified the hardware performance, measured it and found it acceptable. Nevertheless, the entire operation may fail because of poor software. The contrast between the effort spent on hardware specifications and evaluation compared with that spent on software measurement and evaluation is only too evident.

Specifications of performance are readily available for all computing equipment. It is obvious that specifications would also be desirable for software. However, as is well known, they are not generally available, and even when they are available, they are very difficult to measure objectively when quantitative, and nearly impossible to evaluate when qualitative.

Before considering techniques for evaluation, a good understanding of the factors that affect performance is required. Three of the more significant are discussed below.

## influence of design objectives

Most digital computers are described as "general purpose," but we know that each has certain application areas (commercial, scientific, real-time, communications, information retrieval, etc.) in which it performs better than others. Similarly, software is never really "general purpose." This should be recognized and taken into consideration when testing and evaluating. While few would use scientific computations to evaluate a computer that is primarily designed for commercial applications, we commonly find software evaluated while performing jobs ill suited to its design.

In RAND Report #RM-3447-PR, Haverty and Patrick discuss the *design point* of a software system. This is the list of design objectives ranked in order of importance. Currently, for example, the design point for a COBOL compiler may place compilation speed third, following object program execution speed and object program compactness. System integration and lucidity of printouts may be fourth and fifth. and so on.

Furthermore, there are many characteristics built into a software system that seriously affect its performance. Consider, for example, the size of the source program that a compiler might process. Some compilers are designed for short programs, others for "normal" length source programs . . . while some do not function well unless the source program is quite long.

Once one becomes aware of the precedence of characteristics built into the software by its designer, a new light is thrown on evaluation. A reasonable rule is that *the further a system is measured from its design point, the less reliably does that measurement reflect the abilities of that software system.*

From the viewpoint of the evaluation, some major questions involving design point are:

(a) What is the planned design point?
(b) How will this affect testing?
(c) Should the system receive higher rating if it meets the *producer's* design objectives or if it meets the *user's* needs?

## influence of the age of a software system

The process of announcing, producing and maintaining software and, in many cases, of similar actions for second-generation systems, has led the user to view "the life cycle of a programming system."

*Stage*

| | |
|---|---|
| I | Pre-delivery; manuals prepared and distributed. |
| II | First system delivery. |
| III | System operational; early difficulties with reliability and with deviation from specifications. |
| IV | Reliability attained and performance improved. |
| V | System heavily used and well maintained. Successor software announced. |
| VI | System obsoleted by delivery of announced successor. |
| VII | System maintenance discontinued. |

Before evaluating performance, the system should have reached Stage IV. However, factors not involved directly with performance (e.g., diagnostic capability) could be measured earlier. Most current systems produce a system status code or date as part of the output listing. This should be carefully noted as part of the documentation of testing in order to pinpoint the stage at which the test was performed.

## influence of computer size and modularity

One problem in the development of software systems concerns the basic equipment configuration and its variations. In recent years, manufacturers have offered a wide selection of modules to the potential customer. One may order one to $n$ modules of core storage, one to $m$ groups of index registers, one to $p$ magnetic tape units, plus a wide selection of auxiliary storage devices, input/output equipment, etc.

As the number of available configurations of a specific computer increases, the problem of developing matching software becomes quite complex. Ideally, the manufacturer should be asked to supply a software system that will work in an optimum fashion on each configuration of equipment delivered. However, as a matter of economics, the manufacturer generally produces a *basic* software system operable on a *minimum* machine configuration. Such systems are constructed so that an advantage (usually limited) can be taken of increased equipment. Thus, a given compiler may compile faster, may handle larger programs and may optimize better when run on a configuration larger than the basic one.

When comparing software, considerable weight should be given to the *basic* configuration. Thus, for example, a compiler designed to run on a 4K magnetic core computer with four tapes may be tested on a 32K model of the same equipment using eight tapes. However, when comparing the expanded compiler to one originally designed for 32K core and eight tape units, due allowance should be made for the relative sizes of the *basic* configuration.

## testing against absolute standards

While most interest in software evaluation has centered on relative performance of software systems, it is important to remember that certain absolute standards can be used for test purposes. A reference to a list of measurable characteristics (e.g., Fig. 1 for compilers) indicates that several of these are susceptible to absolute measurement.

Algebraic compilers call mathematical subroutines, and many use subroutines for execution of floating point operations. The mathematical precision attained in executing an object program can be determined objectively. This can be done simply by hand-computing the same values to more significant figures than is provided by the object program system. A more comprehensive method of test involves the compilation of expressions whose actual value is completely known. A typical example prints $\sin^2 x + \cos^2 x$ for various values.

Fig. 1 Some Characteristics of Compilers that are Subject to Evaluation

**Time of Compilation**
**Time of Execution of a Compiled Program**
**Memory Space Requirement for a Compiled Program**
**Accuracy of Conversions and Function Evaluations**
**Error Diagnostic Ability of the Compiler**
**Error Detection Ability of Compiled Program**
**Ease of Compiler Maintenance and Modification**
**Quality of Compiler System Documentation**
**Clarity of Messages, Listings and Memory Maps**
**Extent of Errors within the Compiler**
**Degree of Compliance with Source Language Specification**

A test of the consistency of conversion on input/output versus internal compiler conversion can be made quite readily. It is only too true that many compilers will not branch on zero when the value 6.4 is input and subtracted from the value 6.4 as compiled (see Fig. 2).

Another absolute test may be made wherever the manufacturer has provided formulae or statements concerning the speed of his compiler. The statements or formulae should explicitly designate the equipment configuration that attains the specified speeds.

The ability of a compiler to detect errors in the syntax and logic of the source program can be measured by feeding to it carefully prepared source programs which contain specifically contrived errors.

Fig. 2 A Test for Consistency of Conversion in FORTRAN Systems

```
        READ    3, X
   3    FORMAT  (F2.1)
        IF (X - 6.4) 1, 2, 1
   1    STOP  1
C         STOP  1  INDICATES
            INCONSISTENT  CONVERSION
   2    STOP  2
C         STOP  2  INDICATES
            CONSISTENT  CONVERSION
        END
```

**One data card punched 64 in columns 1 and 2 placed in the card reader feed hopper**

The ability of the object program to detect errors developed at execution time can also be tested. Such errors include division by zero, overflow and underflow, a negative argument to a square root routine, arguments greater than 1.0 to arcsine or arccosine routines, etc.

Beyond these that are readily measurable lie others which require more subjective study. Among such characteristics are ease of maintenance, quality of system documentation (obviously related to the preceding), clarity of messages, maps and listings, etc.

The reliability – that is, freedom from compiler bugs – is difficult to measure in Stage IV or beyond. There is usually an exponential decay of such errors. For a major compiler, some of these remain undiscovered for years despite heavy use of the system.

A difficult task is that of determining compliance with a language specification. In a sense, there are two comparisons that can be made:

1. Does the language manual for *this* system agree with the accepted definition of the language?
2. Does the behavior of the compiler itself correspond to the language description in its own manual?

Numerous cases of both types of language deviation are known. Deviations include extraneous statements, nonimplemented language aspects and distortions of the language or its intent. Only the first type of deviation can be regarded as acceptable.

### testing relative performance

In the highly competitive marketing situations that exist today, it is not surprising that a considerable amount of effort is being spent on preparing relative evaluations of software systems. Many report results which should be regarded with skepticism since it is all too easy "with evil intent" to select programs or conditions which enable one's own processor to appear in a favorable light. In fact, if one is familiar with the design details of two systems, it is possible to contrive a test problem which will make A outperform B and another test problem which will make B outperform A. In the following, an attempt is made to lay some ground rules for reasonable tests of relative software performance.

### selection of test programs

For purposes of software evaluation, it is necessary to select a number of test programs. Many evaluations reported in the literature are, in my opinion, not too meaningful . . . primarily because of poor selection of test programs. Currently five "benchmark programs" are often used for testing. These are frequently selected from "typical" applications written and used at large installations. This number of programs is grossly inadequate for an accurate measurement of the relative performance of two systems. Firstly, since most compilers are sensitive to the length of the source program, it is necessary that the program selected include a series of graded length ranging from very short to very long. Secondly, since the total time required for compilation includes the time required to process each statement, the mix of statements within a program will greatly influence the performance. Consequently, accurate measurement requires a large sample of statements. However, since the logical and sequential relations within a single program also greatly affect compilation time, it is necessary to have the statistically large sample of statements embedded within a variety of different program environments. Thirdly, since the measurement of object program execution speed is part of any well-conducted test, it is necessary to compile programs that not only are executable but for which adequate input data is available. Since source programs which process input/output data are often highly dependent on the manner in which the compiler handles buffering, it is essential that a battery of test programs include programs with a wide range of input/output equipment usage and data activity.

A collection of test programs should include not only actual application programs but a set of contrived programs designed to systematically explore some of the software characteristics. Most manufacturers now have available batteries of systematic tests. These are somewhat analogous to the programs used by manufacturing control and field service personnel in working with hardware.

A facet of test program usage that is often ignored is the nature of test data used in conjunction with compiled object programs. Much of what has been said in the preceding paragraphs of proper selection of test programs applies to the proper selection of test data. There should be sufficient volume for accurate timing, and all of the special conditions should be included with the data including invalid punches, unreadable tape records, multi-reel files, incorrect labels, etc.

One characteristic of test data files that leads to results that are difficult to interpret is the relation between the data structure and the logical design of the hardware. Suppose, for one computer, test data has been carefully organized into multiples of six characters, and the corresponding data description to be compiled also describes records and items in terms of six-character groupings. Then consider the effect of compiling the program and executing it on a machine on which the word size is oriented toward four or five-character fields. The difference in performance will be dramatic even though the second compiler might be vastly superior to the first. Thus, in selecting test data, one must be careful to avoid choosing data that is strongly biased in favor of one software-hardware combination.

## other significant considerations

As the total computing systems design continues to evolve, the problem of software evaluation becomes more and more complex. At one time we considered a single computer with a single, relatively simple input/output scheme using a single type of intermediate storage (namely magnetic tape). In the environment of 1964 and, even more strongly in the near future, we will be considering software systems for hardware configurations that are vastly more sophisticated and which permit great latitude in the functional assignment of their elements.

To be more specific, computers will use more and more multi-programming and multi-processing. They will be processing programs fed directly from remote locations. Furthermore, data will be processed on-site with continual interruption for acceptance of on-line, real-time inputs. Where once the intermediate storage medium was magnetic tape, it may well be a systems option as to whether a given function is carried out by magnetic tape, magnetic disc, magnetic drum or any other device. Where the output once appeared on a console typewriter or a line printer, it may now be directed to cathode ray displays, remote terminals or to control mechanisms for various devices.

All of this development means that the process of performing relative evaluations is becoming more and more elusive. How does one compare a system which is simultaneously compiling and executing against a system which can only compile or execute? How does one compare a system in which compilation must be continually interrupted to allow acceptance and validation of remotely generated inputs with one that does not allow such interruptions? It appears that such evaluations will require a detailed analysis of the microsecond-by-microsecond activity and its distribution into functional periods.

## separation of software characteristics from hardware characteristics

Fortunately, for those interested in the measurement of software characteristics, second and third generation compilers are now in existence. Thus, for the machines listed in Fig. 3, two or more compilers are available that accept the same (or similar) source programs. This allows one to keep the hardware constant, to keep the source program constant, and to substitute compilers. If we recompile a 5.0 minute source program with the identical computer and found that the time was reduced to 2.0 minutes, we now see one approach to quantitative software measurement.

**Fig. 3 Partial List of Language-Computer Pairs for Which More Than One Compiler Exists.**

| | | |
|---|---|---|
| IBM | 1410 | FORTRAN |
| IBM | 7030 | FORTRAN |
| IBM | 7090 | FORTRAN |
| Honeywell | 800 | FORTRAN |
| CDC | 1604 | FORTRAN |
| IBM | 1410 | COBOL |

In relative evaluation of two software systems performed on two different hardware systems, we recognize many difficulties. Many of these disappear when the compilation is performed on the identical equipment. However, the tester should be reminded about the significance of the design point difference. Characteristically, successor compilers are written with a different design point than the predecessors. It is rare, indeed, for a compiler to be rewritten with similar design goals for the same machine. Only when this is applicable can one really compare two software systems on an absolute basis.

Thus far this discussion of software measurement and evaluation has centered on compilers. This emphasis has been made for several reasons. Most of the evaluation that has been performed and reported has been on compilers, and interest in their relative performance continues high. The emphasis on compilers, however, indicates a general absence of knowledge of the testing of monitor-operating systems, checkout systems and advanced loaders.

Much of what has been said about compilers applies to assembly programs. The evaluation of competing assembly programs for the same or similar equipment requires careful consideration of the design goals for each assembly program.

One software area for which evaluation has become a science rather than an art is that of sorting and merging. Here, one can test software and hardware combinations and measure the time required to sort a given file. Since sorting time is usually the only significant goal (most other considerations revolve around facility for manipulating records as they are being sorted), it is much simpler to evaluate. The developers of these programs generally publish formulae or tables describing the exact performance to be expected. Evaluators should, of course, verify the accuracy of the formulae with actual machine tests.

The acceptance of tables and/or formulae for specifying performance for sorts and merges suggests one useful direction for the producers of compilers. In at least one case the manufacturer has supplied formulae which may be regarded as a specification of the compiling system speed.

Currently, specifying and evaluating the performance of software systems is a rarely practiced art. Let us hope that in the future it becomes a commonly practiced science. ■

# THE NEW PROGRAMMING LANGUAGE

**by IBM out of SHARE**

by DANIEL D. McCRACKEN

[Ed. note: Advanced programmers generally take a healthy interest in any new language. But when that new programming language is being considered for implementation on the largest family of computers by the largest manufacturer in the industry, it becomes highly significant to *all* users, whatever their brand of machine. For, marketing considerations may require other manufacturers to include it in their software package. It is that important. The language, like FORTRAN, has become an issue. This article is perhaps the first public shot in a battle which will probably produce many heated arguments — and some casualties — over the next several years. The author, a recognized programming language authority, believes the course of events may well be shaped by what transpires in the next several months. It is hoped that this frank, provocative analysis and appraisal will help stimulate thoughtful and articulate discussion of the new language. For now, the floor is Mr. McCracken's].

Some people like fights and some don't. Those of us who do have been restless lately: an unaccustomed calm had settled over the programming languages battleground, with the sluggish acceptance of FORTRAN-IV the hottest of some lukewarm issues.

With the announcement this spring of the joint IBM/SHARE New Programming Language (NPL*), all this has changed. One suspects that the battles over ALGOL and COBOL will in time be regarded as mere warm-ups for this one.

In the opinion of this observer, the language is an excellent one, deserving of wide support. Even if everyone were to agree, which will not happen, there would still be economic and emotional considerations sufficient to sustain a struggle lasting, I predict, at least five years.

## the history in brief

NPL is the product of a group called the SHARE Advanced Language Development Committee. There were six members, which according to one of Parkinson's laws is small enough to get some work done. The three SHARE representatives were Hans Berg of Lockheed-Burbank, James Cox of Union Carbide, and Bruce Rosenblatt of Standard Oil of California. The IBMer's were George Radin, C. W. Medlock, and Bernice Weitzenhoffer. (It might be worth noting that these six have an average of 10 years' experience in a broad range of programming applications, and that the last two named have been heavily involved in compiler writing for the past several years).

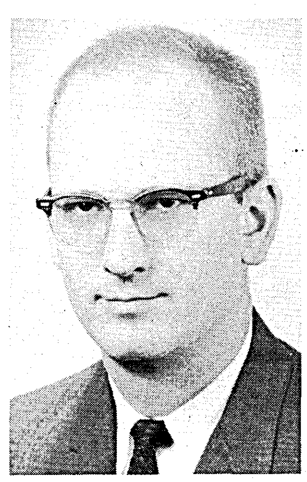
The committee began its work in October of 1963, following a SHARE meeting in September at which officials of SHARE and IBM agreed that a joint effort to develop a new language would be a fine thing. There seems to be little substance to the more or less automatic accusations according to which both parties are supposed to have led the other by the nose. It was apparently a joint effort, jointly initiated. Meetings were held once or twice a month from October of last year through this May, after which the committee disbanded.

The first report shown to anyone but the family was dated March 1. This document was revised to remove certain ambiguities and inconsistencies and to make the language more general. It was expected that a "final" report would be ready July 1. This was to be available to the public, so to speak, although it is not planned to publish it in a journal. (It would be a bit long).

## what's it like?

At first glance, NPL gives the appearance of an extension of FORTRAN in the direction of ALGOL, with a few commercial data processing features thrown in. Closer inspection discloses a language that, although extensive, is unified and carefully designed.

Compared with FORTRAN, NPL is both simpler and more powerful, depending on the point of view of the



*Mr. McCracken is a New York-based consultant who spends most of his time writing. He is the author/co-author of eight books on programming, the latest being "Numerical Methods and FORTRAN Programming," written with W. S. Dorn of IBM, and published just last month. McCracken holds degrees in mathematics and chemistry from Central Washington State College.*

* FORTRAN-VI? FORALBOL? ALGBOLTRAN? FORBOL, so an advanced version can be called EIGHTBOL? There is no general agreement what the name is, but the developers say there will be a definite one later. We shall call it NPL here.

# THE FEATURES OF NPL

*The following listing is not exhaustive, but it may serve to give a general idea of the features of the language. It is assumed that the reader knows FORTRAN. The listing begins with changes from FORTRAN, shades into FORTRAN extensions, and ends with items more or less related to COBOL.*

- *The format of writing statements has been relaxed considerably. Statement labels are identifiable by the presence of a period after the last character, so labels may begin anywhere. Each statement must be terminated by a semicolon, so that there is no assumed length for a line, and the continuation card concept has been dropped. A program now consists of a linear string of characters, rather independent of the input medium. (This is actually a very significant change, permitting, for instance, implementation with remote terminals).*
- *Labels (FORTRAN statement numbers) are now formed by the same rules as for variables; they cannot be pure numeric. Embedded periods are permitted, making for names like SALESMAN.RESULT, if one wishes.*
- *All words must be separated by delimiters, such as blanks or arithmetic operation symbols. This is no real inconvenience to the source programmer, but makes a tremendous difference to the compiler writer. No longer will there be cases like the FORTRAN statement*
  *DO 123 I = J12345, K12345*
  *in which it cannot be established whether this is a DO or a simple replacement until reaching the comma.*
- *Comments may be written anywhere. This change was dictated, in part, by the dropping of the assumption of card input. With paper tape or remote typewriters, there is no such thing as "column 1."*
- *In the absence of declarations (declaration by default), variables whose names begin with I, J, K, L, M, or N are assumed to be fixed and all others floating. Using suitable declarations, those naming conventions can be overridden and a variable can be defined to the complex, logical, character, or bit. The character and bit operations are available for work with packed records in business data processing, operations on partial-word fields in command and control or other real-time applications, or any other purpose. There is no separate double-precision variable as in FORTRAN-IV, but a SIZE clause is available for this and other purposes.*
- *There is a compound statement: a group of statements enclosed between the "statement brackets" DO and END is treated for control purposes as one statement.*
- *Mixed mode arithmetic is permitted.*
- *A subscript may be any arithmetic expression.*
- *The logical IF statement has been expanded to include an ELSE path, so that the programmer can specify the actions desired if the logical expression is true and false.*
- *The DO statement is much expanded. The controlled variable is not restricted to fixed point; the beginning, ending, and incrementing values may be any legal expressions; there is a WHILE option that keeps repeating as long as a logical expression is true.*
- *Format information may be omitted. On input, data items are assumed to be separated by blanks or spaces; on output, a standard format is used and values are identified by name.*
- *The FORTRAN arithmetic statement function has been dropped. This is no real loss to the programmer, since a suitable procedure can do all the same things. (This saves much trouble for the implementor).*
- *Certain operations are defined for arrays. For instance, if all variable names are arrays, then A = B + C*

---

user. A subset can readily be defined that is considerably easier to learn than simple versions of FORTRAN. Really. There are fewer restrictions and fewer rules to learn. Mind you, this subset would require only a cut-down manual, not a cut-down compiler; the naive user simply doesn't know that there is any more, and what he doesn't know doesn't hurt him. (That last isn't trivial; not all languages work that way).

At the other end of the scale, NPL offers dozens of features simply not present in FORTRAN. These range from an ELSE path on the IF statement, through alternate entry and return point for procedures, to provision for asynchronous operation of mass storage devices. Some of these features are summarized in the limited space above.

A fundamental design philosophy comes into play when we consider the extreme range from simplest to most sophisticated. The basic approach is this: there is a "standard" way to do virtually everything; if the programmer likes this standard way, or doesn't even know there is anything else, he simply says nothing. He has picked the standard method "by default." On the other hand, if the standard way isn't what he wants, the full power of the language is there for the asking.

For instance, the DECLARE verb is available to give a very complete description of variables—yet it is not required that a program even contain a DECLARE. Without it, the IJKLMN convention distinguishes between fixed and floating point, standard precision is assumed, and that's that. The programmer who wishes can change lengths, specify any naming convention he likes, switch to partial-word bit or character fields, or give a picture of the desired edited result.

For another example, if the programmer is satisfied with the standard input and output media—whatever they are in a particular installation—and if he can make do with the standard format-free style, he can write an input or output command that consists of nothing but the word READ or WRITE and a list of variables. (And sometimes you don't even need the list)! Yet the programmer who is deeply concerned with I/O problems, as most professional programmers of course *calls for element-by-element addition and replacement. This does not mean that there is a complete matrix code, but some matrix operations are much simplified.*

- *Dynamic storage allocation for arrays is permitted, i. e., the dimension of an array may be stated in terms of variables, as long as the variables have previously been given values. Functions are available so that a procedure using variable-dimension arrays can determine their actual current sizes.*
- *A powerful overlay capability is provided through an extensive storage classification scheme that gives the programmer explicit control over the local/global nature of variables.*
- *Procedures may contain other procedures, to any depth. (In FORTRAN, a procedure may call another procedure, but not contain another procedure). This feature becomes the conceptual framework of program structure. For instance, if procedure A contains procedures B and C, then any variable declared in procedure A is "known" in B and C. No COMMON statement is required. The basic ideas of program organization follow the best of ALGOL.*
- *A procedure may have multiple entry points, it may return to different points in the program that called it; a procedure may call itself, if designated as recursive.*
- *The COBOL and Commercial Translator PICTURE clause has been included, somewhat expanded. (Symbolic of a shift in something or other, one writes PIC rather than PICTURE)!*
- *It is possible to define, through a suitable DECLARE statement, a structure. This is simply a collection of data; unlike an array, the parts of a structure are not required to be all of the same type or have any other common characteristics. In practical terms, a structure is like a COBOL record, and can be used for the same purposes.*
- *The COBOL level structure is here, although specified somewhat differently. Movement of groups of fields is therefore possible, as is qualification of data names, and there is an equivalent of the MOVE CORRESPONDING verb in COBOL. Name qualification is indicated by writing the most inclusive level name first, and separating names by dollar signs. Thus, if QUANTITY is a name in two records named MASTER and TRANS, we can indicate the one we want by writing MASTER$QUANTITY or TRANS$QUANTITY.*
- *Editing procedures give the programmer very complete control—if he wants it, as always—over the manner and time of data format conversions. The contrast is with COBOL, where the corresponding conversions are definable only in terms of the characteristics of sending and receiving areas. For instance, if one wishes to read a tape record into core just as it appears on tape, test the key, and only then decide whether to convert the full record. COBOL requires the use of two record areas. The first would be described in terms of the tape format; the second in terms of the converted format. The actual conversion can only be called for by specifying data movement.*

  *You can do it that way in NPL, too, if you want to, but here you have other choices. In a nutshell, various verbs are provided that give the programmer direct control over the conversions. If the external format must eventually be completely modified before processing, the programmer can call for any part of the transformation, from none to all, to be done as part of the READ. Analogous considerations apply on output.*
- *There is provision for handling dynamic, unscheduled conditions like machine overflows and end-of-file signals. Various features make it possible to write programs for a multi-processor, concurrent calculation system. Subroutines may be asynchronous; a program can be suspended to wait for parts of a calculation to finish. The implications for large real-time applications are obvious.*

---

are, will find facilities of much greater power and flexibility than in FORTRAN.

Quite a number of features have been incorporated that are recognizable as derived from ALGOL: compound statements, a much more powerful DO, an explicit distinction between local and global variables, the best parts of the ALGOL block structure, and, optionally, recursive procedures, among other things. These features simplify many complex programming operations that in FORTRAN require complicated or tricky maneuvers to accomplish. They are not especially difficult to learn, and the person who doesn't need them doesn't have to learn them.

NPL shows less resemblance to COBOL than to FORTRAN or ALGOL. Some of the differences between NPL and COBOL, however, are more superficial than substantial. An NPL program will tend to be shorter and less discursive than if written in COBOL. This is a backing-off from the English-language aspect of COBOL, which in my opinion is coming to be regarded as one of the less important advantages of COBOL. NPL appears to give the programmer direct access to operations that in COBOL are implied by data division entries.

On the other hand, some COBOL operations and terminology are carried over quite directly: OPEN, CLOSE, READ, WRITE, ROUNDED. Partial-word operations are permitted. The equivalent of MOVE CORRESPONDING is available. Data names can be of any length, an embedded period substitutes for the COBOL hyphen, and name qualification is permitted. If one wishes to emphasize the similarities, it is possible to write a program that at first glance has a distinct COBOL look about it.

There is no denying, however, that in searching for a single language to handle most programming, the NPL committee chose more FORTRAN and ALGOL features than COBOL. This will please some, displease others. I commend the committee for having had the courage to pick the one way they thought best, rather than taking the easier way out and including every suggestion as an option.

Some of the features of NPL are not found in present versions of FORTRAN, ALGOL, or COBOL. Perhaps the most important is asynchronous operation of input/output (and in some cases procedures). Where the equipment permits it, the word COMPLETE can be appended to a statement. This instructs the object machine to proceed independently; when the operation has been completed, a logical variable that had been false is set to true. A simple means of testing a set of logical variables is provided.

The new statement format, which regards a program as a linear string of characters without such concepts as continuation cards or margins, makes it much easier to use the language from remote time-shared terminals. These may very well become the dominant mode of operation within a few years. The advantages of the string-of-characters view of a program are easily important enough to be worth the trouble of learning to write the semicolons at the ends of statements.

Certain rudimentary string-manipulating capabilities are provided. NPL will not be mistaken for a list-processor, but certain operations are possible. These will be of direct value in commercial data processing, for instance, in connection with variable-length records.

### will it make it?

What may be expected to happen to the language? Right now it is committed for only the IBM 360. Whether it becomes available for other machines—IBM and otherwise—depends on many factors. One, obviously, is the judgment of the industry on the merits of the language; clearly, some observers are going to be less enthusiastic than this one. Those heard from so far use words ranging from "abortion" and "hodge-podge" to "the best yet" and "finally, the universal language."

One class of comments seems to raise the horrifying possibility of another ALGOL-type fiasco in which the language is viciously attacked by people who obviously don't understand it, since the defining reports are not intended as popularizations. The authors of the NPL report go to some pains to emphasize that their current document is designed as a guide to implementors, not as a primer. This being agreed, it is devoutly to be hoped that the primer now in preparation turns out to be a good one and becomes available soon.

Another problem, of somewhat more substance, is the question of whether NPL is sufficiently better than FORTRAN or COBOL to be worth the cost of conversion. I would be inclined to believe that the answer depends on the time scale: over the first six months or a year, probably not. In a longer span, very possibly so, especially if the universal language aspect pans out at all.

On this score, an insidious rumor has it that 360 users will not be offered a fair choice because the best efforts will go into the NPL compiler whereas FORTRAN and COBOL will get perfunctory implementations. This is instantly denied by IBM. Still, those in charge of software at IBM would be less than human if they did not base their priorities to some extent on what they hear their customers saying about NPL over the next few months.

As a matter of fact, one gets the impression that although the battle will almost certainly rage for years, the scales will be tipped in one direction or the other in the next six months or so. The reasoning goes like this.

If IBM customers, particularly SHARE, should show marked enthusiasm for the language, more support would be thrown behind the 360 implementation, presumably leading to a better compiler. Since the zilchiness of the first compiler has a large effect on the field reaction to a language, a bandwagon effect could start. Furthermore, if the customers were vocal enough, IBM would probably implement NPL for the 7094, perhaps beginning fairly soon. Once a significant number of IBM customers were using NPL, say in two or three years or so, other manufacturers would find themselves forced to implement it by marketing considerations, just as they were forced to implement FORTRAN whether they really liked it or not.

If all this happened, NPL could conceivably become the dominant language within three to five years.

If, on the contrary, the general reaction to NPL in the next six months or a year should turn out to be strongly negative, or simply apathetic, it would be reasonable to expect that IBM would, with some disappointment, implement NPL for the 360 as promised but not really push it. IBM cannot push too much harder than the customers want to be pushed; the company has shown recognition of this economic fact of life in the past. Finally, it is probably true that if IBM doesn't push NPL, nobody else will. In short, an unenthusiastic response in the coming months could spell the eventual demise of NPL, although the funeral might not be held for many years.

Which way will the scales tip? The answer is exceedingly difficult to predict because of the large number of considerations. Is the language really any good? Can it be implemented to give fast compilation of efficient programs? (The developers say yes, but users are going to make decisions before the claim can be demonstrated by a running compiler). Even granting NPL's superiority, is it better to throw out the old languages completely, or modify them in a more gradual fashion? Is the easy subset claim valid? How costly would the retraining be for FORTRAN programmers? For COBOL programmers? Will the battle shape up so that a graceful transfer of allegiance is possible, or will the lines be so rigidly established that changing one's position causes trauma and total loss of face? (Don't laugh; it's been that way before).

My extremely tentative guess: after great disputation and some modification, NPL will gradually be adopted by scientific users of IBM equipment. Acceptance by commercial users and by non-IBMer's will be exceedingly slow, but it will come.

# EXAMPLES OF NPL CAPABILITIES

It can be pretty difficult to get any clear idea of what a language is like just by reading a defining report, especially if that report was necessarily slanted toward compiler writers. The following examples will hopefully give a little of the flavor of NPL. No attempt is made to exhibit all the new features, which would be quite impossible in the space available. The emphasis is thus on the simpler aspects of NPL, which is somewhat unfair to the language since the simplicity of doing simple things is only half the story. NPL also has powerful capabilities for handling complex operations that in other languages are difficult or impossible to program.

**Example 1.** S is given as a function of R, according to the following formulas:

$$S = \begin{cases} 17{,}000 - 0.485\ R^2 & \text{for } R<120 \\ \dfrac{18{,}000}{1 + \dfrac{R^2}{18{,}000}} & \text{for } R\geq 120 \end{cases}$$

R is to be computed for S values ranging from 20 to 200 in steps of 5.

The program shown in Fig. 1 illustrates that the full power and complexity of the language need be no deterrent to the unsophisticated user doing simple things. Observe the semicolons terminating the statements, the mixed mode arithmetic, the label (AGAIN) terminated by a period, the relational operators (LT for less than, etc.), the ELSE path on the logical IF statement, and the formatless output statement. This example could be written to demonstrate even more graphically how similar to FORTRAN an NPL program can be, at the expense of making the logic more cumbersome.

**Example 2.** The transfer function of a certain servomechanism is given by:

$$T(i\,\omega) = \frac{K}{i\,\omega(1 + iT_1\omega)(1 + iT_2\omega)}$$

All the letters stand for problem variables of no great interest to us, except *i* is the imaginary unit. The result, T, is a complex number.

Values of the parameters are to be read, together with values of W, WL, C, and L. T is to be computed for values of omega starting with W and not exceeding WL; omega is to be incremented by C if L = 1, and multiplied by C otherwise.

The program in Fig. 2 begins with a procedure declaration, since every program is technically a procedure, even if as in this case the net effect is simply to give a name to the program. Next is a declaration that T and I are complex and the other variables are floating point. Actually, the variables W, T1, T2, WL, and C could have been omitted from the declaration, since the IJKLMN naming convention would have identified them as floating point by default. The READ statement again is format-free.

The DO statement here shows the WHILE option,

Fig. 1

```
R = 20;
AGAIN. IF R LT 120 S = 1.7E4 — .485*R*R ELSE S = 1.8E4/(1 + R*R/1.8E4);
WRITE (R, S);
R = R + 5;
IF R LE 200 GO TO AGAIN ELSE EXIT;
```

Fig. 2

```
SERVO. PROCEDURE;
DECLARE (K, W, T1, T2, WL, C, L) FLOAT; (T, I) COMPLEX;
I = 1I;
READ (K, T1, T2, W, WL, C, L);
LOOP. DO WHILE W LE WL;
  T = K/(I*W*(1 + I*T1*W)*(1 + I*T2*W)   );
  WRITE (W, T);
  IF L = 1 W = W + C ELSE W = W * C;
  END LOOP;
END;
```

Fig. 3

```
PDE. PROCEDURE;
DECLARE U( (0,60), (0,60) )FLOAT;
READ (OMEGA, EPS, MAXIT) (F(10), F(10), F(3) );
MORE.DATA. READ (I, J) (F(3) ), (ELEMENT) (F(10) );
IF I LT 0 GO TO COMPUTE; ELSE U(I, J) = ELEMENT; GO TO MORE.DATA;
COMPUTE. ITN = 1; A = OMEGA/4; B = 1 — OMEGA;
NEW.ITERATION. D = 0;
INNER.LOOP. DO J = (1, 59); DO I = (1,59);
IF I GE 18 AND I LE 42 AND J GE 18 AND J LE 42 GO TO SKIP; ELSE DO;
  UNEW = A*(U(I+1,J) + U(I—1,J) +1) + U(I,J—1) ) + B*U(I,J);
  RESIDUAL = ABS(UNEW — U(I, J) );
  IF D LT RESIDUAL D = RESIDUAL;
  U(I, J) = UNEW;
  SKIP. END;
END INNER.LOOP;
IF D LT EPS GO TO OUT;
ITN = ITN + 1;
IF ITN LE MAXIT GO TO NEW.ITERATION;
ELSE WRITE ('FAILS TO CONVERGE IN', ITN, 'ITERATIONS') (A, F(5), A);
OUT. WRITE (U). (10F(11.5), RECS);
END;
```

under which the controlled segment is repeated as long as the logical expression is true. The range of the DO extends down to the END LOOP statement. Because of the presence of the complex variable I in the replacement statement in the loop, that entire computation would be done using complex arithmetic. The declaration of T as complex would cause the real and imaginary parts to be printed separately by the format-free output statement.

**Example 3.** A square pipe with a square hole cut out is partially submerged in an ice bath and is carrying a hot fluid. It is desired to find the temperature distribution within the pipe material by solving the LaPlace equation with appropriate boundary conditions.

It was felt convenient in the mathematical formulation of the problem to use zero subscripts. The program in Fig. 3 does likewise, as declared. All variables except U are declared by default. The READ statement here assumes the standard input medium, since the file is not named, but now there is format information. This, we note, is written in parentheses immediately following the list of variables. FORTRAN programmers will recognize the changes in the way of writing field specifications.

The second READ is repeated as many times as necessary to get all the boundary values into storage, and is accordingly given a label. It is assumed that a negative subscript signals the end of the data cards.

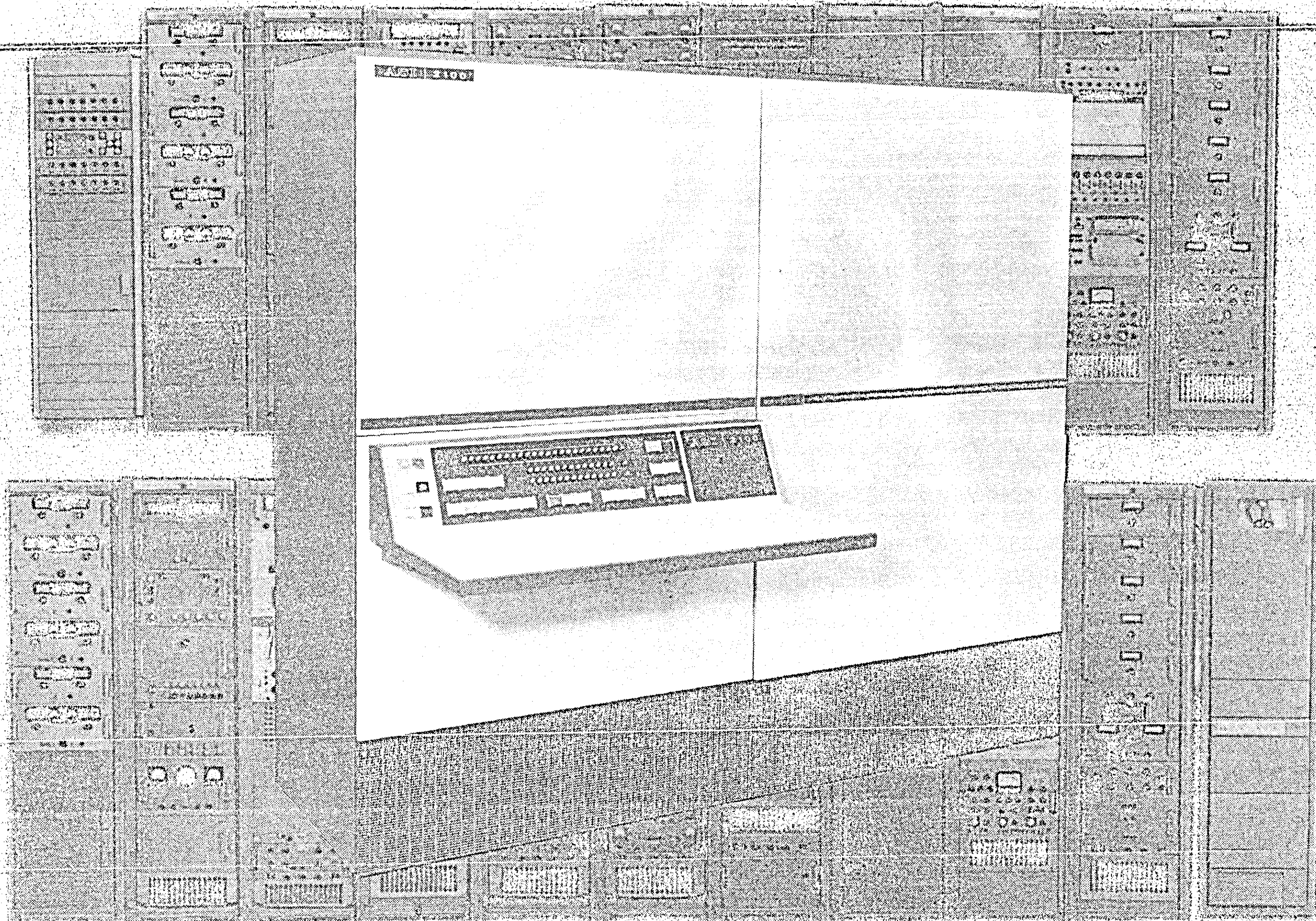
In the body of the program, beginning with the label COMPUTE, we see the NPL version of the FORTRAN DO statement, a logical expression that in FORTRAN would take four IF statements, and a transfer to an END statement to get the effect of a CONTINUE. The output statement assumes that the programmer really wants to print all 3721 values, in which case he simply names the array without subscripting information. The formating given would print 10 values to a line until completion.

**Example 4.** An illustrative file processing application involves a master file containing the model number, unit price, and description for each item that a company sells. The file is on tape; a standard end of file sentinel is used. The file of transactions contains records specifying model number, quantity, and customer; the sentinel is a record having a dummy model number of 99999. The total price of each order is to be calculated, allowing a 2% discount on orders over $100.00; a report is to be printed showing the relevant information for each order.

The program in Fig. 4 has obviously been designed to emphasize similarities with COBOL. The scheme of indenting is not required by NPL, but it is permitted. The DECLARE statement continues down through the description of TOTAL.PRICE, just before the OPEN statement. The processing statements are fairly standard, with the exception of the two that move items to the REPORT record: BY NAME is equivalent to COBOL's MOVE CORRESPONDING. The WRITE statement here references the earlier description of the REPORT record, which contains editing information. Therefore the format may be described simply as alphabetic, by writing the A. ■

Fig. 4

```
PRICING.PROGRAM PROCEDURE;
DECLARE (TRANSACTION.FILE, MASTER.FILE, REPORT.FILE) FILE,
  1 TRANSACTION,
     2 MODEL.NUMBER PIC(5D),
     2 QUANTITY FIXED,
     2 CUSTOMER PIC(20A),
  1 MASTER,
     2 MODEL.NUMBER PIC(5D),
     2 UNIT.PRICE FIXED(6, 2),
     2 DESCRIPTION PIC(20A),
  1 REPORT,
     2 CUSTOMER PIC(20A),
     2 MODEL.NUMBER FILL(2) PIC(5D),
     2 DESCRIPTION FILL(2) PIC(20A),
     2 QUANTITY FILL(2) PIC(3D),
     2 UNIT.PRICE FILL(2) PIC(3D.2D),
     2 TOTAL.PRICE PIC(5ZD.2D);
OPEN (TRANSACTION.FILE, MASTER.FILE) INPUT, REPORT.FILE OUTPUT;
TRANSACTION.READING. READ TRANSACTION.FILE (TRANSACTION) (A, F(5), A);
  IF TRANSACTION$MODEL.NUMBER EQ 99999 GO TO WRAPUP;
MASTER.READING. READ MASTER.FILE (MASTER) (A, F(6), A), ON
  END.FILE GO TO ERROR; IF TRANSACTION$MODEL.NUMBER NE
  MASTER$MODEL.NUMBER GO TO MASTER.READING;
  TOTAL.PRICE = TRANSACTION$QUANTITY * MASTER$UNIT.PRICE, ROUNDED;
  IF TOTAL.PRICE GT 100.00 TOTAL.PRICE = 0.98*TOTAL.PRICE, ROUNDED;
  REPORT = TRANSACTION, BY NAME; REPORT = MASTER,  BY NAME;
  WRITE REPORT.FILE (REPORT) (A); GO TO MASTER.READING;
WRAPUP. CLOSE TRANSACTION.FILE, MASTER.FILE, REPORT.FILE; EXIT;
ERROR. PAUSE 'END OF FILE ON MASTER'; EXIT;
END PRICING.PROGRAM;
```



# Squeeze more computer out of your systems' budget? Yes!

You can with ASI's new 2100 — in two important ways. The 2100 communicates better with any system devices whether they are man, machine or other computer, because it has been designed for system use at a low cost — $87,800. Buffer/"black box" systems costs are lower too because the 2100 is designed for direct external device communications.

The 2100 communicates at large scale computer speeds — 500,000 words per second and it has the capability of communicating directly with external systems with or without using its buffered input/output channels. Multiple priority interrupts are possible, each with its own order of priority permitting automatic program transfer to any chosen external device that the program dictates. The 2100 computer system is adaptable to virtually any system design. The hardware is designed to allow the systems user to be concerned about a minimum number of "black boxes." Both hardware and software are upward and downward compatible assuring that system changes can be made without radical modifications, extensive delays, or exorbitant costs.

ASI has the experience that can benefit you. ASI computers are now operating in a variety of system applications — as a part of aerospace and aeronautical simulators — in a nuclear research installation performing real-time data analysis in on-line experiments — in a research laboratory devising data from an optical scanning system — in on-line seismographic analysis — and many others. ASI's experience in this area has been demonstrated — the result is guaranteed performance for you.

For complete information on your systems application, write to:

ASI

ADVANCED SCIENTIFIC INSTRUMENTS / DIVISION OF ELECTRO-MECHANICAL RESEARCH, INC.

8001 Bloomington Freeway, Minneapolis, Minnesota 55420

CIRCLE 21 ON READER CARD

# COMPUTER CHARACTERISTICS

**18 new mainframes**

by CHARLES W. ADAMS

On April 7, 1964, IBM made the announcement modestly billed as the most momentous in its history. In terms of technological novelty, more startling announcements have surely been made in the past and still others will undoubtedly be made in the future. But for sheer magnitude and majesty, the April 7 unveiling was unquestionably the greatest the computer world has yet witnessed or is likely to see soon again.

Along with the six models of IBM's System/360, a veritable rash of newly-announced speed-ups and other refurbishings of existing computers and more than the usual two or three newly-designed machines one expects to see born each quarter adds no fewer than 18 new listings to Section I of the June issue of the Adams Associates *Computer Characteristics Quarterly.** Also reported are a wide variety of minor changes as well as some major ones—for example, the reduction of almost 40% in the rental price of the RCA 3301 and the availability for that computer of the 3488 card random access file.

These increase to 101 the total number of entries (produced by 22 manufacturers) in Section I of the *Quarterly*. In addition, of course, there are listed in Section II the systems aimed at process control, communications and other specialized

*Available at $10 annually from Charles W. Adams Assoc. Inc., 142 The Great Road, Bedford, Mass.

## SECTION 1

| System | *Monthly Rental* Typical Range | *First Delivery* Month and Year | *Processor Speed* Complete Add Time in Microseconds | Storage Cycle Time in Microseconds | *Internal Storage* Capacity in Words | Type | *Logic* Word Size Instr. Addresses | *Magnetic Tape* Thousands of Characters per Second / Buffering | Maximum Units Attachable | *Random Access File* Capacity / Access Time in Milliseconds | *Peripheral Devices* Cards per Minute In — Out | Paper Tape Characters per Second In — Out | Printer Lines per Minute | Off-line Equipment | *Other Features* Program Interrupt | Index Registers | Indirect Addressing | Floating-point Arith. | Console Typewriter | *Software* Algebraic Compiler | Business Compiler |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **IBM 360 Model 70** | $70,000<br>(45-170) | 9/65 | 1.05[C] | 1[D] | 256-512 | core[E] | 1a<br>0, 1, 2 | 22.5-340K | 256[K] | 224M[L]<br>180 | 1000<br>250 | —<br>— | 600[Q]<br>1100 | 360-40<br>360-50 | √ | √[T] | — | √[V] | I/O | /65[X] | /65[Y] |
| **IBM 360 Model 62** | $50,000<br>(34-100) | 9/65 | 1.87[C] | 1[D] | 256-512 | core[E] | 1a<br>0, 1, 2 | 22.5-340K<br>MRWC | 256[K] | 224M[L]<br>180 | 1000<br>250 | —<br>— | 600[Q]<br>1100 | 360-30<br>360-40 | √ | √[T] | — | √[V] | I/O | /65[X] | /65[Y] |
| **IBM 360 Model 60** | $40,000<br>(25-90) | 9/65 | 2.81[C] | 2[D] | 128-512K | core[E] | 1a<br>0, 1, 2 | 22.5-340K | 256[K] | 224M[L]<br>180 | 1000<br>250 | —<br>— | 600[Q]<br>300 | 360-30 | √ | √[T] | — | √[V] | I/O | /65[X] | /65[Y] |
| **CONTROL DATA 3400** | $25,000<br>(20-45) | 11/64 | 3 | 1.5 | 16-32K | core | 48b<br>1[G] | 15-120[H]<br>MRWC | 512 | 800M<br>225 | 1200<br>250 | 350<br>110 | 1000 | 160A | √ | 6 | √ | √ | I/O | 11/64[X] | 2/65[Y] |
| **IBM 360 Model 50** | $25,000<br>(14-55) | 9/65 | 4[C] | 2[D] | 64-256K | core[E] | 1a<br>0, 1, 2 | 22.5-340K<br>MRWC | 192[K] | 224M[L]<br>180 | 1000<br>250 | 1000<br>— | 600[Q]<br>1100 | 360-30 | √ | √[T] | — | √[V] | I/O | /65[X] | /65[Y] |
| **IBM 360 Model 40** | $12,000<br>(5-35) | 9/65 | 11.88[C] | 2.5[D] | 16-256K | core[E] | 1a<br>0, 1, 2 | 22.5-340K | 128[K] | 7.25M[L]<br>97 | 1000<br>250 | 1000<br>— | 600[Q]<br>1100 | — | √ | √[T] | — | √[V] | I/O | /65[X] | /65[Y] |
| **HONEYWELL 300** | $8,500[A]<br>(2.2-14) | 11/65 | 3.5 | 1.75 | 4-32K | core | 24b<br>1 | 20-83<br>MRWC[J] | 64 | 807M[L] | 800<br>250 | 600<br>110 | 650<br>900 | — | √ | 6 | √ | √ | I/O | 11/65[X] | — |
| **CONTROL DATA 160G** | $7,500<br>(5-20) | 1/64 | 3 | 1.5 | 8-131K | core | 13b<br>1[G] | 15-120[H]<br>MRWC | 448 | 33M[L]<br>225 | 1200[N]<br>250 | 350<br>110 | 150<br>1000 | — | √ | 62 | √ | — | I/O | 1/64[X] | 1/64[Y] |
| **GENERAL ELECTRIC 415** | $7,300<br>(4.8-13.5) | 7/64 | 27.6 | 9.2 | 32K | core | 24b<br>—[G] | 15-120<br>MRWC | 56 | 94M[L]<br>199 | 900<br>100 | 500<br>150 | 1200 | — | √ | √[T] | √ | — | I/O | √[X] | √[Y] |

**IBM 360 Model 70:** C. Overlapped core banks allow increased internal speed. Add time is for four characters or 32 bit word. D. Cycle time for eight characters. E. Each character is eight bits or two decimal digits. K. Magnetic tapes read in forward and reverse directions. L. Mass storage devices include 7.5, 112 and 224 million character disc files; 400 character magnetic strip file; and .83 and 4.1 million character drums. Access times vary from 8.6 to 600 milliseconds. Transfer rates vary from 55,000 to 1,200,000 characters per second. Q. Numeric information only printed at 1400 lpm. T. Sixteen general registers can be used as accumulators or index registers. V. Double-precision floating-point included. X. FORTRAN IV. Y. COBOL '61 Extended.

**IBM 360 Model 62:** C. Add time is for four characters or 32 bit word. D. Cycle time is for eight characters. E. Each character is eight bits or two decimal digits. K, L, Q, T, V. See IBM 360 Model 70. X. FORTRAN IV. Y. COBOL '61 Extended.

**IBM 360 Model 60:** C. Overlapped core banks allow increased internal speed. Add time is for four characters or 32 bit word. D. Cycle time is for eight characters. E. Each character is eight bits or two decimal digits. K, L, Q, T, V. See IBM 360 Model 70. X. FORTRAN IV. Y. COBOL '61 Extended.

**CONTROL DATA 3400:** G. Instructions stored two per word. H. CDC 604 and 607 tape units read in forward and reverse directions. Tape units IBM compatible. X. FORTRAN. Y. COBOL.

**IBM 360 Model 50:** C. Add time is for four characters or 32 bit word. D. Cycle time is for four characters. E. Each character is eight bits or two decimal digits. K, L, Q, T, V. See IBM 360 Model 70. High speed drum not available. X. FORTRAN IV. Y. COBOL '61 Extended.

**IBM 360 Model 40:** C. Add time is for four characters or 32 bit word. D. Cycle time is for two characters. E. Each character is eight bits or two decimal digits. K, L, Q, T. See IBM 360 Model 70. High speed drum not available. V. Single and double precision floating-point optional. X. FORTRAN IV. Y. COBOL '61 Extended.

**HONEYWELL 300:** J. Optional. L. One to eight disc units of 4.2 to 100.8 million characters each. See Honeywell 200 for drum data. X. FORTRAN.

**CONTROL DATA 160G:** G. Instructions use no-address, direct-address, indirect-address, constant address and relative-address modes. H. CDC 603, 604, 606 or 607 tape units. Models 604 and 607 read in forward and reverse directions. Magnetic tapes IBM compatible. L. Four disc files per controller. N. 250 cpm reader available. X. FORTRAN. Y. AUTOCOMM.

**GENERAL ELECTRIC 415:** G. Binary, decimal, single, double, triple and quadruple arithmetic instructions included. L. Up to 28 disc storage units with 23.5 million characters each available. T. Each word of memory can be used as an index register. X. FORTRAN. Y. COBOL.

functions (26 systems built by 13 companies, five of which are not included in the aforementioned 22); while in Section III there appear the characteristics of general-purpose computers manufactured abroad (83 by 27 firms). Thus the June issue contains information on a total of 210 computers produced by 54 manufacturers.

The new announcements rumored or promised as the closing date for the issue was reached (including, for instance, in the Control Data line, substantial price reductions and several new computers) seem sure to make the next few months exciting ones for computer users in the United States.

The compatibility within its own line established by IBM raises questions of when such hardware compatibility across competitive lines will become widespread. In software, for example, IBM FORTRAN, after appropriate polishing, has become such a standard that some manufacturers see little need to prepare a basic FORTRAN manual, relying instead on the use of IBM's. And Honeywell has already profited enormously by designing its H-200 to give hardware compatibility with the now passé IBM 1401.

With IBM apparently fairly committed for the next few years, its competitors have a clear and relatively immobilized target. System/360 offers a refreshingly clean, versatile and powerful instruction logic, one in which almost everybody would find changes he would make and gimmicks he would add, but few that a majority would agree on.

All in all, not a bad design to standardize on. Perhaps it will not be long until there is a considerable degree of hardware compatibility in many of the computers of various manufacturers added to each issue of our *Computer Characteristics Quarterly.* ■

## SECTION 1

| | *Monthly Rental* Typical<br>Range | *First Delivery* Month and Year | *Processor Speed* Complete Add Time in Microseconds | Storage Cycle Time in Microseconds | *Internal Storage* Capacity in Words | Type | *Logic* Word Size<br>Instr. Addresses | *Magnetic Tape* Thousands of Characters per Second | Maximum Units Attachable<br>Buffering | Access Time in Milliseconds | *Peripheral Devices* Cards per Minute In — Out | Paper Tape Characters per Second In — Out | Printer Lines per Minute | Off-line Equipment | *Other Features* Program Interrupt | Index Registers | Indirect Addressing | Floating-point Arith. | Console Typewriter | *Software* Algebraic Compiler | Business Compiler |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **UNIVAC 1050 Model IV** | $7,250<br>(2.4-14.5) | 5/65 | 35[C] | 2 | 8-65K | core | 1a<br>1 | 8.5-133[H] | 16[K]<br>RWC | 528M<br>92 | 900<br>300 | 1000<br>110 | 600[Q]<br>700 | 1004-III | √ | 7 | — | — | I/O | 1/65[X] | 1/65[Y] |
| **IBM 360 Model 30** | $7,100<br>(2.7-20) | 9/65 | 39[C] | 2 | 8-64K | core[E] | 1a<br>0, 1, 2 | 22.5-170K | 128[K] | 7.25M[L] | 1000<br>250 | 1000<br>— | 600[Q]<br>1100 | — | √ | √[T] | — | —[V] | √ | /65[X] | /65[Y] |
| **BURROUGHS B160, B170, B180** | $4,300[A]<br>(1.9-6.2) | 4/64 | 690[C] | 10 | 4.8K | core | 1a[F]<br>3 | 24 | 4<br>none | —<br>— | 475[N]<br>300 | —<br>— | 475[Q] | — | — | — | — | — | — | — | — |
| **COMPUTER CONTROL DDP-224** | $3,350<br>(3.0-8) | 12/64 | 3.8 | 1.9 | 4-65K | core | 24b<br>1 | 25-150[H] | 64<br>MRWC | —<br>— | 200<br>100 | 300<br>60 | 300 | — | √ | 1[T] | √ | √ | I/O | 12/64[X] | — |
| **UNIVAC 1004 Models II, III** | $2,375[A]<br>(1.3-2.8) | /64 | 91 | 6.5 | 961 | core | 1a<br>—[G] | 8.5-34 | 2[K]<br>none[J] | —<br>— | 615<br>200 | 400<br>110 | 600 | — | — | 0 | — | — | — | — | — |
| **GENERAL ELECTRIC 205** | $2,175<br>(1.7-5.5) | /64 | 72 | 36 | 4-16K | core | 20b<br>— | 8-15 | —<br>RWC | 94M[L]<br>199 | 400<br>100 | 1000<br>110 | 300[P] | — | √ | 96[T] | √ | √ | I/O | 5/63[X] | — |
| **DIGITAL EQUIPMENT PDP-7** | $1,800[A]<br>(1.6-2.4) | 1/65 | 3.5 | 1.75 | 4-32K | core | 18<br>1 | 15-60 | 8[K]<br>MRWC | —<br>— | 800<br>100 | 300<br>63 | 1000[Q] | — | √ | 0[T] | — | — | I/O | 5/63[X] | 3/64[Y] |
| **H-W 208** | $950[A]<br>(.8-1.5) | /65 | 80[C] | 40 | 4-32K | core | 24<br>1 | 15 | 64[K]<br>RC, WC | —<br>— | 200[N]<br>100 | 300[P]<br>110 | —[Q] | — | — | 3 | — | — | I/O | /65[X] | — |
| **DSI 1000** | $333[A]<br>(.22-1.5) | 2/64 | 3.2 | 200 | .26-2K | delay | 12b<br>1-3[G] | 4-40 | 4[K]<br>— | 32K<br>8 | —<br>— | 500<br>110 | — | — | √ | 1 | √ | — | I/O | 12/64[X] | — |

**UNIVAC 1050 Model IV:** C. Add time assumes a five-character decimal field. H. Numeric information can be transferred at 200,000 ch/sec. K. Magnetic tapes read in forward and reverse directions. IBM compatible tape units are available. Q. Alphanumeric information printed at 750 or 922 lpm depending on printer. Higher speeds possible for numeric printing only. X. FORTRAN. Y. COBOL '61 Extended.

**IBM 360 Model 30:** C. Add time for four characters or 32 bit word. E. Each character is eight bits or two decimal digits. K, L, Q, T. See IBM 360 Model 70. High speed drum not available. V. Single and double precision floating-point optional. X. FORTRAN IV. Y. COBOL '61 Extended.

**BURROUGHS B160, B170, B180:** A. Model 170, when used in proof and transit operations, has a six-tally register. The card-only system (B160) rents for $2,350. C. Add time assumes a five-character field. F. Instruction word is twelve characters. N. 200 and 800 cpm readers available. MICR documents can be read at 1200 per minute. Q. Six-tally 1250 lpm lister or 700 lpm printer available.

**COMPUTER CONTROL DDP-224:** H. Magnetic tapes are IBM compatible. T. Two additional index registers available. X. FORTRAN II, IV. Note. System is program-compatible with the DDP-24.

**UNIVAC 1004 Models II, III:** A. Rental price for Model III with two magnetic tape units. Model II is a card system with a typical rental of $1,575. G. Plugboard serves as instruction storage unit. J. Magnetic tape stop time may be overlapped with compute time. K. Magnetic tape units are IBM compatible.

**GENERAL ELECTRIC 205:** L. 23.5 million characters per unit. P. 450 lpm printer available. T. Three index registers standard; additional 93 optional. X. FORTRAN.

**DIGITAL EQUIPMENT PDP-7:** A. No rental prices announced. Prices derived from purchase price. Q. 300 lpm printer available. T. Eight auto-indexing memory locations. X. FORTRAN II. Y. Business Package.

**H-W 208:** A. No rental prices announced. Prices derived from purchase price. C. Add time assumes a four-character field. K. Magnetic tapes are IBM compatible. N. Up to 400 cpm possible when reading less than 80 columns. P. Twenty cps reader and punch standard. Q. 15.6 ch/sec typewriter. X. FORTRAN-type.

**DSI 1000:** A. No rental prices announced. Prices derived from purchase price. G. Instructions may specify one to three addresses. K. Magnetic tapes are IBM compatible. X. FORTRAN.

# TESTING REAL-TIME SYSTEMS

## Part 1: development & management

by ROBERT V. HEAD

Real-time systems, at the present stage of their development, differ substantially from non-real-time systems and as a consequence present significant new problems in system implementation. The factors that contribute to system complexity in a real-time environment heighten the difficulties of system testing, a phase of system development which, even in the non-real-time world, is the least amenable to control.

"System testing" here means whatever work must be performed by the system development staff from the completion of programming until the system achieves operational status. This is not limited to checkout of the various real-time programs; it extends also to the necessary checkout of the equipment subsystems and to the integrated testing of combinations of equipment and programs. Final system testing and the subsequent period of system operation require a thoroughly checked out and reliable combination of equipment and programs.

Because the real-time system tends to be applied to the more vital facets of an organization's operation and because, as a consequence, any malfunction of the system which causes it to become totally or partially unavailable has an immediate and potentially disastrous effect, there must be more painstaking testing than has sufficed in the past.

Many non-real-time systems can be converted even when there is abundant evidence that the programs still harbor a large number of errors. Conversion under such conditions may be decidedly ill-considered because of the amount of rerun time consumed when bugs are encountered, but the risk of damage to records or operations is usually not extremely high. Moreover, it has been demonstrated in non-real-time systems work that not all bugs can or need be found even during a protracted system test. To become completely error free, programs must be subjected to "live" operations and to all the variations and permutations in data and operating conditions that occur over a fairly lengthy period of productive system operation.

Although it is doubtful that any real-time system development staff will ever be able to warrant a system as error free, they will have to go far beyond satisfying the testing standards which sufficed for non-real-time systems in order to permit the real-time system to go into operation with a high degree of confidence in its workability. There will be the same "moment of truth" in real-time system development as in conventional systems, when the system manager declares the test phase ended. But in real-time systems, this declaration must be based upon much more solid evidence of error-free programs than in the past.

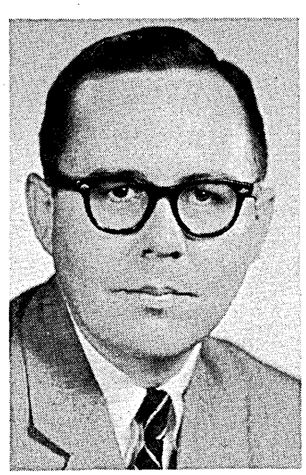
Operating management must be prepared to face disruptions during conversion and the early stages of operation until both the equipment and the programs achieve the requisite level of reliability. At the same time it must be the objective of the development staff, through its approach to testing, to try to prove that a pessimistic attitude on the part of management is unwarranted. The system development staff can best do this by going to virtually any reasonable length to purge errors from the programs.

One of the most pitiable collusions imaginable in real-time system development is that between operating management and the system manager in agreeing to go ahead with cutover prematurely simply because the schedule calls for it or because "we planned it that way." If this is the attitude, there is certain to be a great deal of time lost through confusion and frustration caused by trying to make a system work when the system is just not yet ready to work.

The nature of a real-time system confronts the development staff with a two-fold testing problem: 1) more intensive testing is mandatory to achieve reliable operational status, and 2) it is more difficult to satisfy this heightened testing standard. Let us look further into this dilemma to identify those attributes of a real-time system which, on the one hand, fortify the need for preconversion testing and, on the other, make such testing harder to carry out.

### 1 magnitude of programming effort

Many real-time systems are of such size that there are substantial numbers of programs to be tested and fitted together. One large file-oriented system contained—in addition to its control program and off-line support programs—over 100,000 real-time application program instructions. These were divided up into some 800 programs



*This article is adapted from a forthcoming book, "Real Time Business Systems," by Mr. Head, who is vp and manager, Systems Planning Div. of the Security First National Bank, Los Angeles. Previously with IBM, he was associated with the American Airlines SABRE system as a senior systems engineer, and, before that, with GE on the ERMA project. He is a graduate of George Washington U., and holder of the DPMA's certificate in DP.*

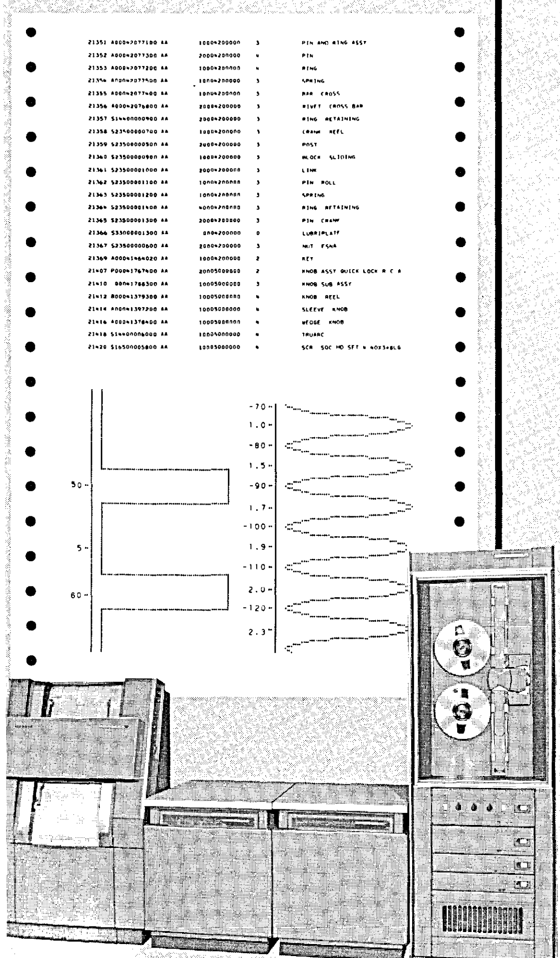
# Concerned about wasting valuable 1401 time on routine printing chores?

# Like a low cost, high speed read/write converter to link data phones and computer?

*Here are two new Potter developments you'll want to know about*

**NEW**
FROM POTTER

# an independent off-line high-speed printing system that both prints and plots... yet costs less than half what people thought they had to pay.



It doesn't usually make good business sense to burden a $4000/month computer with routine printing chores. Especially when there's a system like Potter's new 3501 that lets you prepare tapes on your computer — then print out hard copy independently. A system that not only prints at 1000 line-a-minute speeds but is equipped to give high speed graphical presentation of data as well.

*It's new. New from Potter.* Write today for proof of how the 3501 Graphic Printer System can step up your data processing efficiency.

For detailed information on the Potter Off-Line Graphic Printing System circle No. 8 on the Reader Service Card.

POTTER INSTRUMENT CO., INC.
*Sunnyside Boulevard • Plainview, New York*

THE LEADER IN EDP EQUIPMENT • • • • •

NEW

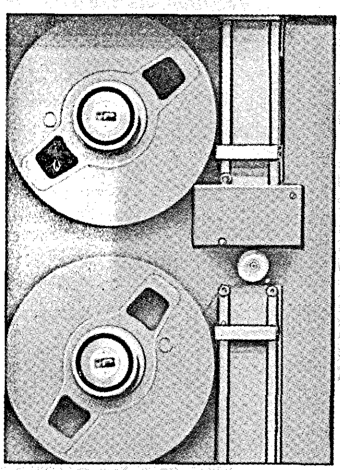
FROM POTTER

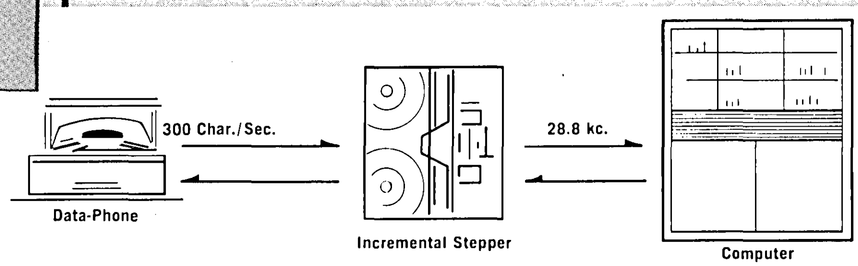
# low cost magnetic tape stepper that provides a high-speed read/write link between data-phone and computer...without buffering

Devices that record digital data on tape, a character-at-a-time are nothing new. But to have one transport (IBM compatible) capable of both reading and writing 1 character-at-a-time (asynchronously) — and also block-at-a-time (synchronously) — *that is new! New from Potter.*

Write today for full details on Potter's new model 1501 Incremental Transport.

For article entitled "Applications for Magnetic Steppers" circle 9 on Reader Service Card.






POTTER INSTRUMENT CO., INC.
*Sunnyside Boulevard • Plainview, New York*

THE LEADER IN EDP EQUIPMENT • • • • •

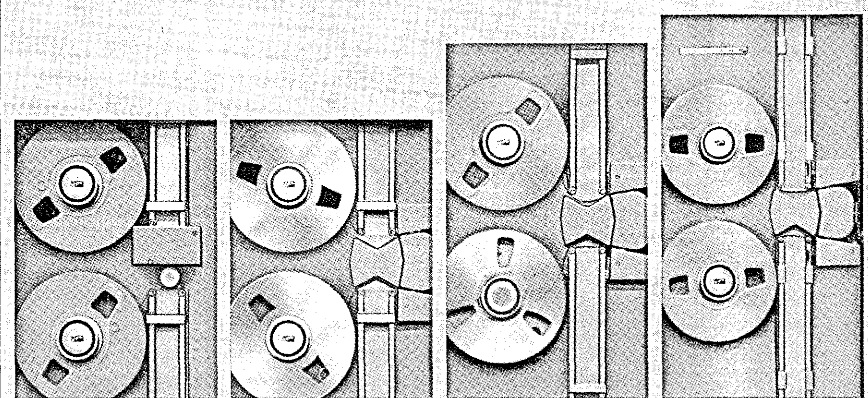
**NEW**

FROM POTTER

# a complete family of magnetic tape transport systems ranging from asynchronous to 60 kc

**They look alike, operate alike, are designed alike . . . with most parts interchangeable.**

Here's a whole new "family" of vacuum column transports each designed to serve a specific data spread range. From asynchronous to 75 ips, 200 to 800 bpi . . . but common in operation, parts, service and maintenance procedures. The benefits of standardization — yet a wide choice of performance and price. *It's new. New from Potter.* Write Potter today to find what the family concept can mean to you.



**1501**
Incremental Transport
IBM compatible, 200 bpi
Read, write or read/write capability to 300 cps asynchronous—28.8 kc synchronous

**MT-24**
1-36 ips
IBM compatible
200 commands per sec.
packing densities to 800 bpi
data transfer to 28.8 kc

**MT-36**
1-50 ips
IBM compatible
200 commands per sec.
packing densities to 800 bpi
data transfer to 40 kc

**MT-75**
1-75 ips
IBM compatible
200 commands per sec.
packing densities to 800 bpi
data transfer to 60 kc

Complete technical data circle 10 on Reader Service Card.

POTTER INSTRUMENT CO., INC.
*Sunnyside Boulevard • Plainview, New York*

THE LEADER IN EDP EQUIPMENT . . . . .

which were produced in accordance with the contents of 400 written program specifications. These 800 programs were brought together in varying combinations to form some 80 input-oriented "packages" of related programs needed to process various entries. The personnel who worked on this system were organized into four basic groups, each doing a portion of the job divided up along functional lines, with many programs being common to more than one group.

### 2 misinterpreted specifications

It is during system testing that the specification and programming work performed by each individual and group must be gradually tied together into a unified operational skein. Considering the application programs alone and ignoring control and support programs, there are numerous interfaces to be tested at both the specification and the programming level. Even assuming that all specifications are consistent among themselves, i.e., that incongruities in specifications that impinge upon each other have been discovered and corrected prior to the start of testing, it is necessary to assure that each programmer involved:

- Understood the specifications.
- Understood the interpretations placed upon the specifications by his fellow programmers.
- Reflected both these understandings in his program.

When one recognizes the imperfections in the methodology of preparing specifications and the ambiguities inherent in both narrative descriptions and flow charts, he can anticipate that misunderstandings, great and small, will be built into programs. It is a combination of the imperfection of written specifications and the fallibility of those who program from them which contributes to the need for extensive testing in order to achieve a workable system.

### 3 repeatability

Upon detection of a program error in a non-real-time system, as typified by a magnetic tape batch processing application, the usual procedure is to locate the error (with the aid of instruction traces, printouts, memory dumps, etc.), attempt to correct it, and then rerun with the same inputs as before to see whether the correction "took." This general procedure is followed, not only during system testing but beyond, when bugs are encountered in production runs.

Unfortunately, such a relatively straightforward debugging method is not feasible for real-time systems. Entries generated at remote terminals in random and asynchronous fashion often cannot be repeated in exactly the same sequence needed to make an error condition repeat. The difficulties resulting from this limitation on repeatability are not insurmountable, but they are formidable. They necessitate employment of terminal simulator programs during the early stages of system testing in order to exercise the programs independent of this uncontrollable terminal input. Later, when testing has progressed to the point where terminals are placed on-line for more realistic tests, data logging programs and procedures must be devised to capture the input in its original form and sequence. Even then, in a multiprogrammed system, it may be impossible to make all errors repeat because of variations in allocation of main memory, lengths of main memory queues, and varying status of relocatable programs.

It is these complexities which make real-time programmers yearn for the days when input tapes could be remounted and a repeat run initiated under constant conditions following insertion of a program "patch."

### 4 equipment interaction: multiprocessing

There may in a real-time system be several equipment subsystems, each with stored program capability. This multiprocessing potential imposes extra demands during system testing because of the necessity of real-time communication among processors. This may include the communication of data, as in the case of entries originating at terminals and passing first into a communication multiplexor such as the IBM 7740 and thence to the central processing unit. Or it may involve communication of control data, as in the switchover procedures for a duplex system. In any case, the components of the system must exchange information at least occasionally and in many systems frequently. This means that all interfaces must be tested not just from a hardware standpoint but to make certain that all specifications and assumptions about the format and condition of data have been correctly understood by the programmers working on either side of the interface in developing their respective software subsystems.

Not surprisingly, there is a tendency in multiprocessing systems to organize the programming along machine lines. The presence of a stored program multiplexor, for example, offers a convenient way to begin making a division of programming labor. Thus, we may encounter a "message assembly and routing" group, whose sole concern is multiplexor programming, and a "message processing and file maintenance" group, whose interests are restricted to the central processor. Once established, machine-oriented groups work more or less in isolation to produce their program subsystems. Eventually there must be testing of all these subsystems operating in conjunction with each other. It is here, when pieces are put together to form the overall processing system, that the consequences of a multiprocessing approach begin to make themselves felt in the form of program errors and inconsistencies.

This tendency toward workload assignment along machine lines may be heightened by the geographic or organizational separateness of the programming groups. The equipment manufacturer might, for example, undertake to program the communication multiplexor, leaving the task of programming the central processing unit a customer responsibility. In this event, one programming group might be based at the manufacturing plant and the other at the eventual computer site. When this happens, it is frequently the line of least resistance to fail to communicate during the specification and programming phases. At system testing time the consequences of this lack of communication come to the surface.

### 5 program interaction: multiprogramming

Where several programs share the same computer at the same time, each engaged in processing a different transaction, the possibilities of error due to unplanned and illegal program interaction are enormous. System testing must channel this interaction into the patterns demanded by the control program, the application program specifications, and the programming standards. Conditions whereby one program operates upon the data of another are suggestive of the kinds of errors that occur. Many of these would immediately put the system out of operation if they were to arise during productive operation.

Some equipment protection may be available to safeguard the system against the depredations of a real-time program gone amuck. A memory protection feature can be obtained to guard certain areas of main memory by causing an interrupt and reversion of control to the control program if an illegal reference is made by an application program beyond a permissible range of main memory locations. An interval timer can aid in endless loop detection by causing an automatic interrupt and transfer to the control program if any one application program usurps the computer by failing to execute a control program macro instruction periodically.

The nature of a multiprogrammed system complicates the problem of repeatability. At a given instant when the system "hangs up" in some application program, it may be impossible, without special debugging aids, to determine what processing preceded the activation of this particular program and whether the predecessor processing was logical or illogical. Often one may attempt to reintroduce the same entry conditions without success because the exact string of program segments to be executed the second time may differ from the original string, perhaps only because the variation of a few milliseconds in file access time due to a different positioning of the disc access arms causes one application program rather than another to be activated and its file request fulfilled sooner than before.

### 6 inherent logical complexity

Commercial data processing systems, whether real-time or not, are sometimes regarded a trifle superciliously by those engaged in scientific computation. The feeling seems to be that, while there are challenging aspects to commercial system analysis, the programming of such a system is mere child's play as compared to the difficulty of programming a scientific problem. Comparison invariably reveals that the arithmetic steps in an accounting application are trivial when contrasted to those in a typical scientific problem. Despite this, the coding in a commercial system may be more difficult to test than that in the scientific program.

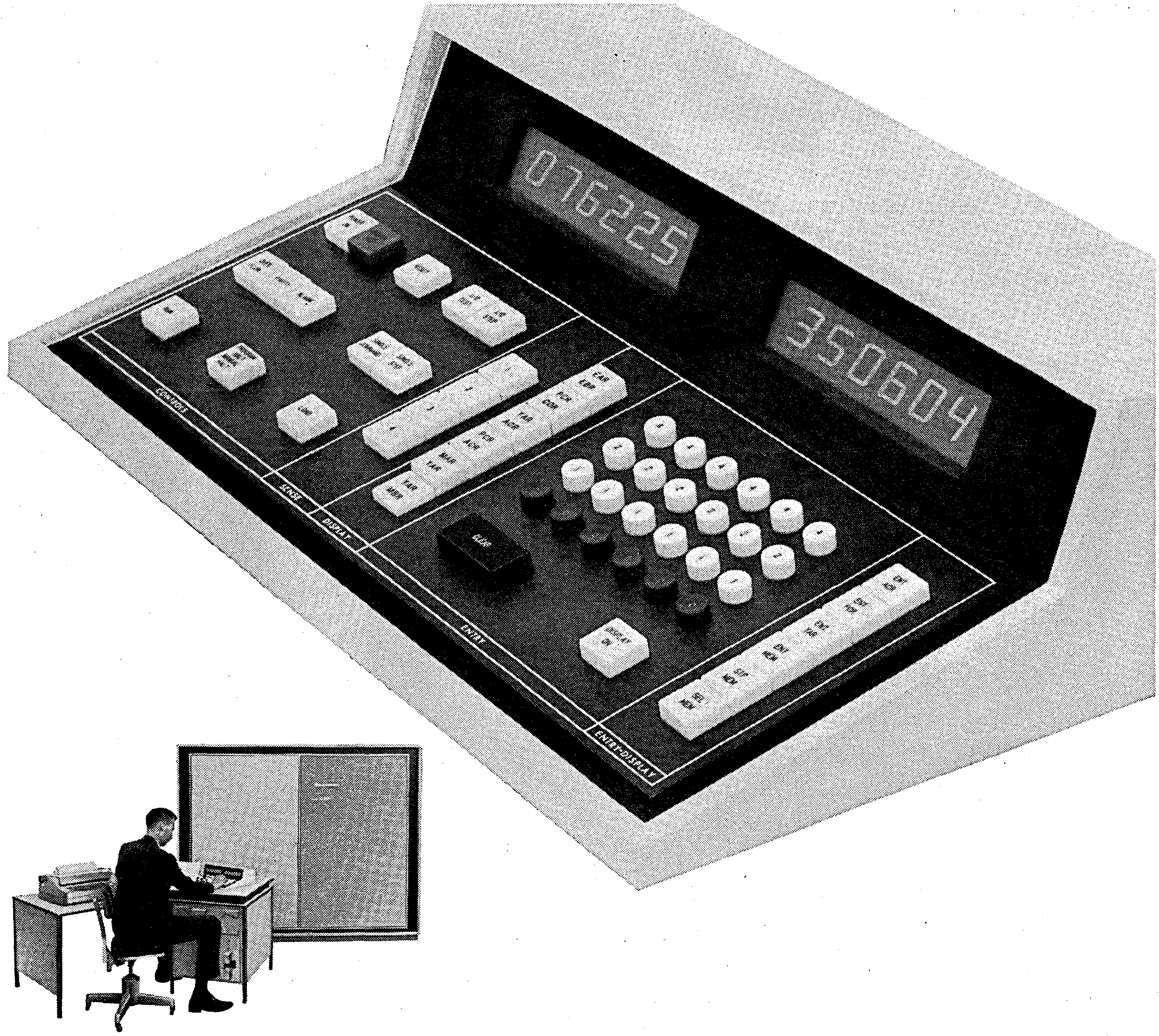
Behind this seeming paradox is an interesting characteristic of commercial systems: there are usually many *more decision points* in an accounting application than in a scientific computation problem of equivalent length. No one has as yet done a comparative study of this phenomenon, but scrutiny of even the simplest data processing program reveals numerous major decision points. Thus, while arithmetically primitive, the logic and decision making potential of real-time commercial programs may be very complex. And where such logic dictates a branch to a new program which must be read into main memory and linked to the one already there, error possibilities begin to abound.

### 7 random access storage limitations

One feature of non-real-time systems which permits them to go into operation after only minimum testing is the relative invulnerability of magnetic tape files to program produced errors. In testing and initial conversion of a batch processing system, it is often the practice to keep "grandfather" copies of master files as well as the corresponding activity tapes. Then, if an error is discovered, a rerun can be made to produce a revised current master file free of error. This protection of data through tape retention is certainly not foolproof, as a posting error may not be discovered until so many cycles have been gone through that the regeneration process cannot be repeated. (Perhaps the "great-grandfather" tape has been "scratched," perhaps the rerun time would be prohibitive, perhaps statements containing the erroneous data have already been issued and must be corrected by reversal procedures rather than rerun). Nevertheless, rerun to correct erroneous master file records is feasible more often than not in tape systems.

By contrast, it is impossible to do this in a real-time system employing random access storage. Here there is no periodic updating of a master file by running batched and sequenced transactions against it, but rather an unpredictable consultation of master file records caused by entries randomly generated at terminals and requiring immediate processing. A real-time program error which affects master records may: 1) be difficult to discern in the normal course of processing, and 2) fail to leave a trail of which records have been updated in error. Consider an exception path in an airline system seat inventory program as an example. Suppose that whenever the "seats available" field in an inventory record show exactly four seats remaining, the number of seats requested by the current entry is erroneously and arbitrarily decreased by one seat more than requested whenever the inventory record is updated, and the response sent back to the terminal. The result of this error will be a nagging and persistent overbooking condition which may hamper the system for weeks before detection. And once this type of error is discovered, one cannot revise the erroneous inventory records by rerun. For it cannot be determined which of the many flight date inventory records maintained by the system have experienced the erroneous updating and which have not. ■

*(Next month: Part II, Levels of Real-Time Testing)*




# the computer for real-time data systems

That's the Beckman Model 420. It's in production and especially engineered for data acquisition and processing systems, telemetry ground stations, communication systems, etc.

**Real-time and on-line** uses of the Model 420 include: System checkout and calibration. Sampling, processing and recording from hundreds of analog or digital sources. Decommutating data, detecting and correcting for loss of synchronization. Converting to engineering units. Testing selected channels for alarm conditions, selecting and formating data for readout and control.

**Input/output flexibility** is achieved by fully buffering each channel, allowing them and the central processor to function independently. Thus the computer can respond to real-time I/O requirements without waiting for completion of instructions currently in process.

**Performance and features** are outstanding. Operating console has numerical readout, table-top operating controls and audible alarm. Indexing is unlimited, since each memory cell can be used as an index register. Circuitry is all-solid-state. Full read/restore cycle takes 3.2 μsec, addition time 6.4 μsec. Word length is 18 bits. System expansion is simple, since I/O channels and memory can be increased entirely within the 420's standard cabinet. For details call your Beckman Field Engineer or write

**Beckman®** INSTRUMENTS, INC.

**SYSTEMS DIVISION** FULLERTON, CALIFORNIA • 92634

INTERNATIONAL SUBSIDIARIES: GENEVA, SWITZERLAND; MUNICH, GERMANY; GLENROTHES, SCOTLAND; PARIS, FRANCE; TOKYO, JAPAN; CAPETOWN, SOUTH AFRICA

CIRCLE 24 ON READER CARD

# EVOLUTION OF THE PROGRAMMING SYSTEM

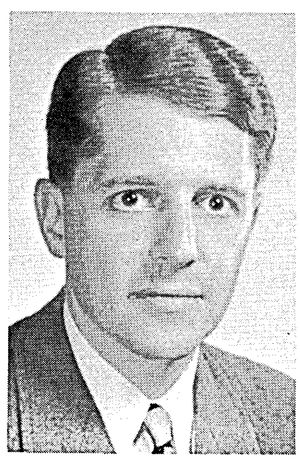
by MARK HALPERN

As the craft of programming approaches its majority, a number of its more thoughtful students have paused to survey and sum up its brief but spectacular past.[1] The richness of that past has naturally made them cast about for some organizing principle or metaphor around which to arrange its events so as to exhibit them as a system rather than miscellany. Quite as naturally, almost all of them have adopted for that role the concept of evolution; it is hard in the 20th century to think of any alternative explanation of a process of continuous development taking place in an environment of intense competition. And there is good reason to think that the concept, applied in its full power, can go far toward making sense of the past and giving hints for the future. But the concept of evolution adopted by most historians of programming seems to be little more than the notion of continuous progress—a notion very different from that of biological evolution, with its mysterious survival of the apparently helpless and dying out of the powerful, and its pursuit of several seemingly parallel lines of development at the same time. The course of programming-system development, brief as it is, shows the same irrelevancies, back-trackings, and false starts; no simple principle of steady, single-minded improvement of the breed can account for it any more than for the origin and distribution of natural species.

The common view of programming-system evolution sees a single line of development starting from absolute machine language and progressing through symbolic machine language to the present plateau of procedure-oriented compiler languages. Above this plateau we see, but cannot yet quite reach, the problem-oriented languages (which will presumably require yet another kind of processor, in keeping with the tradition that each jump in language level necessitates a new kind of vehicle). A few of the chronicler of this evolution do mention parenthetically that an increasing specialization of application is one feature of this process, so that the later species are not simply bigger and better versions of the earlier. But the consensus is clearly that there has been steady progress, with compilers like FORTRAN, COBOL, ALGOL, and JOVIAL representing today's highest achievement and pointing the way to tomorrow's further triumphs.

The writer and a group of associates do not share this view, but have formulated a sharply different alternative, and draw from that dissenting view some unorthodox conclusions about the direction such evolution should take now.[2] The facts on which we base our own view are commonplace, but perhaps need to be drawn together and put in a format convenient for critical inspection if they are to impress others as they have impressed us. We attempt such a presentation in Table I. We have there rated what seem to us the four principal species of programming systems under each of four categories: Applicability, Leverage, Extendibility and Tolerance.[3] By *Applicability* we mean the appropriateness or "naturalness" of the system language to the job at hand. By *Leverage* we mean the amount of useful work that gets done for each statement written by the programmer—a magnitude roughly proportional to the average number of machine-language instructions generated by those statements. By *Extendibility* we mean openness to non-trivial extension, by users, of the operation repertoire—i.e., extension that is neither a mere packaging of existing operations, nor a mere relapsing into machine language with consequent loss, in such hand-coded parts of a program, of the amenities offered by the higher-level language. By *Tolerance* we mean ability to accommodate programmer idiosyncrasies and laxity by accepting programmer-defined notation.

Most of these ratings should require no discussion, but a few may raise questions that need answering before



*Mr. Halpern leads a group currently engaged in developing a meta-language processor at Lockheed Missiles & Space Co., Palo Alto, Calif. He entered the computer field with IBM's Programming Research and Programming Systems departments. He is a member of the ASA X3.4.2 language standards subcommittee, and is writing a book, "The Craft of Programming." He holds degrees from City College of New York and Columbia Univ.*

[1]Some of the better of these surveys — those to which our strictures will least apply — are Orchard-Hays (1961 and 1962), Elbourn and Ware (1962), and Rosen (1964). Mr. Orchard-Hays in particular seems to lean toward the view that will be developed here. (Full citations for papers referred to in this short form will be found in an appended list).

[2]Unorthodox, but not altogether unheard of: conclusions very similar to ours are to be found in Mealy (1962). The "SSD" (SHARE secretary distribution) correspondence, the source of this reference, is a very rich and insufficiently noticed source of information on what working programmers and their immediate supervisors think about hardware and software.

[3]We choose these four dimensions because, in our view, they have characteristic and necessary values for each species of system. Other dimensions may be of interest in other contexts — efficiency of space usage, for example, or object program quality — but since their values may vary widely from one example to another of a given species, they are useless for our purpose.

we go on to suggest some conclusions to be drawn from the tabulation as a whole. By "macro-instruction expander" we mean not merely those assembly programs with some macro capability, such as IBMAP, but also systems like AUTOCODER, in which macros are not an incidental feature but the standard programming tool. In giving these processors a Leverage rating of 1:25 (meaning that source language statements generate an average of 25 machine language instructions), we are drawing on experience with such systems as well as the better-known assembly type. This particular rating is intended as nothing more than a qualitative indication, of course, since the ratio will depend on the programmer's willingness to define elaborate macros. This in turn will depend on the attractiveness of the processor that will be their vehicle: how much of the tiresome mass of detail it relieves them of, and how convenient it makes their use, once defined, in the original and later programs. These points will be returned to later; we will state here only the experience-proven fact that a good macro processor quickly leads to the definition of macros considerably more powerful than conventional compiler statements. The rating given this species under Tolerance reflects the ability it gives the programmer to specify some, at least, of his own notation.

## compiler defined

What we mean by "compiler" is a processor designed solely to translate a particular higher-level language (i.e., not merely a symbolic representation of machine language) designed for some particular application. In this it differs qualitatively from all other species of programming system, which are perfectly general in application. From this alone some rather broad conclusions may fairly be drawn, we believe, but we will first complete our program of supporting the ratings given in Table I. The leverage of a compiler is a measure of the number of instructions generated by its statements; this will certainly average several per statement, but will be less than that of such macros as confirmed users of macros will soon define for themselves.[4]

Table I

| | Applicability | Leverage | Extendibility | Tolerance |
|---|---|---|---|---|
| Absolute Mach.-Lang | Minimal | 1:1 | None | None |
| Symbolic Mach.-Lang (Assembler) | Low | 1:1 | None | Low |
| Macro-Instructions (Expander) | High | User's control; typically, 1:25 | Substantial | Substantial |
| Application Language (Compiler) | Either near-ideal or near-useless | Typically, 1:12 | None | Little or none |

In Applicability a given compiler will either be near-perfect or practically unusable, depending on whether or not it was designed specifically for the application the programmer has in mind. Thus FORTRAN is practically useless for bit- and character-manipulation; COMIT is practically useless for computation.

## applicability vs extendibility of compilers

In rating the compiler, it is hard to keep Applicability and Extendibility distinct: what makes a compiler unsuitable for a given application is the lack of some needed statement-types (this reflects on its Applicability) together with the impossibility of adding such statement-types within a source program, as would be possible in a macro-processor (this reflects on its Extendibility). Many compilers try to evade this restriction in Extendibility by permitting shifts into assembly language where an operation not provided in the compiler language is needed, but this facility can be used only at the sacrifice of the chief advantages the compiler offers: "natural" notation, at least potential machine-independence, and freedom from coding-level errors. The crushing objection to this expedient, though, is that most compiler-users don't know assembly language, and are unable to avail themselves of it even if willing to pay the price. (We repeat, to forestall certain possible objections, that the ability to label a sequence of statements and then order that sequence performed a desired number of times, with parameter variation at each repetition, does not constitute Extendibility in our sense. Not even the ability to define such a sequence as a closed subroutine suffices. These devices, invaluable as they are, only permit convenient packaging of such operations as the compiler's authors thought to include).

We hope it is evident from the survey summarized in Table I that the picture so often drawn of steady and unqualified progress as we move from machine language to compiler language is false. Our expansion of the customary survey to include the usually overlooked macro-processor helps to show just where this simple model fails to conform to reality. Note that as we move down the columns in Table I, there is monotonic increase in all categories—until we come to the compiler. Here for the first time, there is some regression. There is, for any given compiler, the possibility that it may be completely useless for the job at hand—a possibility unknown before. It follows immediately, we contend, that the compiler does not represent the next plateau of all-around merit above assembly language or macro-processor language. It is rather a qualitatively different thing, not rightly to be compared with the other species at all. These others are, in a strict sense, *programming languages*—languages designed to permit programmers to describe arbitrary procedures to a computer. Compiler languages are not programming languages; they are *mechanized application languages* (MAL's), and have a place and a justification all their own.

We have criticized the way others have employed the analogy of evolution to give order and direction to their surveys of programming systems; we would apply it in finer detail, and thereby account for the MAL. There is, as we see it, a single main line of descent from absolute machine language to the macro-processor. The distinguishing mark of this family is that they are all, in the sense just defined, programming languages. As with any evolutionary process occurring under the influence of natural selection, sports appear, sometimes prosper briefly, then mostly dwindle or vanish as the environment changes beyond their ability to compete or adapt (for example, programmed floating-point computation, and octal coding

[4]We chose the 1:12 ratio intuitively; later we came across this passage: "In the problems mentioned . . . the ratio of the number of output machine instructions to the number of input FORTRAN statements for each problem varied between about 4 and 20" [Backus (1957), p. 197].

systems). An occasional sport, however, finds an ecological niche—a small but permanent part of the environment to which it is so perfectly suited that, although it cannot evolve further, survives indefinitely. In the animal kingdom, a classic example of such perfect design is the cockroach, which has survived essentially unchanged since the beginning of paleontological time. Knowing the authors of FORTRAN as we do, we feel sure they will accept in the spirit in which it is meant the assertion that they have created the cockroach of the programming kingdom.

What is true of FORTRAN is true in lesser degree of the other MAL's, such as COBOL, ALGOL, and COMIT. They have all found niches of some degree of permanence, and there they stay, their survival more or less guaranteed, but unable to spread outside. What are these ecological niches where these offshoots prosper? The special condition that must obtain, we suggest, is the existence of a generally accepted notation for the MAL's domain of application, and the MAL's success and survival is directly proportional to the universality of the notation it mechanizes. This notation will ideally be one that is well-established beforehand, as in the case of FORTRAN's algebraic notation, or—less satisfactorily—one of disputed merit, but backed by force great enough to squash opposition, as in the case of COBOL's data processing notation. Given agreement on notation, however arrived at, and given also that the notation is good enough so that the loss of Extendibility is negligible, the MAL survives.

### return to the macro-processor

It survives, but cannot, we think, point the way for future development. The number of these ecological niches—or, to revert to literal usage, the number of accepted special notations awaiting mechanization—is small, and indeed the number may already have been exhausted by the MAL's mentioned above. From now on, the MAL-writer will have to create, and not simply mechanize, his application-oriented language. The difficulty of doing this to the general satisfaction of the programming community should be evident from the COBOL experience, in which little agreement could be secured among computer manufacturers and potential users on a language for the comparatively simple procedures of business data processing until the Department of Defense banged heads together. The chance of another MAL as widely successful as FORTRAN seems small.[5]

It is time, then, for a return to the main line of programming-system development: the line of general programming languages. The years of the MAL's have been profitable and instructive—we hope to show later that we have learned something from them—but, as we have tried to indicate, little more is to be expected from pursuing this line of development. To return to the last point on the main line, the point from which the MAL's digressed, is to return to the macro-processor. This said, two possible misunderstandings must be dealt with before they take root.

First, we are not "against compilers;" we do not advocate the abandoning of any that has shown itself satisfactory for its purpose. In our view the compiler is simply a special case of the macro-processor[6]—one that gives up Extendibility and breadth of Applicability in order to gain a more perfect Applicability to a single well-defined problem area and, sometimes, greater object-program efficiency. We think that very few problem areas exist for which this is a reasonable tradeoff (we will later argue that today no tradeoff whatever is necessary to secure the qualities that compilers aim at), and that most of those few have had their compilers written already. The bewildering proliferation of compilers since FORTRAN, none of which has won anything like FORTRAN's well-deserved success, bears out our contention that that success has been widely misattributed. It is in spite of its being a compiler that FORTRAN has won such wide acceptance, not because of it; the application-area of FORTRAN is practically unique in surviving and even thriving on the compiler tradeoff.[7]

### achieving real progress

Second, we do not advocate "a return to the macro-instruction processor" in the sense that we would have programmers adopt existing processors like IBMAP and AUTOCODER; we say only that such systems, and not compilers, should be taken as the base or jumping-off point for programming-system development. That plenty of development beyond the level reached by existing macro-instruction systems is necessary we not only grant, we insist; real progress consists not in making new compromises, but in adding new features to the good ones already achieved. We want the open-endedness and high Applicability of macro systems, but will not be satisfied with, for example, the comparatively crude, machine-oriented notation of most macro-processors after enjoying the more application-oriented compiler notations; we will not tolerate the glaring coding inefficiencies often seen at the boundaries between macros; we will not be content to compute in a form less convenient than algebraic notation. The test of the view of programming-system evolution just outlined lay, for us, in the fruitfulness of the suggestion it offered for creating a new programming system. In a forthcoming article we will present for the reader's judgment the system that resulted from following that suggestion. ■

### REFERENCES

Backus, John W. et al., "THE FORTRAN Automatic Coding System," *Proc. Western Joint Computer Conference,* 1957, p. 188-198.

Orchard-Hays, William, "The Evolution of Programming Systems," *Proc. IRE,* January 1961, p. 283-295.

————, "Computer Languages," *International Science and Technology,* December 1962, p. 33-41.

Elbourn, R. D. and W. H. Ware, "The Evolution of Concepts and Languages of Computing," *Proc. IRE,* May 1962, p. 1059-1066.

Mealy, G. H., Letter to D. J. Farber suggesting alternative to current direction of programming system development, SHARE Secretary Distribution 92, Serial C-2594, May 1, 1962.

Moore, Ramon E., "Interval Arithmetic and Automatic Error Analysis in Digital Computing," Technical Report 25, Applied Mathematical and Statistical Laboratory, Stanford Univ., Stanford, Calif., 1962.

Halpern, Mark, "Computers and False Economics," *Datamation,* April 1964, p. 26-28.

Rosen, Saul, "Programming Systems and Languages: A Historical Survey," *Proc. Spring Joint Computer Conference,* 1964, p. 1-16.

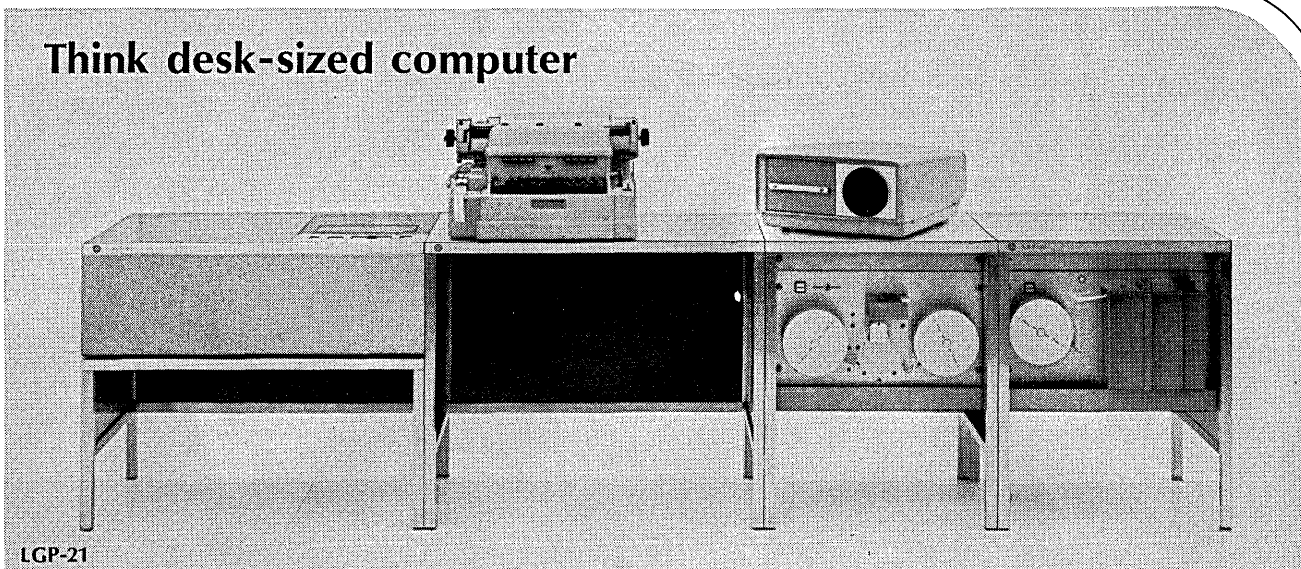
---

[5]We can think of two MAL developments still needed; both are mechanizations of parts of mathematical notation untouched by FORTRAN. One is the algebraic-symbol manipulator, the other the "interval arithmetic" processor that would permit the specification of the precision of numeric data, and indicate the precision of the answers. See Moore (1963).

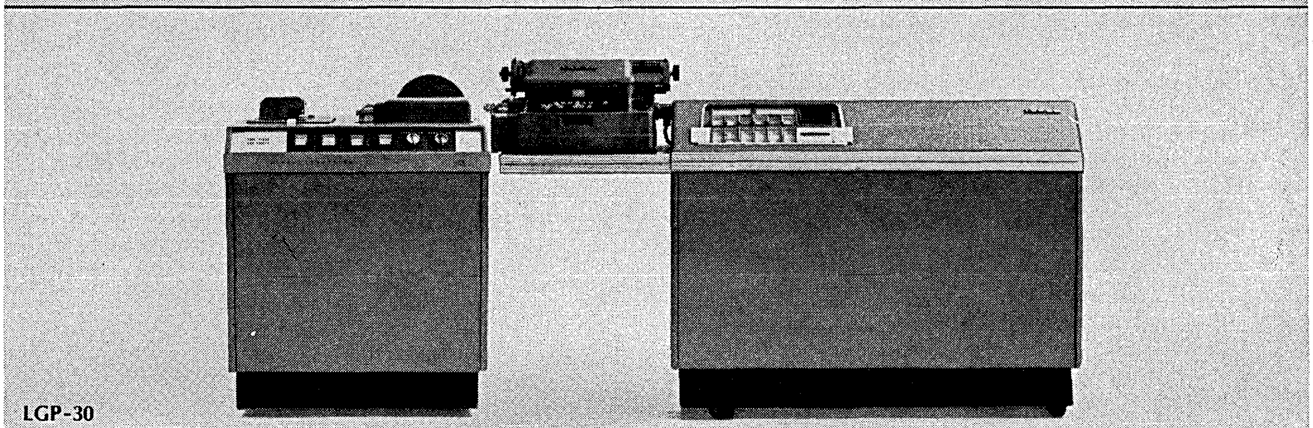
[6]Elbourn and Ware (1962, p. 1060) apparently join us in seeing the compiler as a variety of macro-processor, saying," . . . generally a compiler permits more complex macro-instructions than an assembler . . ."

[7]That the shape taken by FORTRAN is due to the very special, perhaps unique, circumstances surrounding its creation has never been a secret, but does not seem to be sufficiently appreciated. We have tried in an earlier paper [Halpern (1964)] to give some account of those circumstances. (To better document our regard for at least one compiler, we will mention that the paper cited, until an unfeeling editorial pencil struck, bore the title, "On the Economics of Using Computers: Reminiscences and Prophecies on the Occasion of FORTRAN'S Tenth Birthday").

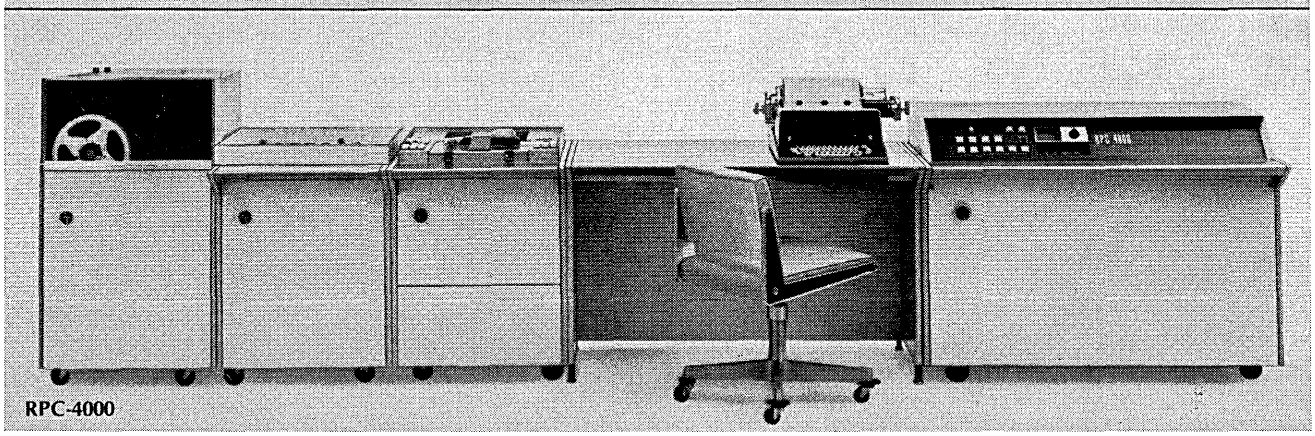
# Think desk-sized computer



LGP-21



LGP-30



RPC-4000

**Desk-sized and dollar-conscious... the LGP-21, LGP-30 and RPC-4000 general-purpose digital computers make electronic computing economically practical for almost every organization from small businesses to large corporations.**

LGP-21 COMPUTER
A low-cost, high-capacity computer for business and engineering applications. The LGP*21 can be used by present personnel for economic short-run or keyboard operation as well as for more complex problems.

*Trademark, General Precision, Inc.

LGP-30 COMPUTER
The first and still most widely used desk-sized computer. The LGP*30 is supported by one of the most extensive software packages available, including programs for accounting, payroll, engineering applications, etc.

RPC-4000 COMPUTING SYSTEM
A medium-scale system with many features ordinarily found only in larger computing systems. The RPC*4000 can process large volumes of data at speeds required for engineering and scientific calculations.

LIBRASCOPE GROUP
GENERAL PRECISION
COMMERCIAL COMPUTER DIVISION
808 WESTERN AVENUE • GLENDALE 1, CALIF.

Write today for illustrated brochures on the descriptions and applications of General Precision's desk-sized computers.

G43-121

CIRCLE 26 ON READER CARD

# NON-LINEAR PROGRAMMING

by E. SINGER

The single largest use of digital computers is for the solution of linear programming problems. Furthermore, if you boil down Operations Research, the largest lump on the bottom of the pot will be LP. Linear Programming has even achieved recognition from the nontechnical press. *Time* magazine considers it as one of the reasons for teaching mathematics in the grades,[1] "in a day of computers, automation, games theory, quality control and linear programming." But the men whose jobs are to make decisions, having at last been persuaded of the advantages of LP, are now told that the L (for linear) has to be changed to NL (for non-linear).

What is the fuss all about? For the uninitiated, let us clear the air. In the more familiar jargon of the computing trade, the "programming" of "linear programming" has nothing to do with coding (or for that matter with the computer) but rather with the application. "Linear *Scheduling* Problems" would be more comfortable for the computer *programmer*.

Here's the situation. A business man, oil refiner or military strategist (it's amazing how many personality types could be inserted here) is faced with making a decision. How much money shall he spend on advertising in magazine X? How much light cat-cracked gasoline shall be put in premium gasoline? How many ICBM installations shall be activated? The primary criterion for these decisions is the net "value" of each decision—the profit he can generate minus the cost of the ad, the increase in amount of premium gasoline minus the cost of production of the light cat-cracked gasoline, the protection for the country minus the impact on the economy of the country. Unfortunately there are secondary considerations that dictate the implementation of these decisions. These are the "Yes, but . . ." considerations. For our three examples the "Yes buts" would be: "Yes, but our total advertising budget is only N thousand dollars" . . . "Yes, but if you put in that much light cat-cracked gasoline the sensitivity of the gasoline is more than x" . . . "Yes, but it you activate that many ICBM's, the Navy will squawk." So the scheduling problem can be stated as, "Find those values of the decision variables which yield the highest value of the objective function provided that certain constraints are not violated."

Now that we have stated the mathematical programming problem, what's LP and what's NLP? Sometimes the effect of doubling the decision variable is to approximately double its "value" to the system. Twice the advertising in magazine X produces twice the sales (?), Twice the light cat-cracked gasoline doubles the profit from premium gasoline (?). Twice the ICBM's doubles the protection (?). Also the effect on the constraints is sometimes linear in the decisions: doubling the magazine X billing gets us twice as close to the limit of the budget, doubling the cat-cracked gasoline gets us twice as close to the sensitivity limit, doubling the ICBM's gets us twice as close to the Navy's squawk point. When this so-called linear approximation is sufficiently accurate, LP is a good way to answer the scheduling problems. When it's not (and more and more exceptions are being uncovered), we are in the soup and need an NLP.

Very elaborate and efficient computing systems have been set up to solve the LP problems. [2, 3, 22] The input and output of an LP run can be stylized for all sorts of applications from oil refining to high school class scheduling. The structure of the problem is always the same:

Maximize F where

$$F = \sum_{i=1}^{N} c_i x_i \qquad \text{subject to all} \qquad C_j = \sum_{i=1}^{N} a_{ij} x_i \geqslant b_j$$

Since each of the a's, b's and c's have easily statable significance, it is possible to prefabricate programming systems and report writers. At execution time the definition of each item can be read as data, and the report almost writes itself. Besides the I/O advantages there are two extremely nice properties of LP answers, i.e., the optimal slate of decisions. The first is that at the optimal there are as many constraints exactly satisfied as there are decisions to make. The other is that when you have found the answer you know you haven't gotten into a "blind corner." More about these properties later.

Many computer manufacturers and consultants have taken advantage of these properties of LP and produced highly efficient application software geared to get LP answers for extremely large LP problems . . . many decisions and many constraints.

However, as soon as you lose the linearity you lose all of these advantages. For example, many response functions (either objective or constraint functions) cannot even be stated explicitly. The design of a fractionating column is done by successive approximation; no one has been able to state explicitly the cost of the column as a function of the separations requested and feed rates and quality. Then, too, in a general non-linear problem there can be many "blind corners" and the answer can lie on any number of constraints from none to as many as there are decisions. Fig. 1 (A & B) gives an idea of this problem in a two-decision case.

Despite these difficulties inherent in NLP, a number of quite practicable techniques have been worked out, and it is worthwhile to review them. These techniques might be classified as basically analytical, direct search, or gradient search. All of these basic techniques have been attempted at one time or other on digital computers. The perfecting of analog and hybrid computers has introduced some exciting possibilities. We will discuss first the "older" digital methods and then touch on some of the suggested methods for the analog.

One of the most obvious solutions is to use basic calculus and algebra . . . *the Analytical Method.* The recipe

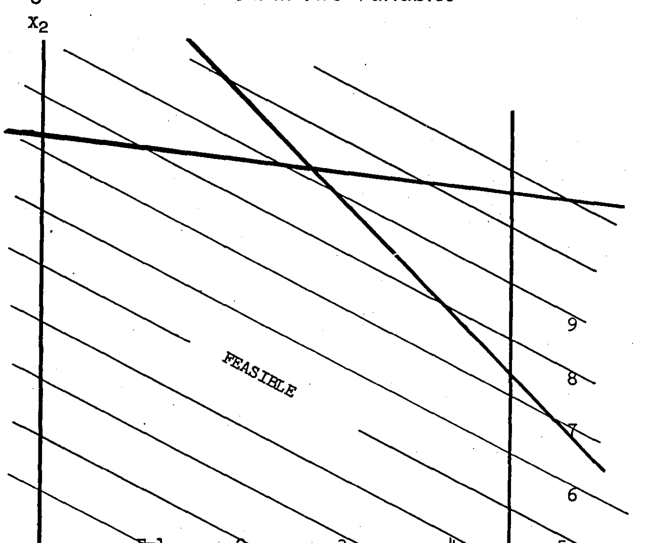
***Dr. Singer is president of Management Decisions Inc., a Houston, Texas, computer consulting firm engaged in the development of proprietary programs. He was formerly head of Technical Services for the Houston center of CEIR Inc., and previously spent 11 years with Shell Development Co. in studying the uses of digital computers. He holds degrees in chemical engineering from Columbia and Princeton Universities.***

goes as follows: Use elementary calculus and algebra to determine all proper maxima* possessed by the objective function. The highest of these may be the solution, if feasible. Next, determine constrained maxima† subject to each of the constraints. (Lagrange's method of indeterminate multipliers[4] is convenient for this). The highest of these may be the solution, if feasible and not surpassed in the earlier calculations. Next, do the same for local maxima subject to *pairs* of constraints. There are $(2m + n)(2m + n-1)/2$ such pairs. Next, take *trios* of constraints. There are $(2m + n)(2m + n-1)(2m + n-2)/(3*2)$ such trios. Then *quartets*, etc. Continue in this way until all local maxima on groups of constraints or bounds have been isolated. Then select the global maximum. (A brief description of this method is given in reference 4). The objections to this method are fairly obvious when one considers the amount of calculation required for any reasonable number of variables and constraints. For example in a four-variable, four-constraint problem there are almost 800 simultaneous equation problems to solve, many of which can be overwhelmingly complex. So much for the analytical methods. Next we can review the direct search methods.

The most direct of the direct search methods is to thoroughly examine the entire "space" of decisions. This is the so-called *Grid Technique* whose recipe is as follows: Space values at reasonable intervals for each variable (e.g., dividing the entire range of each variable into 10 parts), determine which grid points are feasible. Calculate F at each feasible member of the $(j_1 + 1)(j_2 + 1) \ldots (j_k + 1)$ resulting grid points (where $j_u$ is the number of intervals for variable $x_u$). Designate the point with the highest F value as the "first solution." If more precision is required, explore the vicinity of the first solution on a finer mesh than before (or else apply some interpolation device, such as a second-degree "hypersurface"). Continue this until satisfactory precision is obtained. Check the solution by making gradient or gradient-projection excursions. To reduce computing time when using this technique, eliminate portions of the grid whenever possible by considering analytical properties of the constraints. This technique is commonly called the "Case Study" in design work. The objections to this technique are:

1) There are an awful lot of points to look at. For example, a four-variable problem with 10 grid points over the range of each variable (10% accuracy) requires that we calculate the objective and the constraint functions 10,000 times. If we want to decrease the mesh size by 1/10 on a second pass (to get "accuracy" of 1% on the decisions) we have to do another 10,000.
2) If the mesh is too large, a valuable fish may slip through. See Fig. 2.

**Fig. 1A Linear Problem in Two Variables**



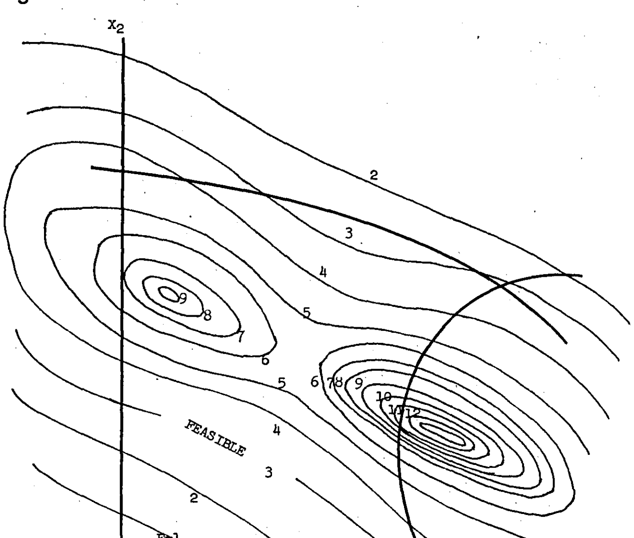

A technique which is closely related to the Grid is the *Cross Section*, whose recipe goes as follows: With fixed starting values of $x_2, x_3, \ldots x_k$, vary $x_1$ by preselected steps, spanning its range. At each feasible point, calculate F. Select the best F so far and set the corresponding value of $x_1$. Then vary $x_2$ in the same way, and set $x_2$ according the best feasible point so far. Continue in this way until a complete cycle produces no further change in F. As with the Grid technique, a finer mesh may be introduced after the final approximation is achieved.

There is a more sophisticated and efficient way to perform the cross-section search. The root idea of the Fibonaccian search method is to provide a criterion for eliminating large portions of "space" from consideration. It is possible, when the objective function is unimodal, to progressively delimit the area for examination until it is smaller than a preset tolerance. The arithmetic for this method is described in References 5, 6, 7, 8. Unfortunately, there are two severe limitations to this technique:

1) When non-linear non-trivial constraints enter the picture, the method fails miserably.
2) When the method is extended to more than a few decisions, the computing time becomes enormous.

The *Monte Carlo* techniques are somewhat akin to these already mentioned but more like the coward's way out. The recipe goes as follows: Use a random number table (or generator) to select a point in the bounded region. Retain this point if feasible, and calculate its F. Repeat for another random point. Retain the new point if its F is higher than that of the old point. Continue in this way for a predetermined number of points (calculated from probability considerations to give a preset confidence level). The last retained point is taken as an approximate solution. Check this solution by making gradient or gradient-projection excursions.

**Fig. 1B Non Linear Problem in Two Variables**




A variation of this method uses the retained point as a start for the next move. Then the direction of the move is determined at random, but the magnitude is fixed. If a certain number of moves does not produce an improvement, the magnitude of the move is decreased. See Reference 9 for some discussion of the possibilities of these methods.

The next major class of methods are those I have loosely

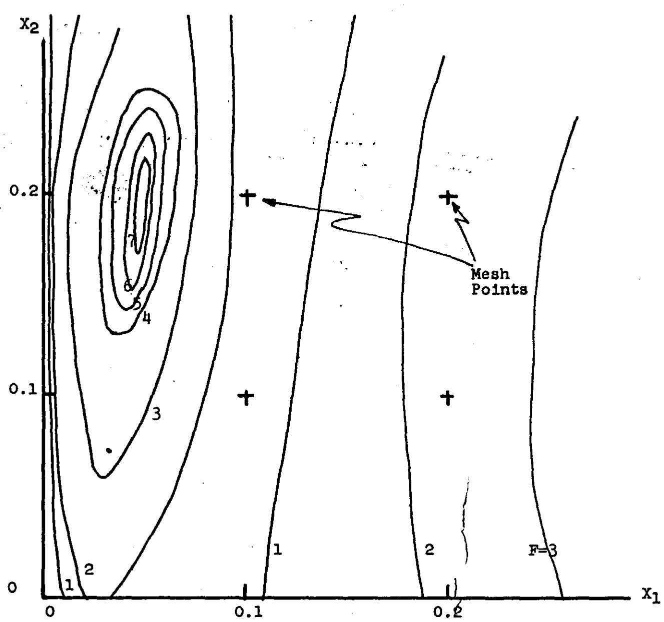
* A proper maximum is a maximum in the objective function, itself.
† One caused by constraints.

classified as gradient search. A very useful approach is to take advantage of the work done on LP when dealing with non-linear problems. What you do is to guess at the location of the optimum, make a linear approximation to the non-linear functions which is good in the vicinity of the assumed optimum, solve the resulting LP and check the solution against the assumption. Repeat the operation as desired. Many variations of this basic recipe have been developed, typical of which may be cited the MAP (Method of Approximate Programming) method of Shell Oil Company[10] and POP (Process Optimization Programming) of IBM.[12] It works very well, provided the functions do not have too much curvature nor too many changes in curvature. The results, when poorly behaved functions are attempted, is lack of convergence or convergence on a less-than-optimal answer.

This technique can be somewhat generalized by assuming that the objective function is a pure quadratic (the constraints must still be linear). Under these conditions the basic algebraic apparatus of LP can be used profitably and a step made in the direction of general non linearity [11, 11a]

The technique most suited to the solution of the generalized non-linear problem is a steepest ascent (descent) technique of one sort or another. The logic, algebra and computer implementation of this concept have been worked out in considerable detail for general non-linear objective functions. The capability for handling constraints, however, is still in the process of development. An analogy to the basic concept of steepest ascent is obtained if one considers how one might find the highest point on a mountain range using only an altimeter and a compass in a dense fog which cuts visibility to 10 feet. A set of instructions for doing this might read as follows. Find the highest point within 10 feet, proceed in the direction of that highest point for about 100 feet, and then make a new survey of the situation. The first thing to note is whether a higher point has indeed been reached. If so,

Fig. 2 Non Linear Problem in Two Variables—Mesh Too Large




make a new 10-foot survey. If not, retreat 50 feet and survey the situation at that point. If this point is an improvement, a new 10-foot survey is made, and a new direction derived. If not, another half-way retreat is made, and so on until no improvement is visible within 10 feet. A procedure somewhat similar to this was dubbed by Amundson[12] "a technique for stubborn mountain climbers" This works rather well, so long as there are no constraints.[13]

A very interesting approach to the problem of constraints has been used which entails the insertion of the constraints into the objective function. In other words, one seeks the high point in the objective function proper but penalizes the value of each slate of decisions for an approach to the constraints.[14] This often makes good sense in the context of the problem as well as for the mechanics of its solution. In the formal mathematical programming statement of a decision problem one completely disallows answers which "violate" constraints. This is tantamount to *infinite* penalty, which, for many problems, is quite unrealistic. For example, we say that premium gasoline sells for $x$ cent a gallon provided it has an octane number at least as high as $n$ and that it cannot be sold at all when it is less than $n$. One instinctively rebels at such rigidity. Although one less octane number makes the gasoline less satisfactory as auto fuel, it might be grabbed up at a price less than $x$ cents a gallon. Although this method of handling constraints is attractive, one must be cautious about setting realistic penalties and realizing that realistic penalty functions may create extremely steep contours which unduly complicate the mechanics of optimization.

Let us return to the more conventional statement of the problem, i.e., with distinct objective and constraint functions. One of the techniques for handling constraints is to project the gradient of the objective function onto the constraints as they are encountered. If we return to our mountain climbing analogy, we may think of the existence of fences or impassable barriers on the mountain range. When the steepest ascent runs us into one of these fences we must proceed along the fence in the direction of ascent. It is possible that there will be two fences on the mountain range so arranged that the path of steepest ascent takes us to the intersection of these fences and prevents us from reaching the very highest point in the mountain range. The implementation of these ideas is fraught with severe problems which have to do with the fact that non-linear functions can be non-linear in an awful lot of ways. First off, if we project the gradient onto the constraint and travel some finite distance in a straight line, how can we be sure we are still on the right side of the fence? Maybe the fence curves inward towards the feasible region. If it does we are bound to be on the wrong side. How do we get back? Another problem has to do with "finding" the fence. The constraint functions which define the "fences" can be weird and/or implicit. Finding an intersection between a line of travel and such a difficult constraint may be as much of a job as finding the optimum.[15, 16, 17, 18] The methods are still evolving and important progress is in the making.

The advent of sophisticated analog computers with their special kinds of memory, rapid calculation potential and improved reliability has presented some exciting possibilities. For example, the Grid technique which requires so many calculations of the objective function is not nearly so frightening on an analog as on a digital machine. One might put variable 1 on a rapidly increasing "ramp" in repetitive operation, variable 2 on a slower one, 3 still slower, etc. Then only the best, feasible solution and its location need be saved. *Voila!* a global optimum! Caution 1: the answer will be located with an accuracy equal to the ratio of successive ramp speeds, times the range of the slower of the two variables. Caution 2: as so often happens with an analog, you may run out of electronic gear before you have the simulation complete. Solution: go to a hybrid.

There's another technique potentially usable on the digital, but more particularly adaptable to the analog.[19] The recipe is to set all but one of the decision variables on ramps of increasing speed. Set the objective function at an arbitrary value. Solve for the remaining variable by analog means. Calculate the constraints continuously. During the time that the solution is feasible, increase the objective function (or decrease, depending on whether you are maximizing or minimizing). When the objective function no longer moves, you are at a completely constrained optimum. This method gives global optima even when there are several constrained local optima of varying excellence. Caution: the method will not work on partially constrained optima (where the optimum lies on fewer constraints than the number of decision variables).

Well, where do we stand now? How do we decide what techniques to use? Unfortunately there are no hard and fast rules. Past experience is still the best guide. One can say, however that:

(1) When the number of decisions or constraints is over 25 *and* the functions are well represented as linear, use LP methods. You can get the report writers and canned algorithms free from computer manufacturers. What is "well represented"? No violent points of inflection. Less than 20% error in extrapolating over half the range.

(2) When there are a very large number of decisions and extremely non-linear functions, use Monte Carlo methods perhaps combined with direct search.

(3) In the large no-man's-land, one must lean on past experience to choose from among the Grid Techniques, Cross Section, LP and Gradient Search.

Each particular NLP problem has a preference for one or another of the techniques described. By preference, we mean speed of solution and confidence that a solution can be found. But for a general purpose program, the opinion of Philip Wolfe,[20, 20a] expressed in 1957, is probably still valid: "Finally, least efficient in speed, but most powerful in the variety of problems they can handle are the creeping methods" (i.e., steepest ascent with projection on constraints). ■

## REFERENCES


1. *Time Magazine*, January 31, 1964, p. 34.
2. Gass, S. I., *Linear Programming*, McGraw-Hill, 1958.
3. Charnes, A.; Cooper, W. W.; Henderson, A., *An Introduction to Linear Programming*, John Wiley, 1953.
4. Sokolnikoff, I. S. and E. S., *Higher Mathematics for Engineers and Physicists*, p. 163, Second Edition, McGraw-Hill, 1941.
5. Kiefer, J., *Proc., Am. Math Soc 4*, 502, 1953.
6. Wilde, D. J., *Advances in Chem. Eng.*, Vol. 3, p. 277.
7. Krolak, P.; Cooper, L., *Comm. ACM6*, 639, 1963.
8. Kaupe, A. F., *Comm. ACM 7*, 38, 1964.
9. Brooks, S. H., *Oper. Res. 6*, 244, 1958.
10. Griffiths, R. E.; Stewart, R. A., *Management Sci. 7, #4*, 379, 1961.
11. Wolfe, P.; Frank, M., *Naval Res. Log. Quar 3*, 95, 1956.
11a. Wolfe, P., *Econometrica 27*, 382, 1959.
12. Amundson, N. R., AIChE Tenth Annual Institute Lecture, Cincinnati Meeting, December 1958.
13. Zellnik, H. E.; Sondak, N. E.; Davis, R. S., *Chem Eng Prog 58*, 35, 1962.
14. Carroll, C. W., *Oper. Res. 9*, 169, 1961.
15. Rosen, J. B., *J Soc Indust Appl Math 8*, 181, 1960.
16. ______________, *J. Soc Indust Appl Math 9*, 514, 1961.
17. Singer, E., *Chem Eng Prog Symp, Series 58*, No. 37, p. 62, 1962.
18. *Non-Linear Programming*, Management Decisions Inc., Houston, 1963.
19. Andrews, J. M.; Jannasch, C. F.; Moore, C. E.; Munson, J. K., Paper presented at AIChE meeting, Houston, December 1963.
20. Wolfe, P., "Computational Techniques for Non-Linear Programs," Princeton U. Conf. on Linear Programming, March 1957.
20a. Graves; Wolfe (editors), *Recent Advances in Mathematical Programming*, McGraw-Hill, 1963.
21. Sauer, R. N.; Colville, A. R. Jr.; Burwith, C. W., *Petroleum Refiner 43*, No. 2, p. 84, February 1964.
22. Dantzig, G. B., *Linear Programming and Extensions*, Princeton Univ. Press, 1963.

# BOTHERED BY THE HIGH COST OF *LIVINGa?

NORSCAN™ will help—it's an economical common language data acquisition, data transmission and data recording system which North Electric Company custom assembles from low-cost, highly reliable telephone switching modules. NORSCAN Central Control Units can dial directly to NORSCAN Remotes over the telephone common carrier switched net—no need for expensive full time communication circuits. One NORSCAN Control can dial up, command and receive data from up to 999 Remotes. For more on NORSCAN call Product Manager, Data Systems, at Galion, Ohio.

***LIVINGa** = Logging Individual Variables In Non-adjacent Geographical areas.

OTHER DIVISIONS:
DEFENSE SYSTEMS—*Electronics • Command and Control Systems*
POWER EQUIPMENT—*Custom Power Systems • Battery Chargers*
TELECOMMUNICATIONS—*Public Telephone Systems & Equipment*

**NORTH ELECTRIC**
**ELECTRONETICS DIVISION • GALION, OHIO**

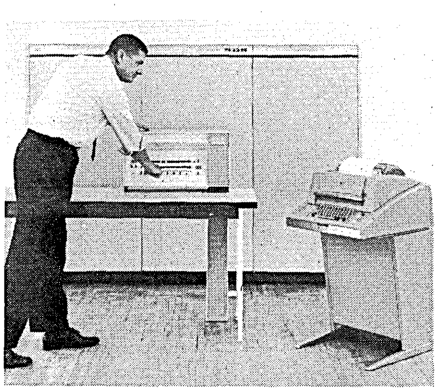
CIRCLE 29 ON READER CARD

# THE SDS 925 & 92

No sooner had they finished scoping the new SDS 930 but Scientific Data Systems announces two more main frames, bringing to six the number of computers offered by the Santa Monica, Calif., firm. Tucked into the middle of its line is the $80K SDS 925, and tagged onto the bottom is a $29K SDS 92. All six are said to be compatible.

The 925 has a 1.75-usec cycle time and a 3.5-usec add time. Memory size is 4-16K (24 bits plus parity) words. Add a few I/O instructions to the command structure of the 910 and you have it for the 925. Likewise, the I/O system resembles that of the 930 and 9300, consisting of time-multiplexed communication channels (250,000 words per second) and direct access channels (500,000 words per second). The priority interrupt system provides up to 1,024 channels of interrupt, each with an address in memory. The software package includes FORTRAN II and ALGOL.

The SDS 92 has a 12-bit (plus parity) word length and a 2.1 usec memory cycle time. Add time is 4.2 usec, and multiply/divide hardware is optional. Memory size: 2-32K. The I/O is with Magpak, the stereo tape cartridge system, and the transmission rate is 238KC. Up to 256 levels of interrupt are available as optional equipment. ■



# WESCON '64

## convention preview

Information-processing-oriented papers will be among those given at the four-day Western Electronic Show and Convention (WESCON/64) in Los Angeles from Aug. 25-28. Technical sessions will be at the Statler Hilton Hotel, and the exhibits at both the L. A. Sports Arena and Hollywood Park—with registration possible at all three locations. It is sponsored by the San Francisco and Los Angeles sections of the IEEE and the Western Electronic Manufacturers Assn. Computer-oriented sessions include the following:

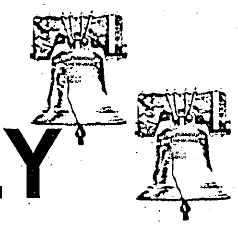
**Information Sciences,** Aug. 27, 2-4:30 p.m. Papers in this session are: "Multiple Access Computing," Robert M. Fano, MIT, Cambridge, Mass.; "Optical Data Processing," Louis Cutrona, Conductron Corp., Ann Arbor, Mich.; "Seismic Detection," Paul Green, Lincoln Laboratory, Lexington, Mass.

**Learning Systems,** Aug. 25, 9:30-noon. "Some Uses and Misuses of Bi-Computer Philosophy," M. L. Babcock, Univ. of Ill.; "Training a Threshold Logic Unit with Imperfectly Classified Patterns," R. O. Duda, R. C. Singleton, Stanford Research Institute, Menlo Park, Calif.; "A Multi-Layer Learning Network," R. A. Stanford, Philco Research Laboratory, Newport Beach, Calif.; "Learning Systems In and Out of the Factory," R. M. Stewart, Space General Corp., El Monte, Calif.; "Generalized Learning Theory," R. E. Jackson, S&ISD, North American Aviation Inc., Downey, Calif.

**Pattern Recognition,** Aug. 26, 9:30-noon. "Experimental Evaluation of a Rational Property Selection Technique for Pattern Recognition," M. G. Spooner, C. W. Swonger, J. B. Beach, Cornell Aeronautical Laboratories, Buffalo, N. Y.; "Computer Recognition of Hand-Written First Name Signatures," F. N. Marzocco, System Development Corp., Santa Monica, Calif.; "Aerial Photographic Pattern Recognition," N. C. Randall, Philco Corp., Blue Bell, Pa.

To accommodate those at the exhibit areas—the Sports Arena and Hollywood Park—viewing rooms will be set up and closed-circuit TV systems will telecast the sessions. Two-way voice communications will enable exhibit-viewers to participate in the sessions.

# THE ACM NATIONAL IN PHILLY



Technical sessions on language processors, standardization, the computer's effects on management and labor, and real-time, operating, and time-sharing systems are among those scheduled for the upcoming ACM national conference in Philadelphia, Aug. 25-27. The sessions and exhibits will be housed in the Sheraton Hotel.

Sponsored by the Assn. for Computing Machinery, the conference will have registration fees of $9 for members ($7 before Aug. 25), $25 for nonmembers, and $3 for students. It will feature as luncheon speaker on Wednesday the board chairman of IBM, Thomas J. Watson Jr., who will be preceded by incoming ACM president George E. Forsythe, director of the Computer Science Div. and Computing Center, Stanford Univ.

Other technical sessions will cover Pattern Recognition, Numerical Analysis, Programmer Training, Formal Manipulation of Mathematical Expressions, Information Retrieval, Administration of University Computing Centers, Computers & Communication, and Theoretical and Invited Papers.

Panel discussions will cover The Computer Revolution: Its Effects on U.S. Business and Labor, Computer Science Curricula, and Computer Education for the Management, Behavioral and Biological Sciences. In the Hall of Discussion: the Special Interest Group on Information Retrieval.

Conference chairman is Howard Bromberg of CEIR Inc., Jenkintown, Pa., and technical program chairman is Herbert S. Bright of Philco Corp., Willow Grove, Pa. The conference proceedings, to be distributed to registrants, will also be available from the ACM at 211 E. 43rd St., New York City 10017. ■

# "Glad we bumped into you," Marv said as I rubbed my bruises. "I want you to meet Carson Maim, who is coming to work for us. Carse, this is Paul Daly, our senior research engineer in charge of data reduction."

"My pleasure!" An over-diameter young man with a high C.G. grabbed my hand and ground it into what felt like a gruesome tangle of pulp and splintered bone.

"I'd appreciate it, Paul," Marv went on, "if you'd show Carse the D.R. system."

"Glad to," I gritted between clenched teeth. "I've just got time before I'm due at Emergency Hospital."


"Fun-nee," Marv said. "Carse, would you wait over there for a minute?"

"Glad to," said the megawatt power supply, moving off down the aisle.

"What's up, Marv?"

"His bowling score. Up around two twenty."

I was impressed. "Makes up for losing Evans, huh?"

"We had to do something to stay on top of the league. Now listen, Paul. Carse may not be much of a data handling engineer, but I'm counting on you to turn him into one before Mr. Robertson gets around to looking him over. Otherwise we'll be minus our only chance of snatching that trophy for keeps."

I started the guided tour at the tape transport cabinets.

"We receive the FM-FM flight test telemetry on these tapes," I told Carse.

He made a noise that sounded like "nngg." I hoped it meant I was getting through to him.

"First we put it through these discriminator banks and this analog multiplexer."

"Nngg."

"Next," I said, "we come to the Analog-to-Digital Converter where the pulse-amplitude data is digitized before it goes to the buffer that—Woops! I beg your pardon, Mr. Robertson, I'll get right out of your way."

"Wait, Daly," the Chief said as I made a futile attempt to keep Carse out of sight. "Fanning and I were just coming to look for you. Who's this young man?"

Well, there goes the league trophy, I thought.

"This is Carson Maim," I said. "Our new associate data handling engineer. Carse, meet Mr. Robertson, Vice President in Charge of Research."

The Chief stuck out his hand and said, "It's a pleasure to mmmmwwwooowww!"

"Sorry, Mr. Robertson," I said, dragging Carse back. "He was just trying to be friendly."

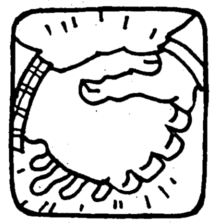
"I see," said the Chief, rubbing his hand. "As I was saying, Fanning seems to be having trouble mating the buffer to the A-D Converter. One minute it's in sync and the next minute it's either ahead or behind."

I said, "Why don't we—"

"Just a minute!" the Chief said with a vengeful gleam in his eye. "Let's see what this young man has to say about it."

"Nngg," said Carson Maim.

Oh, brother, I thought.



"Well?" the Chief said.

"Nngg, with this kind of an A-D Converter, you only have a two or three microsec interval in which the Converter output register contains a legitimate word. Then it starts being replaced by the next word, a bit at a time."

I found my mouth was hanging wide open, and managed to get it closed.

"Go on," the Chief said, looking interested.

"I'd say as long as the buffer only has such a short time to grab off the word, you're going to have problems."

"And you know some trick for lengthening the interval?" the Chief challenged him.

"Sure," Carse said. "If you were using the A-D Converter EECO makes, you could plug in a second output register. Then each word transfers to the second register and stays available while the first register is filling up with the next word. It gives the buffer at least eight times as long to snatch each word, and lets you process data faster."

"EECO, huh?" the Chief said. "I know their products have a good reputation. How's the price?"

"Their A-D Converters start at under two K,"[1] Carse told him.

"Then what's the tradeoff?"

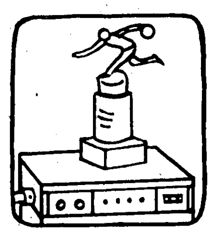
"None that I know of. In fact, EECO ADC's have some other extras. For instance, if an external trigger command is received during the conversion cycle, an EECO ADC will store it until the cycle has been completed, and then act on it. And you can get their converters with a gated display for looking at one channel at a time. In case you want to moniter the signal from a certain transducer."[2]

"That's good?"

"I leave it to Mr. Daly."

"It's good," I agreed.

"Also," Carse went on, "every EECO A-D Converter has an automatic mode. This gives you continuous digital display of the incoming analog data. It sure simplifies calibration and test procedures when you're trying to isolate problems in a system. You see, EECO has done a lot of system work. That's why they have this interface approach that eliminates a lot of problems before they start."



"That right, Daly?" the Chief asked me.

"It sure is," I said. "In fact, we've got an EECO ADC on order right now." We didn't, actually, but we would have as soon as I could arrange it, if Carse's facts checked out.

They did.

"Carse, where did you learn so much about EECO A-D Converters?" I asked him that night as we watched Marv trying to pick up a split. "Did you use to sell them?"

"No. I bowled in the same league with the EECO team. We beat them pretty bad, but I learned a lot about A-D Converters[3] in between turns. There comes your ball, Mr. Daly."

By the way, when you're in our neighborhood, drop in and see our bowling trophy. Also our EECO A-D Converter. They're both permanent.

1 He was talking about the EECO 761 ADC with resolution to 10 bits binary or 12 bits BCD. Converts at 18 KC at max. resolution. If you need higher speed, the EECO 760A ADC gives resolution to 12 bits, converts at 44 KC, is priced from $3700. That's the best deal we've heard of—how about you?

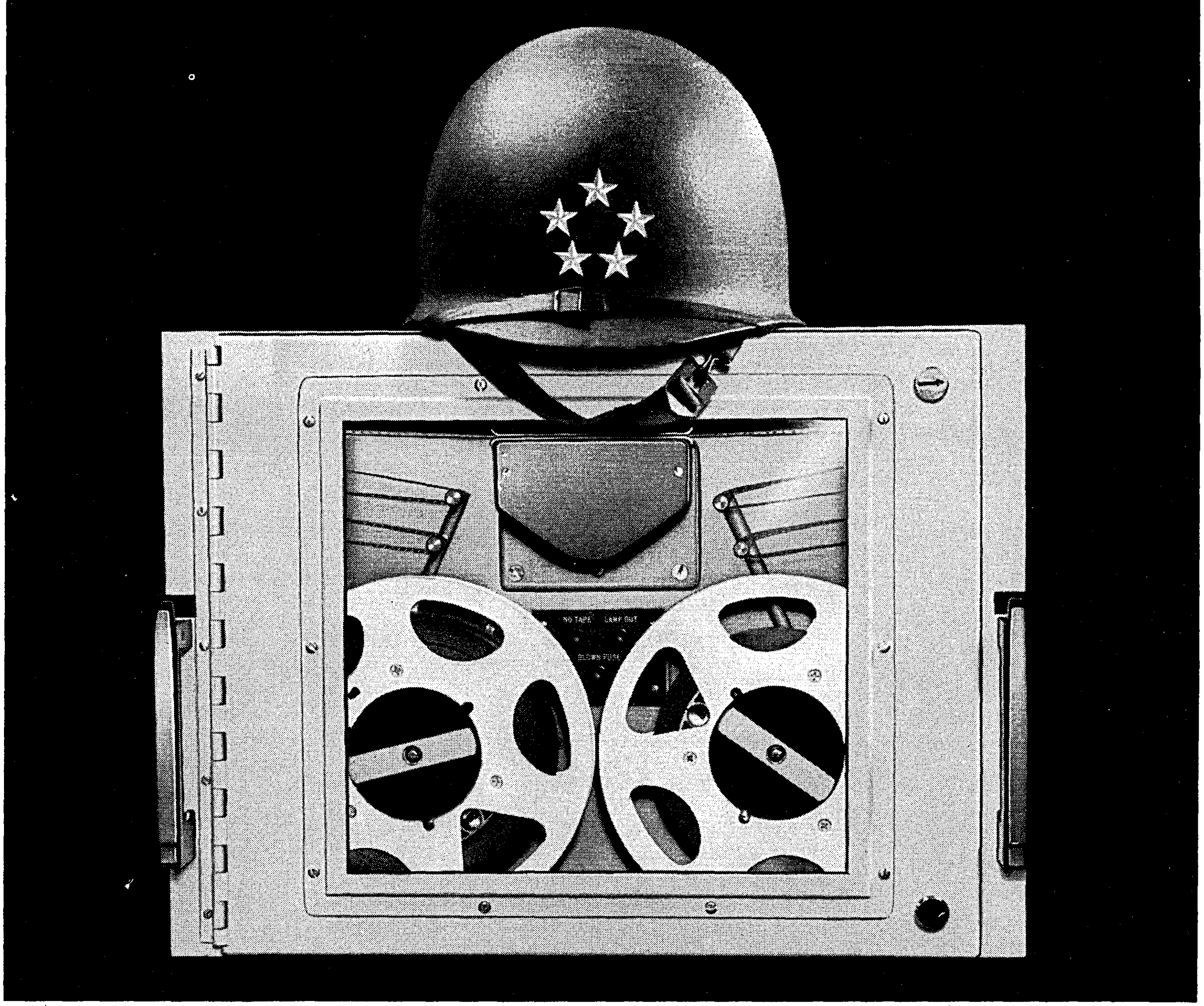
2 Also ask about EECO Multiplexers with simultaneous sample and hold for use with EECO ADC's.

3 So can you. Circle reader card number for full information on the EECO 761 Analog-to-Digital Converter. You'll also receive **FREE** vellum of engineering drawing and specifications of Terrific Terri the Test Engineer. **ADULTS ONLY** or permission of indulgent parent.
EE 4-5R

ELECTRONIC ENGINEERING COMPANY *of California*

CIRCLE 32 ON READER CARD

# TOP MILITARY PERFORMER



# The new fully-militarized 500 RM Tape Reader

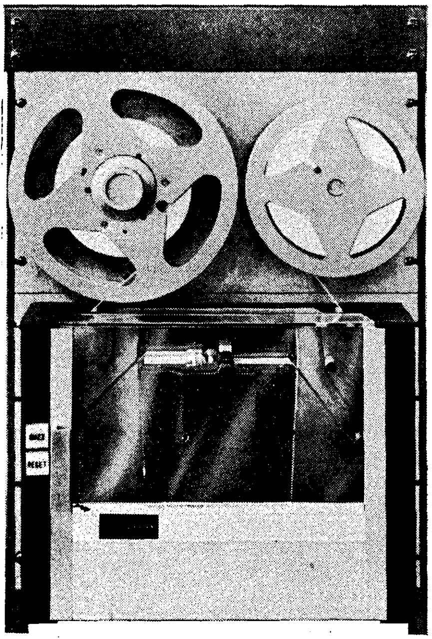
At last! . . . a reader conceived, designed and built specifically for military applications — the tape reader that meets all applicable MIL specs including requirements for low temperature operation.

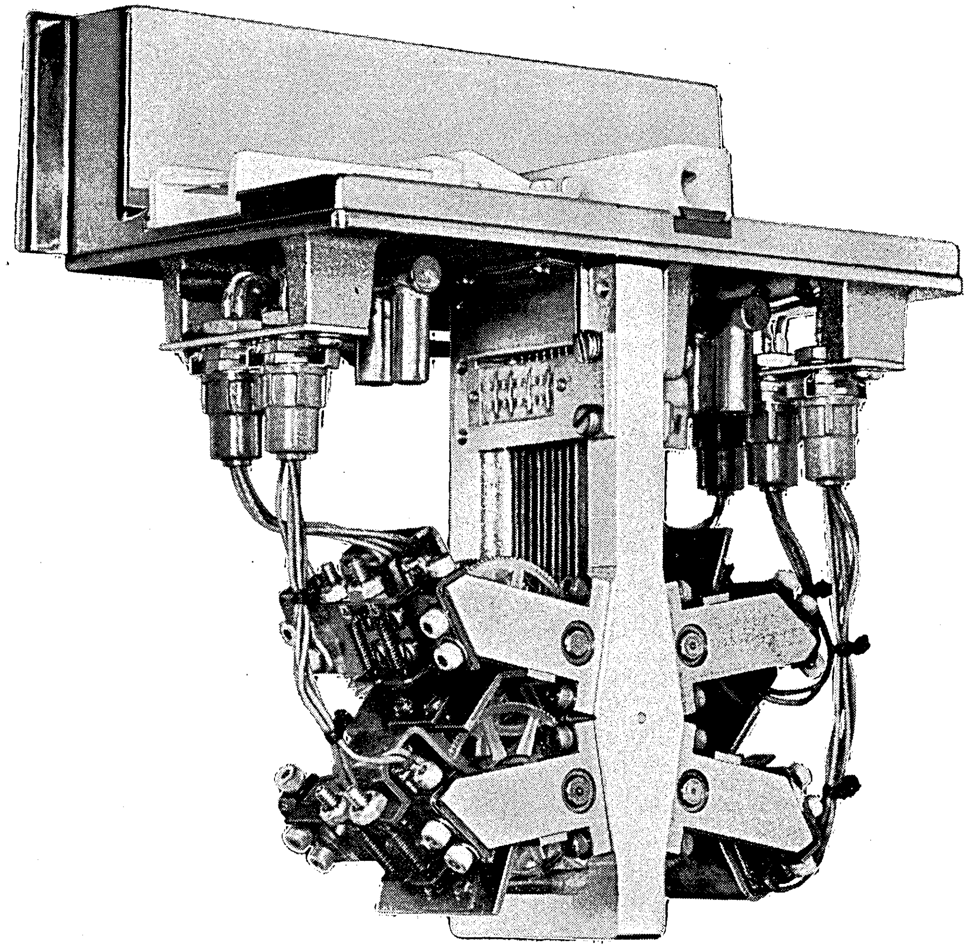
It's Photocircuits' new high-speed 500 RM . . . the fully-militarized photoelectric reader that gives you reading speeds up to 1,000 char/sec. and 8″ reels for maximum data storage. The new 500 RM's exclusive printed-motor direct-capstan drive eliminates completely all the brake and pinch-roller problems typical of today's hybrid or conventional units — giving you accuracy plus reliability unequalled by any ordinary tape reader.

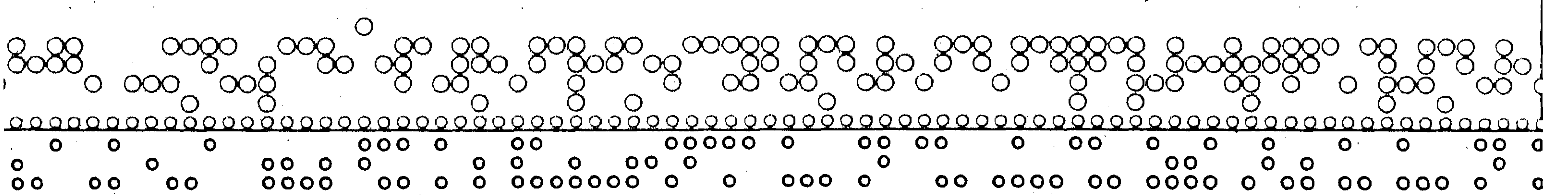
Details and specifications on the fully-militarized 500 RM are yours for the asking . . . or, if commercial/industrial applications are your interest, ask about Photocircuits' new high-speed 500 R. Write today to:

**Photocircuits**
CORPORATION
**TAPE READER DIVISION**
GLEN COVE, NEW YORK

CIRCLE 33 ON READER CARD







# *You can trust the P-150; it corrects its own mistakes*

All of Tally's perforators (over 3,000) have made mistakes. We haven't built a perfect machine and neither has anyone else.

But Tally has built the next best thing—a perforator that can tell when it makes a mistake and can correct it—all at 150 characters per second.

Tally calls this new perforator the P-150. Even before we added the self-correction feature, it was a dependable machine. Now it's by far the industry's most trustworthy high speed perforator.

**How You Can Use It**

Computer and machine tool programming, photo composing, and many data logging applications demand accurate paper tapes. With the P-150, you don't need to design a system which includes post reading the tape for error or going through a cleanup routine which takes time and more equipment. Instead, you can design around the perforator's accuracy. Tally gives your system these lines to communicate with the perforator: (1) 8 data lines and a sprocket line; (2) a line to introduce the parity test signal; (3) two lines to check parity condition, one odd and one even; (4) a line for advancing tape; (5) a line for reversing tape.

With the P-150 your system can produce clean tape in less time, with less equipment, and at lower cost.

**The Price is Right**

The cost of the industry's most trustworthy perforator is: mechanism, $2,850; drive electronics, $1,400; tape handler (1,000-foot), $450.

For additional details on how Tally's P-150 perforator can solve your design problem, please write to Tally Corporation, 1310 Mercer Street. Seattle, Washington 98109.

TALLY®

See us at WESCON
Booth 1132

CIRCLE 34 ON READER CARD

# NEWS BRIEFS

## UNIVAC GETS ENIAC PATENT, BEGINS LICENSING TALKS

Following the issuance this year of an "all-inclusive" patent on its ENIAC, Univac's Sperry Rand Corp. has formed Illinois Scientific Developments Inc., Chicago, to "promote, hold, and do research in the area of patents." The firm is presently negotiating with unnamed computer manufacturers for a licensing arrangement covering patented computer design features.

ENIAC (electronic numerical integrator and computer) is generally credited as the world's first electronic digital computer. Completed in the fall of '45 at a cost of some $500K, it went on the air the following January at the Ballistic Research Labs of the Aberdeen Proving Ground in Maryland. Original conception and design are credited to Dr. John W. Mauchly and J. P. Eckert, Jr. The latter is now a Univac vp.

## 1st DP TRAINING BEGINS UNDER NEW MANPOWER ACT

Some 35 unemployed and partially employed residents of the Washington, D.C., area have been selected for the first 7-9 month course in dp to be given under the U.S. Manpower Development and Training Act. Selected from among hundreds of applicants, the students (whose ages range from 18-46) will receive EAM and computer operator and programmer instruction while receiving subsistence payment from the Dept. of Labor.

The students, mainly Negro and including four women, will receive math and English instruction for two hours daily in the D.C. public schools, and 10 weeks of introduction to EAM equipment. Following this, and depending upon their qualifications, they will go into tab or computer operator courses and, after more screening, programming and analysis.

The dp instructors are being donated by Datatrol Corp., Silver Spring, Md., originators of the project. Datatrol and Giant Food Stores have stated a willingness to accept graduates of the course, necessary for the project to qualify for federal financing under the MDT act.

## LIBRARY OF MEDICINE GETS PHOTO-TYPESETTER

The phototypesetting machine that people didn't see on the SJCC tour through the National Library of Medicine in Bethesda, Md., is there now. In the MEDLARS information retrieval system, the machine will set type in page form for a monthly index of medical articles.

The Photon 900 operates at up to 552 cps, or 30,000 characters per minute, from a 264-character set of varying fonts and sizes. Maximum line length is 8 inches. The system consists of photo and control units and a paper or mag tape handler, and the copy can be on film or paper, in positive or negative, and be right or wrong reading.

CIRCLE 100 ON READER CARD

## NCR, UNIVAC ANNOUNCE AIRLINE SYSTEMS

A two-megabuck combination passenger reservation and teletype message switching system has been ordered by Northwest Orient Airlines. Central hardware will be a Univac 490, which will replace a Univac system in operation since 1959. With the new system, the inventory transaction rate will increase from the present 3,600 per hour to more than 30,000. On-line will be 187 agent sets in 14 cities, contrasted with the thousand-or-so in the American Airlines/IBM SABRE system. As before, reservation offices in smaller cities will communicate with the system by teletype, except that the information will be input automatically instead of manually.

NCR's system for Pacific Southwest Airlines consists of a 315 with 20 inquiry machines in the next room. Ticket agents in the boondocks must phone in, inquiries are made to CRAM units, and, *voila*, PSA has another SRO flight.

## BURROUGHS, CDC GET 8-, 13-MEGABUCK ORDERS

An order for five D825 computer systems has been placed with Burroughs Corp.'s Defense and Space Group by the Navy's BuShips. The 8.3-megabuck order is for hardware developed for the Naval Research Lab in 1962.

Control Data Corp. will deliver nine 160G computer systems to NASA for its Apollo (lunar landing) spacecraft acceptance and checkout program. Under the 13.5-megabuck contract, dual computer systems will go to North American Aviation, Grumman Aircraft, NASA Houston, Cape Ken-

## ISO PROPOSES WORLDWIDE STANDARDS IN THREE AREAS

Worldwide standards on optical and magnetic ink characters and an information interchange code have been recommended for adoption by its members by the International Organization for Standardization (ISO). The U.S. member is the ASA, American Standards Assn. ISO's Technical Committee 97 on computers and information processing, meeting for the first time in the U.S., recommends a six-bit (64 characters) and seven-bit (128 characters) code which is said to be generally compatible with the American Standard Code for Information Interchange (ASCII), approved last year by the ASA.

In the magnetic ink area, two type fonts are being proposed: the E13B presently in use in the U.S., and CMC 7, a powerful bar code with both symbols and alphanumerics and currently big (but not standard) in Europe—especially among postal services.

Four standard character sizes in standard shapes have also been defined to allow optical reading of imprints from credit cards, cash registers, typewriters, and line printers.

Agreement on a set of flowchart symbols came close to achievement. A subcommittee was asked to circulate a first-draft of proposed symbols for eventual submission to TC 97. Drawn up was a list of 32 symbols, ranging from "punched card" to "parallel mode."

## NEWS BRIEFS . . .

nedy, and General Electric, and will reportedly insure compatible checkout procedures up to launch-time.

● It would take one man two to three million years to match a year's output by one big computer, using conservative estimates, according to Dr. Michael A. Melkanoff, chief of programming at UCLA's computing facility. This is based on one machine-second being equal to one man-month —the man working an eight-hour day, and a 7094 on three shifts. Assuming a $7K/year man, it reportedly would cost $35K to equal the $2 output (one minute) of the computer.

● Prof. George E. Forsythe of Stanford Univ. has been elected president of the Assn. for Computing Machinery for the two-year term, 1964-66. Elected vice president is Herbert S. Bright of Philco Corp., and Eugene H. Jacobs of System Development Corp. is secretary. Elected to four-year terms as members-at-large were Paul Armer, RAND Corp., Bruce Gilchrist, Service Bureau Corp., and Anthony G. Oettinger, Harvard Univ. And serving the last half of their terms are Robert W. Bemer, Alston H. Householder, and David M. Young, Jr.

● A 16K (72 bit) word core memory with a cycle time of one usec has been delivered by Ampex Computer Products, Culver City, Calif., to the Army Ballistic Research Labs, Aberdeen Proving Grounds in Maryland. It becomes the central storage in the BRLESC computer, and will be followed by delivery of two more.

● Four peripherals for its 315 computer have been announced by National Cash Register Co., Dayton, Ohio. They are a 1,000 lpm printer (120 characters), 66 and 120KC mag tape units (556 bpi) with a file adapter feature, and a 250 cpm card punch.

● The fifth move to expanded facilities in as many years has been announced by FMA Inc., Los Angeles. From a three-man, 1,000-sq-foot operation in '59, the firm has grown to a 260-man, 100,000-sq-foot firm.

● Ownership of Holley Computer Products Co., formerly shared by Holley Carburetor Co. and Control Data Corp., has transferred wholly to CDC. In exchange, CDC gives an undisclosed amount of its stock to Holley Carburetor. The Computer Products firm makes drum printers, now will share R & D activities with CDC's Peripheral Group.

● The Air Force Academy at Colorado Springs, Colo., is installing a B5000 in its Seiler Research Lab. In addition to supporting work for the AF's Office of Aerospace Research, the hardware reportedly will make possible a required course in computer applications for all cadets. The 24K system includes two drums and four tape drives.

● Programmers as you need 'em, when you need 'em, trained on your kind of machine and industry are being offered by a New York City company, DATAtemp Inc. Some 500 programmers reportedly are on the staff, apparently the largest undertaking of this type of service. The firm says there are 1,000 computers and 10,000 programmers in the metropolitan area, the former figure to double by '66.

CIRCLE 101 ON READER CARD

● A 7090 program for selecting the best routes for intercity mail has been written by the National Bureau of Standards, and a 1401 program for routing airmail has been announced by the Post Office Dept. The 90 program is capable of making route selections from among 2,000 trip segments, including up to 80 transfer points.

● A device to keypunch or type the Chinese language has been developed by the Itek Corp., Lexington, Mass., under a study grant from the Research & Technology Div. of the AF Systems Command, Rome Air Development Center. The Chicoder is capable of producing over 10,500 ideographs from a keyboard with only 36 top and 32 bottom features of a character. Following the keying in of these features, a typist has a family of ideographs displayed to her—from which she selects the one she wants. Speeds above 40 characters per minute by a typist unfamiliar with the Oriental language, and following little training, are claimed. Hardcopy and paper tape output are produced.

● First delivery of the PB440 has been announced by Packard Bell Computer, Santa Ana, Calif. It is the central element in a simulation system designed by the IIT Research Institute, Chicago, for the human factors lab at Wright-Patterson AFB, Ohio. Subsequent deliveries of the 440 have been made to NASA-Huntsville and the Navy's weather research facility in Norfolk, Va.

● A 10-year study of the effects of technological change and automation on individuals, the economy, and government will be undertaken by Harvard Univ., the result of a 5-megabuck grant from IBM. Among areas to be looked into are methods of forecasting technological developments and new advances in automation, and changing skill requirements and educational policy. All you professional students, get in line.

● First installation of a GE 415 at NASA-Cape Kennedy follows by 60 days the hardware's announcement. No mention is made of the incorporation of the Capacitrix, built-in 1401 translator.

● A Detroit firm, Haig G. Neville Assoc., is underwriting insurance for service bureaus, designed to cover legal liability arising from errors and omissions, cost of reassembling destroyed tapes, cards, etc. This type coverage was recently negotiated by ADAPSO with Lloyd's of London.

CIRCLE 102 ON READER CARD

● Third installation of an S-C 4020 recorder in Europe is by the French power company, Electricite de France. The General Dynamics/Electronics device will be used to print and plot off a 7094, and for studies of the French high voltage transmission network. Other installations: the British and French atomic energy groups.

CIRCLE 39 ON READER CARD

# people IN DATAMATION

■ Dr. Louis T. Rader has rejoined GE as vp and general manager of the Industrial Electronics Div., composed of five departments, including the Computer Dept. He was president of Univac Div. of Sperry Rand Corp. since 1962, and is replaced by Frank Forster.

■ Joe Weizenbaum, who has been on leave from GE at Project MAC, has accepted an appointment as a resident Associate Professor of Electrical Engineering at MIT. He will continue his association with the time-sharing research project there.

■ N. Louis Senensieb of TRW Space Technology Lab., Redondo Beach, Calif., has been elected president of the International Systems and Procedures Assn. Named vp is Allan L. Burns, consultant for Peat, Marwick, Mitchell & Co., Dallas, and Michael Miskulin of Computer Methods Corp., White Plains, N.Y., is secretary.

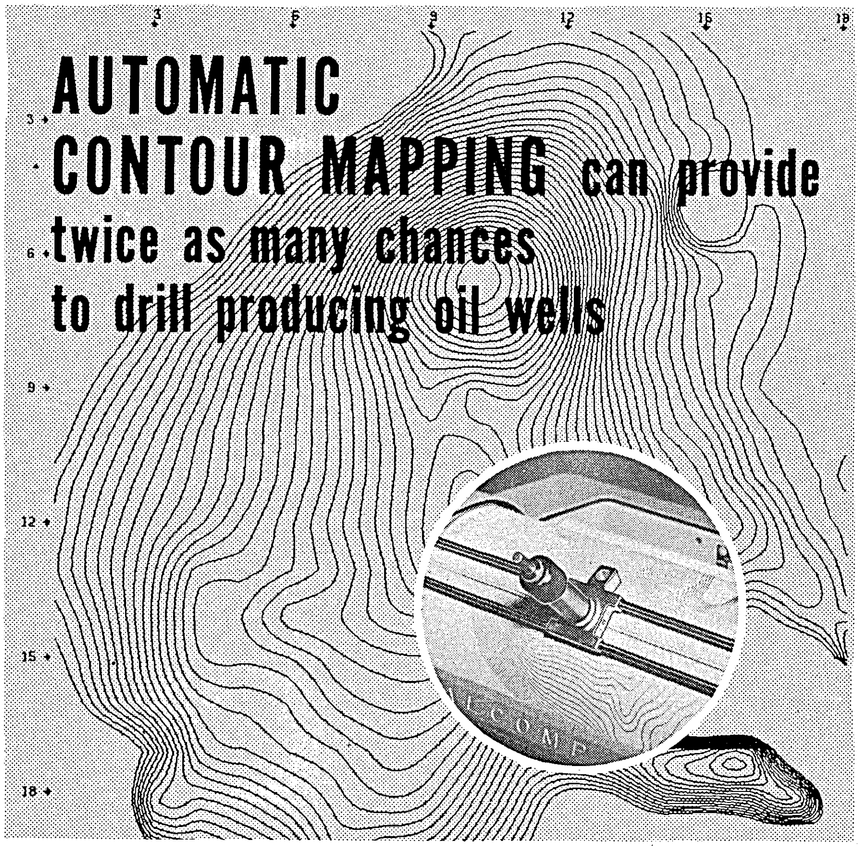
■ Warren C. Hume, formerly president of IBM's DP Div., has been elected vp-group executive of the division. He is succeeded by Frank T. Cary, who had been a division vp.

■ The election of Samuel Levine to vp of Systems and Product Planning has been announced by Teleregister Corp., Stamford, Conn. Assistant vp of Advanced Systems Development since 1962, he joined the firm in 1955.

■ Planning Research Corp., Los Angeles, has expanded its corporate structure by creating four new offices and elevating Stuart A. Krieger to executive vp, and Dr. Philip Neff, Dr. Alexander Wylly and Stanley L. Gendler to vp-general managers.

■ Don D. Bushnell of System Development Corp., Santa Monica, Calif., has been elected 1964-65 president of the Assn. for Educational Data Systems. He succeeds Dr. John G. Caffrey.

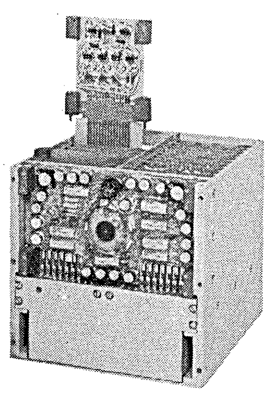
■ Research and development programs for Univac Div. of Sperry Rand Corp., will be directed by H. Burke Horton. With the firm since 1959, Horton has been general manager of Philadelphia operations since 1963.



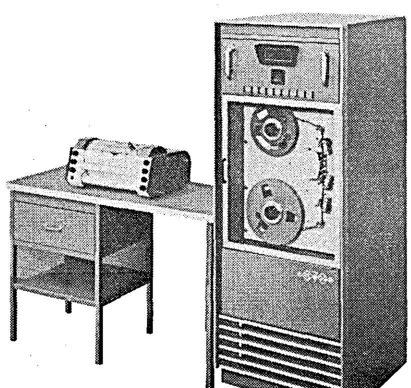

## Output of Scientific and Professional Manpower Increased by 100%

● Contour mapping by computers and CalComp plotters doubles the number of studies that can be generated by an oil company's scientific and professional manpower—thereby increasing by 100% the chances of drilling producing wells.

Other examples: CalComp plotters are used to plot and annotate rocket firing data in 2½ minutes, instead of 5 hours manually; plot missile test firing telemetry data in 3 hours instead of 2 weeks.



GOVERNMENT — Microminiature techniques were used in this pulse code modulated telemetry system for NIMBUS and TIROS weather satellites.



Model 670

INDUSTRY & BUSINESS — Latest CalComp magnetic tape plotting system speeds data reduction, mapping and automatic drafting.

CALCOMP

Specialists in Digital Technology

CALIFORNIA COMPUTER PRODUCTS, INC.

305 Muller Avenue
Anaheim, California
Write Marketing

CIRCLE 41 ON READER CARD

# MAKING THE MOST OF YOUR POTENTIAL

Because we work exclusively with computer and O/R people and keep in constant touch with thousands of computer installations, we have a vast and intimate knowledge of computer and O/R position openings throughout the country.

***Following are just a few of the opportunities available to you now through CPC:***

**COMMERCIAL**

- **EDP Consultants**
  Manufacturing $18,000
- **Lead Systems Analysts**
  Banking $13,000
  Manufacturing 20,000
- **Senior Systems Analysts**
  Banking $11,000
  Manufacturing 11,000
- **Junior Systems Analysts**
  Manufacturing $ 9,000
- **Senior Programmers**
  Manufacturing $11,000
- **Junior Programmers**
  Banking $ 8,000
  Manufacturing 8,700

**SCIENTIFIC**

- **Applied Programmers**
  Industrial Controls $ 9,500
- **Senior Programmers**
  Aerospace $12,000
- **Junior Programmers**
  Manufacturing $ 8,500

**COMPUTER MANUFACTURERS**

- **District Sales Managers**
  Scientific $20,000+
- **Senior Account Reps**
  Scientific $20,000+
- **Territory Sales Reps**
  Commercial $15,000+
  Scientific 15,000+
- **Service Center Sales**
  Scientific $12,000+
- **Systems Representatives**
  Commercial $15,000
  Scientific 15,000

**OPERATIONS RESEARCH**

- **Lead O/R Analysts**
  Manufacturing $18,000
- **Senior O/R Analysts**
  Manufacturing $20,000

*CPC gives you the benefits of:* Client-Paid Fees ■ Professional Handling ■ Confidential Relationships

CPC—nationally recognized for its field research of computer personnel characteristics.

Put yourself in a position to be considered for these and other opportunities.

T.M. WRITE, VISIT OR PHONE US *NOW*

CPC

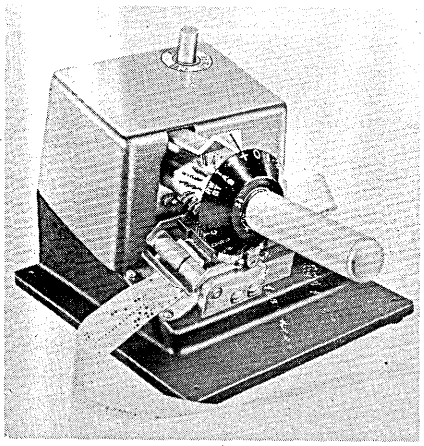
**COMPUTER PERSONNEL CONSULTANTS, INC.**
135 South LaSalle Street—Suite 1460
Chicago, Illinois 60603—Phone 312—236-0900

CIRCLE 78 ON READER CARD

# NEW PRODUCTS

### manual tape punch

For the N/C machine-tool operator, this hand puncher reportedly can produce over 200 x-y locations per hour. Tape feed accuracy is to .008-.010 inch per four inches of tape.



Punching spindles for different tools are available. PRODUCTION ENGINEERING CORP., Minneapolis, Minn. For information:
CIRCLE 200 ON READER CARD

### sheet microfilm reader

The Mark IV accepts microfiche to 5 x 8 inches, has 18x magnification, and one knob page position control. MICROCARD CORP., West Salem, Wisc. For information:
CIRCLE 201 ON READER CARD

### printer-plotter

Off-line system consists of mag tape drive and 1,000-lpm printer with graphic capability. Paper speed is 75 ips, plot resolution is 30 points per inch, and printout speed is independent of lateral plot density. POTTER INSTRUMENT CO. INC., Plainview, N.Y. For information:
CIRCLE 202 ON READER CARD

### wff'n proof

The new series of games designed to teach mathematical logic consists of 21 games and 222-page programmed text. For 12-year-olds on up. Price: $6. SCIENTIFIC EDUCATIONAL PRODUCTS CORP., 30 E. 42nd St., New York, N.Y. For information:
CIRCLE 203 ON READER CARD

### gp computer

Designed for rack mounting, the PDS 1068 has a magnetostrictive delay line memory in increments of 1K (4 digits + sign) words. Average add time is 9.2 msec, and basic command repertoire is over 40. Prices begin at less than $15K. PACIFIC DATA SYSTEMS INC., Santa Ana, Calif. For information:
CIRCLE 204 ON READER CARD

### line printer

The D401 has 120 columns, prints alphanumerically at 300 lpm. Used with a Reverse Channel Data-Phone, its double-bucket buffer provides simultaneous printing and data transmission. DIGITRONICS CORP., Albertson, N.Y. For information:
CIRCLE 205 ON READER CARD

### control computer

The Prodac 50 is small-scale hardware for on-line applications. The 4-16K (14 bit) core has a cycle time of 4.5 usec, and add time is 18 usec. Commands number 25, and there are 64 channels and external interrupts. WESTINGHOUSE ELECTRIC CORP., COMPUTER SYSTEMS DIV., Pittsburgh, Pa. For information:
CIRCLE 206 ON READER CARD

### control computer

The M/97400 is for industrial data logging, process supervision and control. It has 4K words of core and optional drums to 64K, more than 100 built-in instructions, indirect addressing and auto-indexing. FOXBORO CO., Foxboro, Mass. For information:
CIRCLE 207 ON READER CARD

### duplicate paper tape

The tape regenerator operates at 50 cps, features semi-automatic tape loading on the reader, reverse feed on both reader and punch, and tape tension switches. Accommodates 11/16, 7/8, and 1-inch tape with 5, 6, 7, or 8-level code. ROYAL

# BREAK THE SOFTWARE PERFORMANCE BARRIER

When the chips are down and the only thing that counts is performance, call SPC, the one company that always delivers, the one company with total software competence . . . over 100 computer specialists in systems, in real time, in software, in scientific applications.

That's why the tough problems go to SPC . . . the big ones, the dirty ones, the ones never tried before. Performance-wise, SPC routinely cracks computer and engineering software systems assignments that would break the backs of giants.

**CASE IN POINT: Douglas Aircraft Company, Missile and Space System Division, where the S-IVB stage of the SATURN vehicle is under development. In support of this massive effort, directed by NASA Marshall Space Flight Center, one of the world's most advanced and complex automatic checkout and ground data acquisition systems is now taking shape.**

**The systems use three CDC-924-A computers; continuously monitor over 800 analog, digital, and discrete data points; accept commands through an engineering-oriented automatic checkout and test compiler; initiate stimuli and provide automatic conditional test alternatives according to responses; permit automatic control of static test firing; control emergency shutdown and restart; and provide both a continuous running picture and quick-look analysis of test status.**

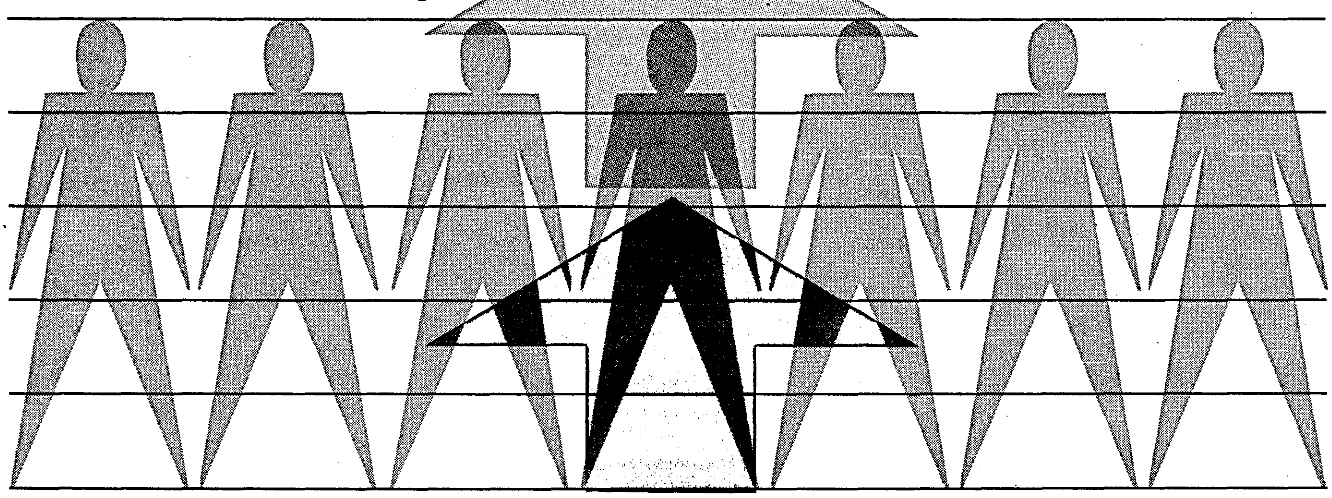
**SPC systems programming specialists, together with Douglas and Mesa Scientific Corporation systems engineers, are providing the advanced analysis program system design, and unprecedented systems programming performance needed to get the big bird off the ground.**

Exceptionally sticky problem? Sure. But smashing performance records is as regular as clockwork at SPC. Total Competence where it really counts — in systems, in real time, in software, in scientific applications — substantially reduces the conventional norm for time and cost.

In real time engineering control systems, computer software systems, data acquisition systems, multiple computer systems, simulators, process control systems, information systems, management systems, command and control systems — wherever the systems and programming problems are toughest — for complete project responsibility or expert on-site assistance — SPC outperforms all others.

For your computer system problem, call SPC. SPC wants your business and a chance to demonstrate what expert knowledge and superior performance **really** mean.

An **Equal Opportunity Employer,** SPC wants more expert, hard-charging people who live to **outperform.** If that's you, if this is your kind of company and you'd like to share in our growth, write personally to: J. N. Graham, Vice President and General Manager.



# SYSTEMS PROGRAMMING CORPORATION

1833 East 17th Street, Santa Ana, California • Phone: (714) 547-0069 • Offices in Inglewood, Pasadena, Washington, D.C., Boston • An affiliate of Mesa Scientific Corporation

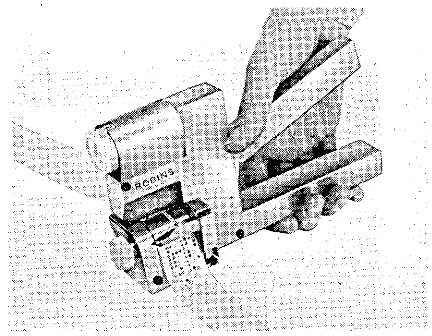
CIRCLE 43 ON READER CARD

## NEW PRODUCTS

McBEE CORP., INDUSTRIAL PRODUCTS DIV. Hartford, Conn. For information:
CIRCLE 208 ON READER CARD

### hand-held punch

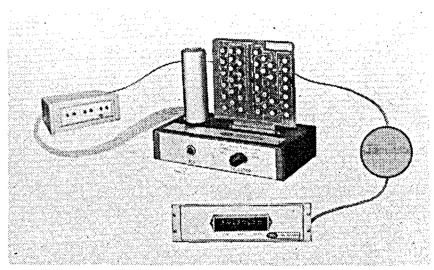
The Data Coder punches from 5-8 channel paper tapes with interchangeable code wheel, automatically moves




tape to next position. Weighs three pounds. ROBINS DATA DEVICES INC., Flushing, N.Y. For information:
CIRCLE 209 ON READER CARD

### remote digital display

The Remote Readout System uses a two-conductor transmission line to display BCD information from a digital clock in hours, minutes, and



seconds. Various configurations of electronics are available. INTERNATIONAL RESISTANCE CO., Philadelphia, Pa. For information:
CIRCLE 210 ON READER CARD

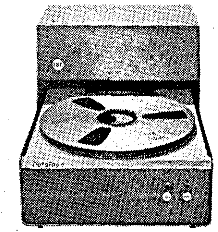
### crt display line

A character generator, vector generator, light pen, and coupler are designed for use with the firm's computers. The character generator has a 64-character set, 5 x 7 point array, four sizes. SCIENTIFIC DATA SYSTEMS, Santa Monica, Calif. For information:
CIRCLE 211 ON READER CARD

### disc pack vault

Fire and moisture-proof safe stores packs horizontally on slide-out shelves. Various capacities and shelf arrangements are available for CRAM units and aperture cards. DATA-AMERICAN EQUIPMENT CO., Chicago, Ill. For information:
CIRCLE 212 ON READER CARD





DATAMATION

# NEW LITERATURE

**ACCOUNT NUMBERING SYSTEMS:** 40-page manual explores adoption of numbering systems compatible with conventional, semi-electronic and fully electronic systems and equipment to assist banks in selecting a system which suits their needs. BURROUGHS CORP., Detroit, Mich. For copy:
CIRCLE 130 ON READER CARD

**PB440:** Four-page bulletin has updated information on hardware and performance specifications, plus software features. PACKARD BELL COMPUTER, 2700 So. Fairview St., Santa Ana, Calif. For copy:
CIRCLE 131 ON READER CARD

**A-D CONVERSION HANDBOOK:** 74-page publication covers conversion phases from concepts to calibration, includes circuit diagrams and tables. DIGITAL EQUIPMENT CORP., 146 Main St., Maynard, Mass. For copy:
CIRCLE 132 ON READER CARD

**HYBRID SYMBOLS:** 25" x 38" wall chart composed of diagramming symbols used in preparation of block diagrams for hybrid simulation programming is available. ELECTRONIC ASSOC. INC., West Long Branch, N.J. For copy:
CIRCLE 133 ON READER CARD

**CRT CATALOG:** 12-page booklet lists firm's line of cathode ray tubes, with physical and electrical characteristics and applications. GENERAL ATRONICS CORP., ELECTRONIC TUBE & INSTRUMENT DIV., 1200 E. Mermaid Lane, Philadelphia, Pa. For copy:
CIRCLE 134 ON READER CARD

**FLOOR TILE:** Computer-room flooring is the subject of this four-page bulletin. Perma-Kleen is a high-pressure plastic laminate with a non-porous surface. GENERAL ELECTRIC CO., Coshocton, Ohio. For copy:
CIRCLE 135 ON READER CARD

**FORMAT CONTROL:** System for converting raw data into computer compatible format, design features and applications of the system are outlined in this eight-page brochure. INDIANA GENERAL, Keasbey, N.J. For copy:
CIRCLE 136 ON READER CARD

**TAPE TRANSPORT:** A 300 cps incremental feed mode using an IBM tape format which reads and writes is described. This unit is available as an asynchronous read/write model (MTB-1501 SRW) and an asynchronous write-only model (MTB-1501 SW). POTTER INSTRUMENT CO., INC. 151 Sunnyside Blvd., Plainview, N.Y. For copy:
CIRCLE 137 ON READER CARD

**FELLOWSHIPS:** The National Science Foundation has released bulletins describing the following fellowship programs: Postdoctoral, Senior Postdoctoral, NATO Postdoctoral in Science,

# RCA STANDS FOR PROFESSIONAL & PERSONAL GROWTH IN EDP

If you have a minimum of two years EDP experience you can step up to more responsibility with RCA Electronic Data Processing. In these highly professional positions you'll have the opportunity to work on the latest software and hardware developments.

## OUTSTANDING CAREERS IN THESE SPECIALTIES

**SYSTEMS DEVELOPMENT**—Investigation and development of techniques, programs and concepts for the application of computers to scientific markets and to the development of management sciences.

**ADVANCED PROGRAMMING**—Investigation and development of total software systems—compilers, assemblers, operating systems, sorts, random access and real time systems.

**DESIGN AUTOMATION**—Work in computer-aided design as it applies to logic design, circuits, simulations, or packaging.

**SYSTEMS AND METHODS SPECIALISTS**—Development and support of EDP methods relating to: Systems specification coordination, software development liaison, methods technique developments, technical publication organization and complete field marketing hardware and software support.

**PRODUCT PLANNING**—Work with management, marketing, and engineering personnel, and be directly concerned with translating customer requirements into effective engineering and sales programs.

**INDUSTRY MARKET OPERATIONS**—Maintain maximum sales volume for RCA EDP equipment in your own field. Identify market potential, establish sales goals, plan and provide sales support resources, audit performance.

**EDP EDUCATION AND TRAINING**—Develop training programs and lecture in the areas of advanced software systems design and computer applications, particularly, real-time applications.

**Openings are in RCA Electronic Data Processing at Cherry Hill, New Jersey, and other locations throughout the United States.**

*To arrange a confidential interview with RCA Management, please submit your résumé to:*

**Mr. E. F. Grubic**
**RCA Electronic Data Processing**
**Building 202-1, Dept. AP-13**
**Cherry Hill**
**Camden 8, New Jersey**

*An Equal Opportunity Employer*

**The Most Trusted Name in Electronics**

CIRCLE 83 ON READER CARD

# DESIGN & DEVELOPMENT ENGINEERS

## *HIGH SPEED DIGITAL CIRCUITS, EXTREMELY HIGH PERFORMANCE CORE MEMORIES*

The success of CONTROL DATA CORPORATION has resulted in an expanded product line of high speed digital computers. Design and development activities are being expanded in line with the growth of products and corporate activities.

Several excellent positions are now being staffed within the Advance Design and Development Department. Openings exist in both circuit and memory groups for:

## *DEVELOPMENT SUPERVISORS, DESIGN ENGINEERS, DEVELOPMENT PROJECT ENGINEERS*

These are challenging assignments on state-of-the-art development projects for future generations of CONTROL DATA general purpose computers. A dynamic approach combined with engineering excellence creates outstanding opportunities for professional growth.

Successful performance of these assignments requires an exceptional degree of creativity and engineering competence. The ability to manage complex technical activities is a requisite.

If you are a graduate electrical engineer with experience related to the development of high speed digital circuits or high performance core memories, we would like to discuss these opportunities with you.

## *PLEASE SEND RESUME TO, OR CALL COLLECT:*

**R. D. Arent ■ Computer Division ■ 4201 No. Lexington Avenue ■ St. Paul, Minnesota 55112 ■ 631-0531, ext. 334, area code 612**

*An Equal Opportunity Employer*

**CONTROL DATA CORPORATION**

CIRCLE 79 ON READER CARD

## NEW LITERATURE . . .

Cooperative Graduate, Science Faculty, and Graduate. Closing dates for applications range from Sept. 8 to Dec. 11, 1964. NATIONAL ACADEMY OF SCIENCES, 2101 Constitution Ave., N.W., Wash., D.C. For copy:
CIRCLE 138 ON READER CARD

**90 - PLOTTER SOFTWARE:** Bulletin ORNL-3581 describes Intrigue, an IBM 7090 subroutine package for making linear, logarithmic and semi-logarithmic graphs using the Calcomp plotter. Dated March '64, price is $0.50. U.S. DEPT. OF COMMERCE, OFFICE OF TECHNICAL SERVICES, Washington, D.C., 20230.

**GLASS MEMORIES:** 12-page form places glass delay lines in perspective with other memories, describes characteristics, and catalogs the firm's line of products. CORNING ELECTRONICS DIV., CORNING GLASS WORKS, Bradford, Pa. For copy:
CIRCLE 139 ON READER CARD

**USED HARDWARE:** Bulletin describes used dp equipment available from broker. INFORMATION PROCESSING SYSTEMS INC., 200 W. 57th St., New York, N.Y., 10019. For copy:
CIRCLE 140 ON READER CARD

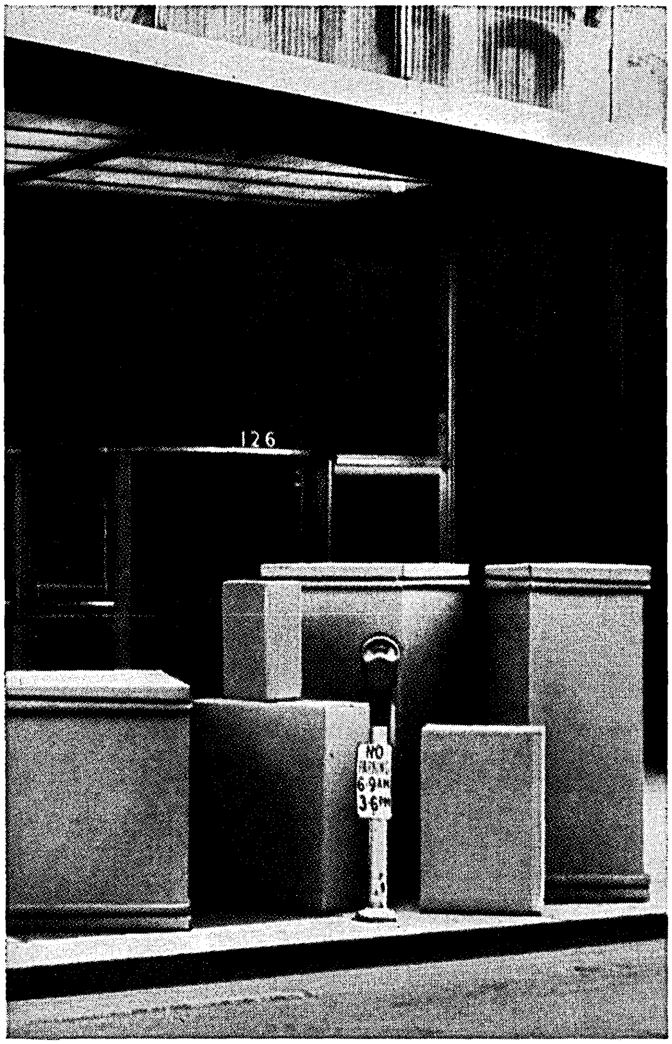
**H-200 SOFTWARE:** Tabsim 200 bulletin discusses the hardware's capabilities as a simulator for tab equipment operations. Another bulletin describes I/O routines available with the 200. HONEYWELL EDP, 60 Walnut St., Wellesley Hills, Mass. For copy:
CIRCLE 141 ON READER CARD

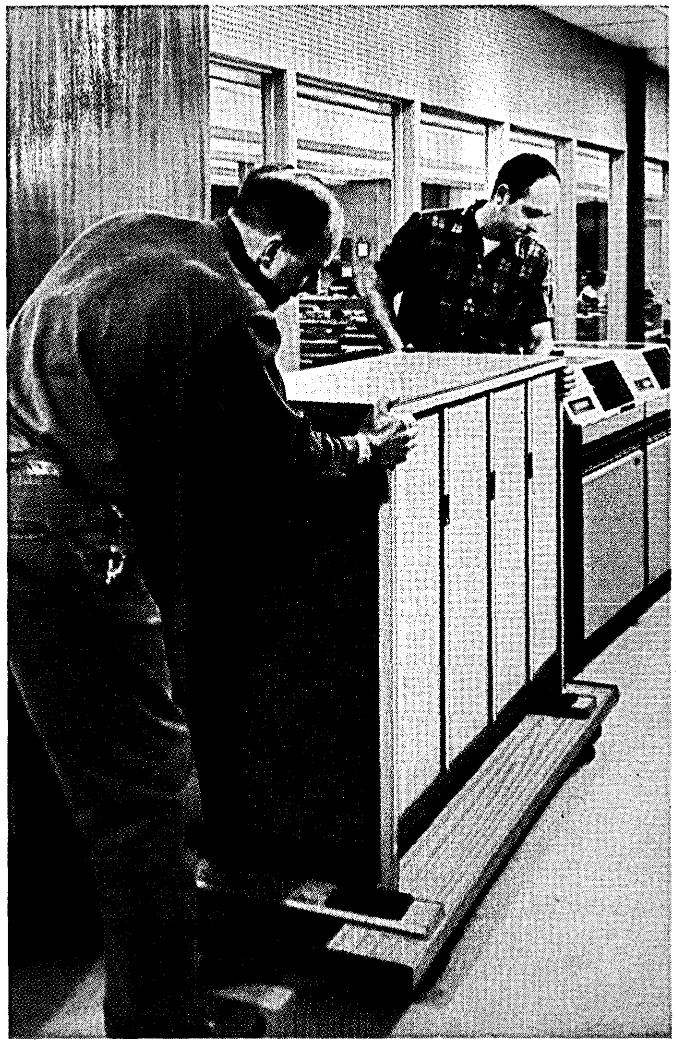
**X-Y RECORDER:** Describes 8½ x 11-inch HR-96 plotter with 0.25% accuracy and 1 mv/inch to 10 v/inch sensitivity. HOUSTON INSTRUMENT CORP., 4950 Terminal Ave., Bellaire, Texas. For copy:
CIRCLE 142 ON READER CARD

**TEXTBOOKS:** Six-page bulletin describes 10 primers and advanced texts on hardware design, languages, and programming. Software includes ALGOL 60 and FORTRAN primers. ADDISON-WESLEY PUBLISHING CO. INC., Reading, Mass. For copy:
CIRCLE 143 ON READER CARD

**CORE MEMORY:** Bulletin describes and illustrates the RZ high-speed, core memory. It also lists specifications and standard options. AMPEX CORP., 401 Broadway, Redwood City, Calif. For copy:
CIRCLE 144 ON READER CARD

# We go farther than the sidewalk:



## *Datacase is sold, installed and serviced locally!*

■ If the service you've been getting from your DP auxiliary supplier stops at your door, you'll be delighted to do business with your local Steelcase dealer. He's the one who handles Datacase—the color-coordinated auxiliary units with the "computer look" made by Steelcase. He has the experience and knowledge to help you plan or amplify your DP department, to create offices and machine rooms that are as efficient as they are attractive. And, as we said, he not only sells, but delivers, installs and services locally. Call him as your machine room needs arise. He's in the Yellow Pages. Steelcase Inc., Grand Rapids, Michigan, and La Mirada, California; Canadian Steelcase Co., Ltd., Don Mills, Ontario.

**DATACASE® / BY STEELCASE**

CIRCLE 46 ON READER CARD

*Continued from page 21 . . .*

column that DOD had been ''stuck'' with $162.5 million worth of outmoded computers in its recent mammoth purchase of equipment formerly on lease (see March Datamation, p. 21).

Under the provocative heading, ''Windfall for IBM,'' Anderson noted darkly that the whopping purchase deal, which principally involved that manufacturer's hardware, had been concluded just before IBM unveiled its System/360, which immediately obsoleted all predecessors. Participating in this decision, according to Anderson, was Edmund T. Pratt Jr., an assistant secretary of Defense for financial management — formerly with IBM's World Trade Corp. The columnist duly noted that Pratt had severed all his connections with IBM and that the final decision to purchase had been made by others, but concluded his article anyway with a quote from the company's recent annual report, describing its recent upsurge in sales as due primarily to government purchasing of equipment formerly leased.

The columnist's innuendos were ''unfortunate and unfounded,'' said Paul H. Riley, deputy assistant secretary of Defense for supply and services, who reiterated that final decisions on purchase plans had been made in his bailiwick, not Pratt's. Anderson really didn't understand the problem, Riley indicated, and was over his head when he tried to develop something sensational out of DOD's purchase actions. It's no secret, though, that DOD's choice of equipment caused mass unhappiness in a large section of the non-IBM Washington computing fraternity, who saw many potential sales effectively kiboshed by the action. Could be that one of the disaffected reached Anderson with the tidbit about Pratt's former relationship.

## THE BROOKS BILL: ONE YEAR LATER

The Clewlow committee, doing the spade work on the first comprehensive study of government dp procurement and management practices (May, p. 63), was in almost continual session as it approached the June 30 deadline. That date may be extended because the blue-ribbon advisory committee which will review Clewlow's work has not yet had its full say on the contents of the report.

But the Senate Government Operations committee still expects the report to come its way on time. It's in this committee's archives that the Brooks Bill, H.R. 5171, has long resided — July 18 is the first anniversary of its passage by the House — awaiting completion of the Clewlow study.

## COMPUTERS-IN-GOVERNMENT HEADCOUNT AVAILABLE

The last smidgeon of input for the BuBudget's fourth annual survey of computers in the federal government — bigger & better edition — has been sent to the printers, and by mid-July copies should be available. For yours, write to either the House Post Office and Civil Service committee, or to the Government Printing Office, Washington, D.C.

# Career Opportunities at Boeing

The Boeing Company, world leader in jet transportation, has immediate openings in the Airplane Division for graduates in engineering, mathematics and business and production management, with experience in the fields of Manufacturing and Business Systems, Engineering Systems, Computer Methods and Standards, Analog Computation and Flight Simulation, and Systems and Operations Research.

These positions provide professional challenge and scope, with unique opportunities to move ahead in your field.

**MANUFACTURING AND BUSINESS SYSTEMS** Graduates in industrial engineering, production management or engineers with an interest in computer technology. Experience preferred in systems analysis and digital computer programming for commercial applications. Assignments involve programming in such areas as production control, manpower forecasting, finance, facilities, quality control, materiel inventory control and management information systems.

**ENGINEERING SYSTEMS** Graduates in engineering, physics or mathematics including some training in numerical analysis with large scale IBM 7000 series digital computer programming experience. Work involves programming engineering applications (structural analysis, digital simulation, fluid dynamics, propulsion systems analysis, etc.) with emphasis on the integrated system approach. Use of geometric mathematical models is also involved.

**COMPUTER METHODS AND STANDARDS** Engineering and business administration graduates with experience in large scale digital and peripheral hardware, systems and languages. Responsibilities include the development of computer software, establishing computer standards, and the evaluation and selection of digital computer equipment.

**ANALOG COMPUTATION AND FLIGHT SIMULATION** Graduates in engineering or mathematics with experience in aerodynamics, analog and digital computer applications in flight simulation. Assignments include developing and applying simulation techniques to the solution of complex aircraft problems, using such techniques as adaptive design, mathematical models and hybrid methods.

**SYSTEMS AND OPERATIONS RESEARCH** Advanced degree in mathematics, statistics, engineering or physical sciences with a knowledge of systems engineering or operations research. Knowledge of computer programming is desired. Systems Research positions involve studying and developing analytical models in support of technical management for evaluation of alternate airplane or system design concepts. Operations Research assignments include research in management sciences involving decision making and operational problems, and assisting in the formulation and solution of these problems.

Salaries are competitively commensurate with experience and educational background. Boeing pays moving and travel allowances to newly hired personnel. Boeing is an equal opportunity employer.

These Airplane Division assignments are near Seattle in the uncongested Pacific Northwest, noted for mild year-round climate, nationally famous recreational facilities, excellent schools and housing, and healthful outdoor Western living for the whole family.

**BOEING**
AIRPLANE DIVISION

MAIL COUPON TODAY

Mr. Ivan Phillips, Airplane Division, The Boeing Company, P.O. Box 707-AAW, Renton, Washington.

Name__________
Address__________
Degree(s) Held__________
School(s), Year Graduated__________
__________
Field of Interest__________ Yrs. Exp.______

CIRCLE 86 ON READER CARD

# SYSTEMS ANALYSTS

# PROGRAMMERS

... with Scientific or Business Computer experience in the application of data processing systems for the collection, programming and dissemination of information at a high input-output level. Orientation in large Communications Systems desirable. B.S., M.S., or Ph.D. in Electrical Engineering, Mathematics, or Physics essential.

**HIGHLY DESIRABLE ASSIGNMENTS IN**

- PENNSYLVANIA
- MASSACHUSETTS
- WASHINGTON, D.C.
- CALIFORNIA
- FLORIDA
- MARYLAND

## ■ PROGRAM SYSTEMS ANALYSTS

To develop requirements and prepare specifications for design evaluation tests, to examine operation of experimental and production models of the system. Design of system tests and special test operating procedures. Will participate in live system testing of various complex systems. Will analyze test data and prepare documents which spell out results and conclusions to be derived from system tests. These conclusions should cover adequacy of the design logic and implementation of equipments, computer programs, and control manning.

## ■ SENIOR PROGRAMMERS

Will be responsible for the overall planning and supervision of computer programs. Will assign, outline and coordinate work of programmers and write and debug complex programs involving mathematical equations. Requires experience in the operation and programming of large electronic data processing systems, such as the AN/FSQ-7 or 8, IBM 700 or 1400 series, RAMAC 1301, or Philco 2000 series.

## ■ COMPUTER PROGRAMMERS

To develop and/or analyze logic diagrams, translate detailed flow charts into coded machine instructions, test run programs and write descriptions of completed programs. Requires experience in the operation and programming of large electronic data processing systems, such as the AN/FSQ-7 or 8, IBM 700 or 1400 series, RAMAC 1301 or Philco 2000 series.

Direct Resumes in Confidence To Dept. 801
G. T. EVANS
Employment Manager

**PHILCO**
A SUBSIDIARY OF FORD MOTOR COMPANY

**TECHREP DIVISION**
P.O. Box 4730 · Philadelphia, Pa. 19134
An Equal Opportunity Employer

CIRCLE 87 ON READER CARD

# DATAMATION'S FEATURE INDEX

# JANUARY-JUNE, 1964

### JANUARY

**Year of the Tiger** pp. 24-37

A special ten-article section previews the coming year. Authors highlight forthcoming computing trends from the standpoint of the manufacturer and the user. . .and in the federal government, government research, scientific computing, service bureaus, consulting and standardization. One article, "Automation, Unemployment and Utopia," argues that unemployment is a prize of technological success to which we must adjust.

**Inventory Control Without Batching** p. 45

by Joe A. Phillips

Discusses the use of on-line disc files to process in random, "real time" fashion the inventory at a US Air Force Military Air Transport Service base.

**Computer X: A User's Critique** p. 55

by Gomer Wheatly

Critical of the govt's purchasing policy and a manufacturer, the author describes a small-scale computer that was chosen because of its low price. . .but proved in the long run an expensive mistake. The author suggests that programmers should be more out-spoken, take a stronger stand in computer evaluation.

**Computers and the Continent** p. 57

by W. K. De Bruijn

Complementing the European computer coverage of 1963, this article describes the hurdles which European computer manufacturers have to overcome. Among these: competition from U.S.; inability to expand markets beyond their own borders; and the difficulty of financing computer installations.

### FEBRUARY

**Microprogramming and Stored Logic** pp. 24-36

A special five-article section gives an introduction to stored logic and microprogramming; describes and discusses organization of the TRW-133, PB 440 and C-8401; and discusses the questions of flexibility and programming complexity of the machines.

**Using Decision Structure Tables** p. 42

by D. T. Schmidt and T. F. Kavanagh

In part one, covering principles and preparation, the authors include the when and where's of using decision structure tables and supply six principles (accompanied by case history example), for structuring a successful decision problem. The article includes a brief primer, "What are Decision Structure Tables?"

**Data Processing Management Association** p. 54

by Robert B. Forest

In a four-page interview with R. Calvin Elliott, executive director, and James Adams, Jr., director of education, DPMA is reviewed. A brief history, chapter objectives, educational functions, hierarchy, general membership information and future plans of DPMA are discussed.

### MARCH

**Real-Time Information Processing** pp. 24-40

Four articles highlight general principles and applications of real-time data processing. T. B. Steel describes some of the standard definitions of real-time and proposes some standards by which it can be identified. Norman J. Ream, in the first of a two-part article, describes eight different levels of complexity in real-time management information systems and some of the principles by which management can decide which is most appropriate for his firm. Walter Bauer and Sheldon Simmons describe a real-time data handling system at the Pacific Missile Range, and Hans Van Gelder describes an on-line stock quotation system.

**The MITRE Corporation: An Interview** p. 42

by Robert B. Forest

Interview with C. W. Halligan, head of a nonprofit research firm sponsored primarily by the Air Force, focuses on the organization's goals, organization and work. Covered are the more general questions of the role of a nonprofit firm, and the advantages and disadvantages it offers compared to profit-oriented enterprise . . . and the question of how a democracy makes technical defense decisions.

**Using Decision Structure Tables, Part 2** p. 48

by D. T. Schmidt and T. F. Kavanagh

Discusses the benefits of decision structure tables as applied to manufacturing, describes PRONTO, a program for the automatic conversion of conventional manual manufacturing planning into instructions required by numerically programmed machine tools.

### APRIL

**Computers and False Economics** p. 26

by Mark Halpern

The author suggests that the early respect for computer time is out of date in the light of the twin facts of diminishing computing costs and rising personnel salaries. He offers three dimensions for which critical values exist to indicate whether the optimization process in a compiler is worth its price. And he stresses the necessity for gathering some statistics to determine a) the number of compilations required, on the average, before a satisfactory object program is produced; b) price paid at each compilation for object program optimization; c) the number of times object programs are executed on the average; and d) the savings through optimization realized at each execution.

**Burroughs D 825** p. 30

by James P. Anderson

A description of the design rationale, system structure, system modules, communications and system controls, interrupt features and software of a computer system designed for military command and control.

**Automated Payload Checkout** p. 35

by E. K. Montoya

Describes the use of a small-scale computer to speed up and automate the checkout of a satellite payload.

**On-Line Management Information System, Part 2** p. 39

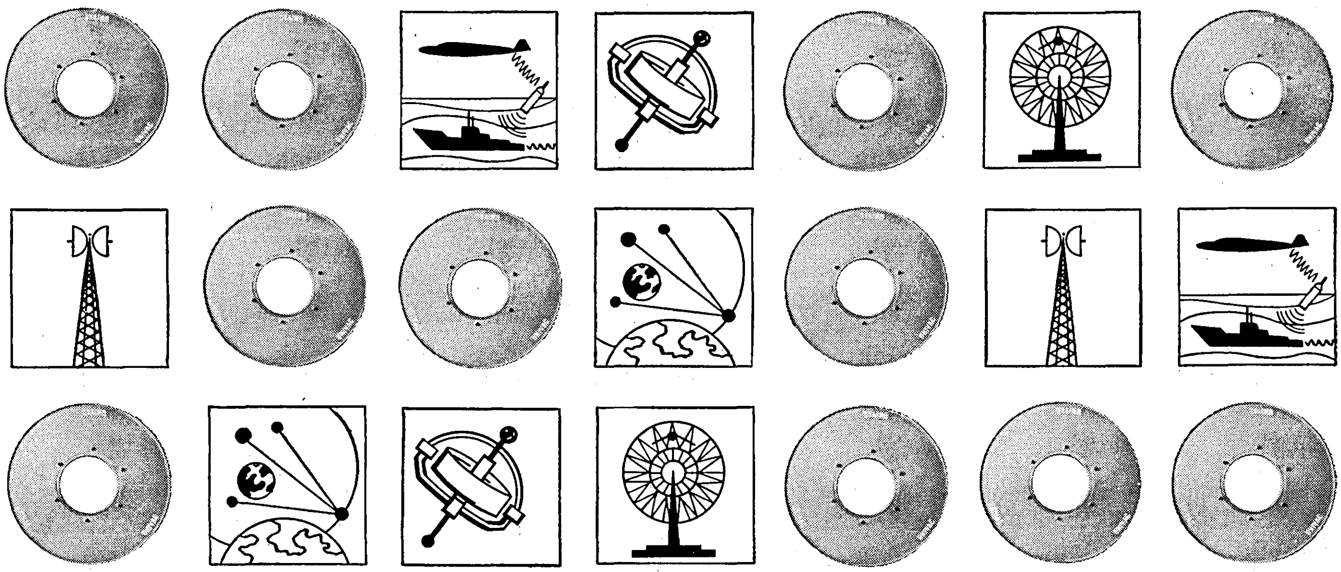
by Norman J. Ream

Second part of this article describes the stages of installation: systems design, hardware selection, programming design and execution, testing and integration, and system cutover. Under general implementation problems, the author discusses project organization, auditing, information protection, training, and the role of management.

**IBM's New System/360** p. 51

Description of the main modules and general features of the system, which offers six upward and downward compatible main frames and 44 I/O devices.

(Continued next month: May & June)



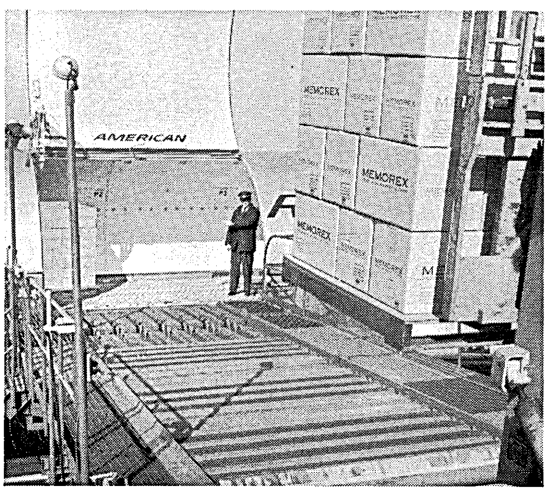
# 4 REASONS WHY MEMOREX LEADS IN COMPUTER TAPE

## (And what it means to you!)

**Premium Quality** Ask the most demanding computer centers whose tape is used for their ultra-critical applications. You'll find a multi-million dollar vote of confidence placed in the integrity of Memorex products — the result of uncompromised and unvarying product superiority. With Memorex precision magnetic tape you always benefit from measurably longer tape life, freedom from rejects, greater reliability.

(Photo courtesy of Stanford University Computer Center)


**Extra Technical Support** Authoritative technical backing is the hallmark of Memorex marketing. Manufacturing engineers and tape specialists with experience born of years solving digital recording problems offer help for the asking. Other aids to users include the Memorex Monograph series of informative literature; the Memorex tape slide rule; and technical liaison, of course. Next time you face recording problems, call Memorex!




**Added Service** Close attention to a customer's needs is the rule at Memorex. But not all are technical. It is important to have the right product at the right time and in the right condition. To assure timely delivery and extra care in transportation, Memorex routinely ships via air carrier at no extra cost. What is more, specially designed containers and material handling methods keep tape error-free and pre-test perfect throughout shipment.

**Continued Research** Memorex Computer Tape is scientifically designed (not the result of cut-and-try methods) to meet highest standards of consistency and reliability. Intensive research in oxides, binders, and process innovations assures continued quality advances. Some results are subtle (an increase in coating durability); some are obvious (the super-smooth surface). But all improvements pay dividends in better performance to Memorex tape users.

Memorex tape is premium tape. No need to pre-check it. You can place Memorex computer tape directly in service — reel after reel.

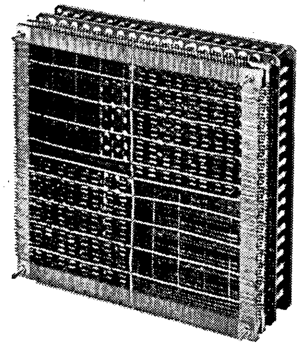
Memorex certification means what it says: Memorex computer tape is error-free. Extra care, extra steps and scrupulous attention to every detail make it that way. We know the importance to you of having a tape you can depend on.

Are you on our mailing list to receive the Memorex Monograph Series of informative technical literature? Write 1172 Shulman Road, Santa Clara, California

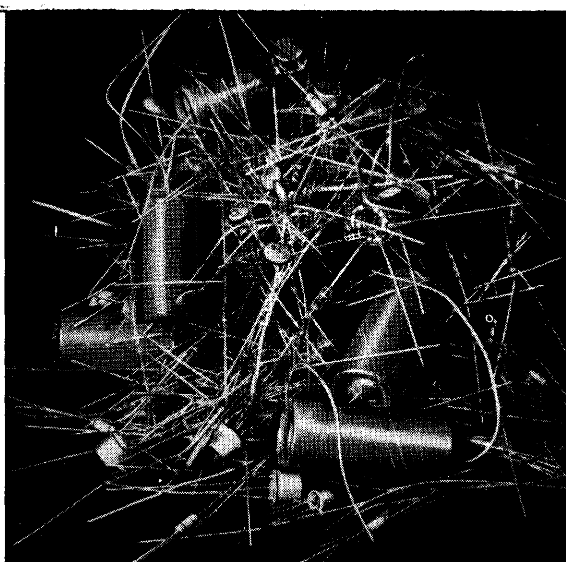
MEMOREX

PRECISION MAGNETIC TAPE

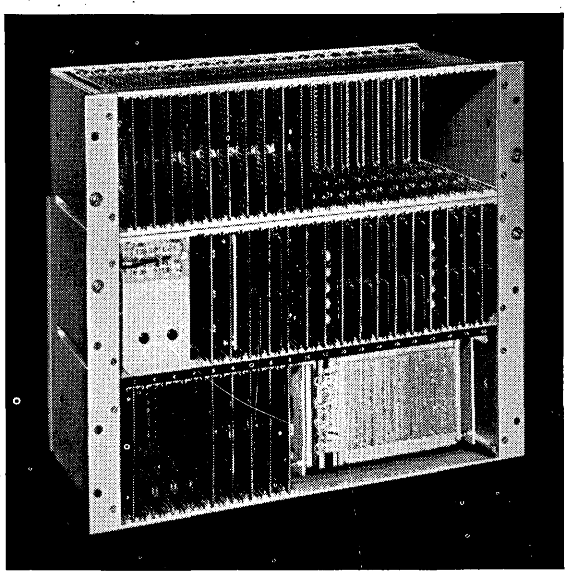
CIRCLE 2 ON READER CARD



# switch core matrix...



# eliminates components...



# injects added reliability to compact **5 and 6 microsecond** memory systems

Ferroxcube's new 5 and 6 microsecond Memory Systems feature this unique switch core matrix drive **and** selection technique. Consequently, semiconductor requirements are cut by 30%. Size is reduced. A typical unit measures only 19″ x 18″ x 8″ with a capacity of 8192 words — each 26 bits long. Reliability is increased — hermetically sealed, all silicon semiconductor circuit modules and L.T.C. cores provide the wide temperature range (0°C to +65°C) of operation required for extreme environments. Write for complete details on Ferroxcube's fast growing family of Memory Systems, or call 914-246-2811 for application assistance.

**FERROXCUBE**
CORPORATION OF AMERICA
SAUGERTIES, NEW YORK
T.M.

CIRCLE 3 ON READER CARD