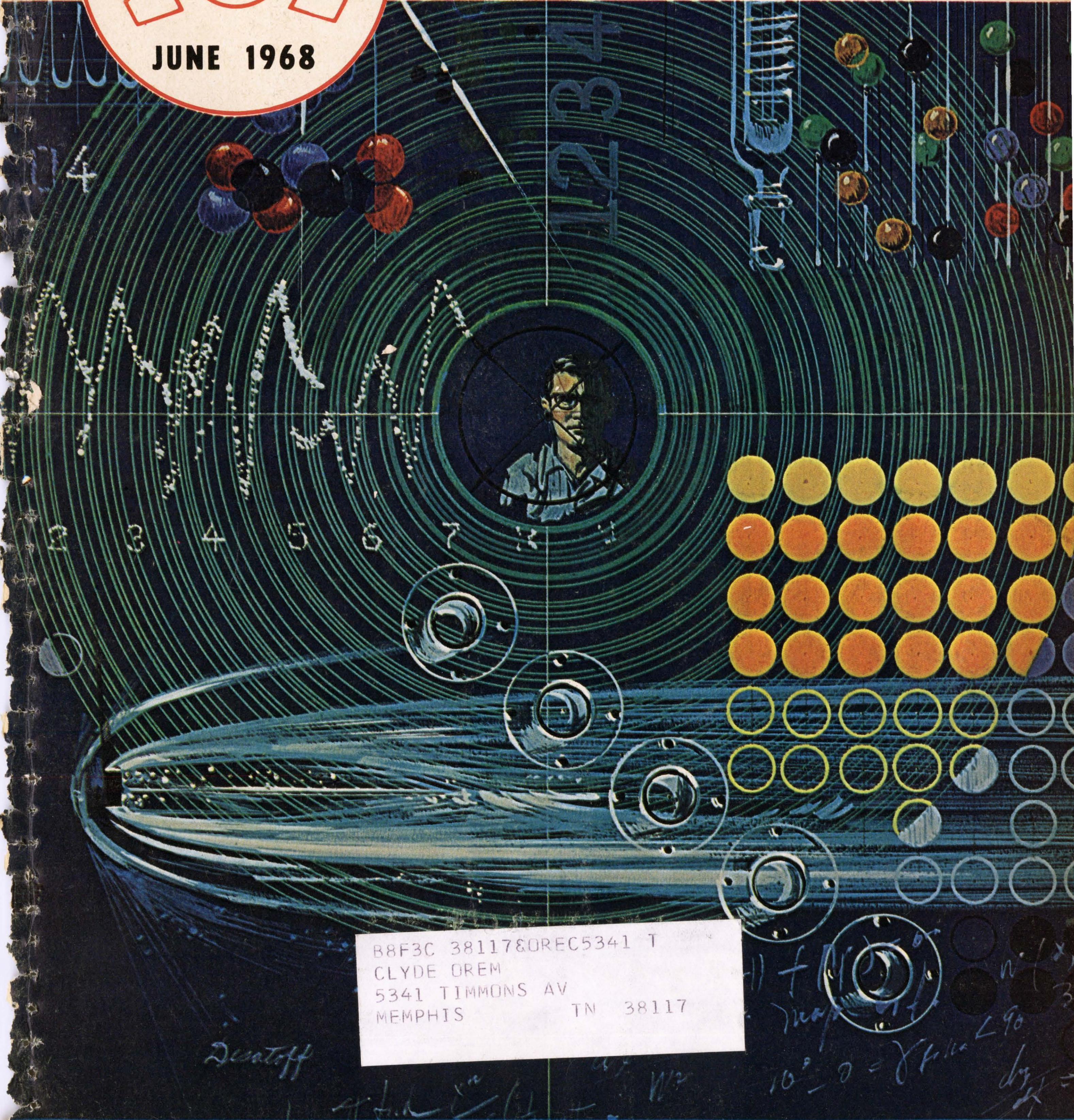
software age

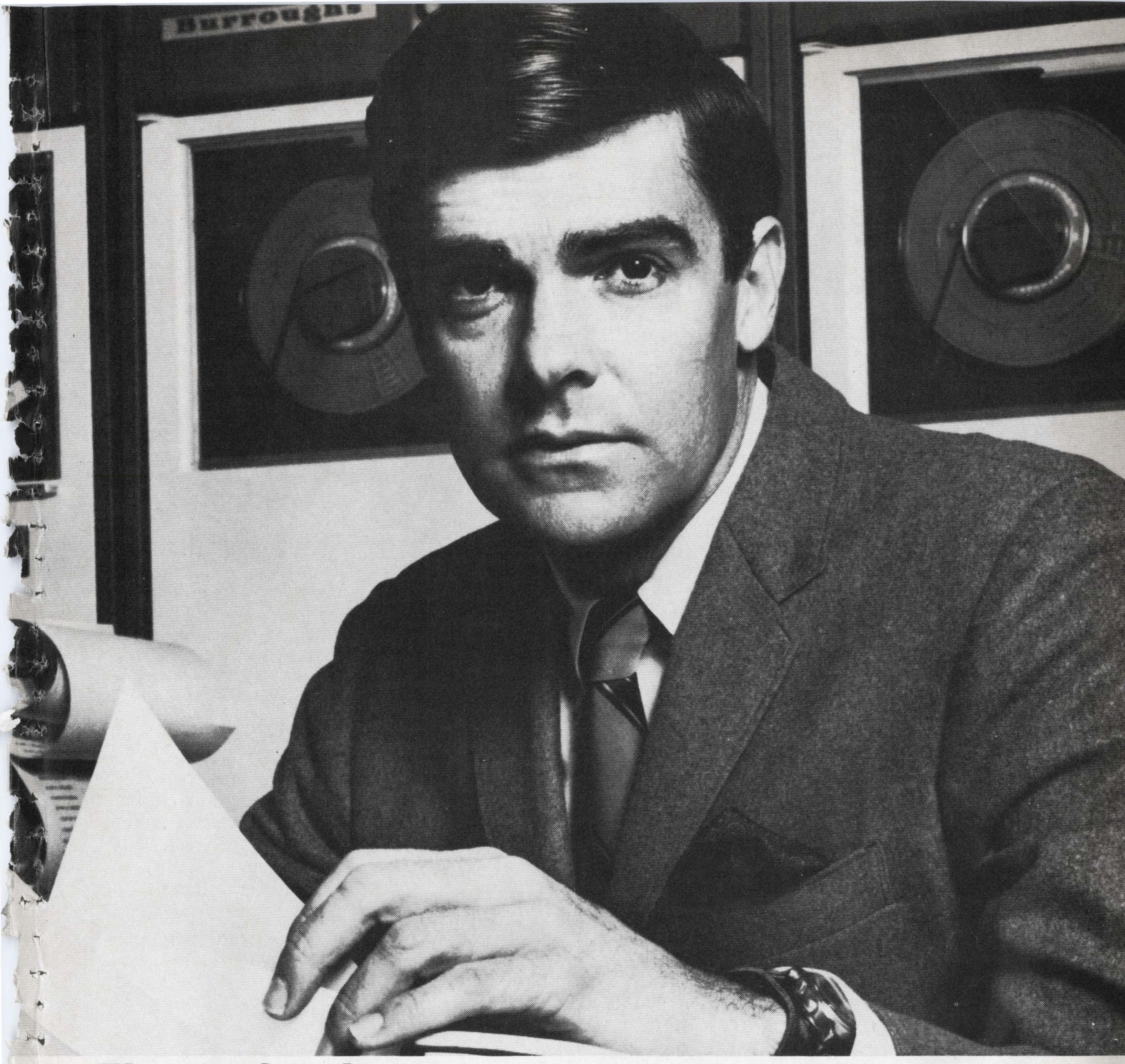
JUNE 1968

PSEUDO DEVICES IN OS/360
SEE PAGE 8



B8F3C 38117&OREC5341 T
CLYDE OREM
5341 TIMMONS AV
MEMPHIS TN 38117

THE MAGAZINE FOR PROGRAMMERS • MATHEMATICIANS • ANALYSTS • EDP MANAGERS



software age

JUNE, 1968 Vol. 2–No. 5

CONTENTS **PAGE**

Multi-Dimensional Models — Otto Kozma 6
Pseudo Devices in OS/360 — David N. Freeman 8
The Computer Meets the Shell — 20
Efficient Programming in FORTRAN — Charles Erwin Cohn 22
The Winners of Trouble-Tran Problem 1 — 32
Trouble-Tran Adventures for Fun and Profit — 33
Market Place—The Classified Ad Department of Software Age — 38
New Products — 40
Confidential Inquiry Form — 45
Index to Advertisers — 46

Circulation of this issue more than 91,000

Software Age is published monthly by Press–Tech, Inc. (312) 869–1244

1020 Church Street, Evanston, Illinois 60201. Subscrip- free to qualified readers. Others, $10/yr. Individual copies, $1. Foreign subscriptions, $10/yr.

Main Sales Office: Norman Jacobs (Adv. Mgr.) or Norman Brodsky, 1020 Church Street, *Evanston, Illinois* 60201—Telephone (312) 869–1244. Gerald Green, 60 E. 42nd, *New York, N. Y.*—Telephone (212) 697–5356. Richard Faust, 9800 S. Sepulveda Blvd., *Los Angeles, California* 90045—Telephone (213) 776–0100. David S. Wood, P. O. Box 189, *Canton, Massachusetts* 02021—Telephone (617) 542–1466. Richard D. Clemmer, 4900 Red Fox Drive, Annandale, Virginia 22003 *(Washington, D. C.)*—Telephone (202) 461–9792. Richard Faust, *Palo Alto, California*—Telephone (415) 327–8340.

CONTROLLED CIRCULATION POSTAGE PAID AT MADISON, WISCONSIN

MULTIDIMENSIONAL MODELS

Otto Kozma

Supervisor of Programming Group
Cleveland Crane & Engineering Division of McNeil Corporation
Wickliffe, Ohio

Multidimensional Models

■ The complexity of engineering and management systems stimulated the creation of the critical path method and PERT to solve problems including many components that influence the cost or completion date of a complex project.

The network usually represents the operations, the time required to complete them and the sequential relationship of these operations.

The value of these techniques are well known.

In evaluation or optimization of an old system and designing a new system, the problem is usually how to represent the large number of components that have a large number of parameters that characterize each and still have the proper sequential relationship.

Efforts to convert one characteristic into another for a component characterization by a single parameter, has the shortcoming that the physical meaning is lost in that process with most of its implications.

The real problem exists in Reimann's dimensional space. In many of these problems the three space coordinates X, Y, Z has often much less significance than money, time, quality, type of equipment, function and many other coordinates that are important in Reimann's space.

The graphical representation of most of the important coordinates is possible in a flow diagram of preferably three dimensional blocks. The three different visible planes of the block are to contain three different groups of coordinates, the front plane is preferable to contain the function of the "block" that is the component of the system. The two other planes may represent present and future characterizations of means that provide the shown function with the corresponding performance. See Fig. 1.

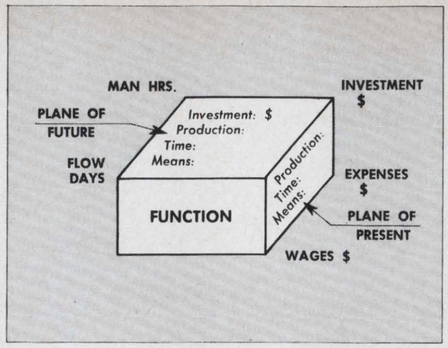


Fig. 1.

The drawing of three dimensional blocks with templets can be done without difficulty. Nearly all the patterns can be used that are available on a computer programmer flow diagram templet.

At each of the six exposed corners, the total along the flow path of certain coordinates can be shown like:

(Continued on page 44)

Fig. 2. Flow diagram of the process of golf ball development and marketing.

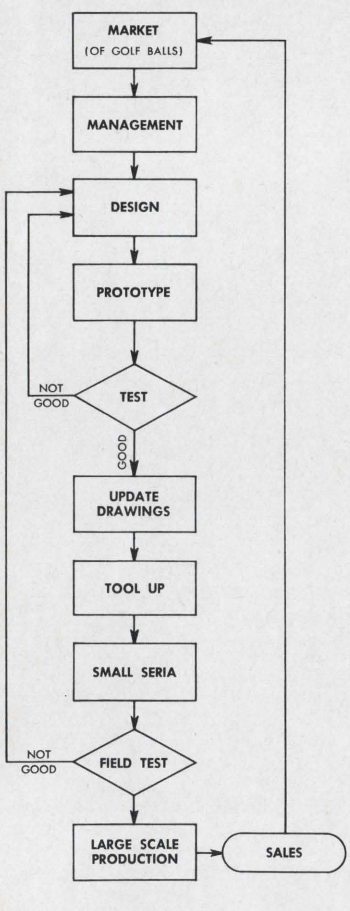
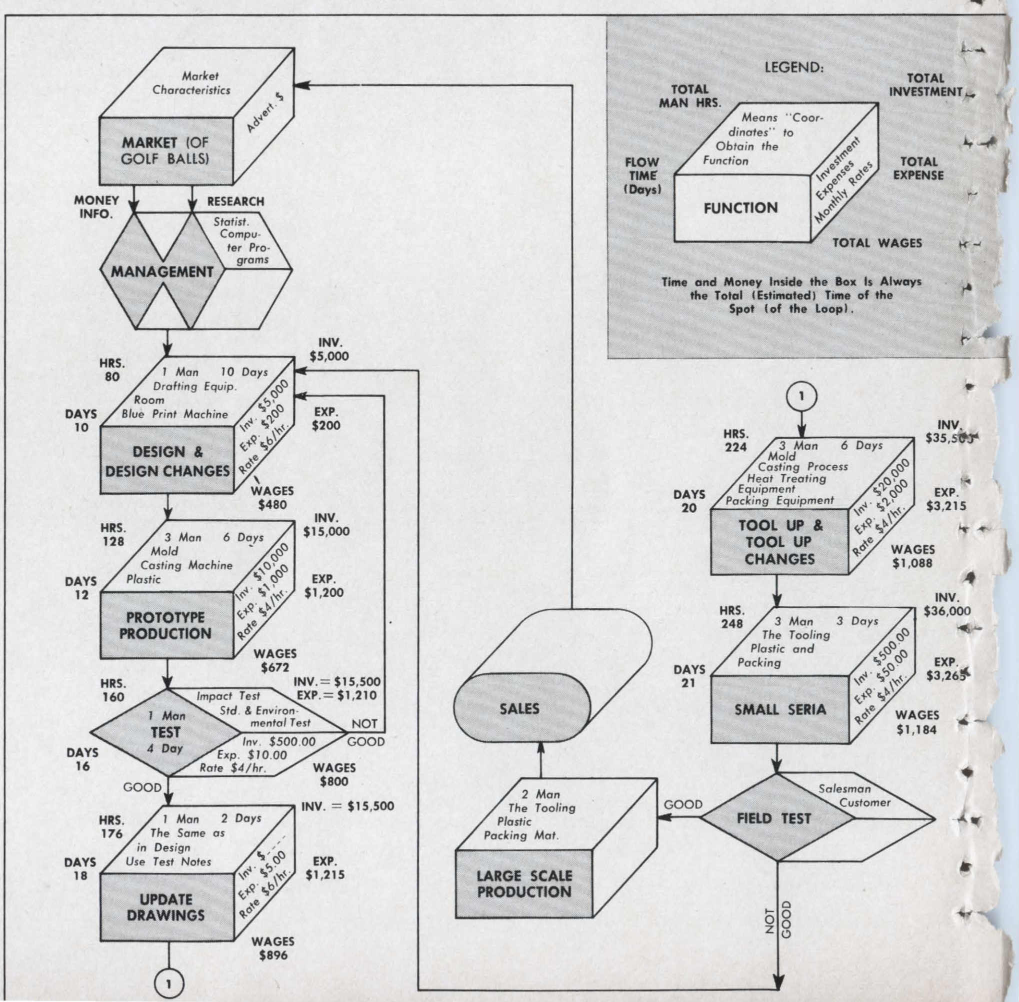


Fig. 3. Flow diagram of the business of golf ball development and marketing.



Pseudo Devices in OS/360

David N. Freeman
Triangle Universities Computation Center
Research Triangle Park, North Carolina

Introduction

When IBM announced Operating System/360 in 1964, many programmers—presumably including the OS/360 designers—felt that device independence could be achieved without significant performance loss. The most important elements of "loss" are as follows:

1. Core storage for control blocks and I/O-control logic;
2. Overhead in time to open and close data sets;
3. Aggregate overhead in time to furnish system-I/O services to compilers, sorts, utilities, user programs, and—most importantly—the control program itself.

In OS/360, the following control-program data sets have the greatest impact on performance:*

- SYSJOBQE = scratch storage for job-management subroutines;
- SVCLIB = subroutines for special I/O services (e.g. OPEN, CLOSE, device error recovery) and job-management services (e.g. allocate track areas on disk);
- PROCLIB = standard collections of job-control cards resembling macro definitions, both functionally and syntactically;
- LINKLAB = the principal library of executable programs.

These data sets were originally (1966) kept entirely on direct access storage devices (DASDs) such as drums and disks. However, DASDs were poorly matched with internal compute speeds of all but the slowest models of System/360. For example, the mean time to retrieve one subroutine from SVCLIB on IBM's fastest drum is 10ms., approximately 10,000 instruction-times on a Model 75. Since the "mean free path" of OS/360 between requests for SVCLIB—in an environment dominated by short, nonsetup jobs—is of the order of 3000–5000 instructions, the system was retrieving SVCLIB members for a significant fraction of clock time—perhaps 20–40%.

In addition to the control-program data sets, OS/360 performance depends critically on the following functions:

- SYSIN = the stream of card images feeding each of several processing partitions;
- SYSPRINT and SYSPUNCH = streams of corresponding output images;
- SYSUTx, SYSLIN, SYSLMOD = scratch storage for the compilers, linkage editor, sort programs, etc.;
- FORTLIB, PLILIB, ALGLIB, COBLIB = libraries of object-time subroutines for various compilers.

SYSLIN, SYSPRINT, and SYSPUNCH have evolved from ancient, honorable, unit-record concepts. Typical users find it convenient to submit card decks and receive printed listings and card decks in return, interactive computing to the contrary. OS/360 has retained these SYSIN/SYSOUT representations—80-byte input, 120-132-byte, output—as standards for job control cards (JCL), compiler input, object-program input, and the corresponding outputs.

Fundamental Performance Problems

The OS/360 designers badly underestimated the three elements of performance loss due to device independence:

1. Control blocks and IOCS logic—particularly for DASDs—required several thousand bytes to perform I/O for an average compilation or user program, rather than the anticipated figure of several hundred bytes. Not surprisingly, the simplest devices required the least IOCS core storage: card reader/punches and printers.

Surrendering core storage to IOCS has always been disagreeable to users with large-core problems, even for such worthy objectives as device independence. For them, loss of core has a special performance impact—or reprogramming impact—if they must reduce array sizes or are obliged to divide large programs into overlays.

* Reference [1] is a satisfactory overview of OS/360 and its terminology.

2. To open a typical DASD data set required over one second with early disk-resident versions of OS/360. Most jobs—then and now—open 10–20 such data sets. These data sets are generally closed as often as opened, incurring additional overhead.

Since 1966, significant improvements to OPEN and CLOSE times have been achieved by the following tactic: commonly-used SVCLIB subroutines—e.g. those opening reader, printer, and disk data sets—can be optionally made core-resident (or simply, *resident*). This tactic reduces OPEN and CLOSE times by factors of at least 3. In either case—resident or non-resident SVCLIB subroutines—reader and printer OPEN times are much smaller than disk OPEN times, reflecting the greater complexity and parametric variability of disk data sets. Characteristically, opening a printer data set requires less than one-quarter the time to open a disk data set.

However, inserting SVCLIB subroutines into the nucleus can severely reduce core storage available to user programs: alleviating loss #2 aggravates loss #1. Most managers of OS/360 systems with at least 256K core storage gladly trade core storage for improved OPEN/CLOSE times. A typical 256K system might include 10K bytes of SVCLIB subroutines in its nucleus; a 512K system might include 25K bytes; a megabyte system might include 80K bytes.

The TUCC system [2–5] has 512K bytes of fast core and 2 megabytes of large capacity core storage (LCS). We have included 13K bytes of SVCLIB subroutines in the nucleus and—by local modifications to the nucleus-initializing subroutine—80K bytes of less-frequently-used SVCLIB subroutines in LCS. LCS is relatively cheap: 0.6¢ per byte-month vs. 3.7¢ per byte-month for fast core. Thus, the TUCC system deliberately incurs additional instruction-execution overhead for SVCLIB subroutines, since the ratio of instruction times between LCS and fast core is 10:1 on the average. (LCS has an average access time of almost 8μs, fast core of approximately 0.5μs. However, the Model 75 pre-fetches instructions and data; this somewhat mitigates degradation due to executing SVCLIB subroutines from LCS, since their data is in fast core.)

At TUCC, the improvement in OPEN/CLOSE times attributable to 93K bytes of core-resident SVCLIB has been at least five-fold. This improvement is attributable to reducing average access time from 10ms. to 0, somewhat balanced by the slower execution rate from LCS. (Roughly 200 instructions are executed by an average SVCLIB subroutine; assuming an overhead of 8μs per instruction executed from LCS, the average overhead per usage of a subroutine in LCS is 1.6ms.)

The TUCC system requires less than 0.2 seconds to open a typical disk data set, contrasted with 1.0 seconds for an unmodified system. A printer data set can be opened in less than 0.08 seconds. TUCC has made resident all card-reader/punch, printer, and disk SVCLIB subroutines required by the control program, compilers, sorts, and the vast majority of user programs. (Since TUCC receives relatively few magnetic-tape jobs and disk jobs using exotic access methods, the corresponding SVCLIB subroutines have not been made resident. To open a magnetic-tape data set requires approximately 0.5 seconds on our system.) As we shall describe below, the principal system functions use reader/punch, printer, and disk devices—real and simulated—plus the communications devices furnishing streams of jobs to/from remote terminals.

On our system—and, we suspect, on most Model 75-class systems—magnetic tape is inappropriate for system functions, because of (a) slow READ/WRITE times, (b) prolonged REWIND times, (c) strictly serial operation, precluding multiple job streams to/from a single drive, (d) infeasibility of dynamic job-scheduling algorithms (fast turnaround for small jobs), and (e) non-sharability among multiprogrammed tasks, e.g. SPOOL task feeding processing tasks.

3. The overhead—in time—to furnish a stream of input card-images to a processing partition is a function of

(a) complexity of the deblocking logic,

(b) complexity of buffer management,

(c) complexity of initiating channel service to the input device and responding to interrupts therefrom, and

(d) average access time to the device.

Since card reader/punches and printers are inherently unblocked, and since their channel programs are trivially simple, the system overhead to GET one card image from a reader is less than 40% of that required to GET an identical card image from a disk data set. The fundamental problem in using card/printer devices on any powerful computer is item (d). The next topic explains how a number of OS/360 installations have reduced this item for various classes of I/O devices.

Pseudo-Devices

At least four distinct variants of OS/360 based on pseudo-devices have been successfully implemented during the last twelve months. We define a *pseudo-device* as a control block and associated I/O-supervisor logic which is represented to OS/360 as a device—typically a card reader, punch, or printer—but is actually implemented as a combination of core storage, other I/O devices, or even communications lines. This combination functionally imitates the corresponding device; most of OS/360—control program, compilers, the Fortran I/O library, etc.—is unaware of the substitution.

What is the rationale behind pseudo-devices? Raising system performance significantly above what is attainable with IBM's own variants of OS/360, and interposing a cheap interface with telecommunications equipment, pseudo-devices represent a universal, simple interface to the OS/360 control program and compilers, leaving system programmers complete freedom to simulate devices when and how they choose. Pseudo-devices represent capitulation to device dependencies of OS/360. Furnishing simple, simulated devices for system functions is becoming widely accepted as the *sine qua non* for (a) raising OS/360 performance to acceptable levels and (b) achieving simple user control of the job-collection, job-selection, and job-disposition functions.

If you're interested in working five years ahead of the times, learn about IBM's Federal Systems Division.

We're working at state-of-the-art levels—and we need programmers and analysts now.

Our Federal Systems Division is involved in a series of real-time projects for the Government that are about as advanced as anything in the field of computer technology today: trajectory analysis, orbit prediction, multi-programming, multiple-access concepts, acoustic instrumentation. And most of this work is being done using the latest computers and peripheral equipment available.

Our positions are located in Northern New Jersey and the Washington, D.C., area.

At IBM you'll find completely company-paid benefits that include life, medical and hospitalization insurance. IBM also offers you an exceptional educational assistance program.

Are you one of these people?

We need programmers and analysts with experience in one of two basic areas: information-handling systems and scientific-engineering programming.

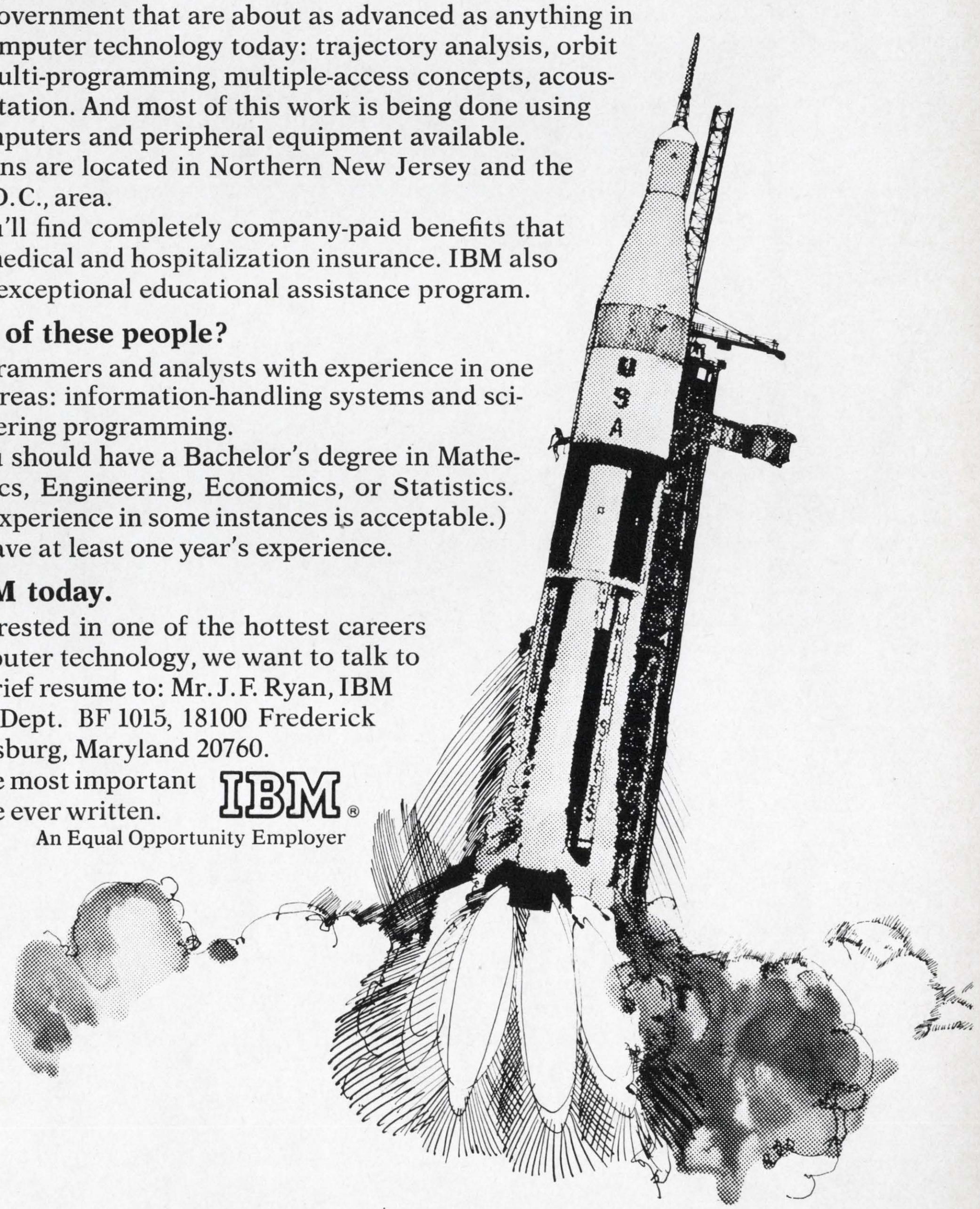
Ideally, you should have a Bachelor's degree in Mathematics, Physics, Engineering, Economics, or Statistics. (Equivalent experience in some instances is acceptable.) You should have at least one year's experience.

Contact IBM today.

If you're interested in one of the hottest careers going in computer technology, we want to talk to you. Send a brief resume to: Mr. J.F. Ryan, IBM Corporation, Dept. BF 1015, 18100 Frederick Pike, Gaithersburg, Maryland 20760.

It could be the most important resume you've ever written. IBM®

An Equal Opportunity Employer



Logically, a card reader is an infinite sequence of 80-byte images. When a program issues an Execute Channel Program (EXCP) macro-instruction to a card reader—real or simulated—one of the following events results: (a) the next card image is furnished and the requestor proceeds, or (b) the next card image is not yet available and the requestor WAITs. In case (b), the requestor yields control of the CPU to some other dispatchable task in the system. If no tasks are dispatchable, the system is idled until an I/O interruption permits the corresponding task to continue.

Logically, a printer is also an infinite sequence, with the following response to EXCP: (a) the next print line is accepted from the requestor, or (b) the printer is still busy with the preceding image and the I/O request is queued (until an I/O interruption signifies that the printer is freed). For pseudo-printers, case (b) is usually modified so that the requestor immediately yields control to another task when he "floods" the device, i.e. furnishes print lines too fast for the simulator to move them out of core.

A punch is logically identical to a printer, except for control commands such as Stacker Select (punch) and Skip/Space lines (printer). The pseudo-devices described below typically convert all control commands to first-character control. User A may use ASA first-character printer control, user B (of the same pseudo-printer) may use machine-command first-character control, user C may furnish no first-character control. By the time OS/360 issues EXCP, control specifications for all three users have been converted to a simple repertoire of machine commands (CCWs). The TUCC pseudo-printers reconstruct machine-command first-character control from CCWs as a *universal* SYSOUT representation. In some cases, this necessitates insertion of a blank line together with a control command, but such tampering with SYSOUT does not change either (a) the ultimate printed images or (b) the gross performance of the pseudo-printer.

By insuring unique first-character control for printers and punches, two streams from a single job bound for a single destination can be merged. This is particularly convenient if SYSPRINT and SYSPUNCH images are simultaneously received from two or more processing tasks while a telecommunications task is transmitting already-processed streams to appropriate destinations. Merged SYSPRINT/SYSPUNCH streams reduce access-mechanism movement if the streams are queued on disk; they reduce aggregate core requirements if queued in core.

Pseudo-disks are the final category of this paper, discussed in detail under "The TUCC Remote Job Entry System." An elementary pseudo-disk has been implemented in IBM's Type-I support for resident SYSJOBQE. In the "Introduction" SYSJOBQE was identified as a DASD data set with major impact on system performance. Until the pseudo-disk implementation, SYSJOBQE accesses cost the OS/360 control program significant I/O-watt time. The pseudo-disk support of SYSJOBQE operates as follows:

Only two channel programs are admissible for SYSJOBQE; the disk simulator converts track and record addresses into equivalent core-storage addresses. These addresses are entirely within an area of the nucleus (in LCS in the TUCC system) which is dedicated to holding SYSJOBQE data at all times. Thus, I/O operations to the SYSJOBQE data set are intercepted and simulated by data-move operations.

The Houston Automatic Spooling Priority System (HASP) [6]

Although HASP has not been widely documented in IBM publications or the professional literature, it is today (early 1968) the most widely-used variant of OS/360 on large systems (Model 65 and above). HASP drives its pseudo-reader, -punch, and -printer by means of minimal-seek disk operations issued by a partition (P0) operating independently of the batch partition (P1). P0 controls all real unit-record equipment. It runs each card reader at full speed so long as (a) there are cards in the hopper and (b) the input-queue disk space is not full. In case (b), P0 merely stops the card reader until at least one track is freed for more input. Control flows as shown in Figures 1–4:

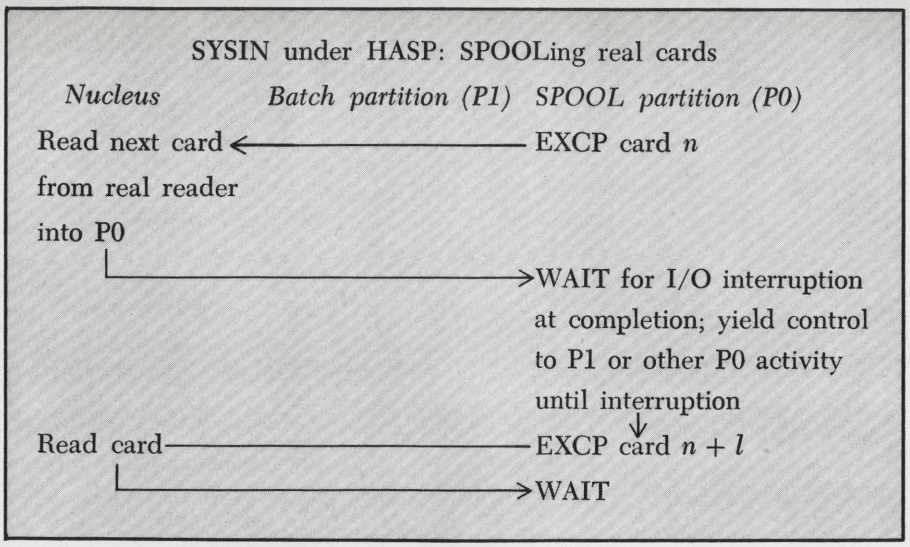


Figure 1

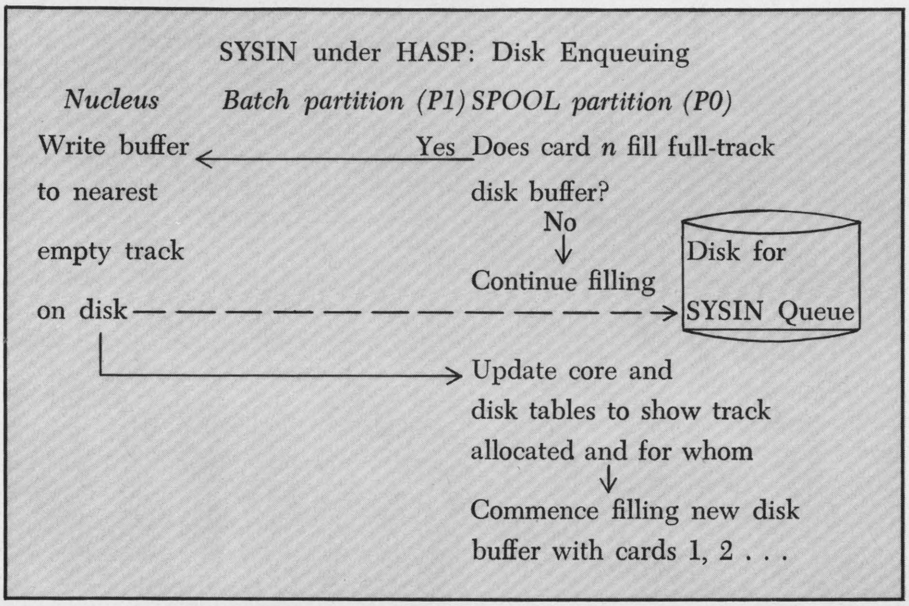
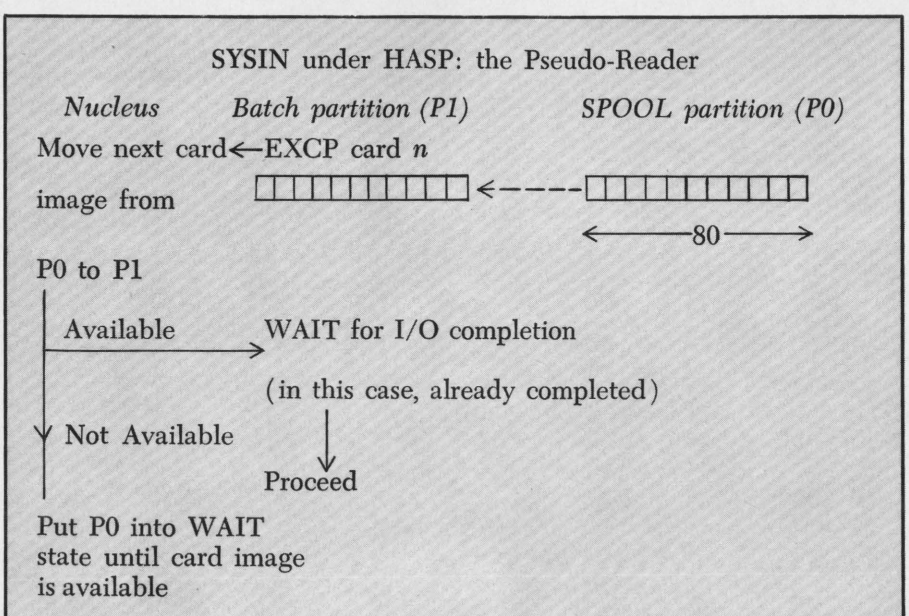


Figure 2

Figure 3



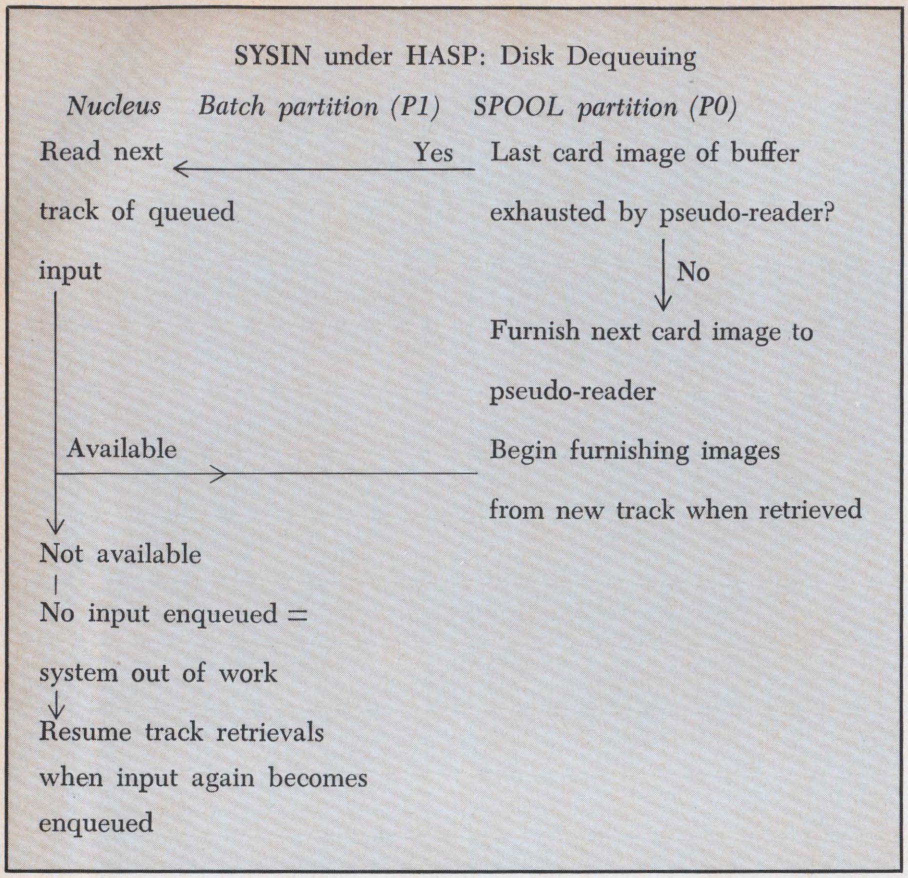


Figure 4

HASP thus levels card-reader and printer activity across jobs. Typically, a compilation, sort, or user program runs severely input-bound, then compute-bound, then output-bound. HASP overlaps card reading for future jobs with printing or already-processed jobs and simulated SYSIN/SYSPRINT for in-process jobs.

For a Model 75-class system, the following figures shows the wide range of available speeds:

Mean GET/PUT Times for One Record

Function	Unit record	Disk (without blocking)	Tape (without blocking	HASP	Disk with full-track blocking)
SYSIN	60ms.	25ms.	15ms.	1ms.	0.2ms.
SYSPRINT	60ms.	25ms.	16ms.	1ms.	0.2ms.

Table I

The reader will at once ask, "Why not issue blocked-record disk I/O directly from P1? Use HASP to SPOOL SYSIN/SYSOUT streams, but let P1 perform its own I/O." The answer has been cited above: residual device dependencies in OS/360. *Any* OS/360 processor will accept input from a card reader, *most* will accept blocked-record input from disk, the exceptions to this rule ruin the simplicity of direct input to P1 from disk.

A more fundamental advantage of HASP is the master-slave relationship between P0 and P1. If P1 selected its own jobs, P0 could not readily pre-fetch tracks of card images into core nor gulp multiple tracks of print images while P1 was writing output—or after P1 had advanced to the next job. For these and other reasons, P0 and P1 must synchronize their I/O; this is neatly effected by pseudo-devices.

The master-slave structure permits P0—which typically requires 1–10% of CPU time—to optimize reading and printing functions for P1 as follows. Every card image queued onto disk by P0 must ultimately be read back by a P1 pseudo-reader. Enqueuing is relatively "leisurely" (not critical to system throughout); dequeuing must be fast and efficient, since P1 can request track after track of SYSIN images at core-transfer speeds. The P0 track-management algorithm reflects these considerations; it writes each bufferload of newly-collected card images into an empty track at/near the current SYSIN mechanism position, to avert long mechanism movements for impending service to P1. The P0 algorithm attempts to collect multiple tracks of card images for a single job onto one disk cylinder. P0 uses an analogous algorithm to control mechanism movement for the SYSOUT disk.

Additional advantages from the pseudo-device approach include (1) automatic cutoff of jobs which exceed estimated SYSIN card counts or estimated SYSOUT lines, (2) automatic cutoff of jobs which exceed their running-time estimates, (3) "snooping" of SYSIN and SYSOUT for unwanted lines or lines which indicate special control functions to HASP, e.g. forms-change messages for remote terminals. These same advantages over standard OS/360 are offered by ASP, CLASP, and the TUCC system.

An important advantage of pseudo-readers is their simplicity for servicing multiple, concurrent job streams. Insofar as OS/360 permits, pseudo-device systems can furnish efficient SYSIN/SYSPRINT/SYSPUNCH streams to multiple partitions. The P0 function in each system can "spread" streams over distinct channels and disk drives, reducing contention for channels or access mechanisms. This optimization is *global* and *transparent* to the processing partitions, since they are accessing card-like devices rather than explicit disk data sets. When P0 detects adverse changes in the "system spread", e.g. unanticipated contention for a particular access mechanism, it can immediately remedy the situation by switching SYSIN and/or SYSPRINT to a different disk drive.

The Attached Support Processor (ASP) [7] and Closely-Linked ASP (CLASP) [8]

ASP is a two-CPU system, CLASP is a two-partition, single-CPU system with all of the support-processor functions of ASP included in P0. Their usage of pseudo-devices is almost identical; for narrative simplicity, we let P0 denote either the support processor (ASP) or the SPOOL partition (CLASP), and P1 similarly.

P1 accesses SYSIN, SYSPRINT, SYSPUNCH and a collection of tape-like devices through a channel-to-channel adapter attached to P0. To P1, these devices are indistinguishable from tapes (except that SYSIN, SYSPRINT, and SYSPUNCH are infinitely long; the other functions are usually constrained to tape-like lengths.).

Each request from P1 for a card image is translated—by a special ASP subroutine—into a channel-to-channel I/O operation with P0. Just as in HASP, P0 attempts to read ahead from real card readers, queue card images on disk, and maintain at least one track in core storage, ready to satisfy the next burst of requests from P1. The difference from HASP is principally in the pseudo-devices implementation. Whereas HASP either instantly furnishes a card image or blocks the requestor, ASP translates the request into a channel-to-channel I/O operation. Termination of the I/O operation signifies receipt of the card image, delays in the I/O operation permit P1 to continue processing while P0 services the request. If P0 is unable to satisfy the request within a few milliseconds, P1 typically WAITs on the request; P1 can resume only when the I/O operation finally terminates.

HASP uses the CPU to move images between P0 and P1 and is unable to overlap this "I/O" with processing in P1, whereas ASP uses two channels to move images, permitting processing overlap. However, if a card image is already available in P0 when P1 requests it, this overlap is inconsequential to

Programmers have a rising future at JPL.

We need programmers right now who can grab hold and fly. Who can take off in any direction. And soar with the best of them.

System Programmers — You'll be responsible for design, implementation and conversion of computer programs for real-time data processing and control function in large general-purpose computers.

Qualifications: BS in math plus three years' experience with real-time data reduction systems desirable.

Data Reduction and Processing Programmers — You'll work in JPL's Flight Telemetry Processing Group designing, developing, implementing and documenting telemetry processing computer programs in support of flight projects. Spacecraft telemetry to be processed includes engineering, scientific command and television data. Related programs such as star mapping, communications predictions and command generations are also developed in the Group.

Qualifications: BA or BS minimum with a major in math preferred, but not mandatory. One to two years' minimum experience in programming. Background in telemetry processing/data reduction preferred. Machine language (FAP, IBMAP) experience preferred.

Real-Time Programmers — You'll be responsible for design, implementation and documentation of computer program systems supporting various flight projects. Particular areas of work include real-time processing of spacecraft telemetry, ground support operations and simulation modeling.

Qualifications: BA or BS minimum with a major in math, physics, engineering or related areas. Two years' minimum experience in programming with experience in real-time applications on medium to large scale computers. Machine language experience required.

Software Research and Advanced Development Programmers — You'll do independent research and advanced development in computer program applications such as real time spacecraft simulation, telemetry data storage and retrieval, and analytical mathematics by time-shared computers.

Qualifications: Programmer/Analysts with advanced degrees, MS or beyond, and at least four years' experience in technical computer applications.

Send your resume in confidence to Mr. Wallace Peterson, Supervisor, Employment.

jpl **JET PROPULSION LABORATORY**
4800 Oak Grove Drive, Pasadena, California 91103
Attention: Professional Staffing Department 600

"An equal opportunity employer." • Jet Propulsion Laboratory is operated by the California Institute of Technology for the National Aeronautics and Space Administration.

system performance; HASP and ASP require approximately the same time, in this case. Initiation and termination of I/O requests on the channel-to-channel adapter is more complex than simulating instantaneous card reading; CPU overhead for each EXCP sequence in ASP is somewhat greater than for HASP.

In HASP, ASP, CLASP, and the TUCC system, P0 performs many additional functions such as scanning card images:

- to validate job-control information, e.g. insert delimiters if necessary
- to determine job characteristics for specialized system-loading, e.g. automatically select a compute-bound job to run concurrently with an I/O-bound job
- to determine needs for private tape reels or disk packs; P0 issues volume-mounting instructions to the machine operator before any system resources are committed to a job
- to identify short, non-stop jobs; these are normally given expedited service, and they can also be processed as "filler" work while tape reels are being mounted for a setup job.

The Westinghouse Remote Terminal Operating System [9]

Connected to a large central S/360 at Westinghouse are various card-oriented satellite computers. The satellites are typically several miles away, linked to the central facility through broad-band synchronous transmission adapters and common-carrier telephone lines. The satellites operate as pseudo-devices attached to pseudo-controllers; whenever they are not initiating I/O to the central facility—or pre-processing input to this facility or post-processing output—they work exclusively on replenishing their core-buffers with card and print images.

In the central facility, eight pseudo-devices represent each satellite: console, reader, punch, two printers, plotter, paper tape reader, and paper tape punch. A single communications-IOCS logic serves all satellites. Thus, I/O operations for the pseudo-reader are translated into communications-line READs which in turn initiate communications-line WRITEs from the satellite device. Just as in HASP, ASP, and CLASP, the communications IOCS tries to keep a stack of card images in the central machine, to satisfy bursts of requests. Likewise, it permits accumulation of a stack of print images before interlocking the pseudo-printer; this stack is quickly reduced by sending images to the appropriate satellite at broadband speed (5KB).

Thus, each of several partitions in the Westinghouse system can operate with card input and printer output—the actual devices are located where data is stored, programmers and engineers work, etc.

The TUCC Remote Job Entry System [2–5]

Connected to the Model 75 at TUCC are the following satellites:

- IBM 360/30s and 360/40s, using Tape and Disk Operating Systems/360 as well as the MFT option of OS/360; over broadband lines;
- IBM 1130s (small scientific computers); over voice-grade lines;
- IBM 1978s (reader/printer terminals); over voice-grade lines;
- IBM 1050s (typewriter/card-reader terminals); through 15 cps adapters;
- IBM 2741s (typewriter terminals); through 15 cps adapters;
- Teletype 33–35 ASRs (typewriter/paper-tape terminals); through 10 cps adapters.

The system is not constrained to these satellites; others under consideration include the following:

- IBM 360/30s connected over voice-grade lines;
- Univac DCT 2000s (reader/printer terminals); over voice-grade lines;
- IBM 2780s (reader/printer terminals); over voice-grade lines;
- PDP-8s (small scientific computers); over voice-grade lines.

Of these terminals, only the 2741s are inappropriate for *remote job entry* (RJE), i.e. batched jobs from a satellite. 1050s and Teletypes can be used for either RJE or interactive computing. (TUCC offers the latter facility through the IBM–Allen Babcock Corporation Conversational Programming System. CPS operates independently of the RJE system.)

To support RJE, TUCC uses two pseudo-readers, two pseudo-punches, two pseudo-printers, and one pseudo-disk. (Although it is merely another pseudo-device, the latter was christened *hyperdisk* a year ago; the latter designation will be retained, for consistency with other documentation.)

All but the hyperdisk are quite similar to the HASP implementation; they were designed at approximately the same date, although HASP was completed somewhat earlier.

Just as in HASP, most sources of input—the local card reader and all terminals accessed over broadband and voice-grade lines—are continually interrogated by P0 to collect jobs. The 10–15 cps. lines are dial-up; as soon as a TUCC number has been successfully dialed by a low-speed satellite, TUCC types out all output from jobs previously submitted (if any). Immediately thereafter, TUCC reads input until End of Transmission to conclude a single dialogue.

Likewise, TUCC writes already-processed job output to printers, punches, and plotters (360/30s and 360/40s only) as soon as (a) a terminal indicates readiness to receive and (b) all input to the central facility has been sent. The entire process of establishing/breaking contact with TUCC is automatic at the central facility; P0 requires no operator intervention during each dialogue. Accounting information is automatically accumulated on disk, so that each job receives appropriate billing for its usage of machine resources. From individual transactions, monthly invoices are accumulated and written by computer. (Invoices themselves are sent by U.S. Mail, but we are confident this is only a temporary inconvenience for a computer utility!)

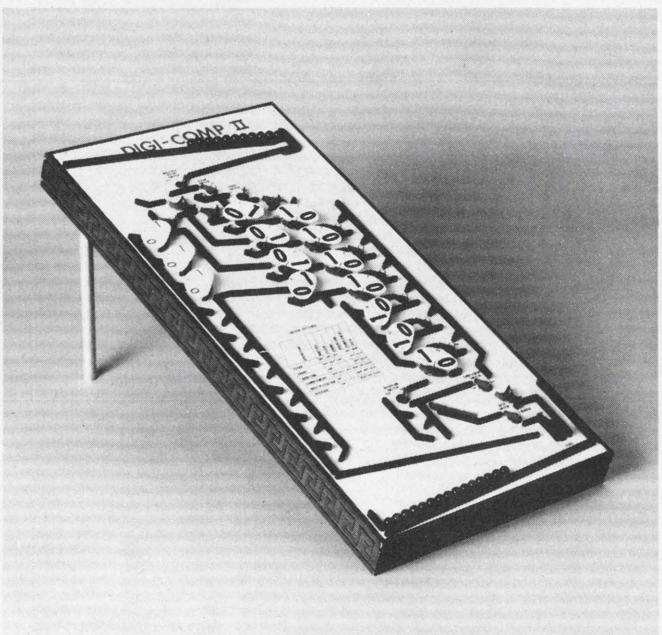
All jobs are collected into a common queue of work which feeds two processing partitions. One partition contains a small-job monitor which functionally replaces the OS/360 Job Scheduler for that partition. P0 gives this partition small-core jobs requiring no setup and little I/O. These jobs comprise almost half of the TUCC load, by number. Their CPU-time requirements are typically 0.1–10 secs. They aggregate only 10% of our total computational load. However, a large proportion of these

(Continued on page 34)

Now . . .

DIGICOMP II - the "second generation" of the famous DigiComp,

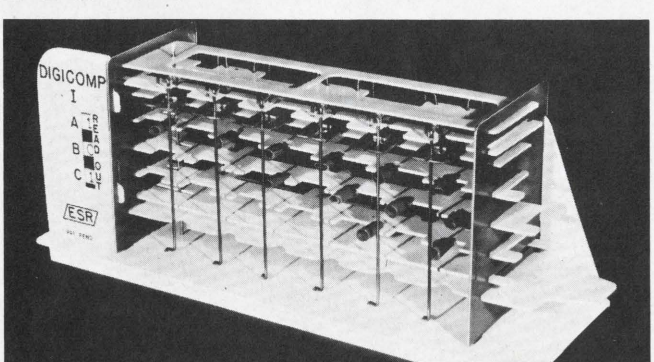
enormously more "sophisticated" as additional "switches" and "core positions" permit many more decisions and "branching" possibilities.



Like its predecessor the DigiComp-I (see below) it is entirely mechanical in operation but affords the functions common to electronic computers in a form far more comprehensible. The best way ever to learn (or to teach) the principles of modern data processing.

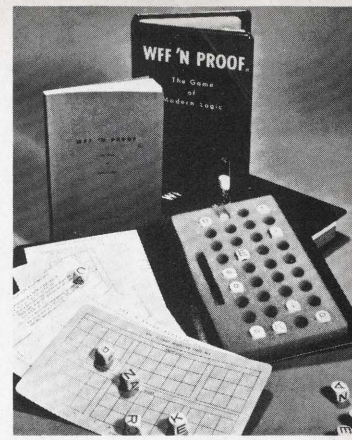
$16.95 Postpaid

DIGICOMP-I COMPUTER



Ingenious mechanical equivalent of an electronic digital computer. Can be "programmed" to perform, individually, every operation of a digital computer. Best thing ever for teaching binary math, how a computer operates, how and why problems must be broken down for machine solution. In kit form—1 hr. to assemble. $5.98 Postpaid

WFF 'N PROOF



Developed in the course of the ALL Project (Accelerated Learning of Logic) Wff'nProof incorporates 21 games of progressive difficulty; has been used to teach propositional calculus to elementary school children but challenges the intellect of professional logicians. Complete with 224-page manual which is an introduction to the world of modern logic; filled with valuable tips to clear thinking.

$6.25 Postpaid

OH WAH REE

The ancient game of mathematical strategy which has intrigued millions for more than 30 centuries. A game at once so simple in its basics that a pre-schooler can enjoy it yet so complex in its possibilities that its fascination multiplies with the mathematical grasp of the player.

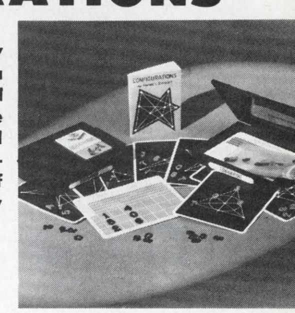
$7.95 Postpaid

TWIXT

Another famous "bookshelf" game, package matches that of Oh-Wah-Ree. A stimulating game with a fascinating chess-like strategy of move and countermove. Two or four players.

$7.95 Postpaid

CONFIGURATIONS



Based on concepts from the geometry of incidence, CONFIGERATIONS is a series of intriguing mathematical and geometric puzzles that will challenge and delight those who enjoy careful reasoning. It is authored by Professor Harold L. Dorwart, Chairman of the Mathematics Department, Trinity College, Hartford.

$4.95 Postpaid

SCIENCE/SYSTEMS P. O. Box 1176 Evanston, Ill. 60204 S/A-6

Please send me at the address below:

. . . .

. . . .

. . . .

. . . .

. . . .

(Games are shipped from separate warehouses)

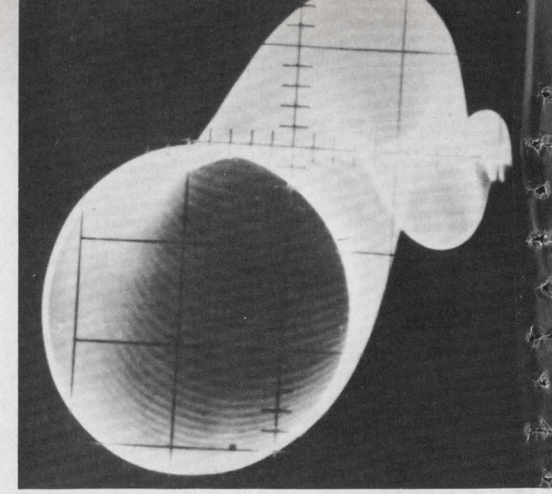
(name)

(address)

(city) (state) (zip code)

Illinois residents add 5% OES

The Computer Meets The Shell



■ The shell game—or, the geometric analysis of the theoretical morphology of the coiled shell—and the computer.

The what?

The study of shells . . . all kinds of shells . . . snail shells, clam shells, oyster shells. Even the study of shell forms which don't exist in nature.

How?

With a modern analog computer.

Professor David M. Raup of the University of Rochester's Department of Geological Sciences has been involved in extensive studies of shell geometries. In his studies, he has used both digital and analog simulation methods. "The digital method has the disadvantage that it is relatively costly in terms of computer time. A much cheaper and nearly as rigorous method has been developed in which a standard Electronic Associates, Inc. TR-20 Analog Computer and oscilloscope are used."

The study, according to Prof. Raup, was designed to analyze the functional significance of the coiled shell. In general, it was found that the geometry of a shell form was directly related to biological phenomena.

The coiled shell, in geometric terms, is a tapered, hollow tube. It is open at its larger end and is coiled on a fixed axis. Growth takes place principally at the open end.

To develop the analog computer program, it was necessary to develop a mathematical model representing the geometry of the shell. This was done. The model has four principal parameters which can be adjusted, with the result being different geometric forms.

The first parameter is the shape of the generating curve—in general, the outline of the shell's growing edge or its open end. The second parameter is the whorl expansion rate—the rate of increase in the generating curve per revolution. The third is the position of the generating curve in relation to the coiling axis—the distance between the open end and the axis. Last is the rate of whorl translation—the rate of movement of the generating curve (open end) along the axis per revolution.

With these four variables, Prof. Raup was then able to simulate the entire spectrum of geometrically possible shell forms. By plotting the geometries of existing shell forms, it was found that some regions of the spectrum are essentially empty with regard to natural forms.

The conclusion drawn from the analog computer study was that the forms in the empty regions are geometrically possible—but biologically impossible because of functional inefficiency.

This dependence of form on biological evolution holds true with fossils as well as with actual, existing shell forms. For example, the shell of a bivalve (a clam) must have a functional hinge. Therefore, the whorls of the shell must not overlap.

In the analog simulation, a circle was generated and was traced rapidly enough to appear solid. Its offset from both the horizontal and vertical axes was controlled. The circle, simulating the generating curve, was rotated around the vertical axis, simulating the coiling axis. The circle became smaller as it rotated, thus simulated shell growth in reverse.

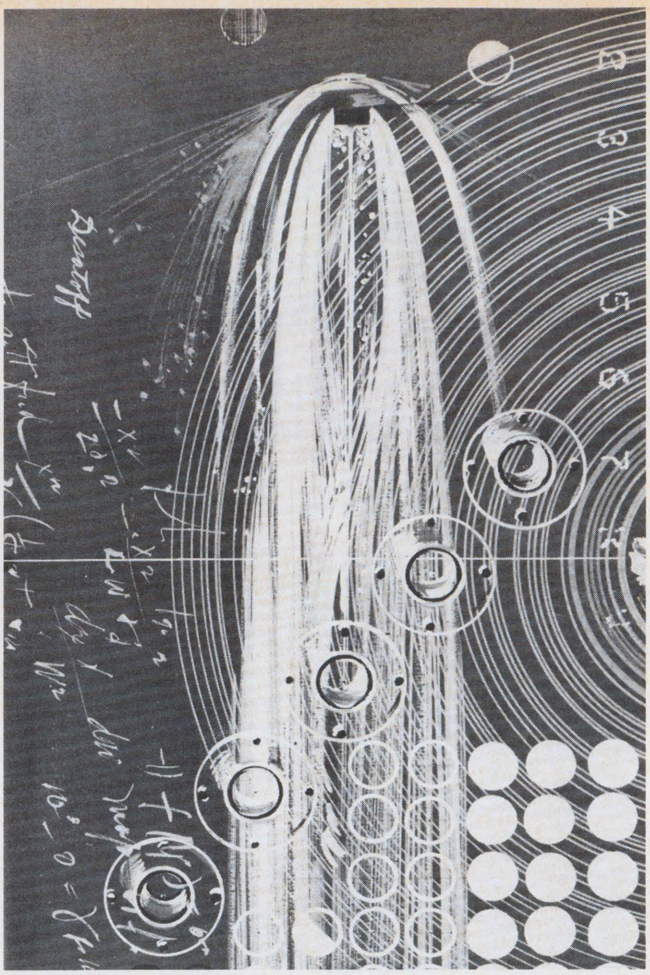
The translation of the whorl was introduced by varying the vertical offset of the initial circle. The relative distance of the generating curve from the coiling axis was established by the initial horiziontal offset.

The result, with pictures of each trace, was the format for Prof. Raup's analysis of the functional significance of variation in coiling geometry. ■

Efficient Programming in FORTRAN

Charles Erwin Cohn
Argonne National Laboratory
Argonne, Illinois

A number of techniques can be used to decrease the running-time and memory-space requirements for FORTRAN programs. The usage of these depends upon the characteristics of a particular compiler and system.



The above art and cover by John Desatoff of TRW Systems.

Introduction

■ Real mastery of a problem-oriented language implies the ability to describe a job so that it will be done efficiently, i.e. with a minimum of unnecessary extra operations. At present, concern for program efficiency is considered archaic by many people. However, it assumes renewed importance with the advent of the many small computers which can execute problem-oriented languages, in particular FORTRAN. It does not take much of a FORTRAN program to tax the capabilities of these small machines. Therefore, efficiency can make the difference between being able to run such a program in a straightforward manner, or, on the other hand, having to segment the program, resort to machine language, or look for a larger computer—all of which are inconvenient.

The measures that are necessary to improve program efficiency depend upon how optimally each type of FORTRAN statement is handled by a given compiler. In general, the compilers for the smaller machines do less well in this respect, so that more attention is required from the programmer to obtain optimal code. Therefore, the programmer must know the characteristics of the particular compiler which he uses. Because of the variety of compilers now available and the rate at which new ones are being introduced, it is not practical to give those characteristics here. They can be ascertained by examination of the machine-language output produced by the compiler from some benchmark programs. (Ideally, such information should be provided by the computer manufacturers.)

Normally, the measures discussed in this paper save both running time and memory space. (It is the latter that normally limits the size of problem that can be run on a small computer.) Where a trade-off between the two is involved, that will be noted.

Although the discussion in this paper is in terms of FORTRAN, it applies in general to any problem-oriented language. The points discussed may seem trivial, but it is the author's experience that many "professional" programmers and practically all "amateur" programmers are unaware of them. They are not covered in texts.

Arithmetic Expressions and Replacement Statements

Arithmetic expressions and replacement statements can be optimized by eliminating the repeated calculation of redundant subexpressions. In the statement

Z = (A*B/C)*SIN(A*B/C),

the redundant subexpression (A*B/C), although appearing twice, should be calculated only once and the result saved until it is needed again. Most compilers will do this automatically if the redundant subexpression is set off by being enclosed in parentheses, as shown here.

If a particularly rudimentary compiler does

* Work performed under the auspices of the U. S. Atomic Energy Commission.

not optimize this case automatically, the programmer must do it as follows:

```
TEMP1 = A*B/C
Z = TEMP1*SIN(TEMP1)
```

Practically no compilers, except the most sophisticated, will optimize subexpressions that are redundant between two or more arithmetic replacement statements. These must invariably be optimized by the programmer. For example, the statements

```
X = SIN(A*B/C)
Y = CØS(A*B/C)
```

should instead be written:

```
TEMP1 = A*B/C
X = SIN(TEMP1)
Y = CØS(TEMP1).
```

In both of the above two cases, additional running time and memory space are required for the instructions that store the result of the redundant subexpression in the memory location assigned to the variable TEMP1. This extra time and space is outweighed, however, by the savings resulting from elimination of the instructions required to calculate the subexpression a second time. The net savings become greater, of course, as the subexpression becomes more complicated and as it is used a greater number of times. The saving in time is especially noteworthy, as floating-point operations are unduly time-consuming on small computers that do not have floating-point hardware. The memory space consumed by the temporary-storage variable TEMP1 can be used most efficiently by using the same name wherever temporary storage is required.

If a loop contains an expression whose variables do not change value during the course of the loop, time (but not space) may be saved by evaluating the expression once, outside the loop, and holding the result until needed. For example, the loop

```
      DØ14I = 1,N
14    Y(I) = A*B*X(I)/C
```

can be rewritten as

```
      TEMP1 = A*B/C
      DØ14I = 1,N
14    Y(I) = TEMP1*X(I)
```

The latter version may incur a slight space penalty from the extra instructions needed to store and retrieve the result of the expression.

Arithmetic operations on whole numbers are best done in integer mode, with the results converted to real where needed.

Constants

Where mixed-mode arithmetic is allowed, it is best to write constants in the dominant mode of the expression to avoid needless conversions. For the expression 2*A, most compilers will store the 2 as an integer and convert it to real each time the expression is evaluated. With the expression in the form 2.*A (i.e. with the decimal point shown) the constant is stored as real and the conversion is eliminated.

Arithmetic operations on constants should be performed by the programmer before writing an expression. In the expression 4.*A/3. the two constants are stored separately by most compilers and the division is performed each time the expression is evaluated. It should instead be written 1.333333*A. This and other transcendental constants should be written to as many significant figures as the computer handles in its arithmetic, so that full advantage is taken of the computer's precision. There is no penalty for the additional digits.

Some compilers store constants only as magnitudes. Where a negative constant is used as a subprogram argument, extra instructions to negate the constant are inserted. Where the same negative constant is used in more than one argument list, it is efficient in such a system to assign the value to a variable and use the variable name in the argument lists. For example, instead of

```
CALL SUBRA(W,X,—3)
. . .
CALL SUBRB(Y,Z,—3)
```

you would write

```
M3 = —3
. . .
CALL SUBRA(W,X,M3)
. . .
CALL SUBRB(Y,Z,M3)
```

Powers

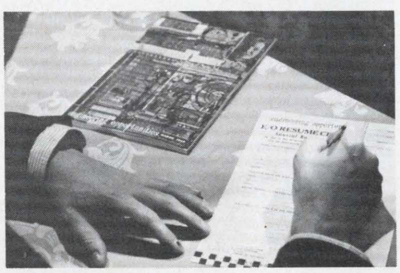
In many compilers, the use of the "**" notation calls upon a special subroutine in the library. If only small whole-number powers are to be calculated, the time and space required for this subroutine can be saved by avoiding the "**" notation. Thus, for example, for X ** 2 write X * X, for X ** 3 write X * X * X, and for X ** 4 write (X * X)*(X * X) (with the redundant subexpression (X * X) handled as described above).

Where the use of the "**" notation is appropriate, the mode of the exponent can make a difference. That is because many systems use different library subroutines for real or integer exponents of real arguments. If such a program already contains a real exponent, memory space can be saved by making whole-number exponents real, thus eliminating any need for the integer-exponent subroutine. Other systems have a subroutine for real exponents only, and convert all integer exponents to real before calling the subroutine. There, the exponents might as well be shown as real to begin with, thus eliminating the conversion step.

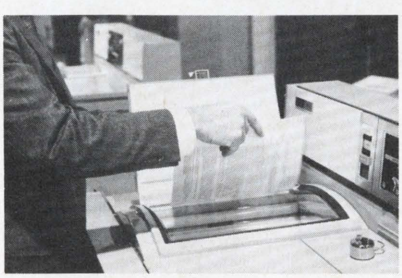
can an engineer attending the WESCON show find that better job <u>without</u> paying a fee...<u>without</u> letting the world know he's looking?

sure he can!

read how engineering opportunities' exclusive resume center protects your privacy . . . speeds your search . . . costs you nothing!

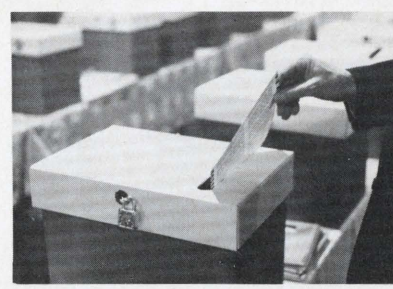


1. You Hand Pick the Companies. 75 top companies in all areas and fields will be represented in the Resume Center. In similar operations you may have attended, all companies represented at the center saw your coded resume whether you were interested in them or not. *At the ENGINEERING OPPORTUNITIES Resume Center, you will deposit your resume in only those company lock boxes which are of interest to you!* No one else sees or touches your resume. You have complete control.



2. Free Resume Duplication Service. Special resume forms will be available for you to fill in your hotel, room, telephone number. Tell our Xerox 2400* operator how many free copies of your resume you would like. They will be duplicated on the spot, in front of you, and *all* returned to you immediately at no cost.

* Trademark of Xerox Corporation



3. You Hand-Deposit Your Resume. Next, you'll drop your resume in the lock box of the company you're interested in. The lock box is under lock and key. And only the company interviewer has the key.

4. You Make Your Own Appointment, Your Own Arrangements. If the company interviewer is interested in you, he will leave a message at your hotel for you to give him a call. You will set up the interview with him at your convenience. *All arrangements are made by you and the company. No one else.* There's no middleman, and no fee to anyone. You or the company.

As the nation's leading magazine devoted to career advancement, ENGINEERING OPPORTUNITIES created the E/O RESUME CENTER to help you find the job you want with the company you want. Without cost. And without ever surrendering the controls. You control your own job search.

engineering opportunities resume center

Los Angeles, Calif., International Hotel, Sepulveda at Century Blvd.
Noon, Monday, August 19—Noon, Thursday, August 22

The first non-profit central clearing house of job opportunities, managed by a publication. Another service for engineers from the only publication dedicated to their career advancement. ENGINEERING OPPORTUNITIES MAGAZINE, 1020 Church St., Evanston, Ill. 60201
Norman Jacobs, Advertising Director

Polynomials

Optimization of polynomials is useful for any compiler. For a polynomial of the form

```
Y = A + B * X + C * X ** 2 + D * X **3,
```

calculation requires three additions, three multiplications, and two exponentiations. There is a saving if the polynomial is instead written in *nested* form as

```
Y= A + X * (B + X * (C + X * D))
```

which is obviously equivalent. Here, the additions and multiplications remain but the exponentiations have been eliminated. It is straightforward to put any polynomial into this optimal form.

Special treatment is needed when one of the terms carries a minus sign. Since, in the nested form, the minus sign would multiply all subsequent terms, the next term must also carry a minus sign to cancel the effect of the previous minus. Thus, for example, the polynomial

```
Z = A + B * X − C * X ** 2 + D * X ** 3
  + E * X ** 4
```

would be written in nested form as

```
Z = A + X * (B − X * (C − X * (D + X
  * E))).
```

Care must be taken, of course, to close the expression with the correct number of parentheses. The coefficients, shown here as single variables, may be expressions enclosed in parentheses, with any redundancies handled as described above.

Statement Numbers

Ordinarily, there is no penalty for attaching a number to a statement even when not needed for reference by another statement. However, such a penalty can arise under two special circumstances.

First, some small-machine compilers are very limited in the size of program that they can compile because of limited memory space for the assignment tables that keep track of variables, statement numbers, etc. There, elimination of unneeded statement numbers will reduce the burden on the available space.

Secondly, some compilers perform limited optimization on sequences of arithmetic replacement statements. In particular, a variable needed in one statement may not have to be fetched from memory if it is already in a register as the result of a previous statement. This optimization is done only if none of the statements has a number, indicating that the sequence is never entered in the middle. If the last statement in the sequence ends the range of a DO, its number can be removed and attached to a CONTINUE statement following. There is never a penalty for the use of a CONTINUE statement.

IF Statements

Where the quantity calculated in the expression imbedded in an IF statement, or any part of it, is also used earlier or later in the program, the unnecessary repeated calculations of that quantity should be avoided as described above for redundant subexpressions. For example, the program segment

```
    X = A*B+E
    IF(A*B-C*D) 1,2,2
  1 Y = C*D+F
  2 . . .
```

can be optimized as

```
    TEMP1 = A*B
    TEMP2 = C*D
    X = TEMP1 + E
    IF(TEMP1 - TEMP2) 1,2,2
  1 Y = TEMP2 + F
  2 . . .
```

(Note that here two variables are needed for temporary storage.)

The FORTRAN-IV logical IF statement is not handled efficiently by some compilers. These set up an intermediate logical variable according to the results of the relational operations specified, and then test this logical variable to determine the outcome of the IF statement. With such a compiler it is more efficient to replace the logical IF statement with one or more three-branch IF statements as required.

Subscripted Variables

Retrieval or storage of subscripted variables always requires more work than the retrieval or storage of unsubscripted variables. This is because the address of the datum must be calculated from the subscript combination. In more advanced systems, this arithmetic is done through indexing so no additional time or space is required that can readily be eliminated. However, less-advanced compilers insert additional instructions to perform address arithmetic wherever a subscripted variable is referenced. Here, it helps to cut down on such references.

In the statement

```
C(I) = C(I) + X
```

there are two references to the same subscripted variable. A well-designed compiler will perform the necessary address arithmetic only once and save the result until needed. If a compiler is not so optimized, nothing can be done here, because the two references to the subscripted variable are on opposite sides of the equal sign.

In the statement

```
Z = C(I)*SIN(C(I)*D) ,
```

the compiler should again do the address arithmetic only once for the two references. If that is not the case, the two references can here be treated as redundant subexpressions as explained previously, because they appear on the same side of the equal sign.

Where the same subscripted variable is referenced in two or more statements, these references may be handled as redundant subexpressions, provided that the values of the subscripts do not change through the sequence. The statements

```
      DØ1I = 1,M
      IF INDEX(I,1) – INDEX(I,2))2,1,2
    2 JRØW = INDEX(I,1)
      JCØLUM = INDEX(I,2)
    1 CØNTINUE
```

may be optimized as

```
      DØ1I = 1,M
      ITEMP1 = INDEX(I,1)
      ITEMP2 = INDEX(I,2)
      IF(ITEMP1 – ITEMP2)2,1,2
    2 JRØW = ITEMP1
      JCØLUM = ITEMP2
    1 CØNTINUE
```

Where the value of a subscripted variable is formed in one statement and used in a subsequent statement, the extra reference may be eliminated similarly. The statements

```
      DØ 4I = 1,N
      C(I) = A(I) + B(I)
    4 WRITE(5,1000)A(I), B(I), C(I)
```

may be changed to

```
      DØ 4I = 1,N
      TEMP1 = A(I)
      TEMP2 = B(I)
      TEMP3 = TEMP1 + TEMP2
         C(I) = TEMP3
    4 WRITE(5,100) TEMP1, TEMP2, TEMP3 .
```

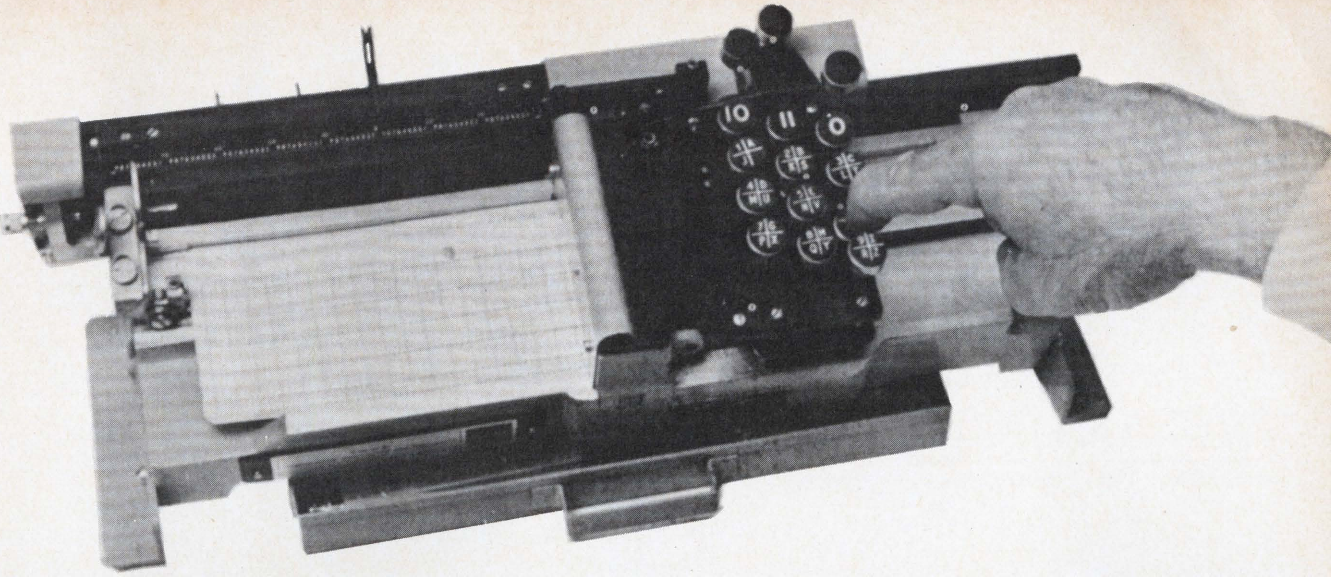
However, if one of the statements has the subscripted variable on both sides of the equal sign, such a change might not save anything if the compiler would do the address arithmetic only once for the statement in any case.

Subscripted-variable references with constant subscripts need not be handled in this way. Most compilers will perform the address arithmetic during compilation, so that the subscript reference incurs no penalty. For those systems which leave even this address arithmetic until program execution, an unsubscripted variable name may be made equivalent to the element in question and used in all references. Thus, the coding

```
  DIMENSIØN B(38)
  . . .
  B(14) = . . .
```

may be replaced by

```
  DIMENSIØN B(38)
  EQUIVALENCE (B14, B(14))
  . . .
  B14 = . . .
```



Sometimes a two- or three-dimensional array may be handled more efficiently by making it equivalent to a one-dimensional array. The initialization of a matrix to zero is usually written as

```
      DIMENSIØN A(20,10)
      . . .
      DØ 3 J = 1,10
      DØ 3 I = 1,20
    3 A (I,J) = 0.
```

This may be done more efficiently with any compiler by

```
      DIMENSIØN A(20,10), AA(200)
      EQUIVALENCE (A,AA)
      . . .
      DØ 3 I = 1,200
    3 AA(I) = 0
```

which saves some address arithmetic as well as the instructions for the inner DO loop. This approach clearly offers an advantage only in those special cases, such as the one shown, in which computation of the subscript is not necessary.

Input–Output Statements

The input or output of an entire array may be specified either by a DO-implying loop over the array or by mention of the name of the array with no qualification. In many systems, the latter saves space by causing fewer instructions to be compiled, and also saves central-processor time.

In many systems there is a space penalty for specifying additional input-output modes, since each mode requires its own subroutine from the library. For example, consider a program whose primary output is on a line printer. If the programmer decides to include monitor output on the console typewriter for the convenience of the operator, the space penalty incurred may include the typewriter-output subroutine in addition to the coding for the output statements. If memory space is critical, it might be best to forego the monitoı output or take it on the line printer.

Subprograms

Use of subprogram organization incurs a time and space penalty because of the linkage instructions. The space penalty is more than made up, however, if the subprogram is called from more than one place in the main program, because the coding to perform the subprogram's functions need then not be repeated at each place it is needed. If, instead, a subprogram is called from only one place in the main program, it is most efficient to eliminate its separate identity as a subprogram and incorporate it directly into the main program. Where a subprogram is a function that can be executed in one replacement statement, it may best be included in the main program as an arithmetic statement function (but see below).

Attention must also be paid to the manner of linking variables between a main program and a subprogram. There are two ways of doing this —through argument lists and through COMMON statements—and each has its proper role. There is a time and space penalty associated with the use of argument lists. The subprogram must contain coding that will fetch the address of an argument from the main program and plant that address where it is accessible to those instructions in the subprogram that require it. With COMMON linkage, on the other hand, there is no penalty.

Therefore, we may state the following rule: where a variable in the subprogram always corresponds to the same variable in the main program at every call of the subprogram, then the linkage should be through COMMON (or parameters, for an arithmetic statement function). On the other hand, where a variable in the subprogram may correspond to different variables in the main program at different calls of the subprogram, then the linkage should be through the argument list. (Of course, a reference to a FUNCTION subprogram must always have at least one argument so that the compiler may distinguish it from a reference to an ordinary variable.)

There is a time penalty and there may be a space penalty associated with each additional reference to an argument within a subprogram. Therefore, if an unsubscripted variable is referenced more than once in a subprogram, it could be worthwhile to use a local variable in its stead. The local variable is made equal to the argument or vice versa at the beginning or end of the subprogram, depending upon whether the argument is an input or output variable. Examples are as follows:

```
    FUNCTIØN PØLY(X)
    CØMMØN A,B,C,D
    XA = X
    PØLY = A + XA*(B + XA*(C + XA*D))
    RETURN
    END
    SUBRØUTINE SUM(TØTAL,X,N)
    DIMENSIØN X(N)
    TEMP = 0.
    DØ1I = 1,N
  1 TEMP = TEMP + X(I)
    TØTAL = TEMP
    RETURN
    END
```

(For an array, this substitution would of course require a DO loop. It may or may not be worthwhile, depending upon the length of the array and the number of times it is referenced in the subprogram.)

Some compilers handle arithmetic statement functions like open subroutines or macros, repeating the instructions for the function at each place where it is called in the program. This saves a little time (by eliminating linkage) at the cost of much space. If a program on such a system is space-limited, the arithmetic statement functions should be replaced with function subprograms.

COMMON storage has an important use in addition to the linkage of variables. In most systems, variables declared as COMMON (blank COMMON for FORTRAN-IV) are assigned to the memory area that is occupied by the loader during object-program loading. This space is otherwise unavailable to the FORTRAN programmer. Therefore, a program that taxes memory can obtain some relief by use of COMMON storage, even when subprogram linkage is not in question. For best use to be made of this feature, enough arrays and unsubscripted variables should be declared COMMON to fill the loader area.

Conclusion

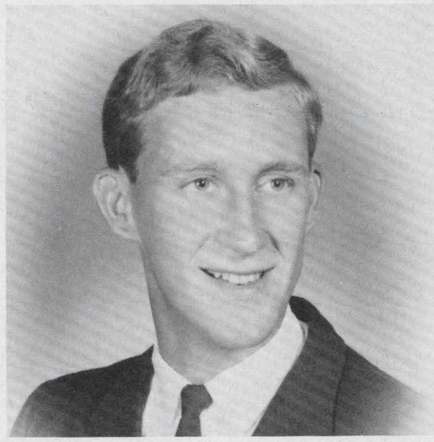
We have seen how the efficiency of a FORTRAN program can be improved through a number of measures, depending on the properties of the compiler and system. The saving from each of these measures is small individually, but throughout a program their sum total can be quite significant. If applied to all the production programs in a computer installation, a worthwhile reduction could be made in the installation's workload. ■

Software Age Announces

The Winners of Trouble-Tran Problem 1

(March p. 38)

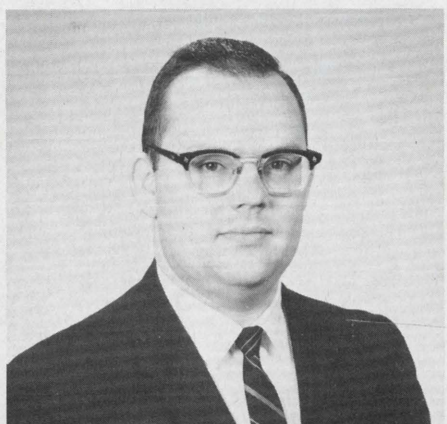
FIRST PLACE



DANIEL R. HILL
1309 Berwin Ave., Kettering, Ohio 45429

Mr. Hill works as a scientific programmer for Monsanto Research Corp. at Mound Laboratory, Miamisburg, Ohio. In 1967, he graduated from the University of Dayton where his major was Computer Science. Most of his work involves programming Mass Spectographic and Crystalographic techniques and he has programmed primarily in FORTRAN.

SECOND PLACE



C. W. WILSON
212 Iroquois Rd., Oak Ridge, Tenn. 37830

Mr. Wilson has been employed as a mathematician by Union Carbide Corp.—Nuclear Division at Oak Ridge, Tenn., since 1960. He received both his bachelor's and master's degrees from Vanderbilt University.

BY GEORGE N. VASSILAKIS/TRW ART: A. DOUGHERTY/TRW

TROUBLE-TRAN

for fun and profit

Send your ANSWER to the problems posed here in each issue to:

TROUBLE-TRAN EDITOR

software age

1020 Church St., Evanston, Illinois 60201

You can also profit by submitting PROBLEMS for this feature. If your problem in FORTRAN programming is selected for use in this feature, you will receive $25.00

Contest Rules:

1. ASA standard FORTRAN IV is assumed.
2. Version 12 of the IBM 7094 FORTRAN IV Compiler is used in verifying answers.

The correct answer bearing the earliest postmark will net the reader submitting it $25.00

The second correct answer with earliest postmark wins $15.00

XTRAN'S ANSWER TO LAST MONTH'S PROBLEM

Answer to Problem 3:

As you recall, we had a main program using a function subprogram SIGN. The program had a statement A = SIGN (ALPHA, BETA), and the problem was to find out what the value of A would be after the execution of this statement.

If your answer is A = — 1, you are wrong. The answer is A = + 90. You are, of course, aware of the existence of the FORTRAN Built-in Functions, and you may have recognized SIGN as such a function. However, you may have argued that the compiler should have used the function which was supplied by the programmer, as it would have done in the case of SIN, COS, or TAN.

SIN is a library function subprogram and when it is used the compiler generates a CALL to it, and the Loader loads first the routines supplied by the user.

In the case of a Built-in Function, the compiler does not generate a CALL; instead it provides the necessary on-line code to perform whatever function is requested. Therefore, the function SIGN in our problem was loaded but never used at execution time.

Since this could happen to you, I would suggest you take time to check the list of Built-in Functions every time you write a function subprogram.

Additional comments on Problems 1 and 2:

The answers to both problems depend on the particular compiler one uses. The statement 100 FORMAT (X3H) = (I1O) in problem 1, subroutine YTRAN, was classified as a FORMAT by the IBM System/360 H-level compiler, as Arithmetic Statement Function by the G-level compiler, and as illegal statement by the E-level compiler. The statement READ (2,100) (X (I), Y (I), I = K, KK), in problem 2, includes an implied DO. After the READ is executed, the value of location I may be I = K, I = KK, I = KK + 1 or whatever value I had before the READ.

P. S. I would like to thank those of you who have sent problems for consideration as TROUBLE-TRAN material. Such problems will be considered for publication starting in September.

XTRAN

PSEUDO DEVICES IN OS/360
(Continued from page 18)

jobs are debugging runs; turnaround demands on the small-job partition are severe.

Most users will wait no more than 20 minutes for turnaround. Thus, the small-job partition is idle much of the day—but it gives excellent average turnaround, as a direct consequence. It also serves to "soak up" I/O-wait time from other-partitions, since it has the lowest priority for CPU control.

The hyperdisk is a pseudo-device unique to TUCC; it is described in detail in [4]. It comprises a disk simulator (for a single 2314 drive), 1.5 million bytes of LCS, and ⅔ of a real 2314 disk pack. Most system data sets described in the "Introduction" are resident on the hyperdisk:

- SYSJOBQE
- PROCLIB
- LINKLIB
- SYSUTx, SYSLIN, and SYSLMOD
- FORTLIB, PL1LIB, ALGLIB, COLBLIB, etc.

SYSJOBQE, SYSUTx, SYSLIN, and SYSLMOD are *write-read* data sets; they are re-written by each job and do not survive from one job to the next (except for a few entries in SYSJOBQE, which survive across jobs in each processing partition). Write-read data sets are small, on the average. However, certain jobs will—without advance warning—require immense amounts of write-read storage. It is almost impossible for application programmers to predict these needs, and manifestly impossible for P0 to predict these needs. Prior to the hyperdisk concept, there seemed to be no satisfactory technique for processing compilations, assemblies, and sorts entirely in core storage (fast core or LCS) without *a priori* information about job sizes. Small jobs could indeed be processed in-core; large jobs obviously required disk storage for intermediate text. Distinguishing "small" from "large" jobs would require talented, co-operative—and lucky—users.

The hyperdisk "writes" and "reads" track images in LCS until the latter becomes full. It then begins to spill images of inactive tracks into its private disk pack. Figures 5–7 show, respectively, the simulation of a READ operation issued to the hyperdisk, the structure of the hyperdisk area on the real disk, and the tables and chained lists controlling track retention.

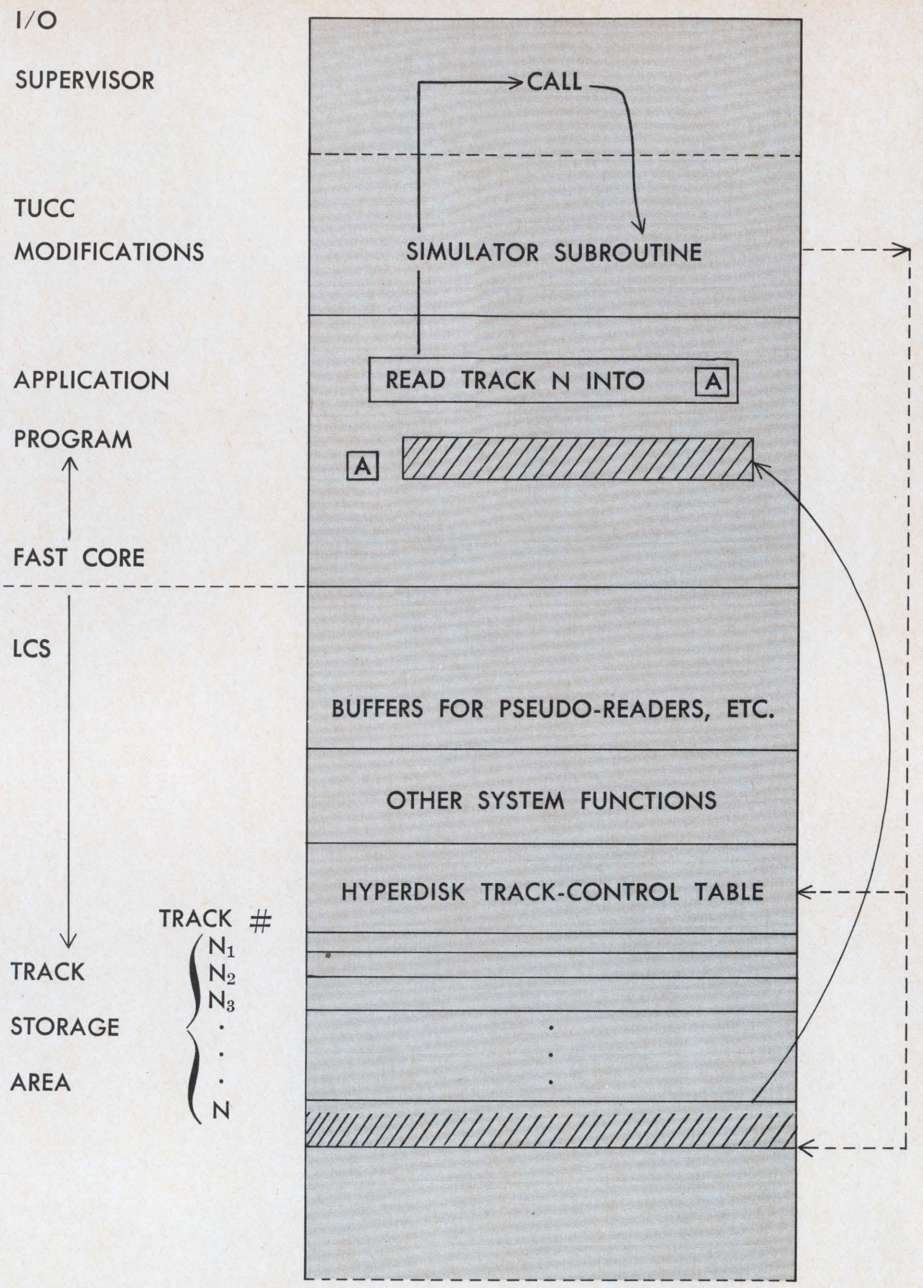


Figure 5

CORE MAP OF A HYPERDISK SIMULATION USING ONLY LCS

As shown in Figure 6, read-only data sets comprise half of the hyperdisk. Control of track areas in LCS for read-only data sets is even easier than for write-read data sets. Whenever tracks from the latter drift to the bottom of the LUFO (longest unused, first out) chain, they must be spilled to real disk. Whenever tracks from read-only data sets drift to the bottom of the LUFO chain, they are discarded; their LCS areas can be instantly reclaimed for more urgent hyperdisk needs, since—by definition—a perfect copy of each read-only track remains on disk.

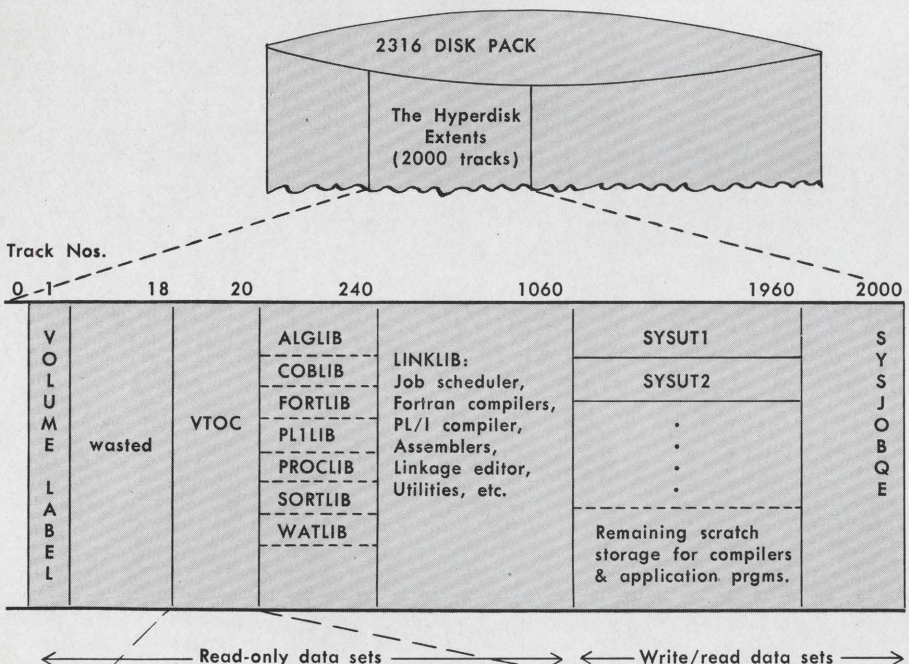
Numerous statistics are gathered

automatically by the hyperdisk to monitor its performance and to furnish details of its behavior with real job streams. These statistics are analyzed in detail in [10]. Certain statistics indicate the efficiency of its usage of LCS:

- The average TUCC job uses 13 write-read tracks. OS/360 logically allocates an average of 500 tracks per job but uses less than 3%. Thus, any job can use up to 500 tracks without exhausting its quota on hyperdisk; of these 500, approximately 200 are in LCS.
- Since average track usage is so low, few jobs—less than 2%—spill write-read tracks to disk.
- The ratio of READ to WRITE operations on hyperdisk is approximately 9:1. Since WRITE operations are exclusively to write-read tracks, and since approximately the same number of READs and WRITEs are issued to each write-read track, we conclude that the hyperdisk is much more heavily used for systems residence than for system scratch space.
- The re-reference ratio for tracks in LCS ranges between 100:1 and 200:1 over reference to tracks on

Figure 6

HYPERDISK STRUCTURE



VTOC for hyperdisk

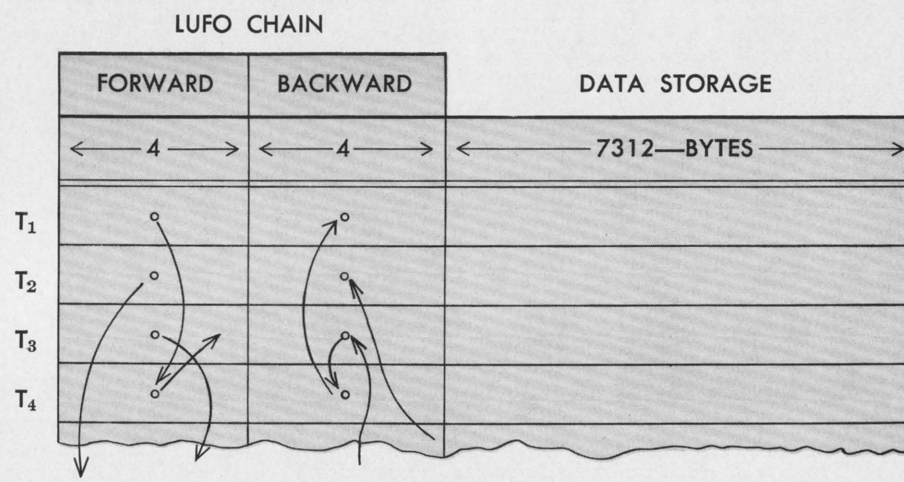
Re-loaded into LCS during IPL (as tracks 18–19)

DATA SET NAME	ATTRIBUTES	TRACK NOS.
VTOC	Pack type, size, etc.	18–19
ALGLIB	BLKSIZE = 3625, RECFM = U	20–39
COBLIB	BLKSIZE = 3625, RECFM = U	40–59
FORTLIB	BLKSIZE = 3625, RECFM = U	60–99
LINKLIB	BLKSIZE = 3625, RECFM = U	359–1059
SYSJOBQE	----------------------	1960–1999
SYSUT1	(As determined by each user)	aa–bb
SYSUT2	" " " " "	bb + 1 — cc

TRACK CONTROL TABLE

← 8 CONTROL BITS →	← 24-BIT LCS ADDRESS OR 16-BIT TRACK ADDRESS →
00000000	(LCS ADDRESS OF TRACK 1)
80000000	(REAL–2314 ADDRESS OF TRACK 2)
00000000	(LCS ADDRESS OF TRACK 3)
00000000	Ø (= TRACK NOT YET USED)

TRACK STORAGE & CHAINED LISTS



COMMENTS

(1) T_1 was accessed more recently than T_4

(2) T_4 was accessed more recently than T_3

(3) T_2 is on the chain of available tracks, i.e. all tracks not on the LUFO chain.

Figure 7

SAMPLE TRACK CONTROL TABLE AND CHAINED LISTS

Table II

SELECTED TUCC PERFORMANCE BENCHMARKS: 1966–67

		Time (in secs.) with SYSIN/SYSPRINT/SYSRES =	
Job Name	Description	Tape/tape/disk	Pseudo-reader/Pseudo-printer/hyperdisk
RUNCO	Cobol-E compile, link, and execute. 130-card source deck, full listings, 500 lines execution output	96	18
PL1CP	PL/I compilation. 1100 source cards, normal listings	40	13
BFORT	Fortran-E compile, link, and execute. 775-card source deck, full listings, 130 cards and 220 lines of execution I/O	108	32
SPLI	PL/I compile, link, and execute. Trivial source deck, trivial execution	50	9
SAMB	Assembler-F assemble, link, and execute. Trivial source deck, trivial execution.	46	10

disk. That is, of every 100–200 SEEK orders directed to the hyperdisk, only 1 requires retrieval of a track from real disk. The variability depends on (a) the frequency of switching among large compilers, (b) activity in the small-job monitor, and (c) the number of large assemblies and hyperdisk-oriented sorts.

The performance gains attributable to hyperdisk and other pseudo-devices at TUCC have been most satisfactory. A selection of benchmark jobs is described in Table II.

Mean job time at TUCC has decreased from 3 min. to 30 secs., the jobs/day average has risen from 300 to over 1500. (Frequently the system runs idle half of the midnight shift.) This throughput gain has been achieved in the face of exponentially rising demand and an ever-increasing percentage of jobs which run several minutes entirely compute-bound (typically production jobs for faculty and graduate students). Survival of TUCC would have been in serious doubt without these performance gains, attributable in large part to successful implementation of pseudo-devices.

Conclusions

Four projects for modifying OS/360 have developed similar solutions for (a) a universal SYSIN/SYSOUT interface, (b) relieving painful I/O bottlenecks in OS/360, and (c) permitting local code to control the selection, preparation, and disposition of jobs at a level above the OS/360 Job Scheduler. We recommend that OS/360 incorporate additional pseudo-device techniques beyond resident SYSJOBQE. The hyperdisk should be particularly valuable to systems with heavy loads of compilations, sorts, and small jobs. Another satisfactory solution would be for OS/360 to correct the shortcomings enumerated above, so that pseudo-device artifices would no longer be required. At this writing, the direction of development is unclear. ■

REFERENCES

1. G. H. Mealy, B. I. Witt, and W. A. Clark, "The Functional Structure of OS/360," *IBM Systems Journal*, Vol. 5, No. 1 (1966).
2. J. F. Walker, "OS/360 Throughput Problems," *Proceedings of the 6th Annual ACM Southeastern Regional Meeting* (1967).
3. J. W. Stephenson, "Use of Bulk Core to Improve System Performance," *Proceedings of the 6th Annual ACM Southeastern Regional Meeting* (1967).
4. D. N. Freeman, "A Storage Hierarchy System for Batch Processing," to appear in *Proceedings of the Spring Joint Computer Conference* (1968).
5. D. N. Freeman and Robert R. Pearson, "Efficiency vs. Responsiveness in a Computer Utility," submitted to *IFIP '68*.
6. IBM Houston GEM Office, Houston, Texas, "The OS/360 Houston Automatic Spooling Priority System," (May, 1967).
7. *"IBM System/360 Attached Support Processor (ASP): System Description,"* IBM publication H20–0223 (1966).
8. E. A. McDorman and J. M. Russell, "A performance improvement plan for the CDPSB," working paper of the Central Data Processing Service Bureau, Government of Canada, Ottawa, (April, 1967). [Description of CLASP].
9. R. Herbst, "Detailed specifications for remote terminal operating system," working paper of the Telecomputer Center, Westinghouse Electric Corp., Pittsburgh, Penna. 15221 (June, 1966)
10. D. N. Freeman, "Microanalysis of the hyperdisk," TUCC Memorandum IS-51 and Addendum to IS-51 (Dec., 1967).

market place

The Classified Ad Department of Software Age

MOONLIGHT SERVICE

NEW ORLEANS & VIC.: systems and programming for retail applications—inventory/purchasing controls; billing and integrated credit controls; payroll, including commissions; sales analysis; sales-performance analysis, etc., for single-store or multi-store operations. Completely familiar with retail applications on all tab equipment, NCR-315 and/or IBM-1401 and 360, including interface. Hour rates or contract. Write: Dept. SA-611, SOFTWARE AGE, 1020 Church St., Evanston, Ill. 60201.

NEW ENGLAND AREA: college student has limited time and unusual experience for sale. Background mainly in unique real-time applications of CDC-6600, IBM-7094, UNIVAC-1108 and assorted analog equipment to scientific experiments and control of same. Write: Dept. SA-612, SOFTWARE AGE, 1020 Church St., Evanston, Ill. 60201.

MOONLIGHTER WANTED

PHILADELPHIA AREA: manufacturer needs "moonlight" help from a programmer/analyst completely familiar with application of linear programming to least-cost product formulation working within multiple constraints. Must be able to analyze the job and write for minimum equipment. (Equipment to be installed depends upon demands of this application.) Also must be able to communicate effectively and accurately with technologically-capable but EDP-naive management and supervision. Write: Dept. SA-621, SOFTWARE AGE, 1020 Church St., Evanston, Ill. 60201.

SOFTWARE WANTED

INVOICING PACKAGE: incorporating automatic computation and credit of volume discounts and short-rates (within a time increment) and automatic punched card summarization of year-to-date data. Must provide for both direct sales and agency sales with provision for commission in the case of agency sales, cash discount on both types. Package must be runnable on (or readily adaptable to) IBM-360/20, 12K, w/2560 MFCM, two tape drives and 2203 printer with single-feed carriage. Write: Dept. SA-601, SOFTWARE AGE, 1020 Church St., Evanston, Ill. 60201.

SOFTWARE OFFERED

CIRCULATION & MAILING LISTS: complete tested (working) software for card or mag-tap conversion, maintenance and label printing on 360/20. Card version require 20/20, 8K with 2560 and 2203 with 120-c line. Tape version require 20/20, 12K with 2203 or 1403, 2560 and two tape drives. Produces 4-up E-W Cheshire labels @ 15,000/hr with 2203; up to four lines of address, with or w/o additional matchcode line, using unusual 1-cd. format providing 4 cols. (up to 8 items by using zone and digit pchs separately) of classification, demographic and/or expire data. Programs identify "suspect" duplications (i.e. near but not total duplication) for operator decision. Automatic zip markings to speed mailing operations. State selection, every Nth name and other selective label print programs also available. Write or phone H. L. Rothra, PRESS—TECH, Inc., 1020 Church St., Evanston, Ill. 60201. (312) 869—1244.

TIME WANTED

TAPE/TAPE SORTING: one-time job involving a list of 14-million-plus name/address records currently being read into tape from 20-odd card and tape files, some in geo-alpha order and some in zip sequence so must be assumed to be random order as received on tape. Must be sorted into a single zip-code and match-code sequence involving 17 positions in three fields. Educated estimate on time and price invited. Write: Dept. SA-631, SOFTWARE AGE, 1020 Church St., Evanston, Ill. 60201.

TIME AVAILABLE

CHICAGO N. SUBURBAN: 360/20, 12K, two tapes, 2560 MFCM, 2203 printer with 52-c bar, 120-c line, single-feed carriage. Up to 80 hrs/mo. available on day shift and/or 5 PM–8 AM every day. Write D. J. Fawcett, SCIENCE/SYSTEMS, P. O. Box 1176, Evanston, Ill. 60204 or Phone (312) 273–4818.

COMPUTER PROGRAMMING SERVICE: for system analysis, system simulation, component design and analysis. Write: Dept. 641, SOFTWARE AGE, 1020 Church St., Evanston, Ill. 60201.

EMPLOYMENT AGENCIES

SOUTHEASTERN JOB OPPORTUNITIES

Manager for textile systems department to $20,000. Other jobs in industry, finance and consulting for analysts, programmers at all levels. All fees paid. Dick Terry, DP Specialist, Continental Consultants, 1316 Main St., Columbia, S. C.

EMPLOYMENT SERVICES

OUTPUT-PROGRAMMER TRAINEES
System/360 DOS/TOS ALC/COBOL
MAJOR VENDORS have been satisfied with our product. Perhaps your expectations are equally critical. If you're interested in qualified trainees, with NO PLACEMENT FEE involved, contact Mr. Mengel.
COMPUTER TRAINING CENTRE
(Div. of Systems Associates, Incorporated)
356 Penn Street Reading, Pa. 19602

FOR SALE

BUY AND SELL OR LEASE: used computers through DA-PEX Computer Brokers, 334 Francis Building, Louisville, Ky. Phones 451–7457, 585–5454.

COMPUTER BUG PINS: Tie-Tac, Charms, Lapel, Girls pins. Bronze $3.00, Gold Plate $3.25, Sterling Silver $3.50. Legend, Head-Disc storage, eyes-magnetic core, body-microelectronic chip, neck and legs-paper tape, arms and hands-magnetic tape, feet-punch cards. Computer Bug, Box 20268, Houston, Texas 77025.

FOR SALE

IBM Model 704 computer with 8 Model 727-C tape drives. Write ADVANCE COMPUTER SYSTEMS, 2032 Cotner, Los Angeles, California 90025 or call (213) 270–3106.

market place

CLASSIFIED ADS in software age

where you can reach **91,000** programmers, mathematicians, analysts and EDP managers.

CLASSIFIED ADVERTISING

Non-Display Classified: For firms or individuals offering commercial products or services, 75¢ per word (including name and address). Minimum order $7.50. For Blind Ad Service, an additional $10.00. For "Positions Wanted" Ads, 55¢ per word (including name and address). No minimum. **Payment must accompany copy** except when ads are placed by accredited advertising agencies. Frequency discounts: 5% for 6 months; 10% for 12 months paid in advance.

Display Classified: One inch by one column, $70.00. Column width 2¼". Photographs accepted for an additional $20.00. Advertiser to supply all photo, art, cuts, or camera ready copy.

General Information: One inch display Help Wanted and Employment Services ads **will be** accepted in the classified section. Employment ads ⅛ page or more will appear run of book, will be keyed to the resume form in back of publication, and will qualify to free daily resume service.

Closing Date: 1st of preceding month (for example, May issue closes April 1st).

Send order and remittance to: Classified Dept., Software Age, 1020 Church Street, Evanston, Illinois 60201.

software age
1020 CHURCH STREET • EVANSTON, ILLINOIS • 60201

CLASSIFIED ADVERTISING ORDER FORM

Please refer to the above information for complete data concerning terms, frequency discounts, closing dates, etc. Cash with order.

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25
26	27	28	29	30
31	32	33	34	35

______Words { @ 55¢ (Reader Rate) / @ 75¢ (Commercial Rate) } = $______

______Include Photograph @ $20.00 additional (**Display Ads Only**)

☐ Run Ad Blind $10.00 Insert______time(s) Total Enclosed $______

NAME______

ADDRESS______

CITY______ STATE______ ZIP______

Signature______

WORD COUNT: Include name and address. Name of city (Des Moines) or of state (New York) counts as one word each. Zone or Zip Code numbers not counted. (Publisher reserves right to omit Zip Code if space does not permit.) Count each abbreviation, initial, single figure or group of figures or letters as a word. Symbols such as 35mm, COD, PO, AC, etc., count as one word. Hyphenated words count as two words.

SA-6

new products

The Chrono-log Programmable **Clock/Calendar** is now available for use on System/360 Computers using Series 2400 Tape Drives, Mod 4, 5 and 6 with Phase Encoding recording (P. E.) as well as non-return to zero (NRZI).

The Clock/Calendar, which is designed for plug-in installation, is said to require no computer wiring changes and to use existing instructions. When interrogated, the Clock/Calendar reads the date in months and days, and the time of day to the nearest 1/60th second into computer memory. This allows the monitor program to perform billing and accounting functions for each job automatically without need for operator intervention.

The Clock/Calendar keeps time independently of the computer and corrects automatically for 30 and 31 day months and Leap Year.

For more information, circle No. 80 on the Reader Service Card

* * *

The **Data Communications Systems' Series Acoustical Couplers** are designed to be connected through a standard telephone handset to a terminal such as Teletype or equivalent over telephone lines. Its primary applications include on-line time sharing, remote inquiry and general data transmission. The couplers have packaging design and unique filter circuits to give better noise rejection. It's size is only 10 x 6 x 6 plus handle and weighs approximately 6 pounds. There is space for additional modular circuitry to provide, serial to parallel, parallel to serial, code modification circuits and more. Provision for monitoring is accomplished with an assurance speaker with volume control. It is switch controllable between EIA (RS232B) and Teletype interface. Half or full duplex operation is standard. It is a completely self-contained, portable unit and requires 110 volts AC. The upright/inverted capability with switch is optional.

For more information, circle No. 79 on the Reader Service Card

* * *

Business Supplies Corporation of America, is now marketing a heavy duty **Disk Pack Storage Dray** featuring unusual ease of access. Fifteen Disk Packs are safely stored in individual swing out compartments each separately numbered with space for identifying the contents. Opening the compartment automatically puts the Disk Pack within easy reach.

The specially designed metal-clad outside panels and tubular steel frame, combine strength with light weight. The Dray rolls on oversized wheels and is designed to harmonize with computer room equipment. It can be ordered with the BSC Micro-Tolerance Disk Packs or for present packs. Size is approximately 22" deep, 42" high and 55" long.

For more information, circle No. 78 on the Reader Service Card

* * *

The physical and performance characteristics of a new **digital computer tape** has been developed for use in extreme environments by the U.S. Magnetic Tape Company, Huntley, Ill., a subsidiary of Wabash Magnetics, Inc.

The polyimide-based X^n Tape handles up to 1600 BPI in ambient temperatures ranging from —55F to 180F, far beyond the extremes that would embrittle or melt any tape heretofore available. X^n Tape will also withstand 90% relative humidity.

An outgrowth of U.S. Tape's Duramil 7 development program (high-performance tape for conventional computer application), X^n Tape brings new feasibility and practicality to the use of computers in polar and tropical regions, in outer space and at remote industrial and manufacturing sites.

For more information, circle No. 77 on the Reader Service Card

* * *

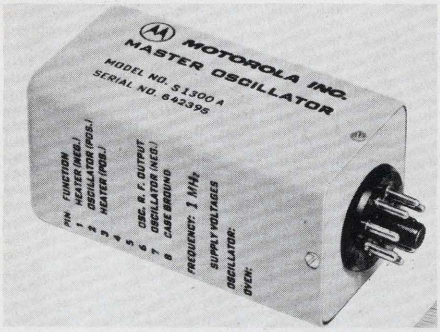
A new **subminiature axial radio-frequency shielded inductor** for density circuits has been introduced by Lenox–Fugle Electronics, Inc., of Watchung, N.J. The new NANO–RED inductor offers a ¼ watt shielded inductor for density circuits that measures just 1/10 of an inch in diameter by ¼ of an inch long, and weighs no more than 0.28 gram. In volume, the new NANO–RED envelope, of non-flammable epoxy, is a scant 0.002 cubic inches.

Designed to meet the reliability standards of MIL-C-15305C, to comply with the need of design engineers for large inductance in a small envelope for density circuit applications, the NANO–RED inductor is available in 49 stock values from 0.10 uh to 1,000 uh ± 10%.

For more information, circle No. 76 on the Reader Service Card

* * *

The S1300A, an advanced design, **high-stability plug-in oscillator,** has been announced by the Precision Instrument Department of Motorola, Inc.



The oscillator features silicon solid-state circuitry, proportional oven control, cold-weld construction and high reliability.

On a 10 day average, aging rates for standard models are 5×10^{-10} per day. Increased accuracy is provided by temperature stabilities as low as $1 \times 10^{-10}/C°$. For accurate measurements and control, standard short-term stability is better than 1×10^{-10} rms/sec.

Normal weight is 20 ounces and temperature range is from: —20°C to +55°C. Normal output voltage: 1 volt rms into 500 ohm load and input voltage: 28 vdc ± 2v neg. or pos. ground.

For more information, circle No. 75 on the Reader Service Card

* * *

The FO-200 Series of **Voltage Followers** is an operational amplifier type device used as a convenient means for signal isolation and introduced by the GPS Instrument Co., Inc.

They will reproduce an input signal and all models have been optimized to have extremely high input impedance. In this way they can accept signals from a high impedance source or sensor without loading or affecting the source or sensor. The Voltage Follower output is said to have a low impedance and can drive as much as 20 milliamps; consequently, it can be used directly to drive a meter movement or other monitoring equipment.

The Voltage Followers feature a carefully matched pair of field effect transistors (FET) at the input to give the very high impedances required.

For more infomation, circle No. 74 on the Reader Service Card

* * *

MONITOR Systems, Inc. announced the availability of information relative to assembling a binary-to-BCD converter by using MONITOR'S MONILOGIC (TM) **integrated circuit cards.**

The informative sheet describes a simple method that may be used to convert a 12-bit binary number to the binary-coded-decimal equivalent within a 4 microsecond time span. The technique suggested is applicable to binary words of any length.

For more information, circle No. 73 on the Reader Service Card

* * *

Non-Linear Systems offers a **digital ohmmeter** which automatically measures resistances between 100 microohms and 120 megohms.

Measurements are presented on a 6 window display when using the NLS Model X-1 digital ohmmeter/voltmeter. Eight full scale resistance ranges of 12/120 Ω 1.2/12/120 KΩ, 1.2/12 and 120 megohms and five full scale voltage ranges of 120 mv and 1.2/12/120 and 1000 volts in one 5¼" high rack mountable package provide both physical and measurement versatility.

Automatic ranging and polarity are standard. Resistance measurements are 4-wire, guarded. Box-in-box construction provides 120 db CMR. A standard 3-pole active filter allows accurate resistance measurements even in the presence of large amounts of degrading noise.

The X-1 wide range ohmmeter has mid-scale specifications of: ±(0.001% of full scale + 0.01% of reading); measurement speed of 10 readings per second; and passes a current through the resistor under test of one milliamp or less.

For more information, circle No. 72 on the Reader Service Card

Houston Instrument, Division of Bausch & Lomb, Inc. announces a new remote **digital plotting system** for use in computer time sharing systems.

The CØMPLØT$_{tm}$ Plotter/Teletypewriter Controller PTC-3 will serve as the interface between a Dataphone Model 103A2, or equivalent, or an appropriate acoustical coupler and a Teletypewriter Model 33 or 35 and the CØMPLØT$_{tm}$ Digital Plotter DP-1.

Information from the computer center is transmitted over standard voice grade transmission line into the Dataphone or acoustical coupler; it is then fed to the CØMPLØT$_{tm}$ PTC-3. The PTC-3 monitors the incoming data and transmits it to either the Teletypewriter or the CØMPLØT$_{tm}$ DP-1. If the data is for Teletypewriter presentation, the PTC-3 transmits directly to it with no change in the information. If, however, the data is to be plotted, it is coded by the PTC-3 to the appropriate signals and then routed to the plotter.

The Teletypewriter serves as the input/output keyboard device that allows communication with the computer. Output that lends to graphic display is plotted on the CØMPLØT$_{tm}$ DP-1 Plotter, thus giving the system the additional capability of a graphic display of data. The DP-1 Plotter is a digital incremental plotter capable of operating at a maximum rate of 300 increments/second.

For more information, circle No. 71 on the Reader Service Card

A new software package, **AUTODIAGRAMMER II**, has been developed by Aries Corporation to automatically generate a full set of program documentation for rapid and accurate checkout of COBOL programs **before** they are compiled.

AUTODIAGRAMMER II enables the programmer to actually see, in final form, his flow chart and data printout with only one computer pass of his source deck.

Designed primarily for use with IBM System 360 computers, it will accept a COBOL source program before it has been compiled and will automatically generate: a detailed flow chart, a higher level logic chart, an input/output diagram, formats of all defined records, sample formats of any reports which the program will be printing, and a series of diagnostic tables and cross reference lists.

The program is also readily adaptable for use with UNIVAC 9300 systems, and BAL and FORTRAN languages are available as options.

For more information, circle No. 70 on the Reader Service Card

* * *

CL/1 is **a complete information processing and reporting system** designed for a basic configuration consisting of an IBM System 360 computer with 64K capacity, two disk units and four tape or disk drives. It may be operated in either a Disk Operating System (DOS) or Operating System (OS) environment, according to the Computer Sciences Corp.

A key part of CL/1 is a report generator which will either produce reports in the exact formats specified by users or, in the absence of specific instructions, establish its own formats.

The output of CL/1 includes 56 types of reports, 42 of which are optional. The list of required reports includes statements, daily transaction journals, notes with zero principal and active interest listings, changes to interest accruals, daily accounting summaries and general ledger transactions.

Among CL/1's special features are capabilities for handling multi-bank processing; division and branch accounting; indirect liabilities; account relationships; collateral, maintained and sold; and participation, bought and sold.

For more information, circle No. 69 on the Reader Service Card

* * *

A **1/8-in. diameter neon lamp**—the smallest ever offered—has been announced by Los Angeles Miniature Products, a subsidiary of Oak Electro/Netics Corp.

The subminiature T-1 lamp, which operates at any current above 90 volts AC, was designed for indicator applications such as voltage regulators, counting devices and other electronics and aircraft uses. It is available tipless or lensed, with or without base.

Advantages of the high-brightness type neon lamp include end-on viewing, elimination of stepdown transformer, increased shock and vibration resistance and weight and space savings.

It allows four times the density in console applications and, therefore, a computer console may be reduced to 1/4 size, with resulting easing or monitoring and enhanced esthetics.

For more information, circle No. 68 on the Reader Service Card

* * *

Tape-Stor Div., Helps Company has produced a new **Tape Winder** for winding Numerical Control tapes. This tape winder has a roll capacity of up to 3½ inches in diameter. Back-up plates for holding the tape in position when winding, are furnished—these bask-up plates also aid in the easy removal of the roll of tape from the winder. The 7/8 diameter hub has a threading slot. The tape will not crimp or wind too tight. This 7 inch high winder will fold down to 7" x 3" x 2¾" for easy storage.

For more information, circle No. 67 on the Reader Service Card

* * *

An expanded line of **regulated d-c power supplies** is now available from General Electric for use in improving the performance of electrical and electronic equipment. Now included are a total of thirty 50- and 60-hertz models with output voltages ranging from 10 to 200 volts d-c.

Output voltage will not vary more than ± one percent for a change in input voltage over the rated range of 97 to 130 volts a-c; and a change of one percent in frequency will only change the output voltage level approximately 1½ percent. The full-load efficiency for these General Electric power supplies ranges from 70 to 85 percent for normal output voltage levels.

General Electric d-c power supplies can be installed with either polarity. Output voltage riple level is one percent or less (RMS).

For more information, circle No. 66 on the Reader Service Card

* * *

Atlantic Software Inc. announces the immediate availability of a new **proprietary program, DOC-U-MENT. DOC-U-MENT** is a complete file management system specifically tailored to handle the unique requirements of documentation in a data processing installation.

It performs the following functions:

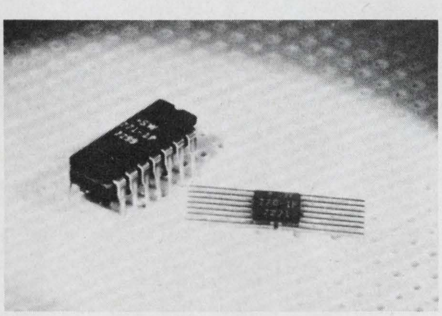
a. file maintenance of the documentation.
b. retrieval of any desired section of documentation.
c. display of the documentation in one of several appropriate formats—narrative text, card image, or flow chart.

Input to the **DOC-U-MENT** system is in the form of simplified shorthand statements. Because of this simplified input, key technical personnel can be freed from the laborious documentation job.

It will handle all forms of documentation—procedures, systems charts, files, program specifications, flow charts, object decks, and source decks. Any user can apply it to his current documentation procedures or use it as a tool to develop a new documentation standards system.

For more information, circle No. 65 on the Reader Service Card

* * *

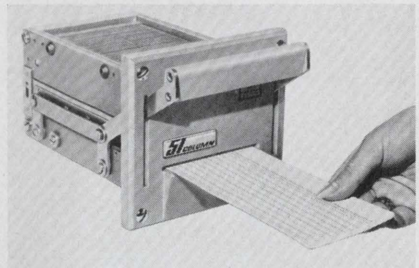


Ten new **integrated circuits** are now available from Stewart–Warner Microcircuits, Inc.—each one of which performs expanded functions that previously could be attained only by the improvised interconnection of two or more units of the DTL 930 series.

The new devices consist of: a 10-input complementary gate (SW770 & SW771), a triple R-S flip-flop (SW772 & SW773), a triple 3-input AND gate (SW774 & SW775), a dual AND/OR gate (SW776 & SW777), and a dual 4-input complementary gate (SW778 & SW779). Even-numbered units are with 6K pull-up resistors for lower power consumption, and odd-numbered units are with 2K resistors for faster rise times.

For more information, circle No. 64 on the Reader Service Card

* * *



A new 51 x 12 **tab reader** that supplies a full 612 bits of information from a standard 80 x 12 IBM punched card has been designed for applications in data acquisition, process control and production control by Sealectro Corporation.

The new Sealectrocard tab reader features an advanced contact design that eliminates thru-card contacts and prevents lint and dirt from impairing reliability. This design also eliminates varying contact resistance due to accumulation of contaminants on usual PC switch elements under a punched card. Units can be supplied with a number of different terminations to fit specific customer requirements.

For more information, circle No. 63 on the Reader Service Card

* * *

A new **automatic contact and relay testing system** which is now available from Advanced Technology & Systems Corporation, provides an easy and convenient way of determining contact resistance for all types of plug-in relays with any contact arrangement. A built in square wave generator permits the display of the dynamic test function when used with type PRO oscilloscope.

Test parameters and sequence can be manually selected in the model ARP or pre-programmed by means of a program plug-in in the model ARP-E. Both instruments feature full flexibility of voltage selections, sequence of test and can be stopped at any point in the test sequence to permit the observation of any test step.

For more information, circle No. 62 on the Reader Service Card

* * *

Digital Devices has announced its type **203 delay line,** a low cost memory suited for commercial applications requiring storage of 1024 bits or less.

Available at data rates up to 2 MHZ and with access times of less than 1 ms, the memory is designed to perform in standard commercial environments. Input drive is 40 ma, producing a 10 mv output.

The memory has found application in various small scale commercial digital equipments.

For more information, circle No. 61 on the Reader Service Card

MULTIDIMENSIONAL MODELS
(Continued from page 6)

time, investment, material cost, expense etc.

The problem is then analyzed. A plane functional flow diagram is drawn up for definition of proper sequential relationship of functions. After the check of the flow diagram the three dimensional block diagram can be drawn. The flow diagram of functions is given in Fig. 2 for a fictitious process of golf ball development and marketing.

This diagram gives the sequential relationships of functions thus it represents a process of operations.

Fig. 3 represents the same process but the additional "dimensions" of business shown makes this no longer an operational flow diagram but it becomes a model in many dimensions of the business.

The proper selection of coordinate groups is very important and that can make this method very powerful.

The current numerical values of coordinates at the corners of blocks, together with the local component values of coordinates in the blocks can reveal most of the critical factors in the business.

The name of the means to perform the functions retains the physical feel for the model that allows to change it to an optimum configuration, through a thinking evaluating process, played with the model.

The model can serve when it is finished for writing computer programs for various purposes if equations representing the blocks can be derived. The purpose of programs may be production control for multiple project system, scheduling and other.

Summary

A method of solving complex systems optimization problem had been presented.

The multi-dimensional model in Reimann's N dimensional space retains the sequential relationship of the large number of system components characterized by a large number of coordinates that also retain their physical meaning and numeric value. The real like form of model allows easy multiple variable optimization. ■

software age

CONFIDENTIAL INQUIRY

Your original copy of this form will be retained at the offices of SOFTWARE AGE and will be used for no other purpose than to notify the specific firms which you have checked (on the reverse side) of your interest.

THIS INQUIRY IS IN DIRECT RESPONSE TO YOUR ADVERTISEMENT IN SOFTWARE AGE MAGAZINE

TYPE OR PRINT CLEARLY FOR PHOTO REPRODUCTION

JOB DESIRED: ______

List computer hardware knowledge (names of systems, tape, disk, terminals, etc.): ______

Programming specialties and years of experience (commercial, scientific, theoretical, experimental, analog, etc.): ______

Systems programming on which you have had development experience (compilers, assemblers, executives, monitors, O.S., etc. Indicate for what computer): ______

Programming languages used and extent of experience (COBOL, FORTRAN, etc.): ______

Applications programmed (aerospace, banking, insurance, math subroutines, compilers, etc.): ______

Systems analysis experience (card design, flow charting, operation analysis, etc.): ______

EDP management experience (include years and number of people reporting to you): ______

SALARY: ______ (current) ______ (desired) **DATE OF AVAILABILITY:** ______

EDUCATION: Indicate major as well as degree unless self-explanatory.

Degrees			
Years			
Schools			

EMPLOYMENT: Indicate present employment and previous jobs below.

Employer			
City			
Years	to	to	to
Title or Function			

Name ______ Age ______

Home Address ______ Home Phone ______

______ (city) (state) (ZIP code) U.S. Citizen? ______

Security Clearance ______ Location Preference ______

Marital Status ______

Military Status ______

BE SURE YOU HAVE CHECKED ON REVERSE SIDE THE COMPANIES YOU WANT TO SEE THIS INQUIRY. PUT FORM IN STAMPED ENVELOPE AND MAIL TO:

software age
MAGAZINE
1020 CHURCH ST., EVANSTON, ILL. 60201

AFTER AUG. 10, THIS FORM WILL NOT BE PROCESSED

WRITE ADVERTISERS DIRECTLY OR WRITE US FOR CURRENT S/A ISSUE

check your interests here

Fill in the confidential inquiry form on the other side of this sheet. This form provides all the information advertisers require to screen applicants. If further information is desired, you will hear from the advertiser direct. Then, circle below the numbers of those companies to which you want copies of your form sent. Mail to ENGINEERING OPPORTUNITIES, 1020 Church Street, Evanston, Illinois 60201. **(Please do not send us your own resume. We will only process this form. A new form must be filled out for each issue in which you are answering ads.)**

CIRCLE BELOW:		Page
1.	Abbott Laboratories	31
2.	Automatic Electric Laboratories	4
3.	Army & Airforce Exchange	35
4.	Ebasco Services, Inc.	3
5.	Gerber Scientific Instrument Company	38
6.	Honeywell, Inc., Electronic Data Processing	3rd Cover
7.	IBM Corp.	11
8.	ITT Federal Electric Corp.	15
9.	Jet Propulsion Laboratory	16
10.	Lawrence Radiation Laboratory	42
11.	Lockheed Electronics Co.	28
12.	McDonnell Douglas Corp.	17
13.	Montgomery Ward Data Center	44
14.	National Cash Register Co.	27
15.	Northern Natural Gas Co.	12
16.	Parke–Davis & Co.	32
17.	Pratt & Whitney Aircraft, Div. of UAC	21
18.	Radiation, Inc.	36
19.	RCA Information Systems Div.	5
20.	Sikorsky Aircraft	7
21.	Statistical Tabulating Corp.	41
22.	Stearns–Roger Corp.	43

CIRCLE BELOW:		Page
23.	Univac Data Processing Div.	24
24.	Univac Federal Systems Div.	13
25.	Vitro Laboratories	37
26.	Xerox Corp.	2nd Cover

EMPLOYMENT AND SEARCH AGENCIES

		Page
27.	Cadillac Associates, Inc.	26
28.	Callahan Center for Computer Personnel	38
29.	Computer Personnel Agency, Inc.	40
30.	Computer Professions, Inc.	20
31.	Fox–Morris Associates	20
32.	Robert Half Personnel Agencies	30
33.	LaSalle Associates	35
34.	Lawrence Personnel	40
35.	Management Scientists, Inc.	26
36.	Professional Career Centers	30
37.	RSVP Services	41
38.	Search, Inc.	23
39.	Source EDP	28
40.	Systemat®	20

PRODUCTS AND SERVICES

	Page
Business Supplies Corp. of America	29
Hexco, Inc.	43
Science/Systems	18, 19
Wilson Jones Co.	4th Cover

☐ I do not now receive S/A. Please enter my FREE subscription.

....

Name   Address

....

My Specialty

....

Prime Experience in What Industry

☐ Technical Degree   ☐ Non-Technical Degree   ☐ No Degree

....   ☐ I Have Analog/Hybrid Experience

Year Born

software age

1020 Church Street
Evanston, Ill., 60201