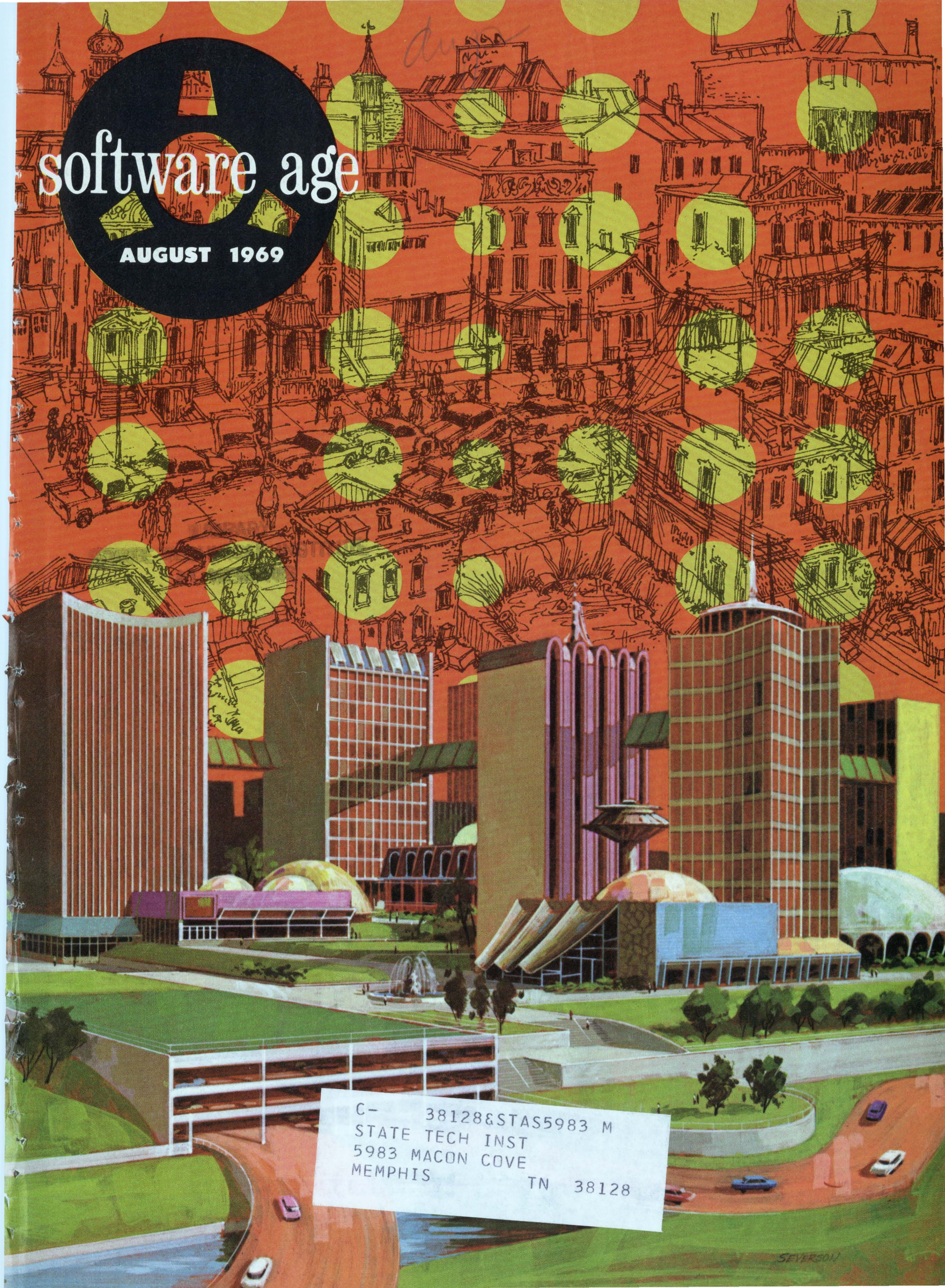
software age

AUGUST 1969



software age

AUGUST, 1969 **Vol. 3–No. 8**

Publisher .... David W. French
Editor .... John F. Sikora
Managing Editor .... Jean S. Logan
Assistant Editor .... Elspeth Daly
Traffic Manager .... Betsy Pavkovich
Circulation .... Ann Vickers
Business .... Stephenie Jensen
Resume Control .... Anne Wildeman

CIRCULATION OF THIS ISSUE
MORE THAN 130,000
BPA Audit Applied For

COVER: David Severson's August cover depicts the image and role of computers in converting our urban complexes from polluted, over-populated, traffic-congested areas into livable, viable and stimulating communities.

SOFTWARE AGE

2211 FORDEM AVENUE, MADISON, WISCONSIN 53701
PHONE: (608) 249–0128 OR (800) 356–8141

David W. French

New York, N. Y.: (212) 697–5356
Los Angeles, Calif.: Richard Faust, 9000 S. Sepulveda Blvd. (213) 776–0100
San Francisco, Calif.: Ernesto Montano, 595 Buckingham Way, Suite 226 (415) 731–8240
Philadelphia, Pa.: Richard D. Clemmer, 27 Acoma Lane, Collegeville, Pa. (215) 489–9141
Boston, Mass.: (617) 542–1466

SOFTWARE AGE is published monthly by

press tech inc.

2211 Fordem Avenue, Madison, Wisconsin 53701

Subscription free to qualified readers. Others, 10/yr. Individual copies, $1. Foreign subscriptions, $15/yr.

For change of address please return your address label from the magazine.

Controlled circulation postage paid at Madison, Wisconsin.

financial currents

Honeywell's new computer equipment manufacturing plant at Billerca, Massachusetts, is aiming for a $30 million production per year rate by end of 1969, which would help the company toward its 1975 target of $1 billion in overall computer revenues.

The plant, dedicated June 19 and situated about 30 miles northwest of Boston, would in six years contain a manufacturing, research and engineering complex employing nearly 3,500 people. It would be the largest facility in Honeywell's EDP Division.

Should the EDP Division carry its load in achieving the billion-dollar goal, its revenue will have to triple in five years, its sales force double in three, its engineering force double in four, its programmers double in three and its field service people will have to triple in five years.

Besides the new plant, which is headquarters for the division's Peripheral Device Operations, the EDP division leases or owns space in six other Massachusetts locations.

* * *

Automation will reach into Wall Street's "cage"—the area where large numbers of stock certificates are received and dispatched—when a securities clearance system being phased into operation becomes fully installed this fall at Goodbody and Company.

Aimed at eliminating clerical logjams plaguing Wall Street during periods of heavy trading, the system—AutoCage—will also be offered to other brokerage firms in the hope of reducing "fails"—the inability to deliver certificates on time. A recently formed company, Goodbody Systems, Inc., will market AutoCage, which utilizes a battery of advanced Bunker Ramo video terminals linked into the firm's RCA third-generation computer. Other securities firms, including the smaller operations, can use the system either in conjunction with their own computers or through the shared-computer concept.

* * *

Data Transformation Corporation, an all-black founded firm which specializes in urban data systems and centers, education and aerospace computer consulting, plans to launch an intensive program to computer users with special emphasis on sociological applications. The national computer software consulting firm is headquartered in New York . . . A new data and computer systems firm, Daconics, Inc., has been formed in Sunnyvale, California, to specialize in providing turnkey systems to its customers. Goal of the new firm: "The synthesis and amalgamation of all the techniques necessary to make small and medium size computers real working tools where they are first installed." . . . Computer Input Services, Inc., a wholly owned subsidiary of Modern Data Techniques, Inc., has been established in Upper Darby, Pennsylvania, to provide computer and computer-related services in the Philadelphia area.

A new computer complex which will link the business community of North Jersey with a network of computer facilities around the nation has been opened in Paramus. Telecommunications will link data processing equipment at Creative Computer Corporation, subsidiary of Creative Logic, to 1108 Univac computer facilities in Mineola, Long Island; Phoenix; Chicago; and San Francisco. . . The remote terminal market is the target of Remcon Systems, Inc., of Garland, Texas, a new generation computer equipment company. Expecting to be in production by first quarter 1970 with a family of remote batch terminal systems priced below that of the competition, Remcon says it is preparing "to challenge the top-ranking firms in the computer equipment business for a share of the peripheral business."

The first Sigma 7 Computer in the Delaware Valley area was unveiled in Open House ceremonies at Comserv, new computer utility in Philadelphia, which claims "a fresh and exciting approach to the exploding field of computer utility service," and "a new philosophy of information." . . .

Programmers/Analysts:

Give IBM a hand and we'll try to make it a free one

If predictable thinking were what we wanted, we'd just program it. But what we're looking for are imagination and ingenuity. If you have these qualities, and you qualify for a job with IBM, you can be sure you'll get to use them.

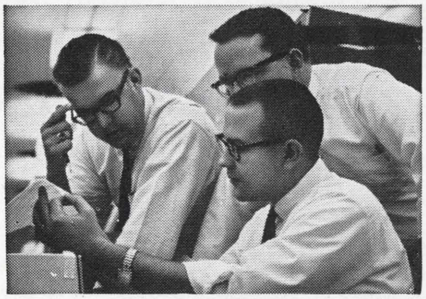
Immediate openings

There are openings for Programmers/Analysts at IBM Endicott, N.Y.

You'd design, program, test, and install application programs for our new Manufacturing Information Systems. These systems will aid manufacturing and production planning functions at all IBM facilities.

You will be involved with real-time and on-line processing applications, using IBM System/360 programming.

A Bachelor's degree and at least a year's experience are required. You should have a knowledge of basic assembler language and large-scale computer concepts.



IBM programmers and systems analysts work in small teams on an IBM System/360.

Grow with IBM

Today's major growth industry is information handling and control. And IBM is a leader in that field.

Our job is to help IBM improve its manufacturing and production planning functions by applying information handling and control techniques.

If you're a problem-solver who wants a personal sense of achievement for your hard work in a growth company, consider IBM.

Call or write

Dom Santoni is the man to talk to. Call him collect at (607) 755-2855. Or write him at IBM Corp., Dept. CH1006, 1701 North Street, Endicott, N.Y. 13760.

An Equal Opportunity Employer

IBM®

conference countdown

AUGUST

19–22 Western Electronic Show and Convention (WESCON), San Francisco, Calif. Contact: WESCON, 3600 Wilshire Blvd., Los Angeles, Calif. 90005.

25–29 Datafair 69, Manchester, England. Contact: The British Computer Society, 21 Lamb's Conduit St., London W.C. 1, England.

26–28 ACM National Conference & Exposition, San Francisco, Calif., Contact: ACM, 1133 Avenue of the Americas, New York, N. Y. 10036.

SEPTEMBER

8– 9 Society for Management Information Systems First Annual Meeting, Minneapolis, Minn. Contact: G. W. Dickson, Management Information Research Center, School of Business Administration, University of Minnesota, Minneapolis, Minn. 55455.

8–10 American Institute of Aeronautics and Astronautics Computer Systems Committee Conference, Los Angeles, Calif. Contact: Dr. Eugene Levin, Aerospace Corp. P.O. Box 95085, Los Angeles, Calif. 90045.

8–12 International Symposium on Man–Machine Systems, Cambridge, England. Contact: Robert McLane, G-MMS Meetings Chairman, Honeywell Inc., 2345 Walnut St., St. Paul, Minn. 55113.

15–17 Joint Conference on Programming Languages for Numerically Controlled Machine Tools, Rome, Italy. Contact: E. L. Harder, R & D Center, Westinghouse Electric Corp., Beulah Rd., Pittsburg, Penn. 15235.

22–23 Systems Documentation Workshop, Chicago, Ill. Contact: Tony Raitz, Systemation, Inc., Box 730, Colorado Springs, Colo. 80901.

28–Oct. 1 International Systems Meeting of the Association for Systems Management, New York, N. Y. Contact: Richard Irwin, Association for Systems Management, 24587 Bagley Rd., Cleveland, Ohio 44138.

OCTOBER

1– 5 American Society for Information Science Annual Meeting, San Francisco, Calif. Contact: ASIS, 2011 Eye St., N.W., Washington, D.C. 20006.

2– 3 Second Advanced EDP Audit and Control Conference, New York, N. Y. Contact: Harold Weiss, Automation Training Center, 1930 Isaac Newton Sq. E., Reston, Virginia 22010.

9–11 Developing Professional Manpower through Action, conference of DPMA Division 3, Little Rock, Arkansas. Contact: Robert Redus, 6901 Murray St., Little Rock, Arkansas.

19–22 1969 National Conference of the American Records Management Association, St. Louis, Mo. Contact: ARMA National Headquarters, 24 N. Wabash Ave., Chicago, Ill. 60602.

26–30 Joint Conference on Mathematical and Computer Aids to Design, Anaheim, Calif. Contact: E. G. Kimme, Collins Radio Company, 19700 Jamboree Road, Newport Beach, Calif. 92663.

27–29 Data Processing Supplies Association's Fall Meeting, New York, N. Y. Contact: Data Processing Supplies Association, 1116 Summer Street, Stamford, Conn. 06905.

27–31 BEMA Annual Exposition & Conference, New York, N.Y. Contact: Paul Notari, BEMA, 235 E. 42nd St., New York, N.Y. 10017.

28–30 24th Annual Instrument Society of America (ISA) Conference & Exhibit. Contact: Ray Cooley & Associates, Inc., 4848 Guiton St., Houston, Texas 77027.

A PHILOSOPHY FOR DIGITAL SIGNAL PROCESSORS

L. D. Ingwersen

Introduction

In recent years, a number of methods for shortening computer time used in digital processing of time series data have been presented in apropos journals and trade magazines. These methods are important to two groups of computer users. The first computer user group processes signals on a time-shared company device or at a data center. He wants to minimize the amount of time which appears on his bill.

The second group of computer users processes large amounts of time series data. This user probably has a computer system[s] which is specially configured to enhance the throughput of time series data analysis problems. As the user's quantity of time series data to be processed increased, many of these users purchased *peripheral* equipment designed to do specific steps of the signal processing arithmetic at high speeds.

The basic operation to be performed on time series data is convolution. So, many computer manufacturers offered high speed digital convolvers. Customers ordered faster Convolvers as data quantities increased further. These first generation convolvers were hardwired to do convolution operations. They were unable to calculate, with sufficient speed, the more recent processing methods which reduce the total calculation steps for given length or types of input data and for given end results.

The objective of this article is to introduce a signal processing philosophy which will allow the above multiplicity of usage at high speeds. The philosophy is to be implemented in a *peripheral* computation device.[1]

Capabilities

To start development on a signal processing algorithm module one would ask himself to list the capabilities needed by the signal processing industry. The algorithm module should be capable of convolution which is the backbone of filtering and correlation techniques. The device should have high speed recursive (or feedback) capabilities to implement this class of high speed filtering techniques. Further, the module should be capable of Fast Fourier Transforms [FFT]. These basic capabilities would give the user a device capable of computing via most methods being used now.

Because new techniques of some merit appear quite often, our machine should be structured such that new techniques can be added quite easily.

The device should be capable of Matrix and Vector algebra and running totals. This capability can lead to calculations such as sum of square in addition to Matrix multiplication and addition.

Speed

All of the above capabilities can be obtained by purchasing a general purpose computer. That places us in the first user grouping. Our *peripheral* signal processing algorithm module should be capable of very high speed arithmetic operation. A reasonable measure of the convolvers speed is its effective multiply-add time. The philosophy to be described can also be measured in this manner.

The signal processing algorithm module should be capable of a very short optimum effective multiply-add time [125 nanoseconds, for example]. To further enhance our speed, the algorithm module should be a completely parallel device. That is, it should be capable of doing a string of jobs without control from the central processor. This parallelism feature, along with the ability to easily keep abreast of new techniques, leads to the conclusion that the device must be programmable.

Arithmetic Instruction Set

The next issue is the instruction set. If we should decide upon arithmetic instructions which effect only a single multiplication or a single addition, or both, we will be no faster than a general purpose computer. We really need a method where one instruction initiates a large number of multiply and add operations . . . a macro instruction. We could use some type of hardwired, read-only memory to control the operation performed by an arithmetic unit. Unfortunately, this would necessitate a hardware memory for each method or operation to be used. Further additions, or modifications, would lead to hardware changes to the processor. Thus,

[1] The philosophy has been implemented in Control Data Corporation's Matrix Algorithm Processor [MAP].

Ingwersen *graduated from the South Dakota School of Mines in 1967. Immediately following graduation he joined Control Data where he is a project engineer in the Development Division for CDC's 3000–1700 computer systems. During the time at Control Data Ingwersen has worked extensively on algorithm modules for the seismic industry.*

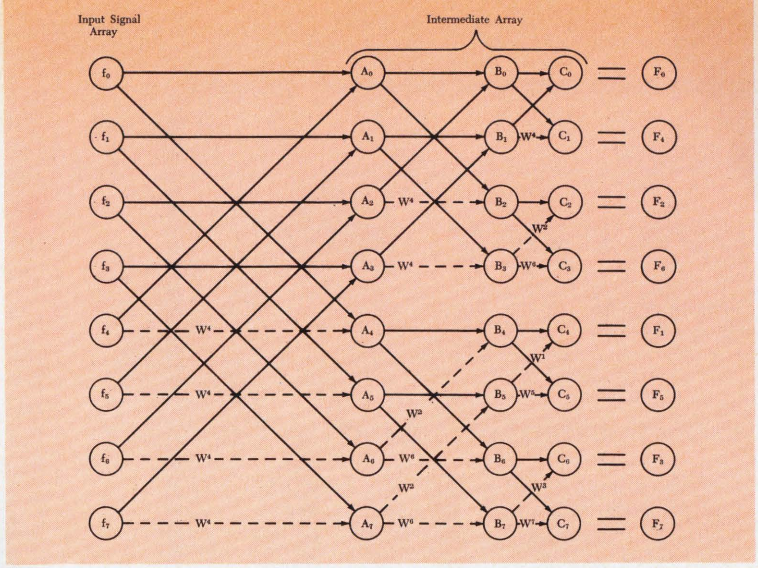


Fig. 2. Signal flow graph illustrating harmonic analysis of an 8-point input signal array $[f_0, f_1, \ldots f_7]$ by the Cooley–Tukey algorithm. Solid and dashed lines between circles represent multiplying factors of 1.0 and powers of $W[=e^{-2\ /N}]$, respectively. [adapted from Rader [McCowan [1966]].]

neither hardware nor software macro instructions provide an adequate answer.

What we really need is a compromise. We need an *Execute* macro instruction which can combine several arithmetic operations into one instruction. The *macro* needs to be structured such that we can easily chain very few execute instructions together and develop a complex routine such as a Fast Fourier Transform [FFT] or matrix multiplication.

Our hardwired arithmetic macro should be flexible. Thus, it should have variables which are entered, incremented, decremented, and shifted via non arithmetic instructions. These instructions we shall refer to as Augmenting Instructions.

Below is the equation for the macro Execute instruction used in CDC's MAP II.

$$\text{Eq. 1} \quad C_j = d[Rj] + \sum_{i=0}^{LC-1} Ap.\ bq \quad j = 0, 1, 2, \ldots PC - 1$$

where: $P = i\ ATF + jASAI$ and $q = i\ BIF + jBSAI$.

a_n = n^{th} A Array Variable
b_n = n^{th} B Array Variable
LC = number of multiply-adds/answer
PC = number of answers/macro
$[R_j]$ = contents of location where Result C_j is to be stored
AIF = A Array Indexing Factor
BIF = B Array Indexing Factor
ASAI = A Array Starting Point Index value
BSAI = B Array Starting Point Index value
d = 1 or 0 only

Note: R_0, a_0, or b_0 are variables set by the programmer for the macro as are LC, PC, AIF, BIF, ASAI, and BSAI. These values serve as a definition of array core areas.

This macro can be made into a simple multiply or add by using the proper variables for the equation. Further inspection shows that one execute instruction can be used for a convolution, Partial Matrix multiplication, vector multiplication, sum of squares, recursive filter operation, and others.

Comparing the equation of discrete convolution [Equation 2] with Equation 1 we see that

$$\text{Eq. 2} \quad C_j = \sum_{i=0}^{LC-1} a_{iAIF}\, b_{iBIF + jBSAI} \quad j = 0, 1, \ldots PC - 1$$

convolution is merely a *subset* of the macro capabilities. By making ASAI = 0, and d = 0, the macro becomes a convolution instruction.

For further illustration of the use of the macro variables, if BSAI was also zero, and the A Array was equal to the B Array $[a_0 = b_0]$ and PC = 1, the macro becomes a sum of squares instruction.

Augmenting the Execute Instruction

To Augment the Execute Instruction, we will need housekeeping instructions. First, we need a variable load instruction. When executed, this instruction will place its operand field in the designated variable's storage area. This instruction may be used for the initial entry of variables.

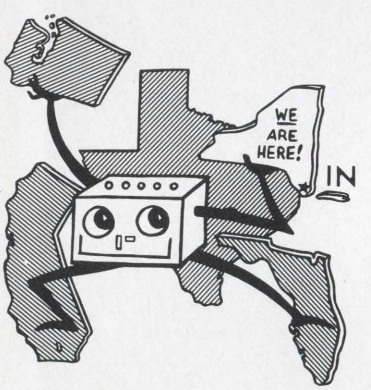
Several of the methods in which we may develop interest perform an operation several times with only minor alterations of variables. [A good example of this is the Cooley–Tukey Algorithm (6).] It would certainly reduce our programs if we had instructions to slightly alter macro variables, and an instruction which would allow us to use the same slightly altered macro[s] repeatedly.

To alter variables, we will need shift, add, and subtract instructions. These instructions will alter the previous contents of a variable according to the instruction. In order to use the macro subroutines repeatedly, we need an indexed jump instruction.

Ins 1
Ins 2
.
.
Ins $_{n-1}$
.
.
Ins $_m$
.

The nested indexed jump instructions are Instruction n − 1 and m.

Figure 1.



Further, we would like to be able to nest indexed jump instructions to give us capabilities for as many nested *do loops* as we can get into the program core area. Thus, the indexed jump instruction will give us subroutining and nesting capabilities.

FFT Applications

Looking at Equation 1, it is not immediately obvious that the Cooley–Tukey Algorithm can be computed in this manner. To develop an FFT from this equation, one looks to Shanks and Cairns (2) where the FFT is developed into a series of particle matrix multiples as shown below.

Figure 2 shows an 8 point example of the decimation in time version of the FFT.

From this graph, note that any component of the intermediate arrays can be described by an equation similar to Equation 3.

Eq. 3 $A_4 = F_0 + W_4F_4$

Expanding Equation 3 into real and imaginary components:

$$a_4 + i\propto_4 = f_0 + i\text{ɟ}_0 + [W4 + iv_4]\ [f_4 + 1\text{ɟ}_4]$$

Further:

$$a_4 + i\propto_4 = f_0 + 1\text{ɟ}_0 + w_4f_4 + iw_4\text{ɟ}_4 + iv_4f_4 - v_4\text{ɟ}_4$$

Separating this into real and imaginary parts:

Eq. 4 $$a_4 = f_0[1] + [0]\text{ɟ}_0 + w_4f_4 - v_4\text{ɟ}_4 \text{ and}$$
$$\propto_4 = [0]f_0 + [1]\text{ɟ}_0 + w_4\text{ɟ}_4 + v_4f_4$$

Thus, each of the values which constitute A_4 are calculated via a multiply-add operation. In fact, only four macro executes are necessary to compute this version of the Cooley–Tukey Algorithm (for up to 4096 points) in CDC's MAP II.[2]

Feedback Applications

To do the various types of feedback filtering, we need the capability to restoring answers into an array area. This area is equal to the operand area in each level of the FFT. Some forms of recursive filtering require a replace-add to storage areas. This explains the $[R_j]$ seen in Equation 1. Note that when the variable $d = 1$, we do a replace-add to storage and when $d = 0$, we do only a replace to storage.

Matrix Operations

The formula for vector multiplication is shown in Equation 5.

Eq. 5 $$\vec{a}[\vec{b}] = \sum_{i=0}^{LC-1} a_{jASAI} \cdot b_{jBSAI} \quad j = 0, 1, \ldots PC - 1$$

This is also a subset of the macro execute instruction. The other Matrix applications are also this simple.

---

[2] For further explanation, see Shanks and Cairns paper.

Flexibility

The only goal not fully discussed in flexibility, the ability to integrate new methods easily. One group of methods now being used in digital processing of radar traces places skirts on the frequency domain power or magnitudes spectrum. Here the K^{th} frequency [Ak] is given added power depending on the frequencies on either side.

For example:

$$A'_k = dA_{k-2} + eA_{k-1} + A_k + eA_{k+1} + dA_{k+2}$$

This is again a subset of Equation 1 and can be done with one execute instruction.

Another example of flexibility is the use of the macro in separating answers from a dual trace FFT operation. In general, the input array for an FFT program contains all real valued points [the imaginary values = 0]. If a second real valued trace is placed in the imaginary locations, both traces can be transformed at one time. The answer separation is done via the equations:

$$A_k = 1/2[RE(F_k + F_{n-k}) + iIm(F_k - F_{n-k})]$$ and

$$B_k = 1/2[Im(F_k + F_{n-k}) - iRE(F_k - F_{n-k})],$$

where array A was the *real* trace. An example program for the separation process contains one execute instruction inside two *do loops* such as those in Figure 1.

Conclusions

The purpose of this paper is to present a method for building a *peripheral* signal processor. In the introduction, the goals were presented. These goals are listed below in short phrases.

Convolution

Feedback Filtering

FFT

Flexibility

Matrix Applications

The method used to reach the goals is a macro arithmetic instruction. The value of the *macro* execute instruction stems from the fact that most of the problems incurred are a subset of the high-speed hardwired macro. Other complex problems, such as an FFT, are easily programmed with very few execute instructions. Note that the method with which the FFT operands are placed in core makes variable increment core storage an important factor.

Although the FFT programmed on this device will not be quite as fast as an algorithm module built specially to do the Cooley–Tukey algorithm, the macro algorithm module is still many times faster than a central processor program. Further advantages are realized by freeing the mainframe for parallel operation.

The device described in this paper is a compromise between a general purpose computer and a single purpose algorithm module. It will effect a savings of time in computing a wide variety of matrix and signal processing problems. By making the device *peripheral,* the module can then be added to scientific oriented systems where its effects would be most helpful. ■

References

1] J. L. Shanks, "Recursion filters for digital processing" *Geophysics,* vol 32, pp. 33–51, February 1967.

2] J. L. Shanks, Thomas W. Cairns, "Use of a Digital Convolution Device to Perform Recursive Filtering and the Cooley–Tukey Algorithm", *IEEE Transactions on Computers,* vol C–17, pp. 943–949, October 1968.

3] G-AEC Subcommittee on Measurement Concepts, "What is the fast Fourier transform?" *Proc. IEEE,* vol 55, pp. 1664–1674, October 1967.

4] H. D. Helms, "Fast Fourier transform method of computing difference equation and simulating filters", *IEEE Trans Audio and Electroacoustics,* vol AU–15, pp. 85–90, June 1967.

5] C. M. Rader and B. Gold, "Digital filter design in the frequency domain", *Proc. IEEE,* vol. 55, pp. 149–171, February 1967.

6] J. W. Cooley and J. W. Tukey, "An algorithm for the machine computation of comples Fourier series," *Math Comp.,* vol 19, pp. 297–301, April 1965.

TROUBLE-TRAN PRESENTS XTRAN'S ADVENTURES IN FORTRAN

By GEORGE N. VASSILAKIS

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

TROUBLE-TRAN EDITOR

software age

P. O. Box 2076
2211 Fordem Ave., Madison, Wis. 53701

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 $50.00

TROUBLE-TRAN'S Objectives:

1. To have fun.
2. To promote USA Standard FORTRAN by pointing out differences and inconsistencies of existing FORTRAN Compilers.
3. To alert programmers to the physical limitations of hardware.

PROBLEM OF THE MONTH

Problem 16: Illegal use of the DO loop control variable

Submitted by J. Kittle

```
      DØ 10 I = 1,10
      I = I + 1
      WRITE (6,5) I
    5 FØRMAT (I5)
   10 CØNTINUE
      STØP
      END
```

What values of I will be printed?

This problem clearly violates the FORTRAN rules; yet, many compilers don't bother to give a diagnostic. The author claims that this problem generated four different results on four different computers. On one computer the same problem coded on three different languages (FORTRAN, BASIC, AID) generated three different answers. Also, eleven computers executed this program without any diagnostics.
What did your computer do?

Dear TROUBLE-TRAN Reader:

This column is now in its second year and, like most programs you and I write, it must be revised and improved.
The first and most important change is that, from now on, the answer to a problem will appear on this column two months after the problem is published. This is necessary, in order to be able to read your comments before I write the answers. Writing the answer two weeks before a problem is published does not even give me the opportunity to comment on typographical errors.
A few months ago, I had asked for volunteers to test the problems before they are published and I received many offers. Unfortunately, the task of corresponding with volunteers and meeting publication deadlines is time consuming (more time than I can spare). My job will be simplified and the answers will always be correct if I wait until I read the mail.
Another change which may disappoint some of you is the termination of prizes for correct solutions. This decision resulted from several complaints of readers who always receive their copy of Software Age late. In fact, many times I receive mail from readers several days before I receive my copy of this magazine.
Instead of first and second prizes, this column will recognize those who send correct solutions by publishing their names. You may still profit by submitting problems which are worth $50.00 each when published. For the next few months, I would like to see more problems similar to last month's problem on "Prime Numbers".

USA Standard FORTRAN

Those of you who are interested in purchasing a copy of this document should write to USA Standards Institute, 10 East 40th Street, New York, N. Y. 10016 and inquire about the following documents:

X3.9–1966 FORTRAN
X3.10–1966 Basic FORTRAN

Sorry I do not know the price. I received my copy free from a CDC representative.

XTRAN

TROUBLE-TRAN Winners:

$25.00 for submitting May problem: R. A. Howell, The Dow Chemical Company, BIS-ADS, Hopkis Building, Midland, Michigan 48640.
$25.00 first prize for May: A. I. Wasserman, The University of Wisconsin, Computer Sciences Department, 1210 West Dayton Street, Madison, Wisconsin 53706.
$15.00 second prize for May: F. E. Miller, 1624 Alabama Street, Lawrence, Kansas 66044.
$25.00 for submitting June problem: J. S. Blanchard, General Electric Co., Ordnance Systems, Room 2468, 100 Plastics Ave., Pittsfield, Mass. 01201.
$25.00 first prize for June: E. S. Jenkins, The Pennsylvania State University, Computer Building, University Park, Pennsylvania 16802.
$15.00 second prize for June: Mary E. Rafter, State of Michigan, Department of Public Health, 3500 N. Logan, Lansing, Michigan 48914.

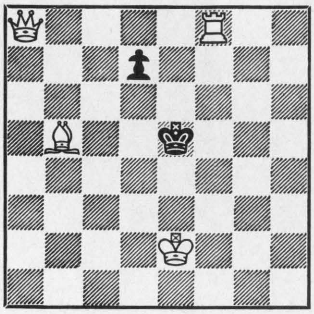

Don't just sit there.
Take off...with RCA.

CHECKMATE

by
GEORGE N. VASSILAKIS
TRW SYSTEMS GROUP

Problem 18

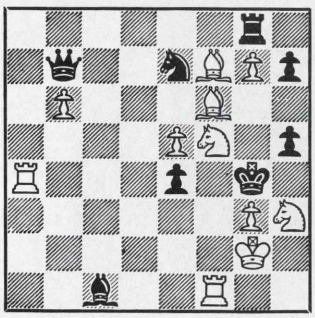
A. BELLAS



White Mates in Three

Problem 19

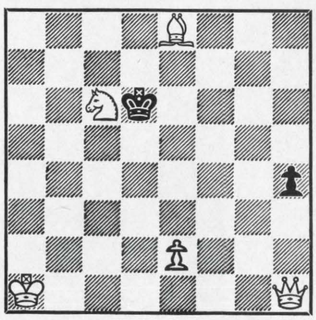
E. HOLLADAY



White Mates in Two

Problem 20

G. DOBBS



White Mates in Three

THE MINI-MINIGAME

According to Stephen E. Lowe of Van Nuys, California, this game was played in Philadelphia, in 1936.

ARNOLD (White)	HANAUER (Black)
1. P–Q4	N–KB3
2. P–QB4	P–K4
3. P–Q5	B–B4
4. B–N5	N–K5 !
5. BxQ ?	BxP mate

World Champion Paul Morphy (1858–1859)

Morphy, the greatest USA chess player that ever lived, was born in New Orleans on June 22, 1837. He learned the game from his father at the age of ten, and two years later he was able to beat his uncle, the chess king of New Orleans.

At the age of twenty he went to New York, where he gained first place in the first American chess championship. The following year he traveled to Europe, where he defeated every opponent he played, including Adolf Anderssen. His greatest disappointment was the refusal of Staunton to play him.

Morphy returned to New Orleans and declared his brief but brilliant chess career closed, at the age of twenty one. After trying unsuccessfully to take up law (his father was a judge), Morphy gradually relapsed into a state of seclusion and eccentricity, and died of congestion of the brain at the age of forty-seven.

During his illness, Morphy imagined himself persecuted by people and suspected his brother-in-law of trying to poison him. He would see no one except his mother. His mode of life was a daily walk and going to the opera alone, never missing a performance. He would only talk about his father's fortune ($146,162.54), and the mention of chess was sufficient to irritate him.

Solution to Problem 15

1 N–R8,K–Q3; 2 K–Q4,K–B3; 3 Q–Q5 mate

Solution to Problem 16

1 Q–R6! if 1 . . . . , PxQ; 2 NxP mate
if 1 . . . . , P–N4ch; 2 QxP mate

Solution to Problem 17

1 R–Q8ch,RxR; 2 R–B8ch,KxR; 3 N–N6 mate
if 2 . . , K–R7 or K–B7; 3 P–N6 mate

The PL/1 a new dimension

David Bernstein

Purpose

The purpose of this article is to explain the functions and capabilities of the PL/1 pre-compile time facilities. The main emphasis will be placed on describing how pre-compile time facilities can be used to solve specific classes of programming problems. The technical details of the operation of the preprocessor will be kept to a minimum. The reader is referred to the reference for more details.

Judicious use of preprocessor capabilities can result in many advantages to the programmer. Initially, the dangers of handling large decks of cards can be avoided because entire programs can be placed on a source library, and can be de-bugged by changing individual program statements on the library. This saves the programmer from the usual dangers involved in handling card decks. Coding time can also be saved because common routines and record formats can easily be included in any program once they have been written and placed on the library. Under second generation computer concepts the programmer wasn't concerned with the size of his program as long as it was small enough to fit into the machine. However, the machine can be used more efficiently by writing smaller programs so that, in multi-programming, more jobs can be running simultaneously. The preprocessor allows you to compile only those portions of your program that are necessary for a particular run. This is especially applicable in the case of writing utility programs. In addition, the use of pre-compile time facilities opens up an entirely new way to write programs and encourages programmers to come up with imaginative ways to solve problems.

Description of the Preprocessor

The preprocessor is a program that operates on source text before the text is compiled by the processor stage of the compiler. At the preprocessor level the user's source coding is scanned for special expressions called preprocessor statements. These special statements cause the original source coding to be altered in a way specified by the programmer. The altered text then serves as input to the second stage which is the regular compilation. The preprocessor scans each character of the input source coding looking for special pre-compile time statements. As long as these special statements are not encountered, the original coding is undisturbed. When a pre-compile time statement is encountered, it is executed right there at the preprocessor stage. This execution can cause the scanning of the source program to be altered in one of two ways:

PREPROCESSOR:

n programming

RE (OLDMASTINPUT, NEW
), 432,8), PFILE OUTPUT, 1
TER (7), 2 DESCR CHAR
XED, 1 TRANS, 2 TNUMBER
(ED, 2 TQ FIXED (5), CODE
DURE; DECLARE (OLDMAST
3LOCK (FIXED, 432,8), PFIL

1. The preprocessor may be instructed to continue its scan from a different point in the program. This can result in one of two possible situations. In the first place source text that was already scanned can be re-scanned and can be included in the program many times. In the section called *Using the Preprocessor* there is an example illustrating this point.

   The second result that is possible when a re-scan takes place is that portions of the source text may be skipped for a particular run of a program. For example, a multi-purpose program could be designed to perform more than one function. If the program size is to be kept to a minimum only the needed sections of the program are compiled. This type of operation is called conditional compilation.

2. Replacement activity may be initiated. This capability can best be described by the use of an example.

   1. % DECLARE A CHARACTER;
   2. % A = 'B + C';
   3. X = A;

   The first statement is a preprocessor DECLARE statement that indicates that A is eligible for replacement wherever it is encountered in the program (all preprocessor variables must be declared). The percent sign is the preprocessor indicator. Statement 2 is a preprocessor assignment statement which assigns the character string 'B + C' to A.

   The third statement is a non-preprocessor statement and is not executed at this time. However, because this statement contains a preprocessor variable (A) the current value of A will replace it in that statement. Statement 3 will then appear in the compiled listing as:

   X = B + C;

Once preprocessor statements have been evaluated, they are blanked out and appear in the compiled listing as blank lines.

To summarize thus far, the preprocessor performs two functions, conditional compilation of parts of a program, and the replacement of variable names by some value generated at pre-compile time.

Using the Preprocessor

There are many preprocessor operations available and the syntax of some are similar to the regular PL/1 operations; however, the results of these manipulations are nothing more than changes in the status of the scanning and outputting of source text. The rest of this article will show how the pre-compiler is used in specific situations.

Perhaps the simplest use of the preprocessor is for incorporating strings of source text into a program. By using the standard Operating System Utilities, it is very easy to create and maintain a Partitioned Data Set of source text. Each group of source statements is given a name and may be used by any programmer. The important thing to keep in mind is that the source statements loaded onto a library do not have to be complete programs. By the use of a simple '% INCLUDE PGNAME' the indicated text is brought into the program at the specified point. For instance, all of a department's record formats could be added to the library and with one statement the appropriate format could be included in the program. For example, suppose we have a format called "RECORDA" that has three elements:

```
1. RECORDA,
   2 CUSTOMER# CHAR (4),
   2 ACCT-RECEIV FIXED
     (6,2),
   2 DATE  CHAR (4),
```

With one statement the entire format can be included into any program:

```
PROCA: PROCEDURE:
     DECLARE
        .
        .
        .
     % INCLUDE RECORDA;
        .
        .
        .
        .
```

The following source program is generated:

```
PROCA: PROCEDURE:
     DECLARE
        .
        .
     1 RECORDA,
       2 CUSTOMER# CHAR
         (4),
       2 ACCT-RECEIV FIXED
         (6,2),
       2 DATE  CHAR (4),
        .
        .
```

This simple example shows a very short format being introduced into the program, but the kind of text that can be included isn't limited to record formats or to size. Any valid (or invalid for that matter) PL/1 text is permissible, since the preprocessor only checks for unmatched quotes and unmatched comma delimiters.

In addition to record formats being included in a program, general purpose routines that have to be tailor-made are especially appropriate. An example of such a routine is a heading routine. The author has written a routine that can be included in any PL/1 program and performs the following functions:

1. Opens the standard print file.
2. Writes heading and footing lines when the end of page is reached.
3. Automatically maintains and updates the page count.

All that the user supplies to the program is any heading or footing lines that he wants printed. If none are supplied standard ones are printed. The significance of this type of routine is very simple. Those types of functions are done in practically every program. By writing it once and allowing other programmers to use it, countless programmer hours wasted in duplication are saved.

Another very useful function of the compile time operation is that of substitution. This ability is especially useful when one is writing general purpose utility program. For instance, let us say that we have a program that prints the value of any field from a record. The field and the length of the field are supplied at pre-compile time. The following shows how the program looks before the preprocessor operation takes place:

```
PROCA: PROCEDURE;
  % DECLARE VAR1 CHAR,
    VAR2 FIXED,
       .
       .
  % VAR1 = 'FLDA';
  % VAR2 = 5;
       .
       .
       .
    PUT EDIT (VAR1)
      (A(VAR2));
```

Notice that all of the preprocessor statements have been prefixed with a percent sign. This informs the preprocessor that these statements must be evaluated at pre-compile time. Notice, also, that any pre-compile statement must first be declared and that the only two types of variables that are allowed are fixed decimal and character. The length of the variables is not specified. The character variables are of varying length, and the decimal variables have the default length (5,0). The values are assigned to the two variables as indicated in the program. When the print statement is encountered, the current values of VAR1 and VAR2 are substituted in the statement. In the source program the statements that contained pre-compile time variables show up as blank lines and the substituted values show up in the program:

```
PROCA: PROCEDURE;
       .
       .
       .
       .
       .
       .
       .
PUT EDIT (FLDA) (A(5));
```

This technique is extremely useful in writing multi-purpose programs

Programmers and systems analysts...

Univac: Where you can help make the world a little bit better

There is no pat solution to the problem of our ghettos. But concerned governmental agencies and private industry, with the help of UNIVAC® computers, are making steady progress toward a healthier environment in the depressed Watts area of Los Angeles.

One of the world's largest architectural and engineering firms is Daniel, Mann, Johnson and Mendenhall. The company uses a sophisticated UNIVAC computer system to supply its planners with information on population density, housing data and other facts in fractions of the time that used to be required. As a result, Watts residents are nearing a residential and business environment dramatically improved from conditions only a short time ago.

We at Univac know that our kind of people want to feel that they are helping find new solutions to the world's social problems. So you won't find yourself doing busy work here. You will be occupied with meaningful projects.

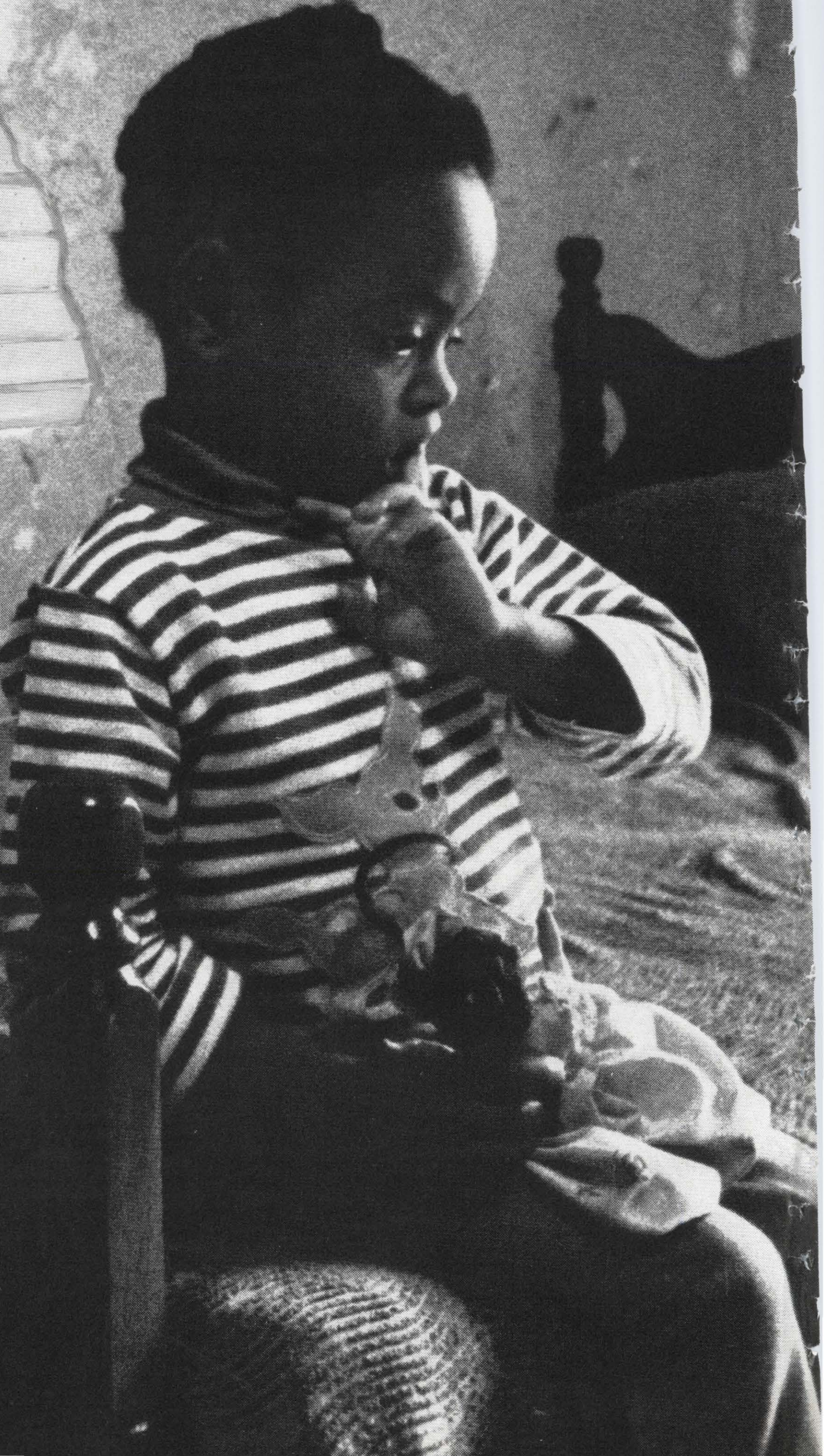
You will be making a real contribution. You will receive compensation and promotion as fast as you are ready, of course. But your ultimate reward will go beyond purely material matters.

If we sound like your kind of company, we know that you are our kind of person. Please look over the job opportunities on the facing page.

SPERRY RAND

UNIVAC

An Equal Opportunity Employer M/F



because, when high level languages are used, record fields are usually referred to by the use of a symbolic name rather than by relative position from the beginning of the record as is the case when using AUTOCODER or BAL.

The use of conditional statements at the preprocessor level allows the user to compile pieces of his program to satisfy specific situations. This ability makes the source program smaller which in turn can help increase throughput when operating in a multi-programming mode. For instance, suppose a program has two routines, one for printing a record and one for writing a tape copy of the record. The user has the option of either getting tape output, printer output or both. If one of the options is not required, there is no need to compile that particular routine into the program. For this example suppose that all the user wanted was printed output.

```
PROCA: PROCEDURE;
  % DECLARE (TAPE, PRINT)
    CHAR;
  % TAPE = 'NO';
  % PRINT = 'YES';
      .
      .
  % IF TAPE = 'NO' % THEN
    % GO TO NOTAPE;
  TAPERTN:
      .
      .
      .
      .
  % NOTAPE:;
  % IF PRINT = 'NO' % THEN
    % GO TO NOPRINT;
  PRINTRTN:
      .
      .
      .
      .
      .
  % NOPRINT:;
```

When the first preprocessor conditional statement is encountered, it is evaluated and since the condition is true, a branch is made to the preprocessor label % NOTAPE. What this means is that for this running of the program the tape routine will not be compiled. Since the next statement turns out to be false, the preprocessor falls through and therefore scans and outputs all of the statements in the print routine for eventual compilation. The final program that is compiled will look like this:

```
PROCA: PROCEDURE;
      .
      .
      .
      .
  PRINTRTN:
      .
      .
```

Perhaps the most powerful ability of the preprocessor is that of actually generating source text. One way this is accomplished is through the use of the compile time DO loop. Let us first consider a very simple application. Suppose we wanted to generate a particular statement a different number of times depending on the program requirements. Every time that the program was run, a number 'CNT' would be supplied to the program indicating the number of times that the statement is to be generated. The preprocessor coding would be as follows:

```
% DECLARE (I, CNT) FIXED;
% CNT = 3;
% DO I = 1 TO CNT;
  Z (I) = X (I) + (I);
  % END;
```

For this run, the statement in the loop would be generated three times since 'CNT' was set equal to 3. The following statements would be generated:

```
Z(1) = X(1) + Y(1);
Z(2) = X(2) + Y(2);
Z(3) = X(3) + Y(3);
```

The above is fairly trivial example of a DO loop. To get a little more sophisticated, suppose we had a program that could print up to five fields from a record, but the actual number of fields is a variable that is supplied by the user. This variable is again called 'CNT'. The names of the fields to be printed out are defined to the program as 'FLDx' where x can be any number from 1–5. In this example we make use of the concatenate operator and the SUBSTR built-in function which are both perfectly legitimate operators to use at the preprocessor level. For this particular run, we will print 3 fields out: 'NAME', 'ADDRESS', 'ACCT #'.

```
% DECLARE (I, J, CNT) FIXED
(FLD, K, L, FLD1, FLD2, FLD3,
  FLD4, FLD5) CHAR;
% CNT = 3;
% FLD1 = 'NAME';
```

```
% FLD2 = 'ADDRESS';
% FLD3 = 'ACCT #';
      .
      .
      .
% K = '123';
% DO J = 1 TO CNT;
% L = SUBSTR (K, J, 1);
% FLD = 'FLD' || L;
PUT EDIT (FLD) (A(20));
% END:
```

The following source statements will be generated:

```
      .
      .
      .
PUT EDIT (NAME) (A(20));
PUT EDIT (ADDRESS)
   (A(20));
PUT EDIT (ACCT #) (A(20));
      .
      .
      .
```

The loop is performed 3 times because 'CNT' was initialized to 3. The operation of the SUBSTR operator is as follows: From the string K assign to L the first character beginning at the jth position of K.

The first time through the statement % FLD = 'FLD' || L; is immediately executed since it is a preprocessor statement. Since L initially has the value 1 the value of % FLD is equal to 'FLD1'. A scan of the PUT EDIT statement then takes place. Since FLD is a preprocessor variable it is immediately replaced by its value 'FLD1'. The statement is then re-scanned and the value of 'FLD1' which is 'NAME' is replaced in the PUT EDIT statement. A scan of the expression then takes place. Now, since 'FLD1' is a preprocessor variable, its value 'NAME' is replaced in the expression. The first iteration is now complete, and the value of J is incremented, and the second iteration begins.

Conclusion

The above examples give the flavor of what the capabilities of the preprocessor are. They have by no means presented a complete range of what can be done by using the full PL/1 language capabilities of the preprocessor. As alluded to in the last example, many different PL/1 operations can be performed at pre-compile time, and one or more can be combined into one expression to perform complex operations. However, to re-emphasize, the most powerful and simplest preprocessor operations are substitution and conditional compilation of sections of a program. By doing these two operations alone the programmer can certainly use PL/1 with a great deal more power compared to using the language without the preprocessor. The only requirement for the use of the compile time facilities is that the programmer has to shake off his old ways of doing things and begin to expand his thinking into a new and more flexible dimension of programming.

Reference

IBM System/360 PL/1 Reference Manual, C28-8201-1, IBM Corporation, White Plains, New York.

David Bernstein *received an MBA from the Bernard M. Baruch School of Business Administration in 1969. Since that time, as President, he has managed and directed the activities of Advanced Computer Software Services, Inc. For three years before this, he was a Senior Associate Analyst at IBM. Bernstein has, above all, extensive machine and language abilities.*

personal lines

Gerald G. Wisz and **Edward N. Murray** have been elected Assistant Vice Presidents of Marine Midland Services Corporation . . . **Charles G. Abbott** has been appointed Vice President in charge of the Consulting Services Division of Data Management Services, Inc. . . . Freas–Rooke Computer Center at Bucknell University has appointed **George L. Weber** as a systems analyst.

Larry Vehorn, Marvin Leeger, Larry Serbie and **Errol Rovner** have all been promoted to Director, Management Services, at Wellington Computer Systems Incorporated . . . Commed, Inc., a subsidiary of Comress, has named **John S. Williams** as Vice President and Technical Director . . . **Joseph F. Cashen** has been named Engineering Director for Industrial Controls Systems at Honeywell's Computer Control Division in Framingham . . .

Max Muir has joined Information International as senior project engineer, **Richard J. McQuillin** as senior analyst–graphic systems . . . New at Computek Corporation are **Ronald A. Zarski** and **Kenneth K. Warlick**—both as senior project specialists . . . **James A. Ferguson** has joined Infodata Systems, Inc., as Manager, Engineering Software Systems . . . Computer Image Corporation has named **William W. Jacquish** as Director of Special Projects.

Virgil L. Swearingen has been appointed Applications Programming Manager of the Technical Division at Tymshare Corporation . . . COM-NET has named **Douglas M. Hutt** as Manager of Operating Systems in Washington, D. C. . . . **Joseph E. Izzo** is new Director of Western Region Services at Comress, Inc.

David W. Coats has been appointed Manager, Computer Center Services, by the Field Services Division of Computer Industries, Inc. . . . Brandon Systems Institute has appointed **Edwin E. Klein** as Director of Research and Development . . . Tally Corporation announces a new manufacturing management team: **David H. Moultan,** Director of Manufacturing; **Bart Parsini,** Production Manager; **Michael Tremoglie,** Production Control Manager.

John Certo has been named Senior Applications Engineer, Product Marketing Group, at Applied Logic Corporation . . . Electro-Optical Systems has promoted **Donald L. Smelser** to Manager of Organization Development . . . Appointed Manager of Systems Programming for the Computer Systems Division of Graphic Controls Corporation was **Dr. Eamonn McQuade . . .**

John G. Seay has joined Leasco Systems and Research Corporation as Seminar Consultant. **Peter F. Urbach** has come aboard as Director of Product Planning for Leasco Information Products . . .

Applied Computer Sciences, Inc., has appointed **Richard N. Germano** as Senior Consultant . . . At System Development Corporation, **Marc Bendick** was named Manager, Engineering and System Evaluation Department, in the Air Operations Division.

new products

A 4800 bit per second, automatically equalized, **high-speed data modem** has been developed by American Data Systems, Chatsworth, California.

Called, the ADS–448, the new modem eliminates the problem of manual equalization. The receiver automatically equalizes within milliseconds after turn on, and automatically and continuously tracks any changes in prevailing telephone line characteristics without adjustments of any kind. Error rate is said to be five times less than other existing modems.

The new modem transmits and receives data at 4800 bits per second (or any combination of 1200, 2400, 3600, equaling a total of 4800 bits per second), over Bell System Series 3002 telephone data lines. These lines can be either unconditioned, or C1, C2 or C4 conditioned.

Another major feature of the new modem is its front panel display, which provides not only a visual indication of relative Line Conditions, but also of Receiver and Transmitter Data and Baud Rates, Carrier Detection and Receiver Phaselock.

ADS–448 also incorporates a Loop Back Capability for both Local and Remote operations, as well as other internal self-checks to aid in system trouble-shooting.

For more information, circle No. 7 on the Reader Service Card.

* * *

A computer-oriented **employee attendance reporting system** has just been announced by North Electric Company, Galion, Ohio, a subsidiary of United Utilities, Inc.

According to Norman Hockler, director of marketing, the bottle-neck of time clocks, collection and distribution of time cards and in and out timeclock punching are all eliminated with the new attendance reporting system (ARS). Simple insertion by the employee of his own, personal, specially-prepared identification card into an ARS station automatically provides payroll time-keeping information at the computer center.

Employees are issued wallet-sized plastic cards pre-punched with their payroll identification code. When they insert their card into any ARS station, their time-in and time-out is automatically recorded on punched cards, magnetic tape or as typewritten copy and then it is fed into the computer.

A single digital clock assures time accuracy, and the entire time-in or time-out procedure is handled within two seconds. Separate attendance reports for any employee or group of employees can be produced at any time as a by-product of the ARS in addition to its providing payroll information input.

For more information, circle No. 8 on the Reader Service Card.

A new unit, the **Model 208 Control Computer,** has been added to the family of mini-computers of Computer Automation, Inc., Newport Beach, Calif. It is an 8-bit, stored program, parallel computer and features a new high speed memory with 2.6 micro-seconds full cycle time.

The Model 208 is expected to find service in concentrating and routing messages, monitoring and controlling traffic and computing service charges. It is also expected to find wide application in data acquisition and process control systems where high speed, reliability, I/O flexibility and programming convenience are important.

For more information, circle No. 9 on the Reader Service Card.

* * *

GDI Inc., a manufacturer of computer peripheral equipment, has announced **the Model CT–300 Card Transmitter,** designed to transmit punched card data at a rate of up to 2400 BPS over standard dial-up or private lines. The system will interface directly with a Bell 202E2 or 202C or equivalent data sets. The interface conforms to EIA standard RS–232B and transmits data in standard bit serial USASC II language. A directly compatible interface with MDS 1103 and 6403 data recorders is also available.

The model CT–300 is designed for use in applications where the prime requirement is for transmission of punched card data from remote sites to a central location where it can be received by a data recorder, a central data switching unit, or the central computer. In these applications, keyboards and recorders are unnecessary and have been eliminated. This greatly simplifies the system and the result is a low cost reliable system.

For more information, circle No. 10 on the Reader Service Card.

DEEP/360 is a support package designed especially **for BAL and COBOL programmers** who are frustrated by data exception errors while testing. DEEP/360, a product of Macro Services Corp., Boston, Mass., eliminates the problem of premature job cancellation caused by invalid data. It *repairs* faulty data, re-executes the affected instruction, reports the action via console message, then continues with the user's program. The purchase price includes a BAL source listing, a job stream for cataloging DEEP/360 into the user's relocatable library, and operating instructions.

For more information, circle No. 11 on the Reader Service Card.

* * *

The **PUNCHMASTER is a new automated control** from California Computer Products, Inc. of Anaheim, Calif. The unit is attached to an IBM cardpunch or verifier to speed the conversion of source data into punched cards. The time-saving features include: a keypunch buffer allowing the operator to keypunch or verify at normal typing speeds; a program memory capable of handling up to 22 different card formats at one time; automatic left zero fill, with or without two-field accumulator; and an optional instruction display for faster, more reliable operator training.

For more information, circle No. 12 on the Reader Service Card.

* * *

IBM has announced several major additions to its tele-processing product line, including **a small keyboard terminal for telephone audio-response systems.** The terminal is called the 2981 alphanumeric keyboard. It is a special order device that allows users to enter alphabetic and numeric information, as well as queries, into a system/360 and get computer spoken responses over the telephone.

Once the user has established telephone contact with the computer, he keys in his message, selecting from among 48 keys. The 26 alphabetic keys are arranged sequentially so that anyone can locate a letter quickly. There are also 22 keys representing numbers and special characters. As each key is depressed, an audible tone confirms that the information has been sent to the computer.

The 2981 uses elastic diaphragm switch technology (EDST)—flat, prewired switches that eliminate mechanical key linkage. The absence of moving parts in this technology is expected to keep maintenance requirements to a minimum.

For more information, circle No. 13 on the Reader Service Card.

PLUS, a **Program Library Update System** has been announced by Cullinane Corp., Boston, Mass. The program provides for storage and maintenance of source language programs on tape or disk. Any program language may be stored or intermixed in the file including COBOL, Assembly, PL/1, and FORTRAN. Test data decks, object decks, and job control language decks may also be stored on the library. The program also will generate a job stream file with job control setup to compile or assemble modified programs.

One feature of the program is the generation of a Library Index Report following each run. This report is a table of contents of the PLUS program library. It includes such information as a Version Number automatically given to each program and increased with each entry of an entirely new program version, and a Modification Number given a program and increased each time any change is made. The Program description, the date of last revision, the language, the author, the number of statements, and the number of changed statements are also reported.

In addition to the Library Index report, two other reports are produced. A Report of Changes itemizes changed statements, providing a valuable historical record of all maintenance activity on a program. A Job Schedule Report, listing job control setups, is produced when this facility of the program is used. PLUS also reproduces all or portions of any source programs on the printer listing or on punch files.

For more information, circle No. 14 on the Reader Service Card.

* * *

A new unit, the DIMBO–10, manufactured by BCD Computing Corp., Deer Park, New York, is a self-contained **electronic information storage and retrieval system.** It utilizes a disk memory for information storage and has optional input-output configurations.

The DIMBO–10 system will store 1200 to 40,000 32-character records; 64-character or larger record lengths, and larger file capacities (practically without limit) are available as options. Other options include magnetic or paper tape input and output for file loading and audit trail, an arithmetic capability, and remote consoles for inquiry into the central file.

Average access time is less than 0.1 second. Standard operation includes: ENTER a new item; DELETE an item; SEARCH for an item; ALTER an item and restore it; and PRINTOUT the entire file.

DIMBO–10 is intended for the user who must search a large list of entries in order to retrieve information on one of them. A search consists of locating an item from its identifier which is then typed in the DIMBO–10. The system locates the proper item and types all the stored information for the user's study. The item may then be altered and returned to the file or deleted and a new item entered.

For more information, circle No. 15 on the Reader Service Card.

S/A market place

where you can reach 130,000 programmers, mathematicians, analysts and EDP managers

RATES

Classified—75¢ per word, minimum order $7.50. Blind ads are an additional $10.00. Payment must accompany copy.

Display Classified—One inch by one column, $75.00. Column width is 2¼". Help wanted and employment services ads will be accepted in the classified section. Ads of ⅛ page or more will be listed in the Advertisers' Index, and will qualify for the free resume service.

Closing Date—the 10th of the month preceding (i.e. May issue closes April 10th). Send order and remittance to:

software age
Classified Department
P. O. Box 2076, Madison, Wisconsin 53701

SITUATIONS WANTED

TAX AUDITOR without EDP exp. seeks part time work, Los Angeles area. S/A Box 803, 2211 Fordem Ave., Madison, Wis. 53701.

EMPLOYMENT AND SEARCH COMPANIES

SCIENTIFIC PROGRAMMER

Offering a stimulating variety of work with primary emphasis directed toward the analysis and programming of reactor engineering and physics problems. Fortran programming utilizing IBM 1130, 1800, 360/40, 360/50, 360/75 and CDC 6400. BS or MS degree in mathematics, physics or engineering with 2 or more years scientific programming experience required. Resume should include references and salary. Write or phone Frank Nitzberg, Personnel Coordinator, phone (313) 962–9510. ATOMIC POWER DEVELOPMENT ASSOCIATES, INC., 1911 First Street, Detroit, Michigan 48226. An Equal Opportunity Employer

PROGRAMMERS (3)

Large financial firm needs (1) Senior, salary to $28,000; (2) Intermediates, salaries $16,000 to $23,000 to help open new company locations. Entire fee, traveling, if necessary—moving expenses are all paid by your new employer.

ERIN EXECUTIVE AGENCY
132 W. 34 St., New York, N. Y. 10001
Phone: (212) 594–8296

DATA PROCESSING POSITIONS IN THE SOUTHWEST
Contact:
DATA PROCESSING CAREERS
Richard Kemmerly, Suite 1109, Stemmons Towers West, Dallas, Texas 75207. 214/637–6360.

PENNSYLVANIA, NEW JERSEY, DELAWARE contain diversified opportunities for professional talent in all functions of E.D.P. If you desire to pinpoint your search in our area, contact us. With 9 highly effective years behind us we need not employ gimmicks nor shotgun duplicating techniques. We are large enough to have extensive knowledge of the market yet disciplined enough to remain personal and confidential. All fees paid. K. J. Gillespie, ANDERSON–TAYLOR Associates, 230 S. 15th St., Philadelphia, Pa. 19102. (215) KI 6–6333.

PROFESSIONAL SERVICES

ADD / SUBTRACT 6-DIGIT HEXADECIMALS in seconds with 100% accuracy. The pocket HEXADDER, $15. Free brochure from HEXCO, Dept. SA, P. O. Box 55588, Houston, Texas 77055.

CONSULTANTS: Commercial applications 1400, 360. Conversions, programming, systems design, testing, documenting. Job or contract. Computerized business mailing list, United States, $35 per 1000 names. ALLIED COMPUTER CONSULTANTS, Box 133, Fairlawn, N. J. 07410.

EXP. PROG./ANAL. SEEKS P/T WORK: S/360 DOS COBOL. Resume upon request. Reply Apt. B6, 5502—14th Ave., Brooklyn, N.Y. 11219.

EDP RESUMES

A comprehensive manual on how to write your own resume, written especially for EDP personnel. Send $4.25 to

K & H ASSOCIATES
Dept. S, P. O. Box 265
Horseheads, New York 14845

WE WILL CONTRACT system planning, application, or system work to $10,000. We possess a complete background in statistics, linear programming, mathematics, civil engineering, processors, business data systems, data retrieval/processing; with a unique knowledge of OS/360 software and hardware. For more information in your field, write: SYSTEM PLANNING AND SOFTWARE MANAGEMENT, S/A Box 802, 2211 Fordem Avenue, Madison, Wisconsin 53701.

PRODUCTS

WANTED—USED CARD FILES: MacDonald, Box 68–501, Miami, Florida 33168.

CLASSIFIED ADVERTISING ORDER FORM

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 = $_________

☐ Blind Ad $10.00 Insert_________time(s) Total Enclosed $_________

NAME_________

ADDRESS_________

CITY_________ STATE_________ ZIP_________

Signature_________

WORD COUNT: Includes name and address (unless blind ad). The name of the city (Des Moines), state (New York) and zip code are counted as a total of two (2) words. Each abbreviation, initial, single figure or group of figures or letters count as a word. Symbols such as 35mm, COD, AC, etc. count as a single word. Hyphenated words count as two (2) words.

Coming in September SA's Annual Guide to EDP Employers

You probably aren't our kind of engineer.

No offense intended. We just know what we want.

We're computer memory specialists. If you know our products, you know we're good.

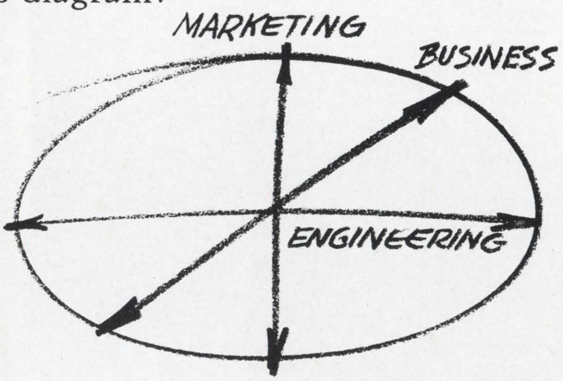
Not surprisingly, we're growing fast.

So we need engineers. Good minds. Good technical backgrounds. But that's not all.

You see, back when we were small we learned an important lesson. Each of our engineers had to get involved in everything: research, design, costing, marketing, production – you name it. Being involved, they became committed. And worked better for it.

That's how we got our reputation for doing what we do better than anyone else. We intend to keep it.

That's why we take such a close look at all applicants for our engineering positions. We'll talk to you about interests and aspirations and experience. And, at some point along the line, we'll draw this diagram:



It's the essence of what we mean by our kind of engineer. An interest in and understanding of all aspects of our company's activities. You could call it balance. Whatever it is, it works for us.

* * *

What can we offer you?

First, a good salary. (We want the best and we're willing to pay for it.) Second, the opportunity to advance rapidly along either engineering or administrative lines. (Lots of outfits will tell you this, then stick you in an isolation ward. We mean what we say.) Third, the opportunity to grow with the best engineering staff in the industry. Fourth, the satisfaction and pride of seeing the products you work on produced and sold.

* * *

We make memory systems and stacks and planes and printed circuits. We're looking to the future with research in plated wire and films. And we're in the process of introducing a brand new product: MAC 16, a compact, inexpensive Multi-Application Computer. Our division is going to keep right on growing. And we'll need even more of our kind of engineers.

If we sound like your kind of company and if you have an M.E. or E.E. degree and computer-related experience, let's get together.

Write me a letter: Mr. E. A. Gage, 6201 E. Randolph Street, Los Angeles, California 90022. Or, call me collect at (213) 722-6810.

Naturally, we're an equal opportunity employer.

LOCKHEED ELECTRONICS COMPANY

Data Products Division • Lockheed Aircraft Corporation

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 __________ Signature __________

Home Address __________ (city) __________ Home Phone __________

__________ (state) __________ (ZIP code) Age __________ 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
P. O. BOX 2076
2211 FORDEM AVE., MADISON, WIS. 53701

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, check below the boxes of those companies to which you want copies of your form sent. Mail to Software Age, P.O. Box 2076, 2211 Fordem Avenue, Madison, Wisconsin 53701. (**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.)**

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

....
Name **Street Address**

....
City **State** **Zip Code**

................................../................................
Prime Experience in What Industry **My Specialty**

☐ **Technical Degree** ☐ **Non-Technical Degree** ☐ **No Degree**

.... ☐ **I Have Analog/Hybrid Experience**
Year Born

software age

P. O. Box 2076
2211 Fordem Avenue
Madison, Wisconsin 53701