PL/I BULLETIN NO. 2

August 1966

CONTENTS

Item No.	Description	Page
PB2.0	EDITOR'S NOTES	1
PB2.1	NEWS ITEMS	4
PB2.2	CORRESPONDENCE (A. P. Yershov, T. B. Steel, Jr., Prof. Dr. Leon Lukaszewicz, W. M. McKeeman, Bryan Higman, R. A. Zernlin	6
PB2.3	WORKING PAPERS	
PB2.3.1	Bryan Higman: Some Reactions to PL/I	10
PB2.3.2	I. D. Greenwald: Some Selected Published Comments on PL/I	12
PB2.3.3	Robert F. Rosin: PL/I Macro Processor - Progress Report	14
PB2.3.4	Robert F. Rosin: Macros in PL/I	16
PB2.3.5	W. N. Holmes: Some Remarks on PL/I	22
PB2.8	REFERENCES	24
PB2.9	CIRCULATION LIST	25

The PL/I Bulletin is sponsored by Working Group 4 (WG4) of the Special Interest Group on Programming Languages (SIGPLAN) of the Los Angeles Chapter of the Association for Computing Machinery.

Reproduction of the PL/I Bulletin is being provided by Logicon, Inc., Redondo Beach, California.

Distribution of the PL/I Bulletin is being provided by the Business Equipment Manufacturers Association (BEMA).

All inquiries and contributions should be addressed to:

R. N. Southworth (Editor, PL/I Bulletin)
c/o Logicon, Inc., 205 Avenue I
Redondo Beach, California 90277

PB2.0 EDITOR'S NOTES

PB2.0.1 About the PL/I Bulletin

The PL/I Bulletin is intended to be an informal publication for the interchange of information and opinions relating to PL/I. It is expected that the contents will be primarily of interest to PL/I specialists or language specialists. The contents will be divided into several sections, as indicated by the table of contents on page zero. The section headings for some sections are self-explanatory: "EDITOR'S NOTES", "REFERENCES", "CIRCULATION LIST", "CORRESPONDENCE", and "NEWS ITEMS".

Three sections, however, deserve some special description of the intended content: "WORKING PAPERS", "PROGRAMS" and "PITFALLS & PRAGMATISMS". The WORKING PAPERS section is to be a place for informal publication of technical papers. Presumably, publication in the PL/I Bulletin would not preclude later publication in a journal or magazine. Hopefully, the waiting time for publication in the PL/I Bulletin will not be as long as it is for most journals, so that quick reactions can be got from new concepts. Very little editing will be performed in working papers and, as long as space permits, judgment as to content will not be supercritical. Papers which are obviously intended for sales promotion will be rejected.

The PROGRAMS section is intended to be a place to publish programs and parts of programs written in PL/I language. Contributions may be a group, a block, several blocks or a complete program with input/output statements. It would be appreciated if contributions for the PROGRAMS section can be prepared in accordance with the Publication Language Guidelines published in Section PB1.3.1 of this Bulletin (or later revisions thereof). Editing of programs to be published will be minimal. Selection of programs will be performed by a review board of WG4 members.

The intended contents of the PITFALLS & PRAGMATISMS section is indicated by the title. Discussions of pitfalls will be brief and to the point, and contributions may be heavily edited to maintain continuity of style and format. Pragmatisms will be demonstrations of ways of using PL/I; ways which are novel and unusual as well as practical and useful.

PB2.0.2 About the Sponsor

The purpose of Working Group 4 (WG4) of the Special Interest Group on Programming Languages (SIGPLAN) of the Los Angeles Chapter of the Association for Computing Machinery is twofold:

1) to study, to evaluate and, in other ways, to acquire knowledge relating to the programming language known as PL/I (previously known as NPL),

2) to communicate knowledge relating to PL/I, within SIGPLAN and to other interested persons, by means of meetings, seminars and publications.

This statement of purpose is consistent with the statement of purpose of the L.A. SIGPLAN given in their bylaws: "Article I, Section B, Purpose. The purpose shall be to promote the acquisition and exchange of information on the theory, design, implementation, and application of programming languages in order to educate its members and advance the state of the programming art."

WG4 was conceived in late 1964 and first announced in the SIGPLAN "Notices" in December 1964. The decision to sponsor the PL/I Bulletin was made in October, 1965. The group felt that this was one of the most fruitful ways in which they could fulfill their purpose.

PB2.0.3 Solicited Correspondence

Your editor wrote to a number of computer specialists requesting them to submit their views regarding PL/I and its future for publication. The majority of the responders (for various reasons) preferred not to make any statement. Responses from those who did state their views are presented in the CORRESPONDENCE section with the heading "PL/I and Its Future".

PB2.0.4 About Items PB2.0.1, PB2.0.2 and PB2.0.3

These items of the Editor's Notes were copied from PB1 because PB2 is going to be distributed to many people who did not receive issue Number 1. The requests for copies of PB1 have been considerably greater than the supply.

PB2.0.5 Editor's Apology

We regret the length of time between the publication of PB1 and PB2. The delay was partly due to great expectations that working papers and other materials of great interest would be forthcoming momentarily. We hope that all of you who have been holding back your contributions will immediately submit them.

PB2.0.6 Distribution of the PL/I Bulletin

The distribution of PB1 (and, we hope, PB2) has been through the courtesy of BEMA. There is considerable likelihood that at some point in the future the PL/I Bulletin will be incorporated into a "combined programming languages bulletin" for which a subscription fee will be charged to cover publication and mailing costs. If plans for the combined bulletin are completed before PB3 is ready we will send an announcement to each of those on the PB mailing list.

PB2.1 NEWS ITEMS

PB2.1.1 Correction. R. A. Zemlin of CDC writes us that "...Control Data has made no commitments for PL/I implementations on any machine." This is counter to our news item (PB1.1.2) in PL/I Bulletin No. 1. His letter is reproduced in full in item PB2.2.6.

PB2.1.2 DIGITEK has nearly completed the first PL/I compiler for the GE600 series computers. We hear that it requires only 30,000 words. We will try to get a description of this compiler for PL/I Bulletin No. 3.

PB2.1.3 H Level Compiler. We have heard rumors that IBM's H Level Compiler has been cancelled and that in its place there will be a G Level one which will be an upgraded F Level Compiler.

PB2.1.4 UNIVAC 9000. The announcements of the UNIVAC 9000 series computers make no mention of any PL/I facilities.

PB2.1.5 ASA PL/I Committee. ASA Committee X3.4 has a new sub-committee, X3.4.2C, ad hoc group on PL/I. They have held three meetings: April 14, June 2 and July 27, 1966. The committee has been subdivided into four task groups:

1. Language Development
2. Formal Definition
3. Subsets
4. Character Sets

Each of these groups have been meeting in between the general meetings of the ad hoc group. The scope, program of work, and membership of the committee are given below.

SCOPE:

To study the language, PL/I, with the aim of considering its suitability as a candidate for standardization work.

PROGRAM OF WORK

1. to study PL/I and all relevant reports
2. to propose modifications to the language, as appropriate
3. to propose a precise specification of the language including subsets of the language, as appropriate
4. to report on the suitability of the proposed specifications as a candidate for standardization work

MEMBERSHIP:

Paul Abrahams, Information International
Richard E. Batchelor, Honeywell, Inc.
Frank Bates, Union Carbide Corporation
C. Miller Clarke, RCA-EDP
Frederick C. Cowburn, CSD -- Dept. of the Army

Philip H. Dorn, SHARE, Computer Applications Inc.
Mark Elson, IBM Corporation
W. P. Frink (Secretary), General Electric Company
Myron H. Goldberg, GUIDE, J. C. Penny Company
Marjorie L. Green, Electronic Associates, Inc.
Thomas W. Kern, National Cash Register Co.
Charles Kerpelman, RCA-EDP
Patricia Langendorf, Rome Air Development Center (USAF)
J. A. N. Lee, COMMON User Group, University of Massachusetts
Lawrence Liddiard, CO-OP User Group, University of Minnesota
John L. Little, Bureau of the Budget
Fred M. Magness, Control Data Corporation
Robert W. Mitchell, Computer Associates, Inc.
Elliott C. Nohr (Chairman), IBM Corporation
James Norton, CCC User Group, Grumman Aircraft Corporation
J. T. O'Neill, Headquarters USAF
Stanley H. Park, UNIVAC - Division of Sperry Rand
Robert E. Rountree, Jr., National Bureau of Standards
Maxim G. Smith, Honeywell, Inc.
Conrad H. Weisert, Union Carbide Corporation

PB2.2 CORRESPONDENCE

PB2.2.1 PL/I and Its Future

U.S.S.R.
Novosibirsk 90
Computation Center
December 30, 1965

I do not belong to those who cannot imagine that a general-purpose programming language should require 500 pages for its description, nor to those who chuckle to themselves after having read 3000 ambiguous or unclear statements in a language outline. However, I see that at the present stage of its existence, PL/I presents the programmer with more problems than it solves.

There are two approaches to the design of language systems that embrace the entire spectrum of machine applications -- these are the nucleus language and the shell language. A nucleus language consists of a finite set of basic items and operations, but with a highly developed system for the concatenation of linguistic elements and of structures for hierarchical superpositioning. It is important that the system itself be expressible in terms of the nucleus language.

A shell language has a large stock of descriptors, directly symbolizing all basic operations and items employed in today's problems. It is important that this comprehensive language be easily decomposable, so that any narrow application of this language could be directly satisfied by means of defining a corresponding subset of the shell language.

The nucleus language is the philosophy of the IFIP Working Group on ALGOL, while PL/I is a shell language. Which way is best is a scientific problem that must be solved using scientific methods and, above all, by painstaking work in both directions.

It is difficult for me to predict the future destiny of PL/I, since to do so one need not know PL/I so much as to know the American environment.

A. P. Yershov

PB2.2.2 PL/I and Its Future

System Development Corporation
Santa Monica, California
January 25, 1966

In your letter of 10 December you asked for a statement regarding my views on PL/I. This is, of course, a complex subject and one more worthy of a substantial paper rather than a simple letter. I will try,

however, to provide a brief encapsulation of my feelings about PL/I, its present status and its future prospects. Very briefly put these are:

1. PL/I is a good language but does have some faults such as lack of generality leading to unnecessary complexities.
2. The present status of PL/I is shakey in view of the tardiness of the compilers and the stirring about of BEMA/DPG.
3. Its future is assured in view of the number of people jumping on the band wagon.

Each of these three observations calls for expansion. I will try to elaborate on them soon after I return from a trip to, among other places, Hursley. I would like to be on the PL/I Bulletin mailing list.

Very truly yours,
T. B. Steel, Jr.

PB2.2.3 PL/I and Its Future

Polska Akademia Nauk
Instytut Maszyn Matematycznych
February 3, 1966

We welcome the news about the organization of an information service concerning the PL/I language. The Institute of Mathematical Machines starts with the PL/I language implementation for the ZAM family computers being manufactured in our Institute. We are treating the PL/I language as the concrete basis for the world standard elaboration in the field of programming languages. In a short time we shall probably be able to forward you certain contributions with respect to this language. As yet, we are most interested in getting the PL/I Bulletin copies.

Sincerely yours,
Prof. Dr. Leon Lukaszewicz

PB2.2.4 PL/I Implemented on 7090

Stanford University
Stanford, California
February 11, 1966

Congratulations on a good start with the PL/I Bulletin. I would like to pass on some information that may be of interest to your readers.

The Stanford Computation Center implemented PL/I last summer on the IBM 7090. While our translator-interpreter is not currently available for distribution, interested users can write the Center for a descriptive publication. Briefly, we have implemented the "scientific subset", re-

defining the arithmetic to conform to 7090 integer and floating arithmetic. Recursion is there; structures are not. The interpreter is about 50 times slower than comparable FORTRAN executions; the compiler is correspondingly faster. The entire program is written in 7090 BALGOL and FAP. We warn our users that the translator has no bugs, only a multiplicity of "features".

For our 7090 BALGOL users we have also provided a BALGOL to PL/I source language translator written in Burroughs B5500 ALGOL. This program is available but the Center must charge for any Stanford machine time involved in its use.

Finally, hurrah for your forthright stand demanding a reasonable style in published programs. I would like to refer your readers to my ALGOL 60 reference language editor, Algorithm 268, Comm. ACM8, 11 (November 1965).

Respectfully,
W. M. McKeeman

PB2.2.5 PL/I and Its Future

University of London
Institute of Computer Science
44 Gordon Square, London WC1

The combined effects of illness in the family and a change of job have had a devastating result on my correspondence, and I must apologise for some four months delay in replying to your letter.

Obviously I have already disappointed you as regards a "starting process" contribution. As regards anything else at this stage, I find it a bit difficult. I have just completed a manuscript for a book entitled "A Comparative Study in Programming Languages" in which it has proved convenient to treat CPL and PL/I as contemporaries and to cover them by the "compare and contrast" method in one chapter of some 9000 words. Also, in the context of a comparative study, I have felt it legitimate to refer to all three drafts. In both languages I have had printed sources and lecture sources but no personal contacts. The criticisms and commendations go sometimes one way and sometimes another, often on minute points of detail. One of my new colleagues who was active in designing CPL says I have been very fair; I can only hope that your colleagues will ultimately feel the same! But you will see that it would be a major work to reconsider all my thoughts with a view to reexpressing them as a review of PL/I in isolation; however, I will try to put something on another piece of paper, and if that - or this paragraph - is of use to you, please make use of it.

Yours sincerely,
Bryan Higman

(Editor's Note: Mr Higman's reactions to PL/I are included in the Working Papers section of this Issue. PB2.3.1.)

PB2.2.6 News Item Correction

Control Data Corporation
Palo Alto, California
21 April 1966

On page 3 of PL/I Bulletin No. 1, January 1966, it is stated that: "Control Data Corporation has made definite commitments that it will have PL/I implemented for its 3000 and 6000 series machines."

This statement is incorrect. At the present time, Control Data has made no commitments for PL/I implementations on any machine. We do have a PL/I implementation study group in Palo Alto, but no specific compiler products are presently planned.

We would very much appreciate your correcting this statement about our intentions in the next issue of the Bulletin.

Incidentally, I feel that your publication is a valuable service, and I hope that present efforts to expand the support and distribution of it will be successful.

R. A Zemlin, Director
Advanced Systems

PB2.3 WORKING PAPERS

PB2.3.1 Some Reactions to PL/I

Bryan Higman

(Editor's Note" These notes were received with a letter (see PB2.2.5) in response to a request from the Editor (see PB1.0.3). The response was felt to be worthy of inclusion as a working paper, rather than as correspondence.)

(1) Slight regret, though I realise the pressures from vested interests, at the concession to card codes which ought to be obsolescent.

(2) Wholehearted approval of most of the Objectives. However, (a) a bad programming language is ipso facto a bad algorithmic language, so I find Objective 6 meaningless, and (b) continuous regret that Objective 1 seems so often to receive only lip-service (examples later).

(3) The limit of 31 alphanumerics on an identifier will worry nobody (except the Germans?) but it is the first example of going back on Objective 1. Also a pity that RATE_OF_PAY is not available in the 48-character set.

(4) Wholehearted approval of the general syntactical principle of attributes and options.

(5) Default precision is presumably machine precision and efficiency is improved thereby. This being so, there is a need for either double precision, or a means of making an initial enquiry about machine precision.

(6) There are advantages and disadvantages in using a bit-variable with multiplication instead of a logical variable; I think the latter have been underestimated. The thought that go to if B1 then L1 else if B2 then L2 else L3 might be "optimised" into

```
GO TO L3 + (L1-L3)*B1 + (L2-L3)*B2 + (L3-L2)*B1*B2
```

and then used with B1 true and B2, L2, L3 unallocated, gives me the willies.

There are two distinct points here, of course, use with labels and use with unallocated booleans guarded by a true from being looked at.

(7) The issue of calling by "name" or "value" is one which we cannot yet say we have mastered. The PL/I solution is interesting in determining call type at call time, but, in my opinion, ducks the real issue in a deplorably retrogressive way.

(8) STATIC, AUTOMATIC, CONTROLLED and DEFINED are really four states of one and the same attribute, and should be so defined. This is an excellent feature, and it is obvious on looking at all three reports that a lot of thought has gone into it. But DEFINED wants a lot more thought still. Objective 1 demands that it shall not be restricted to linear forms in subscripts. Among the possibilities opened up by this are (a) as in Report 1

```
DECLARE A(M,N) DEFINED X((1SUB*(1SUB-1))/2 + 2SUB)
```

or (b)

```
DECLARE C(P,Q) DEFINED A(1SUB)*B(2SUB)
```

or (c) as in the next comment - in fact almost anything covered by CPL's "Initialisation by substitution".

(9) Lists are inadequately provided for. Structures are too rigid, and pushdown stores expose only their tops. The following, while open to abuse, allows one to search a list

```
      DECLARE X DEFINED DYNAMICALLY
        :
      DEFINE X AS Y
LOOP: FREE X
        :
```

(10) One thing I like about the "attribute" principle is the way it overcomes some "silly" problems - I am thinking, for example, of BUILTIN. But a thought here - why not make all labels implicitly part of a structure? Then variables which are "inaccessible" because the name has been redeclared can be made accessible by qualifying them with the block label - e.g.

```
K: BEGIN; DECLARE X;
   X = K.X                    (or X in K (COBOL))
```

The ALGOL switch S := A, if B then C else D, Y[n] seems to me to call for

```
DECLARE S (3) LABEL INITIAL (A, E, F)
DECLARE E DEFINED B*C + (NOT B)*D
DECLARE F DEFINED T(N)
```

which re-raises issues already mentioned.

(12) It is good to see a "modern" language which does as well as COBOL on "emergencies" but PL/I does better by adapting the same techniques to include parallel processing (possibly the only language so far to do so except for ones of only local interest). A good word, too, for the macro facility, though it will take time to assess this.

(13) Input/Output. Here I am a heretic, for I hold that this is a language-interface problem, not to be dealt with by languages acting independently. On this basis the ACM ALGOL proposals come out better because of one thing; if a common format system for all languages is to be laid down, then each language must accept it in the form of a string (protected from the language's private syntax by string quotes) - and the ACM ALGOL does this whereas the PL/I format statement does not.

PB2.3.2 Some Selected Published Comments on PL/I

I. D. Greenwald
24 February 1966

In what follows, the reader should be aware that the comments were often addressed to different documents, ranging from the first report of the SHARE Advanced Language Development Committee to the most current IBM publications.

I have attempted to quote those phrases which convey my impression of what the writer's basic attitude encompassed. If I have misinterpreted, I apologize.

The letters (1) contained in PL/I Bulletin #1 will not be excerpted here.

Galler states: "It is clear ... that NPL should not even be considered as a new 'standard' ... until attempts at implementation have shown up the weak spots ... and a better integration of the new features is accomplished" (2). McCracken concludes that: "... NPL is a well designed language" (2) and makes even stronger statements vis-a-vis FORTRAN in (4). Rosen finds Report II of the SHARE Advanced Language Development Committee "... an unsatisfying document" (2) stating that; "It contains too much -- too loosely specified" and "... It will be difficult and expensive to implement ... use ... and ... teach and learn." Wagner's experience is in direct disagreement with respect to the learning (2). He also states that to an installation manager .., "... it is very attractive to concentrate on one primary programming language which can be used, with an acceptable degree of success, for almost any computing purpose."

Ershow sees "... that at the present stage of its existence, PL/I presents the programmer with more questions that it gives answers" (3) and goes on to state: "It is difficult for me to predict the fate of PL/I, since to do so one need not know PL/I so much as to know the American environment." Smith states: "... the committee ...produced ... a patch-worked palliative rather than a long overdue cure." (5) According to Knuth, "It is apparent that NPL will become the most popularly used programming language in the years to cime, and I comment the designers on their fine job." (6)

Most of the authors seem to agree on one point: The weight of IBM/SHARE is almost certain to be the determining factor in the final fate of PL/I.

Bibliography of Comments on PL/I

1. PL/I Bulletin #1, January 1966

D. Knuth
M. Halpern
P. Z. Ingerman

J. W. Carr, III
A. d'Agapeyeff
H. Bromberg

2. Computing Reviews, Vol. 6, #2, March-April 1965

pp. 108-112
B. A. Galler
D. B. McCracken
S. Rosen
F. V. Wagner

3. A. P. Ershov: (Translation from enclosure to a letter.)

4. D. D. McCracken: "The New Programming Language," Datamation, July 1964, p. 31

5. J. W. Smith: Letter to SHARE Advanced Language Development Committee, 24 March 1964

6. D. E. Knuth: Letter to J. Cox, Director of NPL Language Control, 30 January 1966.

PB2.3.3 PL/I Macro Preprocessor - Progress Report

Robert F. Rosin
25 March 1966

This note describes the first attempt at a PL/I macro preprocessor implemented in SNOBOL. It is in no way complete, nor does it include all of the facilities in the SHARE XXVI memo. It is expected that the next step will be to define some rather flexible string manipulation operators for PL/I, rewrite the preprocessor in this extended PL/I and then process the definitions and the newly written preprocessor through the SNOBOL program. The result would be a preprocessor written in "dash 2" PL/I. It is expected that this will actually be an iterative process, since deficiencies in the existing preprocessor will be discovered as the bootstrapping is attempted.

// DEFINE $mode_r$ = $mode_a$, operator, $mode_b$, .*

Note: The 48 character-set instructions are used throughout. The mode information is presently used only to establish whether the macro is unary, binary and yields a result. It will be used later for full context analysis.

```
// PRECEDENCE rltn existingop,.
             (definition)
// END ,.
```

"mode$_r$ =" is optional

"mode$_a$ " is optional

", mode$_b$ " is optional

"rltn" must be either -, GT or LT

"existingop" must be a built-in or previously defined macro, operator, or keyword

In the definition: R is the pseudonym of the results

A is the pseudonym of the left-hand operand

B is the pseudonym of the right-hand operand

.L is the prefix of any local variable name

Examples:

```
//DEFINE , UNTIL, BIT, . //PRECEDENCE = WHILE, .
           WHILE NOT ( B ) , . / /END , .

//DEFINE , ORELSE, . / /PRECEDENCE = IF, .
           END, . ELSE DO, . //END,

//DEFINE CHARACTER = CHARACTER, FXLN, FIXED, .
// PRECEDENCE GT CAT, .
           R = A , . IF LENGHT( R ) GT B THEN
               R = SUBSTR( R , 1, B ) , .
                   ELSE IF LENGTH( R ) LT B THEN
               DO . L1 = LENGTH( R ) +1 TO B , .
                   R = R CAT ' ', . END, .
// END, .
```

Note that operands and local variables must be set off from other elements by at least one blank

Macro Variables:

```
// var = expn, .
```

causes expn to be substituted for var in all succeeding statements. var must be set off from other elements in these statements by at least one blank.

```
// DISABLE var, .
```

deletes the effect of any preceding establishment of a macro variable.

All macros are expanded in any statement before macro variables are replaced.

Replacement and definition can be recursive. USER BEWARE.

PB2.3.4 Macros in PL/I

Robert F. Rosin
28 February 1966

This proposal is based on the premise that the use of the term "macro" in the PL/I SRL is not accurate. A Macro, as used in the sense of FAP, MAP, etc,. is used to create open subroutines, and, therefore, to add new operators to the language. (See McIlroy, CACM April, 1960). As far as is known to this author, MAD is the only algebraic-algorithmic language containing such facilities, and much of this proposal is based on experience with MAD.

The Use of Macros in an Algebraic Language

A simple example arises in the development of the relational operator in MAD for comparing strings to determine whether one is greater than the other in terms of a particular BCD collating sequence. Such an operator (see Appendix I) was developed in the past year as part of a class problem by students in an introductory programming course. The operands are two 6-character 7094 words. The result is a Boolean value, true or false. Once the operator is available, manipulation of multiword BCD data is trivial in MAD.

A more powerful use has been made of macro facilities in MAD in the incorporation of operators to perform ammy of the string manipulation features of SNOBOL. For example, one can write such statements as:

```
A .ST. B .C. C * $T$
C .ST X / 3          .FAIL. STMT1
B .ST. $,$ .C. RMD .C. $,$ .EQ. $,$
```

which can be interpreted to mean:

In the string A, scan for an occurance of the string B, concatenated with a string whose value replaces C, concatenated with the literal T.

In the string C, scan for a string of length three whose value replaces X. In the scan fails (i,e,, there are fewer than three characters in C) then go to STMT1.

In the string B, scan for an occurance of the literal ",", concatenated with a string whose value is the same as the value of BMD.

concatenated with a second ",". Replace the substring so matched with a single literal ",".

It should be noted that the macro definition of the operator .ST. includes a call to a closed subroutine, in effect an interpreter, in this particular implementation.

In contrast, consider the string generic functions in PL/I. With one exception, SUBSTR, these are all functions of one or two operands and, thus, quite amenable to operator-operand notation. SUBSTR and INDEX are probably the most useful of these functions since they are most directly related to string transformation. Their use in combination to perform complex manipulation results in either a large number of simple function calls or an unreadable set of nested function calls. A macro facility in PL/I would allow the development of a more complete and usable set of string manipulation operators in that language.

Outlines for a General Macro Facility

In the definition of any subroutine, open or closed, several specifications are necessary:

1. Name
2. Dummy arguments and result
3. Physical beginning
4. Physical end
5. Entry points
6. Return to calling program or logical termination
7. The value of the routine

In defining an operator (macro) in an algebraic language, several other characteristics must be known:

8. The arithmetic mode or type of the operands
9. The arithmetic precedence of the operator with respect to existing operators
10. The numbers of operands; i.e., whether the operator is unary or binary
11. The mode or type of the result

Furthermore, in the context of an algebraic language, additional facilities are needed or desired:

12. Conditional compilation on the basis of newly defined tests
13. Ability to redefine existing or built-in operators and macros
14. Ability to define production of code for unlikely operators; e.g., ",," "(" ")" and right and left terminators of a statement
15. Ability to generate local variables and labels
16. Ability to define new keywords or redefine existing ones
17. Ability to assign different arithmetic precedences to a given operator depending on the modes of its operands
18. The ability to append new modes to the language

Macro Facilities in PL/I

Although there are valid agruments to the contrary, it seems reasonable to allow only PL/I statements in PL/I macro facilities for the important considerations of machine independence and compatibility with other implementations. (An example of a simple MAD defined operator which depends on a pseudo assembly language for definition is given in Appendix I.)

The names of operators should include existing operators, single characters which will not be interpreted ambiguously, such as "?", and a set of alphabetics and/or numerics preceded and followed by an established escape character. An example of the latter case might be the use of "." as an excape character in naming an operator ".RTSHFT.".

In conjunction with the compile time facilities specified in the PL/I SRL and the Elson paper, the % notation is used for macro definition. Two new compile time statements are needed. They are %DEFINE and %CONTEXT. The former is qualified by such keywords as BINARY, UNARY, PRECEDENCE, OPERATOR, KEYWORD, etc. and appears in the contexts

$$\%\text{DEFINE}\ \left\{\begin{matrix}\text{UNARY}\\ \text{BINARY}\end{matrix}\right\}\ \text{OPERATOR op , PRECEDENCE}\ \left\{\begin{matrix}>\\ =\\ <\end{matrix}\right\}\ \text{op}'$$

%DEFINE KEYWORD word

%CONTEXT mode = mode op mode

Appendix II contains an example of a possible use of a macro facility in PL/I. This makes use of all items in the list of desired capabilities except numbers 13, 16 and 17. Appendix III contains an example using item 12.

Conclusions

It is apparent that a facility as described above has proven useful in the pase, and would probably be of value in the future. It gives the sophisticated programmer the ability to redefine the compiler he is using without the necessity of rebuilding the compiler itself. It also gives him the ability to enrich the language with which he is working. It could result in new and powerful capabilities in some future version of PL/I, (PL/n?).

If such facilities can be built using macros in a language and processor which dates back 6 years, surely the same useful capabilities should be made available in today's latest languages. The resultant ability of the programmer to further develop and enrich the language is important and powerful. On the other hand, it is likely that this facility falls into the same category as the PL/I list processing facilities, i.e., not of interest to the casual user of PL/I.

APPENDIX I

The definition of the operator .GC. in MAD

Given two operands, each one 6 character 7094 word of BCD (integer) mode, one might like to compare them with respect to the defined BCD collating sequence. Use of the operator .G. (comparable to .GT. in FORTRAN) results in the anomaly that initial characters J through Z appear to be negative, e.g., "less than" a, due to the internal coding of the BCD characters. The following defined operator allows one to write such statements as:

```
WHENEVER X .GC. Y .OR. L .GC. $ABC$
DEFINE BINARY OPERATOR .GC., PRECEDENCE SAME AS .G.
MODE STRUCTURE 2 = 1 .GC. 1
```

```
JMP     *+1, LA  *+4
JMP     *+9, AT, *+1
SLW     T
JMP     *+6
JMP     *+1, AC *+3
STO     T
JMP     *+3
JMP     *+1, MQ, *+2
STQ     T
CAL     A
LAS     B
TRA     LOC+4
NOP
PXD     0, 0
TRA     LOC+2
CLA     = 1
OUT     AC
END
```

The digits 1 and 2 in the MODE STRUCTURE statement indicate that the mode of the operands is integer and the mode of the result is Boolean. Three pseudo-ops are used in this sequence: JMP which initiates a conditional or unconditional jump within sequence; OUT which indicates the logical end of the sequence; END which indicates the physical end of the sequence. Locations denoted as *+i are compile time addresses. Locations denoted as LOC+i are object code addresses. A and B are the left and right operands of .GC. T is a pseudo name for the next available temporary storage location. LA, AT, BT, AC, and MQ indicate tests which are true is the result of the previous operations is:

in the logical accumulator,

the A operand of this operator,

the B operand of this operator,

in the AC or in the MQ, respectively

Therefore, JMP *+1, AC, *+2 will result in the transferring to the next command if the result of the previous operation is in the accumulator. Otherwise, control is transferred to two instructions beyond this one.

APPENDIX II

Definition of an Operator in PL/I

It might be convenient to produce a fixed length string based on a PL/I variable. For example, if A is a character string, then

```
X = A .FXLN. 4
```

should result in X containing the first four characters of A, or, if A is not four or more characters long, X should contain the value of A concatenated with sufficient blanks to yield four characters. The existence of this operator would allow one to generate a card image in FORTRAN format by writing:

```
CARD = LABEL .FXLN. 6 || ' ' || STMT .FXLN. 66
```

The definition which follows is based on the use of % to indicate a compile-time statement as in existing PL/I notation, A and B to represent the left and right operands and R to represent the resulting value. Local variables are denoted by L_i

```
% DEFINE BINARY OPERATOR .FXLN.,
% PRECEDENCE = REAL**REAL
% CONTEXT CHARACTER = CHARACTER .FXLN. FIXED;
    R = A
    IF LENGTH(A) ¬= B THEN DO;
        DO L1 = LENGTH(A) + 1 TO B;
            R = R || ' '; END;
        ELSE
            R = SUBSTR(R, 1, B);
        END;
% END
```

A more powerful definition, taking advantage of defaults and compile time conditionals is:

```
%  DEFINE BINARY OPERATOR .FXLN., PRECEDENCE = **,
%  CONTEXT CHARACTER | BIT= CHARACTER | BIT
%  .FXLN. FIXED | REAL;
     R = A
     IF LENGTH(A)- = B THEN DO;
          IF LENGTH(A) < B THEN
               DO L1 = LENGTH(A) + 1 TO B;
%    IF MODE(A) = 'BIT' THEN
               R = R || '0'B  END;
%    ELSE
               R = R || ' '; END;
          ELSE
               R = SUBSTR(R, 1, B)
%  END
```

Note the use of the test function MODE.

Definition of a New Keyword in PL/I

```
%  DEFINE KEYWORD 'UNTIL' ;
%  CONTEXT UNTIL BIT;
          WHILE-(B)
%  END
```

Example of the use of UNTIL

```
DO UNTIL X < 4 OR A(I) = 2;
```

PB2.3.5 Some Remarks on PL/I

W. N. Holmes

a) A single semi-colon should be a definite end to a statement even within a comment or a character string constant (where it could be represented by a double semi-colon). This seems a quite practical way to minimize the effect of inadvertant omission of terminal delimiters.

b) Comparison operations. The operations = and ¬= should be defined when both operands are labels.

c) Bit-String operations. The operations ¬, & and / should not be interpreted in the restricted sense of PL/I as at present. The following meanings would be more useful and appropriate:

¬ := The operation which results in the successor to its operand

& := The operation which results in the minimum of its two operands

/ := The operation which results in the maximum of its two operands

These operations would apply bit by bit or character by character for strings and the extended meanings are a more or less compatible extension. However, a predecessor operation becomes desirable and certain built-in functions become unnecessary.

d) UNTIL should be provided as the inverse of WHILE.

e) UNLESS should be provided as the inverse of IF.

f) COME FROM should be provided as the inverse of GO TO. This facility would be particularly useful in simplifying program modification particularly when testing.

g) The comment delimiters are very poorly chosen and will discourage the use of comments. A better delimiter, used at either end of a comment, would be two or more consecutive hyphens (minus signs) and the programmer then writes a long dash. The use otherwise of double minusses would be no real loss. An alternative is ≪comment≫. Comments should be allowed in data- and list-directed input data.

h) INITIAL attribute. Constant expressions should be allowed.

i) Additional variable types.

   (i) NAME variables - similar to LABEL variables except used for addressing data. This would allow, for example, indirect addressing to any depth.

   (ii) OPERATOR variables - This would allow programming techniques similar to those enjoyed at the machine level by virtue of the EXECUTE instruction - however this is perhaps too perturbing for the language.

j) The rather specific capabilities provided by SAVE and RESTORE statements should be generalized. One way of doing this would be to allow the attribute STACK to be declared for any variable, with fairly obvious and simply invoked effect. STACK should be qualifiable by such adjectives as:-

   (i) FIFO and LIFO to specify the mode of stacking operation;

   (ii) SIZE (m, n) to specify the size of stack required in terms of m primary storage slots and n secondary storage slots - provision of EMPTY and FULL ON-conditions would be necessary and perhaps a SIZE built-in function.

The appearance of a STACK variable on the left-hand side of an assignment statement '=' would ask for storing of the assignment value (or values) into the named STACK, otherwise the appearance of a STACK variable on the right-hand side of an assignment statement '=' would ask for the use and removal of the appropriate value from the stack.

k) The problems associated with groupings of the units of the IF statement could be simply solved by allowing the alternative general format:-

IF scalar-expression THEN [(unit-1)]/[unit-1]
[ELSE[(unit-2)]/[unit-2]]
that is, by allowing either or both of the IF units to be enclosed in parentheses. If either is in parentheses then a terminating semi-colon for the IF statement may be required.

1) The provisions for complex arithmetic seem a little restricted - electrical engineers at least would appreciate a polar representation. It could be possible to secondarily declare associations of named variables (and possibly constants) independently declared with the particular association (complex, polar etc.) defined for certain operations and built-in functions. This would render certain complex built-in functions unnecessary. Further, the programmer might be allowed to define his own associations and hence could, for example, compute with quaternions.

PB2.8 REFERENCES

PB2.8.1"PL/I: Language Specifications, IBM Operating System/360" Form No. C28-6571-2, (January 1966) International Business Machines Corporation.

PB2.8.2"Tentative Steps Toward a Formal Definition of Semantics of PL/I", K Bandat (Ed.) Technical Report TR25.056, (July 1965) IBM Laboratory Vienna.

PB2.8.3"Syntax of PL/I" H. Chladek, V. Kudielka, E. Neuhold, Technical Report TR25.058, (18 September 1965) IBM Laboratory Vienna.

PB2.8.4"The Structure of the Programming Language PL/I", W.H. Burge, (July 1965) Univac Division of Sperry Rand Corporation, New York.

PB2.8.5"On the Formalization of Syntax and Semantics of PL/I", P. Lucas, Technical Report TR25.060, (November 1965) IBM Laboratory Vienna.

PB2.8.6"A PL/I Primer", Form No. C28-6808-0, International Business Machines Corporation.

PBB.9 CIRCULATION LIST

SEYMOUR ALTSCHER
NATIONAL DAIRY PRODUCTS CORP
260 MADISON AVENUE
NEW YORK, NEW YORK 10016

M. D. BARNEY
UNIVERSITY OF LONDON
EXHIBITION ROAD
LONDON S.W.7 ENGLAND

KARL BALKE
COMPUTER SCIENCES CORPORATION
650 N. SEPULVEDA BOULEVARD
EL SEGUNDO, CALIFORNIA 90245

W. S. BARTLETT
COMPUTING SCIENCES DEPARTMENT
BELL TELEPHONE LABORATORIES
WHIPPANY, NEW JERSEY 07 81

DR. HANS BERIC
IBM LABORATORY VIENNA
PARKRING 10
1010 VIENNA/AUSTRIA

REX C. BIGAZ
AMERICAN CEMENT CORP.
2404 WILSHIRE BLVD.
LOS ANGELES, CAL. 90057

R. BRADEN
COMPUTATION CENTER
STANFORD, CALIFORNIA 94305

W. R. BRITTENHAM
IBM DEPT. 076/BLDG. 706
P.O. BOX 390
POUGHKEEPSIE, NEW YORK 12602

LEE BUCHOLZ
351 GRENOLA
PACIFIC PALISADES, CALIF. 90272

THEODORE G. CAMERON
SYSTEMS PROG., BLDG. 204-2
RCA, EDP
CHERRY HILL, NEW JERSEY 08034

P. S. CASTILLO, JR.
CONTROL DATA CORPORATION
8100 34TH AVENUE SOUTH
MINNEAPOLIS 20, MINNESOTA

ANSON E CHAPMAN
NAA DEPT 200-001 EA18
12214 LAKEWOOD BOULEVARD
DOWNEY, CALIFORNIA

DOAN COMBELIC
ITT DATA SYSTEMS
1 RUE RABELAIS
PARIS 8, FRANCE

P. DES JARDINS
GATEWAY EAST, SUITE 468
1800 AVENUE OF THE STARS
LOS ANGELES, CALIF. 90067

MICHEAL A. DUGGAN
5319 BROOKS HILL ROAD
BETHESDA, MARYLAND 20014

M. ELSEN
IBM
1271 AVENUE OF THE AMERICAS
NEW YORK, N.Y.

JAMES G.
4558 NORTH HAMONA
LOS ANGELES, CALIFORNIA 90065

PHILIP R. BAGLEY
115 MONTGOMERY AVENUE
BALA CYNWYD, PENNA. 19004

FRANK K. BANNERGER
NORC, UNIVERSITY OF CHICAGO
5720 WOODLAWN AVENUE
CHICAGO 37, ILLINOIS

FRANK BATES
COMPUTER APPLICATIONS INC.
555 MADISON AVENUE
NEW YORK 22, NEW YORK

JIM BENTSON
IBM FEDERAL SYSTEMS DIVISION
16915 EL CAMINO REAL
HOUSTON, TEXAS 77058

GARY T. BOSWELL
1200 N. ALMA ROAD
BUILDING 425-003
RICHARDSON, TEXAS

DR C B BRAGASSA
MEDICAL COLLEGE OF GEORGIA
EUGENE TALMADGE MEMORIAL HOSPITAL
AUGUSTA, GEORGIA 30902

E R BROADWIN
HONEYWELL EDP
60 WALNUT STREET
WELLESLEY HILL, MASS

T. D. BUETTELL
INFORMATICS INC.
5340 VAN NUYS BOULEVARD
SHERMAN OAKS, CALIFORNIA 91401

DR. D. G. CANTOR
MATHEMATICS DEPT.
UCLA
LOS ANGELES, CAL. 90024

EUGENE CEA
SPERRY GYROSCOPE COMPANY
GREAT NECK, NEW YORK 11020
MAIL STATION 3S105

C. CHRISTENSEN
RCA - EDP
BUILDING 204-2, CHERRY HILL
CAMDEN, NEW JERSEY

MRS. LAURA C. COLSINS
COMPUTER APPLICATIONS INC.
ONE NORTH 13TH STREET
PHILADELPHIA, PENNSYLVANIA

PHILLIP DEUEL
U C COMPUTER CENTER
201 CAMPBELL HALL
BERKELEY, CALIFORNIA 94720

F. G. DUNCAN, EDITOR, ALGOL
3 ELLES AVENUE
MERROW, GUILDFORD
SURREY, ENGLAND

CHARLES F ERICKSON
22045 PORTINE AVENUE
TORRANCE, CALIFORNIA 90502

J. BABCOCK
GATEWAY EAST, SUITE 468
1800 AVENUE OF THE STARS
LOS ANGELES, CALIF. 90067

JOHN T. BAGWELL, JR.
170 ENGINEERING RESEARCH LAB
UNIVERSITY OF ILLINOIS
URBANA, ILLINOIS 61803

A. T. BARRETT
1019 SAN RAMON AVENUE, SE
HUNTSVILLE, ALABAMA

J. P. BEERMAN
CCC USERS GROUP
205 JEFFERSON STREET
CENTERPORT, NEW YORK

JOSEPH BEX
13 ELLERY STREET
CAMBRIDGE 38, MASSACHUSETTS

CONRAD P. BRACKLEIN
CONTROL DATA CORPORATION
3260 HILLVIEW AVENUE
PALO ALTO, CALIFORNIA 94300

NORMAN BRENNER
4 FROST STREET
CAMBRIDGE, MASSACHUSETTS 02140

HOWARD BROMBERG
C-E-I-R INC.
1200 JEFFERSON DAVIS HIGHWAY
ARLINGTON VA. 22202

TECHNICAL LIBRARY
BURROUGHS CORPORATION
460 SIERRA MADRE VILLA
PASADENA, CALIFORNIA 91107

W. P. CASH
CONTROL DATA CORPORATION
3145 PORTER DRIVE
PALO ALTO, CALIFORNIA

CAROL L. CHAN
APT. B-19
2024 COMMONWEALTH AVENUE
ST. PAUL, MINNESOTA 55108

R. F. CLIPPINGER
HONEYWELL INC.
60 WALNUT STREET
WELLESLEY HILLS, MASS. 02181

V. DELLA VALLE
UNIVAC
PLANT II, P.O. BOX 500
BLUE BELL, PENNSYLVANIA 19422

T. J. DOWD
SERVICE BUREAU CORPORATION
2450 WATSON COURT
PALO ALTO, CALIFORNIA 94303

ROBERT A. ELLIS
WASHINGTON UNIVERSITY
700 SOUTH EUCLID AVENUE
ST. LOUIS, MISSOURI 63110

ARTHUR EVANS, JR.
ROOM 5194
545 TECHNOLOGY SQUARE
CAMBRIDGE, MASSACHUSETTS 02139

B. R. FADEN D/692 E13
NORTH AMERICAN AVIATION
1700 EAST IMPERIAL HIGHWAY
EL SEGUNDO, CALIFORNIA 90245

MISS MARGARET R. FOX
TECHNICAL INFORMATION EXCHANGE
CENTER FOR COMP SCI AND TECH - NBS
WASHINGTON, D.C. 20234

M. GIBSON
UNILEVER RESEARCH LABORATORY
PORT SUNLIGHT, CHESHIRE
ENGLAND

R. C. GONZALEZ
LOCKHEED-CALIFORNIA CO.
BURBANK, CALIFORNIA
DEPT 74-54, BLDG 67, PLANT A-1

MRS. MARJORIE GREEN
7 PENFROW DRIVE
TRENTON, NEW JERSEY 08618

M. C. HARRISON
COURANT INSTITUTE OF MATH. SCIENCE
251 MERCER STREET
NEW YORK, N.Y.

BRYAN HIGMAN
INSTITUTE OF COMPUTER SCIENCE
UNIV. OF LONDON, 44 GORDON SQ.
LONDON WC1, ENGLAND

DR. HANS-JUERGEN HOFFMAN
MOERCHBERG 15
703 BOEBLINGEN
GERMANY

DR. GRACE MURRAY HOPPER
UNIVAC
1624 LOCUST ST.
PHILADELPHIA, PA. 19103

MRS. RUTH ISCOL
UNIV. OF CALIFORNIA, BERKELEY
COMPUTER CENTER
BERKELEY, CALIFORNIA

DR. GEORGE KIRBY
AEROSPACE CORP.
P.O. BOX 95085
LOS ANGELES, CAL. 90045

R. C. KURZ
NATIONAL CASH REGISTER
DAYTON, OHIO 45409

L. T. LEPINE
PHILCO CORPORATION
1002 GEMINI AVENUE
HOUSTON, TEXAS 77058

A. C. M. LEWIS
PURDUE UNIVERSITY
SCHOOL OF CIVIL ENGINEERING
LAFAYETTE, INDIANA 47907

GEORGE W. LOOSEMAN
NEW YORK UNIVERSITY
UNIVERSITY HEIGHTS
NEW YORK, NEW YORK 10453

ROBERT L MCALLESTER
COMPUTER USAGE DEVELOPMENT CORP
3181 PORTER DRIVE
PALO ALTO, CALIFORNIA 94304

LIBRARIAN
FEDERAL RESERVE BANK OF CHICAGO
CHICAGO 90, ILL.

ALAN L. GASTON
LAZARUS
COLUMBUS, OHIO 43216

JOHN M. GILBERT
INFORMATION MANAGEMENT CORP.
432 CLAY STREET
SAN FRANCISCO, CALIFORNIA 94111

P. B. GOODSTAT
BEMA/OPG
235 EAST 42ND STREET
NEW YORK, N.Y. 10017

ARMIN F. GUMERMAN
NORTHWESTERN MUTUAL LIFE
720 EAST WISCONSIN AVENUE
MILWAUKEE, WISCONSIN 53202

LEWIS HART
SCIENTIFIC DATA SYSTEMS
2526 BROADWAY
SANTA MONICA, CAL. 90406

RICHARD F. HILL
INFORMATICS INC.
5430 VAN NUYS BOULEVARD
SHERMAN OAKS, CAL. 91401

R. E. HOFFMAN
G E INFORMATION SYSTEMS DIVISION
2621 VICTORY PARKWAY
CINCINNATI, OHIO 45206

EDWARD O. HOWELL
HONEYWELL INC.
315 NORTH VALLEY HILL ROAD
WOODSTOCK, ILLINOIS 60098

EUGENE H. JACOBS
SYSTEM DEVELOPMENT CORP.
2500 COLORADO AVENUE
SANTA MONICA, CALIFORNIA 90406

CAREL O. KNUDSEN, C1-263
DOUGLAS AIRCRAFT CO., INC.
3855 LAKEWOOD BOULEVARD
LONG BEACH, CALIFORNIA 90801

HOWARD LANG
1564 1/2 S. BUNDY DRIVE
LOS ANGELES, CAL. 90025

WALTER E LESEMAN
1461 N. EVERGREEN STREET
BURBANK, CALIFORNIA 91505

A. N. LIPPITT
UNIVAC
P. O. BOX 500
BLUE BELL, PENNSYLVANIA

JOHN FRANCIS LUBIN
COMPUTER CENTER
UNIVERSITY OF PENNSYLVANIA
PHILADELPHIA 4, PA.

M. M. MORGEMAN
COMPUTER SCIENCE DEPARTMENT
STANFORD, CALIFORNIA 94305

DANIEL FLANAGAN
7921 GATEWAY BOULEVARD
WASHINGTON, D.C. 20028

D. M. GEORGE
MONROE INTERNATIONAL, INC.
550 CENTRAL AVENUE
ORANGE, NEW JERSEY 07051

J. GODFREY
GENERAL ELECTRIC CO.
BLACK CANYON HIGHWAY
PHOENIX, ARIZONA

D. T. GOODWIN
7 THE CROFT
PENKHULL STOKE-ON-TRENT
ENGLAND

JOHN L. HALL
MESTA MACHINE COMPANY
P.O. BOX 1466
PITTSBURGH, PA. 15230

DAVID R. HEYM
CHEMICAL ABSTRACTS SERVICE
THE OHIO STATE UNIVERSITY
COLUMBUS, OHIO 43210

JOHN R. HILLEGASS, EDITOR
AUERBACH CORPORATION
1634 ARCH STREET
PHILADELPHIA, PENNSYLVANIA 19103

WILLIAM F HOLMES
WASHINGTON UNIVERSITY
700 S. EUCLID AVENUE
ST. LOUIS, MISSOURI 63110

INFORMATICS INC.
5430 VAN NUYS BOULEVARD
SHERMAN OAKS, CALIFORNIA 91401
ATTN LIBRARIAN

R. R. KENYON, DEPT. 75, BLDG 32
MCDONNELL AUTOMATION CENTER
P. O. BOX 516
ST. LOUIS, MISSOURI 63166

DR. VIKTOR KUDIELKA
IBM LABORATORY VIENNA
PARKRING 10
WIEN 1, AUSTRIA

DR. J. C. LASKI
ATLAS COMPUTER LABORATORY
CHILTON, DIDCOT
BERKSHIRE, ENGLAND

LEONARD LEVINE G178
STANFORD RESEARCH INSTITUTE
MENLO PARK, CALIFORNIA 94025

J. L. LITTLE
EXECUTIVE OFFICE BUILDING - RM 55
WASHINGTON, D.C.

DR. LECH LUKASZEWICZ
INSTYTUT MASZYN MATEMATYCZNYCH
WARSZAWA 2, UL KOSIYKOWA 79
POLAND

J. G. MACKINNEY
RADIO CORPORATION OF AMERICA
8500 BALBOA BOULEVARD
VAN NUYS, CALIFORNIA

BRAD MCKENZIE
BURROUGHS CORPORATION
460 SIERRA MADRE VILLA
PASADENA, CALIFORNIA

STUART E. MADNICK
MASS. INSTITUTE OF TECHNOLOGY
52 MASSACHUSETTS AVENUE
CAMBRIDGE, MASSACHUSETTS 02139

T. W. MARTIN
SCIENTIFIC DATA SYSTEMS
2426 BROADWAY
SANTA MONICA, CALIFORNIA

WARREN E. MEYER
SYSTEM DEVELOPMENT CORPORATION
2500 COLORADO AVENUE
SANTA MONICA, CALIFORNIA 90406

BOB MOORE
DIGITEK CORPORATION
6151 WEST CENTURY BOULEVARD
LOS ANGELES, CALIFORNIA 90045

E. C. MOHR
IBM
ARMONK, NEW YORK 10504

DAVE OPPENHEIM
12754 PACIFIC APT. 1
LOS ANGELES, CAL. 90066

LAURENCE N. REED
RCA CORP. ECP
BUILDING 204-2 RAY 31
CHERRY HILL, NEW JERSEY

ROBERT R. RITZ
1315 GLADWICK STREET
COMPTON, CALIFORNIA

RONALD G. ROSS
UNION CARBIDE CANADA LTD.
123 EGLINTON AVE. EAST
TORONTO 12, CANADA

DANIEL SABSAY
UCLA METEOROLOGY DEPT
405 HILGARD AVENUE
LOS ANGELES, CALIFORNIA 90024

HEDLEY G. SAVILLE
DE HAVILLAND AIRCRAFT OF CANADA
DOWNSVIEW, ONTARIO
CANADA

D. V. SCHORRE
SYSTEM DEVELOPMENT CORP.
2500 COLORADO AVENUE
SANTA MONICA, CALIFORNIA 90406

RICHARD J. SCOTT
5728 CORNING AVENUE
LOS ANGELES, CALIFORNIA 90056

J. C. SHAW
THE RAND CORPORATION
1700 MAIN STREET
SANTA MONICA, CALIFORNIA

BRUCE D. SHRIVER, SR.
WILLARD D. SHRIVER COMPANY
9221 E. WHITMORE STREET
ROSEMEAD, CALIFORNIA 91771

W. S. MACLEAN
UNION CARBIDE CORPORATION
270 PARK AVENUE
NEW YORK, NEW YORK 10017

BARRY J. MAILLOUX
MATHEMATISCH CENTRUM
2E BOERHAAVESTRAAT 49
AMSTERDAM 101, NETHERLANDS

FRANK B MEEK
467 LEVERING AVENUE
LOS ANGELES, CALIFORNIA 90024

JAY MICHTOM
3424 WILSHIRE BLVD.
LOS ANGELES, CAL. 90005

EDWARD F. MYERS
INFORMATICS INC.
BLDG 64, ROOM 24, SCHIPHOL AIRPORT
AMSTERDAM, THE NETHERLANDS

PETER NORDYKE, JR., EA32
NORTH AMERICAN AVIATION, SID
12214 LAKEWOOD BOULEVARD
DOWNEY, CALIFORNIA

W. M. PACKSLAY
PACIFIC GAS AND ELECTRIC CO.
245 MARKET STREET
SAN FRANCISCO, CALIFORNIA 94105

FRED L RICHARD
CONTROL DATA CORPORATION
6133 MAYNARD AVENUE SOUTH
SEATTLE, WASHINGTON 98108

DR. J. C. ROBERTSON
INFORMATICS INC.
TERESA ZWAARTSTRAAT 114
THE HAGUE, NETHERLANDS

R.J. ROSSHEIM
AUERBACH CORPORATION
1634 ARCH STREET
PHILADELPHIA, PENNSYLVANIA 19103

GEORGE SADOWSKY
THE BROOKLINGS INSTITUTION
1775 MASSACHUSETTS AVE., NW
WASHINGTON, DC 20036

N. R. SAVILLE
UNIVAC, REMINGTON HOUSE
65 HOLBORN VIADUCT
LONDON , E.C.1, ENGLAND

G. SCHWACHHEIM
CENTRO BRASILEIRO PESQUISAS FISICAS
AV. WENCESLAU BRAZ 71 - ZC-82
RIO DE JANEIRO - GB - BRAZIL

HENRI SEHARNE
CALIFORNIA LEGISLATURE
ROOM 5015, STATE CAPITOL
SACRAMENTO, CALIFORNIA

R G SHAW
1445 WEST GEORGIA STREET
VANCOUVER 5, B C
CANADA

M. R. SHURA-BURA, PROFESSOR
MOSCOW STATE UNIVERSITY
MOSCOW K-134

RICHARD J. MCQUILLIN
INFORMATICS INC.
146 MAIN STREET
MAYNARD, MASSACHUSETTS 01754

W. C. MALLONEE
SCIENTIFIC DATA SYSTEMS
1649 SEVENTEENTH STREET
SANTA MONICA, CALIFORNIA

HOWARD M. METCALFE
PLANNING RESEARCH CORPORATION
1100 GLENDON AVENUE
LOS ANGELES, CALIFORNIA 90024

R. W. MITCHELL
COMPUTER ASSOCIATES
LAKESIDE OFFICE PARK
WAKEFIELD, MASS.

HANS W. MINTZEL
COMPUTER USAGE DEVELOPMENT CORP.
8939 SEPULVEDA BOULEVARD
LOS ANGELES, CALIFORNIA 90045

JERRY OGDIN
HELICOPYNE CORP.
SSD/SMRS DATA CORRELATION FAC.
NORTON AFB, CAL. 92409

F. PROSSER
RESEARCH COMPUTING CENTER
INDIANA UNIVERSITY
BLOOMINGTON, INDIANA

CHARLES L RIMPLEY
ROOM 201 CAMPBELL HALL
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA

ROBERT P. ROSIN
YALE UNIVERSITY, COMPUTER CENTER
60 SACHEM STREET
NEW HAVEN, CONNECTICUT 06520

ROBERT E ROUNTREE, JR
3208 ROLLIN ROAD
FALLS CHURCH, VIRGINIA 22042

J G SANDERSON, DEPT OF COMP SCI
UNIVERSITY OF ADELAIDE
NORTH TERRACE, ADELAIDE
SOUTH AUSTRALIA

MARGARET S. SCHNEIDER
1453 COLBY AVE.
LOS ANGELES, CAL. 90025

G. SCOTT
CONTROL DATA CANADA LTD.
410 BOURKE AVENUE
DORVAL, QUEBEC, CANADA

J. E. SHANNON
PHILLIPS PETROLEUM CO.
BARTLESVILLE, OKLAHOMA 74004

RICHARD E. SHEPARDSON
12644 SOUTH YORK AVENUE
HAWTHORNE, CALIFORNIA 90250

R. A. SIBLEY
IBM FIELD SYSTEMS CENTER
2911 ORGAN SPRINGS ROAD
DALLAS, TEXAS 75219

HARVEY SIGAL
UNION CARBIDE CORPORATION
270 PARK AVENUE
NEW YORK, NEW YORK 10017

DONALD N. SILCOTT
10209 OAKLAND AVENUE
KANSAS CITY, MISSOURI 64134

RICHARD D. SMITH
306 HIGHLAND AVENUE
MORRISTOWN, NEW JERSEY

ROBERT L. SMITH, JR.
TEXAS A AND M UNIVERSITY
COLLEGE OF ENGINEERING
COLLEGE STATION, TEXAS

W. A. SMITH
BELL TELEPHONE CO. OF CANADA
10TH FLOOR, 1050 UNIVERSITY STREET
MONTREAL 1, QUEBEC, CANADA

ALLEN SPRINGER
1712 LEFORGE ROAD
YPSILANTI, MICH. 48197

RICHARD A. STACK
9133 OAK PARK AVENUE
MORTON GROVE, ILLINOIS 60053

LIBRARY
COMPUTER SCIENCE DEPARTMENT
STANFORD UNIVERSITY
STANFORD, CALIFORNIA 94305

J. C. STANSBURY
MALDON INTERNATIONAL INC.
2 PARK AVENUE
NEW YORK, N.Y. 10016

T. B. STEEL, JR.
SYSTEMS DEVELOPMENT CORP.
2500 COLORADO AVENUE
SANTA MONICA, CALIFORNIA 90406

JOSEPH F. SULLIVAN
LINCOLN NATNL. LIFE
FORT WAYNE, INDIANA

THE LIBRARY
DEPT OF COMPUTER SCIENCE
UNIVERSITY OF TORONTO
TORONTO 5, ONTARIO, CANADA

JOHN R. TRUITT
ELECTRIC PLANNING DEPT.
CINCINNATI GAS + ELECTRIC CO.
CINCINNATI, OHIO 45201

C. C. TSAO
I B M
2651 STRANG BOULEVARD
YORKTOWN HTS, NEW YORK 10598

E. E. UTMAN
P.O. BOX 200
PRINCETON, NEW JERSEY

C. WALKER
CDC
OLD CONNECTICUT PATH
FRAMINGHAM, MASSACHUSETTS

J. G. WALKER
ENGLISH ELEC-LEO-MARCONI COMP LTD
KIDSGROVE, STOKE-ON-TRENT
STAFFS, ENGLAND

R. E. WAYCHOFF
BURROUGHS
440 SIERRA MADRE VILLA
PASADENA, CALIFORNIA

J. M. WEGSTEIN
NBS
U.S. DEPT. OF COMMERCE
WASHINGTON, D. C. 20234

JOHN R B WHITTLESEY
MANDREL IND. INC.
6909 SOUTHWEST FRWY, P.O. BX.36304
HOUSTON, TEXAS 77036

GIU WIEDERHOLD
COMPUTATION CENTER
STANFORD UNIVERSITY
STANFORD, CALIFORNIA 94305

W. A. WILLIAMS
DATA PROCESSING RESEARCH
STANFORD RESEARCH INSTITUTE
MENLO PARK, CAL. 94025

W.O. WOOTEN
PLANNING RESEARCH CORP.
611 WEST 6TH STREET
SANTA ANA, CAL. 92701

M. YASKOFF
ILLINOIS INSTITUTE OF TECHNOLOGY
COMPUTATION CENTER
CHICAGO, ILLINOIS 60616

DR. A. P. YERSHOV
NOVOSIBIRSK 90
COMPUTATION CENTER
USSR