PASCAL USER'S GROUP

USER'S

GROUP

PASCAL NEWSLETTER

NUMBER 6

COMMUNICATIONS ABOUT THE PROGRAMMING LANGUAGE PASCAL BY PASCALERS

NOVEMBER, 1976

TABLE OF CONTENTS

\#
                                         \#

POLICY -- PASCAL USER'S GROUP AND PASCAL NEWSLETTER

USER'S GROUP POLICIES

Purposes - are to promote the use of the programming language Pascal as well as the ideas behind Pascal. Pascal is a _practical_ language with a _small, systematic_ and _general purpose_ structure being used for:

> * teaching programming concepts
> * developing reliable "production" software
> * implementing software efficiently on today's machines
> * writing portable software

Membership - is open to anyone: particularly the Pascal user, teacher, maintainer, implementor, distributor, or just plain fan. Institutional memberships, especially libraries, are encouraged. Membership is per academic year ending June 30. Anyone joining for a particular year will receive all 4 quarterly issues of _Pascal Newsletter_ for that year. (In other words, back issues are sent automatically.) First time members receive a receipt for membership; renewers do not to save PUG postage.

Cost of membership per academic year is $4 and may be sent to:
Pascal User's Group/ %Andy Mickel/University Computer Center/ University of Minnesota/Minneapolis, MN 55455 USA/ phone: (612) 376-7290
In the United Kingdom, send £2.50 to:
Pascal Users' Group/ %Judy Mullins/Mathematics Department/The University/ SOUTHAMPTON/SO9 5NH/United Kingdom/ (telephone 0703-559122 x2387)

NEWSLETTER POLICIES

The _Pascal Newsletter_ the official but _informal_ publication of the User's Group. It is produced quarterly (usually September, November, February, and May). A complete membership list is printed in the November issue. Single back issues are available for $1 each. Out of print: #s 1,2,3
#4 available from George Richmond/Computing Center/U of Colorado/Boulder/80309

The contribution by PUG members of ideas, queries, articles, letters, and opinions for the _Newsletter_ is important. Articles and notices concern: Pascal philosophy, the use of Pascal as a teaching tool, uses of Pascal at different computer installations, portable (applications) program exchange, how to promote Pascal usage, and important events (meetings, publications, etc.).

Implementation information for the programming language Pascal on different computer systems is provided in the _Newsletter_ out of the necessity to spread the use of Pascal. This includes contacts for maintainers, documentors, and distributors of a given implementation as well as where to send bug reports. Both qualitative and quantitative descriptions for a given implementation are publicized. Proposed extensions to Standard Pascal for users of a given implementation are aired. Announcements are made of the availability of new program writing tools for a Pascal environment.

Miscellaneous features include bibliographies, questionaires, and membership lists. Editor's notes are in Pascal style comments (**).

WRITTEN INFORMATION FOR THE _Newsletter_ IS EASIER TO PRINT IF YOU TYPE ALL MATERIAL 1½ OR DOUBLE SPACED SO THAT IT IS IN "CAMERA-READY" AND "PHOTO-REDUCIBLE" FORM FOR THE PRINTER. REMEMBER, ALL LETTERS TO US WILL BE PRINTED IN THE _Newsletter_ UNLESS THEY CONTAIN A REQUEST TO THE CONTRARY. AN OVERRIDING GUIDE SEEN IN AN OLD "AD" MAGAZINE APPLIES! "_all the news that fits, we print!_" - Andy Mickel, editor, John P. Strait, associate editor. Nov. 10, 1976.

PART I - Standards

Wow! It took only one issue of PUG's Pascal Newsletter to bring on an avalanche of "Where do we go from here?"s! It was first put clearly in print with a short note in PUGN #3 by George Poonen who noted that various implementations had diverged and that a standard was necessary. Now we have: Tony Addyman, Frank Brewster, Charles Hedrick, and Willett Kempton (see News in HERE AND THERE); Mike Schneider, Rich Cichelli, and Arthur Sale (see ARTICLES); and Steve Young, Tony Addyman (again), Duke Haiduk, Judy Mullins, Arthur Sale (again), and Tim Bonham (see OPEN FORUM) all discussing the topic of standards. The concern, I believe, is out of our desire to see Pascal succeed. We are in a computing environment which is not altogether friendly to Pascal. We want to be able to respectably use Pascal in the future.

I have been very confused on the subject of Pascal standards in the past. Mike Schneider and Rich Cichelli have (I think) straightened me out. You see, I thought we already had a Standard Pascal, with the Revised Report and the Axiomatic Definition. These two concise and elegant (although not perfect - but yet what do you want?) documents were produced by Niklaus Wirth and his associates and coworkers. And I believe that Pascal has merit because it was produced by a single man of the calibre of Niklaus Wirth, who (as evident from his work) profoundly understands programming language design, from linguistics to implementation. This one person could decide what to meld when meeting all of the design goals set out from the start. I wanted to do what I could to call for adherence to what Niklaus Wirth called "Standard Pascal". Because with time, I increasingly appreciated what he had written in several articles. He pointed out for example that certain "favorite features" had to be omitted in order to meet the design goals of a small and efficient system. Also that some aspects were best left undefined. And that other features were omitted with good reason to achieve the goal of providing a tool with which to produce reliable software (okay - you could call it: "protect the error-prone human programmer from himself or herself." It may not be pleasant, but experiencing is believing; a good dose of egoless programming goes well with this.) It goes without saying that Pascal is not the ultimate programming language, or perfect, or that it is all things to all people. All good so far.

EDITOR'S CONTRIBUTION

But then other events took place. The Revised Report suffered "revisionism": Nov., 1972, July, 1973, Dec., 1973, the User Manual and Report, first edition (1974), second edition, first printing (1975), and now the second edition, third printing (1976). How can one call for adherence to the "standard" when the same(?) "standard" keeps changing?

Also among the many ill-conceived suggestions for "improvements" to the language by users, there were some very few that seemed reasonable to dyed-in-the-wool Pascalers. There was no mechanism for sounding these out for worthiness and acceptance, save writing to Niklaus and Urs in Zurich. This has been very frustrating because we didn't know where we were heading. (What was Pascal's future destined to be according to its creators?) We were told on the one hand, "no more changes." We relaxed and said "fine." Then a revision came along and we felt cheated. We weren't kept informed of what other users had suggested, either.

Rich and Mike have pointed out that Pascal can't continue to be what Niklaus Wirth says it is. And that Andy Mickel can't arbitrarily restrain attempts to change it because 1) Andy fears destruction of the language by attempts to "save" it, or 2) Andy doesn't want them to destroy the essential simplicity of Pascal which is probably its most likely reason for success. They also pointed out that we don't have an officially accepted standard; a "political standard" if you will. Really, when that concept dawned on me it made sense. A major computer manufacturer, when choosing a common language for all its software development, democratically decided to pick the one that most of its programmers wanted to use. With the choice of language X 30%, Pascal version A 25%, Pascal version B 13%, and Pascal version C 27%, language X won by a plurality (and by default!) and too bad - as we all can see. If we want Pascal to ultimately and completely succeed, we can't have this!

Now how do we resolve the conflict(s)? Many persons suggest a "PUG Standards Committee", and frankly, although I think committees are inherently evil; I don't see any other choice. The alternative at this point is to lower our expectations, quit striving for excellence, quit "dreaming the impossible dream" of seeing Pascal take over the majority of industrial and academic computing (wiping out Cobol and Fortran within our lifetimes).* Then we could say regretfully - "wow, Pascal's nice, but..." as so many of our half-hearted supporters and critics do now.

I feel that: 1) we should continue to debate this topic; 2) a PUG Standards Committee when set up should be small (less than 8 members); 3) its charter be initially agreed on so as to limit its power; 4) within the committee's initial charge

* This brings to mind two acronyms: John Easton's SHAFT or Society to Help Abolish Fortran Teaching, and Mitch Wand's ACS or the American Cobol Society - analogous in meaning to the American Cancer Society.

EDITOR'S CONTRIBUTION

the action should be to get the Revised Report (User Manual and Report, Second Edition third printing) accepted as an official standard as is (even if only provisionally);
5) later the committee could recommend subsequent actions.

Look up the articles in this issue of PUG Newsletter by Mike and Rich with their excellent analyses of the current situation. Rich bluntly hints that many features are best left to separate software writing tools. In all honesty, I don't see how Arthur Sale can say in his October 22 letter to Judy Mullins, "Of course I agree that standard Pascal must be adhered to" and also say that it is best in specific cases to add features that all Burroughs Algol programmers are used to. Pascal was meant to be a departure from the past. See also the article "Experience from the Standardization of the SIMULA Programming Language", by Jacob Palme, SOFTWARE, Practice and Experience Vol. 6, No. 3 July-Sept, 1976, pp 405-409. (It seems that each issue of SOFTWARE, Practice and Experience always has some good articles for the practical programmer!)

We are indeed in a unique position in computer science history as people (rather than large organizations) responsibly influencing an influential language.

PART II - Pascal User's Group and Pascal Newsletter

1) PUG has 516 members in 22 countries and 43 states. (We had 317 at last writing.)
I'm sorry this newsletter is so late. But this year the November issue will have in it feedback to the September issue.

2) Mr. Judy Mullins and Prof. D. W. Barron of the University of Southampton have done us all a favor by creating a European distribution center for PUG newsletters and a clearing house for PUG memberships in the United Kingdom. Judy was concerned that members in the U.K. would not get fast mail service, while at the same time having to pay a relatively high exchange rate for $4. We in fact had decided to send the first 2 newsletters (#5 & #6) air mail because we could afford it and Pascal needed the shot in the arm. What has transpired between Southampton and Minnesota is no less than 6 letters east to west and 5 letters and a phone call west to east on the subject of cheaper ways to send the newsletters (air freight, etc.) These 11 letters are not reproduced here; they mostly contained calculations and mechanics of mailing.

3) While we are on the subject of finances, I'm happy to report that we're doing just about right. We've been able to afford to send out 250 issues of #4, and do a large mailout requesting implementation information. We still plan to print and mail #7 and #8, so don't worry. The next sheet contains a breakdown:

516 members @ $4	$2064.00	
8 members not paid yet	- 32.00	
6 members for 2 years	24.00	extra
1 member for 5 years	16.00	extra
ABM + JPS contribution	29.00	
	$2101.00	Total Assets

postage, mass mailings	$ 52.00	
refunds for overpayment	4.00	
printing and mailing #5	487.10	(700 printed, 368 mailed)
buying 230 copies of #4	100.00	no bill for mailing yet
postage for #5 backissues	27.40	so far
printing newsletter titles	5.60	
	$676.10	Total Expenses

Theoretical balance = 2101.00 - 676.10 = 1424.90

Cash on hand	$ 77.76
PUG UCC Account	$1353.30
Actual balance =	$1431.06

4) Backissues. See the section in HERE AND THERE. Our offer to send #4 to persons in North America who didn't already get one directly from George Richmond expired on October 2nd. We simply ran out. But we did buy time. And now the problem of trying to include information in #6 that was in #4 is not so acute because #6, #7, and #8 will gradually make up for that. We will be updating nearly all the news which appeared in #4. So for those of you who joined after October 2nd and still want the newsletter #4, order one from George Richmond.

5) I apologize for announcing our policy of: "all the news that fits, we print" in the same issue that we put the policy into practice. We modelled the policy after SIGPLAN Notices. Feedback to Newsletter #5 has been mostly favorable; the unfavorable comments have been largely unwritten. Some heretofore unwritten comments went like this:

"Your organization could be improved."
"It was fun reading the News section in HERE AND THERE."
"It's good to see the correspondence you had with Zurich."
"It's taken a long time to get my newsletter in the mail."
"The articles you printed weren't so hot."

6) Last issue we tried to plan events so that you would receive the newsletter at the beginning of September. But we didn't come close. Our cutoff date for material was supposed to be July 15, but it lagged to July 31. We began putting the newsletter together July 4. We went to press August 13 (and here's the bad news) 25 days later we got our 700 copies on September 7. We had it all in the mail September 9. In the U.S. we know (so far) that some arrived as late as October 2! This issue will probably arrive by Christmas (no kidding) but we began November 4 to put it together and we are going to press November 15 - much better than last time, except we have a late start. Our cutoff for material for this issue was originally October 1 but lagged to November 5. Issue #7 will probably be smaller as it will go to press probably before we get reaction to this issue. By being smaller, it also won't cost as much to print.

7) Offers to help. In #5, N. Solntseff and W. Richard Stevens offered to help with the User's Group. Now that some things have been established, several tasks are becoming clear. These are:

. managing distribution of software writing tools for Pascal written in standard Pascal

. managing distribution or cataloging of library and applications programs for Pascal written in standard Pascal

. maintaining a bibliography on all publications about Pascal (including articles and books)

Any takers?

8) Two encouraging trends. First, with microprocessor interest spreading (real computer power to the people!) it is important to have a Pascal subset compete with BASIC in 16K. Mark Rustad understands this very well - see his Motorola 6800 description in IMPLEMENTATION NOTES. Mark would like to hear from those persons interested. Second, John and I have been getting lots of inquiries about Pascal and implementations in the form of phone calls and letters - with most of them from persons in industry. Predominate are small software writing firms and minicomputer companies. So next time someone says Pascal is okay, but it's not "real world" tell them that it's happening right now.

9) Thanx are due to all the people who have sent in information to print - that makes the newsletter. Thanx to John, Tim Bonham, Jim Miner, and Herb Rubenstein for helping put together this issue.

– Andy

November 14, 1976

ANNOUNCEMENT OF A PASCAL USERS' GROUP
DISTRIBUTION CENTRE IN THE
UNITED KINGDOM

AIMS

1. To expedite distribution of the P.U.G. Newsletter to the U.K. and the rest of Europe, the Near and Middle East and Northern Africa.
2. To collect memberships in P.U.G. from U.K. members avoiding high bank charges on transfers of £ to $.

DISTRIBUTION

1. Central P.U.G. at Minnesota will send the original of the newsletter to Southampton for reprinting.
2. Newsletters will be mailed (second-class postage) from Southampton to members in Europe, the Near and Middle East and Northern Africa.

PASCAL USERS' GROUP MEMBERSHIPS

1. The address for U.K. Region memberships is

   > Pascal Users' Group
   > c/o Judy Mullins
   > Mathematics Department
   > The University
   > SOUTHAMPTON.   SO9 5NH
   >
   > (telephone 0703-559122 x2387)

2. Members can pay £2.50 by cheque or postal order to, PASCAL USERS' GROUP (UK) at the above address, and will receive a receipt and member certificate directly.
3. Membership forms will be forwarded at short intervals to Minnesota (at least in time to catch the next newsletter); a copy is kept at Southampton.

AVOIDING CONFUSION

1. There is only one membership list and labelling program - Minnesota's.
2. Therefore anyone can join directly by writing to the U.S.A.
3. Using the U.K. Distribution Centre only saves money.
4. No matter how he/she joined, a member with an address in the U.K. will receive newsletters via Southampton.
5. All correspondence other than subscriptions (such as change of address, articles for the newsletter, or questions about compilers) must go direct to Minnesota. If it inadvertently arrives at Southampton it will be sent on by airmail.

August, 1976.   J.M. Mullins.

Rev. November, 1976.   A.B. Mickel.

NEWS (ALPHABETICAL BY LAST NAME)

A. M. Addyman, Department of Computer Science, The University, Manchester M13 9PL
United Kingdom (PUG member): "I would like to join the Pascal Users' Group.
Also, I am engaged in an effort to have Pascal standardised by a major
standard's organisation, e.g. ANSI or ISO. How may I use your newsletter to
contact people who would be interested in this, or alternatively to discover
that there is considerable opposition?" (*9/10/76*)

Urs Ammann, Institut fur Informatik, ETH - Zentrum, CH-8092 Zurich, Switzerland
(PUG member): "...By the way: What is your philosophy with the letters you
received as to their publication in the Newsletter? I was somewhat astonished
to see private correspondence in it. While I agree that this kind of
information distribution makes editorship most easy, it is my strong opinion
that any letter which is not explicitly marked as "letter to the editor" should
not be published in full length, since this clearly exceeds or even
contradictes (sic) the purpose of private correspondence.
"Please don't misinterpret this statement! I have nothing against transparency,
on the contrary! Any information of general interest you find in your
correspondence should be passed on. But you will agree that with some effort
from the editor, information can be passed on without letting everybody read
private correspondence...." (*9/29/76*)

Diosdado P. Banatao, 3060 Bilbo Drive, San Jose, CA 95121 (PUG member): "I
would like to be a member of the Pascal Users Group... My interests are in
microprocessors and microcomputers and involved in both hardware and software
design..." (*10/10/76*)

Philip N. Bergstresser, 128 Jackson Ave., Madison, AL 35758 (PUG member): "We
at TRW Systems are using Pascal on the CDC 7600, CDC 6400 and TI-ASC and claim
the Guinness record for program size." (*9/21/76*)

Frank M. Brewster, 4701 Kenmore Ave #1009, Alexandria, VA 22304 (PUG member):
"...It's been pointed out that many BASICs are 'non-standard'. I have yet to
hear anyone ask, 'Why?'. The answer seems obvious: the language initially didn't
have 'legal' provision for many of the users' real problems. The current ANSI
BASIC proposal still demonstrates this failing. E.g., the CHR and SEQ(or ORD)
functions are optional; how can anyone do general work without these functions?
So BASICs will continue to be 'non-standard', as people fill in the gaps. If a
car were sold without say, steering wheel, no one should complain if a buyer adds
a tiller. The point is that if the automotive designer finds steering wheels
uninteresting and refuses to specify them as standard equipment, the user has
two options (assuming he buys the car in spite of its failings): design his own
steering apparatus, or cooperate with others in filling the gap in the 'standard'.
If the designer won't see the issue, users will. The letters in the newsletter
mention, e.g., Appay harping and formatted input problems; Apparently Wirth's
not concerned. If you and others do nothing, then everybody either abandons
Pascal or invents their own wheel (tiller?). But why don't those of you with
early and practical experience with the language-
-list your complaints & problems, ranked, one list per man. (Maybe in a newsletter
   section, 'What's wrong with Pascal?'?)
-compare notes for similarities
-see if you can agree on solutions to any of these
-implement experimental changes; test till working
-promulgate as PUG-US 'extensions'
"The last item is the tackiest one. "A camel is a horse designed by a committee."
Standards - the real ones, in actual use - are designed by those who are actually
working in the field, in the course of their work. So if you and other of the
few presently experienced Pascal users won't add to or alter Wirth's
pronouncements, don't be surprised at the later irreverence of others.
"All of you (me too someday) may owe a lot to Wirth. His opinions deserve
respect and attention. But if he's to be treated as God, and his language as the
ten commandments, how can Pascal be improved? The time to 'standardize' is not
now, but after user problems have been faced frankly, and solutions found...."
(*10/09/76*)

C. E. Bright, E.I. Du Pont de Nemours & Co.; Engineering Development Lab, 101
Beech Street, Wilmington, DE 19898 (PUG member): "Have: PDP-11 series machines:
04, 05, 10, 15, 20, 40, 45. Using: (1) Prof. Per Brinch Hansen Solo Pascal
Compilers, (2) University of Illinois DOS V4 Pascal Compiler, (3) Pascal P2
System.
"All of the above systems have their drawbacks. My interest is in a better
transportable system for use on μCPU applications. I am very happy with the
CDC 6000 version 3.4 at Purdue University; however, achieving the same degree
of performance on a mini-computer has been and will continue to be a challenge
Mr. Stephen C. Schwarm, a coworker, is in the process of starting a DECUS
SIG PASCAL for PDP users of Pascal." (*9/13/76*)

HERE AND THERE WITH PASCAL
(NEWS FROM MEMBERS, CONFERENCES, NEW BOOKS, APPLICATIONS PROGRAMS, ETC.)

HERE AND THERE WITH PASCAL

(NEWS FROM MEMBERS, CONFERENCES, NEW BOOKS, APPLICATIONS PROGRAMS, ETC.)

K. Frankowski, Computer Science Dept., 114 Lind Hall, University of Minnesota, Minneapolis, MN 55455 (PUG member): "If one wants to have formatted input, simply read several integers (if they are run together) in one integer and use div and/or mod to extract the values desired." (*10/15/76*)

Dennis Graham, Amdahl Corp., 1250 E. Arques Ave., Sunnyvale, CA 94086 (PUG member): "I am interested in running Pascal on an Amdahl-470 V/6 system and am contacting the University of Manitoba about their compiler." (*10/26/76*)

David J. Griffiths, Academic Computer Centre, Tyler Hall, University of Rhode Island, West Warwick, RI 02881 (PUG member): "I am investigating the possibility of implementing Pascal on our IBM360-370. Concurrent Pascal would be the ideal, since we wish to investigate more advanced operating systems, however, we are prepared to settle for less." (*10/3/76*)

Donald E. Grimes, 90 Sylvia Street, Arlington, MA 02174 (PUG member): "Congratulations on a timely Newsletter #5, and thanks for your efforts in establishing PUG." (*10/8/76*)

Charles Hedrick, 183 Commerce West, University of Illinois, Urbana, IL 61801 (PUG member): "When considering operational definitions of portability maybe it is useful to distinguish among versions that are:
-machine independent
-semi-machine dependent and concepts universal
-machine dependent and site independent
"The last choice may not be so bad for Pascal to shoot for." (*10/15/76*)

Carl Henry, Computer Center, Carleton College, Northfield, MN 55057 (PUG member) "...We are using...the University of Illinois version (Mickunas, et al) and runs under DOS V4 on an 11/20, (very little use has been made of it so far.). "A brief description of our facilities: 6 PDP-8s - home brewed version of TSS/8 and OS/8; PDP 11/20 - DOS, RT-11, RSTS V4; PDP-11/40 - RSTS V6, UNIX V6." (*10/15/76*)

Mark Hersey, 323 Village Drive Apt. 534, East Lansing, MI 48823 (PUG member): "currently modifying P2 version of Janus compiler for readability, fixing bugs, expanding subset processed, and improving portability. "All work being done on Michigan State University's CDC 6500." (*10/4/76*)

Brian W. Johnson, 1525 Westlake, Plano, TX 75075 (PUG member): "I am particularly interested in μ processor versions. We have it on the PDP-10 and PDP-11 at UT Dallas." (*11/4/76*)

Willett Kempton, 2512 San Gabriel St., Austin, TX 78705 (PUG member): "Thanks for the newsletters. ... I was delighted to see that dynamic array parameters will be implemented in CDC Pascal; this is clearly an extremely important feature; and I would urge that it become a feature of the standard language; rather than an extension available in some implementations (and not others). "Keep after those implementers to accept standard Pascal programs!!" (*10/27/76*)

C. A. Lang, Cambridge University Press, Pitt Building, Trumpington St., Cambridge CB2 1RP, United Kingdom (PUG member): "We are interested in publishing books concerned with Pascal." (*10/26/76*)

Michael Lutz, School of Computer Science and Technology, Rochester Institute of Technology, Rochester, NY 14623 (PUG member): "...I would also appreciate any information you might have on Pascal implementations for the Xerox (Honeywell) Sigma 5 - 9 and PDP 11 computers. We have both a Sigma 9 and a PDP 11/T34 (with 48K words of memory) here at R.I.T., and we are interested in obtaining Pascal for use in our courses...." (*10/27/76*)

John Montague, Los Alamos Scientific Laboratory, Group C11 - Mail Stop 296, Los Alamos, NM 87545 (PUG member): "...We plan to bring up Pascal on the CRAY-1, probably using the P-code compiler to bootstrap." (*10/18/76*)

Judy Mullins, Computer Studies Group, Department of Mathematics, The University, Southampton SO9 5NH, United Kingdom (PUG member): "...Pascal is alive and happy in Southampton. One hundred nineteen-year-olds are pushing in programs by the hundreds ... and doing amazingly well. I do believe it is the language that is so friendly that increases their interest and output...
"...I was wondering if it would be appropriate to have a section of PUGN for exchange of course ideas, examples etc. This would have to be firmly controlled space-wise, but could prove very informative expecially for universities who have Pascal but don't teach it yet. Later on a survey on the use of Pascal in teaching would be of great interest. Addyman's survey showed Pascal is growing and therefore its growth should be monitored every year.
"Another thought was for book reviews. Pascal primers are beginning to proliferate and we have strong views on the ones we've seen. Once again, to have the right effect this section would need to be controlled, and I'm not sure that we want to start issuing PUG marks of approval or anything like that. However, reviews in normal journals are only opinions and it does seem fitting for opinions of Pascalers on Pascal books to be in the Pascal Newsletter." (*11/3/76*)

Fred Powell, Computer Center, Mary Baldwin College, Staunton, VA 24401 (PUG member): "We have Pascal P2 and are interested in implementing Pascal on an IBM 1130 and possibly a System 3. Other possibilites include investigating data bases and disk access techniques with Pascal." (*9/24/76*)

Douglas H. Quebbeman, 2235 Lombardy Drive, Jeffersonville, IN 47130, (PUG member): "...Having seen the article in the June '76 Random Bits (Indiana University's Computing Center Newsletter) on the Pascal User's Group, I decided to join. I am a student and part-time operator - programming consultant and have only recently begun using Pascal, but I am quite enthused about its flexibility (especially considering my wrestling bouts with Fortran) and hope to become more proficient in it. So, thanks (for forming the User Group) and I hope to hear from you soon." (*9/24/76*)

Peter A. Rigsbee, Code 5494, Naval Research Laboratory, Washington, DC 20375 (PUG member): "...My connection with Pascal is that my group is trying to get Per Brinch Hansen's SOLO operating system to run on a PDP 11/40, and once this is done, will be using Pascal as a primary systems programming language...." (*8/25/76*)

Sérgio de Mello Schneider, Departamento de Computação, Univ. Federal de S. Carlos, C.P. 384, 13560 S. Carlos SP, Brazil (PUG member): "We have a HP 2100A at our installation (32K words core, 2 disks, 1 tape, DOS) and we are looking for a Pascal compiler. There is no way we can produce one in the next 3 years. Could you help us?" (*10/21/76*)

Stephen C. Schwarm, E. I. du Pont de Nemours Co., 101 Beech St., Wilmington, DE 19898 (PUG member): "I am chairman of DECUS SIG Pascal and I will be glad to help with distribution any systems on DEC PDP-11's." (*10/29/76*)

Dave Tarabar, Data General Corp., Field Engineering, 235 Old Connecticut Path, Framingham, MA 01701 (PUG member): "I was very pleased to receive and read the first PUG Pascal Newsletter. It was full of interesting information. The newsletter will be very useful in publishing the correspondence with Zurich and other implementors and your summary of all known implementations was great. Keep up the good work." (*10/18/76*)

William P. Taylor, L-315, University of California, PO Box 808, Livermore, CA 94550 (PUG member): "I am interested in obtaining information about implementations of Pascal on 16-bit mini-computers. I am especially interested in implementations for the PDP-11 as we will be getting one soon. Also, some of my fellow employees here at Lawrence Livermore Laboratory wish to implement a structured programming language like Pascal for system development on a new mini-computer." (*10/3/76*)

APPLICATIONS PROGRAMS

STANFORD UNIVERSITY

STANFORD LINEAR ACCELERATOR CENTER

Mail Address
SLAC, P. O. Box 4349
Stanford, California 94305

18 October, 1976

Request for Programs

Pascal Users:

I am presently doing research on Pascal to determine how various parts of the language are used and what patterns of execution occur in actual programs. This will be similar to a study done by Knuth on Fortran (1).

In this regard I am interested in obtaining a sample of programs from a wide range of users in hopes that the results of this study might be representative of the actual use of Pascal.

If you have or know of programs which can be lent to this effort, I would very much like to hear from you. I can be contacted by mail at

Mail Drop 88
Stanford Linear Accelerator Center
P. O. Box 4349
Stanford, CA. 94305

or by phone at

(415) 854-3300 X2802.

John Banning

(1) D. E. Knuth, "An Empirical Study of FORTRAN Programs", Software Practice and Experience, Vol. 1 (1971), 105-133.

(*Note: John also enclosed a note which said: "With regards to the enclosed request, I expect that the mentioned study will complete sometime in the first quarter of 1977. I would be most happy at that time to provide a summary of the results for the Newsletter if you are interested - ... Does there exist some formal mechanism for Pascal program interchange (between users), and, if so, who is running it and how can I contact them?"*)

THIRD ANNUAL COMPUTER STUDIES SYMPOSIUM

UNIVERSITY of SOUTHAMPTON

PASCAL IMPLEMENTATION AND APPLICATIONS

24 AND 25 MARCH 1977

AIMS

Few languages since FORTRAN have had the same run-away success as Niklaus Wirth's PASCAL, which shows signs of becoming a *de facto* standard for Computer Science teaching and research, as well as pointing the way to a new generation of sparse, simple languages.

The purpose of this symposium is to explore "what's going on" in PASCAL at the present time. Leading authorities will describe new implementations and applications in systems programming, research and education. The symposium will end with an open discussion about the future of PASCAL.

In the tradition of the Southampton symposium, speakers will be allowed ample time for their presentations, together with provision for a discussion at the end of each lecture. Attendance will be kept to 100 and it may be necessary to limit applications from each institution. Applicants are expected to have a working knowledge of PASCAL.

Full preprints of the Proceedings will be available on registration; the Proceedings will subsequently be published in book form.

SYMPOSIUM CHAIRMAN
PROFESSOR D.W. Barron

SYMPOSIUM ORGANIZER
MISS J.M. MULLINS

SPEAKERS

Dr.Urs Ammann
Technische Hochschulen
Zurich
SWITZERLAND
***

Prof.David Barron
Mathematics Department
The University
SOUTHAMPTON
***

Dr.Per Brinch Hansen
Computer Science Program
University of Southern
  California
LOS ANGELES
***

Dr.Barry Hood
Electronics Department
The University
SOUTHAMPTON
***

Miss Judy Mullins
Computer Studies Group
The University
SOUTHAMPTON

Dr.Mike Rees
Computer Studies Group
The University
SOUTHAMPTON
***

Dr.David Watt
Computer Science Department
University
GLASGOW
***

Dr.Graham Webster
Computer Science Department
Teesside Polytechnic
Middlesborough,
CLEVELAND.
***

Dr.Jim Welsh
Computer Science Department
Queen's University
BELFAST
***

PROGRAMME

IMPLEMENTATION

Dr.U.Ammann	The Zurich Compilers
Dr.J.Welsh	Two ICL 1900 Compilers
Dr.D.A.Watt	A Diagnostic System
Dr.M.J.Rees	PASCAL on an Advanced Architecture
Miss J.M.Mullins	A PASCAL Machine?

APPLICATION

Dr.P.Brinch Hansen	Concurrent PASCAL
Dr.G.Webster	PASCAL in Education
Dr.B.Hood	PASCAL in Research

THE FUTURE

Panel Discussion introduced by Prof.D.W.Barron

BOOKS AND ARTICLES

(*There has been no news of new books on Pascal. In future issues of the
Newsletter, we should also list current articles appearing in journals and
other computer science literature. Apologies for the void in this section
in this issue.*)

A. M. Addyman and H. R. Addyman, "Which Language?", Computer Bulletin,
June, 1976, pp 31-33. [an article which surveys the language
used at various institutions teaching computer science]

(*Last issue there were a couple mistakes in the list of books; these are
corrected below.*)

A Primer on PASCAL by Richard Conway, David Gries, and E. C. Zimmerman,
Winthrop Publishers; 1976; 448 pages; paperbound; $9.95.

Introduction to Problem Solving and Programming with Pascal, by G. Michael
Schneider, Steven W. Weingart, and David M. Perlman, Wiley, to
be published in late 1977. (*A complete soft cover manuscript
will be available March 1, 1977 and may be ordered from Michael
Schneider, Computer Science Department, 114 Lind Hall, University
of Minnesota, Minneapolis, MN 55455. Such copies may be
duplicated(once received) for local use.*)

PASCAL User Manual and Report, by Kathleen Jensen and Niklaus Wirth,
Springer-Verlag, 1974, 1975, 167 pages, paperbound, $5.90.
Second (study) edition. (*This book is selling well; it's in
its third printing which now incorporates the errata that
appears on the next page as reproduced from Newsletter #4. In
Newsletter #5 we printed out of date errata because no one
kindly informed us of anything more up to date. So like the
implementation notes, we are only as good as what people send
us to print. Note that this errata includes the change to the
language Pascal - namely the generalization of the read and
write procedures to perform I/O on files of any type, not just
text files.*)

ALL ENQUIRIES TO

PASCAL SYMPOSIUM,
MATHEMATICS DEPARTMENT,
THE UNIVERSITY,
SOUTHAMPTON SO9 5NH
ENGLAND

TELEPHONE

0703-559122 EXT 733
ASK FOR PASCAL SYMPOSIUM SECRETARY

INTERNATIONAL TELEPHONE

44-703-559122 EXT 733

TELEX

47661

ARRANGEMENTS

APPLICATIONS   must be made on the official form (or
a copy) to reach Southampton by
FRIDAY 18 FEBRUARY 1977
accompanied by full payment of the fee.

COST   £45 , inclusive of

- Accommodation on Wednesday and Thursday nights
- Lunches and teas on Thursday and Friday
- Medieval Banquet at Beaulieu in the New Forest
- Complete Preprints of the Proceedings.

Applicants from overseas should pay
in sterling by Bankers Draft or International
Money Order.

No Value Added Tax is payable.

ARRIVAL   Delegates are expected to arrive on
Wednesday evening as these first session
starts at 0930 on Thursday.

DEPARTURE   Transport to the station will be arranged
after the last session at 1630 on Friday.

ACCOMMODATION   will be provided at Chamberlain Hall of
Residence. Further details, addresses and
maps will be sent on accepting instructions
on 25 February 1977.

Errata to   p   l   c

PASCAL   13 -3 r   "if" by "If"

User Manual   45 -18 r   "<unsigned constant>" by "<constant>"

and Report   51 16 r   "(output)" by "(output):"

Second Edition.   56 -6 r   "fi" by "f(i)", "g(i+1)" by "g(j+1)", "gi" by "g(j)"

                             63   2 r   "1" by "n", "2" by "n-1", "3" by "n-2",
                                         "(n-1)" by "2", "n" by "1"

                             69 23 r   "stricly" by "strictly"

KEY: 75   7 r   "i,j" by "i"

p = page number   77 18 r   "   begin inorder(pↃ.llink);" by "begin inorder(pↃ.llink);"

1 = line number   78 -14 r   ", <formal" by "; <formal"

    (blank lines   81   2 r   "extent" by "extend"

    are ignored   84 -15 r   "as ne" by "as one"

    and negative   86   8 i   The procedure read can also be used to read from a file f which
                                         is not a textfile.
    line numbers   read(f,x)
                                         in this case stands for
    are counted   begin x := fↃ; get(f) end
    from the   87   8 i   The procedure write can also be used to write onto a file f
                                         which is not a textfile.
    bottom)   write(f,x)
                                         in this case stands for
c = code   begin fↃ := x; put(f) and
    (that is: 98 10 r   "debby" by "debby   ;"
    r = replace   102   7 r   " r" by "or"
    i = insert)   102 20 r   "bufffer" by "buffer"
                           103 -6 r   "scaler" by "scalar"
                           103 -7 r   " char, and alfa" by "and char are listed"
                           105   8 r   "   " by "105"
                           105 12 r   " nly" by "only"
                           105 -1 i   dispose(p,t1,...,tn) can be used to indicate that storage occu-
                                         pied by the variable pↃ (with tag field
                                         values t1...tn) is no longer needed.
                           117   r   (diagram expression) "≤" by "<=", "≥" by ">=", "≠" by "<>"
                           120 -18 r   "neither be formal nor non local" by
                                         "not be declared on intermediate level"
                           121   4 i   177: assignment to function identifier not allowed here
                                         178: multidefined record variant
                                         179: X-opt of actual proc/func does not match formal declaration
                                         180: control variable must not be formal
                                         181: constant part of address out of range
                           121   8 i   205: zero string not allowed
                                         206: integer part of real constant exceeds range
                           121 -8 i   268: too many exit labels
                           124 -15 r   "14" by "15"
                           124 -14 r   "14" by "15"
                           127 27 r   "18.A" by "4.A"
                           133   3 r   "two" by "to"
                           135   5 r   "althought" by "although"
                           135 30 r   "subtstitute" by "substitute"
                           140 11 r   "structure type" by "structured type"
                           161 -17 r   whole line by
                                         "addition to the procedures get and put. The textfiles these"
                           161 -16 r   whole line by
                                         "standard procedures apply to must not necessarily represent"
                           162   3 r   "end of line" by "end of line"
                           162 -15 i   The procedure read can also be used to read from a file f which
                                         is not a textfile. read(f,x) is in this case equivalent to
                                         x := fↃ; get(f).
                           162 -6 i   The procedure write can also be used to write onto a file f
                                         which is not a textfile. write(f,x) is in this case equivalent
                                         to fↃ := x; put(f).

PAST ISSUES OF PASCAL NEWSLETTER

Reproduced below is a complete description of Newsletters 1; 2; 3; and 4.
Numbers 1, 2, and 3 are out of print, but they did appear in issues of SIGPLAN
Notices, the ACM Special Interest Group on Programming LANguages monthly
journal. Number 4 is available for $1.00 from George H. Richmond, Computing
Center, 3645 Marine Street, University of Colorado, Boulder, CO 80309.

#1 January, 1974 (also SIGPLAN Notices Vol. 9 No. 3 1974 March) 8 pages.

#2 May, 1974 (also SIGPLAN Notices Vol. 9 No. 11 1974 November) 18 pages.

#3 February, 1975 (also SIGPLAN Notices Vol. 11 No. 2 1976 February) 19 pages.

#4 July, 1976 (copies may be obtained for $1.00 from George Richmond,
    ... ...
    the editor, as explained on the previous page) 103 pages.

ROSTER 11/14/76

For our mutual benefit in communication, here is the 516 member PUG roster spanning 22 countries and 43 states. It is sorted (intelligently, we think) by zip (mail) codes (U.S. first) and then alphabetically by country. You can see at a glance who is at a well known organization at a well known place or who is in your area (or on your street!). Now, if you need an index by last name, there is one at the end, cross-referencing with zip (mail) code.

* * * * * * * * * *

HENRY F. LEDGARD
COMPUTER AND INFO. SCI.
U OF MASSACHUSETTES
AMHERST   MA 01002
(413) 545-2744

NORMAN E. SONDAK
COMP. SCI. DEPT.
WORCESTER POLYTECHNIC INSTITUTE
WORCESTER   MA 01609
(617) 753-1411

HANK EDWARDS
2C BRACKETT ROAD
FRAMINGHAM   MA 01701
(617) 620-1066 (HOME)
(617) 897-5111 X6809

BERNIE ROSMAN
MATH/CS DEPT.
FRAMINGHAM STATE COLLEGE
FRAMINGHAM   MA 01701
(617) 872-3501

DAVID TARABAR
FIELD ENGINEERING
DATA GENERAL CORPORATION
235 OLD CONNECTICUT PATH
FRAMINGHAM   MA 01701
(617) 620-1200

LLOYD DICKMAN
25 HAWTHORNE VILLAGE
CONCORD   MA 01742

ATTN: LIBRARY
ML5-4/A20
DIGITAL EQUIPMENT CORPORATION
MAYNARD   MA 01754

RONALD F. BRENDER
BLISS LANGUAGE DEVELOPMENT
ML3-5/E82
DIGITAL EQUIPMENT CORP.
146 MAIN STREET
MAYNARD   MA 01754
(617) 897-5111 X2520

ALBERT S. BROWN
PK3-1/M12
DIGITAL EQUIPMENT CORP.
146 MAIN STREET
MAYNARD   MA 01754
(617) 897-5111 X2391

N. AMOS GILEADI
APPLIED SYSTEMS GROUP
ML 21-4 E-20
DIGITAL EQUIPMENT CORP.
146 MAIN STREET
MAYNARD   MA 01754
(617) 897-5111 X4402/X3888/X6472

RONALD J. HAM
ML5-5/E40
DIGITAL EQUIPMENT CORPORATION
146 MAIN STREET
MAYNARD   MA 01754
(617) 897-5111

WILLIAM F. SHAW
ML5-5/E40
DIGITAL EQUIPMENT CORPORATION
146 MAIN STREET
MAYNARD   MA 01754
(617) 897-5111

EDWARD STEEN
119 SHERMAN STREET
LOWELL   MA 01852
(617) 454-9320

AARON SAWYER
DEPT 330
THE FOXBORO COMPANY
FOXBORO   MA 02035
(617) 543-8750 X2029

WARREN R. BROWN
D.330
THE FOXBORO COMPANY
38 NEPONSET AVE.
FOXBORO   MA 02038
(617) 543-8750 X2023

VICTOR S. MILLER
DEPT OF MATHEMATICS
BLDG 2
U OF MASSACHUSETTS
HARBOR CAMPUS
BOSTON   MA 02125
(617) 287-1900 X3170/X3161

MICHAEL MEEHAN
WINTHROP PUBLISHERS
17 DUNSTER STREET
CAMBRIDGE   MA 02138
(617) 868-1750

ATTN: READING ROOM
INFORMATION PROCESSING CENTER
39-430
MIT
CAMBRIDGE   MA 02139

GABRIEL CHANG
575   TECHNOLOGY SQUARE
HONEYWELL INFORMATION SYSTEMS
CAMBRIDGE   MA 02139
(617) 491-6300

F. J. CORBATO
NE43-514
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
545 TECHNOLOGY SQUARE
CAMBRIDGE   MA 02139
(617) 253-6001

JEANNE FERRANTE
123 ANTRIM ST.
CAMBRIDGE   MA 02139
(617) 876-8635

R. STERLING EAMES
SOFTECH
460 TOTTEN POND ROAD
WALTHAM   MA 02154
(617) 890-6900

R. KRASIN
FIRST DATA CORP.
400 TOTTEN POND ROAD
WALTHAM   MA 02154
(617) 890-6701

DAVID SOLOMONT
COMPUTER SERVICES
MILLER HALL
TUFTS UNIVERSITY
MEDFORD   MA 02155
(617) 628-2943

PETER COLBY
289 MILL ST.
NEWTONVILLE   MA 02160
(617) 527-2394

CHRISTOPHER K. JOHANSEN
FREEKSHOW ELECTONWORKS
176 GROVE STREET
AUBURNDALE   MA 02166
(617) 969-2399

GEORGE POONEN
13 ORCHARD AVE.
WABAN   MA 02168
(617) 969-4684

DONALD E. GRIMES
90 SYLVIA STREET
ARLINGTON   MA 02174
(617) 646-4129

MICHAEL HAGERTY
18 HAMILTON ROAD
ARLINGTON   MA 02174
(617) 492-7100

RICHARD D. SPILLANE
DEPT OF MATH/C.S.
WILLIAM PATTERSON COL.
WAYNE   NJ 07470
(201) 881-2158

T. A. D'AURIA
CENTER FOR COMPUTING ACTIVITIES
COLUMBIA UNIVERSITY
NEW YORK   NY 10027

NEWTON J. MUNSON
COMPUTING CENTER
CLARKSON COLLEGE
POTSDAM   NY 13676
(315) 268-7721

TERRENCE M. COLLIGAN
RIVERSIDE OFFICE PARK
MANAGEMENT DECISION SYSTEMS INC.
RIVERSIDE ROAD
WESTON   MA 02193
(617) 891-0335

RON PRICE
PERKIN-ELMER DATA SYSTEMS
106 APPLE ST.
TINTON FALLS NJ 07726

WILLIAM BARABASH
DEPT. OF COMP. SCI.
SUNY STONY BROOK
STONY BROOK   NY 11794
(516) 246-7146

TED TENNY
COMPUTER SCIENCE DEPT.
SUNY - POTSDAM
POTSDAM   NY 13676
(315) 268-2954

DAVID J. GRIFFITHS
ACADEMIC COMPUTER CENTER
TYLER HALL
UNIVERSITY OF RHODE ISLAND
WEST WARWICK RI 02881
(401) 792-2701

RON OLSEN
ROOM 3E207
BELL LABORATORIES
HOLMDEL   NJ 07733
(201) 949-5537

GARRY MEYER
COMPUTING CENTER
SUNY STONY BROOK
STONY BROOK   NY 11794
(516) 246-7047

WILLIAM C. HOPKINS
207 RIDGEWOOD DRIVE
AMHERST   NY 14226
(716) 634-6348

ANDRIES VAN DAM
ANDRIES VAN DAM
BROWN UNIVERSITY
BOX F
PROVIDENCE   RI 02912
(401) 863-3088

STEVE LEGENHAUSEN
STEVE LEGENHAUSEN
12 BARNARD STREET
HIGHLAND PARK NJ 08904
(201) 572-6585

M. ELIZABETH IBARRA
M. ELIZABETH IBARRA
DEPT. OF APPLIED MATH
BROOKHAVEN NATIONAL LABORATORY
UPTON   NY 11973
(516) 345-4162

G. FRIEDER
G. FRIEDER
DEPT. OF COMPUTER SCIENCE
SUNY BUFFALO
4776 RIDGE LEA RD.
BUFFALO   NY 14226
(716) 831-1351

ATTENTION: R. D. BERGERON
DEPARTMENT OF MATHEMATICS
KINGSBURY HALL
U OF NEW HAMPSHIRE
DURHAM   NH 03824
(603) 862-2321

WILLIAM HENRY
117 E. TENTH ST.
NEW YORK   NY 10003

J. SCOTT MERRITT
36 OAKWOOD AVE.
TROY   NY 12180
(518) 271-7553

JAMES MOLONEY
DEPT. OF COMP. SCI.
SUNY BROCKPORT
BROCKPORT   NY 14420
(716) 395-2384

WILLIAM J. VASILIOU JR.
COMPUTER SERVICES
KINGSBURY HALL
U OF NEW HAMPSHIRE
DURHAM   NH 03824
(603) 862-2323

EDWARD R. FRIEDMAN
CIMS/CS DEPT.
NEW YORK UNIVERSITY
NEW YORK   NY 10012
(212) 460-7100

GEORGE H. WILLIAMS
EE/CS DEPT.
UNION COLLEGE
SCHENECTADY   NY 12308
(518) 370-6273

MICHAEL J. LUTZ
SCHOOL OF COMPUTER SCIENCE
ROCHESTER INSTITUTE OF TECHNOLOGY
ROCHESTER   NY 14623
(716) 464-2995

JAMES P. SHORES
344 GLENWOOD AVE.
NEW LONDON   CT 06320
(203) 442-0771 X2126

PETER PAWELCZAK
UNIVERSITY COMPUTER CENTER
C/O LIBRARY
CUNY
555 W. 57TH ST.
NEW YORK   NY 10019

J. L. POSDAMER
SCHOOL OF COMP. AND INFO. SCI.
313   LINK HALL
SYRACUSE U
SYRACUSE   NY 13210
(315) 423-4679

RICHARD CONWAY
DEPT. OF COMPUTER SCIENCE
CORNELL UNIVERSITY
ITHACA   NY 14850
(607) 256-3456

ROSEMARY HOWBRIGG
36 MENUNKETESUCK DRIVE
CLINTON   CT 06413
(203) 669-5812 (HOME)
(203) 442-0771 X2965 (WORK)

HOWARD D. ESKIN
CENTER FOR COMPUTING ACTIVITIES
ROOM 712
COLUMBIA UNIVERSITY
612 W. 115TH ST.
NEW YORK   NY 10025
(212) 280-2874

MICHAEL N. CONDICT
PATTERN ANALYSIS AND RECOGNITION CORP
ON THE MALL
ROME   NY 13440
(315) 336-8400 X36

JOHN H. WILLIAMS
OCS
418   UPSON HALL
CORNELL U
ITHACA   NY 14850
(607) 256-5033

MIKE LEMON
752   WEBSTER HALL
4415 FIFTH AVENUE
PITTSBURGH   PA 15213
(412) 621-6131
(412) 624-6454

GARY LINDSTROM
COMPUTER SCIENCE DEPT.
U OF PITTSBURGH
PITTSBURGH   PA 15260
(412) 624-6455

MARY LOU SOFFA
COMPUTER SCI. DEPT.
335   ALUMNI HALL
UNIVERSITY OF PITTSBURGH
PITTSBURGH   PA 15260
(412) 624-6454

HOWARD E. TOMPKINS
COMPUTER SCIENCE DEPT
INDIANA UNIVERSITY OF PA
INDIANA   PA 15701
(415) 357-2524

ATTENTION: RUTH DROZIN
FREAS-ROOKE COMPUTER CENTER
BUCKNELL UNIVERSITY
LEWISBURG   PA 17837
(717) 524-1436

DANIEL C. HYDE
COMPUTER SCIENCE PROGRAM
BUCKNELL UNIVERSITY
LEWISBURG   PA 17837
(717) 524-1392

JOHN W. ADAMS
DEPT. OF I.E.
19   PACKARD LAB
LEHIGH UNIV.
BETHLEHEM   PA 18015

DAVE ENGLANDER
302 SUMMIT STREET
BETHLEHEM   PA 18015
(215) 865-9027

S. L. GULDEN
DEPT. OF MATH
LEHIGH UNIVERSITY
BETHLEHEM   PA 18015
(215) 691-7000 X341
(215) 691-7000 X341

V. LALITA RAO
506 W. THIRD STREET APT. 4
BETHLEHEM   PA 18015
(215) 865-6448

RAMON TAN
P.O. BOX 2
BETHLEHEM   PA 18016
(215) 866-7195

STEPHEN TITCOMB
1111 NORTH BLVD.
BETHLEHEM   PA 18017

RANCE J. DELONG
MORAVIAN COLLEGE
BETHLEHEM   PA 18018

MARILYN HOFFMAN
531 W. UNION BLVD.
BETHLEHEM   PA 18018
(215) 865-6937

JOHN A. WEAVER
2112 PENNSYLVANIA AVE. F-6
BETHLEHEM   PA 18018
(215) 867-1085

JOSEPH A. MEZZAROBA
BOX 164
E. GREENVILLE PA 18041
(215) 691-7000 (OFFICE)
(215) 679-9900 (HOME)

RICHARD J. CICHELLI
901 WHITTIER DRIVE
ALLENTOWN   PA 18103
(215) 797-9880

CHESTER J. SALWACH
2124 DIAMOND STREET
SELLERSVILLE PA 18960
(215) 723-8301

ROBERT KEZELL
UNIVERSITY COMPUTER ACTIVITY
TEMPLE UNIVERSITY
PHILADELPHIA PA 19122
(215) 787-8527

FRANK RYBICKI
COMPUTER ACTIVITY
TEMPLE UNIVERSITY
BROAD AND MONTGOMERY
PHILADELPHIA PA 19122

JOHN T. LYNCH
BURROUGHS CORP.
P.O. BOX 517
PAOLI   PA 19301

E. L. ROWE
BURROUGHS CORP.
BOX 517
PAOLI   PA 19301
(215) 648-2218

JOHN D. EISENBERG
COMPUTING CENTRE
SMITH HALL
U OF DELAWARE
NEWARK   DE 19711
(302) 738-8441 X57 (OFFICE)
(302) 453-9059 (HOME)

WILLIAM Q. GRAHAM
COMPUTING CENTER
U. OF DELAWARE
13 SMITH HALL
NEWARK   DE 19711
(302) 738-8441

C. E. BRIDGE
ENGINEERING DEVELOPMENT LAB
E. I. DU PONT DE NEMOURS AND CO.
101 BEECH STREET
WILMINGTON   DE 19898
(302) 774-1931

STEPHEN O. SCHWARM
E.I. DU PONT DE NEMOURS CO.
101 BEECH ST.
WILMINGTON   DE 19898
(302) 774-1669

MIKE FRAME
FIRST DATA CORP.
2011 EYE ST. NW
WASHINGTON   DC 20006
(202) 872-0580

TERRY P. MEDLIN
SCIENTIFIC RESEARCH UNIT - DPSA
NATIONAL INSTITUTE OF DENTAL HEALTH
BETHESDA   MD 20014

WAYNE RASBAND
BLDG 36 ROOM 2A-03
NATIONAL INSTITUTES OF HEALTH
BETHESDA   MD 20014
(301) 496-4957

DAVID A. GOMBERG
DEPT. OF MATH. STAT. AND COMP. SCI.
AMERICAN UNIVERSITY
MASSACHUSETTS & NEBRASKA AVES.
WASHINGTON   DC 20016
(202) 686-2393

ARTHUR A. BROWN
1101 NEW HAMPSHIRE AVE. NW APT.1002
WASHINGTON   DC 20037
(202) 785-0716

PETER A. RIGSBEE
CODE 5494
NAVAL RESEARCH LABORATORY
WASHINGTON   DC 20375
(202) 767-3181

THOMAS A. KEENAN
DIVISION OF MATHEMATICAL AND COMPUTER
NATIONAL SCIENCE FOUNDATION
WASHINGTON   DC 20550
(202) 632-7346

CAROL A. OGDIN
SOFTWARE TECHNIQUE INC.
100 POMMANDER WALK
ALEXANDRIA   VA 22314
(703) 549-0646

GREGORY J. WINTERHALTER
DIGITAL COMMUNICATIONS
135 TECHNOLOGY PARK
NORCROSS   GA 30092
(404) 448-1400

ATTN: DIRECTOR
NORTHEAST REGIONAL DATA CENTER
253   SSRB
U OF FLORIDA
GAINESVILLE   FL 32611
(904) 392-2061

SHMUEL PELEG
COMPUTER SCIENCE CENTER
U OF MARYLAND
COLLEGE PARK MD 20742

ATTN: J. F. MCINTYRE - LIBRARIAN
COMPUTING CENTER
GILMER HALL
U OF VIRGINIA
CHARLOTTESVIL VA 22904
(804) 924-3731

ATTENTION: JERRY W. SEGERS
OFFICE OF COMPUTING SERVICES
GEORGIA INSTITUTE OF TECHNOLOGY
ATLANTA   GA 31333
(404) 894-4818

ATTN: LIBRARIAN
CIRCA
411   WEIL
U OF FLORIDA
GAINESVILLE   FL 32611
(904) 392-0907
(904) 392-0907

BEN SHNEIDERMAN
DEPT. OF INFO. SYS. MGMT.
U OF MARYLAND
COLLEGE PARK MD 20742

STEPHEN J. HARTLEY
113 FERNCLIFF DR.
WILLIAMSBURG VA 23185
(804) 229-0337 (HOME)
(804) 827-2897 (WORK)

GERALD N. CEDERQUIST
ELECTRONICS RESEARCH BLDG
EES STL/ASD
GEORGIA INSTITUTE OF TECHNOLOGY
ATLANTA   GA 30332
(404) 894-3417

FRED L. SCOTT
BROWARD COMMUNITY COLLEGE
3501 DAVIE ROAD
FT. LAUDERDAL FL 33314
(305) 581-8700

JACOB C. Y. WU
SYSTEM SCIENCES DIVISION
COMPUTER SCIENCES CORPORATION
8728 COLESVILLE ROAD
SILVER SPRING MD 20910
(301) 589-1545 X276

ANN D. DAVIES
UNIVERSITY COMPUTER CENTER
VIRGINIA COMMONWEALTH UNIVERSITY
1015 FLOYD AVE.
RICHMOND   VA 23284
(804) 770-6339

PHILLIP H. ENSLOW JR.
SCHOOL OF INFO. AND COMP. SCI.
GEORGIA TECH
ATLANTA   GA 30332
(404) 894-3152

DONALD B. CROUCH
DEPT.OF COMPUTER SCIENCE
U. OF ALABAMA
P.O. BOX 6316
UNIVERSITY   AL 35486
(205) 348-6363

RAINER F. MCCOWN
MCCOWN COMPUTER SERVICES
9537 LONG LOOK LANE
COLUMBIA   MD 21045

DAVID A. HOUGH
529 HELM DRIVE
NEWPORT NEWS VA 23602
(804) 874-3387

JAMES N. FARMER
OFFICE OF COMPUTING SERVICES
GEORGIA TECH
225 NORTH AVE. NW
ATLANTA   GA 30332
(404) 894-4660

PHILIP N. BERGESTESSER
128 JACKSON AVE.
MADISON   AL 35758
(205) 837-2400

JOE C. ROBERTS
613 MARKET STREET
POCOMOKE CITY MD 21851
(804) 824-3411 X641

J. C. KNIGHT
LANGLEY RESEARCH CENTER
M/S 125A
NASA
HAMPTON   VA 23665

JOHN J. GODA JR.
SCHOOL OF INFORMATION AND COMPUTER SCI
GEORGIA TECH
ATLANTA   GA 30332
(404) 894-3131

ATTENTION: DAVID MADISON
ADVANCED SOFTWARE TECHNOLOGY DEPT.
TEXAS INSTRUMENTS INC.
304 WYNN DRIVE
HUNTSVILLE   AL 35806
(205) 837-7510

ANDREW S. PUCHRIK
11623 CHARTER OAKS #202
RESTON   VA 22090

FRED W. POWELL
COMPUTER CENTER
MARY BALDWIN COLLEGE
STAUNTON   VA 24401
(703) 885-0811

JOHN P. WEST
OFFICE OF COMPUTING SERVICES
GEORGIA TECH
225 NORTH AVE. N.W.
ATLANTA   GA 30332
(404) 894-4676

SAMUEL T. BAKER
1310 STONEWALL BLVD.
MURFREESBORO TN 37130
(615) 896-3362 (HOME)
(615) 741-3531 (OFFICE)

FRANK BREWSTER
4701 KENMORE AVE #1009
ALEXANDRIA   VA 22304
(703) 370-6645

STEVEN M. BELLOVIN
DEPT. OF COMP. SCI.
U OF NORTH CAROLINA
CHAPEL HILL NC 27514
(919) 933-5698

T. P. BAKER
DEPT. OF MATH
225   LOVE BUILDING
FLORIDA STATE U
TALLAHASSEE   FL 32304
(904) 644-2580

ATTENTION: GORDON R. SHERMAN
COMPUTER CENTER
200   STOKELY MGMT. CENTER
U OF TENNESSEE
KNOXVILLE   TN 37916

CHARLES PFLEEGER
COMP. SCI. DEPT.
U OF TENNESSEE
KNOXVILLE   TN 37916
(615) 974-5067

ATTN: DEPT. OF COMPUTER SCIENCE
U OF MISSISSIPPI
UNIVERSITY   MS 38677

GAY THOMAS
COMPUTER SCIENCE DEPT.
DRAWER CC
MISS. STATE   MS 39762
(601) 325-2942

LAVINE THRAILKILL
COMPUTING CENTER
U OF KENTUCKY
LEXINGTON   KY 40506
(606) 258-2916

ROY F. REEVES
1640 SUSSEX COURT
COLUMBUS   OH 43220
(614) 422-4843

R. B. LAKE
BIOMETRY
WEARN BUILDING
UNIVERSITY HOSPITALS
CLEVELAND   OH 44106
(216) 791-7300

T. S. HEINES
DEPT. OF COMPUTER SCIENCE
CLEVELAND STATE UNIVERSITY
CLEVELAND   OH 44115
(216) 687-4762
(216) 687-4760

ROBERT L. BRIECHLE
THE COMPUTER CENTER
U OF AKRON
302 E. BUCHTEL AVE.
AKRON   OH 44325
(216) 375-7172

E. C. ZIMMERMAN
COMPUTER CENTER
THE COLLEGE OF WOOSTER
WOOSTER   OH 44691
(216) 264-1234 X304

PATRICIA VAN DERZEE
PROCESS CONTROLS DIVISION
CINCINNATI MILACRON INC.
LEBANON   OH 45036
(513) 494-5320

ROBERT J. SNYDER
GR.FL. UNION BUILDING DATA CENTER
INDIANA U - PURDUE U AT INDIANAPOLIS
1100 WEST MICHIGAN STREET
INDIANAPOLIS IN 46202

ATTN: DOCUMENTS ROOM LIBRARIAN
COMPUTING CENTER
U OF NOTRE DAME
NOTRE DAME   IN 46637
(219) 283-7784

DOUGLAS H. QUEBBEMAN
2235 LOMBARDY DRIVE
JEFFERSONVILL IN 47130
(812) 945-2731

GEORGE COHN III
316 N. WASHINGTON
BLOOMINGTON   IN 47401
(812) 337-9255
(812) 337-1911

HAL STEIN
BOX 102 WRIGHT QUAD
INDIANA UNIVERSITY
BLOOMINGTON   IN 47401
(812) 337-7081

ALFRED I. TOWELL
WRUBEL COMPUTER CENTER
INDIANA UNIVERSITY
BLOOMINGTON   IN 47401
(812) 337-1911

DAVID S. WISE
101   LINDLEY HALL
INDIANA U
BLOOMINGTON   IN 47401
(812) 337-4866

STEPHEN W. YOUNG
WRUBEL COMPUTER CENTER
HPER BUILDING
INDIANA UNIVERSITY
BLOOMINGTON   IN 47401
(812) 337-1911

JAMES R. MILLER
425-21 SOUTH RIVER ROAD
W. LAFAYETTE   IN 47906
(317) 494-8232 (OFFICE)

EDWARD F. GEHRINGER
DEPT. OF COMPUTER SCIENCE
MATH SCIENCES BUILDING
PURDUE UNIVERSITY
LAFAYETTE   IN 47907

JOSEPH H. FASEL III
COMPUTER SCIENCES
442   MATH SCIENCES BUILDING
PURDUE UNIVERSITY
W. LAFAYETTE   IN 47907
(317) 494-8566

ALAN A. KORTESOJA
701 W. DAVIS
ANN ARBOR   MI 48103
(313) 995-6124
(313) 995-6000

L. RICHARD LEWIS
5806 COOLIDGE ROAD
DEARBORN   MI 48127
(313) 274-6871

WILLIAM GROSKY
MATH DEPT - COMP. SCI. SECTION
WAYNE STATE UNIVERSITY
DETROIT   MI 48202

RONALD G. MOSIER
17596 WILDEMERE
DETROIT   MI 48221
(313) 956-2417

R. NEIL FAIMAN JR.
8235 APPOLINE
DETROIT   MI 48228

MARK HERSEY
323 VILLAGE DRIVE APT. 534
EAST LANSING MI 48823
(517) 351-5703 (HOME)
(517) 355-1764 (OFFICE)

THOMAS C. SOCOLOFSKY
SYSTEMS RESEARCH INC
241 E. SAGINAW
EAST LANSING MI 48823
(517) 351-2530 (OFFICE)
(517) 351-2530 (HOME)

JOHN B. EULENBERG
COMP. SCI. DEPT.
MICHIGAN STATE U
EAST LANSING MI 48824
(517) 353-0831

STEVEN L. HUYSER
COMPUTER LABORATORY
MICHIGAN STATE U
EAST LANSING MI 48824
(517) 353-1800

MARK RIORDAN
USER SERVICES
COMPUTER LABORATORY
MICHIGAN STATE UNIVERSITY
EAST LANSING MI 48824
(517) 353-1800

H. G. HEDGES
DEPT. OF COMP. SCI.
MICHIGAN STATE U
E. LANSING   MI 48824
(517) 353-6484

GORDON A. STEGINK
COMPUTER CENTER
325   MANITOU HALL
GRAND VALLEY STATE COLLEGE
ALLENDALE   MI 49401
(616) 895-6611 X571

GEORGE O. STRAWN
DEPT. OF COMPUTER SCIENCE
IOWA STATE U
AMES   IA 50011
(515) 294-2259

ATTN: SERIALS DEPT.
UNIVERSITY LIBRARIES
UNIVERSITY OF IOWA
IOWA CITY   IA 52242

ATTN: UCC LIBRARIAN
UNIVERSITY COMPUTER CENTER
LCM
UNIVERSITY OF IOWA
IOWA CITY   IA 52242
(319) 353-3170

W. A. HINTON
SAN W 570 C
U OF WISCONSIN - MILWAUKEE
P.O. BOX 413
MILWAUKEE   WI 53201
(414) 963-4005

BROOKS DAVID SMITH
4473 N. NEWHALL ST.
SHOREWOOD   WI 53211
(414) 332-6646

HERMAN BERG
108 E. DAYTON STREET
MADISON   WI 53703

ATTN: FRIEDA S. COHEN
ACADEMIC COMPUTING CENTER
U OF WISCONSIN
1210 W. DAYTON ST.
MADISON   WI 53706

CHARLES N. FISCHER
MACC
U OF WISCONSIN
1210 WEST DAYTON ST.
MADISON   WI 53706
(608) 262-7870

FRANK H. HORN
ACADEMIC COMPUTER CENTER
U OF WISCONSIN
1210 WEST DAYTON STREET
MADISON   WI 53706
(608) 262-9841

ED GLASER
COMPUTING SERVICES
U OF WISCONSIN - GREEN BAY
GREEN BAY   WI 54302
GREEN BAY   WI 54302
(414) 465-2309

DAVID A. NUESSE
DEPARTMENT OF COMPUTER SCIENCE
U OF WISCONSIN - EAU CLAIRE
EAU CLAIRE   WI 54701
(715) 836-2526

RUDOLPH C. POLENZ
INFORMATION SYSTEMS AND COMPUTING SERV
U OF WISCONSIN - EAU CLAIRE
EAU CLAIRE   WI 54701
(715) 836-4428

BRUCE A. PUMPLIN
DEPT OF COMPUTER SCIENCE
U OF WISCONSIN - EAU CLAIRE
EAU CLAIRE   WI 54701
(715) 836-2315

CARL HENRY
COMPUTER CENTER
CARLETON COLLEGE
NORTHFIELD   MN 55057
(507) 645-4431 X504

TIMOTHY W. HOEL
ACADEMIC COMPUTER CENTER
ST. OLAF COLLEGE
NORTHFIELD   MN 55057
(507) 663-3096

GLENN MILLER
2317 N. HENRY ST.
N. ST. PAUL   MN 55109
(612) 777-2483

MARK RUSTAD
525 HARRIET AVE #213
ST. PAUL   MN 55112

ROBERT D. VAVRA
741 TERRACE DRIVE
ROSEVILLE   MN 55113
(612) 483-6123

STEVE M. WEINGART
MS 4953
SPERRY-UNIVAC
2276 HIGHCREST DRIVE
ROSEVILLE   MN 55113

PETER H. ZECHMEISTER
926 W. MONTANA AVE.
ST. PAUL   MN 55117
(612) 489-5166

ATTENTION: ROBERT E. NOVAK
DSPL DEVELOPMENT GROUP
SPERRY UNIVAC
UNIVAC PARK / P.O. BOX 3525
ST. PAUL   MN 55165
(612) 456-5551

LEO J. SLECHTA
DSD
SPERRY UNIVAC
BOX 3525 MS U1U25
ST. PAUL   MN 55165
(612) 456-2743

DAVID HELFINSTINE
1136 5TH AVENUE SOUTH
ANOKA   MN 55303
(612) 421-8964

PAUL CHRISTOPHERSON
M.S. MN11-1611
HONEYWELL INC.
600 SECOND STREET N.
HOPKINS   MN 55343
(612) 542-6438

MARK BILODEAU
ENGINEERING SYSTEMS 4TH FLOOR
NORTHERN STATES POWER
414 NICOLLET MALL
MINNEAPOLIS   MN 55401
(612) 330-6749
(612) 330-5899

CHRIS EASTLUND
ENGINEERING SYSTEMS 4TH FLOOR
NORTHERN STATES POWER
414 NICOLLET MALL
414 NICOLLET MALL
MINNEAPOLIS   MN 55401
(612) 330-6749
(612) 330-5899

JOHN URBANSKI
1929 FREMONT AVE. S. APT. 23
MINNEAPOLIS   MN 55403
(612) 377-3198

SCOTT BERTILSON
2929 42ND AVE. S.
MINNEAPOLIS   MN 55406
(612) 729-0059

INDULIS VALTERS
2810 E 22ND AVE.
MINNEAPOLIS   MN 55406
(612) 341-4430

DON HAMNES
4215 PLEASANT AVE. SO.
MINNEAPOLIS   MN 55409
(612) 823-3030

JOHN FUNG
425 13TH AVE S.E. #406
MINNEAPOLIS   MN 55414
(612) 376-5464 (OFFICE)
(612) 378-0427 (HOME)

WALT PERKO
727 15TH AVE. S.E.
MINNEAPOLIS   MN 55414
(612) 331-6984

GEOFF WATTLES
407 ERIE STREET
MINNEAPOLIS   MN 55414
(612) 331-7087

KEITH HAUER-LOWE
4819 COLUMBUS AVE. SO.
MINNEAPOLIS   MN 55417
(612) 633-6170 X3362 (WORK)
(612) 824-8026 (HOME)

JONATHON R. GROSS
DIGITAL EQUIPMENT CORP.
8030 CEDAR AVENUE SOUTH
MINNEAPOLIS   MN 55420
(612) 854-6562

ROBERT A. STRYK
5441 HALIFAX LANE
EDINA   MN 55424
(612) 920-5434 (HOME)
(612) 887-4356 (OFFICE)

RON THOMAS
7725 WASHINGTON AVE. S.
MINNEAPOLIS   MN 55425
(612) 941-6500

HUGO MEISSER
3021 WISCONSIN AVE. N.
MINNEAPOLIS   MN 55427
(612) 544-2349

RANDALL W. WARREN
HQS06B
CONTROL DATA CORPORATION
P.O. BOX 0
MINNEAPOLIS   MN 55440

TIM BONHAM
D605/1630 S. 6TH ST.
MINNEAPOLIS   MN 55454
(612) 339-4405

R. K. NORDIN
1615 SOUTH 4TH ST. APT.M3607
MINNEAPOLIS   MN 55454
(612) 339-5232 (HOME)
(612) 482-3751 (OFFICE)

ATTENTION: PAUL C. SMITH
CONSULTING GROUP ON INSTRUCTIONAL DESI
205   ELLIOTT HALL
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-5352

ATTENTION: STEVE REISMAN
SCH. OF DENTISTRY/CLINICAL SYS. DIV.
8-440 HEALTH SCIENCE UNIT A
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 376-4131

ATTN: COMPUTER SCIENCE DEPT.
114   LIND HALL
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-0132

ATTN: REFERENCE ROOM
UNIVERSITY COMPUTER CENTER
227   EXP ENGR
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-7744

KEN BORGENDALE
C.SCI. DEPT.
114   LIND HALL
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 824-3389

JEFFREY J. DRUMMOND
UNIVERSITY COMPUTER CENTER
LAUDERDALE
U OF MINNESOTA
MINNEAPOLIS   MN 55455
(612) 373-4573

JOHN T. EASTON
SSRFC
25G   BLEGEN HALL
U OF MINNESOTA
WEST BANK
MINNEAPOLIS   MN 55455
(612) 373-9917

LINCOLN FETCHER
UNIVERSITY COMPUTER CENTER
227   EXPERIMENTAL ENGINEERING
UNIVERSITY OF MINNESOTA
MINNEAPOLIS   MN 55455
(612) 376-1631

KEVIN FJELSTAD
UNIVERSITY COMPUTER CENTER
227   EXP. ENGR.
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-4181

K. FRANKOWSKI
COMPUTER SCIENCE DEPARTMENT
110H   LIND HALL
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-7591

KRISTINA GREACEN
C.SCI. DEPT.
114   LIND HALL
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455

JOEL M. HALPERN
UNIVERSITY COMPUTER CENTER
227   EXP. ENGR.
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-4181

BRIAN HANSON
UNIVERSITY COMPUTER CENTER
227   EXP. ENGR.
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 376-5262 (OFFICE)

THEA D. HOOGE
UNIVERSITY COMPUTER CENTER
227   EXP. ENGR.
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-4599

TIMOTHY J. HOFFMANN
364   FRONTIER HALL
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-6957

GEORGE D. JELATIS
BOX 15 MAYO
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-8941

DAN LALIBERTE
UNIVERSITY COMPUTER CENTER
227   EXP. ENGR.
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-4181

LAWRENCE A. LIDDIARD
UNIVERSITY COMPUTER CENTER
227   EXP. ENG. BLDG.
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-5239

DENNIS R. LIENKE
UNIVERSITY COMPUTER CENTER
214   EXP. ENGR.
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-1572

SHIHTA LIN
UNIVERSITY COMPUTER CENTER
227   EXP ENGR
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-4886

JOHN E. LIND
139   TERRITORIAL HALL
UNIVERSITY OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455

MICHAEL ROBERT MEISSNER
1541   PIONEER HALL
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455

DAVID C. MESSER
UNIVERSITY COMPUTER CENTER
227   EXPERIMENTAL ENGINEERING
UNIVERSITY OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 376-2787

ANDY MICKEL
UNIVERSITY COMPUTER CENTER
227   EXP. ENGR.
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 376-7788

JAMES F. MINER
JAMES F. MINER
SSRFC
25   BLEGEN HALL
U OF MINNESOTA
WEST BANK
MINNEAPOLIS   MN 55455
(612) 373-9918

JOHN NAUMAN
901   MIDDLEBROOK HALL
U OF MINNESOTA
WEST BANK
MINNEAPOLIS   MN 55455
(612) 376-6596

DAVID PERLMAN
COMPUTER SCIENCE DEPARTMENT
114   LIND HALL
UNIVERSITY OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-7581

HERBERT RUBENSTEIN
UNIVERSITY COMPUTER CENTER
227   EXP. ENGR.
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-4181

TIMOTHY J SALO
UNIVERSITY COMPUTER CENTER
LAUDERDALE
U OF MINNESOTA
MINNEAPOLIS   MN 55455
(612) 376-5607

BOB SCARLETT
PHYSICS DEPT.
148   PHYSICS
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-0243

G. MICHAEL SCHNEIDER
C.SCI. DEPT.
114   LIND HALL
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-7582

JOHN P. STRAIT
UNIVERSITY COMPUTER CENTER
227   EXP. ENGR.
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-7788

BILL WOOD
BILL WOOD
C.SCI. DEPT.
114   LIND HALL
U OF MINNESOTA
EAST BANK
MINNEAPOLIS   MN 55455
(612) 373-7746

ATTN: SSRFC LIBRARY
SSRFC
25   BLEGEN HALL
U OF MINNESOTA
WEST BANK
MINNEPOLIS   MN 55455
(612) 373-5599

MITCHELL R. JOELSON
SSRFC
25   BLEGEN HALL
U OF MINNESOTA
WEST BANK
MPLS.   MN 55455
(612) 781-7323 (HOME)

DAVID SARANEN
117 7TH ST. SO.
VIRGINIA   MN 55792
(218) 741-1378

ATTENTION: DAN BURROWS
UMD COMPUTER CENTER
178   M.W.ALWORTH HALL
U OF MINNESOTA
DULUTH
DULUTH   MN 55812

MARK LUKER
MATH SCIENCES DEPT.
U OF MINNESOTA
DULUTH
DULUTH   MN 55812
(218) 726-8240

L. W. YOUNGREN
1505 N.W. 41ST ST. APT. 18F
ROCHESTER   MN 55901
(507) 285-9696

JAMES F. MARTINSON
1210 WILLMAR AVE
WILLMAR   MN 56201
(612) 796-2147

R. WARREN JOHNSON
R. WARREN JOHNSON
DEPT. OF MATH AND COMP. SCI.
ST. CLOUD STATE U
ST. CLOUD   MN 56301
(612) 255-2147

HAROLD DE VORE
BOX 7161 UNIVERSITY STATION
GRAND FORKS   ND 58202
(701) 746-6977

R. I. JOHNSON
COMP. SCI. DEPT.
U OF NORTH DAKOTA
BOX 181 UNIVERSITY STATION
GRAND FORKS   ND 58202
(701) 777-4107

GARY J. BOOS
517 N. 7TH STREET
BISMARK   ND 58501
(701) 223-0441 (WORK)

ATTENTION: KYU LEE
DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF MONTANA
MISSOULA   MT 59801
(406) 243-2883

MARK S. NIEMCZYK
HEWITT ASSOCIATES
102 WILMOT ROAD
DEERFIELD   IL 60015
(312) 945-8000

ALBERT STEINER
VOGELBACK COMPUTING CENTER
NORTHWESTERN U
2129 SHERIDAN ROAD
EVANSTON   IL 60201
(312) 492-3682

BRIT J. BARTTER
850A FOREST AVENUE
EVANSTON   IL 60202

JONATHAN SACHS
JONATHAN SACHS
TRANS UNION SYSTEMS CORPORATION
111 WEST JACKSON BLVD
CHICAGO   IL 60604
(312) 431-3330

DAVID E. CARLTON
DEPT. OF INFO. SCI.
NORTHEASTERN ILLINOIS U
5500 N. ST. LOUIS AVE.
CHICAGO   IL 60625

TED FISHMAN
6049 KIMBALL
CHICAGO   IL 60659

ATTN: CONSULTING OFFICE
COMPUTING SERVICES OFFICE
116   DIGITAL COMPUTER LAB
U OF ILLINOIS
URBANA   IL 61801
(217) 333-6133

RICHARD BALOCCA
114B   DIGITAL COMPUTER LAB
U OF ILLINOIS
URBANA   IL 61801
(217) 344-5284

ROGER GULBRANSON
PHYSICS DEPT.
U OF ILLINOIS
URBANA   IL 61801
(217) 367-8470 (HOME)
(217) 333-3191 (OFFICE)

CHARLES HEDRICK
183   COMMERCE WEST
U OF ILLINOIS
URBANA   IL 61801
(217) 333-4515
(217) 356-8425

M. D. MICKUNAS
297   DCL
U OF ILLINOIS
URBANA   IL 61801
(217) 333-6351

CARLTON MILLS
MILLS INTERNATIONAL
203 NORTH GREGORY
URBANA   IL 61801
(217) 328-2436 (HOME)
(217) 333-6971

ATTN: RECEIVING CLERK
CERL - SOC
U.S. ARMY
P.O. BOX 4005
CHAMPAIGN   IL 61820
(217) 352-6511

FRED P. BAKER
302 E. GREGORY
CHAMPAIGN   ,   IL 61820
(217) 344-7511

AVRUM ITZKOWITZ
505 E. CLARK APT. 22
CHAMPAIGN   IL 61820
(217) 359-9644 (HOME)
(217) 352-6511 (WORK)

DANIEL M. O'BRIEN
601 E. CLARK #10
CHAMPAIGN   IL 61820
(217) 356-2718

JACK THOMPSON
MID. ILLINOIS COMPUTER CO-OP
COTTONWOOD ROAD
EDWARDSVILLE IL 62025
(618) 288-7268

DENNIS S. ANDREWS
INFORMATION PROCESSING
SOUTHERN ILLINOIS UNIVERSITY
CARBONDALE   IL 62901

LARRY D. LANDIS
UNITED COMPUTING SYSTEMS
2525 WASHINGTON
KANSAS CITY   MO 64108
(816) 942-6065

JEFFERY M. RAZAFSKY
UNITED COMPUTING SYSTEMS INC.
500 W. 26TH STREET
KANSAS CITY   MO 64108
(816) 221-9700

HOWARD D. PYRON
MATH - C.SCI.
U OF MISSOURI - ROLLA
ROLLA   MO 65401
(314) 341-4491

STEVEN S. MUCHNICK
DEPARTMENT OF COMPUTER SCIENCE
U OF KANSAS
LAWRENCE   KS 66045

LYNNE J. BALDWIN
DEPT. OF MATH/COMP. SCI.
U OF NEBRASKA
BOX 688
OMAHA   NE 68101
(402) 554-2836

TERRY E. WEYMOUTH
C/O J.W.WEYMOUTH
6110 MEADOWBROOK LANE
LINCOLN   NE 68510

SHARAD C. SETH
DEPT. OF COMP. SCI.
U OF NEBRASKA
LINCOLN   NE 68588
(402) 472-3488

D. B. KILLEEN
COMPUTER LAB
RICHARDSON BLDG.
TULANE UNIVERSITY
NEW ORLEANS   LA 70118

FREDERICK A. HOSCH
COMPUTER RESEARCH CENTER
U OF NEW ORLEANS
NEW ORLEANS   LA 70122

WARREN JOHNSON
U OF SOUTHWESTERN LOUISIANA
BOX 4-2770 USL STATION
LAFAYETTE   LA 70504
(318) 234-7349

ED KATZ
COMPUTER SCIENCE DEPT.
U OF SOUTHWESTERN LOUISIANA
BOX 4-4330 USL STATION
LAFAYETTE   LA 70504

STEVE LANDRY
COMPUTER CENTER
U OF SOUTHWESTERN LOUISIANA
P.O. BOX 4-2770
LAFAYETTE   LA 70504
(318) 234-7349

DAVID LANDSKOV
U OF SOUTHWESTERN LOUISIANA
USL BOX 4-4154
LAFAYETTE   LA 70504
(318) 234-7640

A. I. STOCKS
P.O. BOX 4-1039
USL STATION
LAFAYETTE   LA 70504
(318) 233-3850 X538

TERRY M. WALKER
COMPUTER SCIENCE DEPT.
U OF SOUTHWESTERN LOUISIANA
P.O. BOX 4-4330
LAFAYETTE   LA 70504
(318) 234-7610

JOHN MANNALLY
HARDING COLLEGE
BOX 744
SEARCY   AR 72143
(501) 268-6161 X257

RICHARD V. ANDREE
MATH DEPT.
U OF OKLAHOMA
NORMAN   OK 73019
(405) 325-3410

RALPH HOMENSTINE
2313 CRESTMONT #208
NORMAN   OK 73069

STEPHEN A. PITTS
305 EAST JARMAN DRIVE
MIDWEST CITY OK 73110
(405) 732-4060

GILBERT J. HANSEN
3104 BONNIEBROOK DRIVE
PLANO   TX 75075
(214) 423-7837

BRIAN W. JOHNSON
1525 WESTLAKE
PLANO   TX 75075
(214) 690-2885

DENNIS J. FRAILEY
COMP. SCI. DEPT.
SOUTHERN METHODIST UNIV.
DALLAS   TX 75222

W. J. MEYERS
4-2148 THE TIMBERS
13447 N. CENTRAL EXPR.
DALLAS   TX 75243
(214) 231-4869

BARY PEHRNQUIST
SOUTHERN METHODIST UNIV.
BOX 2119
BOX 2112   TX 75275
DALLAS   TX 75275

RICHARD HUBER
DEPT. OF INDUSTRIAL ENGINEERING
TEXAS A&M UNIVERSITY
TEXAS A&M UNIVERSITY
COLL. STATION TX 77843
(713) 845-5531 X256

WILLIAM L. HANNAH
SUITE 211
S/B/P & C ASSOCIATES
S/B/P & C ASSOCIATES
8705 SHOAL CREEK BLVD.
AUSTIN   TX 78758
(512) 458-2276

WILLIAM M. WAITE
ELECTRICAL ENGINEERING DEPT.
SOFTWARE ENGINEERING GROUP
SOFTWARE ENGINEERING GROUP
UNIVERSITY OF COLORADO
BOULDER   CO 80302

JANET TAYLOR
USER SERVICES
COMPUTING LABORATORY
SOUTHERN METHODIST UNIVERSITY
DALLAS   TX 75275
(214) 692-2900

MIKE GREEN
DATAPOINT CORPORATION
9725 DATAPOINT DRIVE
SAN ANTONIO   TX 78284
(512) 690-7345

HARRY P. HAIDUK
DEPT. OF COMP. INFO. SYSTEMS
WEST TEXAS STATE U
CANYON   TX 79015
(806) 656-3966

LLOYD D. FOSDICK
DEPARTMENT OF COMPUTER SCIENCE
ECOT 7-7
U OF COLORADO
BOULDER   CO 80309
(303) 492-7514

JESSE D. MIXON
DEPT. OF COMPUTER SCIENCE
STEPHEN F. AUSTIN STATE U
P.O. BOX 6167 SFA STATION
NACOGDOCHES   TX 75961
(713) 569-2508

WILLETT KEMPTON
2512 SAN GABRIEL ST.
AUSTIN   TX 78705

MAURICE BALLEW
COMPUTER SERVICES
TEXAS TECH UNIVERSITY
BOX 4519
LUBBOCK   TX 79409
(806) 742-2900

GEORGE H. RICHMOND
COMPUTING CENTER
UNIVERSITY OF COLORADO
3645 MARINE STREET
BOULDER   CO 80309
(303) 492-6934

GINGER KELLY
ICSA
RICE UNIVERSITY
HOUSTON   TX 77001
(713) 527-....

ATTN: DOROTHY SMITH - REFERENCE LIBRAR
COMPUTATION CENTER
U OF TEXAS AUSTIN
AUSTIN   TX 78712
(512) 471-3242

LEONARD H. WEINER
DEPT. OF MATH AND COMP. SCI.
TEXAS TECH. U
P.O. BOX 4319
LUBBOCK   TX 79409
(806) 742-2571

ATTN: USER SERVICES GROUP
UNIVERSITY COMPUTER CENTER
COLORADO STATE U
FORT COLLINS CO 80523
(303) 491-5133

JOHN EARL CRIDER
7201 BROMPTON ROAD #A114
HOUSTON   TX 77025
(713) 665-3016

WILHELM BURGER
DEPT. OF COMPUTER SCIENCES
328   PAINTER HALL
U OF TEXAS AUSTIN
AUSTIN   TX 78712
(512) 471-4088
(512) 471-7316

D. A. CAUGHFIELD
609 E. N. 21ST
ABILENE   TX 79601
(915) 672-1604

DALE H. GRIT
DEPARTMENT OF COMPUTER SCIENCE
COLORADO STATE U
FT. COLLINS   CO 80523
(303) 491-7033

JAMES A. KENDALL
MHMR/TRIMS
TEXAS MEDICAL CENTER
HOUSTON   TX 77030
(713) 797-1976

WAYNE SEIPEL
COMPUTATION CENTER
U OF TEXAS
AUSTIN   TX 78712

RICHARD TAYLOR
2425 RALEIGH ST.
DENVER   CO 80212
(303) 477-7995

ATTN: B1700 PROTEUS PROJECT
COMPUTER SCIENCE DEPT.
3160   MEB
U OF UTAH
SALT LAKE CIT UT 84112
(801) 581-8224

SCOTT K. WARREN
ROSETTA ALGORITHMS
2414 BRANARD #D
HOUSTON   TX 77098

WALLY WEDEL
COMPUTATION CENTER
U OF TEXAS AUSTIN
AUSTIN   TX 78712
(512) 471-3242

ATTN: LIBRARY
67   DENVER FEDERAL CENTER
BUREAU OF RECLAMATION
DENVER   CO 80225

MARTIN L GRISS
COMPUTER SCIENCE DEPT
U OF UTAH
SALT LAKE CIT UT 84112
(801) 581-6542

RUSSELL W ZEARS
BIOMETRY LAB
449   ADMINISTRATION BLDG R7
UNIVERSITY OF TEXAS MEDICAL BRANCH
GALVESTON   TX 77550
(713) 765-1813

DAVID W. HOGAN
4104 AVENUE F
AUSTIN   TX 78751

HOWARD BUSSEY JR.
NATIONAL OCEANIC AND ATMOSPHERIC ADMIN
BLDG. 1 RM 4557
U.S. DEPARTMENT OF COMMERCE
BOULDER   CO 80302

M. A. KLEINERT
COMP. SCI. DEPT.
3160   MERRILL ENG. BLDG.
U OF UTAH
SALT LAKE CIT UT 84112

ED SHARP
COMPUTER CENTER
U OF UTAH
SALT LAKE CIT UT 84112
(801) 581-6802

THEODORE A. NORMAN
COMP. SCI. DEPT.
BRIGHAM YOUNG UNIVERSITY
PROVO   UT 84602
(801) 374-1211 X3027

RICHARD OHRAN
ELECTICAL ENGINEERING DEPT
459   ESTB
BRIGHAM YOUNG UNIVERSITY
PROVO   UT 84602
(801) 374-1211 X4012

PATRICK PECORARO
UNIVERSITY COMPUTER CENTER
U OF ARIZONA
TUCSON   AZ 85721
(602) 884-2901

R. W. MILKEY
KITT PEAK NATIONAL OBSERVATORY
P.O. BOX 26732
TUCSON   AZ 85726
(602) 327-5511

TOM SANDERSON
617 NORTH FOURTH
BELEN   NM 87002

NANCY RUIZ
ORG. 5166
SANDIA LABS
ALBUQUERQUE   NM 87115
(505) 264-1890

HARRY M. MURPHY JR.
AIR FORCE WEAPONS LABORATORY
KIRTLAND AFB NM 87117
(505) 264-9317

BILL BUZBEE
LOS ALAMOS SCIENTIFIC LABORATORY
C-DO MS-260
UNIVERSITY OF CALIFORNIA
P.O. BOX 1663
LOS ALAMOS   NM 87545

ROBERT T. JOHNSON
C-11   MAIL STOP 296
LOS ALAMOS SCIENTIFIC LABORATORY
P.O. BOX 1663
LOS ALAMOS   NM 87545
(505) 667-5014

JOHN MONTAGUE
GROUP C11
MAIL STOP 296
LOS ALAMOS SCIENTIFIC LABORATORY
LOS ALAMOS   NM 87545

JAMES DARLING
NEW MEXICO TECH
BOX 2139 CAMPUS STATION
SOCORRO   NM 87801
(505) 835-5374

T. A. NARTKER
NEW MEXICO INSTITUTE OF MINING AND TEC
SOCORRO   NM 87801
(505) 835-5126

J. MACK ADAMS
COMP. SCI. DEPT.
NEW MEXICO STATE U
BOX 3CU
LAS CRUCES   NM 88003
(505) 646-3723

ATTN: USER SERVICES LIBRARIAN
UNIVERSITY COMPUTER CENTER
NEW MEXICO STATE UNIVERSITY
BOX 3AT
LAS CRUCES   NM 88003
(505) 646-4433

JOHN WERTH
DEPT. OF MATH
U OF NEVADA LAS VEGAS
LAS VEGAS   NV 89154
(702) 739-3715

ATTN: PROGRAMMING ADVISOR
UNS COMPUTING CENTER
22
U OF NEVADA
RENO

GARY CARTER
SEISMOLOGY DEPT.
MACKAY SCHOOL OF MINES
U OF NEVADA RENO
RENO   NV 89507

ATTN: ACADEMIC SERVICES
UNIVERSITY COMPUTER CENTER
U OF SOUTHERN CALIFORNIA
1020 W. JEFFERSON BLVD.
LOS ANGELES   CA 90007
(213) 746-2957

STEVEN BARRYTE
6620 W. 5TH STREET
LOS ANGELES   CA 90048
(213) 653-8697

ERWIN BOOK
3169 COLBY AVENUE
LOS ANGELES   CA 90066

HOWARD H. METCALF
2590 GLEN GREEN #4
HOLLYWOOD   CA 90068

DENNIS HEIMBIGNER
2500 CARNEGIE LANE #8
REDONDO BEACH CA 90278
(213) 371-8938

RALPH L. LONDON
INFORMATION SCIENCES INSTITUTE
U OF SOUTHERN CALIFORNIA
4676 ADMIRALTY WAY
MARINA DEL RA CA 90291

MICHAEL TEENER
TECHNOLOGY SERVICE CORP.
2811 WILSHIRE BLVD.
SANTA MONICA CA 90403
(213) 829-7411

PHYLLIS A. REILLY
19711 GALWAY AVENUE
CARSON   CA 90746
(213) 321-5215

PAUL L. MCCULLOUGH
911 GENOA ST.
MONROVIA   CA 91016
(213) 447-3202

CLARK M. ROBERTS
219 VIOLET AVENUE
MONROVIA   CA 91016
(213) 456-3858 (HOME)
(213) 658-2405 (WORK)

CHARLES L. LAWSON
JET PROPULSION LABORATORY
MS 125/128
CALIFORNIA INSTITUTE OF TECHNOLOGY
4800 OAK GROVE DR.
PASADENA   CA 91103
(213) 354-4321

WILLIAM C. PRICE
480 PEMBROOK DRIVE
PASADENA   CA 91107
(213) 351-6551 X219

GERALD BRYAN
SEAVER COMPUTER CENTER
CLAREMONT COLLEGES
CLAREMONT   CA 91711
(714) 626-8511 X3220

MARK J. KAUFMAN
916 E WASHINGTON APT. 108
ESCONDIDO   CA 92025
(714) 743-5911

MARK OVERGAARD
APIS DEPT.
C-014
U OF CALIFORNIA - SAN DIEGO
LA JOLLA   CA 92093
(714) 452-2728

MICHAEL S. BALL
CODE 2
NAVAL UNDERSEA CENTER
SAN DIEGO   CA 92132

ATTN: COMPUTER SCIENCES INSTITUTE
U OF CALIFORNIA
RIVERSIDE   CA 92507

KURT COCKRUM
3398 UTAH
RIVERSIDE   CA 92507

JOHN M. GRAM
COMPUTING FACILITY
U OF CALIFORNIA
IRVINE   CA 92717
(714) 833-6844

JON F. HUERAS
DEPT. OF INFORMATION AND COMPUTER SCIE
U OF CALIFORNIA IRVINE
IRVINE   CA 92717

WILLIAM L. COOPER
ORG 4400
INTERSTATE ELECTRONICS
707 E. VERMONT
ANAHEIM   CA 92805
(714) 772-2811

DAVID W. GIEDT
5421 WILLOWICK CIR.
ANAHEIM   CA 92807
(714) 772-2811

ATTENTION: EDWARD E. BALKOVICH
SCIENCE AND TECHNOLOGY DIVISION
GENERAL RESEARCH CORP.
5383 HOLLISTER AVE.
SANTA BARBARA CA 93105
(805) 964-7724

ROBERT ALAN DOLAN
SPEECH COMMUNICATIONS RESEARCH LAB
800A MIRAMONTE DRIVE
SANTA BARBARA CA 93109
(805) 965-3011

NEIL W. WEBRE
DEPT. OF COMP. SCI. AND STAT.
CALIF. POLY. STATE UNIV.
SAN LUIS OBIS CA 93401
(805) 546-2986

JAMES L. BEUG
DEPT. OF COMP. SCI.
CALIFORNIA POLYTECHNIC STATE U
SAN LUIS OBIS CA 93407
(805) 546-1255

GARY BABCOCK
110-E RICHMOND ROAD
CHINA LAKE   CA 93555
(714) 446-6733

DAVID ELLIOT SHAW
STRUCTURED SYSTEMS
200 THIRD STREET -SUITE 207
LOS ALTOS   CA 94022
(415) 966-2082
(415) 948-0877

APRIL MILLER CONVERSE
SEISMIC ENGINEERING BRANCH
U.S.G.S.
345 MIDDLEFIELD ROAD
MENLO PARK   CA 94025

GLENN T. EDENS
DACONICS DIV.
XEROX
350 POTRERO AVENUE
SUNNYVALE   CA 94086
(408) 738-4800 (DACONICS)
(415) 494-4464 (XEROX/PARC)

DENNIS GRAHAM
AMDAHL CORP.
1250 E. ARQUES AVE.
SUNNYVALE   CA 94086
(408) 735-4602

M. H. MACDOUGALL
AMDAHL CORP.
1250 EAST ARQUES AVE.
SUNNYVALE   CA 94086
(408) 736-4896

GARY W. WINIGER
P.O. BOX 60835
SUNNYVALE   CA 94088
(415) 964-6982

NIKLAUS WIRTH
XEROX RESEARCH CENTER
3333 COYOTE HILL ROAD
PALO ALTO   CA 94304

PAUL HECKEL
INTERACTIVE SYSTEMS CONSULTANTS
P.O. BOX 2345
PALO ALTO   CA 94305
(415) 965-0237

JOHN BANNING
MAIL DROP 88
STANFORD LINEAR ACCELERATOR CENTER
P.O.BOX 4349
STANFORD   CA 94305
(415) 854-3300 X2802 (OFFICE)
(415) 325-9226 (HOME)

DAVID C. LUCKHAM
COMP. SCI. DEPT.
A.I. LABORATORY
STANFORD UNIVERSITY
STANFORD   CA 94305
(415) 497-4971

BRIAN MCGUIRE
P.O. BOX 1371
FREMONT   CA 94538

JOHN C. BEATTY
L-73
LAWRENCE LIVERMORE LAB
BOX 808
LIVERMORE   CA 94550
(415) 447-1100 X3114

WILLIAM P. TAYLOR
L-315
UNIVERSITY OF CALIFORNIA
P.O. BOX 808
LIVERMORE   CA 94550
(415) 447-1100

BRYAN L. HIGGINS
SCIENCE APPLICATIONS INC.
8201 CAPWELL DRIVE
OAKLAND   CA 94621
(415) 562-9163

JEFFREY BARTH
COMP. SCI. DIVISION
573   EVANS HALL
U OF CALIFORNIA
BERKELEY   CA 94720
(415) 642-4948

BLAND EWING
DEPT. OF ENTYMOLOGY
137   GIANNINI HALL
U OF CALIFORNIA
BERKELEY   CA 94720
(415) 642-6660

ED FOURT
C/O LBL LIBRARY
134   BLDG 50
LAWRENCE BERKELEY LAB
BERKELEY   CA 94720
(415) 843-2740 X5293

SUSAN L. GRAHAM
COMP. SCI. DIVISION-EECS
511   EVANS HALL
U OF CALIFORNIA
BERKELEY   CA 94720

G. CARRICK
MS970
FOUR-PHASE SYSTEMS INC.
19333 VALLCO PARKWAY
CUPERTINO   CA 95014
(408) 255-0900 X281

FAY CHONG
10405 DEMPSTER AVENUE
CUPERTINO   CA 95014
(408) 987-1655

R. GREINER
MS970
FOUR-PHASE SYSTEMS INC.
19333 VALLCO PARKWAY
CUPERTINO   CA 95014
(408) 255-0900 X231

P. LIAO
MS970
FOUR-PHASE SYSTEMS INC.
19333 VALLCO PARKWAY
CUPERTINO   CA 95014
(408) 255-0900 X302

RONALD L DANIELSON
DEPARTMENT OF EECS
UNIVERSITY OF SANTA CLARA
SANTA CLARA   CA 95051
(408) 984-4181

DADO BANATAO
3060 BILBO DRIVE
SAN JOSE   CA 95121
(408) 227-9027

GARY LOWELL
2625 HIDDEN VALLEY
SANTA ROSA   CA 95404
(707) 544-6373

W. W. PETERSON
DEPT OF ICS
U OF HAWAII
2565 THE MALL
HONOLULU   HI 96822
(808) 948-7420

ROY CARLSON
(50-454)
TEKTRONIX
P.O. BOX 500
BEAVERTON   OR 97077

ROD STEEL
MS 60-456
TEKTRONIX INC.
P.O. BOX 500
BEAVERTON   OR 97077
(503) 638-3411 X2523

BOB PHILLIPS
2009 N.E. BRAZEE
PORTLAND   OR 97212
(503) 284-8369

BARRY SMITH
OMSI COMPUTING
4015 SW CANYON ROAD
PORTLAND   OR 97221
(503) 248-5923

DAVID ROWLAND
ELECTRO SCIENTIFIC INDUSTRIES
13900 N.W. SCIENCE PARK DRIVE
PORTLAND   OR 97229

ATTN: COMPUTER CENTER
OREGON STATE U
CORVALLIS   OR 97331

ATTN: DOCUMENTS ROOM
COMPUTER SCIENCE DEPARTMENT
U OF OREGON
EUGENE   OR 97403
(503) 686-4394

LESLIE R. KERR
DAVID L. JOHNSON AND ASSOCIATES INC.,
10545 WOODHAVEN LANE
BELLEVUE   WA 98004

JOHN D. WOOLLEY
6722 128TH AVE. SE
BELLEVUE   WA 98006
(206) 641-3443

ERIC SCHNELLMAN
HONEYWELL MARINE SYSTEMS
5303 SHILSHOLE NW
SEATTLE   WA 98117

ATTN: BOEING COMPANY
87-67 KENT TECHNICAL LIBRARY
P.O. BOX 3999
SEATTLE   WA 98124

HELLMUT GOLDE
DEPT. OF COMP. SCI.
FR-35
U OF WASHINGTON
SEATTLE   WA 98195
(206) 543-9264

A. J. GERBER
BASSER DEPT. OF COMPUTER SCIENCE
UNIVERSITY OF SYDNEY
SYDNEY   N.S.W.   2006
AUSTRALIA

CARROLL MORGAN
BASSER DEPT. OF COMPUTER SCIENCE
U OF SYDNEY
SYDNEY   N.S.W.   2006
AUSTRALIA

BRIAN G. ROWSWELL
UNIVERSITY COMPUTER CENTRE
UNIVERSITY OF SYDNEY
SYDNEY   N.S.W.   2006
AUSTRALIA
692 3491

KEN ROBINSON
DEPT. OF COMPUTER SCIENCE
UNIVERSITY OF NEW SOUTH WALES
P.O. BOX 1
KENSINGTON   N.S.W.   2033
AUSTRALIA
663 0351

ANTHONY P. KYNE
DEPT. OF COMPUTER SCIENCE
UNIVERSITY OF MELBOURNE
PARKVILLE   VICTORIA   3052
AUSTRALIA
345 1844

ATTN: PROGRAM LIBRARIAN
COMPUTING CENTRE
UNIVERSITY OF ADELAIDE
BOX 498 G.P.O.
ADELAIDE   S.A.   5001
AUSTRALIA
61 822 34333 X2720/X2099

C. D. MARLIN
DEPT OF COMPUTING SCIENCE
UNIVERSITY OF ADELAIDE
ADELAIDE   S.A.   5001
AUSTRALIA
223 4333

B. KIDMAN
DEPT OF COMPUTER SCIENCE
UNIVERSITY OF ADELAIDE
ADELAIDE   S.A.   5066
AUSTRALIA
23 4333

ATTN: SECRETARY
DEPARTMENT OF INFORMATION SCIENCE
UNIVERSITY OF TASMANIA
GPO BOX 252C
HOBART   TASMANIA   7001
AUSTRALIA

A. H. J. SALE
DEPT. OF INFORMATION SCIENCE
UNIVERSITY OF TASMANIA
BOX 252C
HOBART   TASMANIA   7001
AUSTRALIA
23 0561

O. BEAUFAYS
MATHEMATIQUES APPLIQUEES
UNIVERSITE LIBRE DE BRUXELLES
AVENUE F.-D. ROOSEVELT 50
BRUXELLES   1050
BELGIUM

ALAIN PIROTTE
MBLE/RESEARCH LABORATORY
AVENUE EM. VAN BECELAERE 2
BRUSSELS   B-1170
BELGIUM
673.41.90
673.41.99

SERGIO DE MELLO SCHNEIDER
DEPARTAMENTO DE COMPUTACAO
UNIVERSIDADE FEDERAL DE SAO CARLOS
SAO CARLOS   SP   13560
BRAZIL

F. CELLINI
DEPT. OF COMPUTER SCIENCE
UNIVERSITY OF WESTERN ONTARIO
LONDON   ONTARIO
CANADA
(519) 679-6051

FRANKLIN B. DE GRAAF
6 CARMICHAEL COURT
KANATA   ONTARIO   K2K 1K2
CANADA

W. MORVEN GENTLEMAN
MATHEMATICS COMPUTING FACILITY
UNIVERSITY OF WATERLOO
WATERLOO   ONTARIO   N2L 3G1
CANADA
(519) 885-1711

ATTN: LIBRARY
PERIODICALS SECTION
UNIVERSITY OF ALBERTA
EDMONTON   ALBERTA   T6G 2J8
CANADA

ATTN: M. DOHERTY
128 TECHNICAL REFERENCE CENTER
UNIVERSITY OF TORONTO COMPUTER CENTER
10 KINGS COLLEGE ROAD
TORONTO   ONTARIO
CANADA
(416) 978-8995

ATTN: REFERENCE ROOM
COMPUTING AND INF. SCI.
QUEEN'S UNIVERSITY
KINGSTON   ONTARIO   K7L 3N6
CANADA

ATTN: PROGRAM LIBRARY
COMPUTING CENTER
223   NATURAL SCIENCE CENTER
U OF WESTERN ONTARIO
LONDON   ONTARIO   N6A 5B7
CANADA
(519) 679-2151

BART W. POLLACK
DEPT. OF COMP. SCI.
U OF BRITISH COLUMBIA
2075 WESBROOK PLACE
VANCOUVER   BC   V6T 1W5
CANADA
(604) 228-6794

F. G. PAGAN
COMPUTER SCIENCE
MEMORIAL UNIVERSITY
ST. JOHN'S   NEWFOUNDLA A1C 5S7
CANADA

R. D. TENNENT
DEPT. OF COMPUTING AND INFORMATION SCI
QUEENS UNIVERSITY
KINGSTON   ONTARIO   K7L 3N6
CANADA

L. C. PORTIL
COMPUTER CENTRE
U OF WINDSOR
WINDSOR   ONTARIO   N9B 3P4
CANADA
(519) 253-4232 X645

DOUG DYMENT
6442 IMPERIAL AVE.
W. VANCOUVER B.C.   V7W 2J6
CANADA
(604) 921-7954

J. W. ATWOOD
DEPT OF COMP. SCI.:H963-10
CONCORDIA UNIVERSITY
1455 DE MAISONNEUVE BLVD. WEST
MONTREAL   QUEBEC   H3G 1M8
CANADA
(514) 879-8130

N. SOLNTSEFF
DEPT. OF APPLIED MATH.
MCMASTER UNIVERSITY
HAMILTON   ONTARIO   L8S 4K1
CANADA
(416) 525-9140 X4689

G. D. DERHAK
COMPUTER CENTRE
U OF MANITOBA
WINNIPEG   MANITOBA   R3T 2N2
CANADA

ATTN: DATALOGISK INSTITUT
COPENHAGEN UNIVERSITY
SIGURDSGADE 41
COPENHAGEN N DK-2200
DENMARK

D. B. COLDRICK
COMPUTATION CENTRE
BLDG. M-60
NATIONAL RESEARCH COUNCIL
MONTREAL ROAD
OTTAWA   ONTARIO   K1A OR6
CANADA

MIKE KIMBER
DATA CENTRE
THE GLOBE AND MAIL
444 FRONT ST. WEST
TORONTO   ONTARIO   M5V 2S9
CANADA

W. BRUCE FOULKES
DEPARTMENT OF COMPUTER SCIENCE
THE UNIVERSITY OF MANITOBA
WINNIPEG   MANITOBA   R3T 2N2
CANADA
(204) 474-8466

LARS CHRISTENSEN
ALDERSHVILEVEJ 16
BAGSVAERD   DK-2880
DENMARK
009 45 2 98 20 09

LUIGI LOGRIPPO
COMP. SCI. DEPT.
U OF OTTAWA
OTTAWA   ONTARIO   K1N 6N5
CANADA

HENRY SPENCER
SP SYSTEMS
BOX 5255 STATION A
TORONTO   ONTARIO   M5W 1N5
CANADA

STEPHEN SOULE
DEPT. OF COMP. SCI.
U OF CALGARY
CALGARY   ALBERTA   T2N 1N4
CANADA
(403) 284-6780

PREBEN TAASTI
COMPUTER CENTER
UNIVERSITY OF AALBORG
STRANDVEJEN 19
AALBORG   DK-9000
DENMARK
(08) 138 788

H. TAYLOR
COMPUTING CENTRE
APPLICATIONS DEPT.
U OF OTTAWA
OTTAWA   ONTARIO   K1N 6N5
CANADA

ANNE STOCCO
COMP. AND INFO. SCIENCE
108   I.C.S.
UNIVERSITY OF GUELPH
GEULPH   ONTARIO   N1G 2W1
CANADA
.... X2259

BRIAN W. UNGER
COMPUTER SCIENCE DEPT.
UNIVERSITY OF CALGARY
CALGARY   ALBERTA   T2N 1N4
CANADA
(403) 284-6316

L. BRANDT
DET REGIONALE EDB-CENTER
AARHUS UNIVERSITET
AARHUS   C.   8000
DENMARK
06 - 12 83 55

ATTENTION: DONALD LINDSAY
DYNALOGIC CORPORATION LIMITED
141 BENTLEY AVENUE
OTTAWA   ONTARIO   K2E 6T7
CANADA
(613) 226-1585

CHARLES H. FORSYTH
APT. 2-304
300 REGINA ST. N.
WATERLOO   ONTARIO   N2J 4T2
CANADA
(519) 884-7531

B. VENKATESAN
DEPT. OF COMPUTER SERVICES
U OF CALGARY
2920 24TH AVE. N.W.
CALGARY   ALBERTA   T2N 1N4
CANADA
(403) 284-6207

ANTTI SALAVA
DEPT. OF COMPUTER SCIENCE
UNIVERSITY OF HELSINKI
TOOLONKATU 11
HELSINKI 10   SF-00100
FINLAND

JUHA HEINANEN
DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF TAMPERE
BOX 607
TAMPERE 10   SF-3310T
FINLAND

O. LECARME
I.M.A.N.
UNIVERSITE DE NICE
PARC VALROSE
NICE   CEDEX   06034
FRANCE

P. MAURICE
INFORMATIQUE
UNIVERSITE PAUL SABATIER
118 ROUTE DE NARBONNE
TOULOUSE   CEDEX   31077
FRANCE

JEAN-PIERRE FAUCHE
DEPARTMENT INFORMATIQUE
IREP
BOITE POSTALE 47
GRENOBLE   CEDEX   38040
FRANCE

ALAIN TISSERANT
DEPARTEMENT INFORMATIQUE
ECOLE DES MINES
PARC DE SAURUPT
NANCY   CEDEX   54042
FRANCE

HORST SANTO
GESELLSCHAFT FUER MATHEMATIK UND DATEN
INSTITUT FUER PLANUNGS UND ENTSCHEIDUN
POSTFACH 1240 SCHLOSS BIRLINGHOVEN
ST.AUGUSTIN 1 D-5202
GERMANY

H.-J. HOFFMANN
FACHBEREICH INFORMATIK
TECHNISCHE HOCHSHULE
STEUBENPLATZ 12
DARMSTADT   D-6100
DARMSTADT   D-6100
GERMANY

ASHOK N. ULLAL
KARLSTR. 1
JETTENBURG   D-7401
GERMANY

LUCIEN FEIEREISEN
EDELSHEIMSTR. 4
KARLSRUHE 1   D-7500
GERMANY

GERHARD GOOS
INSTITUT FUER INFORMATIK II
UNIVERSITAT KARLSRUHE
POSTFACH 6380
KARLSRUHE 1   D-7500
GERMANY
0721/608-3970

ATTENTION: JAN WITT
ZFE FL SAR
SIEMENS AG
HOFMANNSTR. 51
MUNCHEN 70   D-8000
GERMANY
(089) 722-22651

ALBRECHT BIEDL
INSTITUT FUR SOFTWARETECHNIK UND THEOR
DV-GRUNDAUSBILDUNG
TECHNISCHE UNIVERSITAT BERLIN
OTTO-SUHR-ALLEE 18/20
1000 BERLIN   10   VSH 419
GERMANY

GERHARD FRIESLAND
INSTITUT FUR INFORMATIK
UNIVERSITAT HAMBURG
SCHLUETERSTRASSE 70
HAMBURG 13   2
GERMANY

H.-H. NAGEL
INSTITUT FUER INFORMATIK
UNIVERSITAT HAMBURG
SCHLUTERSTRASSE 66-72
HAMBURG 13   2
GERMANY

CARSTEN KOCH
DISTRIKT NORD
CONTROL DATA GMBH
UBERSEERING 13
HAMBURG 39   2000
HAMBURG 39   2000
GERMANY
630 80 81 - 85

W. WEHINGER
LANGUAGES AND PROCESSORS GROUP
RECHENZENTRUM
UNIVERSITAT STUTTGART
STUTTGART 80 7000
GERMANY
(0711) 784/1

ATTENTION: N. V. KOTESWARA RAO
COMPUTER TRG. UNIT
ELECTRONICS CORPORATION OF INDIA
HYDERABAD   (AP)   500762
INDIA
71611

MICHAEL Z. HANANI
COMPUTATION CENTER
BEN GURIAN UNIVERSITY OF THE NEGEV
BEER-SHEVA
ISRAEL

RUTH WEINBERG
COMPUTATION CENTER
HEBREW UNIVERSITY OF JERUSALEM
JERUSALEM
ISRAEL
02-32011/280

GIDEON YUVAL
COMPUTER SCIENCE
THE HEBREW UNIVERSITY
JERUSALEM
ISRAEL

IRVING N. RABINOWITZ
DEPT. OF COMP. SCI.
TECHNION-ISRAEL INSTITUTE OF TECHNOLOG
TECHNION CITY HAIFA
ISRAEL

MARCO SOMMANI
C/O CNUCE
VIA SANTA MARIA 36
PISA   I-56100
ITALY
(050) 45245

MAURO MONTESI
TEMA SPA
VIA MARCONI 29/1
BOLOGNA   40122
BOLOGNA   40122
ITALY
091-267285

GUISEPPE SELVE
TEMA S.P.A.
VIA MARCONI 29/1
BOLOGNA   40122
ITALY
051-267285

TERUO HIKITA
DEPT. OF INFO. SCI.
U OF TOKYO
TOKYO   113
JAPAN
03-812-2111 X2947

EIITA WADA
DEPARTMENT OF MATHEMATICAL ENGINEERING
UNIVERSITY OF TOKYO
BUNKYOKU   TOKYO   113
JAPAN
(03) 812-2111 X7486

TOSHIAKI SAISHO
1-25-7 KITAMAGOME
OOTA-KU   TOKYO   143
JAPAN

MASARU WATANABE
9-16 SHINOHARADAI
KOHOKU-KU   YOKOHAMA   222
JAPAN

MAKOTO ARISAWA
COMPUTER SCIENCE DEPARTMENT
YAMANASHI UNIVERSITY
4-3-11 TAKEDA KOFU
YAMANASHI   400
JAPAN
(0552) 52-1111

NOBUKI TOKURA
DEPT. OF INFORMATION AND COMPUTER SCIE
OSAKA UNIVERSITY
1-1 MACHIKANEYAMA
TOKONAKA   500
JAPAN

CHARLES E. MURPHY
COMPUTER SCIENCE GROUP
UNIVERSITY OF TRIPOLI
P.O. BOX 856
TRIPOLI
LIBYA

ATTN: DEPARTMENT OF INFORMATION SCIENC
VICTORIA UNIVERSITY OF WELLINGTON
PRIVATE BAG
WELLINGTON
NEW ZEALAND

IVAR LABERG
COMPUTER DEPARTMENT
UNIVERSITY HOSPITAL OSLO
RIKSHOSPITALET
OSLO   1
NORWAY
(02) 20 10 50

ATTENTION: E. N. VAN DEVENTER
COMPUTING CENTRE
NATIONAL RESEARCH INSTITUTE FOR MATHEM
P O BOX 395
PRETORIA   0001
SOUTH AFRICA
74-9111

STEN LJUNGKVIS
GUSTAF CLASONS GATA 61
NORRKOPING   S-603 78
SWEDEN

STAFFEN ROMBERGER
COMPUTER SCIENCE
ROYAL INSTITUTE OF TECHNOLOGY
STOCKHOLM   S-100 44
SWEDEN

LARS-ERIK THORELLI
DEPT. OF TELECOMMUNICATION NETWORKS &
THE ROYAL INSTITUTE OF TECHNOLOGY
STOCKHOLM 70   S-100 44
SWEDEN
SWEDEN-05-236320

LENNART OSKARSSON
TELEFONAKTIEBOLAGET L M ERICSSON
FACK
MOLNDAL   S-431 20
SWEDEN

KURT FREDRIKSSON
RINGLEKEN 7
MOLNDAL   S-431 39
SWEDEN
4631-41 05 14 (HOME)
4631-27 50 00-491 (OFFICE)

DAVID BATES
12   CHEMIN DE TAVERNAY
1218 GRAND SACONNEX
GENEVA
SWITZERLAND

R. MOREL
CENTRE DE CALCUL ELECTRONIQUE
COLLEGE DE GENEVE
1211 GENEVE 3
SWITZERLAND
27 22 28

URS AMMANN
INSTITUT FUER INFORMATIK
ZUERICH   CH-8092
SWITZERLAND
(CH ZUERICH) 32 62 11 X2214

SVEN ERIK KNUDSEN
INSTITUT FUER INFORMATIK
ETH - ZENTRUM
ZUERICH   CH-8092
SWITZERLAND

ATTN: RZ - BIBLIOTHEK
ETH - ZENTRUM
ZURICH   CH-8092
SWITZERLAND

CHRISTIAN JACOBI
INSTITUT FUER INFORMATIK
ETH
ZURICH   CH-8092
SWITZERLAND
01 326277 2217

S. BALASUBRAMANIAN
DEPARTMENT MSE
KONINKLIJKE/SHELL-LABORATORIUM
PO BOX 3003
AMSTERDAM
THE NETHERLANDS
(020) 202694

A. C. W. LEYEN
DEPARTMENT MSE
C/O KONINKLIJKE
SHELL-LABORATORIUM
P.O. BOX 3003
AMSTERDAM
THE NETHERLANDS

ANDREW S. TANENBAUM
WISKUNDIG SEMINARIUM
VRIJE UNIVERSITEIT
DE BOELELAAN 1081
AMSTERDAM
THE NETHERLANDS
020 548 24 10

J. J. VAN AMSTEL
COMPUTING CENTRE
EINDHOVEN UNIVERSITY OF TECHNOLOGY
P.O. BOX 513
EINDHOVEN
THE NETHERLANDS
(040) 474547

ATTN: DSM
CENTRAL LIBRARY
P.O. BOX 18
GELEEN
THE NETHERLANDS

D. D. DE VRIES
LANDLEVEN 1
REKENCENTRUM R.U.G.
P.O. BOX 800
GRONINGEN
THE NETHERLANDS

H. VAN LOON
ACADEMISCH COMPUTER CENTRUM UTRECHT
BUDAPESTLAAN 6
DE UITHOF   UTRECHT
THE NETHERLANDS
030-531436

ATTN: BOEKHANDEL VERWIJS EN STAM B.V.
PRINSESSEGRACHT 2
'S-GRAVENHAGE 2005
THE NETHERLANDS

J. A. ALANEN
VAKGROEP INFORMATICA R.U.
BUDAPESTLAAN 6
UTRECT   2506
THE NETHERLANDS

T. J. VAN WEERT
ELZENLAAN 28
PEIZE   8131
THE NETHERLANDS

ATTN: THE LIBRARIAN
DEPT. OF COMPUTER STUDIES
U OF LANCASTER
BAILRIGG   LANCASTER
UNITED KINGDOM
LANCASTER 65201 X4133

C. J. COPELAND
SCHOOL OF COMPUTER SCIENCE
ULSTER COLLEGE
JORDANSTOWN
NEWTOWNABBEY N.IRELAND
UNITED KINGDOM

R. G. DICKERSON
SCHOOL OF INFORMATION SCIENCES
THE HATFIELD POLYTECHNIC
PO BOX 109 COLLEGE LANE
HATFIELD   HERTS   AL10 9AB
UNITED KINGDOM
HATFIELD 68100

MAURICE O'FLAHERTY
444 MEVILLE GARDEN VILLAGE
NEWTOWNABBEY N. IRELAND ANTRIM
UNITED KINGDOM

C. A. LANG
PITT BUILDING
CAMBRIDGE UNIVERSITY PRESS
TRUMPINGTON ST.
CAMBRIDGE   ENGLAND   CB2 1RP
UNITED KINGDOM
CAMBRIDGE 58331

A. BALFOUR
COMPUTER CENTRE
HERIOT-WATT UNIVERSITY
37-39 GRASSMARKET
EDINBURGH   SCOTLAND   EH1 2HW
UNITED KINGDOM

BILL FINDLEY
COMPUTING SCIENCE DEPARTMENT
UNIVERSITY OF GLASGOW
GLASGOW   SCOTLAND   G12 8QQ
UNITED KINGDOM
339 8855 X7391

A. M. ADDYMAN
DEPARTMENT OF COMPUTER SCIENCE
THE UNIVERSITY
MANCHESTER   ENGLAND   M13 9PL
UNITED KINGDOM
061-273 5466

JOHN REYNOLDS
31 BARRINGTON ROAD
LONDON   ENGLAND   N8
UNITED KINGDOM

D. W. BARRON
COMPUTER STUDIES GROUP
THE UNIVERSITY
SOUTHAMPTON　　ENGLAND　　SO9 5NH
UNITED KINGDOM
0703-559122 X700

J. GOODSON
DEPARTMENT OF MATHEMATICS
THE UNIVERSITY
SOUTHAMPTON　　ENGLAND　　SO9 5NH
UNITED KINGDOM

JUDY MULLINS
DEPARTMENT OF MATHEMATICS
THE UNIVERSITY OF SOUTHAMPTON
SOUTHAMPTON　　ENGLAND　　SO9 5NH
UNITED KINGDOM
0703 559122 X2387

CHRIS MARTIN
COMPUTING SERVICES
THE HICKS BUILDING
UNIVERSITY OF SHEFFIELD
SHEFFIELD　　ENGLAND　　S10 2TN
UNITED KINGDOM
78555 X263

ROY EDWARDS
DEPT. OF STAT. AND COMP. SCI.
HOLLOWAY COLLEGE
EGHAM HILL
EGHAM　　SURREY　　TW20 OEX
UNITED KINGDOM
EGHAM 4455

ATTENTION: C. LAZOU
COMPUTER CENTRE
UNIVERSITY OF LONDON
20 GUILFORD STREET
LONDON　　ENGLAND　　WC1
UNITED KINGDOM
01-405-8400

ATTN: LIBRARIAN
PO BOX 48E
LOGICA LIMITED
64 NEWMAN STREET
LONDON　　ENGLAND　　W1A 4SE
UNITED KINGDOM
(01) 580 8361

ROBERT REINHARDT
FABIANIJEVA 39
LJUBLJANA　　61 000
YUGOSLAVIA

JOHN W. ADAMS　　18015
J. MACK ADAMS　　88003
A. M. ADDYMAN　　M13 9PL UNITED KINGDOM
J. A. ALANEN　　2506 THE NETHERLANDS
URS AMMANN　　CH-8092 SWITZERLAND
RICHARD V. ANDREE　　73019
DENNIS S. ANDREWS　　62901
MAKOTO ARISAWA　　400 JAPAN
ATTENTION: C. LAZOU　　WC1 UNITED KINGDOM
ATTENTION: DAN BURROWS　　55812
ATTENTION: DAVID MADISON　　35806
ATTENTION: DONALD LINDSAY　　K2E 6T7 CANADA
ATTENTION: EDWARD E. BALKOVICH　　93105
ATTENTION: E. N. VAN DEVENTER　　0001 SOUTH AFRICA
ATTENTION: GORDON R. SHERMAN　　37916
ATTENTION: JAN WITT　　D-8000 GERMANY
ATTENTION: JERRY W. SEGERS　　30332
ATTENTION: KYU LEE　　59801
ATTENTION: N. V. KOTESWARA RAO　　500762 INDIA
ATTENTION: PAUL C. SMITH　　55455
ATTENTION: ROBERT E. NOVAK　　55165
ATTENTION: RUTH DROZIN　　17837
ATTENTION: R. D. BERGERON　　03824
ATTENTION: STEVE REISMAN　　55455
ATTN: ACADEMIC SERVICES　　90007
ATTN: BOEING COMPANY　　98124
ATTN: BOEKHANDEL VERWIJS EN STAM B.V.　　2005 THE NETHERLANDS
ATTN: B1700 PROTEUS PROJECT　　84112
ATTN: COMPUTER CENTER　　97331
ATTN: COMPUTER SCIENCE DEPT.　　55455
ATTN: COMPUTER SCIENCES INSTITUTE　　92507
ATTN: CONSULTING OFFICE　　61801
ATTN: DATALOGISK INSTITUT　　DK-2200 DENMARK
ATTN: DEPARTMENT OF INFORMATION SCIENCE　　NEW ZEALAND
ATTN: DEPT. OF COMPUTER SCIENCE　　58677
ATTN: DIRECTOR　　32611
ATTN: DOCUMENTS ROOM　　97403
ATTN: DOCUMENTS ROOM LIBRARIAN　　46637
ATTN: DOROTHY SMITH - REFERENCE LIBRARIAN　　78712
ATTN: DSM　　THE NETHERLANDS
ATTN: FRIEDA S. COHEN　　53706
ATTN: J. F. MCINTYRE - LIBRARIAN　　22903
ATTN: LIBRARIAN　　32611
ATTN: LIBRARY　　UNITED KINGDOM
ATTN: LIBRARY　　01704
ATTN: LIBRARY　　T6G 2J8 CANADA
ATTN: M. DOHERTY　　CANADA
ATTN: PROGRAM LIBRARIAN　　3001 AUSTRALIA
ATTN: PROGRAM LIBRARY　　N6A 5B7 CANADA
ATTN: PROGRAMMING ADVISOR　　89507
ATTN: READING ROOM　　02139
ATTN: RECEIVING CLERK　　61820
ATTN: REFERENCE ROOM　　K7L 3N6 CANADA
ATTN: REFERENCE ROOM　　55455
ATTN: RZ - BIBLIOTHEK　　CH-8092 SWITZERLAND
ATTN: SECRETARY　　7001 AUSTRALIA
ATTN: SERIALS DEPT.　　52242
ATTN: SSRFC LIBRARY　　55455
ATTN: THE LIBRARIAN　　UNITED KINGDOM

```
ATTN: UCC LIBRARIAN           52242
ATTN: USER SERVICES GROUP     80523
ATTN: USER SERVICES LIBRARIAN 88003
J. W. ATWOOD         H3G 1M8 CANADA
GARY BABCOCK
FRED P. BAKER                 61820
SAMUEL T. BAKER               37130
T. P. BAKER                   32304
S. BALASUBRAMANIAN                    THE NETHERLANDS
LYNNE J. BALDWIN              68101
A. BALFOUR          EH1 2HW  UNITED KINGDOM
MICHAEL S. BALL               92132
MAURICE BALLEW                79409
RICHARD BALOCCA               61801
DADO BANATAO                  95121
JOHN BANNING                  94305
WILLIAM BARABASH              11794
D. W. BARRON        SO9 5NH  UNITED KINGDOM
STEVEN BARRYTE               90048
JEFFREY BARTH                94720
BRIT J. BARTTER              60202
DAVID BATES                          SWITZERLAND
JOHN C. BEATTY               94550
O. BEAUFAYS                          BELGIUM
STEVEN M. BELLOVIN           27514
HERMAN BERG                  53703
PHILIP N. BERGESTESSER       35758
SCOTT BERTILSON              55406
JAMES L. BEUG                93407
ALBRECHT BIEDL      VSH 419  GERMANY
MARK BILODEAU                55401
TIM BONHAM                   55454
ERWIN BOOK                   90066
GARY J. BOOS                 58501
KEN BORGENDALE               55455
L. BRANDT                     8000  DENMARK
RONALD F. BRENDER            01754
FRANK BREWSTER               22304
C. E. BRIDGE                 19898
ROBERT L. BRIECHLE           44325
ALBERT S. BROWN              01754
ARTHUR A. BROWN              20037
WARREN R. BROWN              02038
GERALD BRYAN                 91711
WILHELM BURGER               78712
HOWARD BUSSEY JR.            80302
BILL BUZBEE                  87545
ROY CARLSON                  97077
DAVID E. CARLTON             60625
G. CARRICK                   95014
GARY CARTER                  89507
D. A. CAUGHFIELD             79601
GARY CEDERQUIST              75275
GERALD N. CEDERQUIST         30332
F. CELLINI                           CANADA
GABRIEL CHANG                02139
FAY CHONG                    95014
LARS CHRISTENSEN    DK-2880  DENMARK
PAUL CHRISTOPHERSON          55345
RICHARD J. CICHELLI          18103

KURT COCKRUM                 92507
WILLIAM L. COHAGAN           78758
GEORGE COHN III              47401
PETER COLONY                 03160
H. B. EN BRIEN      K1A 0R6  CANADA
D. B. COLDRICK      K1A 0R6  CANADA
TERRENCE M. COLLIGAN         02193
MICHAEL N. CONDICT           13440
APRIL MILLER CONVERSE        94025
RICHARD CONWAY               14850
WILLIAM L. COOPER            92805
C. J. COPELAND                       UNITED KINGDOM
F. J. CORBATO                02139
JOHN EARL CRIDER             77025
DONALD B. CROUCH             35486
RONALD L DANIELSON           95051
JAMES DARLING                87801
ANN D. DAVIES                23284
FRANKLIN B. DE GRAAF  K2K 1K2  CANADA
HAROLD DE VORE               58202
D. D. DE VRIES                       THE NETHERLANDS
RANCE J. DELONG              18018
G. D. DERHAK        R3T 2N2  CANADA
R. G. DICKERSON     AL10 9AB UNITED KINGDOM
LLOYD DICKMAN                01742
ROBERT ALAN DOLAN            93109
JEFFREY J. DRUMMOND          55455
DOUG DYMENT         V7W 2J6  CANADA
T. A. D'AURIA                10027
R. STERLING EANES            02154
CHRIS EASTLUND               55401
JOHN T. EASTON               55455
GLENN T. EDENS               94086
HANK EDWARDS                 01701
ROY EDWARDS         TW20 OEX UNITED KINGDOM
JOHN D. EISENBERG            19711
DAVE ENGLANDER               18015
PHILLIP H. ENSLOW JR.        30332
HOWARD D. ESKIN              10025
JOHN B. EULENBERG            48824
BLAND EWING                  94720
R. NEIL FAIMAN JR.           48228
JAMES N. FARMER              30332
JOSEPH H. FASEL III          47907
JEAN-PIERRE FAUCHE           38040  FRANCE
LUCIEN FEIEREISEN   D-7500  GERMANY
JEANNE FERRANTE              02139
LINCOLN FETCHER              55455
BILL FINDLEY        G12 8QQ UNITED KINGDOM
CHARLES N. FISCHER           53706
TED FISHMAN                  60659
KEVIN FJELSTED               55455
CHARLES H. FORSYTH  N2J 4T2  CANADA
LLOYD D. FOSDICK             80309
W. BRUCE FOULKES    R3T 2N2  CANADA
ED FOURT                     94720
DENNIS J. FRAILEY            75222
MIKE FRAME                   20006
K. FRANKOWSKI                55455
KJRT FREDRIKSSON    S-431 39 SWEDEN
G. FRIEDER                   14226
```

EDWARD R. FRIEDMAN	10012	
GERHARD FRIESLAND	2	GERMANY
JOHN FUNG	55414	
EDWARD F. GEHRINGER	47907	
W. MORVEN GENTLEMAN	N2L 3G1	CANADA
A. J. GERBER	2006	AUSTRALIA
DAVID W. GIEDT	92807	
N. AMOS GILEADI	01754	
ED GLASER	54302	
JOHN J. GODA JR.	30332	
HELLMUT GOLDE	98195	
DAVID A. GOMBERG	20016	
J. GOODSON	SO9 5NH	UNITED KINGDOM
GERHARD GOOS	D-7500	GERMANY
DENNIS GRAHAM	94086	
SUSAN L. GRAHAM	94720	
WILLIAM Q. GRAHAM	19711	
JOHN M. GRAM	92717	
KRISTINA GREACEN	55455	
MIKE GREEN	78284	
R. GREINER	95014	
DAVID J. GRIFFITHS	02881	
DONALD E. GRIMES	02174	
MARTIN L GRISS	84112	
DALE H. GRIT	80523	
WILLIAM GROSKY	48202	
JONATHON R. GROSS	55420	
ROGER GULBRANSON	61801	
S. L. GULDEN	18015	
MICHAEL HAGERTY	02174	
HARRY P. HAIDUK	79015	
JOEL M. HALPERN	55455	
RONALD J. HAM	01754	
DON HAMNES	55409	
MICHAEL Z. HANANI		ISRAEL
GILBERT J. HANSEN	75075	
BRIAN HANSON	55455	
STEPHEN J. HARTLEY	23185	
KEITH HAUER-LOWE	55417	
PAUL HECKEL	94305	
H. G. HEDGES	48824	
CHARLES HEDRICK	61801	
DENNIS HEIMBIGNER	90278	
JUHA HEINANEN	SF-33101	FINLAND
T. S. HEINES	44115	
DAVID HELFINSTINE	55301	
CARL HENRY	55057	
WILLIAM HENRY	10003	
MARK HERSEY	48823	
BRYAN L. HIGGINS	94621	
TERUO HIKITA	113	JAPAN
W. A. HINTON	53201	
THEA D. HODGE	55455	
TIMOTHY W. HOEL	55057	
MARILYN HOFFMAN	10013	
H.-J. HOFFMANN	D-6100	GERMANY
TIMOTHY J. HOFFMANN	55455	
DAVID W. HOGAN	78751	
WILLIAM C. HOPKINS	14226	
FRANK H. HOHN	53706	

FREDERICK A. HOSCH	70122	
DAVID A. HOUGH	23602	
ROSEMARY HOWBRIGG	06413	
RALPH HOWENSTINE	73069	
RICHARD HUBER	77843	
JON F. HUERAS	92717	
STEVEN L. HUYSER	48824	
DANIEL C. HYDE	17837	
M. ELIZABETH IBARRA	11973	
AVRUM ITZKOWITZ	61820	
CHRISTIAN JACOBI	CH-8092	SWITZERLAND
GEORGE D. JELATIS	55455	
MITCHELL R. JOELSON	55455	
CHRISTOPHER K. JOHANSEN	02166	
BRIAN W. JOHNSON	75075	
ROBERT T. JOHNSON	87545	
R. I. JOHNSON	58202	
R. WARREN JOHNSON	56301	
WARREN JOHNSON	70504	
ED KATZ	70504	
MARK J. KAUFMAN	92025	
THOMAS A. KEENAN	20550	
GINGER KELLY	77001	
WILLETT KEMPTON	78705	
JAMES A. KENDALL	77030	
LESLIE R. KERR	98004	
ROBERT KEZELL	19122	
B. KIDMAN	5066	AUSTRALIA
D. B. KILLEEN	70118	
MIKE KIMBER	M5V 2S9	CANADA
M. A. KLEINERT	84112	
J. C. KNIGHT	23665	
SVEN ERIK KNUDSEN	CH-8092	SWITZERLAND
CARSTEN KOCH	2000	GERMANY
ALAN A. KORTESOJA	48103	
R. KRASIN	02154	
ANTHONY P. KYNE	3052	AUSTRALIA
IVAR LABERG	1	NORWAY
R. B. LAKE	44106	
DAN LALIBERTE	55455	
LARRY O. LANDIS	64108	
STEVE LANDRY	70504	
DAVID LANDSKOV	70504	
C. A. LANG	CB2 1RP	UNITED KINGDOM
CHARLES L. LAWSON	91103	
D. LECARME	06034	FRANCE
HENRY F. LEDGARD	01002	
STEVE LEBENHAUSEN	08904	
MIKE LEMON	15213	
L. RICHARD LEWIN	48197	
C. H. LEVINE		THE NETHERLANDS
H. P. LIAO	95014	
LAWRENCE A. LIDDIARD	55455	
DENNIS R. LINKE	15213	
JOHN P. IN		
GARY LINDSTROM	15213	
STEN LJUNGKVIST		SWEDEN
LUIGI LOGRIPPO	KIN 6N5	CANADA
RALPH L. LONDON	90291	

GARY LOWELL	95404	
DAVID C. LUCKHAM	94305	
MARK LUKER	55812	
MICHAEL J. LUTZ	14623	
JOHN T. LYNCH	19301	
M. H. MACDOUGALL	94086	
C. D. MARLIN	5001	AUSTRALIA
CHRIS MARTIN	S10 2TN	UNITED KINGDOM
JAMES F. MARTINSON	56201	
P. MAURICE	31077	FRANCE
RAINER F. MCCOWN	21045	
PAUL L. MCCULLOUGH	91016	
BRIAN MCGUIRE	94538	
TERRY P. MEDLIN	20014	
MICHAEL MEEHAN	02138	
HUGO MEISSER	55427	
MICHAEL ROBERT MEISSNER	55455	
J. SCOTT MERRITT	12180	
DAVID C. MESSER	55455	
HOWARD H. METCALF	90068	
GARRY MEYER	11794	
W. J. MEYERS	75243	
JOSEPH A. MEZZAROBA	18041	
ANDY MICKEL	55455	
M. D. MICKUNAS	61801	
R. W. MILKEY	85726	
GLENN MILLER	55109	
JAMES R. MILLER	47906	
VICTOR S. MILLER	02125	
CARLTON MILLS	61801	
JAMES F. MINER	55455	
JESSE D. MIXON	75961	
JAMES MOLONEY	14420	
JOHN MONTAGUE	87545	
MAURO MONTESI	40122	ITALY
R. MOREL		SWITZERLAND
CARROLL MORGAN	2006	AUSTRALIA
RONALD G. MOSIER	48221	
STEVEN S. MUCHNICK	66045	
JUDY MULLINS	S09 5NH	UNITED KINGDOM
NEWTON J. MUNSON	13676	
CHARLES E. MURPHY		LIBYA
HARRY M. MURPHY JR.	87117	
H.-H. NAGEL	2	GERMANY
T. A. NARTKER	87801	
JOHN NAUMAN	55455	
MARK S. NIEMCZYK	60015	
R. K. NORDIN	55454	
THEODORE A. NORMAN	84602	
DAVID A. NUESSE	54701	
JOHN NUNNALLY	72143	
CAROL A. OGDIN	22314	
RICHARD OHRAN	84602	
RON OLSEN	07733	
LENNART OSKARSSON	S-431 20	SWEDEN
MARK OVERGAARD	92093	
DANIEL M. O'BRIEN	61820	
MAURICE O'FLAHERTY	ANTRIM	UNITED KINGDOM
P. G. PAGAN	A1C 5S7	CANADA
PETER PAWELCZAK	10019	
PATRICK PECORARO	85721	
SHMUEL PELEG	20742	
WALT PERKO	55414	
DAVID PERLMAN	55455	
W. W. PETERSON	96822	
CHARLES PFLEEGER	37916	
BOB PHILLIPS	97212	
ALAIN PIROTTE	B-1170	BELGIUM
STEPHEN A. PITTS	73110	
RUDOLPH C. POLENZ	54701	
BARY W. POLLACK	V6T 1W5	CANADA
GEORGE POONEN	02168	
L. C. PORTIL	N9B 3P4	CANADA
J. L. POSDAMER	13210	
FRED W. POWELL	24401	
RON PRICE	07726	
WILLIAM C. PRICE	91107	
ANDREW S. PUCHRIK	22090	
BRUCE A. PUMPLIN	54701	
HOWARD D. PYRON	65401	
DOUGLAS H. QUEBBEMAN	47130	
IRVING N. RABINOWITZ		ISRAEL
V. LALITA RAO	18015	
WAYNE RASBAND	20014	
JEFFERY M. RAZAFSKY	64108	
ROY F. REEVES	43220	
PHYLLIS A. REILLY	90746	
ROBERT REINHARDT	61 000	YUGOSLAVIA
JOHN REYNOLDS	N8	UNITED KINGDOM
GEORGE H. RICHMOND	80309	
PETER A. RIGSBEE	20375	
MARK RIORDAN	48824	
CLARK M. ROBERTS	91016	
JOE C. ROBERTS	21851	
KEN ROBINSON	2033	AUSTRALIA
STAFFEN ROMBERGER	S-100 44	SWEDEN
BERNIE ROSMAN	01701	
E. L. ROWE	19301	
DAVID ROWLAND	97229	
BRIAN G. ROWSWELL	2006	AUSTRALIA
HERBERT RUBENSTEIN	55455	
NANCY RUIZ	87115	
MARK RUSTAD	55112	
FRANK RYBICKI	19122	
JONATHAN SACHS	60604	
TOSHIAKI SAISHO	143	JAPAN
ANTTI SALAVA	SF-00100	FINLAND
A. H. J. SALE	7001	AUSTRALIA
TIMOTHY J SALO	55455	
CHESTER J. SALWACH	18960	
TOM SANDERSON	87002	
HORST SANTO	D-5202	GERMANY
DAVID SARANEN	55792	
AARON SAWYER	02035	
BOB SCARLETT	55455	
G. MICHAEL SCHNEIDER	55455	
SERGIO DE MELLO SCHNEIDER	13560	BRAZIL
ERIC SCHNELLMAN	98117	
STEPHEN C. SCHWARM	19898	
FRED L. SCOTT	33314	

WAYNE SEIPEL	78712	
GUISEPPE SELVE	40122	ITALY
SHARAD C. SETH	68588	
ED SHARP	84112	
DAVID ELLIOT SHAW	94022	
WILLIAM F. SHAW	01754	
BEN SHNEIDERMAN	20742	
JAMES P. SHORES	06320	
LEO J. SLECHTA	55165	
BARRY SMITH	97221	
BROOKS DAVID SMITH	53211	
ROBERT J. SNYDER	46202	
THOMAS C. SOCOLOFSKY	48823	
MARY LOU SOFFA	15260	
N. SOLNTSEFF	L8S 4K1	CANADA
DAVID SOLOMONT	02155	
MARCO SOMMANI	I-56100	ITALY
NORMAN E. SONDAK	01609	
STEPHEN SOULE	T2N 1N4	CANADA
HENRY SPENCER	M5W 1N5	CANADA
RICHARD D. SPILLANE	07470	
ROD STEEL	97077	
EDWARD STEEN	01852	
GORDON A. STEGINK	49401	
HAL STEIN	47401	
ALBERT STEINER	60201	
ANNE STOCCO	N1G 2W1	CANADA
A. I. STOCKS	70504	
JOHN P. STRAIT	55455	
GEORGE O. STRAWN	50011	
ROBERT A. STRYK	55424	
PREBEN TAASTI	DK-9000	DENMARK
RAMON TAN	18016	
ANDREW S. TANENBAUM		THE NETHERLANDS
DAVID TARABAR	01701	
H. TAYLOR	K1N 6N5	CANADA
JANET TAYLOR	75275	
RICHARD TAYLOR	80212	
WILLIAM P. TAYLOR	94550	
MICHAEL TEENER	90403	
R. D. TENNENT	K7L 3N6	CANADA
TED TENNY		
GARTH		
RON THOMAS		
JACK THOMPSON		
LARS-ERIK THORELLI	S-100 44	SWEDEN
LAVINE THRAILKILL	40506	
ALAIN TISSERANT	54042	FRANCE
STEPHEN TITCOMB	18017	
NOBUKI TOKURA	500	JAPAN
HOWARD E. TOMPKINS	15701	
ALFRED I. TOWELL	47401	
ASHOK N. ULLAL	D-7401	GERMANY
BRIAN W. UNGER	T2N 1N4	CANADA
JOHN URBANSKI	55403	
INDULIS VALTERS	55406	
J. J. VAN AMSTEL		THE NETHERLANDS
ANDRIES VAN DAM	02912	
PATRICIA VAN DERZEE	45036	
H. VAN LOON		THE NETHERLANDS

T. J. VAN MEERT	8131	THE NETHERLANDS
WILLIAM J. VASILIOU JR.	03824	
ROBERT D. VAVRA	55113	
B. VENKATESAN	T2N 1N4	CANADA
EIITA WADA	113	JAPAN
WILLIAM M. WAITE	80302	
TERRY M. WALKER	70504	
RANDALL W. WARREN	55440	
SCOTT K. WARREN	77098	
MASARU WATANABE	222	JAPAN
GEOFF WATTLES	55414	
JOHN A. WEAVER	18018	
NEIL W. WEBRE	93401	
WALLY WEDEL	78712	
W. WEHINGER	7000	GERMANY
RUTH WEINBERG		ISRAEL
LEONARD H. WEINER	79409	
STEVE M. WEINGART	55113	
JOHN WERTH	89154	
JOHN P. WEST	30332	
TERRY E. WEYMOUTH	68510	
GEORGE H. WILLIAMS	12308	
JOHN H. WILLIAMS	14850	
GARY W. WINIGER	94088	
GREGORY J. WINTERHALTER	30092	
NIKLAUS WIRTH	94304	
DAVID S. WISE	47401	
BILL WOOD	55455	
JOHN D. WOOLLEY	98006	
JACOB C. Y. WU	20910	
STEPHEN W. YOUNG	47401	
L. W. YOUNGREN	55901	
GIDEON YUVAL		ISRAEL
RUSSELL W ZEARS	77550	
PETER H. ZECHMEISTER	55117	
E. C. ZIMMERMAN	44691	

Indexed Files.

by   S. Knudsen, Institut fur Informatik
                E. T. H., Zurich
translated by   J. H. Loesch, SSRFC
                University of Minnesota

In addition to the possibility of dividing sequential files into segments (creating a "segmented file"), it is also possible to construct, read, and modify indexed files. This feature also covers the need for rapid location and modification of segments.

An indexed file may be thought of as a sequential file divided into segments (that is, as a segmented file). Each segment describes a possibly empty series of component-type elements and is a "logical record" in CDC SCOPE terminology. In contrast to segmented files in which a segment can be located through the use of a segment number relative to the previous segment, a segment of an indexed file can be found through the use of a specific segment reference address (a so-called random index), which is returned from the system during the write operation.

Declaration:
<file type> ::= indexed file of <type>

Example:
type ift = indexed file of t

The component type <type> cannot be char: Indexed textfiles are not implemented.

The standard functions EOF and EOS are defined as for segmented files and are likewise valid with indexed files. The standard procedures PUT, GET, and RESET (hence READ and WRITE) are defined as for segmented files. The procedures REWRITE, PUTSEG, and GETSEG, however, are defined for indexed files as follows:

RESET(f) positions f at the beginning. This allows the first of the segments described by the file to be read.

REWRITE(f) initializes the writing of a new segment at the end of the file f. The new segment is therefore not written at the beginning.

PUTSEG(f,k) must be called when a new segment is to be closed. The segment index (of type $1..2\uparrow24-1$) corresponding to the segment location is returned in k.

REWRITE(f,k) initializes for rewriting the segment with index k. k must be an index that was returned from PUTSEG.

PUTSEG(f) must be called to close a rewrite operation.
    NB. If a segment that is longer than the original segment is rewritten, segments following it may be overwritten.

VL ARTICLES

(FORMAL SUBMITTED CONTRIBUTIONS)

GETSEG(f) is called to initialize the reading of the next segment. An indexed file can therefore be read as a sequential file.

GETSEG(f,k) initializes the reading of a segment with the index k. k must be an index that was returned from a call to PUTSEG.

Some program examples will clarify the way in which indexed files are used. The basic declarations are:

var index: array [1 .. n] of t;
    i, k: integer; p: boolean;

- Write an indexed file f with n segments; the reference address of each segment is maintained in an array of indices.

```
rewrite(f);
for i := 1 to n do
  begin
    while p do
      begin (* fill f↑ *);  put(f) end;
    putseg(f, index[i])
  end
```

- Append a new segment. Its index is returned in k.

```
rewrite(f);
while p do
  begin (* fill f↑ *);  put(f)  end;
putseg(f,k)
```

- Sequentially read an indexed file.

```
reset(f);
while not eof(f) do
  begin
    while not eos(f) do
      begin (* inspect f↑ *);  get(f)  end;
    getseg(f)
  end
```

- Read a segment with index k.

```
getseg(f,k);
while not eos(f) do
  begin (* inspect f↑ *); get(f)  end;
```

- Rewrite a segment with index k.

```
rewrite(f,k);
while p do
  begin (* fill f↑ *);  put(f)  end;
putseg(f)
```

ARTICLES

(FORMAL SUBMITTED CONTRIBUTIONS)

The Need for Hierarchy and Structure
in Language Management

by

G. Michael Schneider

Department of Computer Science

University of Minnesota

I find it quite ironic that so much concern is being paid problems of
structure and organization of statements within the PASCAL language but so
little to the structure and organization of the management of the language
itself. By this I mean that there is currently lacking a formal administrative
hierarchy for the handling of questions relating to language standards,
specifications, and extensions.

When PASCAL usage was small and consisted of only a few installations,
language management could easily be handled by doing whatever you wanted to or
by verbal agreements among all parties concerned. Disagreements could be
settled by simple exchange of letters, telephone calls or over coffee.

The usage of PASCAL has, I believe, now left this embryonic stage of development.
Simply witness the nearly 500 members of the PASCAL Users Group or the dozens
of universities now using it as their primary language.

Yet while the growth of the language has been phenomenal the administration
of the language has not. It has remained a loose knit, informal mechanism
composed of the creators, users, and maintainers of the language. This is a
chaotic way to administer any large system and, worst of all, leaves the
language open to chaotic, unstructured growth. It is also frustrating. To
whom do we submit suggestions on changes, deletions, improvements, or extensions
to the language? To whom do we submit our "beautifully lucid" arguments on
what needs to be done? Currently there is no one. This groundswell of frustra-
tion was clearly demonstrated by the dozens of letters received by the Newsletter
shortly after it began, which described suggested improvements or changes. A
few of the suggestions I felt were good, most quite bad. That, however, is not
the important point. What is important is that these letter writers had been
searching for a vehicle to formally submit proposals and immediately leaped at
the Users Group and its publication as just that vehicle. But, the Users
Group has absolutely no official status as the arbiter of language standards.
The needed administration is still lacking.

What I propose is that we (the User's Group members) begin to discuss what is
needed for the proper administration of PASCAL. I initially suggest that
we adopt the following proposal:

The PASCAL User's Group nominate and vote on a PASCAL
Standards Committee composed of about 10-15 members.
This committee must initially perform three functions:

  1) Attempt to seek formal recognition for itself
     with such groups as SIGPLAN, ACM, and ANSI.
  2) Certify an official PASCAL standard. While this
     will probably be the specifications found in
     the PASCAL report, it should clear up certain
     "grey areas" (e.g., dispose).
  3) Draw up a "constitution" which spells out the role
     of the committee, its term in office, the
     philosophy to be used in evaluating proposed
     standards, and a formal procedure for submitting
     proposals to the committee.

The committee should now accept and consider suggestions
from throughout the user community. It should solicit
opinions and arguments on the proposal, evaluate all
suggestions in light of the stated philosophy
of the PASCAL language and decide to reject it, accept it
as a new standard, accept it as a standard extension, or
postpone any decision. Major decisions could be put to
a vote of the full membership if necessary.

The above proposal omits a great amount of detail that can be worked out by
the committee and the membership. It would be presumptious of me two impose
any further my own feelings on how such a standards committee should operate.

What I care about are not really the details anyway. I care about bringing
order and structure to the area of language management -- the same goals that
PASCAL brought to language design.

On the suitability of a Pascal Compiler in an
undergraduate teaching environment

Before Pascal was adopted by my parent department for teaching purposes,
it was necessary to demonstrate that a suitable compiler was available.
We have access to a CYBER 72 running a timesharing service under NOS and
consequently acquired from Zurich the 6000 -3.4 compiler. The performance
of the compiler during installation gave rise to a great deal of optimism and
a few reservations. The optimism stemmed from the quality of the compiler;
the reservations from a few obvious problems caused by the change from
SCOPE 3.4 to NOS. These problems were almost entirely caused by the change
in the method of use of the compiler not by defects on the code.

The local modifications were all introduced with one purpose in mind -
to facilitate the use of Pascal for undergraduate teaching.

The modifications can be roughly divided into two categories.

1. **Modifications to ease the use of Pascal**
   a) the compiler ignores leading line numbers
   b) compilation diagnostics are sensible with the L-option
   c) post-mortem dump output is re-formatted for 70 character wide devices.
   d) dayfile messages were re-ordered so that the fault reason appears on
      the terminal.
   e) terminal control introduced - a user interrupt will produce a post-mortem
      dump.
   f) the post-mortem dump gives traceback information in terms of line
      numbers not core addresses

2. **Modifications to improve throughput**
   a) A G+ option to automatically enter a correctly compiled program
   b) A W option, which allows the use of blank common for stack + heap.
      This reduces the possibility of rollouts which may be caused if a
      memory request for an increase in field length were made.

   Note
      To minimise store requirements we wish to run Pascal in REDUCE mode,
      and under NOS the KRONOS 'trick' to avoid field length reduction
      after a relocatable load does not work.
   c) Output buffers which are on-line to a terminal are not flushed by the
      Pascal run-time system at the end of a run. This is left to the time-
      sharing system. This change was made as the result of a poor benchmark
      performance.
      To demonstrate that the performance of the compiler was satisfactory
      a simple benchmark was designed to compare Pascal with Algol 60 and

2.

Fortran. It was believed that this benchmark would saturate our
system.

The benchmark consisted of running 75 jobs as rapidly as possible from
15 terminals (5 from each terminal) with no other users on the machine. The
experiment was repeated 4 times for each language; each time with a different
job. The amount of real time elapsed was measured in each case. These figures
include the time for terminal I/O which was expected to be small by comparison
with the total time.

For the experiment we used:
   a Zurich Pascal compiler with local mods (but not 2c above)
   an FTNTS compiler
   an ALGOL 4 compiler via a procedure file which included utilities
   for handling the line number problem.
and   15 'volunteers', many of whom had never used the system before.

Jobs 1 and 2 involved similar programs. In Job 1 the program was
altered to introduce a compilation fault. Job 2 compiles OK and is executed.
Jobs 3 and 4 are related in a similar way. The programs used were genuine
student exercises. The 'same' program was used for each language.

Results

Job Number	Time in seconds for		
	Algol	Fortran	Pascal
1	701	245	426
2	1956	458	190
3	846	254	*
4	2967	440	220

*The Pascal experiment was terminated as the performance was
unsatisfactory.

Modification 2c was included and experiment repeated. The results
were 153, 155, 141 and 201 seconds respectively.

These figures are interesting for two reasons
   1. the improvement from 426 to 153 for Job 1
   2. the fact that Job 3 took less time that Job 1
Fact 1 can be explained by the introduction of the extra modification which
reduced the core <-> disc traffic by
          75 x 4 x 50000B words per benchmark
        = 61.4 million characters
for the benchmark involving compilation errors.

Fact 2 can be attributed to the human learning process. As the experiment progressed the volunteers were able to type the commands faster because they were more familiar with the system.

In fact the performance of the Pascal compiler is such that the figures presented for the second experiment can only be regarded as a <u>lower bound</u> on the throughput because the terminal I/O now accounts for a significant proportion of the time measured, e.g.

in a faulty compilation benchmark:-

no. of characters typed by human at a terminal = 65

"   "   "   "   "system at a terminal = 540

assuming typing speeds of 3 and 10 chars/sec. this accounts for 76 seconds at every terminal. It seems likely that the system is not being saturated by this benchmark when using Pascal.

Conclusions

1. The performance of Pascal is satisfactory
2. These figures represent a lower bound on its performance. More accurate figures would have required the use of a greater number of terminals (to saturate the system) and repetition of the experiments. In the context of the experiments this would have been a waste of time.

A. M. Addyman

PASCAL Potpourri

by Richard J. Cichelli

Topics for the PASCAL user:

Direct access files

"Standard" PASCAL

Software tools

Direct Access Files for PASCAL

The following is presented as an approach to direct access files in PASCAL. We begin with a discussion of current PASCAL file facilities.

Sequential Files in PASCAL

The PASCAL Revised Report defines only sequential files for PASCAL. Thus, a file is a sequence of zero or more items of the same type. A window, or buffer variable, into the sequence is defined. It is referenced by a buffer pointer. Only one element of a file may be accessed at any time. A predicate EOF (end of file) is defined such that when it is true, the operation WRITE (<file item>) or PUT (<buffer variable>) is valid. If EOF is false, READ (<file item>) or GET (<buffer variable>) is possible. As a side effect of the READ and WRITE operations, the buffer pointer is moved through the sequence. EOF becomes true during a sequence of READs when no items remain beyond the buffer pointer. EOF remains true after a WRITE. The operations RESET and REWRITE move the buffer pointer to the beginning of the sequence.

PASCAL sequential files look like tapes.

The Notions of Direct Access Files

Most mass storage based operating systems present files to the user as named data sets (i.e. groups of related items associated under a cataloged name in a directory). The user can request access to a data set by supplying the system with its name. PASCAL sequential files are easily provided in most operating systems. However, the vast majority of third generation operating systems give the user an alternative to the tape-like file organization capabilities of PASCAL files. This alternative allows data items to be accessed <u>directly</u>. That is, if the "file" consists of 1000 items, the user can access the 439th without passing the 438th or rewinding from the 440th.

For direct access files there is no notion of a buffer pointer, and thus there is no EOF, RESET, or REWRITE. Any item may be read or modified "in place". READs and WRITEs can occur in any order.

The nearest notion to this idea that is defined in the PASCAL Report is that of <u>arrays</u>. I propose to extend the PASCAL array concept to provide direct access facilities.*

To accomodate this extension to the language, I propose that the type declaration for arrays be extended from

to

† Andy informs me that someone has already implemented indexed files in PASCAL. From my conversations with him; I do not believe I am duplicating this effort.

A "long" array will be one which <u>might</u> reside on direct access secondary mass storage.

Consequences of the Notation

Treating direct access files as arrays requires only relative record I/O capabilities from the operating system. It seems to me that this provides the potential for all direct access facilities at the most fundamental level. It suggests that such advanced notions as "indexed sequential access" will have to be implemented by the programmer or as utilities in terms of the above primitives.*

Implementation Details

Direct access files are used in two basically different ways - as bulk temporary work space and for fast, non-sequential access to permanent data using keys based upon content or relationships. To serve the first need, the <u>long array</u> can simply be a local array variable. In a virtual memory environment the word "long" might be ignored by the compiler. As far as the programmer is concerned, this type of long array is equivalent to a (possibly) slow access array.

For the second case, long arrays are global to the program. They will be named as formal file parameters in the program heading just as global files are now. Their declarations will be in the variable declarations of the program, or level 0, block.

If a long array file doesn't exist when a program declaring it is executed, one should be created (and should remain upon program termination). If one does exist but is incompatible with

* This is quite simple. A relative record file (long array) is used for the records and another is used for the related index which maps from primary key designator to the record number (i.e. long array index). [...] Both arrays might be mapped into the same system file by using record variant parts for array elements.

the program definition, a fatal error should result. Other fatal
error conditions will arise if the actual file is sequential or
if it is the wrong size (i.e. any type mismatch).

Several programmer notations could be used to guide the compiler
in mapping the data items into efficient store. For example, the
declaration "PACKED LONG ARRAY" might-cause the compiler to try to
block the records efficiently. By extending the dollar sign
comment notation, the programmer might suggest blocking factors
and the number of core resident direct access record areas.

User Interface

Long arrays will be used exactly like in core arrays. Of
course, a long array of files or a long array of long arrays
can not be premitted. Other than this obvious restriction, a
long array will be like any other array. Access notation will
be identical.

There is one glaring disadvantage with this scheme; however:
When a programmer writes expressions using long array elements,
he may invoke significant overhead ... and it will be almost
completely invisible to him in the code text. The phrase
"long array" just doesn't suggest long moments of computer toil
to the programmer. The best sort for a long array may in no way
resemble the best for an in-core array.*

---

* I suspect, however, the days of slow access rotating magnetic
storage are limited. Solid state bulk memory seems destined to
overtake disks. Our notation may be more appropriate for the
future than the present.

Conclusion

I suggest that the concept of "long arrays" is sufficient
for direct access file facilities and is consistent with the
design goals of PASCAL in its simplicity and clarity.

Standards and the Language PASCAL

The standards game is one played by programming language
users. Those with problems in search of solutions look for new
language features. Those with programs searching for customers
with problems look to enforce standards. We PASCALers should
have a total view of the standards problem. We should realize that
no existing programming language standard is a success at its
stated goals and that no language has succeeded without a
standard. With respect to standards, PASCAL has significant advantages
over the popular poorer languages. First, it is defined only in
the Revised Report and not in some jumpout implementation.
Second, as a language it seems particularly easy to formalize in
a humanly understandable fashion. In short, it has a small and
regular syntax.

Do We Need a Formally Recognized Standard?

Let us consider the population concerned with PASCAL and
PASCAL standardization: language designers, language implementors,
program writers, and employers of program writers.

Language Designers

In our case, this is Dr. Niklaus Wirth. He says PASCAL is
what he says it is. But, in fact, PASCAL is too important and
too widely used to have its scope defined and limited by one man.

We all have a legitimate say in this and we can and should exercise
our responsibility. It is important, however, to recognize that
the current success of PASCAL is based on its eloquent design.
We must seek to preserve its simplicity and clarity above all else.

Language Implementors

Dr. Urs Ammann and his group implemented PASCAL in an efficient
and robust fashion on the CDC 6000 computers. Because many users
confuse a programming language with its particular implementations,
Ammann's fine implementations have been the wellspring of PASCAL
user growth. Because many implementors have followed Ammann's lead,
it is likely that the PASCAL compiler is the most efficient
language processor at any shop which has one.

Implementors desire standards to guide their compiler writing.
Frequently however, in order to interface or compete with existing
languages, they stretch or reinterpret the standards to meet real
or imagined user implementation needs. Implementors and compiler
maintainers should take great care not to let ad hoc patches to an
implementation become de facto changes to the language standard.

Fortunately, no hardware vendor has tried to make PASCAL
its own. But we all know that PASCAL will soon be a vendor product.
This should not be viewed as auguring potential corruption, but
instead as a sign of maturation. We should recognize it as such
and provide vendors with an excellent standard to work from. I
personally anxiously await the day when Seymour Cray, Gene Amdahl,
and Ken Olsen market PASCAL machines.

Users: Managers and Programmers

For obvious reasons, organizations and their representatives
(i.e. managers) want standardization in a programming language.
Every organization has learned Whitney's lesson about
interchangeability. In programming this means adherence to
standards.*

Programmers have problems to solve. There are things which
could be added to PASCAL that might make one programmer's job
easier. The problem is to address the entire user community.
Frankly, some languages are better than PASCAL for some applications:
use COBOL's Report Writer for reports, use SNOBOL for string
manipulation, etc. PASCAL can't be all things to all people and
still be simple, concise and easily implemented. Remember the
PL/I syndrome - multi-million dollar compilers won't solve
anyone's problems. There is a revolution coming in computer
software as more programmers learn how to do more things simply.

Getting a Recognized Standard

A standards committee should be set up. (I would particularly
like to see Dr. Waite as a member.) This committee would represent
users and designers first, implementors and vendors second. Its
purpose would be to get a document approved by both PUG members
and the ANSI-X3J3 committees. International standardization is
also desirable. Additionally the committee would be charged with

---

* It is the naive manager who thinks hardware vendors desire
standards. As the current efforts with big languages indicate,
all they want to do is exclude the competition.

certifying that a particular implementation conforms to the standard.

Only with formal recognition will PASCAL be adopted by large conservative organizations and selfish vendors.

There is danger is having a committee for this purpose. When COBOL was being designed two committees were formed. Since the problem of business data processing was regarded as so big, one committee was asked to deliver a quick interim report to use to "make do." The second committee was to solve the DP language problem. The first committee report is in - its product was COBOL. We are still waiting for the long range committee's report.

One final word on previous failures. The new FORTRAN is an obvious disaster; the PL/I standard is an abomination. We can do better! We need be neither upward compatible with previous errors nor a vendor's puppet. We can do it right if we get together and try.

Software Tools for PASCAL

PASCAL implementations for new environments are occurring with ever increasing frequency. As PASCAL is used for more and more production programming, it is important that a universal set of ancillary software tools be agreed upon. Some of these tools can be defined in an environment independent way so that when written in standard PASCAL they can become part of a universal PASCAL software development facility. I here propose an initial list. With PUG membership help the list will develop into a working specification and a powerful set of programming tools.

PASCAL Compilers

Currently there exist PASCAL compilers which produce absolute code, relocatable code, macro code (PASCAL-J) and interpreted code (PASCAL-P). Portable versions exist (PASCAL-P and PASCAL-J). Compiler trunks exist. A standard PASCAL subset (PASCAL-S) exists.

For compiler writers there should be a standard PASCAL language test set. This universal set of PASCAL programs would exercise new PASCAL compilers and help implementors gain confidence in the correctness of their compilers.

An interactive interpreter should be developed. This system would provide interactive symbolic run time debugging facilities; breakpoints; interactive dumps; etc. It should be easy to use better than PL/I's checkout compiler.

The Lecarme and Boehmann compiler writing systems are also important tools for any shop engaged in language development.

Source Program Tools

Wirth has written a cross reference program. Perhaps, if the variable names were improved, a standard version of this program could be among the software tools. A formatter or "pretty printer" is essential for producing documentation quality listings. Mike Condict's might be a good starting place.

A code instrumenter is a very important debugging and refining tool. Instrumenters insert statement counters or timers so that reports of relative usage of code can be made. An instrumenter is invaluable in optimizing programs.

A high level macro preprocessor would also be a valuable facility.

Source Libraries

The CDC source library utility program UPDATE is currently used for distribution of the SCOPE versions of PASCAL. It seems to me that a mini-version of UPDATE (with only sequential program libraries) could be implemented in PASCAL. This would help standardize the distribution of PASCAL tools. (Incidentally, CDC's UPDATE is the best source library system I have ever seen. I think its quality should be emulated.)

For truly large systems (50,000+ lines) a source code data base is desirable. Such a system keeps track of which programs access what data and provides for standard file and record descriptions among programs, etc. I understand such a system for PASCAL exists but is a deep dark military secret.

Documentation Preparation

W. Burger implemented part of Waite's PLAP in PASCAL. We need a universal PLAP-like tool to maintain manuals and other documentation in machine readable form. Justification and hyphenation and facilities for producing high quality printing in upper and lower case should exist. PASCAL documentation should be distributed in machine readable form for ease of publication and distribution.

Object Program Facilities

Work is now in progress on programs which load PASCAL absolute binaries. Facilities for overlay processing should be provided. Automated aids which help create effective overlay structures should be provided. A binary decoder is also a useful tool.

Other Programs

An efficient table processor with facilities like COBOL Report Writer would be desirable. Current work on PASCAL data base management systems, mathematical function libraries, and computer aided instruction systems augur the day of increased use of PASCAL in business, engineering, and education. In the area of function libraries (for mathematics or business), facilities should be provided for not only linking in binary modules but also for including source modules. *

Conclusions

Obviously, where environmental conditions permit we should have a universal PASCAL program implementing each software aid. Where the environmental factors prevent this, we should seek to provide a standard user interface to the desired functions.

Conclusion

The ideas presented in this paper are perhaps still ill-formed. They are meant as a starting point for serious discussion. I hope there will be reaction and feedback from PUG members.

---

* In my opinion, merging programs at the source level is to be preferred to binary level linking. PASCAL compilers are typically faster than linking-loaders.

(*Received 10/12/76*)

THE CASE FOR EXTENDING PASCAL'S I/O

Michael Patrick Hagerty

Abt Associates Inc.

With the introduction and subsequent increase in popularity of PASCAL, a number of papers concerning the language, its features and deficiencies, have appeared in various journals and newsletters. Champions of the language have extolled the virtues of its structure and unambiguous grammar using both example and theory as justification of its usefulness. PASCAL critics, on the other hand, have questioned the claim of the proponents that PASCAL will replace FORTRAN, pointing to the inadequacies of the language in several areas. Wirth (1974) defends the absence of certain "favorite features" as necessary to avoid inefficient programming solutions or reliance upon features which are contrary to the aim of clarity and reliability. When the features being debated refer to the flexible input of large amounts of data, the critics hold the stronger hand, and with much justification.

As a user of PASCAL in an environment where large files of data are the rule rather than the exception, I find the argument that PASCAL's native input facility is sufficient to be without merit. Much of the data analyzed at AAI is produced by the Bureau of the Census or other government agencies and is available only in fixed-format records in multi-file volumes. The absence of a formatted input capability is not merely inconvenient in this instance; it is self-defeating. Several alternatives have been adopted as stopgap measures, including the use of FORTRAN subroutines to handle all read operations. However, it is obvious that if PASCAL is to become one of the more common languages, it must possess an I/O capability which is useful to those who process large amounts of data as well as the compiler writers.

To gain insight into what is required, it is first necessary to examine the deficiencies in PASCAL from the data analyst's point of view. The following list represents a minimal set of those deficiencies:

- PASCAL I/O is asymmetric in that no READ operation exists which is the inverse of the formatted WRITE.

- PASCAL I/O is further asymmetric in that certain types (ALFA and BOOLEAN) may be written, but cannot be input using the native READ procedure.

- Although the most powerful facet of PASCAL is its structuring facility, there exists no simple, direct method of transmitting RECORDS to and from formatted textfiles.

- PASCAL requires the inefficient use of data storage media by not allowing the user to maintain his data in multi-file volumes. Only data between the portion immediately addressed upon RESET and the first EOF can be examined.

As the major portion of data collected in both the business and research communities is stored as formatted files of records more or less card-image size, the absence of a feature which will allow the direct read of specific columns rather than freefield is an extreme shortcoming. The choice of formatted files was not made only for convenience, although the availability of this feature in other languages did encourage its use. It does require space, disk or tape, to store large amounts of data, and the requirement that each variable be separated from its neighbors by a blank(s), gobbles up more space, and therefore costs more.

If it were only the very large data bases which were formatted, an argument could be advanced for special, custom-tailored I/O for these applications. This position loses ground when considered in light of most applications packages which allow both form of input. AAI is a very heavy user of the SPSS and other statistical packages. Within the past year, the freefield input facility of SPSS has been exercised only twice: once to test that it worked correctly; and once again on a problem with only ten cases. With any survey of over 10-20 observations, it is also much more economical (and accurate) to have the data collected in fixed format without blank delimiters.

In the present situation, each user community is left on their own to develop and implement as part of their library, a formatting reader which meets their own needs. The upshot of this, as clearly described by Eisenberg (1976), is that PASCAL will become another BASIC in the area of I/O. As most users are aware, BASIC programs from one system have a very low probability of running under another system as each manufacture, or vendor, chooses to implement I/O in a slightly different manner. The computing world can well do without this form of chaos.

Turning to the second area of concern, we find that there exist certain types of variables in PASCAL which possess a unique property; although we may print them out to see what value they are assigned, it is not possible to read them in directly. ALFA and BOOLEAN are examples of these types, however other implementations may be busy installing additional varieties. In languages which preceeded PASCAL, an ironclad rule existed for I/O, "If you can write it out, you can read it back in."

The third mentioned deficiency is directly related to the first two. It is incongruous that a language as well structured as PASCAL would fall into the trap of requiring the user to transmit the elements of his well-structured records element by element. This deficiency is magnified by the lack of a defined formatting facility for input, the existence of the "special" types, and the absence of a formatting tool which would tie the size of the individual elements to the order within the record itself. FORTRAN has FORMATs; COBOL and RPG have PICTUREs; and PASCAL has nothing comparable. It has been rumored that one implementation of PASCAL uses FORTRAN FORMATs, although this hardly seems

to be the optimum solution to the problem. What is needed is a fresh look into what it means to tie a format specification, even if it implies freefield, to a given element of a structure. Unlike FORTRAN, it should be capable of diagnosing at compile time attempts to read or write integers with decimal points and other such errors.

The fourth deficiency is representative of the attempt to stay clear of defining what constitutes the implementation of files within the context of a system. By requiring that all data sets processed by PASCAL consist of one file, the interface between various systems is kept simple. Unfortunately, this requires the user to either keep one file on each tape or disk library, or copy off a single file from the multi-file volume to satisfy PASCAL. Keeping partially filled tapes and/or copying desired files does require added expense.

The remainder of this paper will be devoted to several recommendations which, if adopted, will remedy most of the problems in the area of I/O. The form of the recommendations will be to first present what the construct will look like, followed by an explanation of how it is to be implemented. Each of the constructs will be based within the scope of the following declarations:

```
CONST  NCPW = (* Number of Characters Per Word - ALFA TYPE *)

VAR A: ALFA;
    B: BOOLEAN;
    I: INTEGER;
    T: TEXT;
    F: FILE OF arbitrary type;
    N: INTEGER;  (* number of characters read or written *)
    D: INTEGER;  (* number of places to the right of decimal *)
```

READ (T, I:N) will convert N characters beginning with the current position of the file T to INTEGER and store the result in I. Leading blanks are to be treated as zeros, but trailing or imbedded blanks represent an error and should be diagnosed as such. The number may be signed or unsigned.

READ (T, R:N:D) will convert N characters beginning with the current position of the file T to REAL reserving D characters as the fraction. Blanks before the whole number and following the fraction are to be considered zeros. Imbedded non-digits are errors. The only exception is that a decimal may be punched in the data which would override the D specification.

READ (T, B) will read a BOOLEAN variable B freefield from the file T. TRUE and FALSE will be the allowed character patterns.

READ (T, B:N) will be expected to find the characters TRUE or FALSE within the next N character on the file T. Where N is less than 5, only the first N characters will be matched.

READ (T, A) will store a left-justified ALFA of at most NCPW characters in A. The variable will begin with the first occurring non-blank character on the file T and continue with characters until the number of characters is equal to NCPW, a blank character is encountered, or EOLN(T)=TRUE. While spanning leading blanks, EOLN will be ignored. EOLN will be cleared on return from the procedure if it terminated the transfer of characters.

READ (T, A:N) will store the following N characters from the file T left-justified in the variable A with blank padding out to NCPW. No more than NCPW characters may be transferred. EOLN will, as an installation parameter, cause a normal termination with added blanks out to NCPW.

OPENR (F) will cause the system to RESET a file without rewinding it. This allows the user to position the file before executing the PASCAL program.

OPENW (F) is the non-rewinding version of REWRITE.

PUTEOF (F) instructs PASCAL to write the output buffer, followed by an EOF, and invoke OPENW beyond the EOF. PUTEOF is the tool for creating multi-file volumes, a PUTSEG for EOFs.

GETEOF (F) will cause an End-of-File to be read (skipped), with the concurrent resetting of EOF(F) to FALSE. OPENR(F) will then be invoked to open the file past the EOF. If two contiguous EOFs are detected, this will imply the end of the volume, and EOF(F) will be reset to TRUE upon return from GETEOF(F). If EOF(F) is not initially TRUE, data will be skipped until the EOF is encountered, and the normal processing of GETEOF will continue.

GETEOF (F, N) instructs the system to skip N EOF marks on file F. When the N parameter is specified, GETEOF is analogous to GETSEG(F,N), with file, instead of segment marks. GETEOF(F) is equivalent to GETEOF(F,1).

The above eleven proposed extensions to the language are directed to overcoming three of the four mentioned deficiencies. The code necessary to implement these features is simple and readily installed in any complete implementation of PASCAL. (As an example, although trivial, the code for reading ALFAs is included as an Appendix). The observant reader will notice that no attempt has been made to provide a workable solution for the third problem: formatted and freefield I/O for structures.

It is conceivable that a mechanism can be devised which is compatible with the syntax, grammar and structure of PASCAL, and will allow the separate assignment to different elements in a structure of specific formats. After giving the matter much thought, I am at a loss to produce a new and uniquely appropriate scheme. It appears at first blush that some hybrid of PICTUREs and FORMATs might work.

For want of an adequate solution, we at AAI have adopted a strategy which we know to be flawed. It is unacceptable as a solution to the problem of records and textfiles because it simply ignores the existence of textfiles altogether. By specifying a single word format descriptor for each element in a record, it is a simple matter to have a procedure decode a whole record quite efficiently. By building arrays of

```
TYPE FORMAT = PACKED RECORD
              FTYPE: (ALFA, BOOLEAN, INTEGER, REAL);
              DSIZE: 0..777B;        (* DECIMAL PLACES *)
              NSIZE: 0..777B;        (* NUMBER OF CHARACTERS *)
              SBIT:  0..7777B;       (* STARTING BIT IN WORD *)
              NWORD: 1..777777B      (* NUMBER OF WORD IN RECORD *)
              END;
```

of the same size as the records to be read, filling in the five elements with appropriate descriptors, and passing that array, along with a segment of text already read, the text can be converted.

The procedure we use has four parameters: a vector of characters; the FORMAT array; the resultant decoded RECORD of data; and an integer which specifies the number of elements to decode. For the sake of efficiency, the routine was coded in the assembly language of our machine. The only trick is to manage to get into PASCAL a whole segment of text to decode. This is accomplished by fudging the I/O buffer allocated by PASCAL and allocating a array on top of that buffer. Data is then read into that buffer, and decoded directly out. CDC, as well as most other manufacturers, provides a powerful read facility which will initiate a read of a specified number of words (or bytes), and read until either the list is satisfied, or the record on the input device is depleted. It is this feature which I would propose to be an implementation dependent feature.

READBUF (F, X, N) will initiate the reading of N items of X (which is and array with at least N elements) from file F. This operation merely initiates the read, but does not guarantee its completion.

WRITEBUF (F, X, N) is the inverse of READBUF and initiates the write of N elements of the array X. Once again, completion is not guaranteed.

COMPLETE (F) forces the system to complete any pending I/O operation on file F, entering a recall state if necessary.

BUFLENGTH (F) is a function which will return an integer representing the number of elements actually transferred on the last operation on the file F. It is clear that BUFLENGTH should not be used until COMPLETE has forced the end of the operation.

These additional, implementation dependent, features allow the machine to optimize its I/O operations and give the user the opportunity to overlap some independent processing while the machine attends to the task of moving data.

The sophisticated user should recognize that while the suggestions made in this paper apply most directly to the sequential access of large amounts of data (the issue of greatest importance to AAI), no attempt been made to come to grips with the host of other access methods. Files of every variety, indexed, keyed, and hashed, exist; the need now is to propose the manner in which PASCAL will interface them. We are forced to leave the matter as proposed above. While the facilities proposed with data structures in PASCAL, no effort should be spared to provide an equally powerful set of I/O operations. The challenge is to devise an implementation independent scheme for doing it.

REFERENCES:

Eisenberg, J., "In Defense of Formatted Input," PASCAL NEWSLETTER, Number 5, September, 1976.

Jensen, K., Wirth, N., PASCAL USER MANUAL AND REPORT, 2nd Edition, Springer-Verlag, 1976.

```
(*ST-,P-,E+,U+,X0   READ ALFA (LEFT-JUSTIFIED) FREEFIELD.  HAGERTY *)

PROCEDURE RDA (VAR F: TEXT; VAR A: ALFA);

   CONST NCPW = 10;    (* NUMBER OF CHARACTERS PER WORD *)
   VAR I : INTEGER;
       CHBUF: ARRAY[1..NCPW] OF CHAR;

BEGIN
   IF EOF(F) THEN
      BEGIN MESSAGE(≡* TRIED TO READ PAST FOS/EOF≡); HALT END;
   WHILE (F↑=≡ ≡) AND (NOT EOF(F)) DO GET(F);   (* SPAN LEADING BLANKS *)
   IF NOT EOF(F) THEN    (* COLLECT CHARACTERS FOR ALFA *)
      BEGIN
         I:=0;
         REPEAT
            I:=I+1;
            CHBUF[I]:=F↑;
            GET(F);
         UNTIL (F↑=≡ ≡) OR (I=NCPW) OR EOLN(F);
         IF EOLN(F) THEN GET(F);   (* CLEAR EOLN FLAG IF SET *)
         WHILE I<NCPW DO   (* FILL REMAINDER OF WORD WITH BLANKS *)
            BEGIN
               I:=I+1;
               CHBUF[I]:=≡ ≡
            END;
         PACK(CHBUF,1,A)
      END
END (* RDA *);

(*ST-,P-,E+,U+,X0   READ ≡NC≡ COLUMN ALFA IN FIXED FORMAT.  HAGERTY *)

PROCEDURE FRDA (VAR F: TEXT; VAR A: ALFA; NC: INTEGER);

   CONST NCPW = 10;    (* NUMBER OF CHARACTERS IN A WORD *)
   VAR I: INTEGER;
       CHBUF: ARRAY[1..NCPW] OF CHAR;

BEGIN
   IF EOF(F) THEN
      BEGIN MESSAGE(≡* TRIED TO READ PAST EOS/EOF≡); HALT END;
   IF (NC<1) OR (NC>NCPW) THEN
      BEGIN MESSAGE(≡* ALFA FIELD WIDTH ERROR≡); HALT END;
   IF NOT EOLN(F) THEN   (* COLLECT ≡NC≡ CHARACTERS *)
      BEGIN
         I:=0;
         REPEAT
            I:=I+1;
            CHBUF[I]:=F↑;
            GET(F);
         WHILE I<NCPW DO   (* FILL REMAINDER OF WORD WITH BLANKS *)
            BEGIN
               I:=I+1;
               CHBUF[I]:=≡ ≡
            END;
         PACK(CHBUF,1,A)
      END;
END (* FRDA *);
```

GENERAL THOUGHTS ON PASCAL

ARISING OUT OF

CORRESPONDENCE BETWEEN SOUTHAMPTON AND TASMANIA

Arthur Sale
1976 October 20
University of Tasmania

MIXED LANGUAGES

Here is the focus of the survival of PASCAL. If it is possible for
programmers to access and use the vast library of FORTRAN mathematical
routines that have been developed, then there is hope that the
scientific community might be encouraged to transfer their skills and
effort into PASCAL from FORTRAN. A tremendous benefit, and greatly to be
desired. If this is not possible (to mix languages) then even this slim
hope must fade away. The inertia and ecological success of FORTRAN is too
great for any naive competitor to survive and thrive.

I note many PASCAL compilers produce assembly code for their machine.
Shades of IBM7040! Still, the approach does allow mixed languages at
little cost, but some attention must then be paid to (i) efficiency, and
(ii) ease of use of the joint system.

On the Burroughs B6700, the problem is much more significant as no
assembly language exists. (For those who marvel at this, know that no
computing centre director would want one for the security risk it would
be (B6700 integrity relies on software), and know that the structure of
the executable code file would require an elaborate assembler to specify
all that is necessary...) Burroughs Algol is the lowest level of the B6700.
Consequently achievement of mixed languages requires either compilation
into Algol (disregarded!) or the construction of structured code-files
that are quite incredibly complex by the standards of monolithic machines.
The binder in fact has to be able to re-arrange code (especially in the
outermost block) for own and pre-defined objects; namespaces names;
check parameter compatibility; and all this for ALGOL/HUMAN/PUBLIC
at least.

Nevertheless, even in this case, the achievement of mixed-language programming
must be attempted. Even more must this be the case in simpler situations;
the only exceptions being mono-language systems such as Brinch-Hansen's.
And these are not addressed to the same purpose as viable general-utility
compilers.

PORTABILITY

It is important to realize that standardization is not a good in itself;
the present benefits to computer science of standardization are

    (1) transferability of skills between compilers,

    (2) portability of programs written in standard languages, and

    (3) exchange and development of compatible compilers is made more easy.

It must be realized that the semantics of the language is quite as important
as the syntax in realizing objective (2), and in this regard there are
any number of difficulties in standard PASCAL.

First of these is character set. Except for those computers that
persist with 6-bit characters, the EBCDIC and ASCII character sets must
be regarded as the de facto standards of the industry. All PASCAL compilers
should have the capability of working in either (preferably both) of these
two character sets. The implications are that no-one is justified in
inventing a new character set, nor in allowing any other character set to
be used unless it is a firm committment by the computer/operating system;
and that even in CDC and other 6-bit machines an effort should be made to
provide ASCII and EBCDIC characters. I can think of no more common
portability trap than that of ignoring character collating order. Pascal-P
was caught.

Another, not so obvious, is the practice of allowing any-length identifiers,
but permitting only the first few n characters to be significant ignoring
the tail. If a program is compiled on a computer with true any-length
identifiers (e.g. B6700), then it is quite on the cards that it will be
compiled incorrectly by some other computer without warning, as if the
names are TEMPATTOPOFKILN, TEMPATTOPOFFLUE. No, identifiers must be
names are TEMPATTOPOFKILN, TEMPATTOPOFFLUE. No, identifiers must be
true any-length, or fixed up to a length which at the very least a mandatory
warning or better an error thereafter.

There are many more portability traps which act so as to limit the
real portability of programs written on one computer to quite a long way below
that which is achievable at the present state of knowledge. They require
more attention in the case of PASCAL. I find it infuriating to receive a
program that the author claims is "portable" only to find that he or she
is obviously not aware of the most obvious requirements for portability.

INCLUSION OF SOURCE TEXT

The B6700 compiler has a feature (as with all Burroughs compilers) for
including source text from a named file in the compilation. The included
text may be, but is not usually, listed. It may include further included
files, up to nesting depth defined by the number of file buffers reserved.
(In the PASCAL compiler: a depth of 6). In fact this is used as a structuring
aid in the B6700 PASCAL compiler source, which consist of some 20-30 files
(some with alternatives) linked into a tree-like structure by inclusion
references. The facility is also useful for including library routines
(in PASCAL source of course) as in special I/O routines, mathematical
routines, graphics routines, etc. It may mean that the need for the use
of relocatable binary or linked versions of PASCAL programs...

The B6700 construct is a compiler option: it appears on a single line
which begins with a $, and has the following syntax:

    ──► $ ──► INCLUDE ──► filename ─────────────────────────►
                                └── sequencenumberrange ──┘

Examples:

    $INCLUDE PACKAGE1

    $INCLUDE ARTHUR/STANDARD/TYPES 30000-60000

Quite clearly, such a construct in PASCAL could also be embedded in the
compiler option Wirth has implemented, if only it were not so restrictive
in syntax. I would commend such a facility to all PASCAL compiler-writers,
preferably adhering to the above syntax. The default should be to omit
listing the included text.

FILES

Files are PASCAL's biggest problem. In a modern context, PASCAL's files
are an anachronism. PASCAL should have access to random-access files
as well as sequential; should be able to communicate with terminal devices
as well as card readers and line printers; should be able to at least
specify file attributes; and should have a record-oriented i/o subsystem.

On top of this files do not fall into the same class as other VAR objects,
since for the most part their life (extent) is not limited to the extent of
the program or a procedure thereof. They may be already existing (in which
case the declaration is more of the nature of a specification), or may live
on past the program's death (and often have to conform to external
requirements such as line-length).

Therefore all normal VAR operations on files should be carefully avoided.
The assignment and comparisons on files should be regarded as absolutely
meaningless, as should possibilities such as arrays of files, or records
containing files, and other similar nonsense.

The B6700 implementation will include attribute lists in the FILE type
declaration (whether in the TYPE or VAR part) according to the following
syntax:

```
      ──────▶ FILE ─┬─────────────────────────────────┬─▶ OF ────▶──
                    └─( ─▶ machinespecificattributelist ▶ )─┘
```
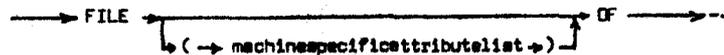
Examples:
    VAR
        INPUT (KIND=READER) OF PACKED ARRAY [0:79] OF CHAR;
        CODE (KIND=PACK,TITLE=EXECUTABLE/CODE/TEST,UNITS=WORDS,
               MAXRECSIZE=30,BLOCKSIZE=300,MYUSE=OUT)
               OF SECTORRECORDTYPE;

The attribute list probably has to be machine-specific at the present state
of operating systems. The declared size of the record of the file is
checked for compatibility with any specified attribute.

All B6700 files are automatically allowed to be accessed sequentially
or randomly, so this question does not arise specifically. Environment
enquiries and attribute changes can be done via calls on the operating system.

/

STANDARDS

Adherence to the PASCAL standard, interpreting this to mean the language
defined in the Pascal Report, and the axiomatic definitions thereof, must
be a very high priority of any PASCAL compiler writer/maintainer. There
are nevertheless several sticky problems which face any person in these
categories. Let me expose them.

  1. There is the question of whether to implement a strict PASCAL, or to
     extend it with various features. Of course there are some areas
     where PASCAL must be extended (see later), but every extension provides
     a user temptation and reduces portability of the resulting programs.
     This works towards implementing PASCAL as she is defined, and that
     alone.

  2. There are the problems associated with undefined parts of PASCAL; for
     example the elaboration of a CASE where the case expression evaluates
     to a value not matched by any label. A compiler writer has to do
     something, and these flaws or loopholes in the definition are left to
     individual discretion.

  3. There are places in PASCAL where the language is seriously deficient;
     primarily in treatment of files and i/o. Individuality here is necessary
     but can be seen to be clearly tending towards the Algol and BASIC
     messes.

  4. There are places in the PASCAL definition where the antecedents of
     the present state show through, in an unwarranted manner. Examples
     of these are (1) the CDC influence in the curious PROGRAM construct
     (often unnecessary, and deriving from CDC FORTRAN), the use of .. or :
     from Algol in array declarations (when TO is more explicit and less
     obscure), and the insistence on FORTRAN's archaic control character
     at the start of a printed line!

Objectively, or as far as I am capable of it, it seems to me that PASCAL
has in fact been frozen too soon, before the defects have had a proper
chance to be eradicated. It is therefore of prime importance that
some procedure be adopted whereby evolutionary change in the language
can be controlled, otherwise proliferation of dialects is inevitable.
Some of the afterthoughts should be recognized as such, and removed from the
province of the standard defining document.

(*Received 10/31/76*)

OPEN FORUM FOR MEMBERS

(SHORT, INFORMAL CORRESPONDENCE)

LEHIGH UNIVERSITY
BETHLEHEM, PENNSYLVANIA 18015

DEPARTMENT OF MATHEMATICS
CHRISTMAS-SAUCON HALL #14

July 28, 1976

Mr. Andy Mickel
PASCAL Users Group
UCC: 227 Exp. Engr.
University of Minnesota
Minneapolis, MN 55455

Dear Andy:

What happened to that new PASCAL release announced
last March?

One of my students has just completed a CAI system
in PASCAL. It features a lesson creation language which
has excellent expressive ability (it is possible to create
structured CAI lessons with it). The compiler for this
language is quite efficient. Its output code is interpreted
by a small (12.5K$_8$ words) PASCAL program. The interpreter
features good run time debugging aids which in many ways
resemble PASCAL's PMD.

In addition to the lesson compiler and interpreter,
there is an object lesson decoder and student monitoring
facility. The student monitoring routines record and
report individual student scores. Lessons can also be
set up which administer tests. The monitoring facility
reports a trace of each student's lesson sessions question
by question. The system keeps track (on a permanent file)
of each student's status and can be used to sequence
students thru a series of lessons and tests.

In total, the system is the most versatile and
efficient CAI system I have ever seen. To a great extent
the viability of the system can be attributed to the fact
that it was built with PASCAL. Should we write up the
system for the Newsletter?

Sincerely,

Richard J. Cichelli

(*Note: The student mentioned above is PUG member David Englander.*)

IN REPLY PLEASE QUOTE:

TELEPHONE: 692 3491
692 1122
EXT 3491

UNIVERSITY COMPUTING CENTRE
THE UNIVERSITY OF SYDNEY
NSW 2006   30th July, 1976

Pascal User's Group C/- Andy Mickel,
University Computer Centre: 227 Exp Engr.
University of Minnesota,
Minneapolis, MN 55455
U.S.A.

Dear Andy,

Please accept an application for membership of the Pascal
User's Group in my name. A cheque for US$4 is enclosed.

Our Computing Centre provides a service to the University
and to some outside users. For the past 18 months we have
been developing some of our applications programs in PASCAL.
We have found a significant improvement in the rate of
production of reliable programs. It is, of course, a
pleasure to write in.

The PASCAL compiler here is maintained by the Basser
Department of Computer Science who make extensive use of it
for undergraduate teaching and research computing. As
we have to continue to provide software products across
hardware changes, we are interested in the transfer of
PASCAL to new machines and in program writing tools. No
doubt your Center has the same concern. As we have only had
our CYBER 72-26 for two years it will be some time before
we may have to face the problem.

Yours sincerely,

Brian G. Rowswell

encl.

2512 San Gabriel St.
Austin, TX 78705
17 Aug 1976

Andy Mickel
Univ Computation Center
Univ of Minnesota

Dear Dr. Mickel,

I would like to join the PASCAL users group, and receive your newsletter.

I am a doctoral candidate in anthropology and my uses of PASCAL are for organization and analysis of data on language acquisition and on cognitive variation. Since programming for anthropological applications usually involves handling odball data (when the study is not a simple statistical one), the ability to strucutre that data has a rather liberating effect on one's programming!

However, I have found the lack of formatted read capability rather annoying at times, and I wonder if other users feel the same way. It would certainly not be difficult to add this capability to the compiler in such a way that it would provide upward compatability with the Jensen and Wirth (1974) standard, for example:

READLN(f, x1:w1, x2:w2, x3:w3);

where x_n is a variable of type integer, real, or packed array of char. For blank fields, integer and real variables are [illegible] the value [illegible] variables [illegible] to [illegible] but the end of line should be assigned here (or blank in the case of strings) or there would be transportability problems between systems which truncate trailing blanks and those which do not. Real numbers such as .05, -.3 should not cause RDR to halt the program (in formatted or freefield reads)! Honestly -- you'd think ETH never writes programs which have to read the output of Fortran programs (i.e. statistical packages).

I'm looking forward to seeing my first PASCAL users group newsletter -- tell me if there are any dues or anything.

Sincerely,

Willett Kempton

Dr. Andrew Mickel, Editor
University Computer Center
227 Experimental Engineering Bldg.
University of Minnesota
Minneapolis, MN 55455

Dear Andy:

As we discussed on the phone, enclosed is an item which we would like to have included in the next issue of the PASCAL Newsletter.

Also, enclosed is a copy of the actual prettyprinting program and documentation, which is for your information.

We have looked at the prettyprinting documentation from Lehigh and it is not at all along the lines of our prettyprinting program. Let us keep in touch.

Sincerely,

Henry F. Ledgard
Associate Professor

HFL:lms

Enclosure

OPEN FORUM FOR MEMBERS

(SHORT, INFORMAL CORRESPONDENCE)

CiS

WEST TEXAS STATE UNIVERSITY
SCHOOL OF BUSINESS CANYON, TEXAS 79016
DEPARTMENT OF COMPUTER INFORMATION SYSTEMS

September 13, 1976

PASCAL User's Group
C/O Andy Michel
UCC: 227 Exp. Engr.
University of Minnesota
Minneapolis, MN 55455

Andy:

Some time ago you sent to all PUG members a set of corrections for
the PASCAL User Manual and Report, Second Edition. In the process
of moving from Drake to West Texas State I have buried my copy in
stacks of yet unpacked papers. Would you please send me another
copy. I would appreciate such.

We are in the process of getting PASCAL up and running on our DEC-10.
I am working without letup on the process of converting to PASCAL
some rather open-minded colleagues. I will use PASCAL as the vehicle
language in the data structures course this spring. (Is there any
other way to do it?)

Do you know if anyone has yet to build a really interactive version of
PASCAL?

When does the first edition of the PASCAL Newsletter come out? I am
looking forward to receiving it.

Sincerely,

H. P. "Duke" Haiduk
Assistant Professor
Computer Information Systems

UNIVERSITÉ DE NICE

LABORATOIRE D'INFORMATIQUE
PARC VALROSE
06034 NICE CEDEX
TÉL. XXXXXXX

Nice, le 16 th SEPTEMBER 1976

Pascal user's Group
c/o Andy Mickel
University Computer Center
227 Exp. Engr.
University of Minnesota
Minneapolis, MN 55455
U.S.A.

B : Andy :

I read with much interest the first newsletter made under your responsibility,
and i was very impressed of the vast amount of useful informations you
managed to pack in it. One problem afraid me : at £/ 1.31 for postage costs,
my fee ifor the group will not even cover them for one year. If you have a
non-negligible number of overseas correspondents, this could be a problem.
Would it not be good to have a European re-distributor, who could make as
much copies of the newsletter as necessary and send them throughout our
whole old continent ? He could also collect fees and keep the part needed
for his own costs. This would be an extension of the suggestion made by
Judy Mullins of Southampton. I am not suggesting that I could do this work,
since we have no really good copying facilities in Nice, but maybe somebody
else could be contacted in a wealthier University. I think that Pascal is a
language which is equally popular on both sides of the Atlantic, and that
it would be a pity not to take advantage of this exceptional situation.
The rest of the present letter is devoted to some remarks, comments and
precisions urged on by the reading of the newsletter. Some of them may be
of interest for other Pascalers and merit publication in the newsletter,
but I think it would be better that you do selection and morever a rewriting,
since I know my English has much deteriorated since my departure from Canada.

Pascal User's Group session at IFIP ' 77 : in the same spirit as
the remark before, I think it would be very interesting to have a world
meeting after the many national meetings in USA, England, France (more
details below), Switzerland, etc. It should be possible to arrange something
with the local organizers in Toronto, or through a TC 2 member. Since IFIP
is a very formal organization, it should be useful to search for some
arrangement as soon as possible.

        Standard Pascal book by Bill Atwood : I am somewhat afraid by
this title, and I will write separately to Bill. What is exactly
Standard Pascal, I do'nt know, and I think it should be very dangerous
that anybody (but Wirth) could say that his own idea of the language
is the standard, without any discussion with the community of Pascalers.
Many people have different ideas on the subject, and I say more on the
subject below.
        News about Pascal in France and other French-speaking parts
of Europe (Pierre Desjardins could say much more than me about the
French-speaking part of America) : Pascal is used in some important
or smaller Universities as a vehicle for teaching programming and
writing software, especially in Paris (Institut de Programmation),
Toulouse (Université Paul Sabatier), Bruxelles (Université Libre),
Lausanne (Ecole Polytechnique Fédérale), Montpellier, Nice, Neuchâtel,
etc. Some important Universities (Grenoble, Rennes) do not use it
because they have made high investments in either Algol W or Algol 68,
which are much better tools than Fortran and therefore stronger
obstacles to the shift to Pascal. Implementations of Pascal have been
made in Grenoble on the IBM 360, Paris on the CII Iris 80 and 10070,
Neuchâtel on the IBM 1130, and one is being made in Nice on the CII
Iris 80. More details follow in the "implementation" part of the
present letter. A two-days meeting about Pascal took place in Nice one
year ago. Sixty persons from France, Belgium and Switzerland attended.
You will receive by separate mail the text of the majority of the papers
which were presented. Additionally, there were panels about the use
of Pascal in teaching, its implementation and its changes or extensions.
I plan to organize a similar meeting in May or June 77.
        About the paper by Richard Cichelli : I think it would be a
very useful policy to request of people sending Pascal programs to
write them in the publication language and not in any particular hard-
ware or implementation language. By "publication language", I mean the
form used in both Wirth's books and in the Pascal reports from Zurich :
free use of lower-case letters, underlined keywords, use of every
simple and aesthetic available characters, such as { and } for comments,
≤, ≥, ≠ and so on. By "hardware or implementation language", I mean the
form used in Jensen & Wirth's book (because of limitations on the
character set) and on every implementation : generally only upper-case

3

letters, (* and *) for comments, <= , >= , <> and so on. I think the
publication language is the only truly readable and aesthetic form, and
that every implementation should be free to give its own interpretation
of the characters which are not available on its particular hardware,
provided it conforms to the general rules stated by Wirth himself. In
fact, any implementation language which can be translated into another
one by means of a one-page Pascal program should be acceptable, and this
includes national variants for key-words. In Cichelli's paper, examples
in the text generally conform to the publication language, comments
excepted, but figure 1 does not underlines keywords and uses a character
set not particulary readable, and the complete program seems to me a good
example of what should be never done : only upper-case letters, no
underlined keywords,   for strings, /* and */ for comments as well as r*
and ' , and so on. I know well that it is very difficult to have any
secretary to type correct program texts, and that is probably the reason
why Jensen & Wirth's book was typed by a computer, but it can be done
and I think it is worth the trouble. Of course, these remarks are not
criticisms about Cichelli's paper, which I find very interesting and
useful.
        About the paper by Timothy M. Bonham :, I agree about point 1.
About point 2, I must recognize that to critize Habermann about the use
of "..." instead of ".." was a petty point and probably a criticism of the
typesetter, but I think also that proof-reading exists for removing such
errors. The symbol "..." itself would cause problems in some scanners,
since it would be the only three-characters delimiter in the language.
I have a more important criticism about the CDC 6000 compiler, which
considers "¬" and "↑" as equivalent, for some historical and obscure
reason. That is probably the main obstacle to a very simple and natural
extension of the case statement, viz. the use of a subrange as a case
label, by similary with the notation for sets.
        About point 3, I think that this long discussion should not
be necessary if the distinction between publication and implementation
language was clearly done : if you have the left arrow and like it,
use it; but it is much more uncommon than you think, and there are much
more Algol users than APL users on the Eastern side of the Atlantic.
        About point 4, I agree , more especially as the French version
of Pascal keywords uses bas and haut (up and down) as translations for
to and downto. About point 5, I disagree, because three different syntaxes

for comments seem really too much. As it is, the current syntax has
more advantages than disadvantages. Once more, however, I think that
every particular lexical convention which may be translated into the
standard one (if "standard" is really defined) by a one-page Pascal
program should be acceptable (but not for publication!).

Pascal bibliography : my translation in French of Wirth's
first book (Systematic programming) is at last in hands of the
publisher (Masson). I expect the book to be released during 1977.

Implementation notes : Although CII 10070 is a nickname
for Xerox Sigma 7, the CII Iris 80 is another machine, more precisely
an extension of the first one. Moreover, the CII operating system is
different from Xerox and transporting a Pascal compiler from a Xerox
Sigma 7 to a CII Iris 80 probably would not be a trivial job.
A Pascal compiler for both CII machines has been written by Messrs.
Thibault and Mancel of IRIA (Research institute in Informatics and
Automatics, a French government agency), by bootstrapping the first
CDC 6000 Pascal compiler. It has now been upgraded to accept Standard
Pascal and to allow separate compilation, and it is officially distri-
buted by IRIA, a case which seems unique. Its overall performance seems
to be quite good, and it is used in French Universities which have one
of these machines.

The CII Iris 50 is a completely different machine, much
smaller, and we have some trouble in Nice when trying to implement
Pascal. Pascal-P presently works interpretatively, but it is unusable
for programs larger than one page, and consequently it cannot be used
as a tool for bootstrapping a true compiler. I plan to write a brief
paper for describing the bootstrap method which will be used, and which
seems to be a unique one. Maybe it could be done in time to be included
in newsletter number 6.

A Pascal compiler for the IBM 360, which was probably the
first one, has been done in one of the Universities of Grenoble.
Unfortunately, the people who made it had no time nor support for
distributing it, although it seems to have impressive performances in
execution time (but less good in storage needed for compilation).
People to contact are Messrs. Henneron and Tassart (Informatique &
Mathématiques Appliquées, B.P. 53, 38041 Grenoble-Cedex, France).

Implementations for Pascal-P, Pascal-S and finally full

Pascal have been done for the IBM 1130 and are in use at the
University of Neuchâtel (Centre de Calcul, Chantemerle 20, Ch-2000
Neuchâtel, Switzerland).

A complete and standard compiler for the Xerox Sigma 6,7
and 9 has been done by Pierre Desjardins, who can give you all
desirable information. Anyway, it seems to be a very good implementa-
tion, especially in the domain of compatibility and conformity with
the standard.

I hope that some of these informations will be of interest
to you, and that my poor English will not be a hindrance. If you
managed to read this long letter in its entirety, thank you for your
long-suffering. I look forward to any news.

Sincerely yours,

O. LECARME

UNIVAC
COMPUTER SYSTEMS

UNIVAC PARK, P.O. BOX 3525
ST. PAUL, MINNESOTA 55165
TELEPHONE (612) 645-8511

September 17, 1976

Andy Mickel
University Computer Center
University of Minnesota
227 Experimental Engineering Building
Minneapolis, Minnesota 55455

Dear Andy,

In response to John Eisenberg's article 'In Defense of Formatted Input'. I would like to make the following remarks:

1. Formatted I/O statements are usually wrapped up in a package of confusing notations which detract from the readability of a program.

2. It is not clear that a system routine which does ord(ch) - ord('0') would be any better than the user's own routine and in addition, it is unlikely to respond in a flexible manner to exceptional numbers (e.g. beyond machine precision).

3. Formatted I/O still does not solve the general problem of number to string and string to number conversions.

The University of Illinois PASCAL compiler has a rather elegant solution to this topic. This implementation of PASCAL allows the user to 'read' or 'write' numbers or strings to and from arrays as well as files.

Sincerely yours,

Robert E. Novak

Robert E. Novak

INDIANA UNIVERSITY
Research Computing Center
Wrubel Computing Center
BLOOMINGTON, INDIANA 47401

TEL. NO. 812—337-1911

September 22, 1976

PASCAL User's Group
c/o Andy Mickel
University Computer Center
227 Exp. Engr.
University of Minnesota
Minneapolis, Minnesota 55455

Dear Andy:

Since your visit to Indiana University last Spring and with some prodding from Al Towell, I've become "hooked" on PASCAL! With my administrative responsibilities in the center I have not been able to spend the time on the language that I would like (i.e. I'm not an "expert" yet); nevertheless, it is clear to me that the language far exceeds (elegance, read-ability, flexibility, etc.) anything else that is generally available. I am gratified to see the formation of PUG (enclosed is a $4.00 check for membership); if ever users (without manu-facturers' interference) have had an opportunity to do a great service to the computing community, this is it...we must not let the language get away from us by allowing local extensions to creep into widespread existence without a proper review proced-ure (Eg. a PUG Standards Committee...we have a "jewel" here that needs protection). I also support Hellmut Golde's belief that the widespread acceptance of PASCAL is dependent on the ability to mix Fortran and PASCAL main- and sub-programs; Fortran (an historical accident) can only be corrected if programmers are able to "grow" to PASCAL gradually.

As my expertise in the language develops I hope to con-tribute to PUG's primary roles...to maintain the integrity and promote the acceptance of PASCAL.

Sincerely,

Stephen W. Young
Director

SWY/pce

P.S. I know of at least two PASCAL users; hence, "PASCAL User's Group" should become "PASCAL Users' Group!"

PROFESSOR OF COMPUTER SCIENCE
T. KILBURN, C.B.E., M.A., Ph.D.,
O.Sc., F.I.E.E., F.B.C.S., F.R.S.
ICL PROFESSOR OF COMPUTER ENGINEERING
O. B. G. EDWARDS, M.Sc., Ph.D., M.I.E.E.
PROFESSOR OF COMPUTING SCIENCE
F. H. SUMNER, Ph.D., F.B.C.S.
PROFESSOR OF COMPUTER PROGRAMMING
D. MORRIS, Ph.D.

DEPARTMENT OF COMPUTER SCIENCE
THE UNIVERSITY
MANCHESTER
M13 9PL

Telephone: 061-273 5466

29th September 1976

Mr. Andy Mickel
University Computer Center
227 Experimental Engineering Building
Minneapolis
Minnesota 55455
Minnesota 55455
U.S.A.

Dear Andy,

Thanks for the letter and detailed information that you sent. In an
attempt to get this to you by 1st October I have only included the
following:

1. Documentation relating to Pascal at UMRCC

2. Details of our CDC 7600 implementation

3. A short note on our experiences with Pascal under NOS
   on the CYBER 72

4. A copy of the updates to the compiler mentioned in item 3.
   Feel free to use, distribute or destroy any of these mods

5. Details of a possible bug in the Zurich compiler. Lack of time
   prevents me from finding the cause, so I will only send you the
   evidence this time.

I would also like to advertise the fact that I wish to see formed within PUG
a Pascal Standard's Group, which I am willing to organise unless a more
suitable candidate volunteers. If I am not overtaken by events I will
contribute a brief outline for Newsletter #7 of how the standards group
might operate.

Yours sincerely,

Tony Addyman

P.S. This will be late - I've been ill.

LEHIGH UNIVERSITY
BETHLEHEM, PENNSYLVANIA 18015

DEPARTMENT OF MATHEMATICS
CHRISTMAS-SAUCON HALL #14

October 9, 1976

Mr. Andy Mickel
PASCAL Users Group
UCC: 227 Exp. Engr.
University of Minnesota
Minneapolis, MN 55455

Dear Andy:

The first PUG Newsletter was really well done - keep up
the good work.

I was sorry to read that Dr. Waite initially declined to
join because of the dues cost. Since I suggested the membership
rate (on the principle that "there ain't no free lunch") and I
regard Dr. Waite as having potentially great positive influence
on PASCAL development, I hereby contribute his dues. I hope Dr.
Waite will use the Newsletter as the two-way communications
channel it was meant to be.

I enclose an article from the British Computer Society
Bulletin. The article, entitled "Which Language?", shows that
within the next five years PASCAL could become the language of
choice for university computer science programs.

I also enclose an article which addresses a mishmash of
topics. In it are presented some frankly half-baked ideas. I
hope the membership can cook them down (by stepwise refinement).

Sincerely yours,

Richard J. Cichelli

...

University of Illinois at Urbana-Champaign

COLLEGE OF COMMERCE AND BUSINESS ADMINISTRATION
DEPARTMENT OF BUSINESS ADMINISTRATION · 350 COMMERCE BUILDING (WEST) · URBANA, ILLINOIS 61801 · (217) 333-4241

October 11, 1976

Andy Mickel
University of Minnesota
University Computer Center
227 Experimental Engineering Bldg.
Minneapolis, Minn. 55455

Dear Andy:

I have some further comments on PASCAL, based on my experience as the local
implementor of the Hamburg DEC-10 version. First, on the compiler itself.
My comments, as published in the Newsletter (No.4) were possibly a bit too
harsh: They did not make it clear that my objections were to the interface
between the compiler and the system, not to the reliability of the compiler
or the quality of the code (aside from one bug in parameter passing - my
blanket attack on procedure linkage seems to be more applicable to the older
PASCAL compiler, rather than PASREL, the one we are now using). However the
main thing I have to say is that I am unable to report on the usage of PASCAL.
As far as I know there isn't any, except a few computer operators who use it
to do homework that was supposed to be done on the Computer Science Depart-
ment's PDP-11 PASCAL system. I thought you might find my analysis of this
situation useful.

Our lab is entirely a research organization. Much of our computer programming
is "peculiar" in one way or another, and PASCAL turns out not to be very use-
ful for it. First of all, we are often doing unusual input/output operations:
controlling a robot, or doing random access work. PASCAL can hardly be blamed
for not having the facilities to handle real-time device control. We have
even had to modify the operating system slightly for that. But it does not
support random access file manipulation, or for that matter any non-buffered
kind of I/O.

Second, we do a considerable amount of work in artificial intelligence. PASCAL
certainly has the ability to build complex data structures, but these abilities
are rather low-level compared to LISP, or even SAIL. To do what we do with
LISP in PASCAL, one would apparently have to write a memory-management system,
either garbage-collecting or reference counting, and then build up a collection
of basic procedures to manipulate the structures. I.e., one would nearly have
to rewrite LISP in PASCAL. More about this below, under runtime memory
management.

Finally, we do a good bit of what might be called "system hacking." This
includes writing system programs such as a mail system, various programs for
displaying information about system status and parameters, for analyzing dumps
of monitor crashes, editors, etc. All of these programs require us to make
many obscure monitor calls and to transform data between the form used in
monitor calls and usable external forms. PASCAL provides no facilities for
doing monitor calls (since these are by definition machine - dependent), nor
does it supply the large library of conversion procedures that SAIL, for
example, does (SIXBIT to ASCII, etc.).

More generally, the lack of string processing facilities makes many tasks
rather inconvenient. This really falls into the same category as my complaint
that PASCAL is not LISP, since we are again talking about a lack of run-time
memory, management. String processing appears to require this as much as list
processing. By run-time memory management I mean the ability to do NEW
repeatedly, without eventually running out of space. This seems to require
garbage collection, reference counts, or some such scheme, as well as an
ability to get more memory from the operating system when it is needed. I
realize that this is a controversial issue. If PASCAL is intended solely as
an implementation language, it should not supply a built-in memory-management
scheme since that is one thing that an implementor will want to design for his
scheme since that is one thing that an implementor will want to design for him-
self. But I believe it is unreasonable to expect applications programmers to
include a garbage collector and/or memory allocator in every program. I find
it hard to believe that any serious use can be made of NEW without some memory

management. Indeed the DEC-10 PASCAL compiler uses extensions to PASCAL to
keep its symbol table within bounds. Possibly built-in garbage collection
should be available only when specifically requested. Then implementers who
wish to design their own memory management would be free to do so.

Also, no attempt seems to have been made to facilitate decoding anything other
than numerical input items. I sat down to write a program that would read a
program name from the terminal and then run it. I knew that I would have to
do the running by a call to an assembly-language subroutine. But when writing
the filename scanner began to look like writing the syntax analyzer for a
small compiler, I gave up and used SAIL. In fact, this was my last attempt
to use PASCAL. (Filenames on the DEC-10 are not trivial: A full-blown one
might be DSKB:PGM.EXE[5,731,SAIL,SRC]. Almost any part can be left out, and
gets a default value. Also, the bracketed item can come first).

I think the real issue is what kinds of problems PASCAL is intended to attack.
It is unreasonable to expect any general-purpose language to have the peculiar
capabilities of LISP. That one could write a LISP interpreter in PASCAL is
possibly all that should be asked. The inability to handle messy I/O and
"system hacks" seems more serious, though. If we give up on that it seems to
limit PASCAL to compiler writing and a replacement for FORTRAN. (Its short-
comings as a replacement for COBOL have been noted elsewhere.) The best
language for such things on the DEC-10 is SAIL, a Stanford University
extended ALGOL. It makes no pretense at machine-independence. There is a
syntax for doing monitor calls directly. Any I/O mode available in the
monitor may be explicitly specified (and is handled automatically, so that
buffered and unbuffered I/O can be done with the same higher-level language
constructs). A huge library of special purpose procedures is provided,
including conversion procedures that allow one to make sense of data in
funny monitor formats. If all else fails, one may insert sections of
assembly language in the middle of a SAIL program. Accumulators are freed
for your use, and constructs are defined to let you refer to the address of
array elements, record elements, etc. in higher-level terms. SAIL also has
several data types that depend upon runtime memory-management: e.g., strings,
lists, and records. (I believe these three classes are separately garbage
collected.)

SAIL is certainly not the ideal programming language, especially when judged
by the "structured programming" and machine-independence ideals that guided
the design of PASCAL. But I find it hard to imagine myself adopting a language
for day-to-day use that did not have most of its features: the ability to
use all of the facilities available in the operating system, run-time
memory-management, and a good library of special-purpose procedures. Unless
PASCAL can find a way to incorporate some of this in a reasonably structured
way, I think it will not get beyond introductory computer science courses.
Unfortunately, PL/I comes very close to meeting my goals (especially when
used with OS/360). I fear I may find myself teaching students PL/I when
I had hoped to use PASCAL.

                                        Sincerely yours,

                                        Charles L. Hedrick
                                        Assistant Professor of Business
                                        Administration

CLH/jh

Xerox Corporation
Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, California 94304
(415) 494-4000

Okt. 15, 1976

Mr. Andrew B. Mickel
Editor, Pascal Newsletter
Computer Center
University of Minnesota
Minneapolis, Minn. 55455

Dear Andy,

Upon reading the last issue of the *Pascal Newsletter* I was surprised to find that you do not distinguish between *private* letters and letters to the *Editor*. I strongly disagree with your elimination of this distinction, and fear that some of the writers of the published letters may resent your zeal for transparency.

I suggest that letters for publication must explicitly be addressed to the *Editor of the Pascal Newsletter* (as shown above), and that no other letters will be published.

Yours sincerely,

Niklaus Wirth

Prof. N. Wirth

CiS WEST TEXAS STATE UNIVERSITY
SCHOOL OF BUSINESS CANYON, TEXAS 79016
DEPARTMENT OF COMPUTER INFORMATION SYSTEMS

October 21, 1976

Mr. Andy Michel
227 Experimental Eng. Bldg.
University of Minnesota
Minneapolis, MN 55455

Andy:

I have just recently finished reading from cover to cover the Pascal Newsletter Number 5. You guys deserve a big commendation from all advocates of progress in programming languages especially advocates of PASCAL and its future development. With such a common forum as the Newsletter, perhaps we can interact in such a way as to encourage if not force, a "standard" series of improvements to PASCAL.

In one of the letters (I have searched several times for the specific one) there was mention of an implementation of Brinch Hansen's concurrent PASCAL. Could you please give me information concerning the availability of concurrent PASCAL for the DEC System - 10. That appears to me to be a desirable way to go with the operating systems course.

Keep up the good work. I will support you in spirit and continue to send in my monetary support for the Newsletter.

Sincerely,

H. P. "Duke" Haiduk

The University of Tasmania

Postal Address: Box 252C, G.P.O., Hobart, Tasmania, Australia 7001

Telephone: 23 0561. Cables 'Tasuni' Telex: 58150 UNTAS

INFORMATION SCIENCE DEPARTMENT.

IN REPLY PLEASE QUOTE:

FILE NO. ....

IF TELEPHONING OR CALLING

ASK FOR ......

22nd October, 1976.

Pascal User's Group,
C/- Andy Mickel,
University Computer Center, 227 Exp. Engr.
University of Minnesota,
MINNEAPOLIS, MN. 55455

Dear Mr. Mickel,

PASCAL NEWSLETTER

Judy Mullins, of the University of Southampton ICL 2900 project, has
I believe been sending you copies of the correspondence we are generating between
Tasmania and Southampton on the common problems of implementing PASCAL on highly
structured computers (B6700 and ICL2900) instead of the more usual monolithic
machines. I enclose therefore my reply to the last letter she sent you, in case
you want to include it in the Newsletter. It contains, I think, discussions
of several important issues.

Some time later I must write a view for the Newsletter specifically
on PASCAL development, for it is clearly going off the rails (clear to me anyway).
A great pity, and something should be done to remedy the residual problems in
definition and to encourage greater interchange of programs and program portability.
I shudder at all those PDP11 and IBM 370 implementations!

I enclose also a release on the status of the B6700 PASCAL
compiler, which I ask you to print. It should clarify the situation for any
interested B6700 users on a machine which is noted for the rarity of any new
compilers.

Yours sincerely,

A.H.J. SALE,
Professor of Information Science.

Encls.

Department of Mathematics

The University, Southampton, SO9 5NH.

Telex 47661

Tel 0703 559122 Ext 2387

The University of Southampton

COMPUTER STUDIES GROUP

October 4, 1976.

PASCAL Newsletter

Thank you for your long and most informative letter. It has raised
two points which are of interest to both our Universities as well as, I believe,
the Pascal community in general. These are

What brand of Pascal should be implemented?

and   How is it implemented on high-level machines?

We have had long discussions on these questions, both previously, and in the
light of your letter. Here follow our views.

What?   We believe, for better or for worse, in Standard Pascal, as laid down
in Jensen and Wirth. The 1900 compiler which we are bootstrapping is a full
implementation of the Standard with one exception: the Pascal 1 I/O routines
are kept (e.g. write(eol), while not (ch=eol) do read (ch) etc). Considering
the mess the Zurich compilers got into when Ammann tried to rationlize the
readln procedure for integers, we consider this a wise choice. However, Jim
Welsh is proposing to support readln, writeln in tandem with the other set, to
ease portability problems. To be realistic one must accept that there is no
such thing as a fully portable program, but if the changes are few and well
understood then an intelligent programmer will have a small, albeit irritating,
task to bring an alien program up on another machine. The syntactic changes in
your implementation are good ones except for the % comment which reminds me
too much of assembler, and the ELSE case label (more of that anon). But if you
allow all these goodies and students get to know of them, it merely widens the
communication gap both between the students and the excellent text books that
are beginning to appear on Pascal, and the students' programs and other machines.
Burroughs installations have for many years fed the Algol 60 communication gap,
and gaily ignored its consequences. Certainly, B 5700 Xalgol programs are more

readable and better in many respects than their equivalents on conventional
machines. But have their ideas been copied? Experience has shown that
people will stick to a standard unless it is quite intolerable. Fortran is
just inside the tolerable, BASIC (as defined at Dartmouth in 1971) is not.
Extensions boomed with the result that the Standardiation Committee cannot
hope to produce a useful document now. Pascal must have a more promising
future.

What should be done then? Firstly, we believe that Pascal is well within
the tolerable. It is a sparse language which gives it its main advantage – tight
and fast implementation. It is well-defined, well-documented and supported by
academic articles in readily available journals. It also represents a great
leap forward in the use of data structures and readable self-documenting
programs. Therefore we shall implement Pascal as in the Report, and that alone
is pretty good. Secondly, there are holes that cannot be ignored and two of
the most pressing are file I/O and diagnostic aids. We are taking the line that
both of these should encroach as little as possible on the Pascal source. What
additional syntax will be needed for files on the 2970 will only be considered
in about January and I shall keep you informed. If you could send me full
details of your I/O interface, it will be around at the vital moment and we can
see if the 2970 version can be made to conform at all.

As regards diagnostics, we are lucky here. Glasgow have implemented a
diagnostic package for symbol table dumps, profiles and tracing on the 1900
compiler and will be putting it onto the 2900 as soon as our compiler is ready.
(They are getting a 2980 eventually). The system is simple and induces minimal
overheads at execution time. A few parasitic procedures write information to
disc and in the event of a failure, the post mortem program (written in Pascal)
comes in to dissect the corpse and produce its report. All very clinical! I
enclose a copy of Glasgow's documentation. In conjunction with David Watt and
Bill Findlay we are going to change the user interface from the horrible
pragmats to more recognisable directives (proposals attached). Global options
will come before the first Pascal statement (be it CONST, PROGRAM OR PROCEDURE)
e.g.

        OPTION LIST = FALSE;

            RETROTRACE = 500;

with suitable defaults, local options come in as comments e.g.

        (*$ LISTOFF, TRACEOFF *)

        One further hole is that of separate and mixed language compilation.
There is a strong feeling to keep to the Burroughs idea of all input in Source.
Could you send me details of the B6700 INCLUDE options as we shall no doubt have
to write our own software to do this? By the way, what do you think the
meaning and implications of assigning one file to another are? If one could do
this in a sensible way, then each INCLUDEd file will be activated in the
compiler by a single toggle? perhaps. The decision for mixed language
programming is a sticky one, but necessary because of the tremendous library
resources available in Fortran. We hope to be able to provide this and the
INCLUDE option. For once, the 2900 design is helpful: the compiler output format
proposed for March 1977 and thereafter, Object Module format, can be input to
the collector or the loader.

I hope you don't mind if I send a copy of this letter to Andy Mickel for the
P.U.G. newsletter as I had been meaning to write to him on the same topic.
On the phone about distributing newsletters, he reiterated his concern about
the health of Pascal and the extent to which it is diversifying. No one is
quite sure what to do, except worry, but I agree that a committee is not the
answer.

I haven't broached the second question (i.e. How?) but that will have to
wait for another letter. Before I close, two quick comments.

ELSE case label: I had the opportunity to discuss this with Wirth and his
objection is a very valid one i.e. the programmer will put the ELSE there to
catch values which he expects, but wants to treat in such-and-such a way. What
will happen is that it will also catch those inevitable values that "can't
possibly occur" and the program will produce wrong results, when it should have
halted in error.

SYNTACTIC SUGAR: Your syntactic options (TO instead of.., OF instead of:)
become almost justifiable if you provide a macro processor, written in Standard
Pascal which will convert any B6700 program to the Standard. This will put
some rein on the extent of the changes, and can be given to any serious
programmer who leaves the B6700.

                        Yours sincerely,

                        Judy Mullins.

Professor A.H.J. Sale,
Information Science Department,
University of Tasmania,
Box 252C, G.P.O.,
Hobart,
TASMANIA,
Australia 7001.

cc. Andy Mickel

Enc. diagnostic system document
    OPTION syntax proposal.

JM/bb

The University of Tasmania

Postal Address: Box 252C, G.P.O., Hobart, Tasmania, Australia 7001

Telephone: 23 0561. Cables 'Tasuni' Telex: 58150 UNTAS

INFORMATION SCIENCE DEPARTMENT.

IN REPLY PLEASE QUOTE:

FILE No.
FILE No.

IF TELEPHONING OR CALLING
ASK FOR

22nd October, 1976.

Miss J. Mullins,
Department of Mathematics,
The University of Southampton,
SOUTHAMPTON. SO9 5NH
UNITED KINGDOM.

Dear Miss Mullins,

Thank you for your letter. Let me try to answer your specific
points; some of which I agree with, and some of which horrify me. I have drafted
part of this letter as a person-person communication, and part as open comments
on aspects of the PASCAL development cycle.

Standards.

Of course I agree that standard PASCAL must be adhered to, but not to the absolute
exclusion of all deviation. A few features of PASCAL ought not to have been
standardized (incontrovertibly in my opinion), and if absolute adherence is demanded
I fear for the future fate of the language. Consider that programmers do not in
general change languages unless they perceive advantages in several areas. A
delightful language embedded in a lousy implementation, or with poor facilities,
may not displace a poor language which has evolved into a good support situation.
PASCAL vs FORTRAN for example.....

I cannot agree with your comments on the likliness of adherence to standards.
Standards are maintained as limits by programmers for reasons which are quite
independent of the tolerability or otherwise of the language. Indeed, I think
that PASCAL is quite as much under pressure from burgeoning differences as was
BASIC, and for very similar reasons. It should not have escaped your notice that
adherence to the limits of the FORTRAN standard is not the common practice of
FORTRAN programmers, not yet of the supplier's compiler-writers. Where it is
adhered to, it arises out of the need to write portable software. This is the
prime determinant of adherence to standards by programmers; a certification process
is of some effect with compiler-writers.

Frankly, I think that insufficient thought has been given to ensuring a healthy
future development of PASCAL, just as insufficient thought was given to the problems
of its portability. The very sparseness of the compilers are an encouragement to
diversity, with all the effects one can witness in the PDP-11 and IBM 370
implementations, not the reverse. I have given a lot of thought to the ecology of
languages, and perhaps I can best give notice of a diatribe on suitable protection
of PASCAL's niche.

Character codes

Ouch! If you are honest about standards you should realize that inventing any new
character set (even if it be ASCII offset by 64) is just plain stupid. No-one
outside Europe would contemplate it, even if ASCII and EBCDIC are not all one could
wish. Neither is your alternative, of course. If I were involved in your project
I'd confabble with a huge, red pen through that proposal, and tell you
to go and look at the standard. For instance I'm a little bit startled
that admits that the alphabetics might not form a compact set (as they do not in
EBCDIC), nor an order between space/digits/lowercase/uppercase. Why is it so
important to you, anyway?

I hope this will make you not quite so pleased about PICS. The set constructs you
quote should be possible given any large enough set implementation, but if one has
to be aware of some structure of the language's character set, I see no reason
why you need to have a new one of your own. Try

    IF CH IN ["A" TO "Z", "0" TO "9", "a" TO "z"] THEN .....

    IF CH IN ["A" TO "I", "J" TO "R", "S" TO "Z"] THEN .....

Frankly, I cannot either see why you think all ASCII's low 64 control codes and
lower-case are so unimportant. Perhaps you are not thinking of an interactive
environment; just of a number-crunch set of users with batch?

Did you realize too that if PICS is available and the default, you will inevitably
have programmers using CHR and ORD writing programs that are non-portable for
quite mysterious reasons?

If you can convince me that you can supply a program (written in Pascal of course)
that will accept Pascal program source text written to use PICS, and will produce
equivalent Pascal program source text that does likewise, I might almost begin to
think the innovation just acceptable.

B6700 & comment

Yet, it is reminescent of assembler isnt it? But not all assembly language
features (nor yet FORTRAN) are bad. Although PASCAL's comment facility is streets
ahead of Algol's, BASIC's or COBOL's, it still has a few defects, such as a
propensity for swallowing text without trace. This is a minor addition (experience
shows that this sort of comment is preferred by programmers than having to close
one off) to keep B6700 compatibility, and to entice B6700 programmers to transfer.
A few lollies help now and then will do wonders.

I have in mind too prime requirements for extensions: an extension should fall
into one of two classes. It can be removed by a simple context-free program to
produce a standard-compatible version, or the facility provided is quite unavailable
in the standard. This falls into the first class.

ELSE in CASE

I have heard the argument you attribute to Wirth before, but I cannot give it
much force. In fact, PASCAL leaves undefined the action of a CASE where the
expression evaluates to a value not matched by a case label. Consider the
situation (I nearly wrote "case") where

    (i)   the value is in-range of the type, but not represented, and

    (ii) the value is out-of-range of the type.

It is arguable that in the first case the effect should be "do-nothing" on the analogy of IF-THEN; or it should be a run-time error on special arguments. I personally incline to the second (as Wirth), but the first is far from indefensible. Only in the second case should a run-time error always be caused; and this may be for reasons quite independent of the structure of the CASE, but simply because of the type involved. There are a few nasty spots with semi-infinite types (integer)...

I remain unrepentant! ELSE in CASE is a feature which mirrors the way programmers' think and offers ways of epxressing things that are unbearably cumbersome without it (try a CASE on char in a lexical analyser), though I will concede that some poor programmers can misuse it. But this is true of other features of PASCAL, for example in the REPEAT where the relation is expressed as an "escape" condition instead of a "continue-looping" condition, encouraging widening the liklihood that infinite loops are created.

As a further argument I will ask you to put yourself in the position of a compiler-writer writing a compiler which you know will be used on the other side of the world from you. You use hundreds of CASEs. All of them should be proof against flaws and bad input. Your end-users will curse you (and not sotto voce) every time the compiler crashes because of an out-of-range CASE. Are you then willing to forego ELSE in favour of laborious and error-prone lists of alternatives not expected? Robustness is as much a virtue as correct behaviour with expected input....

Mixed-language

You put your finger on a severe and very important spot. See my longer comments. The B6700 problems revolve around the complex structure of the code-file, and the complex actions taken by the BINDER to reorganize the outer-block stack for OWN objects, and the segment dictionary for VALUE objects. Tie this up with stack cleanup activities, files, and binding external objects (including files) by name into externally compiled procedures, and checking parameter compatibility at bind-time, and you may realize that the complexity of a codefile with BINDINFO is an order of magnitude larger than an executable codefile.

Files

Yuk! I detest your assignment idea of files. See expanded comments. Distinguish between compile-time actions (an INCLUDE) and run-time actions (an :=); and keep the := for total change of the entity involved. I suspect in fact that files might have been better out of the VAR part, and into an environment specification, which would have been more accurate. Who knows; it's too late?

Diagnostics

No real comment; all seem good ideas, though my Burroughs experience leads me to suspect that I would spend more time fighting the operating system than anything else. I have in mind also allowing a facility so that interactive users can browse through the saved state making enquiries of variables, etc. Dumps of any sort are sledgehammers, where screwdrivers will often do.

Compiler options

Unless you are absolutely stuck on every detail of the CDC PASCAL implementation, you will not implement compiler options in the same way. The mechanism for specification is too terse, unimaginative, and not self-explanatory. Judging from previous remarks emanating from Southampton, there is probably no fear on that score! May I suggest that the B6700 style is quite good, whether it begins with a $ on a newline, or is enclosed within a PASCAL comment. Look it up in the manuals (Algol say), but breifly one can SET, RESET or POP an option name, each being regarded as on a bit-stack (48-bits deep). SET and RESET push the stack down and insert the new value. Some options can have numeric values, in which case SET/RESET/POP are not relevant, and a few are special (INCLUDE). User-defined options are also allowable, allowing the user to set up source text which is parameterizable at compile-time (as assembly-code programs could often be). We use user-options to control the insertion of checking code in the semantic code generation routines (compiler debugging) and probably for B6700/B7700 minor differences.

Examples which might be self-explanatory (unlike T+, etc.):

```
        SET LIST,TABLE,CODE,LINEINFO,EXTRACHECK
        RESET LISTINCL
        POPLISTINCL                 user-option name, freely chosen
        SET OMIT=EXTRACHECK
        ERRORLIMIT=100, HEAP=2000
```

The B6700 native style is

     $SET LIST, TABLE,CODE,LINEINFO

while a PASCAL adaption might be

     (SET LIST, TABLE, CODE, LINEINFO)

SETS

Why do you not make a first implementation of sets which stores them as a packed array of bits, and therefore pointed to by a descriptor? Since there are no set-constants as such (the set-generator is funny), you can get away with it, and operations on sets can be done either by in-line code, or by intrinsic procedure calls. Bit-testing (v in s) can be implemented by using the low 5 bits (=32 bits of your word) as a bit-within-word index, and the remnant of the set-element as a word-within-array index. This is how we shall implement any-size sets, though we move from one-word sets to this. You might choose to implement any-size and introduce an optimization for small sets later.... It might get you out of that silly character problem.

BOUNDS CHECKING

The Welsh's technique of compile-time bounds checking was going to be built into our compiler, but has not been on the first attempt. Primarily this is because we have been focussing on our main problem: the B6700 and its system, and not so much on nice features of the compiler. I think I have a long list of run-time and compile-time improvements which we shall consider when the compiler reaches its second stable point (with a comprehensive file and i/o facility).

Incidentally, I am not at all sure that read(ch) dominates our compiler's speed; intuitively it seems unlikely given the very efficient lexical analyser. My estimate at the moment is that the speed is mainly limited by the symbol table, and by the code generators.

It does not surprize me to hear that Warwick University do not have compiler expertise. It is very rare in B6700 installations; no-one writes compilers for B6700s, or so it seems to us. All sorts of people fudge the existing ones, but that is a quite different activity from creating a complete system. Tell me of your impressions of B6700 expertise available close to you. This is one of the reasons we have been accumulating information in this area, so that we can become experts and hopefully contribute to the future of structured computers.

Arrays, and off-stack storage

I suppose you'd noted that our compiler does not store any multi-word object on-stack (apart from the double-word items such as double-precision and possibly complex variables). Instead there is a descriptor (one-word) for each multi-word object, which describes a linear piece of core containing the object (array of integer, array of record, array of array, record, record of array, etc.). If large enough (say more than 1024 words) the linear store will be segmented into the B6700 256-word pages.

It is not feasible to store arrays within the stack for two good, though not absolute, reasons. Firstly the stack address space is limited; for usual nesting of procedures to 2k to 4k (11 or 12 bits of displacement address), and it is locked into core (not overlayable). One doesn't want to use it up too fast, though it is conceivable that records which do not contain arrays as elements could be in the stack. Secondly, it is presumed that descriptors in the stack refer to off-stack allocated storage, by the MCP. To be sure the Algol compiler has a curious piece of code that can allocate an in-stack array (it turns interrupts off and does some weird things) but I have not yet been able to evoke this action, nor is it likely to be very nice. You see, our solutions are different from your initial ones.

Pointers

Since pointers point only to things in the heap, and since the heap descriptor location is known to the compiler, we store pointers as one-word integers (with zero used as the nil value at present). Since a single vector on the B6700 is limited by the software to about 150k words, we could pack it into less (say 20 bits) if we need to. I'll probably change the nil value to some out-of-bounds index so that the hardware checks will trap it.

Speed and Space

It may interest you to see those sample jobs I sent you and note that PASCAL compiles the twiddly job at about 10% - 20% slower than ALGOL or FORTRAN, but executes in about ½ the code space (4k instead of 8k average) and about 70% of the data space (5k). Space is a global property, so the small size is very welcome. Speed is a local property of programs, and perhaps some tuning will quite reverse the situation, though perhaps not by much.

I await your next letter with interest; I too have forwarded a copy to the Newsletter.

Yours sincerely,

A.H.J. SALE,
Professor of Information Science.

Trans Union
Systems Corporation

111 West Jackson Boulevard
Chicago, Illinois 60604
312/435 3530

An Affiliate of
Trans Union Corporation

October 29, 1976

Mr. Andy Mickel
University Computer Center
227 Exp. Engr.
University of Minnesota
Minneapolis, MN 55455

Dear Mr. Mickel:

Looking over the September issue of the PASCAL newsletter, I quickly concluded that the Tokyo compiler is the only one likely to serve our needs. Since we contemplate using PASCAL in commercial applications we need a reasonably reliable and efficient implementation, and the only other one that meets those criteria is the Manitoba compiler; but it supports I/O only on a card reader and a printer, which (for us) is absurd. But the most recent published information on the Tokyo compiler is dated from the middle of May. Has there been any progress since then? We're strongly interested in getting a copy of it as soon as it is available to run under IBM's OS/370 control program.

Very truly yours,
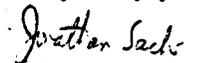
Jonathan Sachs

Timothy M. Bonham

0605/1630 S. 6th St.

Minneapolis, MN 55454

11-04-76

COMMENTS ON SEVERAL ITEMS:

Dynamic Array Parameters...Jacobi's proposal is very welcome; it will
fill an area in Pascal that many users have perceived as wanting in
comparison to other languages. Further, the structure seems to fit
well into the present Pascal syntax and does not appear too difficult
to implement. However, one area of the proposal seems debatable to
me—the standard functions 'low' and 'high'. It seems to me that one
of the reasons for the success of Pascal is the close resemblance of
many of it's features to those of other languages such as ALGOL and
FORTRAN. This makes the task of programmers who already know those
languages and are attempting to learn Pascal much easier. I strongly
agree with C. A. R. Hoare (in his article "Hints for Programming
Language Design") that a main task for the language designer is con-
sistency and consolidation—both within the language, and, if possi-
ble, with other languages. For this reason I would suggest that these
standard functions be given the names 'lowbound' and 'highbound'
instead of 'low' and 'high'. This would be more consistent with the
names used for the similar functions in ALGOL and PL/I. I do not
think that the added disadvantage of slightly longer names is sig-
nificant; especially in consideration of the way in which the names
'lowbound' add 'highbound' more clearly express the meaning of these
standard functions.

Standardisation...It is becoming clear that Pascal is in need of an
"official" standard, formally published by some group such as ANSI.
The language is presently plagued by a host of inconsistent and con-
tradictory additions, extensions, and modifications. If this trend
is allowed to continue, Pascal will soon become no more portable than
BASIC. There were several comments on this in Newsletter #5. I
would like to add my voice to those who seem to be calling for a
Pascal Standards Committee to define a formal "standard". I'm
somewhat nervous about a committee—especially the tendancy they
seem to have to compromise rather than choose the best; but I don't
know of any other acceptable way to go. Hopefully this committee structure
will be similar to that of SIMULA, where the committee members are
themselves implementors (thus insuring that the implementations are
standard and the standards are implementable); and that there will be
a lot of communication between the committee and the users. I would
be very willing to assist such a committee in any way that I could,
and I hope that something is organized soon, before Pascal becomes
more a collection of similar dialects than a single standard, portable
language.

Implementations...Does anyone have any information on a Pascal compiler
for the IBM System 3? (If there are none available, this would seem
to be a good project for some computer science student, since this
machine is widely distributed.) Also, does anyone know of a
compiler for the Control Data 3200?

IMPLEMENTATION NOTES

The IMPLEMENTATION NOTES section of the newsletter is organized as follows:

1) A checklist to consider when sending us information about distributed versions of Standard Pascal.

2) Pascal-P, a "portable" compiler of Pascal for a hypothetical "stack machine". It comes on tape as a kit and may be used to bootstrap compilers onto real computer systems.

3) Other portable compilers: Pascal Trunk compiler, Pascal-J, Pascal-S, and Concurrent Pascal.

4) Implementation independent software writing tools.

5) Compilers and software writing tools for specific computers sorted by computer system.

Our policy is to print only new information. If you do not find what you are looking for in Newsletter #6, check #5.

As Newsletter #6 goes to press, we still do not have enough implementation and distribution information. However, thanks to Timothy Bonham, we sent requests for information to over 80 known implementors late in October. The responses we have received since then have been very gratifying. We thank all of you who have taken the time to respond, the replies have been a big boost for this section of the newsletter.

Again we must stress that we need more information. The PUG Newsletter is the focal point for communications dealing with Pascal; implementors and distributors must keep us informed. We encourage users to share their experiences by sending qualitative and quantitative descriptions of particular implementations. Please realize that individual requests for information are a great drain on our resources.

Those sending information are encouraged to consider the checklist, and if possible, to supply a short order form (both "camera ready"). To further the spread of Pascal, and avoid duplication of effort, this section should be kept complete and up to date.   -John.

CHECKLIST - CHECKLIST - CHECKLIST - CHECKLIST - CHECKLIST - CHECKLIST

1. Name, address, and phone number of implementor and distributor.

2. Machine(s) (manufacturer, model/series).

3. Operating system(s), minimal hardware configuration, etc.

4. Method of distribution (cost, magnetic tape formats, etc.).

5. Documentation available (machine retrievable, in form of a supplement to the book Pascal User Manual and Report).

6. Maintenance policy (for how long, future development plans, accept bug reports).

7. Fully implements Standard Pascal? (why not?, what's different?)

8. Compiler or interpreter? (written in what language, length in source lines, compiler or interpreter size in words or bytes, compilation speed in characters per second, compilation/execution speed compared to other language processors (e.g. FORTRAN))

9. Reliability of compiler or interpreter (poor, moderate, good, excellent).

10. Method of development (from Pascal-P, hand coded from scratch, bootstrapped, cross-compiled, etc.; effort to implement in man months, experience of implementors).

CHECKLIST - CHECKLIST - CHECKLIST - CHECKLIST - CHECKLIST - CHECKLIST

PASCAL-P

It seems that Pascal-P has been stabilized/frozen. In a letter of 14 Sep 76 to George Richmond, Niklaus Wirth stated: "As for Pascal-P, where we have done a major revision this past spring, I cling to the hope that we can leave it at that, merely continuing the handling of new orders."

To order Pascal-P, use the updated form on the following pages (* we apologize for the disjointness of the two parts of the form *). See Newsletter #5 for more complete information on Pascal-P, in particular for explanations of the installation parameters and magnetic tape format.

If you are in Europe, Asia, or Africa, order from:

Ch. Jacobi
Institut fur Informatik
E.T.H.-Zentrum
CH-8092 Zurich
Switzerland
(phone: 01/32 62 11)

Prices are printed on the order form, and include the cost of a mini-tape.

If you are in North or South America, order from:

George H. Richmond
Computing Center: 3645 Marine St.
University of Colorado
Boulder, CO 80309
U.S.A.
(phone: (303) 492-8131)

$50 for P3 and P4 tape and document.
add $10 if Colorado supplies the tape.
add $30 if the version is to be pre-
    configured to your machine
    (necessary if you do not have
    access to a working compiler).
add $10 for a nine-track tape.
add postage if not pre-paid (if you wish
    to be billed).

(* price information for options B *)

If you are in Australia, New Zealand, or Oceania ("Antarctica too?"):

Carroll Morgan
Basser Department of Computer Science
University of Sydney
Sydney, N.S.W. 2006
Australia

$A30 for P3 and P4 tape (option A).

(* price information for options B and C is unavailable. *)

IMPLEMENTATION   NOTES

Order form for the revised Pascal P system.

Please provide us with your revised Pascal P system according to the specifications on the this form.

Address for delivery of the system

The characteristics of our installation are

Machine type

Operating system

Installation parameters (to be filled for case 'A' below)

intsize		intal	
realsize		realal	
charsize		charal	
boolsize		boolal	
ptrsize		ptral	
setsize		setal	
stackelsize		stackal	
strglgth			
intbits			
sethigh		setlow	
ordmaxchar		ordminchar	

(SOURCE INFORMATION, PROPOSALS FOR EXTENSIONS TO STANDARD PASCAL, BUG REPORTS, PROGRAM WRITING TOOLS, ETC.)

A ☐ - Pascal P4 compiler (in Pascal).
     - Pascal P4 compiler (in P4 code).
     - An assembler interpreter of P4 code (in Pascal, for
       documentation purposes, all alignment and size
       constants set to 1).
     - Pascal P compiler implementation notes with update
       list.
     - Pascal P3 compiler (in Pascal).
       With line numbers, to indicate where it differs
       from Pascal P2. (All installation parameters set
       to a standard value.)
     Charge SFr 160.-

B ☐ (For users who have access to a CDC 6000 computer and
     want to experiment with the compiler)
     - Pascal P4 compiler with some changes, so that it is
       accepted by the Pascal 6000 compiler (in Pascal). [All
       accepted by the Pascal 6000 compiler (in Pascal).]
       Installation parameters set to a standard value.)
     - An assembler interpreter (in Pascal, as in package
       'A').
     - Pascal P compiler implementation notes with update
       list.
     Charge SFr 80.-

C ☐ - Update list to 'PASCAL-P compiler implementation notes'.
     Charge SFr 5.-

     Date :   Signature :

UWEC

UNIVERSITY OF WISCONSIN - EAU CLAIRE / EAU CLAIRE WISCONSIN 54701

25 August 1976

PASCAL-P Progress Report

Our interpretive PASCAL-P system has been running since
November 1975. This summer the portable compiler was
rewritten in Burroughs B5700 compatible ALGOL. This new
version of the compiler generates P-code which is sub-
sequently "executed" by our P-code assembler/interpreter.
Compile times have improved by a factor of 16 as com-
pared to the interpretive system. I expect a further
improvement of two to four will be achieved by rewrit-
ing the procedures INSYMBOL and NEXTCH.

Using the (slightly modified) PASCAL source code supplied
with the PASCAL-P implementation kit, this new compiler
has successfully compiled both the PASCAL-P compiler and
the P-code assembler/interpreter. The source code for
the PASCAL-P compiler contains several records (lines in
PASCAL terminology) longer than 80 characters. These
had to be rewritten/shortened to make them acceptable
to our new compiler. (We expect input to come from cards
or a teletype.) Also, one or two long string constants
had to be broken in two to satisfy the STRGLGTH restric-
tion of our system.

The source code for the P-code assembler/interpreter was
a bit more troublesome since it is written in standard
PASCAL rather than PASCAL-P. The problem areas were:
too many long constants in procedure INIT; the standard
types TEXT and ALFA; the standard procedures RESET,
REWRITE, and PACK; an argument of type BOOLEAN in a
WRITE invokation; an octal (that's right, folks!) format
specification in a WRITE invokation; an actual parameter
that was a string constant passed to a procedure expect-
ing a PACKED array of type CHAR; the attempt to refer-
ence the procedure PUSH in procedure EX3 (PUSH is local
to procedure EX0); string constants too long for our
system; standard I/O procedures without file parameters;
semicolons preceding the final END in case statements
(surprise!); and a function (BASE) of type subrange.

Left column

UWEC

UNIVERSITY OF WISCONSIN - EAU CLAIRE / EAU CLAIRE WISCONSIN 54701

PASCAL-P Progress Report	Page 2

Having completed the rewrite of the compiler, the next
step was obvious -- modify it!

The error codes emitted by our new compiler are different
from those emitted by the Zurich compiler. The compiler
tests for over 250 different syntax errors and each of
these errors is now associated with a unique error code.
This allows the corresponding error messages to be more
explicit and, hence, useful to the novice users of our
system.

We have added another (extremely useful) type of comment
to our PASCAL system. A percent sign (%) is used to
signal the compiler that the rest of the current source
image is to be ignored (our system respects card bound-
aries in this case). This allows the programmer to place
short documenting comments after the complete statement. This
type of comment is very useful. It is much less error
prone than the multi-character comment delimiters in
PASCAL and PL/I. It speeds up compilation by reducing
the number of characters the compiler must "look at."
It encourages proper documentation by placing the com-
ment alongside the code. (Assembly language programmers
have been doing this for years with satisfying results.)
If portability is a concern, a very simple editing pro-
gram addition will remove the non-standard comments at
the same time the character set conversion is being done.
Assertion: this type of comment should be a part of all
future high-level programming languages.

A future report will outline some of the ways in which
our PASCAL system is being used in support of our Com-
puter Science program here at the University of Wis-
consin - Eau Claire.

Dr. Bruce A. Pumplin

Right column

PASCAL TRUNK COMPILER   (no new information, see Newsletter #5)

PASCAL-J   (no new information, see Newsletter #5)

PASCAL-S   (no new information, see Newsletter #5)

CONCURRENT PASCAL

August 1976

Termination of the Concurrent Pascal Distribution

The distribution of the reports and system tapes of Concurrent
Pascal and Solo was terminated in August when I left Caltech
to join the University of Southern California. Papers describ-
ing the language and the operating system have been published
in the IEEE Transactions on Software Engineering (June 1975)
and in Software - Practice & Experience (April-June 1976). The
tapes are now so wide-spread that they can be obtained elsewhere.
Several groups are currently moving the system to other computers.

I will be using Concurrent Pascal at USC, but will not continue
the distribution of the present system.

I would appreciate it very much if you would keep me informed
about your experience in implementing and using Concurrent Pascal.

Yours sincerely,

Per Brinch Hansen

Computer Science Department
University of Southern California
Los Angeles, California 90007

Professor Per Brinch Hansen
Computer Science Department
University of Southern California
Los Angeles, California 90007

August 4, 1976

CONCURRENT PASCAL DISTRIBUTION LIST

Since October 1975 reports and system tapes
have been distributed to 252 institutions:

86 companies
119 universities
31 research laboratories
16 others

AEG - Telefunken
Amdahl Corporation
Analog Technology Corporation
Basic Timesharing Inc.
Bell Telephone Laboratories
Boeing Aerospace Corporation
Bolt, Beranek & Newman
Burroughs Corporation
Comptek Research Inc.
Computer Automation
Computer Consoles Inc.
Computer Sciences Corporation
Computer Systems International
Data General Corporation
Digital Equipment Corporation
E. I. duPont de Nemours
Electromagnetic Systems Laboratories
First Data Corporation
Fisher Controls Company
John Fluke Manufacturing Company
Foxboro Company
General Automation Inc.
General Electric Company
General Radio Company
General Research Corporation
GRI Computer Corporation
GTE Laboratories
Hewlett Packard Corporation
Hitachi Research Laboratory
Honeywell Inc.
Inco Inc.
Incoterm Corporation
Intel Corporation
Interdata Inc.
Intermetrics Inc.
International Business Machines
Arthur D. Little Inc.
Logisticon Inc.
Manufacturing Data Systems
Media Reactions Inc.
Metrology Engineering Center
Mills International
Mitre Corporation

Mobydata
National Cash Register
Nuclear Data Inc.
Republic Electronics
Rockwell International
Sanders Associates
Softech Inc.
Sperry Univac
Squibb & Sons
System Development Corporation
Technology Marketing Inc.
Tektronix Inc.
Texas Instruments Inc.
TRW Systems Group
Varatek Computer Systems Inc.
Varian Associates
Wang Laboratories
Westinghouse Electric Corporation
Xerox Research Center

Bell Northern Research (Canada)
C.C.E.T.T. (France)
Databank Systems (New Zealand)
Dynalogic Corporation (Canada)
EDB-Sentret (Norway)
Edfor Information Consultants (Canada)
Electronics Corporation of India
L.M. Ericsson (Sweden)
Finning Tractor & Equipment (Canada)
Ikaslan S.A. (Spain)
IMAG (France)
Logica Ltd. (England)
C. Olivetti (Italy)
Philips CTI (France)
Philips (Germany)
Regnecentralen (Denmark)
Reuters Ltd. (England)
Christian Rovsing (Denmark)
Siemens AG (Germany)
Softlab (Germany)
Software Sciences Ltd. (England)
Systems Designers Ltd. (England)
Telesincro (Spain)

Brandeis University
California Polytechnic State University
California State University, Northridge
Carnegie-Mellon University
Case Western Reserve University
Clarkson College
Cornell University
Duke University Medical Center
Georgia Institute of Technology
Harvard University
Johns Hopkins University
Kansas State University
Massachusetts Institute of Technology
Michigan State University
Naval Postgraduate School, Monterey

New York University
Princeton University
Purdue University
Oral Roberts University
Sangamon State University
Stanford University
Syracuse University
The College of Wooster, Ohio
University of Arizona
University of California, Berkeley
University of California, Irvine
University of California, San Diego
University of Colorado
University of Delaware
University of Florida
University of Iowa
University of Maryland
University of Michigan
University of Minnesota
University of North Carolina, Chapel Hill
University of Texas, Austin
University of Utah
University of Washington
University of Wisconsin
University of Wyoming
Washington State University
Washington University, Missouri
West Coast University
Western Washington State College
West Virginia University

Australian National University
Bezalel Academy (Israel)
Carleton University (Canada)
Chalmers Technological University (Sweden)
Concordia University (Canada)
Durham University (England)
Ecole Polytechnique Federale (Switzerland)
Eidg. Technische Hochschule (Switzerland)
Fachhochschule Reutlingen(Germany)
Simon Fraser University (Canada)
Imperial College (England)
Instituto Politecnico Nacional (Mexico)
Katholieke Universiteit Leuven (Belgium)
Keio University (Japan)
Johannes Kepler Universitat (Austria)
Kyoto University (Japan)
McGill University (Canada)
Monash University (Australia)
Neu-Technikum Buchs (Switzerland)
Politecnico di Milano (Italy)
Queen's University (Canada)
Queen's University (Northern Ireland)
Royal Institute of Technology (Sweden)
Seikei University (Japan)
Technical University of Delft (The Netherlands)
Technical University of Denmark
Technical University of Eindhoven (The Netherlands)

Technische Hogeschool Twente (The Netherlands)
Technischen Hochschule Munchen (Germany)
Technische Universitat Berlin (Germany)
Technische Universitat Wien (Austria)
Trinity College (Ireland)
Universidad Politecnica (Spain)
Universidad Simon Bolivar (Venezuela)
Universite Catholique de Louvain (Belgium)
Universite de Montreal (Canada)
Universite Paris
University of Aarhus (Denmark)
University of Adelaide (South Australia)
University of Alberta (Canada)
University of Amsterdam (The Netherlands)
University of Bonn (Germany)
University of Bradford (England)
University of British Columbia (Canada)
University of Brussels (Belgium)
University of Cape Town (South Africa)
University of Copenhagen
University of Dortmund (Germany)
University of Edinburgh (United Kingdom)
University of Glasgow
University of Hamburg (Germany)
University of Karlsruhe (Germany)
University of Kaiserslautern (Germany)
University of Kiel (Germany)
University of Manitoba (Canada)
University of Newcastle upon Tyne (England)
University of New South Wales (Australia)
University of Oslo (Norway)
University of Saskatchewan (Canada)
University of St. Andrews (Scotland)
University of Stuttgart (Germany)
University of Sydney (Australia)
University of Tampere (Finland)
University of Tasmania (Australia)
University of Tokyo (Japan)
University of Toronto (Canada)
University of Tromso (Norway)
University of Trondheim (Norway)
University of Warwick (England)
University of Waterloo (Canada)
U.S.S.R. Academy of Sciences
Vrije Universiteit (The Netherlands)

Electromagnetic Systems Laboratories, Sunnyvale, California
Jet Propulsion Laboratory
Los Alamos Scientific Laboratory
M.I.T. Lincoln Laboratory
Naval Electronics Laboratory, California
Naval Research Laboratory, Maryland
Naval Undersea Center, California
Naval Underwater Systems Center, Connecticut
New Mexico Institute of Mining & Technology
Oregon Museum of Science and Industry
Pacific Marine Environmental Laboratory, Washington
Research Triangle Institute, North Carolina

Michael Reese Hospital, Illinois
Southwest Regional Laboratory
Stanford Artificial Intelligence Laboratory
Stanford Linear Accelerator Center
Stanford Research Institute
University Hospitals, Cleveland, Ohio
University Hospitals, Cleveland, Ohio
U.S. Army, Fort Meade, Maryland
U.S. Army Research Laboratory, Illinois
USC Information Sciences Institute

Australian Atomic Energy Commission
Central Institute for Industrial Research (Norway)
CERN (Switzerland)
Commonwealth Scientific and Industrial Research Organ (Australia)
Koninklijke/Shell Laboratorium (The Netherlands)
National Physical Laboratory (England)
Norwegian Defense Research Establishment
Oesterreichische Studiengesellschaft fuer Atomenergie (Austria)
Max Planck Institut fur Biochemie (Germany)
Royal Radar Establishment (England)

AN AUTOMATIC FORMATTING PROGRAM FOR PASCAL   (* received 10 Oct 76 *)

Jon Hueras and Henry Ledgard
Dept. of Computer and Information Science
Univ. of Mass. / Amherst, Ma. 01002

Imposing formatting restrictions necessarily imposes a burden on a
programmer, particularly on a student programmer, since he must keypunch or type
in the entire program himself. It is therefore useful to have a facility for
taking arbitrarily formatted source code and automatically prettyprinting it.
However, the design of any such prettyprinter must deal with several serious
issues.

Typically, automatic prettyprinters take a heavy hand in formatting a
program, right down to every last semicolon. Such a scheme either formats
everything in a rigid fashion, which is bound to be displeasing, or else it
provides the programmer with a voluminous set of "options". Furthermore, such
a scheme must do a full syntax analysis on the program, which means that it
falls prey to the bane of all compilers: error recovery. Thus, before a program
may be prettyprinted, it must be completely written and debugged. If the
programmer wishes to prettyprint a program still under development, he is out
of luck, or else he must do it by hand, in which case he has no need for an
automatic prettyprinter when he is done.

We believe that it is not necessary to impose more than a <u>minimum</u> set of
restrictions, and that any prettyprinter should yield to the programmer's
discretion beyond this minimum. No matter how many options a prettyprinter has,
it cannot possibly have one to please everyone in every possible case. We
further believe that a prettyprinter should not commit itself to a full syntax
analysis. It should only do prettyprinting on a local basis, dealing with
individual constructs rather than entire programs.

In order to demonstrate these assertions, we have designed and implemented,
in PASCAL, just such a prettyprinter. It is intended mostly as an editing aid,
and thus does not include most of the "kitchen sink" facilities used by other
prettyprinters. It simply rearranges the spacing and indentation of certain
constructs in order to make the logical structure of a program more visually
apparent. Furthermore, the prettyprinter forces only a minimum amount of
spacing and indentation where needed. Any extra spaces or blank lines found in
the program beyond the minimum required are left there. This leaves the
programmer a good deal of flexibility to use as he sees fit.

The prettyprinter that we have implemented is written entirely in standard
PASCAL (according to the Revised Report in Jensen and Wirth), and should compile
and run using any PASCAL compiler that supports this standard. We have compiled
it using the PASCAL 6000-3.4 compiler from Zurich, and run it in 11K (octal) of
core on our CDC CYBER 74. The program, as written, is highly modularized and
table-driven, and is therefore extremely easy to modify and upgrade.

A copy of the program, with documentation, is available from H. F. Ledgard:
$TF for a standard 7-track tape or cards, or $2 with a user-supplied tape.

SOFTWARE WRITING TOOLS

AMDAHL 470   (see IBM 360/370 series)

BURROUGHS B-1700

TELEPHONE: 692 1122

BASSER DEPARTMENT OF COMPUTER SCIENCE

School of Physics (Building A28),
University of Sydney, N.S.W. 2006

3rd November, 1976

Timothy Bonham,
Pascal Implementations,
University Computer Centre,
227 Experimental Eng. Building,
University of Minnesota,
MINNEAPOLIS.   MN 55455
U.S.A.

Hello,

        This letter is in response to your inquiry re our
B1726 Pascal implementation. Unfortunately this project
was abandoned over a year ago because of lack of time, and
the (then) continuing lack of suitable documentation from
Burroughs.

        However, I can give you details of other B1726 Pascal
implementations. These are:

    (1) Pascal implemented by Elliott Organick's group
         at the University of Utah. This compiler (and
         interpreter) is based substantially on Brinch
         Hansen's Solo Sequential Pascal compiler (i.e.,
         it compiles in 7½ passes) and we have a copy of
         it. We haven't had much experience with it yet,
         but from what we have seen so far we're not very
         impressed. The compiler appears to be somewhat
         glitchy and unfortunately does not adhere to the
         conventions observed by Burroughs-supplied
         compilers.

    (2) At the European B1700 University Users Meeting on
         July 15-16, 1976, the following projects were
         mentioned:

        (a) University of Zurich (which is not ETH,
             incidentally) are working on

            (i) implementation of a P-code interpreter
                for Pascal.

2

            (ii) implementation of a Pascal compiler
                 to compile down to the SDL S-machine as
                 defined by Burroughs.

        The contacts are Mr. P. Schultess for (ii)
        and Mr. K. Hauserman for (i).

    (b) P. Albrich of the University of Karlsruhe
         (Germany) is implementing Concurrent Pascal.
         Nothing was mentioned about (Sequential)
         Pascal.

    (c) M. Ellison of the University of Newcastle-upon-
         Tyne is implementing "another Pascal Machine
         architecture". He reports that an interpreter
         for this system's machine code has been de-
         bugged and that it is to be benchmarked with
         "Zurich's Pascal-P version 1.0".

        There is to be another European user's meeting at
Karlsruhe in February 1977. Our own contact for all the
European information is through the University of Newcastle-
upon-Tyne.

        I hope the above information will enable you to obtain
the material required for the Newsletter.

                                    Yours sincerely,

                                    Antony Gerber

BURROUGHS B-4700

    William C. Price of Burroughs Corporation, 460 Sierra Madre Villa Ave.,
Pasadena, CA 91109, has informed us that he and Robert M. Lansford also of
Burroughs, 3620 Greenhill Road, Pasadena, CA 91107, are preparing a
description of their B4700 Pascal implementation. We will print this
in Newsletter #7.

BURROUGHS B-5700   (implementations exist)

BURROUGHS B-6700/B-7700

STATUS REPORT : BURROUGHS B6700/B7700 PASCAL COMPILER

The University of Tasmania is developing a compiler for PASCAL to
produce executable programs on the Burroughs B6700/B7700 computer systems.
The compiler is currently (1976 October) operational but with only a
simple i/o system (default file declarations and PASCAL 1 i/o statements).
Current work concentrates on implementing general file and file attribute
declarations, and on extending i/o statements to a usefully sized set.
declaration, and on extending i/o statements to a usefully sized set.

Our current policy is not to release the compiler until it reaches this
next stable state as otherwise copies of the simplified version will
proliferate. We would welcome any expressions of interest from B6700
users in the compiler, and will add addresses to our list of interested
people, as this will determine the level of support that will be necessary
to maintain the final system. The anticipated release date is December
1976.

Compiler information

The Burroughs PASCAL compiler is a true compiler for the B6700: it generates
a code-file which can be executed by the system. The code-file contains
no BINDINFO and cannot be bound to any other language at present (as is the
case with BASIC). The execution speed and compilation speed of PASCAL
are comparable to the Algol compiler, though much less code and data space
is needed for compilation.

The compiler is based on PASCAL-P, but is written in Burroughs Algol. It
maintains standard treatment of PASCAL features with the addition of
B6700-compatible compiler-options and other features. The aim is to
produce a compiler which will comply with the B6700 conventions as
embodied in the Algol/FORTRAN/COBOL/etc compilers for that system, and
for which transferable skills from other Burroughs subsystems can be
retained.

Arthur Sale
Professor of Information Science
University of Tasmania
Box 252C G.P.O.
Hobart, Tasmania 7001

The University of Tasmania

Postal Address: Box 252C, G.P.O., Hobart, Tasmania, Australia 7001

Telephone: 23 0561. Cables 'Tasuni' Telex: 58150 UNTAS

IN REPLY PLEASE QUOTE:

FILE NO.

IF TELEPHONING OR CALLING

ASK FOR

Department of Information Science

8 November, 1976.

Pascal Implementations:
University Computer Center,
227 Experimental Engineering Building,
University of Minnesota,
MINNEAPOLIS, MN 55455
USA:

Dear Timothy Bonham,

Burroughs B6700 PASCAL Implementation

1. **Names, etc. of implementors**

   Professor A.H.J. Sale   (Phone Tasmania (STD 002) 23-0561 Ext. 435)
   Dept. of Information Science
   University of Tasmania
   Box 252C G.P.O.
   Hobart, Tasmania 7001.

2. **Machine tested on**

   Burroughs Model III B6700 processor, with vector mode hardware, 196k words
   of main store, disk, 4 pack drives, etc. Machine operates in university
   environment with heavy interactive use.

3. **Operating System, etc.**

   Burroughs MCP Version 11.8 operating system with (few) minor local
   modifications for accounting, etc. Minimal system to operate: not known,
   but unlikely to be any B6700 that small (store demands are low, little
   else is critical).

4. **Method of distribution**

   Not officially released (December?), nor cost determined. Format will be
   via magnetic tape (both 7-track and 9-track drives available).

5. **Documentation available**

   Under development. Will be in form of user manual as well as supplement
   to Pascal User Manual and Report.

6. **Maintenance policy**

   To be maintained for teaching use within University as well as larger
   aims. Reported bugs should be fixed as soon as possible, with notification
   to users. Duration of support not yet determined; several developments
   are also pending.

7. **Standard-compatibility**

   Does implement PASCAL in Report with following exceptions where noted
   with reasons:

program heading: reserved word program is synonymous with procedure;
no parameters (files) are permitted after the program heading.
Reason: CDC anachronism of no utility in our installation, and
likely to be confusing.

set constructor of form A..B not implemented. Reason: future plan.
FORTRAN control character on print line not implemented. Reason:
a ridiculous feature to standardize.
Full Pascal I/O not implemented. Reason: future plans, present
scheme is PASCAL-1-like.

Extensions

Various reserved words, character set transliterations.
Burroughs comment facility
ELSE in CASE
File attributes in declarations
Format declarations
Extensive Burroughs-compatible compiler options (Pascal option mode
not implemented).

8. Compiler or Interpreter, etc.

Compiler:
generates B6700 code-files which are directly executed by the B6700
with MCP. There is as yet no BINDINFO in the codefile so that
it is not possible to link Pascal to modules compiled by other
systems.
written in B6700 Algol entirely.
Characteristics:
compiles about 20% slower than FORTRAN or ALGOL, but in about 2/3 of
their space (for test programs about 4-5k words on average
instead of 8-10k). Elapsed compilation times similar, though
PASCAL slower. Speed should be improved by eventual tuning.
executes at same speed as FORTRAN and ALGOL (code is very similar and
optimal) and takes generally longer elapsed residence time
primarily due to MCP intervention to create new segments for
record structures (not present in FORTRAN/ALGOL). Elapsed
residence times about 20% greater than equivalent ALGOL.
one-pass system: code generated is very close to optimal for B6700
unless checking requirements of Pascal intervene to inhibit this.
options include listing of object code in symbolic and/or absolute
form, editing of input, etc.

9. Reliability

Excellent. Only one system crash during testing attributed to Pascal
(in run #2), and a total of two serious bugs uncovered during extensive
testing. On a machine with minimal protection against aberrant compilers
as is the B6700, a high level of confidence is essential. The compiler
code-generator section incorporates many reasonableness checks on the code to
trap some flaws before they get executed and to aid in tracing errors.

10. Method of development

Hand-coded using Pascal-P as a guide/model. All other paths offered
much greater difficulty on B6700 due to special nature of machine/system.
Man-month details not kept, and project proceeds in fits and starts as
teaching intervenes. Project has thus far been limited to two people:
Prof. A.H.J. Sale and R.A. Freak (support Programmer).

I trust the above comments meet your need. I enclose a copy of a brief
technical report on some thoughts on Pascal implementation around August of this
year. I am, and intend to remain, a member of P.U.G.

Yours sincerely,   A.H.J. SALE ... PROF. OF INFORMATION SCIENCE.

UNIVERSITÄT KARLSRUHE
INSTITUT FÜR INFORMATIK II

Dipl.-Inform. Uwe Kastens

75 KARLSRUHE 1.den 1. Okt. 1976
ZIRKEL 2
POSTFACH 6380
TELEFON (0721) 608 __3972

Pascal User's Group
c/o Andy Mickel
University Computer Center:227 ExpEngr
University of Minnesota

Minneapolis, MN55455

USA

Dear Mr. Mickel,

Until now we have an interpreting PASCAL-System running on the
B6700. We got the "PASCAL-Implementation-Kit" with the P2-Compiler
(as described in Nori, e.a., "The PASCAL P-Compiler Implementation
Notes", ETH Zürich) and translated the assembler/interpreter for
the hypothetical stack computer from PASCAL to Burroughs Extended
Algol. So we can compile PASCAL-programs by interpreting the
PASCAL-Compiler and execute the generated Code for the stack
computer by interpreting it. This technique is very time- and
space- consuming: The PASCAL-Compiler needs about 30 min B6700
processor time to compile itself.

We didn't complete the whole bootstrap yet, because of several
problems concerning the generation of B6700 machine code in
general.

Sincerely

UNIVERSITÄT KARLSRUHE
INSTITUT FÜR INFORMATIK

75 KARLSRUHE I. den 5.11.1976
ZIRKEL 2
POSTFACH 6380

Dipl. Inform. U. Kastens

TELEFON (0721) 608 3495

University of Minnesota
University Computer Center
227 Experimental Engineering Building
Minneapolis, Minnesota 55455
c/o Mr. Tim Bonham
U.S.A.

Dear Mister Bonham,

According to your letter dated October 25, 1976,
directed to Prof. Dr. G. Goos, I will give you some
information about the state of our PASCAL-activities.

We have an interpretive PASCAL-System running on
the B 6700. It is based on the PASCAL-P2 compiler and
the assembler/interpreter from Zürich. We translated the
latter one to a Burroughs Extended Algol-Program. This
interpretive version is naturally very time and space
consuming. (The compiler compiles itself in about 30
minutes CPU-time.) The system is available on magnetic
tape for a nominal charge of $ 20.oo when tape is supp-
lied, $ 30.oo otherwise. We have a short note for users
of the system (in German only).

Another project on the B 6700 is based on an early
version of the PASCAL-JANUS-Compiler (from Boulder, Colo-
rado). PASCAL is translated via JANUS, STAGE II and an
assembler to B 6700 machinecode. We didn't test the
system enough to say how reliable it is.

Both projects were not further developed nor main-
tained because of a lack of manpower at our institute
and deminished computing capacity on the B 6700.

Sincerely,
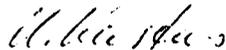
(U. Kastens)

Ka/Wh.

CII 10070, IRIS 50, IRIS 80

Olivier Lecarme of the Universite de Nice, Laboratoire D'Informatique,
Parc Valrose, 06034 Nice Cedex, France, has helped to clear up our
confusion about the CII machines. In a letter of 16 Sep 76, he wrote:
"Although CII 10070 is a nickname for Xerox Sigma 7, the CII Iris 80
is another machine, more precisely an extension of the first one. Moreover,
the CII operating system is different from Xerox and transporting a Pascal
compiler from a Xerox Sigma 7 to a CII Iris 80 probably would not be a
trivial job. A Pascal compiler for both CII machines has been written by
Messrs. Thibault and Mancel of IRIA (Research Institute in Informatics and
Automatics, a French government agency), by bootstrapping the first CDC 6000
compiler. It has now been upgraded to accept Standard Pascal and to allow
separate compilations, and it is officially distributed by IRIA, a case
which seems unique. Its overall performance seems to be quite good, and
it is used in French universities which have one of these machines.
"The CII Iris 50 is a completely different machine, much smaller,
and we have some trouble in Nice when trying to implement Pascal. Pascal-P
works interpretively, but it is unusable for programs larger than one page,
and consequently it cannot be used as a tool for bootstrapping a true
compiler. I plan to write a brief paper for describing the bootstrap
method which will be used, and which seems to be a unique one. Maybe it
could be done in time to be included in newsletter number 6." (* perhaps
Newsletter #7? *)

CONTROL DATA CYBER 18   (an implementation exists)

2550   (Control Data supports a cross-compiler on the
         6000/Cyber 70, 170 series)

3300   (implementations exist)

3600   (an implementation exists)

6000/CYBER 70, 170 SERIES   (see also Newsletter #5)

There is very little fresh news on this implementation. It is rumored
that Zurich has written a first modset (UPDATE1) to Release2 of Pascal6000.
We at Minnesota have not received it yet. There is a new price list for
distribution tapes, but no new order form. See Newsletter #5 for the old one.

If you are in Europe, Asia, or Africa, order from:

Ch. Jacobi
Institut fur Informatik
E.T.H-Zentrum
CH-8092 Zurich
Switzerland
(phone: 01/32 62 11)

The handling charge for Release2 is SFr. 100
which includes the cost of a mini-tape.
Do not pay in advance, you will be
charged at delivery.

If you are in North or South America, order from:

George H. Richmond
Computing Center: 3645 Marine St.
University of Colorado
Boulder, CO 80309
(phone: (303) 492-8131)

$60 for Release2 tape and documentation.
$50 if you supply the tape.
$25 if you have Release1- you supply
the tape, and no documentation is
included.

If you are in Australia, New Zealand, or Oceania, order from:

Arthur Sale
Basser Department of Computer Science   **(ask for folleted tape and document:**
University of Sydney
Sydney, N.S.W. 2006
Australia

CONTROL DATA 7600/CYBER 76

Pascal on the CDC 7600

This Pascal compiler is essentially the Zurich 6000 -3.4 compiler.
The run-time system is based on that of Hans Jorsanstad (see Newletter #4)
The compiler is release 2. It was developed by cross-compiling from our
CYBER 72.

The compiler is currently running under SCOPE 2.1.3 and will re-compile
itself on a 'half-size' (32K SCM) machine.

Compilation Speed

57000 characters/second approx. Compiler re-compiles in less
than 10 seconds.

Execution Speed

Pascal execution speed has been measured by using the obvious encoding
in Pascal of Wichmann's Synthetic Benchmark (see Computer Journal Vol.19 No.1).
The units are in thousands of Whetstones.

Compiler and optimisation level	No runtime Checking	Array Board Checking
ALGOL 4 (0=5)	1996	1230
PASCAL	6850	6240*
FTN (OPT=2)	9345	3174**

* Using T + option i.e. all run time checks included
** forces OPT = 0

Maintenance

The situation is unclear at present. UMRCC will presumably, out
of self-interest assist with bugs in the 7600 dependent code. The vast
majority of the compiler and library is standard Zurich code.

Contacts

Mr. H. D. Ellison or Mr. A. P. Hayes at
UMRCC, Oxford Road, Manchester M13 9PL, England

CRAY RESEARCH CRAY-1

IN REPLY
REFER TO: C-11
MAIL STOP: 296   October 27, 1976

Mr. Andy Mickel
Pascal User's Group
University Computer Center
227 Experimental Engineering Bldg.
University of Minnesota
Minneapolis, Minnesota 55455

Dear Andy:

I want to tell you about an implementation of Pascal for the Cray
Research CRAY-1 computer done here at Los Alamos. I will follow the
checklist outline on page 44 of PASCAL NEWSLETTER #5.

1. Implementor and maintainer:

   Robert T. Johnson
   C-11, MS/296
   Los Alamos Scientific Laboratory
   P. O. Box 1663
   Los Alamos, New Mexico 87545

   phone: (505) 667-5014

2. Machine: Cray Research, Inc., CRAY-1.

3. Operating System: It was called the Benchmark Operating System,
   and is now called BOS after several extensions and enhancements.

4. The compiler has not been distributed elsewhere, and no arrange-
   ments have been made for the distribution.

5. Documentation consists of a two-page description of how to access
   and use the compiler.

6. Maintenance on this compiler is suspended; the compiler is at
   the end of its usefulness on the CRAY-1, because support for
   it cannot keep up with system development and changes. The
   compiler has been superceded for our development work by a
   cross-compiler for the Model language designed and implemented

by James B. Morris, Jr., of this Laboratory. Model was based on
Pascal and retains many of its concepts. (Cf. "Abstract Data
Types in the Model Programming Language," Robert T. Johnson and
James B. Morris, Proceedings of the Conference on Data: Abstraction,
Definition, and Structure, SIGPLAN Notices, Vol. 8, Number 2, 1976.)
Pascal-P2 did not seem to provide a good enough basis for further
work, perhaps P4 will be better.

7. The compiler implements Pascal-P2 except for I/O. Until recently
   no I/O was available on the CRAY system.

8. It is a cross-compiler which runs on a Cyber 70 and generates
   CAL (CRAY-1 Assembly Language). The code generation is straight-
   forward and, consequently, the object code quality is low. The
   CRAY-1 requires a more sophisticated code generator to use its
   register resources and instruction overlap capabilities.

9. The compiler reliability has been good for programs which were
   first tested on the CDC 6000 compiler.

10. The compiler is a cross-compiler written as a translator of
    P-code output from a (slightly modified) Pascal-P2 compiler.
    Both the Pascal-P2 and the code translator use the CDC 6000
    compiler. About 3 man-months of effort have been expended on
    this development.

The quality and format of Newsletter #5 were impressive. Keep up the
good work.

                        Sincerely,

                        Bob Johnson

xc: ISD-5 (2)

DATA GENERAL Nova 800, Nova 1200, Supernova, Eclipse

(no reported implementations - we need information)

DIGITAL EQUIPMENT (DEC) PDP-8   (an implementation is underway)

DIGITAL EQUIPMENT (DEC) PDP-10, DECSYSTEM-10   (see also Newsletter #5)

UNIVERSITÄT HAMBURG

Institut für Informatik
2 Hamburg 13, Schlüterstraße 66-72

INSTITUT FÜR
INFORMATIK

Prof. Dr. H.-H. Nagel

Mr. Andy Mickel
University Computer Center
University of Minnesota
227 Experimental Engineering Building
Minneapolis, Minnesota 55455
USA

Fernsprecher: 040-4123- 4151}
Sebändezentrale: 9.09(.)} Durchwahl

Telex-Nr.: 214732 uni hh d

Datum und Zeichen Ihres Schreibens   Abrechnungs (bei Antwort bitte angeben.)   Datum

                              Na/Ja   September 24, 1976

Dear Mr. Mickel,

the PASCAL Newsletter Number 5 has been studied by me with great
interest.

I congratulate for the lively exchange of opinions that you enable
by your effort. I would like to supplement the two contributions
which refer to the DECSystem-10 PASCAL-compiler developped at Hamburg.

Mr. J. McCool who seems to be utterly disappointed by our compiler
sent me two letters. The first letter from him (dated March 16,76
arrived here April 9,76) presented a program listing with the remark
that this program started execution and then never could be nudged
out of the terminal input state - no matter what was typed in. I
retyped the program, run it and detected that the initial READLN(TTY)
had been forgotten. Therefore, the heavily recursive program was
always one character ahead of what it meant to be. In order to avoid
any delay due to the Easter holidays 76 I wrote him a letter indicat-
ing the trouble and mailed it personally on Good Friday. A few days
later (April 26, 76) I received a letter (dated March 26, 76) from
Mr. McCool complaining of not having received an answer to his first
letter and indicating two additional compiler errors - the date
error (5-Jan-75) and the fact, that 180.0 on input turned out to be
179.99999 on output due to the conversion routines. Both letters
were sent by ordinary mail. Apparently Mr. McCool did not realize
that Hamburg is on this side of the Atlantic and a letter by surface
mail takes longer to get here than the time after which he shot off
his second letter complaining about not being "serviced" properly.
I mailed a fix for the date error (essentially one instruction needs
to be added in the runtime support) within less than a week - and
that has been all I heard from Mr. McCool.

I mention this to indicate with what expectations one might be con-
fronted if one provides programs for which not even a mailing charge
has been asked by me - on the understanding that no regular mainte-
nance can be given.

The contribution by Mr. Hedrick shows how the complete availability
of source code for compiler and runtime support can be used to up-

grade the compiler by the recipients. At Hamburg we only have a
KA-10 processor - so we could not apply the special conversion in-
structions available with the KI-10 processor. I have been very
interested to learn that the use of these instructions can speed up
some programs by a factor of 2. This is only one of many examples
where our compiler has been corrected and adapted to changing needs
by recipients.

The very specific list of unsatisfactory aspects in our PASCAL-com-
piler given by Mr. Hedrick will now be treated by referring to a
new compiler version which currently is introduced at our installa-
tion:

1. Minimum core for stack, heap and I/O-buffers is allocated. More-
   over, a compiler option allows to specify the amount of core with
   which a program should be executed after compilation.

2. The new compiler has been fully integrated into the DECSystem-10
   Concise Command Language (CCL). The COMPILE, LOAD and EXECUTE
   commands can be given together with a large number of compiler
   options in the standard CCL format.

3. Lower case characters are no longer simply dropped. If they are
   input from a source file of CHAR, lower case characters are con-
   verted to upper case characters, only the control characters are
   dropped (except for lincfeed which is treated as END OF LINE).
   [illegible line]
   [illegible line]

   The problem with simply accepting lower case characters consists
   in the difficulty to exclude runtime errors when SET OF CHAR are
   used since up to now, only two words are used for set represen-
   tation, restricting the base type to 72 values at the maximum.

   As a stopgap measure - extending set representations to more than
   two words requires a nontrivial redesign - ASCII as a superset
   of CHAR has been introduced which covers the complete 7 bit ASCII
   character set but will result in runtime error messages if it is
   used in SET operations.

4. The file OUTPUT does no longer appear.

5. The generation of a listing can be controlled by the usual LIST/
   NOLIST switch in the compile command.

   Note: The compiler has been originally developped for students.
         It is my experience that more trouble results from work-
         ing with outdated listings than from creating a listing
         file which does not even have to be printed - at least
         it is available if the suspicion arises that malfunction
         of the program might be due to typing errors or incon-
         sistent "corrections".

6. The standard way to supplement COMPILE etc. commands with option
   -switches is now available.

7. These errors are known to me and I thought they had been fixed
   in the version distributed July 75. I am sorry for the trouble.
   What Mr. Hedrick refers to as a poor design of parameter passing
   is due to the desire to pass parameters in up to five registers
   in order to save code. The situation admittedly gets complicated
   for the - very rare - cases where more than five parameters have
   to be passed although the compiler is designed to handle these
   situations correctly. The activation of funtions during argument
   evaluation has been corrected, too.

8. The desire for error recovery during terminal input is quite under-
   standable. However, it requires a complete redesign of the (assembly)

input conversion routines - something we simply had not yet time
to accomplish.

9. Has been treated above.

The main improvements of the new compiler version not yet mentioned
in the reply to Mr. Hedrick's points are

10. The PROGRAM and LABEL declaration has been implemented.

11. GOTO out of procedures/functions are implemented.

12. The implementation of formal procedures/functions is adapted to
    the new compiler version from the one introduced at Technische
    Hogeschool Twente, Enschede/Netherland, by a student of Dr. C.
    Bron into an earlier compiler version that he obtained from
    Hamburg.

13. PACK and UNPACK have been adapted to the standard definition given
    in the PASCAL User Manual and Report. Two additional optional ar-
    guments may be given to these procedures indicating the length
    of the array to be packed/unpacked and the starting address in
    the packed array.

14. The standard functions DATE and TIME are implemented as in PASCAL
    6000. Two additional standard functions CLOCK and REALTIME have
    been introduced with integer function values to determine CPU-
    and Real-Time in milliseconds.

15. New standard functions FIRST and LAST have been introduced to
    determine the first and last value of a scalar or subrange.

16. LOWERBOUND and UPPERBOUND allow to determine the respective index
    values for array definitions to enable an easier change of con-
    stant or range definitions.

17. MIN and MAX are available standard functions.

18. User defined scalar variables can be input and output using the
    character representation of the value constants. The conversion
    from or to the internal representation is taken care of by the
    runtime support. Values for SET-variables will likewise be con-
    verted on input/output from resp. to the character representation
    as it appears in a source program SET-constant.

19. The introduction of a CALL enables the termination of the current
    PASCAL program and the immediate start of another program.Communi-
    cation between programs is provided by standard DEC TMPCOR files.

20. Procedures are available to determine the value of option switches
    appended to the EXECUTE command from within the PASCAL program.

21. The PASDDT source level debug system has been enlarged by an op-
    tional stack and heap dump in source level code.

22. A POST-MORTEM-DUMP facility at source language level has been ap-
    pended to the PASDDT system.

23. Runtime checks have been extended to cover a larger number of pos-
    sible trouble points. If a program has been compiled with the
    PASDDT debug option, any runtime error detected - including those
    by the TOPS-10 monitor - will transfer control to the PASDDT-system.

24. PASCAL procedures/functions and MACRO-10 routines can be compiled/
    assembled separately and included together with FORTRAN routines
    into a relocatable object code library.

25. A machine retrievable manual for this DECSystem-10 PASCAL imple-
    mentation in English is available.

This system has been implemented by Mr. J. Fieldt and is currently being tested at our institute. Before distribution, I would like to have an extended period of production use at our installation to avoid disappointment after distribution.

A recent statistic about the distribution of our PASCAL compiler version of July 1975 might be of interest.

	Universities and Colleges	Industry	total
USA and Canada	22	16	33
Europe	12	3	15
South America and Australia	3	—	3
	37	19	56

These are the installations about which I know. Since the compiler could be freely distributed it might well be possible that it is used in places not known to me. From the total known number of 56 I have to subtract the installation of Mr. McCool since I assume that they discontinued use of our compiler after their bad experiences. Another five requests for our compiler have arrived in the meantime.

May I add a last remark concerning reports about PASCAL compiler experiences: since all work done so far seems to proceed on a nonprofit basis by voluntary contributions, the feedback of trouble spots to people generating a compiler version is not optimal. One should, therefore, encourage the distribution of specific complaints as, e.g., those given by Mr. Hedrick for our compiler version. Any unspecific complaint should preferably be returned to the writer with the kindly formulated expectation that he supports his claims by at least giving details of where he encountered trouble, indicating the compiler version to which his remarks apply and from whom he obtained it. Such an editorial policy might add to the value of the PASCAL User Group Newsletter since - among other reasons - it might make the user community aware of trouble not yet identified at their installation.

Sincerely yours,

H. H. Nagel

DIGITAL EQUIPMENT (DEC) PDP-11   (see also Newsletter #5)

Stephen G. Schwarm of E. I. du Pont de Nemours Co., 101 Beech Street, Wilmington, DE 19808, has written us: "I am Chairman of DECUS SIG PASCAL and I will be glad to help with distributing any systems on DEC PDP-11s." (* maybe Steve can help organise this section of the implementation notes. *)

SP SYSTEMS
BOX 5255, STATION A
TORONTO, ONT.
CANADA M5W 1N5
25/OCT/76

FORMER ADDRESS:
BOX 302 SUB 6
SASKATOON, SASK.
CANADA S7N 0W0

TO: PERSONS INTERESTED IN MY PDP-11
    PASCAL IMPLEMENTATION.
CC: PASCAL NEWSLETTER

GENTLEMEN:

THIS LETTER IS TO ADVISE YOU OF THE STATUS OF MY PASCAL IMPLEMENTATION FOR THE PDP-11. I APOLOGISE FOR SOME DELAY IN WRITING THIS LETTER; I'VE BEEN BUSY MOVING.

IN A WORD, MY COMPILER IS DEFUNCT.

MY IMPLEMENTATION HAS NEVER MORE THAN A SPARE-TIME PROJECT, AND PROGRESS WAS SLOW AS I HAVE BEEN VERY BUSY. MY MOVE TO TORONTO ESSENTIALLY ELIMINATES BOTH MY SPARE TIME AND MY CHEAP MACHINE ACCESS. AS MY COMPILER NEVER CAME ANYWHERE NEAR OPERATIONAL STATUS, AND IT IS MOST UNLIKELY THAT I WILL BE ABLE TO DO MORE WORK ON IT ANY TIME SOON, I HAVE ABANDONED THE PROJECT.

Henry Spencer

HENRY SPENCER
MEMBER OF TECHNICAL STAFF

HS:PDP11

![logo]

Electro Scientific Industries, Inc.

13900 N.W. SCIENCE PARK DRIVE
PORTLAND, OREGON 97229
(area code 503) 641-4141
Telex No. 360475

Timothy Bonham
University of Minnesota

Nov. 2, 1976

Dear Tim;

Thanks for your letter. We had noticed the small mention of
ESI Pascal in the last PUG newsletter and were a little concerned
because it made reference to the U. of Ill Pascal and implied that
ours was essentially the same. ESI Pascal was based on the U. of Ill
bootstrap compiler (not the student compiler they are now offering),
but has been so completely rewritten and reshaped that we have no
hesitation in claiming it as our own creation. Briefly, our compiler
runs under RT-11 on any PDP-11 processor in 16K or more and compiles
full Pascal with a few differences. The enclosed documents will
explain more.

Taking your points in order:
1) John Ankcorn did most of the work on the compiler. He and I
are the Pascalians here and can be reached at this phone.
2) All 11's. The compiler source has assembly conditionals to
shape it for the desired machine.
3) RT-11, 16K words. We have an RSX-11M version in the works.
4) see enclosures
5) Our supplement is enclosed. We are working on more.
6) Probably will be one year of unlimited fixes and updates,
followed by an annual subscription service.
7) Basically yes, but see the second page of the supplement.
8) Compiler (Pascal to Macro assembler text). Fits in 12K
with the extra space taken by the symbol table. See the enclosures.
John is preparing a paper describing many of the details.
9) Excellent. It has been used on our laser trimming systems
for more than a year, and we have assiduously searched for and
eliminated bugs.
10) The compiler is written in Macro-11. We started with the
U. of Ill. bootstrap, changed the syntax scanner, totally changed
the code generation, wrote our own expression analyzer,and wrote
our own support package for arithmetic, math and file handling.
Effort has been on the order of 2 man-years, though some of this
time was spent on the applications software for our systems. It
was the first compiler for each of us, which is why we are grateful
to the Illini for their initial help.
I am not sure any of this is directly suitable for publication,
and I think you should resist the newsletter becoming an advertising
forum. Nevertheless, we immodestly think that ESI Pascal is probably
the smallest, fastest, most complete and most reliable compiler for
the PDP-11, and we are not about to hide our candle under a basket.

sincerely, *Dave*

David Rowland
manager, programming

ELECTRO-SCIENTIFIC INDUSTRIES OFFERS A COMPLETE
IMPLEMENTATION OF THE PROGRAMMING LANGUAGE PASCAL FOR PDP-11
COMPUTERS. ALL THE FEATURES OF PASCAL ARE INCLUDED, PLUS
EXTENSIONS FOR HARDWARE CONTROL.

THE ESI PASCAL COMPILER RUNS ON ANY PDP-11 PROCESSOR
UNDER THE RT-11 OPERATING SYSTEM. NO OVERLAYS ARE USED.
THE ONLY 16K OF MEMORY IS REQUIRED. THE COMPILER WILL BE IN
THE PRINCIPLE OF RECURSIVE DESCENT AND MAKES A SINGLE PASS OVER
THE INPUT TEXT AT APPROXIMATELY 3500 CHARACTERS PER SECOND
(PDP-11/05). TWO FILES ARE PRODUCED: A LISTING OF THE SOURCE
INCLUDING ERROR MESSAGES, AND A TRANSLATION OF THE SOURCE INTO
MACRO ASSEMBLER CODE. THE MACRO CODE IS ASSEMBLED AND LINKED
TO A SUPPORT MODULE TO PRODUCE THE EXECUTABLE PROGRAM.
THE SUPPORT MODULE CONTAINS ALL THE PRE-DEFINED FUNCTIONS
AND PROCEDURES, PLUS A SIMULATOR FOR THE PDP-11/40 INTEGER
AND FLOATING POINT HARDWARE. TWO VERSIONS OF THE COMPILER ARE
AVAILABLE: ONE THAT GENERATES PDP 11/40 FIS INSTRUCTIONS (WHICH IS
USED ON 11/03, 11/04,11/05 AND 11/40'S) AND A VERSION THAT GENERATES
11/45 FLOATING POINT. THE SUPPORT MODULE CAN BE CONFIGURED
TO INCLUDE ONLY THE ROUTINES NEEDED BY THE PROGRAM.

ALL THE PASCAL DATA TYPES, DATA STRUCTURES AND STATEMENTS
ARE PRESENT. FORWARD PROCEDURES MAY BE DECLARED. "NEW" AND
"DISPOSE" PROCEDURES ARE AVAILABLE FOR THE DYNAMIC
ALLOCATION OF VARIABLES. PROCEDURES MAY BE DECLARED AS EXTERNAL,
PRE-COMPILED AND INSERTED IN THE PROGRAM AT LINK TIME.

ESI'S EXTENSIONS ALLOW VARIABLES TO BE FIXED IN CORE
AT A CHOSEN ADDRESS, THUS GIVING ACCESS TO THE
EXTERNAL PAGE I/O ADDRESSES AT THE PASCAL LEVEL.
ALSO, MACRO CODE CAN BE INSERTED IN LINE WITH PASCAL CODE.

BENCHMARKS INDICATE THAT PROGRAMS COMPILED
BY ESI PASCAL WILL RUN APPROXIMATELY TWICE AS FAST AS SIMILAR
PROGRAMS COMPILED BY DEC FORTRAN IV, AND MANY TIMES FASTER
THAN PROGRAMS EXECUTED BY INTERPRETERS LIKE DEC BASIC.

ESI PASCAL HAS BEEN IN USE SINCE THE SUMMER OF 1975
IN LASER TRIMMING SYSTEMS BUILT BY ESI. IT IS NOW AVAILABLE
TO ALL PDP-11 USERS. IT IS A SUPERIOR LANGUAGE FOR DATA
PROCESSING AND EDUCATION. AS EXTENDED BY ESI. IT HAS BECOME
AN UNEQUALLED TOOL FOR HARDWARE CONTROL APPLICATIONS.

PRICE OF THE SYSTEM IS $1500. THIS INCLUDES THE COMPILER,
THE SUPPORT MODULE, A CROSS REFERENCE DIRECTORY GENERATOR,
A SIMPLE TEXT EDITOR AND AN INSTRUCTION MANUAL.

ELECTRO-SCIENTIFIC INDUSTRIES
13990 NW SCIENCE PARK DR
PORTLAND ORE 97229

(503) 641-4141

(* David Rowland sent us the machine
retrievable user manual which accomp-
anied this page. It is an impressive
70+ pages long! *)

FOXBORO Fox-1

Warren R. Brown of the Foxboro Co., D.330, 38 Neponset Ave., Foxboro MA, 02038, phone (617) 543-8750 x2023, has written to us "In response to previous inquiries about the FOX-1 implementation of Pascal, we are currently formulating a statement for later publication."

FUJITSU FACOM 230-38   (an implementation exists)

FACOM 230-55   (an implementation is underway)

HEWLETT PACKARD HP-2100, HP-3000

THE UNIVERSITY OF SANTA CLARA · CALIFORNIA · 95053   November 8, 1976

TEL. 984-4482

Mr. Timothy Bonham
Pascal Implementations
University Computer Center
227 Experimental Engineering Building
University of Minnesota
Minneapolis, Minnesota 55455

Dear Mr. Bonham:

In response to your letter of October 22, I have taken over responsibility for Pascal implementation here at Santa Clara from Dan Lewis. There has been essentially no progress on implementation since the last contact with George Richmond in May of 1975.

Current plans are to initially implement Pascal via Pascal-P on the University's HP3000/Series II, which is running under MPE with 256 K words of memory. A very rough completion date is January, 1978 (we hope to beat this, but given the realities of implementor time availability, January is as good a guess as any).

Following completion of this task, we intend to implement a (still undefined) subset of Pascal for the Department's HP2100, running under DOS III with 32 K words of memory. The implementation will be in Pascal and cross-compiled from the 3000.

I'll keep in touch as the implementation progresses.

Incidentally, I've enclosed my PUG membership application.

Sincerely,

Ronald L. Danielson
Assistant Professor

RLD:dlm

Encl.

HITACHI HITAC 8800/8700   (see IBM 360/370 series)

HONEYWELL SERIES 6   (an implementation is being considered)

H316

A modified Solo (kernel) Concurrent Pascal interpreter is running on the Honeywell H316. For more information, write or phone Robert A. Stryk of Honeywell Corporate Research, home address: 5441 Halifax Lane, Edina MN 55424, office phone: (612) 887-4356.

6000, LEVEL 66 SERIES   (see also Newsletter #5)

University of Waterloo

August 25, 1976

Mr. Andy Mickel, Editor
Pascal Newsletter
University Computer Center
227 Experimental Engineering
University of Minnesota
Minneapolis, MN 55455

Dear Sir:

Having just read the PASCAL Newsletter Number 4, with its list of PASCAL implementations, I thought I should draw to your attention the PASCAL implementation for the Honeywell 6000 and Level 66 Series machines which we completed for Honeywell earlier this year. This compiler, an independent implementation of the full language which is not related to any previous PASCAL compiler, has been a commercial product of Honeywell Information Systems since May 1976. To my knowledge, it is the first PASCAL implementation to be officially distributed through and maintained by a major manufacturer.

Yours truly,
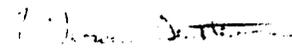
W. Morven Gentleman
WMG:cm   Director

(* note: manuals are available from Honeywell Sales *)

IBM SYSTEM 360/370   (this section also includes the Hitachi 8000 series
                       and the Amdahl 470. see also Newsletter #5)

PASCAL 8000 Implementation Note   November 5, 1976

1. **Implementer:**   Teruo Hikita and Kiyoshi Ishihata
                                   Department of Information Science
                                   University of Tokyo
                                   Tokyo 114 Japan
                                   phone 03-812-2111 ext 2947

2. **Machine:**   Hitac 8800/8700 (Hitachi)

3. **Operating system:**   OS7

4. **Distribution:**   (not yet)

5. **Documentation:**   "PASCAL 8000 Reference Manual"
                                   "Bootstrapping PASCAL Using a Trunk"
                                   (These technical reports are available
                                   from the Department)

6. **Maintenance policy:**   (not yet decided)

7. **Differences from Standard Pascal:**
                                   Standard procedures pack and unpack are
                                   not implemented.
                                   Files must be declared at main program
                                   level.
                                   A few novel language features are
                                   included.

8. **Characteristics of the compiler:**
                                   Written in Pascal (about 5200 source
                                   lines).
                                   Compiler object size is about 100
                                   kbytes.
                                   Compiling speed is about 350 line/sec.
                                   Execution speed is comparable to
                                   FORTRAN-compiled objects.

9. **Reliability of the compiler:** Good

10. **Method of development:**   Modified Naegeli's trunk compiler and
                                   bootstrapped it by Pascal-P
                                   (about three man-months).

RIKSHOSPITALETS EDB-AVDELING
UNIVERSITY HOSPITAL, OSLO; COMPUTER DEPARTMENT

Pilestredet 32
Oslo 1, Norway
Telefon: (02) 20 10 60

Ivar Dahl
Systems Manager

Oslo, 03.03.76

Pascal User's Group
c/o Andy Mickel
University Computing Center
University of Minnesota
USA

Dear Mr. Mickel,

We are currently implementing Pascal on our IBM 370/125
computer, based on the Pascal-P implementation kit.

I believe this implementation differs from other 370-versions
in two important ways, and therefore it might be of interest
to the Pascal User's Group.

1   The Environment

         The compiler is implemented on a 370 model 125 with
         256K bytes of main memory, under the DOS/VS operating
         system. DOS/VS is by far the most common 370-operating
         system, but it is mainly used in small installations
         where business-oriented processing is dominating.

2   The Purpose of the Implementation

         Our aim is to use Pascal in a production environment
         where the bulk of work is in the file-processing field.
         i.e. the administration (economic and other) of a
         large hospital.

         The only programming languages available are FORTRAN
         and COBOL, and since we have a good FORTRAN-milieu,
         FORTRAN is dominating, even in "pure" file-processing
         applications. Naturally we hope to replace them (at
         least partially) with Pascal. However, in spite of
         all the virtues of Pascal, it lacks the necessary
         facilities for our kind of applications. A number
         of features will have to be added to the language,
         and I will list a few which we consider to be most
         important.

Files

To do the file-processing you must have file-handling
routines, and since Pascal only supports sequential
file-structures, extensions will have to be developed.
Generally speaking, one might say that we have two
requirements:

- A way to explicitly control all types of secondary
  storage, like an interface to all available
  access-methods for data transfer.

- A way to define the file and its structure.

DOS/VS has no "file-manager" where Job-Control-Language
may be used to describe the file (record-length,
'blocking'-factor etc.). This forces us either to
make our own file-manager with a special control
language or to make modifications to Pascal's syntax.

We have chosen the first alternative, not because it
is the "best" solution, but rather because we want to
keep our version of the language compatible with the
standard.

External Procedures

It is not only the support for separate compilation
of Pascal procedures we consider to be important.
Far more important is the support for the inter-language
communication, to be able to call routines written
in other languages, whether they are user-written
application routines, library programs, sorting,
data-base- or data-communication software.

External Records

When external procedures are used, the data-transfer
between them will create problems since global variable
may not be accessed, and since the data-transfer through
parameters has certain limitations. We have therefore
introduced a new data type-External records.

An example:

```
type          exrec = record
                      ...
                ends external;

var           r : exrec;
```

Appending the new (reserved) word external to an ordinary
record type definition will cause all variables of that
type to be allocated as separate modules. The name of
each module is the variable-name. This allocation is
static, the variable of the example is not allocated on
the run-time stack, but the scope of the variables the
procedure where it is declared. It is perhaps best
understood by comparing it with FORTRAN's NAMED
COMMON (our main reason for implementing it), or
the STATIC EXTERNAL attribute in PL/I.

The features mentioned here are what we consider to be most
important for us to implement, we would very much want
comments on them, especially for users who (plan to) use
Pascal for file-processing applications, to learn how they
solve similar or other problems.

Yours sincerely

Ivar Labays

NEW MEXICO TECH
SOCORRO, NEW MEXICO 87801

COMPUTER CENTER

September 20, 1976

Mr. Andy Mickel
University Computer Center: 227 Exp Engr
University of Minnesota
Minneapolis, Minnesota 55455

Dear Andy,

We have been working on a PASCAL compiler for the 360/370 series for the past year. The compiler design was done by Dr. Jan V. Garwick, with implementation by Dr. Garwick, Paul Merillat and myself in PL360 and Chris Strachey's GPM. We are about 95% done, and it is a full compiler for PASCAL with the following exceptions:

1) GOTO's and labels are unsupported (and are flagged with a warning if used).

2) UNPACKED arrays are not supported.

3) Sets of characters are not allowed.

4) Tag field specifications in NEW and DISPOSE are ignored, the record is allocated with the maximum space needed.

5) Procedure or program segments each must not exceed 4K bytes.

6) A predefined procedure, CLOSE, has been added to facilitate file operations.

Extensive compile time and runtime error checking is done. The runtime checking is optional, and the compiler will generate runtime checks by use of a toggle which may be set or reset at any time during compilation. There are extensive compile time facilities, including a reformatter and cross referencer.

AN EQUAL OPPORTUNITY/AFFIRMATIVE ACTION EMPLOYER

Andy Mickel
September 20, 1976
Page 2

We have been testing the compiler for about a month now, and the results are good. Runtime facilities are still undergoing debugging, but should be completely done by September 31. We are going to use the compiler in our introductory C.S. course, and hope to unearth pathologies that would not come out in use by an experienced PASCALer.

Distribution of an OS version is expected to start by January 1st, and Dr. Tom Nartker of our C.S./Department will handle inquiries.

If anybody has any questions about the compiler (other than distributional), I would be glad to answer them.

We think the PASCAL newsletter is great, keep up the good work.

Sincerely,
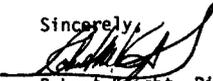
Robert Knight, Director
Office of User Services

RK:pq

STONY BROOK's PASCAL/360 - A STATUS REPORT - NOVEMBER 1976

The Stony Brook Pascal Compiler for IBM 360 and 370 computers is alive and well. As of November, 1976 more than 65 copies have been issued, and several installations are using the compiler under a live load. The compiler project continues at Stony Brook, and a second release is planned for January, 1977.

Release 1 was issued in June, 1976. It provides an almost complete implementation of Standard Pascal except for a variation in the means of specifying print field widths in the Write procedure. Not implemented in Release 1 are nonstandard files, and the standard procedure Dispose. The compiler has been successfully installed under the OS/MVT, MFT, VS1 and VS2 operating systems, and under VM/CMS with modifications to the OS interface. A DOS interface is nearing completion. At present, the main storage requirement is 160K bytes including space for file buffers.

The compiler is coded in XPL, with an assembler-coded monitor that provides the interface with OS. We do not have good statistics on compilation speed, but Release 1 has 1.93 CPU-second overhead to compile a trivial program on a 360/65. This is believed to be mainly due to the complexity of opening and closing files under OS/360.

The execution time of several compiled Pascal programs has been compared with that of equivalent translation into ALGOLW, whose compiler is known to produce good, though not optimized code. The Pascal programs execute faster, in nearly all cases.

Although it would be foolhardy to allege that a compiler that has been field-tested for less than six-months is bug-free, we believe that the majority of errors have probably been corrected. The three updates have repaired all errors reported as of November 1, 1976, as well as improving the resiliance of the compiler in the presence of Pascal source program errors, and reducing the storage requirements from 180 to 160K bytes. Updates are issued in the form of source-language (XPL or BAL) patches to be input to a card-oriented editor. Both the editor and an XPL compiler are furnished on the distribution tape.

Present work is directed toward completing the implementation of nonstandard files, management of heap storage, and external compilation, all of which will be included in Release 2. This release, subject to later updates to correct errors and improve performance, will be the production version of the compiler.

Future work will be directed toward producing an edition specialized for student use. This will offer the same capabilities in a compile-and-go version, except for a limitation on the size of programs that can

be compiled. The design target on the main storage requirement is 120K bytes. Compilation speed will be improved, primarily through the use of corefiles and interpass data communication buffers to reduce I/O. The compiler already includes excellent syntax error recovery, intelligible error messages and runtime diagnostics that enhance its usefullness in education.

For those interested in acquiring the Pascal/360 compiler, the cost is $175.00, which includes distribution, complete system documentation (when available), and maintenance at least through August, 1977.

A 50-page User's guide is available at a cost of $1.00 per copy in quantities of a dozen or more. The User's Guide is intended as a supplement to Jensen and Wirth, and tells everything that a user needs to know about the compiler.

At no cost whatsoever, one can obtain a packet giving additional information on the Pascal/360 compiler by sending a request to:

Pascal Compiler Project
Department of Computer Science
SUNY at Stony Brook
Stony Brook, New York 11794

17 September 1976

Andy Mickel
PASCAL User's Group
University Computer Center
227 Experimental Engineering
University of Minnesota
Minneapolis, MN 55455

Dear Andy,

I was quite surprised to see my last letter to you printed
in the Newsletter. Nevertheless, since it did appear, I feel
compelled to follow up on my comments about the Stony Brook PASCAL
compiler for the IBM S/370.

At the time I wrote the letter, the compiler was, indeed, buggy.
However, response from them has been excellent. I have since received
and installed two updates; the cover letter with the second stated
that it fixed all reported bugs. I have since run at least one
medium-to-large (700 statements) program using it, with no trouble.
And the post-mortem histogram — showing how many times each statement
was executed — is a most useful feature.

Complaints about the compiler? Sure, there's always something
that could be improved. The compiler is a bit too big (180K), and
a bit too slow for small programs (high fixed overhead per compilation),
and, perhaps most serious, they omitted the standard formatted-write
notation. And it would be nice if the compiler wrote out standard
OS-format object modules.

I should note that I ordered the Stony Brook compiler in
preference to the Manitoba version, since it seems more suited to
use with production-quality programs. Particularly serious restrictions
(from my point of view) in the Manitoba compiler are its lack of I/O,
its lack of a full version of NEW, and its restriction on the size of
procedures (4K).

—Steve Bellovin

cc: William Barabash, SUNY at Stony Brook

Telephone 919-94.. 5021

UNIVERSITY OF MINNESOTA
TWIN CITIES

University Computer Center
227 Experimental Engineering Building
Minneapolis, Minnesota 55455
(612) 373.... 6-7290

September 24, 1976

Dear Steve,

I felt a twinge even as I was putting your letter into the last
newsletter. I feel that the interest that other persons had in your opinions
outweighed the fact that I gave you no warning that it would be printed. I'm
glad you wrote a follow up letter and sent a copy of it to SUNY Stony Brook.
And I'm glad that their compiler is working better, also. Funny, we struggle
so hard just to get tidbits of information.

As I guess you can tell from Newsletter #5, we are trying to push hard to
repair the confusion about Pascal implementations.

So, thank you very much for writing. I'll of course print your letter
in Newsletter #6.

Sincerely,

irep

Pascal User's Group
C/O Andy Mickel

University Computer Center
227 Exp Engr
University of Minnesota

Minneapolis, MN 55455

Saint-Martin-d'Hères le 4 Novembre 1976

Monsieur,

En réponse à votre lettre du 25 Ocotbre 1976 voici le point des travaux
faits sur le compilateur Pascal.

- Il est opérationnel sur

              360/67   avec OS/MVT

              370/148   avec VS/MFT

- Demande REGION 220 K pour s'autocompiler.

- Distribution sur bandes magnétiques 9 pistes/800 bpi.

- Il existe un supplément au manuel du langage Pascal, décrivant l'implé-
  mentation sur IBM.

- Langage Pascal accepté est conforme au standard 74 à quelques exceptions
  près.

  Il manque Read/Write mais l'installation est prévue pour la fin 1976.

- Améliorations successives sont obtenues par compilation.

- La vitesse d'exécution moyenne est :

              . compilateur standard

                  6000 lignes sources _   105 secondes CPU

     360/67
              . compilateur "dopé"

                  6000 lignes sources _   84 secondes CPU

...

2.

- Ajouts non standards :

  . Cf. manuel spécification 360

  . procédures assembleur.

- Le compilateur Pascal a aussi été installé en CP/CMS.

Je vous prie d'agréer, Monsieur, l'expression de mes sentiments
distingués.

P.O J.P. FAUCHE

Olivier Lecarme of the Université de Nice, Laboratoire D'Informatique,
Parc Valrose, 06034 Nice Cedex, wrote us in a letter of 16 Sep 76:
    "A Pascal compiler for the IBM 360, which was probably the first one,
has been done in one of the Universities of Grenoble. Unfortunately, the
people who made it had no time nor support for distributing it, although
it seems to have impressive performances in execution time (but less good
in storage needed for compilation). People to contact are Messrs. Henneron
and Tassart (Informatique & Mathematiques Appliquees, B.P. 53, 38041
Grenoble-Cedex, France."

IBM 1130

    Olivier Lecarme of the Université de Nice, Laboratoire D'Informatique,
Parc Valrose, 06034 Nice Cedex, in his letter of 16 Sep 76:
    "Implementations for Pascal-P, Pascal-S and finally full Pascal have
been done for the IBM 1130 and are in use at the University of Neuchatel
(Centre de Calcul, Chantemerle 20, CH-2000 Neuchatel, Switzerland)."

ICL 1900   (an implementation exists)

   2970   (an implementation is underway)

INTEL 8080   (we need more implementation information)

INTERDATA 7/16

   Rod Steel of Tektronix, Inc., MS 60-456, P.O. Box 500, Beaverton, OR 97707
reported: "If we can find the resources, we may bring up a P-compiler on the
Interdata 7/16 at TEK."

   Michael S. Ball of the Naval Undersea Center, San Diego, CA 92132, wrote
in his report on the Univac 1100 implementation that the Center has cross
compilers, running on a Univac 1110 and generating machine code for the
Interdata 7/16, for both Concurrent Pascal and Sequential Pascal. See his
report for more information.

INTERDATA 70   (no known implementations)

MICRODATA 800   (no known implementations)

MITSUBISHI MELCOM 7700   (an implementation exists)

MOTOROLA 6800

PASCAL Implementations
University Computer Center
University Computer Center
227 Experimental Engineering
University of Minnesota
Minneapolis, MN 55455

Gentlemen:

   This letter is in response to your letter dated October 25 in
which you requested PASCAL Implementation information. Following
are my responses to each of your ten points.

   1. Implementor, maintainer:
       Before Nov. 26: Mark D. Rustad
                        Moorhead State University
                        Computer Center
                        1104 7th Ave. s.
                        Moorhead, MN 56560
       After Nov. 26:   Mark D. Rustad
                        585 Harriet Ave.
                        Apt. #213
                        St. Paul, MN 55112
       As yet there is no distributer.

   2. The implementation is specifically designed for the
       Motorola 6800-based MITS Altair 680b, but can easily be
       transported to any 8-bit machine (the Zilog Z-80 would
       be highly recommended).

   3. Since implementation is not complete, precise information
       is not yet available, however, the compiler will definitely
       run on an 8-bit microprocessor with 32K bytes, a TTY and
       no disk capability. It is likely that the memory requirement
       will be somewhat under 32K.

   4. No distribution since implementation is not complete.

   5. No documentation available yet.

   6. This compiler is, more or less, my hobby so a specific
       maintence policy cannot be stated.

7. This implementation is of a subset of PASCAL which I call PASCAL-M (PASCAL for microprocessors). Due to the very limited resources of microprocessors, PASCAL-M does not include the following PASCAL features:
   a. no files - all I/O via READ, WRITE
   b. no REAL type
   .. no declared scalar types
   d. no variant records
   e. no LABEL section
   f. no GOTO statement
   g. no WITH statement
   h. no FOR statement (use WHILE instead!)
   i. no CASE statement (may be put back in)
   j. no run-time checks yet
   k. standard procedures are: READ, WRITE, NEW, RELEASE, READLN, WRITELN, ORD, CHR, EOLN, MARK

   It is possible that the final implementation will have the CASE statement reinstated and that I may produce additional implementations for those having more resources to include REAL and FILE types.

8. The compiler produces an intrepretive code which is output onto an external medium such as paper tape which is then loaded with the intrepreter for execution. The compiler is written in the subset of PASCAL which it compiles and is about 2200 lines of code. The compiler should compile useable programs in under 32K bytes. The compilation and execution speeds can not yet be tested.

9. The reliability of the compiler seems to be excellent.

10. PASCAL-M was developed from PASCAL-P2 and is being cross-compiled by Mike Ball's UNIVAC 1100 PASCAL. I would estimate that about two man-months have gone into this implementation and I expect that about one more man-month to complete it. I have found the PASCAL compiler much easier to work on and understand than I expected and I believe that this is attributable to the language it is written in (PASCAL).

I will be preparing both documentation and reports on this implementation of PASCAL for publication once implementation is completed. For your information, all that remains is to debug the M-CODE (what I call my interpretive code, like P-CODE) interpreter.

Sincerely,

Mark Rustad

Mark Rustad

NCR CENTURY 100, 200, 300   (no known implementations)

PHILLIPS P-1400   (a non-standard implementation exists)

PRIME P-400

Phillip H. Enslow of the School of Information and Computer Science, Georgia Tech, Atlanta, GA 30332, has informed us that Georgia Tech is boot-strapping a compiler for the Prime P-400 using Pascal-P4. The P-400 is a large "mini" with a 32 bit word, and 512 million words of hardware supported virtual memory for each of 64 possible users.

SEL 8600   (an implementation exists)

SIEMENS 4004/157

H.-J. Hoffmann of the Fachbereich Informatik, Techn. Hochschule, Steubenplatz 12, D-6100 Darmstadt, Germany, wrote us: "We have implemented PASCAL P2 in three different versions (fully interpretive, SC-code auto-matically translated to assembly language, code emitters for assembly language) for SIEMENS 4004/157 computer. Usage in some systems programming work."

TELEFUNKEN TR-440   (an implementation exists)

TEXAS INSTRUMENTS TI-ASC   (an implementation exists)

TI-980A   (implementations exist)

UNIVAC 1100 SERIES

NAVAL UNDERSEA CENTER
San Diego, California 92132

2 November 1976

Mr. Andy Mickel
University Computer Center
227 Exp Engr
University of Minnesota
Minneapolis, MN 55455

Dear Mr. Mickel:

Thank you for the Pascal newsletter. I just got number 5 and enjoyed it greatly.

As you know, we have a quite complete implementation of Pascal for the Univac
1100 series. I have enclosed some performance data on the compiler and generated
code which you may find of interest. We kept the implementation as close as
possible to standard pascal, with extensions only to allow interface to the
Univac Exec, and for compatibility with other systems whose code we wanted to
use. The restrictions are essentially the same as those in the CDC compiler.
We have been using the Pascal compiler for about nine months, and its reliability
has been quite good. It should soon approach excellent.

We are using the compiler for general purpose programming and "systems" programming
for the Univac and other machines. Its usage is steadily increasing, and is
currently about 60 to 70 compilations a day. This compares to Fortran which
is about 400, but of course, each Fortran subroutine is counted separately; that
response has been quite favorable, and the interface to the user is at least as
good as the rest of the language processors available for the 110 series. A
data on the cost of the language processors installed here (the reaction) is
large, but unknown, percentage of the use is interactive (demand mode in Univac
terms).

One major use of the system has been the development of compilers for Concurrent
Pascal and Sequential Pascal for an Interdata 7/16. These compilers are based
on those supplied by Per Brinch Hansen, and generate machine code for the 7/16.
They are currently operating as cross compilers, running on the Univac 1110 and
generating code for the Interdata. We are currently in the process of moving
them to the Interdata for self-compilation. The project has been a very
interesting exercise in machine independence, and the code which must be changed
when moving the compilers from the Univac 1110, a 36 bit 1's complement machine,
is surprisingly small. We have not measured it accurately, but it is on the
order of one to two percent.

These compilers are highly optimizing compilers, and the direct machine code
which they generate is up to twenty percent smaller than the interpretive version
generated by the original compilers. Since there was no attempt to make the
interpretive code compact, this is not surprising. The next project along these
lines is to modify the compiler to generate code for the Interdata 8/32.

One problem which we have is keeping up to date on various extensions and changes
to the CDC compilers. As you mentioned in the newsletter, this compiler has
served as an unofficial standard for compatibility, and we would like to know
about things like the "VALUE" section before we see them in some code from another

---

installation. Perhaps this data could be published in the newsletter as it
becomes available? Since we are promoting the language as leading to portability,
we should practice what we preach.

Finally, where can I obtain copies of the new documents from Wirth's group. I
am particularly interested in the paper on Pascal-S.

Sincerely,

Michael S. Ball

PERFORMANCE OF THE PASCAL 1100 SYSTEM   14 October 1976

The following performance was measured on a Univac 1110. All times given
are totals, including both CAU time and CCER time.

1. Compiler Performance.

The compiler performance was measured as it compiled itself. The
compiler is 7,494 lines of code, including comments and blank lines. It
compiles into 34,879 words of code and literals. The library adds 5,917
words (including some data area), for a total of 40,787 words. The Univac
compiler interface routines account for 4,888 words of the library. The
data space allocated for the compiler is 16,108 words, and while compiling
itself the compiler uses 8,060 words in the heap and 7,444 words in the
stack.

The compilation rate is 105 lines per second with an output listing,
and 118 lines per second without a listing.

2. Compiled Code Performance.

The compiled code was compared with that generated by the NUALG and
ASCII FORTRAN processors. For both Pascal and NUALG, tests were done both
with and without run-time checks. The FORTRAN compiler never generates
run-time checks, but does allow for three different levels of optimization.
The normal mode provides no optimization, and optional modes provide local
and global optimizations. The local optimization mode was chosen as the
standard of comparison, since the short test programs which were used
provide an unusually simple case for the global optimizer, and allow it to
perform much better than would be expected for the average program.

The programs used as a basis for comparison were taken from Wirth's
paper on the design of a Pascal compiler. They are all programs which are
easily written in all three languages, and so do not use the expressive
power of Pascal. In addition, the time taken to call a simple procedure
with four value parameters were measured for each processor. The results,
with times reflecting wait conditions for all processors, are summarized in the following tables.

	PASCAL	NUALG	FORTRAN
Time	26.4	108.6	21.9
Rel	1.21	4.96	1.00

Table 1. Procedure Call Times.

	PASCAL		PASCAL NO CHECKS		NUALG		NUALG NO CHECKS	
	Time	Rel	Time	Rel	Time	Rel	Time	Rel
PART	9.36	0.62	9.17	0.61	12.88	0.85	12.67	0.84
PARTNP	1.10	1.18	0.99	1.06	3.06	3.29	2.95	3.17
SORT	24.61	1.37	20.22	1.12	32.92	1.83	26.81	1.49
MATMUL	18.70	1.82	14.69	1.43	21.02	2.05	17.46	1.70
COUNT	4.99	0.30	4.69	0.28	12.15	0.72	11.13	0.66

	FORTRAN		FORTRAN LOCAL OPT.		FORTRAN GLOBAL OPT.	
	Time	Rel	Time	Rel	Time	Rel
PART	15.10	1.0	15.10	1.00	14.94	0.99
PARTNP	0.87	0.94	0.93	1.00	0.79	0.85
SORT	18.01	1.00	18.01	1.00	10.56	0.59
MATMUL	10.27	1.00	10.26	1.00	4.04	0.39
COUNT	16.88	1.00	16.83	1.00	16.40	0.97

Table 2.

The program listed on the left side of Table 2 are:

PART   compute the additive partitions of a number (30 in this case) and print the results. This uses recursion for Pascal and NUALG, and a hand simulated stack for FORTRAN.

PARTNP the same as above, but with no printing

SORT   sort an array of 1,000 numbers by a bubble sort

MATMUL matrix multiply of two 100 by 100 matrices

COUNT   count the characters in a file and print the number of times each occurs. The file was 124,000 characters long.

M. S. BALL

VARIAN 620   (no known implementations)

V73

California State University, Chico
Chico, California 95929

Department of Computer Science
(916) 895-6442

November 2, 1976

Mr. Timothy Bonham
Pascal Implementations
University Computer Center
227 Experimental Engineering Building
University of Minnesota
Minneapolis, MN   55455

Dear Mr. Bonham:

Thank you for your interest in our activities at California State University, Chico.

Due to some staff changes, our Pascal project has not been completed. The implementation is planned on a Varian V73. Pending the completion of hardware changes, this project will remain stagnant for at least another year.

Sincerely,

Orlando S. Madrigal, Ph.D.
Chairman & Professor
Department of Computer Science

OSM:lt

XEROX SIGMA 6, SIGMA 7, SIGMA 9   (see also CII 10070)

Olivier Lecarme, Universite de Nice, Laboratoire D'Informatique, Parc Valrose, 06034 Nice Cedex, France, in his letter of 16 Sep 76:
"A complete and standard compiler for the Xerox Sigma 6,7 and 9 has been done by Pierre Desjardins, who can give you all desirable information. Anyway, it seems to be a very good implementation, especially in the domain of compatibility and conformity to the standard."
(* We do not have Pierre Desjardins's correct address, can someone help? *)

PASCAL USER'S GROUP

USER'S

GROUP

ALL PURPOSE COUPON

* * * * * * * * * * * * * * * * * * *

Clip, photocopy, or reproduce, etc. and mail to: Pascal User's Group
c/o Andy Mickel
University Computer Center
227 Exp Engr
University of Minnesota
Minneapolis, MN 55455

(phone: (612) 376-7290)

/ / Please renew my membership in / enter me as a member of the PASCAL USER'S GROUP for the next / current Academic Year ending June 30. I shall receive all 4 issues of *Pascal Newsletter* for the year. Enclosed please find $4.00. (*When joining from overseas, check the *Newsletter* POLICY section for a PUG "regional representative".*)

/ / Please send a copy of *Pascal Newsletter* Number _____. Enclosed please find $1.00 for each. (*See the *Newsletter* POLICY section for issues out of print.*)

/ / My new address is printed below. Please use it from now on. I'll enclose an old mailing label if I can find one.

/ / You messed up my address. See below.

/ / Enclosed are some bugs I would like to report to the maintainer of the _____ version of Pascal. Please forward it to the appropriate person so that something can be done about it.

/ / Enclosed please find a contribution (such as what we are doing with Pascal at our computer installation), idea, article, or opinion which I wish to submit for publication in the next issue of *Pascal Newsletter*.

/ / None of the above. _____

_____

_____

_____

Other comments: From: name _____

address _____

_____

_____

_____

phone _____

date _____

(*Your phone number helps facilitate communication with other PUG members.*)