

PASCAL USER'S GROUP

Pascal News

(FORMERLY PASCAL NEWSLETTER)

NUMBERS 9 AND 10 (COMBINED ISSUE)

COMMUNICATIONS ABOUT THE PROGRAMMING LANGUAGE PASCAL BY PASCALERS

SEPTEMBER, 1977

TABLE OF CONTENTS

0	POLICY: <u>Pascal News</u>
1	ALL PURPOSE COUPON
3	EDITOR'S CONTRIBUTION
4	HERE AND THERE
4	News
8	Conferences
8	Books and Articles
11	Past Issues of <u>Pascal Newsletter</u>
11	PUG Finances
12	Roster
39	ARTICLES
39	"Pascal at Sydney University" - A. J. Gerber and C. C. Morgan
40	"Disposing of Dispose" - Stephen P. Wagstaff
42	"What is a Textfile?" - William C. Price
43	"Generic Routines and Variable Types in Pascal" - B. Austermuehl and H.-J. Hoffmann
47	OPEN FORUM FOR MEMBERS
54	Special Topic: Micro/Personal Computers and Pascal
58	Special Topic: Pascal Standards
60	IMPLEMENTATION NOTES
60	Checklist
60	General Information
61	Software Writing Tools
61	Portable Pascals
63	Pascal Variants
64	Feature Implementation Notes
73	Machine Dependent Implementations
113	POLICY: Pascal User's Group

POLICY: PASCAL NEWS (77/09/01)

- * Pascal News is the official but informal publication of the User's Group.

Pascal News contains all we (the editors) know about Pascal; we use it as the vehicle to answer all inquiries because our physical energy and resources for answering individual requests are finite. As PUG grows, we unfortunately succumb to the reality of (1) having to insist that people who need to know "about Pascal" join PUG and read Pascal News - that is why we spend time to produce it! and (2) refusing to return phone calls or answer letters full of questions - we will pass the questions on to the readership of Pascal News. Please understand what the collective effect of individual inquiries has at the "concentrators" (our phones and mailboxes). We are trying honestly to say: "we cannot promise more than we can do."

- * An attempt is made to produce Pascal News 4 times during an academic year from July 1 to June 30; usually September, November, February, and May.
- * ALL THE NEWS THAT FITS, WE PRINT. Please send written material for Pascal News single spaced and in camera-ready form. Use lines 18.5 cm wide!
- * Remember: ALL LETTERS TO US WILL BE PRINTED UNLESS THEY CONTAIN A REQUEST TO THE CONTRARY.
- * Pascal News is divided into flexible sections:

POLICY - tries to explain the way we do things (ALL PURPOSE COUPON, etc.).

EDITOR'S CONTRIBUTION - passes along the opinion and point of view of the editor together with changes in the mechanics of PUG operation, etc.

HERE AND THERE WITH PASCAL - presents news from people, conference announcements and reports, new books and articles (including reviews), notices of Pascal applications, history, membership rosters, etc.

ARTICLES - contains formal, submitted contributions (such as Pascal philosophy, use of Pascal as a teaching tool, use of Pascal at different computer installations, how to promote Pascal, etc.

OPEN FORUM FOR MEMBERS - contains short, informal correspondence among members which is of interest to the readership of Pascal News.

IMPLEMENTATION NOTES - reports news of Pascal implementations: contacts for maintainers, implementors, distributors, and documentors of various implementations as well as where to send bug reports. Qualitative and quantitative descriptions and comparisons of various implementations are publicized. Sections contain information about Software Writing Tools for a Pascal environment, Portable Pascals, Pascal Variants, Feature Implementation Notes, Machine Dependent Implementations, etc.

- * Volunteer editors are:

Andy Mickel - editor

Tim Bonham and Jim Miner - Implementation Notes editors

Sara Graffunder - Here and There editor

John Strait and John Easton - Tasks editors

David Barron and Rich Stevens - Books and Articles editors

Rich Cichelli - Software Tools and Applications editor

George Richmond - past editor (issues 1 through 4)

USER'S

GROUP

ALL PURPOSE COUPON

(77/09/01) •

Pascal User's Group, c/o Andy Mickel
University Computer Center: 227 EX
208 SE Union Street
University of Minnesota
Minneapolis, MN 55455 USA

+ Clip, photocopy, or
+
+ reproduce, etc. and
+
+ mail to this address.

// Please enter me as a new member of the PASCAL USER'S GROUP for ___ Academic year(s) ending June 30 _____. I shall receive all 4 issues of Pascal News for each year. Enclosed please find _____ (\$4.00 for each year). (* When joining from overseas, check the Pascal News POLICY section on the reverse side for a PUG "regional representative." *)

// Please renew my membership in PASCAL USER'S GROUP for ___ Academic year(s) ending June 30 _____. Enclosed please find _____ (\$4.00 for each year).

// Please send a copy of Pascal News Number(s) _____. (* See the Pascal News POLICY section on the reverse side for prices and issues available. *)

// My new ^{address} _{phone} 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} _{phone}. See below.

// Enclosed please find a contribution (such as what we are doing with Pasca] at our computer installation), idea, article, or opinion which I wish to submit for publication in the next issue of Pascal News. (* Please send bug reports to the maintainer of the appropriate implementation listed in the Pascal News IMPLEMENTATION NOTES section. *)

// None of the above. _____

Other comments: From: name _____
mailing address _____

phone _____
computer system(s) _____
date _____

(* Your phone number aids communication with other PUG members. *)

JOINING PASCAL USER'S GROUP?

- membership is open to anyone: particularly the Pascal user, teacher, maintainer, implementor, distributor, or just plain fan. Memberships from libraries are also encouraged.
- please enclose the proper prepayment - we will not bill you.
- please do not send us purchase orders - we cannot endure the paper work! (if you are trying to get your organization to pay for your membership, think of the cost of paperwork involved for such a small sum as a PUG membership).
- when you join PUG anytime within an academic year: July 1 to June 30, you will receive all issues of Pascal News for that year unless you request otherwise. You will receive a membership receipt.
- please remember that PUG is run by volunteers who don't consider themselves in the "publishing business." We consider production of Pascal News as simply a means toward the end of promoting Pascal and communicating news of events surrounding Pascal to persons interested in Pascal. We are simply interested in the news ourselves and prefer to share it through Pascal News (rather than having to answer individually every letter and phone call). We desire to keep paperwork to a minimum because we have other work to do.

JOINING THROUGH "REGIONAL REPRESENTATIVES" ?

- anyone can join through PUG(USA) - address on reverse side. International telephone: 1-612-376-7290. PUG(USA) produces Pascal News and keeps all mailing addresses on a common list. Regional representatives collect memberships as a service and reprint and distribute Pascal News using mailing labels sent from PUG(USA) which speeds up delivery overseas.

European Region (Europe, North Africa, Middle and Near East):

send £2.50 to: Pascal Users' Group (UK)
c/o Computer Studies Group
Mathematics Department
The University
Southampton SO9 5NH
United Kingdom
telephone: 44-703-559122 x700

Australasian Region (Australia, New Zealand, Indonesia, Malaysia):

send \$A10 to: Pascal Users Group (AUS)
c/o Arthur Sale
Dept. of Information Science
University of Tasmania
GPO Box 252C
Hobart, Tasmania 7001
Australia
telephone: 23 0561

RENEWING?

- please renew early (before August) and please write us a line or two to tell us what you are doing with Pascal, and tell us what you think of PUG and Pascal News to help keep us honest. To save PUG postage, we do not send receipts when you renew.

ORDERING BACKISSUES OR EXTRA ISSUES?

Our unusual policy of automatically sending all issues of Pascal News to anyone who joins within an academic year (July 1 to June 30) means that we eliminate many requests for backissues ahead of time, and we don't have to reprint important information in every issue - especially about Pascal implementations!

- Issues 1, 2, 3, and 4 (January, 1974 - August, 1976) are out of print.
- Issues 5, 6, 7, and 8 (September, 1976 - May, 1977):
 - Less than 40 copies each remain at PUG(USA) available for \$2 each.
 - Less than 20 copies each remain at PUG(UK) available for £1 each or £2.50 for 6,7,8.
 - None available at PUG(AUS): write to PUG(USA) or PUG(UK).
- Extra single copies of new issues are \$2 each - PUG(USA); £1 each - PUG(UK); and \$A3 each - PUG(AUS).

SENDING MATERIAL FOR PUBLICATION?

(such as ideas, queries, articles, letters, opinions, notices, news, implementation information, conference announcements and reports, etc.) "ALL THE NEWS THAT FITS, WE PRINT." Please send written material for Pascal News single spaced and in camera-ready form. Use lines 18.5 cm wide! Remember: ALL LETTERS TO US WILL BE PRINTED UNLESS THEY CONTAIN A REQUEST TO THE CONTRARY.

MISCELLANEOUS INQUIRIES? Please remember we will use Pascal News as the vehicle to answer all inquiries and regret to be unable to answer individual requests.



UNIVERSITY OF MINNESOTA
TWIN CITIES

University Computer Center
227 Experimental Engineering Building
Minneapolis, Minnesota 55455

(612) 376-7290

Here is another potpourri of topics:

Pascal Newsletter #8

"Green on green" was not our idea (neither was the thick paper - it destroyed our poverty image!). It was a giant disappointment to have worked so hard on #8 and see it come out this way. We agree with the 20 or so people who gently suggested that "we say it in black and white." We were faced with wasting paper and making the newsletter 3 weeks late if we reprinted, or sending it out. We sent it out and were reimbursed by the printer for the extra postage and heavy paper costs. PUGN 8 from the UK was over 2 months late due to circumstances beyond their control, but it was black on white!

Pascal Jobs

Who says you can't get a job "in the real world" using Pascal? Herb Rubenstein, the first research assistant to work for us at the University Computer Center who learned Pascal before he learned FORTRAN, picked Colorado as a place to live when he graduated with a B. Sci. in Computer Science from the University of Minnesota and then he began job hunting. In 2 months he landed a job with a rapidly growing engineering peripherals firm, AutoTrol, and is working almost exclusively with Pascal.

Also see the OPEN FORUM section for a letter from Neil Barta.

New Australasian Distribution Center for PUG

To solve problems with slow mail to Australia (as well as currency exchange), Arthur Sale, prolific PUGN contributor at the University of Tasmania, has kindly set up a distribution center this summer (winter) much like Judy Mullins and David Barron did for Europe a year ago.

The area served is Australia, New Zealand, Indonesia, and Malaysia. We at PUG(USA) are confused about why the price is so high; apparently we were to receive a letter from Arthur over 2 months ago with the details, but it was lost in the mail. Other details are on the reverse side of the ALL PURPOSE COUPON.

Computer Companies Using Pascal

It is past time to print a list we've been keeping of computer companies who are seriously using Pascal. This is so we all can argue back that "Pascal is being used for serious real world work" when accused otherwise!

Total conversion internally to the company:

Texas Instruments, January, 1977 ("from micros to super computers")

Harris Data Communications, March, 1977 ("Pascal is our language - replacing FORTRAN and COBOL" - Tom Spurrier.)

Companies Using Pascal for future software systems:

Cray Research (CRAY-2)

Control Data Corporation (Cyber 270 series)(They have already been using it for the 2550 and the Cyber 18)

DATA 100 Corporation (model 78)

Companies marketing Pascal as a user product:

Honeywell; Computer Automation; Four Phase Systems; Varian Data Machines (Sperry Univac).

New Developments - Micro/Personal Computers

Several PUG members took my request seriously to write to several of the personal computing journals to promote Pascal over BASIC (see Editor's Contribution PUGN 8). David Mundie, George Cohn, and Tim Bonham have written letters. At Frank Brewster's and Rich Cichelli's urging, I sent personal letters and a free copy of #8 (the only free copies we had given) to the editors of 14 computing journals. We received warm responses from half a dozen. Also we've been getting new members from their readership, some who are so curious to know about Pascal that they are dying to get this issue of PUGN! I'm really encouraged at these developments because these computers represent the future and we have an early start (unlike on the current dinosaur systems).

See the OPEN FORUM section.

Pascal News

We changed the name to avoid confusion by people who think a newsletter is 4 pages long. This issue is a combined one because it contains so much material - and it is also late. We had to revise nearly everything: the cover, the coupon, policy, and do a summary for the implementation notes! This has good side effects because PUGN 8 was late in Europe, and renewals have been slow to come in. Next issue will be in February. Deadline for material is the last day in December: (77/12/31).

New Policies

Look at all the new editors! Please read the revised policy pages on the inside covers (front and back). The major change is that we are declaring that we are tired of processing purchase orders and answering requests for information "about Pascal" from people who won't join PUG and read Pascal News. It may sound strange, but we print everything we know about Pascal in Pascal News.

Back Issues

It is really difficult to plan ahead on backissues with a growing membership. Nevertheless we made it through last year with some extra copies of each issue. But we incurred some tremendous distribution problems which caused unjustified delays in sending back issues to people who joined PUG after mid-February. I apologize, and hope that we have learned enough from our mistakes to do better this year.

Membership

We began collecting PUG memberships on 76/03/03. Here are some interesting membership totals: 317 on 76/08/13 (#5 to press); 368 on 76/09/09 (#5 mailed); 516 on 76/11/14 (#6 to press); 560 on 76/12/10 (#6 mailed); 598 on 76/12/29 (#7 to press); 644 on 77/01/13 (#7 mailed); 943 on 77/04/26 (#8 to press); 984 on 77/05/12 (#8 mailed); 1095 on 77/06/30 (end of year); 1306 on 77/09/07 as I write this (759 active). We have 211 new members and 560 renewals since 77/07/01 with renewals still rolling in.

PUG Finances

I last printed information in PUGN 6. Last year (our first) we promised and delivered 4 issues of PUG Newsletter. What we did not know was how popular PUG was going to be. We also delivered a few things we did not promise: 230 copies of backissue #4, mass mailings to get to new and old people, letters to implementors to get compiler information and unfortunately, slow service to late joiners (sorry, but we wish you had joined earlier).

See the HERE AND THERE section for details under "PUG Finances". We show a small loss - almost exactly 1% - and our crude accounting knowledge doesn't account for all the back issues produced with 76-77 money and sold in 77-78 (since July 1, we have sold 243 at PUG(USA) alone. So I claim we did okay.

Andy - 77/09/07.

Editor's Contribution

Here and There With Pascal

(* Here are extracts from almost all of PUG's mail. To reiterate what we've said elsewhere, many of the inquiries we get are answered in previous issues. If you are a member, please try to find answers to your questions from Pascal News before you write to us. If you aren't a member and you want information that's in an issue that's already out, we'll tell you to join rather than to answer each inquiry with a personal letter. *)

Attn: Production Automation Project, Univ. of Rochester Dept. of Elec. Engr., Rochester, NY 14627 (Aristides Requicha): "I also would appreciate any information you might have on the existence and availability of reliable and efficient Pascal compilers for the PDP-11/40. We normally use the RT-11 operating system." (* 77/6/28 *)

Attn: Centro Ciencias de la Computacion, Universidad Catolica de Chile, Casilla 114 D, Santiago, Chile: "Is there any FULL PASCAL implementation for the IBM 370?" (* 77/6/7 *)

Bill Barabash, Dept. of Computer Sci., State Univ. of NY, Stony Brook, NY 11794: "Yes. I want to be the first one on my block to RENEW my membership in the Pascal User's Group. I enclose a check for \$4.00 which entitles me to issues 9-12 of the newsletter plus a Captain Pascal secret decoder ring which glows in the dark. . . ." (* 77/6/8 *)

Philip N. Bergstresser, 128 Jackson Ave., Madison, AL 35758: "TRW has a PASCAL program on the CDC 7600 and TI-ASC with 40000 statements and 1100 procedures, REVS, the Requirements Engineering and Validation System, supporting interactive color graphics, CALCOMP plotting, and a relational data base. We have implemented a complete 7600 PASCAL system." (* 77/8/22 *)

Gus Bjorklund, 2250 Coppersmith Square, Reston, VA 22091: "I am presently working on a Pascal compiler for the IBM Series 1, and should be finished in September 1977." (* 77/6/22 *)

Kenneth Bowles, P.O. Box 1123, Rancho Santa Fe, CA 92067: "Looks like we will be working with CONDUIT on getting a (Standard) ANSI BASIC running under our PASCAL system. Object: entice Basic users over to PASCAL by making a switch very convenient. This will be the only truly portable BASIC we know of." (* 77/6/22 *)

Bill Brennan, 39 Jody Drive, Norristown, PA 19401: "I am presently engaged in implementing PASCAL for Sperry-Univac 9000 computers. (This activity is for my education mostly, not for release.) I certainly could use the information your newsletter will provide. For your information, I heard of the PASCAL user's group from a notice in 'Creative Computing.'" (* 77/9/1 *)

Arthur A. Brown, 1101 New Hampshire Ave. NW, Washington, DC 20037: "I am a professional translator of Russian mathematics, and will be glad to abstract the Proceedings of the All-Union Symposium on Implementation Techniques for New Programming Languages. (* We sent them off right away, but just received word from Arthur Brown that an English translation has been published as Vol. 47 of Springer-Verlag's Lecture Notes in Computer Science. *) (* 77/6/10 *)

Thomas W. Burnett, Computer Center, Dickinson College, Carlisle, PA 17013: "What PASCAL is available for a PDP-11 running RSTS?" (* 77/6/30 *)

Edwin J. Calda, Dept. E152, AAI Corp., P.O. Box 6767, Baltimore, MD 21204: "Would appreciate information concerning the availability of Pascal for the SEL 8000 series or SEL 32." (* 77/7/19 *)

Patrick Chevaux, DEC, Quai Ernest Ansermet 20, B.P. 23, CH-1211 - Geneva 8, Switzerland: "I am urgently looking for a PASCAL compiler running on PDP-11 under RSX-11M operating system, and I wonder if you know about such a product. If so, could you please give me a few indications about it, as well as the person to contact and perhaps how to obtain it." (* 77/7/11 *)

D. Michael Clarkson, DBMS Research and Development, California Software Products, 525 N. Cabrillo Park Dr., Suite 300, Santa Ana, CA 92701: "My company is currently involved in implementing a lot of high-level transportable system software using PASCAL." (* 77/6/27 *)

Kurt Cockrum, 3398 Utah, Riverside, CA 92507: "R. A. Lovestedt should get in touch with Tom Payne, Math Dept., University of California at Riverside, Riverside, CA 92507 for information on HP-3000 implementations of Pascal. I believe that John Hayward of UCR has written a P-code interpreter that runs on the 3000. "Are there any HOBBYISTS doing anything with Pascal? Most of us can't handle tapes (except paper) and some of us are poor." (* 77/6/6 *)

John Collins, 3M Co., Bldg. 235-F247, St. Paul, MN 55101: "We are considering using PASCAL as a Systems Implementation Language for microprocessor based systems, using a PDP11 as a host for cross-compilation and system monitoring." (* 77/6/13 *)

Larry Crane, EDS, 1200 Locust, Des Moines, IA 50309: "Thanks for sending us the PUG newsletters, hopefully we'll be able to get ahold of something good. If not we'll just have to develop it. With luck we'll have an operating system in Pascal. To the bit bucket with Fortran, even COBOL will be overcome. Long Live the Computocracy." (* 77/5/16 *)

(* Response to Andy's letters to personal computing publications has been heartening, if somewhat humorous at times. In Creative Computing, for example, the "Pasacal" User's Group was mentioned, but the address got lost in the press. Nonetheless, high school student Steven Trapp, 5020 Mulcare Drive, Columbia Heights, MN 55421, deduced the address from Andy's name and the name of the building and wrote to ask for an all-purpose coupon. *)

Jack Crone, Systems Analyst, USC School of Medicine, Hoffman Res. Ctr., Rm. 805, 2025 Zonal Ave., Los Angeles, CA 90033: (* From his letter which we saw in Byte, May 1977. *) "At present, supporting a full blown high level language compiler is quite an achievement for a personal computer; supporting several is out of the question. For this reason it is important to make the best possible selection and to select some obscure educational vernacular such as PASCAL because it is esthetically more pleasing, and [sic] would leave personal computing where it is right now: a lot of hardware with very little software."

Kenneth A. Dickey, 1662 Stromberg, Arcata, CA 95521: "I am especially interested in [Pascal] applications dealing with environmental modeling, approximations, simultaneous equations, and text editing." (* 77/7/11 *)

John Dickinson, Dept. of Elec. Engr., Univ. of Idaho, Moscow, ID 83843: "I would also like to ask your help in locating a good implementation of PASCAL for a IBM 370 machine. I understand there are many such implementations and my question for you is which is best for a student environment. I plan to use PASCAL in a beginning computer science class and so I would like a version that is easy to use and one that has clear error messages." (* 77/6/30 *)

Jim Elam, 150 Lombard, No. 601, San Francisco, CA 94111: "I would be interested in information on usage in a production environment and efficiency of generated code on 370 gear?" (* 77/6/2 *)

Gary Feierbach, Advanced Studies Dept., Inst. for Advanced Computation, P.O. Box 9071, Sunnyvale, CA 94086: "We currently have Pascal upon our KI-10 and plan to put it up on several other machines including a version on the ILLIAC IV." (* 77/6/24 *)

Charles N. Fisher, Academic Computing Center, Univ. of Wisconsin, 1210 West Dayton Street, Madison, WI 53706: "We may have a proposal for PL/1 - like varying length strings for for you in the next few months - it appears to extend PASCAL fixed length strings rather nicely. Also, I'll be in Minneapolis for a Univac User's Meeting in mid-October. If its convenient, I may be able to stop by and talk some PASCAL with you (I'll be heading a PASCAL "birds of a feather" session at the meeting). (* 77/8/30 *)

Dan Fylstra, 22 Weitz St. C, Boston, MA 02134 (* To put this letter in context: Dan is an editor/consultant for Byte *): "Initially I plan to write an article explaining the

features and strengths of Pascal, aimed at the BASIC-oriented beginning programmer or casual user. But I'll certainly include notes on the status of Pascal implementations and especially their availability on micros (since the news is so good).

"You can invite people to write or call me if they have late-breaking news that deserves a wider audience than the User's Group itself. Since everyone connected with Byte is enthusiastic about Pascal, articles, new product announcements, and material for "Byte's Bits" or the "Technical Forum" are always welcome. These should be sent to Byte's regular address in Peterborough." (* 77/8/22 *)

Richard Gemeinhardt, Jr., Discipledata, Inc., 110 S. Downey, Indianapolis, IN 46219: "Please advise if Pascal operates on any NCR hardware--such as NCR Century 201 or NCR Criterion." (* 4/25/77 *)

James D. George, Computer Branch, Underwater Sound Reference Division, Naval Research Laboratory, P. O. Box 8337, Orlando, FL 32806: "The Naval Research Laboratory has several PDP-11s, and is using RSX11M and RSX11D. I would be very much interested in finding out more about PASCAL under RSX11, and would appreciate any leads you could provide." (* 77/5/17 *)

Roger Gulbranson, Dept. of Physics, Univ. of Illinois, Urbana, IL 61801: "Even though I know you don't like it, you can add my name to the list of people who want an OTHERWISE (or whatever) clause added to the CASE statement. I particularly liked George Richmond's article. I'm not sure I agree with all the things he said, but most of his points seem reasonable. I'm not sure I agree with his point about partial L->R evaluation of boolean expressions. While I'll admit it will help some problems concerning array indexes and the like I'm finding out that the FTN (* CDC FORTRAN *) method of logical if evaluation (i.e., convert the whole mess into a logical (or boolean) result) and subsequent jump on true/false is faster on machines like the [Cyber] 175 and probably also the 76. Considering the trend toward faster hardware, it may not be a good idea to explicitly demand partial evaluation.

* * *

"I agree with Legenhausen's comment about pushing PASCAL in the appropriate micro computer journals. Maybe the way to do it is to develop a standalone PASCAL compiler for a paper tape based system with no more memory than 8K (16K if you must) and then distribute it for a nominal fee--say \$10 or \$15. And no, I don't have the time to do it." (* 77/6/6 *)

George E. Haynan, 556 Parker Rd., W. Melbourne, FL 32901: "Many maintainers who arbitrarily change Pascal at their sites are guilty of the NIH (Not Invented Here) syndrome: 'If I haven't thought of it then it isn't any good.'

"I'm interested in Sequential Pascal, directly compilable, for the PDP-11, with an RT-11/RSX-11 operating system." (* 77/5/25 *)

Carl Helmers, BYTE Publications, 70 Main St., Peterborough NH 03458: "A couple of comments about the Zilog rumor. All the information came from the same source and later proved premature. At the IEEE Computer Society Asilomar conference this year, a Zilog representative could not confirm Pascal as a programming model for advanced architectures, but hinted strongly that research in the direction of instruction sets optimized for high level languages such as Pascal is being performed. A talk in the lobby of the West Coast Computer Faire's convention hotel with one of Motorola's LSI designers strongly hinted of the possibility of built in microcode for language constructs in the next generation of integrated circuits.

A strong suggestion: people involved with the implementation of languages should seek out LSI design engineers in order to inject ideas about appropriate features to be built into the designs of future microprocessor products. (* 77/6/20 *)

Richard Hendrickson, Cray Research Inc., 7850 Metro Parkway, Suite 213, Minneapolis, MN 55420: "Keep up good work. Articles like the one by Barron and Mullins in No. 7 will do wonderful job of keeping FORTRAN and eliminating PASCAL as major computing language." (* 77/5/23 *)

Sam Hills, 3514 Louisiana Ave. Pkwy., New Orleans, LA 70125: "I am interested in developing a subset of PASCAL to run on a hobby-type microcomputer such as the Altair

or IMSAI, and any information you could supply would be greatly appreciated. (* 77/6/5 *)

Tao-Yang Hsieh, VIDAR, 77 Ortega Ave., Mountain View, CA 94040: "I am considering implementing Pascal on our HP2100 system and would appreciate very much if you could assist me in obtaining a copy of Pascal P-code compiler and a copy of Pascal compiler written in Pascal." (* 77/8/1 *)

Jon F. Hueras, Dept. of Information and Comp. Sci., Univ. of Calif., Irvine, CA 92717: "I'm . . . working for Univac on the side. . . . We would find life a whole lot easier if we had a reasonable file comparison program to work with. You wouldn't happen to know of anyone who's written one in Pascal, would you? Please let us know." (* 77/7/26 *)

Alfred J. Hulbert, Inhalation Toxicology Res. Inst., P.O. Box 5890, Albuquerque, NM 87115: "We are working with John Barr of Hughes Aircraft to get Brian Lucas' NBS PASCAL written in PASCAL for RSX-11 users of DEC PDP-11's (along with real time and character string extensions)" (* 77/6/22 *)

Geoffrey Hunter, Chemistry Dept., York Univ., Downsview M3J 1P3, Ontario, Canada: "Thanks for your memo of 77/05/24. I am of course familiar with Pascal and actually taught a course one year using Wirth's book "Systematic Programming: an Introduction." I used "Algol" rather than PASCAL, Simula, Algol 68, etc. for the Waterloo talk, because it is, as you note, the ancestor of all current structured programming languages.

On first acquaintance I was an enthusiast for PASCAL, but after some practical experience, and after reading Habermann's article in Acta Informatica Vol. 3 (1973) p. 47., I have some reservations about some features of the language. Especially the lack of block structure (environment structuring--as distinct from control structures and procedures in particular), and the lack of dynamically dimensioned arrays, are, it seems to me conceptually oversights of the language. PASCAL's strong point is, of course, data structuring.

* * *

There is a danger with any organisation such as PUG--that it becomes the defendent of a fixed particular definition and implementation of the language. Guard against this. . . ." (* 77/6/1 *)

Aron K. Insinga, 126 Dupont Hall, University of Delaware, Newark, DE 19711: "We are interested in using Pascal under UNIX (and DEC-supported operating systems) as well as on micro-processors (the LSI-11, Motorola M6800, and Intel 8080, in particular) with compilation and assembly done on the larger PDP-11 system. (* 77/7/29 *)

Mitch Jolson, SSRFC, 25 Blegen Hall, Univ. of MN, Minneapolis, MN 55455: "It may interest PUG members to know that the LEAA (Law Enforcement Assistance Administration), a division of the Justice Department, requires, by legally enforceable regulation, that all criminal justice software be in ANSI FORTRAN or ANSI COBOL." (* 77/8/18 *)

Matti Karinen, and Jyrki Tuomi, Compiler Project, Room 2113, Computer Center, Tampere Univ. of Technology, PL 527, 33101 Tampere 10, Finland: "We would appreciate information about PUG and the Pascal Newsletter, especially as we have in mind to implement Pascal on our PDP 11/70." (* 77/8/17*)

Barbara I. Karkutt, Box 942, Easton, PA 18042: "Am interested in the Pascal compiler for the Z-80 microcomputer." (* 77/6/6 *)

Doug Kaye, DuArt Film Labs Inc., 245 W. 55th Street, New York, NY 10019: "I anxiously await Newsletter #9 with writeups about PASCAL on Data General gear." (* 77/7/21 *)

Ed Keith, Citrus College, 18824 E. Foothill Blvd., Azusa, CA 91702: "Please send data on availability of compilers, assemblers etc. I have a XEROX 560, IMSAI 8080, SWTPC 6800." (* 77/4/28 *)

Thomas J. Kelly, Jr.: 120 East Street Road, C3-9, Warminster, PA 18974: "I am interested in obtaining a Pascal compiler for any Burroughs computer; especially for the B5500, B6700, or B7700." (* 77/5/16 *)

Here and There With Pascal

Peter Klauber, c/o Hamburgische Electricitaets-Werke, Ueberseering 12, D-2000 Hamburg 60, Germany: "My intention to use PASCAL, is to introduce the philosophy of structured programming to out commercial COBOL-programmers. For this reason the PASCAL must be able to communicate with normal IBM datasets.

"My question to you is: Do you know a working PASCAL compiler for our IBM 370/158 SVS?" (*77/6/16*)

Jerry LeVan, Dept. of Math. Sciences, Eastern Kentucky Univ., Richmond, KY 40475: "I would like to know if anybody has PASCAL running under RSTS/E on a PDP 11/70 (or 45 etc.)" (* 77/5/2 *)

Donald Lindsay, Dynalogic Corporation Ltd., 141 Bentley Ave., Ottawa, Ontario, Canada K2E 6T7: "I am interested in M6800 Pascal. I have an incomplete implementation of Brinch Hansen's Sequential Pascal. Due to the press of other work, I would be just as happy to purchase a compiler. (It would have to be commercially viable.)" (* 77/6/22 *)

David Lippincott, Information Control Systems, 313 N. First Street, Ann Arbor, Michigan 48107: "We are a computer typesetting firm upgrading to an as yet unknown machine. We will be writing an operating system so any information of similar applications would facilitate my attempts at convincing others that Pascal would be a good choice." (* 77/7/23 *)

R. A. Lovestedt, 20427 SE 192, Renton, WA, 98055: "Will soon be starting a P4 interpreter on HP3000." (* 77/5/24 *)

Tim L. Lowery, Applications Prog. Group, 110 Love Building, Computing Center, Florida State Univ., Tallahassee, FL 32306: "We are very interested in acquiring a Pascal implementation for 8080 development, since Pascal is the favorite and dominant language among the computer science students." (* From a letter to PUG member Peter Zechmeister, 77/7/20 *)

Bruce Mackenzie, Computervision Corporation, 201 Burlington Road, Route 62, Bedford, MA 01730: "We will be implementing PASCAL-P4 on Data General NOVA's and NOVA compatible machines, running under our own operating system. We will also be using Zilog's Z80 in the near future, PASCAL has been mentioned for it. Do you know of anyone planning to implement PASCAL for the Z80?

"I found a little bit of information for you: Ted Park of Loma Linda, California has a PCODE interpreter and assembler written in (Data General) ECLIPSE assembly language and running under RDOS. It took them about a month of work. Ted said he would write you directly." (* 77/8/9 *)

Ian MacMillan, P.O. Box 128, Mount Royal, Quebec, Canada H3N 2T6: "We are running Pascal under NOS (* CDC 6000 operating system *). How do you get that interactive?" (* 77/4/28 *)

Mark T. Marshall, 18229 Topham St., Reseda, CA 91335: "I am going to be using the COMPUTER AUTOMATION LSI 4/90." (* 77/8/29 *)

Jim McCord, Systemetrics Inc., 120 E. de la Guerra Street, Santa Barbara, CA 93101: "I'm a hobbyist with an LSI-11 (PDP-11-03) with dual floppies. If anybody knows of a version of Pascal that will run on this machine, I'd like to hear about it, (especially if it's cheap). (* 77/9/7 *)

Brian Meekings, Dept. of Computer Studies, Univ. of Lancaster, Bailrigg, Lancaster, England, UK LA1 4YX: "I took advantage of the fact that we have an enthusiastic Pascal faction here to collect some subscriptions. (* NINE were enclosed. *) Incidentally, is there a student subscription rate--some of our undergraduates may well be interested." (* There isn't, but where else can a student get a student rate that is much cheaper? *) (* 77/5/18 *)

C. A. Miller, Nuclear Research Centre, Dept. of Physics, Univ. of Alberta, Edmonton, Alberta, T6G 2N5, Canada: "Our computing equipment giving rise to my interest in PASCAL consists of three DATA GENERAL Eclipses." (* 77/6/8 *)

David Miller, 11203A Avalanche Way, Columbia, MD 21044: "Please sign me up for the PASCAL User's Group. I've been so busy developing PASCAL (relocatable, for DEC 11/45)

and an application system, I failed to notice the Group has grown so much. Finally got to reading some SIGPLAN notices, and ran across your letter." (* 77/5/22 *)

Carlton Mills, 203 North Gregory, Urbana, IL 61801: "We are working on Pascal compiler for micro-processors. It is a highly optimized cross compiler running on the B6700 (Burroughs). Currently I am looking for venture capital to get it on the market. I will let you have more details when we are ready to announce it." (* 77/8/22 *)

J. Misra, Dept. of Computer Sciences, Painter Hall 3.28, Univ. of Texas at Austin, Austin, TX 78712: "I would appreciate receiving any information about Pascal implementation on NOVA computers.

"Our department has recently acquired two NOVA's for which we wish to get the compilers. The size of P-compiler would probably make it prohibitive for the NOVA's. If you know of any existing implementation, please send us the information." (* 77/8/29 *)

Tom Moberg, Academic Computing, Grinnell College, Grinnell, IA 50112: "We are looking for a PASCAL system which will run on our PDP 11/70 (RSTS/E)." (* 77/6/7 *)

Gerald Nadler, RBMS Research Center, Brandeis Univ., Waltham, MA 02154: ". . . I was hoping that a list was a available of Pascal implementations on machines other than CDC and PDP-10's." (* 77/8/18 *)

Brian Nelson, Computer Services, 2801 W. Bancroft Street, U. of Toledo, Toledo, OH 43606: "I am trying to locate a Pascal compiler for use on a PDP 11/70 and a PDP 11/40." (* 77/6/2 *)

John W. Nunnally, Harding College, Box 744, Searcy, AR 72142: "Harding College has just ordered a PASCAL compiler from Oregon Museum of Science and Industry (OMSI). It is a modified version of ESI's implementation that is supposed to run under RSTS/E Version 6B (with the RT-11 emulator). We will let you know how it goes." (* 77/5/25 *)

Carol Anne Ogden, Software Technique, Inc., 100 Pommander Walk, Alexandria, VA 22314: "I am preparing some material for publication on PASCAL for micros in my capacity as Consulting Editor of Mini-Micro Systems and EDN." (* From a note to PUG member Peter Zechmeister, 77/6/15. *)

Shmuel Peleg, Computer Science Center, University of Maryland, College Park, MD 20742: "Do you know of any PASCAL compilers working under UNIX?" (* 77/8/28 *)

Lee Potts, DARCOM ALMSA, Attn.: DRXAL-TL, P.O. Box 1578, St. Louis, MO, 63188: "My agency is planning to try Pascal as a systems implementation language on IBM 360 and several minicomputers of varying architecture. Pascal's main attraction to us now is systems portability. (* 77/9/1 *)

Walter F. Prautsch, Albertinenestrasse 29, D-1000 Berlin 37, Germany: "I would like to mention that I am working in the field of system-simulation (methodology, applications in the field of urban and regional planning). If you know any people using PASCAL for the development of simulation-systems (event-oriented as well as continuous), please let me know their addresses." (* 77/6/10 *)

Bruce K. Ray, Polymorphic Computer Systems, P. O. Box 3581, Boulder, CO 80303: "I am interested in developing a PASCAL compiler for use with the NOVA-series computer, and am therefore interested in anything and everything which may help me in the task. Is there a PASCAL written in a mini-PASCAL (subset) which is available which would be easier to bootstrap, and if so, who, how, where, and how much." (* 77/8/16 *)

Harlan R. Ribnik, P.O. Box 3182, Boulder, CO 80307: "I am a graduate student in Computer Science at the University of Colorado working on an implementation of a PASCAL to JANUS compiler. I was informed by someone I met on the CDC PLATO system that I might be able to get some information from you regarding the PASCAL Users' Group." (* 77/8/19 *)

Bo Rojder, AB Atomenergi, Fack, 611 01 Nykoping 1, Sweden: "AB Atomenergi is a research and development center for nuclear and other energy forms. At our data center we have a CDC CYBER 172 with 131 K memory, and NOS 1.2 operating system. We plan to install Pascal on it and hereby apply for membership in Pascal User's Group, as individuals or as an organization, whichever the policy of PUG is." (* 77/8/22 *)

Peter Rouschmayer, Luitpold-Gymnasium, Seeaustrasse 1, D-8000 Munchen, Germany: "We got: A PDP 11/34 with 64 kWords Core, 2 Disks RK05, a LA180 Lineprinter and 7 VT50 screens. RSTS/E Release 6B, BASIC+.

"We ought to: teach Informatics to our pupils aged 10 to 20.

"We would like to get: a PASCAL-Compiler, interactive if possible, running in RSTS if possible.

"Can you help us?" (* 77/4/2 *)

Bernie Rosman, Math/CS Dept., Framingham State College, Framingham, MA 01701: "I'm trying to get (CDC 6000) Pascal 2 for Mass. State College Computer Network (Cyber 72,73). Currently, we have Pascal-Release 1 update 11 which has some bugs; e.g., SQRT doesn't work (fixed by MSCCN). Also: we're now using Pascal in data structures and CS II (2nd semester-freshman) courses. We have, however, not yet switched to Pascal in CS I. Finally, we hope to install Pascal on our new PDP-11/34." (* 77/5/31 *)

David J. Rypka, Dept. of Computer and Info. Science, 2036 Neil Ave. Mall, Columbus, Ohio 43210: "I am an active user of a DEC-10 version and would like to find other versions and documentation for the DEC-10." (* 77/14/6 *)

Carlos Scheel, Depto de Sistemas, Instituto Tecnologico de Monterrey, Sucursal J, Monterrey, Mexico: "We would like to have the compiler of the PASCAL system; please mail me back all the information and prices, manuals, etc." (* 77/8/8 *)

Barry Searle, TowerC Floor 10C, Transport Canada, Section TASX, Place de Ville, Ottawa, Ontario K1A 0N8, Canada: "The Canadian Dept. of Transport will be converting to Pascal on PDP-11 equipment." (* 77/8/25 *)

David Segal, 111 Third Ave. BK, New York, NY 10003: "I am planning to get a microcomputer and would like to implement something more useful than BASIC for it to think in. I first heard about Pascal while trying to track down information on another decent language, BCPL. In my BCPL search I talked to Art Evans and Bob Morgan at Bolt, Beranek and Newman in Cambridge, Mass. From them I gathered that BCPL compilers aren't so easy to come by on small machines, but that Pascal is implemented on several PDP-11's. That was heartening since the microcomputer I'm most interested in is a PDP-11 look-alike with respect to instruction set. . . . If you happen to know of any already existing Pascal implementations on a microcomputer, or anybody working on one, please let me know about it." (* 77/8/18 *)

Bruce Seiler, UCLA Dept. of Chemistry, Los Angeles, CA 90024: "I am interested in the implementation of PASCAL on microprocessor based systems." (* 77/5/23 *)

Michael Settle, 751 Washington, No. 115, Arlington, TX 76011: "I have a confession to make--I don't have any idea what PASCAL is. I work with the huge Insane Business Monsters and tinker with my own Altair. There has been so much discussion of PASCAL in Dr. Dobb's Journal during the past year, that I finally broke down and wrote you. Please enter a subscription to your newsletter for me, and send me the details about your PUG." (* 77/8/15 *)

David Elliot Shaw, Structured Systems Corp., 343 Second St. - Suite K, Los Altos, CA 94022: "You are performing a welcome service for the community of Pascal users, implementors, fans. . . . On the accompanying sheet we describe (as compactly as possible) the STRUCTURED SYSTEMS PASCAL-SS compiler for the PDP-11." (* 77/7/12 *)

Jeffrey G. Shaw, P.O. Box 2678, Menlo Park, CA 94025: "Could you direct me to an individual or group that might have a Pascal compiler for the 8080 or 280 micros?" (* 77/8/18 *)

Evan L. Solley, The Life Support Systems Group, Ltd., 2432 NW Johnson, Portland, OR 97210: ". . . Also enclosed is a write-up and sample listing for a PASCAL cross-referencer we developed some time ago. It is an extension of Wirth's PCREF, which we find much more usable. Its symbol tables are currently set up to process ESI Pascal (V5.5) for RT-11, but can be easily modified for use with other compilers. The program is licensed and distributed in ASCII source form for a fee of \$25.00. Distributable media include magtape (9 track 800 bpi), DECTape, RK05 cartridge, or card deck (800 cards). Media should be provided by Licensees. RT-11 users will additionally

receive a special executable version, with CSI and GCL interface to version 3 of KI-11 and LSSG's RT-11X extension of version 2C." (* 77/4/23 *)

Tom Spurrier, Electronics Systems Division, Harris Corp., P.O. Box 37, Melbourne, FL 32901: "Harris Corp. headquarters has issued a corporate directive that Pascal is our language. There are over 100 computer centers in the corporation. It will be used for systems level development initially and then in applications areas." (* 77/6/21 *)

John P. Stallings, Tymshare, Corporate Offices, 20705 Valley Green Drive, Cupertino, CA 95014: "Once again I find myself potentially involved in a project concerning Pascal and have decided that it is past time for me to associate myself with an appropriate source of information.

"Could you tell me how to go about joining the Pascal User's Group, and if possible, how to obtain a list of available Pascal compilers for the PDP-11?" (* 77/7/18 *)

Rod Steel, MS 60-456, Tektronix Inc., P.O. Box 500, Beaverton, OR 97077: "I have a partially debugged version of Mike Ball and Co.'s Concurrent Pascal cross-compiler for the Interdata 7/16 running on our DEC K110 (translated from Sequential Pascal to the lower case version of PASREL)." (* 77/5/31 *)

W. Richard Stevens, Kitt Peak National Observatory, P.O. Box 26732, Tucson, AZ 85726: (* What follows is extracted from an article Richard wrote for the Kitt Peak Computing Newsletter *) "The PASCAL language, because of features designed into it, has the ability to detect programming errors that would be undetected by any FORTRAN system. I have personally found that this feature alone cuts in half the time needed to develop a new program." (* 77/1/3 - The article mentions other features of PASCAL which make it useful at Kitt Peak. *)

Peter Sumner, Interdata Computers Pty. Ltd., 30 Kings Park Rd., West Perth, Western Australia 6005: "I was delighted to discover the existence of your User's Group as there are a number of interested Interdata customers in Australia. In fact, a Pascal compiler is currently under development at the University of Melbourne, Dept. of Computer Science." (* 77/5/3 *)

Markku Suni, Computer Centre, Univ. of Turku, SF-20500 Turku 50, Finland: "Since I am interested in Pascal and have spent some nice hours kitbashing our Pascal compiler, I would like to join in. . . . We have here a PDP-11 with KA processor, 128Kw of core, 2 RPO3 discs, one TU10 mag tape unit, card reader, line printer, and usual sortiment of terminals." (* 77/4/28 *)

Rodney Thayer, Central Research Group, P.O. Box 451, Harvard, MA 01451: "A few people in my area (myself included) are investigating R. E. Berry's U. of Lancaster PASCAL for the Data General NOVA. If I am closer than England for somebody, they are welcome to write to me to find out about Lancaster PASCAL." (* 77/7/7 *)

Mike Tiller, 2501 N. Lancaster Ln. No. 178, Plymouth, MN 55441: "Interested in Pascal for NOVA/ECLIPSE." (* 77/7/14 *)

Martin Tuori, Behavioral Sci. Div., Defence and Civil Inst. of Environmental Medicine, P.O. Box 2000, Downsview, Ontario, M3M 3B9, Canada: "We will be running ESI Pascal under RSX11M, as soon as ESI has it ready." (* 77/7/26 *)

Univ. of Texas at Austin: (* The statistics from their newsletter indicate that Pascal and Pascal 2 accounted for 5% of their total use in March 1977. *)

James A. Vellenga, System Development, Data 100 Corp., Box 1222, Minneapolis, MN 55440: (* He reports that there is a class in Pascal at Data 100. Ten to fifteen people were enrolled. Nine memberships came from people at the company. *)

Kenneth R. Wadland, Computer Science Dept., Fitchburg State College, Fitchburg, MA 01420: "Although I have not used PASCAL much, I have become quite interested in it from talking to Professor Bergeron of the University of New Hampshire. (He has been modifying a DEC System-10 compiler written in Germany.)

"I intend to teach PASCAL in my Data Structures course and later in my Systems Programming course on a CDC Cyber 72. As a teaching device, I think it is far superior to any of the other standard languages." (* 77/6/29 *)

Walter Wehinger, Pfaffenwaldring 64, Rechenzentrum Uni Stuttgart, D-7000 Stuttgart 80, Germany: "We are running PASCAL 6000.3.4 modified by T. A. Nemeth Uni Adelaide, so we have only minor INTERCOM problems (e.g., EOL-definition). We switched over to NOS/BE 1.0 L.420 without problems."

David H. Welch, P.O. Box 721, Colton, CA 92324: "In the August issue of 'Microcomputer SCCS Interface' the existence of your group, its quarterly newsletter, and dues of \$4.00/yr were mentioned. I'm interested in learning more about Pascal and I think your newsletter might be useful." (* 77/9/2 *)

Richard West, Small Terminal Engineering, Comterm Ltd., 147 Hymus Blvd., Montreal 730, Quebec, Canada: Our software team has decided to change over to using PASCAL to write our systems packages. . . . I would like to have copies of . . . back issues so that we can find the most economical way of obtaining PASCAL for our PDP-11 DOS system and for a variety of microprocessors." (* 77/6/20 *)

Hans-Wilm Wipperman, Univ. of Kaiserslautern, Pfaffeuberstr. 95, D-6750 Kaiserslautern, Germany: "German Chapter of ACM is planning a meeting concerned with PASCAL. . . . I shall inform you about details later on." (* 77/5/20 - See the Conferences section for details of the 77/10/14 meeting. *)

Louis F. Wojnarowski, Mental Health Research Institute, University of Michigan, Ann Arbor, MI 48109: "I am interested in implementing Pascal on my Prime-300. I would like to get more information on the hypothetical stack machine code (Pascal-P I believe) and any macro generating systems before I attempt to order a particular tape from PUG." (* 77/6/27 *)

Joan Zimmerman, MUMPS Users' Group, Biomedical Computer Laboratory, 700 South Euclid, St. Louis, MO 63110: "I have never heard of any other group obsessed with a single language apart from ours: we are all involved with MUMPS as described in the enclosed Pocket Guide (additional copies \$1) and Book of MUMPS (additional copies \$2). "We have about 250 paying members (\$25 annual fee), but about 5000 people on our mailing list. A member has asked me to find out for him if anyone has written MUMPS in PASCAL. If you know of anyone who has, or could query your members about this, I would appreciate any positive information." (* 77/8/22 *)

Karl L. Zinn, Center for Research on Learning and Teaching, University of Michigan, 109 East Madison Street, Ann Arbor, MI 48109: "I am working on uses of PASCAL in personal computing as well as in intro courses." (* 77/8/15 *)

CONFERENCES

German Chapter of the ACM, a meeting on Pascal.

(* This is rather late notice, but we'll hope that interested members will at least be able to attend the conference, if not submit papers. *) Meeting October 14, 1977 in Kaiserslautern. Papers will include such subjects as "Implementations," "Pascal in Schools," "Applications," and "Pascal and Microcomputers." For more information get in contact with G. Nees, German Chapter of the ACM, c/o Siemens AG, E 54, Mozartstr. 33/b, D-8520 Erlangen, Germany; or H.-W. Wippermann, Universitat Kaiserslautern, Informatik, Pfaffenbergstr., Gebaude 14, D-6750 Kaiserslautern, Germany. (* Our thanks to Hans-Wilm Wippermann for keeping us informed about the conference. We hope to have a report from the conference in No. 11. *)

Pascal Day or Pascal Workshop, McMaster Univ., Hamilton, Ontario, Canada.

(* From a letter from Nick Solntseff *) "I am starting to plan a 'Pascal Day' or a 'Pascal Workshop' to be held at McMaster on March 3, 1978. I will be getting in touch with the Regional ACM group and the IEEE Computer Society, to see if they want to sponsor it. I am thinking of asking for brief reports on implementations, use of Pascal for teaching, etc." (* For more information, write to Nick Solntseff, Dept. of Applied Mathematics, 1280 Main St. West, Hamilton, Ontario, Canada L8S 4K1; or call (416) 525-9140. *)

Report on IFIP conference, Aug. 8-12, 1977, Toronto.

(* Thanks for this report to Nick Solntseff *) "I did not have too much interest shown at IFIP in a meeting of PUG, but I am not really surprised as it was almost impossible to get in touch even with people one knew were at the conference.

"The computerized message system was terrible to say the least, but anyone interested should have seen my manual notice on the general notice board.

"In all, I gathered nine people over coffee in the hospitality lounge at various times, but decided that a more formal meeting was not called for."

Meeting of the Pascal sub-group, AFCET, Nice, France, June 13-14, 1977.

(* PUG member Olivier Lecarme, IMAN, University of Nice, Parc Valrose, F-06034 - Nice CEDEX, France, has sent us a bulletin, which he publishes regularly before meetings of the sub-group, of articles to serve as a basis of discussion for the meeting of the sub-group. We'll try to get word to you in advance of the next meeting, but in the meantime, if you wish to receive the bulletin and/or be notified of meetings, write to Olivier Lecarme. *)

Titles of Articles:

"The language Pascal as support for teaching introductory programming," R. Rousseau.

"The future of Pascal (extensions and standardization)," Andy Mickel.

"Some tools for users of Pascal at Rennes," l'equipe Simone.

"Simulator of machines in Pascal," D. Thalmann.

"Pascal/CII-Iris 80 and 10070," P. Maurice.

"Application of parallel algorithms to three simple problems," J. Bezivin, J. L. Nebut, and R. Rannou.

"One year of using the language Simone at Rennes: Judgment and perspectives," J. Bezivin, J. L. Nebut, and R. Rannou.

BOOKS AND ARTICLES

We've had no news from David Barron. Rich Stevens supplied us with one item. George Richmond's bibliography, which we didn't have room for in No. 8, appears separately. A price list for some formerly out-of-print documentation appears under IMPLEMENTATIONS

LANGUAGES

Brinch Hansen, Per, The Architecture of Concurrent Programs, Englewood Cliffs, NJ: August 1977, 366 pp., \$16.95. (Prentice Hall)
(* From the publisher's blurb *) ". . . detailed handbook showing you how to develop simple and reliable operating systems from scratch using Concurrent Pascal."

"Proceedings of the All-Union Symposium on Implementation Techniques for New Programming Languages," Novosibirsk 1975, English translation published by Springer-Verlag as Volume 47 of their Lecture Notes in Computer Science. (* PUG member Arthur Brown, who had offered to abstract the Russian, sent us news of the English translation in lieu of the abstract. We'll try to get more information for No. 11.

TEXTBOOKS

(* A Summary of all known Pascal textbooks, partly reprinted from newsletters 5-8 *)

Atwood, J. W., Standard Pascal, to be published. (* Note: we haven't heard anything new about this book. For more information, write to J. W. Atwood, Dept. of Comp. Sci., Sir George Williams Campus, Concordia Univ., Montreal, Quebec, Canada H3G 1M8. *)

Conway, Richard C., David Gries, and E. C. Zimmerman, A Primer on Pascal, Winthrop, 1976, 448 pp., paper, \$9.95.

An introduction to Pascal for non-programmers which in spite of its length fails to cover any data structures besides arrays. A rewrite of a book based on PL/C which still carries the smell of PL/I--a foreward stating the contrary notwithstanding.

Bowles, Ken (U. of Calif., San Diego), Introduction to Computer Science, to be published by Springer-Verlag in October 1977.

A complete introduction to Pascal for non-programmers using an interactive graphics approach and the Keller teaching method.

Kieburtz, Richard, Structured Programming and Problem Solving with Pascal, to be published by Prentice-Hall sometime in 1977. For more information, write Richard Kieburtz, Dept. of Comp. Sci., SUNY at Stony Brook, Stony Brook, NY 11794.
A rewrite of a book by the same name on PL/I.

Schneider, G. Michael, Steven W. Weingart, and David M. Perlman, Introduction to Programming and Problem Solving with PASCAL, New York: Wiley, to be published in January 1978. A camera-ready copy of the manuscript can be obtained by writing Gene Davenport, Editor, John Wiley and Sons Publishers, 605 Third Avenue, New York, NY 10016. The manuscript may, with written permission, be duplicated for class use until the publication date.

A complete introduction to Pascal for computer science majors.

Webster, C.A.G., Introduction to Pascal, Heyden, 1976. \$11.00, 5.50, DM35.00.

A book for beginning computer science majors which received a bad review in Pascal Newsletter No. 8 because, among other things, there are numerous errors and the old language definition was used.

Wirth, Niklaus, Systematic Programming: An Introduction, Englewood Cliffs, NJ: Prentice Hall, 1973, 169 pp., \$13.96.

(* From the preface *) "A book which introduces programming as the art or technique of formulating algorithms in a systematic manner, recognizing that it is a discipline in its own right." (* This introductory book only covers Pascal through arrays *)

Wirth, Niklaus, Algorithms + Data Structures = Programs, Englewood Cliffs, NJ: Prentice Hall, 1976, 366 pp., \$16.50.

(* From the cover *) "... lucid, systematic, and penetrating treatment of basic and dynamic data structures, sorting, recursive algorithms, language structures, and compilers."

IMPLEMENTATIONS

Price List on Reports of Interest--hard-to-get implementation information:

Through the courtesy of George H. Richmond and his co-workers Karin Bruce and Michele Dowd, reprints of some hard-to-get Pascal documentation is now available. Write to:

Karin and Michele--Pascal Distribution
Computing Center Library: 3645 Marine St.
Univ. of Colorado,
Boulder, CO 80309
or call (303) 492-8131.

(* These all can be ordered from North America at the price listed. All others must include overseas postage. *)

"Pascal-S, A Subset and its Implementation," 63 pages, N. Wirth, ETH, June 1975, \$6.50.
(* Includes an entire listing of a Pascal-S compiler/interpreter in Pascal. *)

"On Code Generation in a Pascal Compiler," 42 pages, U. Ammann, ETH, April 1976, \$4.00.
(* Description of the internal design and performance of Pascal-6000 *)

"The Pascal-P Compiler Implementation Notes," 65 pages, ETH, December 1974, revised July 1976 by K. V. Nori, et. al., \$5.50.

(* Describes the portable Pascal compiler and interpreter. *)

(* Letter received from David Barron - 7/7/75 *)

"I am sorry I have not been able to write earlier with news of the publication of the Proceedings of the Pascal Symposium. We had hoped that these would appear in the Springer-Verlag 'Lecture Notes in Computer Science,' but after an initial favourable reaction Springer delayed, and have finally declined to publish. However, I am pleased to be able to report that Wiley-Interscience have agreed in principle to publish the proceedings. I am currently discussing details with them, and hope to be able to give you firm details very shortly."

APPLICATIONS

Barth, Jeffrey M., "Shifting Garbage Collection to Compile Time," CACM, 20:7 (July 1977), pp. 513-519.

Algorithms expressed in Pascal.

Biedl, Albrecht, "An Extension of Programming Languages for Numerical Computation in Science and Engineering with Special Reference to Pascal," SIGPLAN Notices, 12:4 (April 1977), pp. 31-33.

A description of how to carry attributes of computation such as temperature, energy, fuel consumption, etc., and units expressing these attributes such as celsius, kelvin, joules, liters per km with numerical quantities used in scientific and engineering problems. This circumvents problems which arise in dealing only with pure (dimension-less) real numbers in current programming languages.

Brownlee, J. Nevil, "An ALGOL-Based Implementation of SNOBOL4 Patterns," CACM, 20:7 (July 1977), pp. 527-529.

Algorithms expressed in Pascal.

Bulman, David M., "Stack Computers," IEEE's Computer, May 1977.

Suggests that new machines introduced by semiconductor manufacturers may be called 'Pascal machines' instead of stack machines because the Pascal compiler generates code for a 'hypothetical stack machine' and manufacturers may start building machines, using LSI technology, like the hypothetical one.

Gries, David, and Narain Gehani, "Some Ideas on Data Types in High Level Languages," CACM, 20:6 (June 1977), pp. 414-420.

Algorithms expressed in Pascal.

Hueras, Jon, and Henry Ledgard, "An Automatic Formatting Programming for Pascal," SIGPLAN Notices, 12:7 (July 1977), pp. 101-105.

A larger description of the pretty printer announced as available for distribution in Pascal Newsletter 6, page 70. (* This and the other article below from the July issue were drawn to our attention by PUG member Harry M. Murphy. *)

Leventhal, Lance A., "Talk Your Computer's Language," Kilobaud, August 1977, pp. 34-38.

Mentions Pascal as one high-level language used on small computers, and urges readers to be aware of it.

Peterson, James L., and Theodore A. Norman, "Buddy Systems," CACM, 20:6 (June 1977), pp. 421-431.

Algorithm in Pascal.

Singer, A, J. Hueras, and H. Ledgard, "A Basis for Executing Pascal Programmers," SIGPLAN Notices, 12:7 (July 1977), pp. 101-105.

A set of guidelines for standard naming, formatting and commenting conventions in Pascal programs and why programmers should adhere to them.

Surden, Esther, "Software Thievery Cited as Thorny Hobbyist Problem," Computer World, June 6, 1977.

A report on the National Computer Conference, which lists Pascal as a programming language available on personal computers, but which says that there are few implementations of it so far.

Tennent, R. D., "Language Design Methods Based on Semantic Principles," to appear in Acta Informatica, 1977. (* Rich Stevens let us know about this one. *)

(* from the summary. *) "Two language design methods based on principles derived from the denotation approach to programming language semantics are described and illustrated by an application to the language Pascal. The principles are, firstly, the correspondence between parametric and declarative mechanisms, and secondly, a principle of abstraction for programming languages adapted from set theory. Several useful extensions and generalizations of Pascal emerge by applying these principles, including a solution to the array parameter problem, and a modularization facility."

BIBLIOGRAPHY

- Literature about the Programming Language Pascal January 1977
- George H. Richmond, The University of Colorado Computing Center
- Ammann, U., "The Method of Structured Programming Applied to the Development of a Compiler", "International Computing Symposium 1973", Gunther, et al., Eds., pp. 93-99 North Holland (1974)
- Ammann, U., "Die Entwicklung eines Pascal-Compilers nach der Methode des strukturierten Programmierens, ETH-Diss. 5456 (1975)
- Ammann, U., "On Code Generation in a PASCAL Compiler", Berichte des Instituts für Informatik, Nr. 13, ETH Zurich (April 1976)
- Bachmann, K. H., "Die Programmiersprachen Pascal und Algol 68", Akademie-Verlag, Berlin (1976)
- Burger, W. F., "Pascal Manual", Department of Computer Sciences, TR-22, The University of Texas at Austin (July 1973)
- Eurger, W. F., "BOBSW - A Parser Generator", Department of Computer Sciences, SESLTR-7, The University of Texas at Austin (December 1974)
- Bron, C., de Vries, W., "A Pascal Compiler for PDP11 Minicomputers", Department of Electrical Engineering, Twente University of Technology, Enschede, Netherlands (1974); SOFTWARE-PRACTICE AND EXPERIENCE -6-, 1, pp. 109-116 (January 1976)
- Conway, R., Gries, D., Zimmerman, E., "A Primer on PASCAL", Winthrop Publishers, Inc., Cambridge, Massachusetts (1976)
- Desjardins, P., "A Pascal Compiler for the Xerox Sigma 6", SIGPLAN NOTICES -8-, 6, pp. 34-36 (1973)
- Deverill, R. S., Hartmann, A. C., "Interpretive PASCAL for the IBM 370", Information Science Technical Report No. 6, California Institute of Technology (1973)
- Feiereisen, L., "Implementation of PASCAL on the PDP 11/45", DECUS Conference, Zurich, pp. 259 (September 1974)
- Findlay, W., "The Performance of Pascal Programs on the MULTUM", Report No. 6, Computing Department, University of Glasgow, Scotland (July 1974)
- Friesland, G., et al., "A Pascal Compiler Bootstrapped on a DEC-System 10", Lecture Notes in Computer Science -7-, pp. 101-113, Springer-Verlag (1974)
- Friesland, G., Sengler, H.-E., "Zur Uebertragung von Compilern durch Selbstcompilation am Beispiel des PASCAL-Compilers", Institut fuer Informatik des Universitaet Hamburg, report 1F1-HH-B-13/74 (December 1974)
- Grosse-Lindemann, C.-O., Lorenz, P.-W., Nagel, H.-H., Stirl, P. J., "A PASCAL Compiler Bootstrapped on a DEC-System 10", Fachtagung über Programmiersprachen, pp. 101-113, Lecture Notes in Computer Science -3-, Springer-Verlag (1974)
- Grosse-Lindemann, C.-O., Nagel, H.-H., "Postlude to a Pascal-Compiler Bootstrap on a DEC System-10", Bericht Nr. 11, Institut für Informatik, Universität Hamburg, Germany (1974); SOFTWARE-PRACTICE AND EXPERIENCE -6-, 1, pp. 29-42 (January 1976)
- Habermann, A. N., "Critical Comments on the Programming Language Pascal", ACTA INFORMATICA -3-, 1, pp. 47-57 (1973)
- Hansen, P. B., "Operating System Principles", Prentice-Hall, Englewood Cliffs, New Jersey (1973)
- Hansen, P. B., "The Purpose of Concurrent Pascal", SIGPLAN NOTICES -10-, 6, pp. 305-309 (1975)
- Heistad, E., "Pascal - Cyber Version", Teknisk Notat S-305 Forsvarets Forskningsinstitutt, Norwegian Defense Research Establishment, Kjeller, Norway (June 1973)
- Hikita, T., Ishihata, K., "PASCAL 8000 REFERENCE MANUAL, Version 1.0", Technical Report 76-02, Department of Information Science, Faculty of Science, University of Tokyo (March 1976)
- Hoare, C. A. R., Wirth, N., "An Axiomatic Definition of the Programming Language Pascal", ACTA INFORMATICA -2-, 4, pp. 335-355 (1973)
- Illum, K., "En introduktion til programmeringssporget Pascal", Danmarks Ingeniorakademi, Aalborg (1973)
- Ishihata, K., Hikita, T., "Bootstrapping PASCAL Using a Trunk", Technical Report 76-04, Department of Information Science, Faculty of Science, University of Tokyo (March 1976)
- Jensen, K., Wirth, N., "Pascal User Manual and Report", Lecture Notes in Computer Science, -18-, Springer-Verlag, New York (1974); Springer Study Edition (1975)
- Knobe, B., Yuval, G., "Making a Compiler Indent", Computer Science Department, The Hebrew University of Jerusalem, Israel (November 1974)
- Kristensen, B. B., Madsen, O. L., Jensen, B. B., Eriksen, S. H., "A Short Description of a Translator Writing System (BOBS-System)", Daimi PB-11, University of Aarhus, Denmark (February 1973)
- Kristensen, B. B., Madsen, O. L., Jensen, B. B., "A Pascal Environment Machine (P-code)", Daimi PB-28, University of Aarhus, Denmark (April 1974)
- Kristensen, B. B., Madsen, O. L., Jensen, B. B., Eriksen, S. H., "User Manual for the BOBS-System", Unpublished English Version, University of Aarhus, Denmark (April 1974)
- Lecarme, O., "Le langage de programmation Pascal", Université de Montreal (1972)
- Lecarme, O., "Structured Programming, Programming Teaching, and the Language Pascal", SIGPLAN NOTICES -9-, 7, pp. 15-21 (July 1974)
- Lecarme, O., Desjardins, P., "Reply to a Paper by A. N. Habermann on the Programming Language Pascal", SIGPLAN NOTICES -9-, 10, pp. 21-27 (October 1974)
- Lecarme, O., Desjardins, P., "More Comments on the Programming Language Pascal", ACTA INFORMATICA -4-, pp. 231-243 (1975)
- MacLennan, B. J., "A Note on Dynamic Arrays in Pascal", SIGPLAN NOTICES -10-, 9, pp. 39-40 (September 1975)
- Mancel, P., Thibault, D., "Transport d'un compilateur PASCAL, Ecrit en PASCAL d'un CDC 6400 sur un CII IRIS 80", These de Docteur Ingenieur, Université Paris VI (1974)
- Marmier, E., "A Program Verifier for Pascal", Information Processing 74 (IFIP Congress 1974), North-Holland (1974)
- Mickel, A., "Pascal Newsletter", University of Minnesota Computer Center, Minneapolis; No. 5 (September 1976), No. 6 (November 1976) (see G. Richmond)
- Molster, T., Sundvor, V., "Unit Pascal System for the Univac 1108 Computer", Teknisk Notat 1/74, Institut for Databehandling, Universitet I Trondheim, Norway (February 1974)
- Nagel, H.-H., "Pascal for the DEC-System 10, Experiences and Further Plans", Mitteilung Nr. 21, Institut für Informatik, Universität Hamburg (November 1975)
- Nori, K. V., Ammann, U., Jensen, K., Nagel, H. H., "The Pascal(P) Compiler: Implementation Notes", No. 10, Berichte des Instituts für Informatik, Eidgenossische Technische Hochschule, Zurich (December 1974)
- Richmond, G., edit., "Pascal Newsletter", University of Colorado Computing Center, Boulder; No. 1 (January 1974), SIGPLAN NOTICES -9-, 3, pp. 21-28 (March 1974); No. 2 (May 1974), SIGPLAN NOTICES -9-, 11, pp. 11-17 (November 1974); No. 3 (February 1975), SIGPLAN NOTICES -11-, 2, pp. 33-48 (February 1976); No. 4 (July 1976) (see A. Mickel)
- Rowland, D., "Pascal for Systems", paper presented at DECUS (Digital Equipment Corporation User's Society) (December 1975)
- Saxena, A. R., Bredt, T. H., "A Structured Specification of a Hierarchical Operating System", SIGPLAN NOTICES -10-, 6, pp. 310-318 (June 1975)
- Schauer, H., "PASCAL fuer Angaenger", Oldenbourg-Verlag, Wien, Muenchen (1976)
- Schild, R., "Implementation of the Programming Language Pascal", Lecture Notes in Economics and Mathematical Systems, -75- (1972)
- "SFER PASCAL, Le Langage de programmation PASCAL - compilateur pour les ordinateurs CII 10070, IRIS 80", IRIA (1975)
- Solntseff, N., "McMaster Modifications to the Pascal 6000 3.4 System", Computer Science Technical Note 74-CS-2, McMaster University, Ontario, Canada (November 1974)
- Takeichi, M., "On the Portability of a PASCAL Compiler", Proceedings of the 16-th Programming Symposium, pp. 90-96 (1975) in Japanese
- Takeichi, M., "PASCAL Compiler for the FACOM 230-38: Implementation Notes", Internal Report, University of Tokyo, Department of Mathematic Engineering and Instrumentation Physics (1975)
- Takeichi, M., "PASCAL -- Implementation and Experience", University of Tokyo, Department of Mathematic Engineering and Instrumentation Physics (December 1975)
- Thibault, D., Mancel, P., "Implementation of a Pascal Compiler for the CII Iris 80 Computer", SIGPLAN NOTICES -8-, 6, pp. 89-90 (1973)
- de Vries, W., "An Implementation of the language Pascal for the PDP 11 series, based on a portable Pascal compiler", Technische Hogeschool Twente, Enschede (March 1975)
- Welsh, J., Quinn, C., "A Pascal Compiler for the ICL 1900 Series Computer", SOFTWARE-PRACTICE AND EXPERIENCE -2-, 1, pp. 73-77 (1972)
- Wirth, N., Hoare, C. A. R., "A Contribution to the Development of Algol", COMMUNICATIONS OF THE ACM -9-, 6, pp. 413-432 (1966)
- Wirth, N., "The Programming Language Pascal", ACTA INFORMATICA -1-, 1, pp. 35-63 (1971)
- Wirth, N., "The Design of a Pascal Compiler", SOFTWARE-PRACTICE AND EXPERIENCE -1-, 4, pp. 309-333 (1971)
- Wirth, N., "Program Development by Step-Wise Refinement", COMMUNICATIONS OF THE ACM -14-, 4, pp. 221-227 (April 1971)
- Wirth, N., "The Programming Language Pascal and Its Design Criteria", presented at the Conference on Software Engineering Techniques (NATO Science Committee), Rome (October 1968); published in "High Level Languages", Infotech State of the Art Report 7 (1972)
- Wirth, N., "Systematisches Programmieren" (Taschebuch), Teubner-Verlag, Stuttgart (1972)
- Wirth, N., "The Programming Language Pascal (Revised Report)", Nr. 5, Berichte des Instituts für Informatik, Eidgenossische Technische Hochschule, Zurich (November 1972)
- Wirth, N., "On Pascal, Code Generation, and the CDC 6400 Computer", Computer Science Department, STAN-CS-72-257, Stanford University (1972) (out of print, Clearinghouse stock no. PB208519)
- Wirth, N., "Systematic Programming: An Introduction", Prentice-Hall, Englewood Cliffs, New Jersey (1973)
- Wirth, N., "On the Composition of Well-Structured Programs", COMPUTING SURVEYS -6-, 4, pp. 247-260 (December 1974)
- Wirth, N., "Algorithmen und Datenstrukturen", Teubner-Verlag, Stuttgart (1975)
- Wirth, N., "Algorithms + Datastructures = Programs", Prentice-Hall, Englewood Cliffs, New Jersey (1975)
- Wirth, N., "An Assessment of the Programming Language Pascal", IEEE TRANSACTIONS ON SOFTWARE ENGINEERING -1-, 2, pp. 192-198 (1975); SIGPLAN NOTICES -10-, 6, pp. 23-30 (June 1975)
- Wirth, N., "PASCAL-S: A Subset and its Implementation", Nr. 12, Berichte des Instituts für Informatik, Eidgenossische Technische Hochschule, Zurich (June 1975)
- Wirth, N., "Comment on A Note on Dynamic Arrays in Pascal", SIGPLAN NOTICES -11-, 1, pp. 37-38 (January 1976)

PAST ISSUES OF Pascal Newsletter (now Pascal News)

George Richmond, Computing Center, University of Colorado, started Pascal Newsletter with issue #1 in January, 1974. He proceeded to produce 3 more issues while doing the other thankless chores of distributing 2 Pascal compilers to dozens of sites and promoting Pascal in other ways.

In mid-1975 John Strait and I proposed a Pascal User's Group after having talked to several other Pascalers around the U.S. At the Minneapolis ACM '75 conference in October, 1975, we launched the group at an ad hoc meeting (35 persons) convened by Rich Cichelli and Bob (Warren) Johnson. A year later we began the task of producing 4 issues of Pascal Newsletter which PUG as a group assumed responsibility for.

John and I edited the first 2 issues with help from Tim Bonham on the Implementations section. By issue #8 John had less time for the constant demands of the newsletter and only promised occasional help, but with #8 Jim Miner, Sara Graffunder, and others volunteered to help. With this issue (#9 & #10), we have spread the load quite a bit, which only causes coordination problems!

#1 January, 1974, University of Colorado Computing Center, (also SIGPLAN Notices 9:3 1974 March) 8 pages, edited by George Richmond. (* Mostly contained descriptions of the CDC-6000 implementation of unrevised Pascal. *) out of print

#2 May, 1974, University of Colorado Computing Center, (also SIGPLAN Notices 9:11 1974 November) 18 pages, edited by George Richmond. (* A Pascal history; news of other implementations for unrevised Pascal; news of the new CDC-6000 implementation for revised Pascal.*) out of print

#3 February, 1975, University of Colorado Computing Center, (also SIGPLAN Notices 11:2 1976 February) 19 pages, edited by George Richmond. (* Announcement of the book: Pascal User Manual and Report; Pascal usage questionnaire; revised History of Pascal; bibliography; news of Pascal-P; more on Pascal-6000 for CDC machines; letters to the editor.*) out of print

#4 August, 1976, University of Colorado Computing Center, 103 pages (103 numbered pages), edited by George Richmond. (* 36 letters of correspondence dealing mostly with various implementations; implementors list; bibliography; news of new release of Pascal-P.*) out of print

#5 September, 1976, Pascal User's Group, University of Minnesota Computer Center, 124 pages (65 numbered pages), edited by Andy Mickel. (* Short notes, 5 articles, general correspondence, and implementation notes were featured; Christian Jacobi, ETH Zurich supplied a description of Dynamic Array Parameters. *)

#6 November, 1976, Pascal User's Group, University of Minnesota Computer Center, 180 pages (91 numbered pages), edited by Andy Mickel. (* News from members; a full membership roster; conference notices; information on back issues; 6 articles including 2 proposing directions for Pascal by G. Michael Schneider of the U of Minnesota, and Rich Cichelli of Lehigh University; much implementation news. *)

#7 February, 1977, Pascal User's Group, University of Minnesota Computer Center, 90 pages (45 numbered pages), edited by Andy Mickel. (* More News from members; books; 3 articles; correspondence; implementation notes. *)

#8 May, 1977, Pascal User's Group, University of Minnesota Computer Center, 128 pages (65 numbered pages), edited by Andy Mickel. (* News from members; Conferences; Books; Applications; 6 articles including one by Ken Bowles about a very complete inexpensive implementation for nearly every microprocessor in existence; Special topic: official standardization and clarified definition of Pascal; Portable Pascals, Feature implementation notes, Machine Dependent implementations, Index. *)

Back issue ordering information for #5-#8 is on the back of the ALL PURPOSE COUPON.

PUG FINANCES 1976-1977

Here are the details for our finances last academic year by both PUG(USA) and PUG(UK). For additional information see the EDITOR'S CONTRIBUTION (for a real con) under "PUG Finances."

PUG(USA) Accounts:

Income:

\$3980.00 995 memberships @ \$4 (76-77)
161.73 contributions
70.00 miscellaneous back issues sold @ \$1

\$4211.73 TOTAL Income.

Expenditure:

\$ 123.37 buying (230) and mailing #4 from George Richmond
492.70 printing (700) and mailing #5
1239.50 printing (1050) and mailing #6
697.17 printing (1000) and mailing #7
1071.60 printing (1000) and mailing #8
30.23 mailing originals of 5-8, etc. to PUG(UK) for reprinting
92.13 promoting PUG (mass mailings)
10.00 refunds for overpayment
19.00 backissue requests for #4 forwarded to George Richmond
101.09 miscellaneous postage for automatic backissues

\$3876.79 TOTAL Expenditure.

PUG(UK) Accounts: (submitted by David Barron, 15 August, 1977)

Income:

£249.90 subscriptions for 76-77 (99 @ 2.50; 1 @ 2.40)

£249.20 TOTAL Income.

Expenditure:

£ 70.86 printing 250 copies of No. 6
29.14 printing 350 copies of No. 7
105.34 printing 450 copies of No. 8
171.06 postage for 6, 7, and 8 including back issues

£ 376.40 total production costs

90.01 printing and posting No. 5 (450 copies)

£466.41 TOTAL Expenditure.

=====

PUG(USA) surplus = \$334.94
PUG(UK) deficit = £216.51 = \$380.00 approx.
Total deficit for year = \$ 45.06

Andy Mickel 77/09/01

ROSTER (77/09/09)

The PUG roster is sorted by mail code (USA first) and then alphabetically by country. Members span 31 countries and 47 states. Also supplied is a member index by last name to mail code. Institutional members begin with the prefix ATTN or ATTENTION.

You can see at a glance who is at a well known organization at a well known place. The roster makes a great organizing tool for our mutual communication! Please look yourself up to check for accuracy and then you can see who is nearby; why not phone them and talk about Pascal?

States with over 50 PUG members are: California - 171; Minnesota - 128; Texas - 62; Massachusetts - 61; and countries: United Kingdom - 101; Canada - 59; Germany - 57.

01002 HENRY F. LEDGARD/ COMPUTER AND INFO. SCI./ U OF MASSACHUSETTS/ AMHERST MA 01002/ (413) 545-2744
 01420 KENNETH R. WADLAND/ COMPUTER SCIENCE DEPT./ FITCHBURG STATE COLLEGE/ MAIL BOX NUMBER 6372/ FITCHBURG MA 01420/ (617) 342-2268
 01451 RALPH S. GOODELL/ HILLCREST DRIVE/ HARVARD MA 01451/ (617) 456-8090
 01451 R. L. THAYER/ CENTRAL RESEARCH GROUP/ P.O. BOX 451/ HARVARD MA 01451/ (617) 772-2306
 01609 JOHN DE ROSA JR./ WORCESTER POLYTECHNIC INST./ P.O. BOX 2131/ WORCESTER MA 01609/ (617) 798-8947
 01609 NORMAN E. SONDAK/ COMP. SCI. DEPT./ WORCESTER POLYTECHNIC INSTITUTE/ WORCESTER MA 01609/ (617) 753-1411
 01701 HANK EDWARDS/ 2C BRACKETT ROAD/ FRAMINGHAM MA 01701/ (617) 620-1066 (HOME)/ (617) 897-5111 X6809
 01701 BERNIE ROSMAN/ MATH/CS DEPT./ FRAMINGHAM STATE COLLEGE/ FRAMINGHAM MA 01701/ (617) 872-3501
 01701 DAVID TARABAR/ FIELD ENGINEERING/ DATA GENERAL CORPORATION/ 235 OLD CONNECTICUT PATH/ FRAMINGHAM MA 01701/ (617) 620-1200 X362
 01720 SCOTT D. HANKIN/ 382A GREAT ROAD APT. 103/ ACTON MA 01720/ (617) 263-9121
 01730 BRUCE MACKENZIE/ COMPUTERVISION CORP./ 201 BURLINGTON ROAD - ROUTE 62/ BEDFORD MA 01730/ (617) 275-1800
 01741 MARTHA L. SPENCE/ 145 CARLISLE PINES DR./ CARLISLE MA 01741/ (617) 369-7311 (HOME)/ (617) 449-2000 X2526 (OFC.)
 01742 MARK S. MAYES/ TSD SYSTEMS ENGINEERING/ GEN RAD./ 300 BAKER AVE./ W. CONCORD MA 01742/ (617) 369-4400
 01749 KATHLEEN JENSEN/ 1 FRANKLIN ST./ HUDSON MA 01749/ (617) 897-5111 (WORK)/ (617) 562-9538 (HOME)
 01752 DWIGHT BAKER/ MR2/M64/ DIGITAL COMPONENTS/ ONE IRON WAY/ MARLBORO MA 01752/ (617) 481-7400 X6637
 01752 CARL W. SCHWARCZ/ MR 1-2/E27/ DIGITAL EQUIPMENT CORP./ 200 FOREST STREET/ MARLBORO MA 01752/ (617) 481-9511
 01754 ATTN: LIBRARY/ ML5-4/A20/ DIGITAL EQUIPMENT CORPORATION/ MAYNARD MA 01754
 01754 RONALD F. BRENDER/ BLISS LANGUAGE DEVELOPMENT/ ML3-5/E82/ DIGITAL EQUIPMENT CORP./ 146 MAIN STREET/ MAYNARD MA 01754/ (617) 897-5111 X2520
 01754 ALBERT S. BROWN/ PK3-1/M12/ DIGITAL EQUIPMENT CORP./ 146 MAIN STREET/ MAYNARD MA 01754/ (617) 897-5111 X2391
 01754 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
 01754 RONALD J. HAM/ ML5-5/E40/ DIGITAL EQUIPMENT CORPORATION/ 146 MAIN STREET/ MAYNARD MA 01754/ (617) 897-5111
 01754 RICHARD KIMBALL/ 145 WALTHAM ST./ MAYNARD MA 01754
 01754 DAVID MOBERLY/ P.O. BOX 241/ MAYNARD MA 01754/ (617) 897-8078
 01754 ISAAC R. NASSI/ DIGITAL EQUIPMENT CORP./ 146 MAIN STREET/ MAYNARD MA 01754/ (617) 897-5111 X4487
 01754 WILLIAM F. SHAW/ ML5-5/E40/ DIGITAL EQUIPMENT CORPORATION/ 146 MAIN STREET/ MAYNARD MA 01754/ (617) 897-5111
 01776 LLOYD DICKMAN/ 93 PRATTS MILL ROAD/ SUDBURY MA 01776
 01852 EDWARD STEEN/ 119 SHERMAN STREET/ LOWELL MA 01852/ (617) 454-9320
 01907 JAMES W. HEBERT/ 51 THOMAS ROAD/ SWAMPSCOTT MA 01907/ (617) 581-3807
 02035 THOMAS G. MCGINTY/ DEPT. 330/ FOXBORO CO./ 38 NEPONSET AVE./ FOXBORO MA 02035/ (617) 543-8750 X2031
 02035 AARON SAWYER/ DEPT 330/ THE FOXBORO COMPANY/ FOXBORO MA 02035/ (617) 543-8750 X2029
 02038 WARREN R. BROWN/ D.330/ THE FOXBORO COMPANY/ 38 NEPONSET AVE./ FOXBORO MA 02038/ (617) 543-8750 X2023
 02111 ROGER A. DUE/ SOFTWARE SYSTEMS DESIGN/ TERADYNE INC./ 183 ESSEX STREET/ BOSTON MA 02111/ (617) 482-2700
 02115 ATTN: MATH LIBRARY/ NORTHEASTERN UNIVERSITY/ 360 HUNTINGTON AVE./ BOSTON MA 02115/ (617) 437-2460
 02115 JOHN CASEY/ DEPARTMENT OF MATHEMATICS/ NORTHEASTERN UNIVERSITY/ 360 HUNTINGTON AVENUE/ BOSTON MA 02115/ (617) 437-2450
 02115 JENNIFER CLARKE/ COMPUTATION CENTER/ 25 RICHARDS HALL/ NORTHEASTERN U./ HUNTINGTON AVE./ BOSTON MA 02115/ (617) 437-3183
 02125 VICTOR S. MILLER/ DEPT OF MATHEMATICS/ BLDG 2/ U OF MASSACHUSETTS/ HARBOR CAMPUS/ BOSTON MA 02125/ (617) 287-1900 X3170/X3161
 02134 DAN FYLSTRA/ 22 WEITZ ST. #3/ BOSTON MA 02134
 02138 BRUCE KNOBE/ INTERMETRICS INC./ 701 CONCORD AVE./ CAMBRIDGE MA 02138/ (617) 661-1840
 02138 MICHAEL MEEHAN/ WINTHROP PUBLISHERS/ 17 DUNSTER STREET/ CAMBRIDGE MA 02138/ (617) 868-1750
 02138 CHARLES ROBERT MORGAN/ BOLT BERANEK AND NEWMAN/ 50 MOULTON STREET/ CAMBRIDGE MA 02138/ (617) 491-1850 X502
 02138 ROBERT E. WELLS/ BOLT BERANEK AND NEWMAN INC./ 50 MOULTON STREET/ CAMBRIDGE MA 02138/ (617) 491-1850
 02139 ATTN: READING ROOM/ INFORMATION PROCESSING CENTER/ 39-430/ MIT # CAMBRIDGE MA 02139
 02139 GABRIEL CHANG/ 575 TECHNOLOGY SQUARE/ HONEYWELL INFORMATION SYSTEMS/ CAMBRIDGE MA 02139/ (617) 491-6300
 02139 F. J. CORBATO/ NE43-514/ MASSACHUSETTS INSTITUTE OF TECHNOLOGY/ 545 TECHNOLOGY SQUARE/ CAMBRIDGE MA 02139/ (617) 253-6001
 02139 JEANNE FERRANTE/ 125 ANTRIM ST./ CAMBRIDGE MA 02139/ (617) 876-8635
 02139 JOHN N. STRAYHORN/ BOX 157 MIT BRANCH P.O./ CAMBRIDGE MA 02139/ (617) 923-1133
 02140 KENNETH OLSON/ 16 MONTGOMERY ST./ CAMBRIDGE MA 02140/ (617) 868-3068
 02154 R. STERLING EANES/ SOFTECH/ 460 TOTTEN POND ROAD/ WALTHAM MA 02154/ (617) 890-6900
 02154 JOHN B. GOODENOUGH/ SOFTECH INC./ 460 TOTTEN POND RD/ WALTHAM MA 02154/ (617) 890-6900
 02154 R. KRASIN/ FIRST DATA CORP./ 400 TOTTEN POND ROAD/ WALTHAM MA 02154/ (617) 890-6701
 02154 GERALD NADLER/ RBMS RESEARCH CENTER/ BRANDEIS UNIVERSITY/ WALTHAM MA 02154
 02154 MICHAEL ROONEY/ THE BOSTON SYSTEMS OFFICE INC./ 400-1 TOTTEN POND ROAD/ WALTHAM MA 02154/ (617) 890-0888
 02154 ROY A. WILSKER/ 27 BENEFIT STREET/ WALTHAM MA 02154/ (617) 899-6638
 02155 DAVID SOLOMONT/ COMPUTER SERVICES/ MILLER HALL/ TUFTS UNIVERSITY/ MEDFORD MA 02155/ (617) 628-2943
 02160 PETER COLBY/ 289 MILL ST./ NEWTONVILLE MA 02160/ (617) 527-2394
 02167 GEORGE C. HETRICK/ COMPUTING CENTER/ BOSTON COLLEGE/ CHESTNUT HILL MA 02167/ (617) 969-0100
 02168 GEORGE POONEN/ 15 ORCHARD AVE./ WABAN MA 02168/ (617) 969-4684

02172 FRED EILENSTEIN/ 68 SPRING STREET/ WATERTOWN MA 02172/ (617) 924-2248
02173 G. M. SHANNON/ LINCOLN LAB/ J-148G/ M.I.T./ 244 WOOD STREET/ LEWINGTON MA 02173/ (617) 862-5500 X5719
02174 MICHAEL HAGERTY/ 83 PARK STREET/ ARLINGTON MA 02174/ (617) 492-7100
02193 TERRENCE M. COLLIGAN/ RIVERSIDE OFFICE PARK/ MANAGEMENT DECISION SYSTEMS INC./ RIVERSIDE ROAD/ WESTON MA 02193/ (617) 891-0335
02809 E. R. BEAUREGARD/ 10 HYDRAULION AVE./ BRISTOL RI 02809/ (401) 253-7358
02881 DAVID J. GRIFFITHS/ ACADEMIC COMPUTER CENTER/ TYLER HALL/ UNIVERSITY OF RHODE ISLAND/ KINGSTON RI 02881/ (401) 792-2701
02912 ANDRIES VAN DAM/ BROWN UNIVERSITY/ BOX F/ PROVIDENCE RI 02912/ (401) 863-3088
03060 ATTENTION: JO AN HUESMAN/ NASHUA OPERATIONS/ HARRIS DATA COMMUNICATIONS DIV./ DANIEL WEBSTER HIGHWAY SOUTH/ NASHUA NH 03060/ (603) 883-3313
03301 VINCENT KAYSER/ NORTHEAST ELECTRONICS/ BOX 649/ CONCORD NH 03301/ (603) 224-6511 X-261
03458 CARL HELMERS/ BYTE PUBLICATIONS INC./ 70 MAIN STREET/ PETERBOROUGH NH 03458/ (603) 924-7217
03766 WILLIAM M. LAYTON/ POLYTRONICS/ METHODIST HILL/ LEBANON NH 03766/ (603) 646-2068
03824 ATTENTION: R. D. BERGERON/ DEPARTMENT OF MATHEMATICS/ KINGSBURY HALL/ U OF NEW HAMPSHIRE/ DURHAM NH 03824/ (603) 862-2321
03824 WILLIAM J. VASILIOU JR./ COMPUTER SERVICES/ KINGSBURY HALL/ U OF NEW HAMPSHIRE/ DURHAM NH 03824/ (603) 862-2323
04103 JOHN HEATH/ DEPT. OF MATH. AND COMPUTER SCI./ UNIV. OF MAINE/ PORTLAND ME 04103/ (207) 773-
06035 TIMOTHY DENNIS/ 62 LAKESIDE DRIVE/ GRANBY CT 06035/ (203) 653-4492
06268 EDWARD E. BALKOVICH/ DEPT. OF ELECT. ENGR. AND COMP. SCI./ U-157/ UNIV. OF CONNECTICUT/ STORRS CT 06268/ (203)486-4816
06320 JAMES P. SHORES/ 344 GLENWOOD AVE./ NEW LONDON CT 06320/ (203) 442-0771 X2126
06413 ROSEMARY HOWBRIGG/ 36 MENUNKETESUCK DRIVE/ CLINTON CT 06413/ (203) 669-5812 (HOME)/ (203) 442-0771 X2963 (WORK)
06432 MARK BECKER/ 300 COLLINGWOOD AVE/ FAIRFIELD CT 06432/ (203) 334-3627
06488 CHARLES E. SIMON/ RD #1 BERKSHIRE RD./ SOUTHURY CT 06488/ (203) 264-0640 (HOME)/ (203) 377-4141 X2286 (WORK)
06901 MARK SEIDEN/ NATIONAL CSS INC./ 500 SUMMER ST. - 4 FL./ STAMFORD CT 06901/ (203) 327-9100 X206
07102 PETER ANDERSON/ COMPUTER AND INFO SCI DEPT./ NEW JERSEY INSTITUTE OF TECHNOLOGY/ 323 HIGH STREET/ NEWARK NJ 07102/ (201) 645-5126
07205 NICHOLAS WYBOLT/ 576 LEO STREET/ HILLSIDE NJ 07205/ (201) 688-5328
07470 RICHARD D. SPILLANE/ DEPT OF MATH/C.S./ WILLIAM PATTERSON COL./ WAYNE NJ 07470/ (201) 881-2158
07724 DAN C. RICHARD/ P.O. BOX 188/ EATONTOWN NJ 07724/ (201) 542-3814 (HOME)
07724 RON PRICE/ PERKIN-ELMER DATA SYSTEMS/ 106 APPLE ST./ TINTON FALLS NJ 07724
07733 RON OLSEN/ ROOM 3E207/ BELL LABORATORIES/ CRAWFORD CORNER ROAD/ HOLMDEL NJ 07733/ (201) 949-5537
07757 FRANK KURKA/ P.O. BOX 209/ OCEANPORT NJ 07757/ (201) 229-4487
07828 KEN POLAKOWSKI/ 5D VILLAGE GREEN/ BUDD LAKE NJ 07828/ (201) 347-4375
08540 PAUL S. HELLER/ EDUCOM/ P.O. BOX 364/ PRINCETON NJ 08540/ (609) 921-7575
08903 CHARLES HEDRICK/ COMPUTER SCIENCE DEPT./ RUTGERS/ HILL CENTER/ NEW BRUNSWICK NJ 08903
08904 STEVE LEGENHAUSEN/ 12 BARNARD STREET/ HIGHLAND PARK NJ 08904/ (201) 572-6585
10003 WILLIAM HENRY/ 117 E. TENTH ST./ NEW YORK NY 10003
10003 DAVID SEGAL/ 111 THIRD AVE. #2K/ NEW YORK NY 10003/ (212) 674-0454
10012 EDWARD R. FRIEDMAN/ CIMS/CS DEPT./ NEW YORK UNIVERSITY/ NEW YORK NY 10012/ (212) 460-7100
10012 DAVID SHIELDS/ COURANT INSTITUTE/ NEW YORK UNIVERSITY/ 251 MERCER ST./ NEW YORK NY 10012/ (212) 460-7168
10013 FRANK PAVLIK/ QUOTRON SYSTEMS INC./ 325 HUDSON ST./ NEW YORK NY 10013/ (212) 344-0400 EXT. 71
10016 GENE A. DAVENPORT/ JOHN WILEY AND SONS/ 605 THIRD AVENUE/ NEW YORK NY 10016/ (212) 867-9800
10016 STEPHEN LEIBOWITZ/ 165 EAST 32 ST. - APT. 6D/ NEW YORK NY 10016/ (212) 483-2595/ (212) 889-1035
10019 DOUGLAS R. KAYE/ COMPUTER SERVICES/ DU ART FILM LABORATORIES/ 245 WEST 55 ST./ NEW YORK NY 10019/ (212) 757-4580
10019 PETER PAWELCZAK/ UNIVERSITY COMPUTER CENTER/ C/O LIBRARY/ CUNY/ 555 W. 57TH ST./ NEW YORK NY 10019
10024 STEVE GROSS/ 200 W. 86TH ST./ NEW YORK NY 10024
10025 HOWARD D. ESKIN/ CENTER FOR COMPUTING ACTIVITIES/ ROOM 712/ COLUMBIA UNIVERSITY/ 612 W. 115TH ST./ NEW YORK NY 10025/ (212) 280-2874
10027 T. A. D'AURIA/ CENTER FOR COMPUTING ACTIVITIES/ COLUMBIA UNIVERSITY/ NEW YORK NY 10027
10036 P. J. PLAUGER/ SUITE 3830/ YOURDON/ 1133 AVE. OF THE AMERICAS/ NEW YORK NY 10036/ (212) 730-2670
10598 PETER G. CAPEK/ IBM RESEARCH CENTER/ P.O. BOX 218/ YORKTOWN HTS NY 10598/ (914) 945-1250
11530 REX FRANCIOTTI/ COMPUTER CENTER/ ADELPHI UNIVERSITY/ GARDEN CITY NY 11530/ (516) 294-8700
11740 M. WAITE/ HAZELTINE CORP./ GREENLAWN NY 11740/ (516) 261-7000 X687
11794 ATTENTION: GARRY S. MEYER/ COMPUTING CENTER/ APPLICATIONS SUPPORT/ SUNY STONY BROOK/ STONY BROOK NY 11794/ (516) 246-7047
11794 WILLIAM BARABASH/ DEPT. OF COMP. SCI./ SUNY STONY BROOK/ STONY BROOK NY 11794/ (516) 246-7146
11794 RICHARD B KIEBURTZ/ DEPT. OF COMPUTER SCI./ SUNY AT STONY BROOK/ STONY BROOK NY 11794/ (516) 246-5987
11973 M. ELIZABETH IBARRA/ DEPT. OF APPLIED MATH/ BROOKHAVEN NATIONAL LABORATORY/ UPTON NY 11973/ (516) 345-4162
12180 J. SCOTT MERRITT/ 36 OAKWOOD AVE./ TROY NY 12180/ (518) 271-7553
12181 S. KAMAL ABDALI/ DEPT. OF MATHEMATICAL SCIENCES/ RENSSELAER POLYTECHNIC INSTITUTE/ TROY NY 12181/ (518) 270-6558
12308 GEORGE H. WILLIAMS/ EE/CS DEPT./ UNION COLLEGE/ SCHENECTADY NY 12308/ (518) 370-6273
13032 J. WILSON/ WHITMAN RD. R.D. #3 BOX 224H/ CANASTOTA NY 13032/ (315) 697-3639
13201 J. DANIEL GERSTEN/ COMPUTED IMAGE ENG. - CSP 3-21/ GENERAL ELECTRIC CO./ SYRACUSE NY 13201
13210 J. L. POSDAMER/ SCHOOL OF COMP. AND INFO. SCI./ 313 LINK HALL/ SYRACUSE U/ SYRACUSE NY 13210/ (315) 423-4679
13210 JOHN M. WOBUS/ 453 WESTCOTT ST. APT. 1/ SYRACUSE NY 13210/ (315) 472-4923
13323 WALTER WUENSCH/ BOX 62/ CLINTON NY 13323/ (315) 797-2370
13440 DAVID A. BENNETT/ PAR CORP./ ON THE MALL/ ROME NY 13440/ (315) 336-8400
13440 MICHAEL N. CONDICT/ PATTERN ANALYSIS AND RECOGNITION CORP/ ON THE MALL/ ROME NY 13440/ (315) 336-8400 X36
13676 NEWTON J. MUNSON/ COMPUTING CENTER/ CLARKSON COLLEGE/ POTSDAM NY 13676/ (315) 268-7721
13676 TED TENNY/ COMPUTER SCIENCE DEPT./ SUNY - POTSDAM/ POTSDAM NY 13676/ (315) 268-2954
13760 ROBERT L. KING/ 1452 SANDRA DR./ ENDICOTT NY 13760/ (607) 754-3112
14063 G. H. GOLDEN JR./ COMPUTER CENTER/ MAYTUM HALL/ STATE UNIVERSITY COLLEGE/ FREDONIA NY 14063

14226 G. FRIEDER/ DEPT. OF COMPUTER SCIENCE/ SUNY BUFFALO/ 4226 RIDGE LEA RD./ BUFFALO NY 14226/ (716) 831-1351
 14420 JAMES MOLONEY/ DEPT. OF COMP. SCI./ SUNY BROCKPORT/ BROCKPORT NY 14420/ (716) 395-2384
 14609 EDWARD W. SUOR/ COMPUTER CONSOLES INC./ 97 HUMBOLDT STREET/ ROCHESTER NY 14609/ (716) 482-5000 X291
 14623 MICHAEL J. LUTZ/ SCHOOL OF COMPUTER SCIENCE/ ROCHESTER INSTITUTE OF TECHNOLOGY/ ROCHESTER NY 14623/ (716) 464-2139
 14627 ATTN: PRODUCTION AUTOMATION PROJECT/ ELEC. ENGR. - COL. OF ENGR. AND APPLIE/ UNIV. OF ROCHESTER/ ROCHESTER NY 14627/ (716) 275-3775
 14850 RICHARD CONWAY/ DEPT. OF COMPUTER SCIENCE/ CORNELL UNIVERSITY/ ITHACA NY 14850/ (607) 256-3456
 14850 WILLIAM LYCZKO/ SOFTWARE DEVELOPMENT/ NCR CORPORATION/TERMINAL SYSTEMS/ 950 DANBY ROAD/ ITHACA NY 14850/ (607) 273-5310/ X251 X254
 14850 KEVIN WELLER/ 147 CORNELL QRTS/ CORNELL UNIVERSITY/ ITHACA NY 14850/ (607) 256-4880 (DAY)/ (607) 272-7563 (NITE)
 14850 JOHN H. WILLIAMS/ OCS/ 418 UPSON HALL/ CORNELL U/ ITHACA NY 14850/ (607) 256-5033
 14853 THOMAS P. BISHOP/ DEPT. OF COMPUTER SCIENCE/ CORNELL UNIVERSITY/ ITHACA NY 14853/ (607) 256-4052
 14853 HAL PERKINS/ DEPT. OF COMPUTER SCIENCE/ CORNELL UNIVERSITY/ ITHACA NY 14853
 15260 MARY LOU SOFFA/ COMPUTER SCI. DEPT./ 335 ALUMNI HALL/ UNIVERSITY OF PITTSBURGH/ PITTSBURGH PA 15260/ (412) 624-6454
 15261 JOHN DOW/ WESTERN PSYCHIATRIC INST. AND CLINIC/ U. OF PITTSBURGH/ 3811 O'HARA STREET/ PITTSBURGH PA 15261/ (412) 624-2848
 15701 JOHN NOLD/ COMPUTER CENTER/ G7 STRIGHT HALL/ INDIANA UNIVERSITY OF PA./ INDIANA PA 15701
 15701 HOWARD E. TOMPKINS/ COMPUTER SCIENCE DEPT/ INDIANA UNIVERSITY OF PA/ INDIANA PA 15701/ (412) 357-2524
 16802 BENTON LEONG/ COMPUTER SCIENCE DEPT./ PENNSYLVANIA STATE U./ UNIVERSITY PK PA 16802/ (814) 865-1545
 17011 DONALD L. WRIGHT/ 832 WYNNEWOOD RD./ CAMP HILL PA 17011/ (717) 661-0260
 17257 CHARLES E. MILLER/ RD 5 - CRESCENT DRIVE/ SHIPPENSBURG PA 17257/ (717) 532-9121 X104
 17837 ATTENTION: RUTH DROZIN/ FREAS-ROOKE COMPUTER CENTER/ BUCKNELL UNIVERSITY/ LEWISBURG PA 17837/ (717) 524-1436
 17837 DANIEL C. HYDE/ COMPUTER SCIENCE PROGRAM/ BUCKNELL UNIVERSITY/ LEWISBURG PA 17837/ (717) 524-1392
 18015 JOHN W. ADAMS/ DEPT. OF I.E./ 19 PACKARD LAB/ LEHIGH UNIV./ BETHLEHEM PA 18015
 18015 DAVID B. ANDERSON/ DEPT. OF MATHEMATICS/ 14 CHRISTMAS-SAUCON/ LEHIGH UNIVERSITY/ BETHLEHEM PA 18015/ (215) 867-4253
 18015 DAVE ENGLANDER/ 302 SUMMIT STREET/ BETHLEHEM PA 18015/ (215) 865-9027
 18015 S. L. GULDEN/ DEPT. OF MATH/ LEHIGH UNIVERSITY/ BETHLEHEM PA 18015/ (215) 691-7000 X341
 18015 THOMAS RAMSBERGER/ 1036 BROADWAY/ BETHLEHEM PA 18015/ (215) 868-0905
 18015 V. LALITA RAO/ 506 W. THIRD STREET APT. 4/ BETHLEHEM PA 18015/ (215) 865-6448
 18017 STEPHEN TITCOMB/ 1111 NORTH BLVD./ BETHLEHEM PA 18017
 18018 RANCE J. DELONG/ MORAVIAN COLLEGE/ BETHLEHEM PA 18018
 18018 MARILYN HOFFMAN/ 531 W. UNION BLVD./ BETHLEHEM PA 18018/ (215) 865-6937
 18018 JOHN A. WEAVER/ 2112 PENNSYLVANIA AVE. F-6/ BETHLEHEM PA 18018/ (215) 867-1085
 18041 JOSEPH A. MEZZAROBBA/ BOX 164/ E. GREENVILLE PA 18041/ (215) 472-8365 (WORK)/ (215) 679-9900 (HOME)
 18042 BARBARA I. KARKUTT/ BOX 942/ EASTON PA 18042/ (215) 252-1684
 18049 JOHN W. IOBST/ 22 N. KEYSTONE AVE./ EMMAUS PA 18049/ (215) 965-4677
 18055 ALEX OSTAPENKO/ 346 ELLEN ST./ HELLERTOWN PA 18055/ (215) 838-7171
 18103 RICHARD J. CICHELLI/ 901 WHITTIER DRIVE/ ALLENTOWN PA 18103/ (215) 797-9690
 18104 RAMON TAN/ 2345 UNION ST./ ALLENTOWN PA 18104/ (215) 434-5432
 18353 THOMAS HALLDORSON/ BIRCHWOOD PARK #4/ SAYLORSBURG PA 18353
 18651 STEPHEN J VNUK/ 740 MILL ST./ PLYMOUTH PA 18651/ (717) 779-9741
 18703 JOSEPH A. PARKER JR./ DEPT. OF MATH AND COMP. SCI./ WILKES COLLEGE/ WILKES-BARRE PA 18703/ (717) 824-4651 X448
 18938 BILL CHESWICK/ DARIEN 15B / VILLAGE 2/ NEW HOPE PA 18938
 18960 CHESTER J. SALWACH/ 2124 DIAMOND STREET/ SELLERSVILLE PA 18960/ (215) 723-8301
 18974 TOM KELLY/ APT. C 3-9 ASHWOOD APARTMENTS/ 120 EAST STREET ROAD/ WARMINSTER PA 18974/ (215) 674-9821
 19018 T. L.(FRANK) PAPPAS/ 5130 GRAMERCY DRIVE/ CLIFTON HGTS PA 19018/ (215) 259-1325
 19085 ATTN: MATHEMATICS DEPARTMENT/ VILLANOVA UNIVERSITY/ VILLANOVA PA 19085/ (215) 527-2100
 19085 THOMAS SCOTT/ COMPUTER CENTER/ VILLANOVA UNIVERSITY/ VILLANOVA PA 19085/ (215) 527-2100 X215
 19104 DAVID A. NELSON/ INFORMATION ENGINEERING/ 3401 MARKET STREET/ PHILADELPHIA PA 19104/ (215) 387-5150
 19122 FRANK L. FRIEDMAN/ DEPT. OF COMP. AND INFO. SCI./ TEMPLE UNIVERSITY/ PHILADELPHIA PA 19122
 19122 GIORGIO P. INGARGIOLA/ CIS/ 382 SPEAKMAN HALL/ TEMPLE UNIVERSITY/ PHILADELPHIA PA 19122/ (215) 787-8457
 19122 ROBERT KEZELL/ UNIVERSITY COMPUTER ACTIVITY/ TEMPLE UNIVERSITY/ PHILADELPHIA PA 19122/ (215) 787-8527
 19122 FRANK RYBICKI/ COMPUTER ACTIVITY/ TEMPLE UNIVERSITY/ BROAD AND MONTGOMERY/ PHILADELPHIA PA 19122/ (215) 787-1115
 19174 WILLIAM C. HOPKINS/ DEPT. OF COMP. AND INFO. SCI./ U OF PENNSYLVANIA/ PHILADELPHIA PA 19174/ (215) 243-8549
 19301 JOHN T. LYNCH/ BURROUGHS CORP./ P.O. BOX 517/ PAOLI PA 19301
 19301 E. L. ROWE/ BURROUGHS CORP./ BOX 517/ PAOLI PA 19301/ (215) 648-2218
 19401 BILL BRENNAN/ 39 JODY DRIVE/ NORRISTOWN PA 19401
 19426 LEE LAMBERT/ 967 SCHOOL STREET/ COLLEGEVILLE PA 19426
 19464 RICHARD A. JOKIEL/ P.O. BOX 818/ POTTSTOWN PA 19464/ (215) 385-6324
 19711 JOHN D. EISENBERG/ COMPUTING CENTRE/ SMITH HALL/ U OF DELAWARE/ NEWARK DE 19711/ (302) 738-8441 X57 (OFFICE)/ (302) 453-9059 (HOME)
 19711 WILLIAM Q. GRAHAM/ COMPUTING CENTER/ U. OF DELAWARE/ 13 SMITH HALL/ NEWARK DE 19711/ (302) 368-1513
 19711 DAVID HAWK/ 287 WHARTON DRIVE/ NEWARK DE 19711
 19711 ARON K. INSINGA/ DEPT. OF ELEC. ENGR./ 126 DUPONT HALL/ UNIV. OF DELAWARE/ NEWARK DE 19711/ (302) 738-2406
 19898 C. E. BRIDGE/ ENGINEERING DEVELOPMENT LAB/ E. I. DU PONT DE NEMOURS AND CO./ 101 BEECH STREET/ WILMINGTON DE 19898/ (302) 774-1731
 19898 STEPHEN C. SCHWARM/ E.I. DU PONT DE NEMOURS CO./ 101 BEECH ST. / WILMINGTON DE 19898/ (302) 774-1669
 20006 MIKE FRAME/ FIRST DATA CORP./ 2011 EYE ST. NW/ WASHINGTON DC 20006/ (202) 872-0580
 20012 RICK THOMAS/ 408 DOMER AVENUE/ TAKOMA PARK MD 20012/ (301) 565-2678
 20014 TERRY P. MEDLIN/ SCIENTIFIC RESEARCH UNIT - DPSA/ NATIONAL INSTITUTE OF DENTAL HEALTH/ BETHESDA MD 20014
 20014 WAYNE RASBAND/ BLDG 36 ROOM 2A-03/ NATIONAL INSTITUTES OF HEALTH/ BETHESDA MD 20014/ (301) 496-4957

20014 JOHN M. SHAW/ BLDG 36 / ROOM 2A29/ NATIONAL INSTITUTES OF HEALTH/ BETHESDA MD 20014/ (301) 496-3204
20016 DAVID A. GOMBERG/ DEPT. OF MATH. STAT. AND COMP. SCI./ AMERICAN UNIVERSITY/ MASSACHUSETTS & NEBRASKA AVES./ WASHINGTON DC 20016/ (202) 686-2393
20016 JOSEPH P. JOHNSON/ 3520 QUEBEC ST. NW/ WASHINGTON DC 20016/ (202) 362-8523
20037 MARGERY AUSTIN/ THE URBAN INSTITUTE/ 2100 M STREET NW/ WASHINGTON DC 20037/ (202) 223-1950
20037 ARTHUR A. BROWN/ 1101 NEW HAMPSHIRE AVE. NW APT.1002/ WASHINGTON DC 20037/ (202) 785-0716
20052 RICHARD TABOR/ UNIVERSITY COMPUTER CENTER/ GEORGE WASHINGTON UNIVERSITY/ 2013 G STREET N.W. #201/ WASHINGTON DC 20052/ (202) 676-6140
20052 RAYMOND E. THOMAS/ DEPT. OF STATISTICS/ GEORGE WASHINGTON UNIV./ WASHINGTON DC 20052/ (202) 676-6369
20234 T. HARDY/ SECTION J-640.02/ TECH A367/ NATIONAL BUREAU OF STANDARDS/ WASHINGTON DC 20234
20375 PETER A. RIGSBEE/ CODE 5494/ NAVAL RESEARCH LABORATORY/ WASHINGTON DC 20375/ (202) 767-3181
20433 PETER GUTTERMAN/ COMPUTING ACTIVITIES/ DEPT. N954/ THE WORLD BANK/ 1818 H STREET N.W./ WASHINGTON DC 20433/ (202) 393-6360
20550 THOMAS A. KEENAN/ DIVISION OF MATHEMATICAL AND COMPUTER/ NATIONAL SCIENCE FOUNDATION/ WASHINGTON DC 20550/ (202) 632-7346
20705 TED L. FREEMAN/ RDA INC./ 5012 HERZEL PLACE/ BELTSVILLE MD 20705/ (301) 937-2215
20742 SHMUEL PELEG/ COMPUTER SCIENCE CENTER/ U OF MARYLAND/ COLLEGE PARK MD 20742/ (301) 454-4527
20742 BEN SHNEIDERMAN/ DEPT. OF INFO. SYS. MGMT./ U OF MARYLAND/ COLLEGE PARK MD 20742/ (301) 454-2548
20742 JOYCE A. SMITH/ COMPUTER SCIENCE CENTER/ PROGRAM LIBRARY/ U OF MARYLAND/ COLLEGE PARK MD 20742/ (301) 454-4261
20755 JOHN NOLAN/ NATIONAL SECURITY AGENCY/ R51/ DEPARTMENT OF DEFENSE/ 9800 SAVAGE ROAD/ FT. MEADE MD 20755
20810 M. J. GRALIA/ APPLIED PHYSICS LABORATORY/ THE JOHNS HOPKINS UNIVERSITY/ JOHNS HOPKINS ROAD/ LAUREL MD 20810/ (301) 953-7100 X7386
20810 A. E. SALWIN/ APPLIED PHYSICS LABORATORY/ THE JOHNS HOPKINS UNIVERSITY/ JOHNS HOPKINS ROAD/ LAUREL MD 20810/ (301) 953-7100
20854 CHARLES BACON/ 10717 BURBANK DR./ POTOMAC MD 20854/ (301) 299-2732 (HOME)/ (301) 496-4823 (WORK)
20910 JACOB C. Y. WU/ SYSTEM SCIENCES DIVISION/ COMPUTER SCIENCES CORPORATION/ 8728 COLESVILLE ROAD/ SILVER SPRING MD 20910/ (301) 589-1545 X276
21031 ATTN: M. WATKINS - TECHNICAL LIBRARIAN/ GENERAL INSTRUMENT CORP./ 11126 MCCORMICK ROAD/ HUNT VALLEY MD 21031/ (301) 666-8700 X333
21044 DAVID MILLER/ 11203A AVALANCHE WAY/ COLUMBIA MD 21044/ (301) 992-5665
21045 RAINER F. MCCOWN/ MCCOWN COMPUTER SERVICES/ 9537 LONG LOOK LANE/ COLUMBIA MD 21045/ (301) 730-0379
21204 EDWIN J. CALKA/ E152/ AAI CORP/ P.O. BOX 6767/ BALTIMORE MD 21204
21218 JOHN LEWIS/ MATH. SCIENCES DEPT./ JOHNS HOPKINS UNIVERSITY/ CHARLES AND 34TH STREETS/ BALTIMORE MD 21218/ (301) 338-7207
22090 DAVID AULT/ COMPUTER SCIENCE/ VPI AND SU/ 11440 ISAAC NEWTON SQ. N./ RESTON VA 22090/ (703) 471-4601
22091 GUS BJORKLUND/ 2250 COPPERSMITH SQUARE/ RESTON VA 22091
22091 JAMES K. MOORE/ 12345 COLERAINE COURT/ RESTON VA 22091/ (703) 437-2338
22101 EDWARD W. HURLEY/ XONIGS INC./ 1700 OLD MEADOW ROAD/ MCLEAN VA 22101/ (703) 790-1840
22152 MARK S. WATERBURY/ 8358 L DUNHAM CT./ SPRINGFIELD VA 22152
22201 L. EDWARD REICH/ 805 N. CLEVELAND STREET/ ARLINGTON VA 22201/ (703) 243-3131
22209 WILLIAM A. WHITAKER/ DARPA/ 1400 WILSON BLVD./ ARLINGTON VA 22209
22210 JOHN N. LATTI/ P.O. BOX 1297/ ARLINGTON VA 22210
22304 FRANK BREWSTER/ 4701 KENMORE AVE #1009/ ALEXANDRIA VA 22304/ (703) 370-6645
22311 ARNOLD SHORE/ 5021 SEMINARY RD. #1613/ ALEXANDRIA VA 22311/ (703) 379-2247
22314 RONALD S. NAU/ C/O TELEDYNE GEOTECH/ P.O. BOX 334/ ALEXANDRIA VA 22314/ (703) 836-3882
22314 CAROL A. OGDIN/ SOFTWARE TECHNIQUE INC./ 100 POMMANDER WALK/ ALEXANDRIA VA 22314/ (703) 549-0646
22901 LINWOOD FERGUSON/ 741-B MOUNTAINWOOD RD./ CHARLOTTESVILLE VA 22901/ (804) 293-7816
22901 ROBERT A. GIBSON/ WEST LEIGH/ 2380 KINGSTON RD/ CHARLOTTESVILLE VA 22901/ (804) 977-3233
22901 STEPHEN J. HARTLEY/ 2330-20 PEYTON DR./ CHARLOTTESVILLE VA 22901/ (804) 827-2897 (WORK)
22901 TIM HILL/ MEDICAL COMPUTING CENTER/ MEDICAL CENTER BOX 282/ UNIVERSITY OF VIRGINIA/ CHARLOTTESVILLE VA 22901/ (804) 924-5261
22901 TERRENCE PRATT/ DEPT. OF APPLIED MATH/ THORNTON HALL/ UNIV. OF VIRGINIA/ CHARLOTTESVILLE VA 22901/ (804) 924-7201
22903 ATTN: J. F. MCINTYRE - LIBRARIAN/ COMPUTING CENTER/ GILMER HALL/ U OF VIRGINIA/ CHARLOTTESVILLE VA 22903/ (804) 924-3731
22903 DAVID A. MUNDIE/ FRENCH DEPT./ 302 CABELL HALL/ U. OF VIRGINIA/ CHARLOTTESVILLE VA 22903/ (804) 924-7157
23234 WILLIAM C. MOORE JR./ 3518 LUCKYHEE CRESCENT/ RICHMOND VA 23234/ (804) 275-6676
23284 ANN D. DAVIES/ UNIVERSITY COMPUTER CENTER/ VIRGINIA COMMONWEALTH UNIVERSITY/ 1015 FLOYD AVE./ RICHMOND VA 23284/ (804) 770-6339
23508 FRANCES L. VAN SCOY/ DEPT. OF MATH AND COMPUTING SCIENCES/ OLD DOMINION UNIV./ NORFOLK VA 23508/ (804) 489-6522
23602 DAVID A. HOUGH/ 529 HELM DRIVE/ NEWPORT NEWS VA 23602/ (804) 874-3387
23665 J. C. KNIGHT/ LANGLEY RESEARCH CENTER/ M/S 125A/ NASA/ HAMPTON VA 23665
23666 DAVID E. HAMILTON/ 119G PINEWOOD CRESCENT/ HAMPTON VA 23666/ (804) 827-0758
24401 FRED W. POWELL/ INNOVATIVE MANAGEMENT SYSTEMS/ PO BOX 2585 / 865 MIDDLEBROOK AVENUE/ STAUNTON VA 24401/ (703) 885-4950
27514 STEVEN M. BELLOVIN/ DEPT. OF COMP. SCI./ U OF NORTH CAROLINA/ CHAPEL HILL NC 27514/ (919) 933-5698
28743 CHRISTOPHER K. JOHANSEN/ FREEKSHOW ELECTRONWORKS & XOPHER INFOR/ ROUTE 1 BOX 157/ HOT SPRINGS NC 28743/ (704) 622-3423
29206 HOWARD EISENSTEIN/ 6616 DARE CIRCLE/ COLUMBIA SC 29206/ (803) 782-0544
29208 GERALD STEINBACK/ COMPUTER SERVICES DIV./ U. OF SOUTH CAROLINA/ COLUMBIA SC 29208/ (803) 777-6001
29613 T. RAY NANNY/ DEPT. OF COMPUTER SCIENCE/ FURMAN UNIV./ GREENVILLE SC 29613/ (803) 294-2097
30092 GERALD N. CEDERQUIST/ DIGITAL COMMUNICATIONS ASSOC./ 135 TECHNOLOGY PARK/ NORCROSS GA 30092/ (404) 448-1400
30328 M. L. MCGRAW/ 655 SPALDING DR./ ATLANTA GA 30328/ (404) 394-2017
30332 ATTENTION: JERRY W. SEGERS/ OFFICE OF COMPUTING SERVICES/ GEORGIA INSTITUTE OF TECHNOLOGY/ ATLANTA GA 30332/ (404) 894-4676
30332 PHILLIP H. ENSLOW JR./ SCHOOL OF INFO. AND COMP. SCI./ GEORGIA TECH/ ATLANTA GA 30332/ (404) 894-3187
30332 JAMES N. FARMER/ OFFICE OF COMPUTING SERVICES/ GEORGIA TECH/ 225 NORTH AVE. NW/ ATLANTA GA 30332/ (404) 894-4660
30332 JOHN J. GODA JR./ SCHOOL OF INFORMATION AND COMPUTER SCI/ GEORGIA TECH/ ATLANTA GA 30332/ (404) 894-3131
30332 JOHN P. WEST/ OFFICE OF COMPUTING SERVICES/ GEORGIA TECH/ 225 NORTH AVE. N.W./ ATLANTA GA 30332/ (404) 894-4676
32303 C. EDWARD REID/ RT. 7 BOX 1257/ TALLAHASSEE FL 32303/ (904) 488-2451
32304 T. P. BAKER/ DEPT. OF MATH/ 225 LOVE BUILDING/ FLORIDA STATE U/ TALLAHASSEE FL 32304/ (904) 644-2580
32304 TIM LOWERY/ COMPUTING CENTER/ 110 LOVE BUILDING/ FLORIDA STATE UNIVERSITY/ TALLAHASSEE FL 32304/ (904) 644-3860

32306 R. GARY LEE/ COMPUTING CENTER/ 110 LOVE BUILDING/ FLORIDA STATE U/ TALLAHASSEE FL 32306/ (904) 644-2761
 32604 LE H. NGUYEN/ UNIVERSITY OF FLORIDA STATION/ P.O. BOX 12605/ GAINESVILLE FL 32604/ (904) 377-9879 (HOME)/ (904) 392-0907 (OFFICE)
 32611 ATTN: DIRECTOR/ NORTHEAST REGIONAL DATA CENTER/ 253 SSRB/ U OF FLORIDA/ GAINESVILLE FL 32611/ (904) 392-2061
 32611 ATTN: LIBRARIAN/ CIRCA/ 411 WEIL/ U OF FLORIDA/ GAINESVILLE FL 32611/ (904) 392-0907
 32611 JAMES B. CONKLIN JR./ CIRCA/ 411 WEIL HALL/ U. OF FLORIDA/ GAINESVILLE FL 32611
 32806 J. D. GEORGE/ COMPUTER BRANCH/ NAVAL RESEARCH LABORATORY/ P.O. BOX 8337/ ORLANDO FL 32806/ (305) 859-5120
 32901 SAM HARBAUGH/ E.E. DEPT./ FLORIDA INST. OF TECHNOLOGY/ P.O. BOX 1150/ MELBOURNE FL 32901/ (305) 723-3701 X332
 32901 GEORGE A. SEYFERT/ HARRIS CONTROLS DIVISION/ P.O. BOX 430/ MELBOURNE FL 32901/ (305) 727-5675
 32901 TOM SPURRIER/ ELECTRONICS SYSTEMS DIVISION/ HARRIS CORP./ P.O. BOX 37/ MELBOURNE FL 32901
 32901 CASEY TUBBS/ ELECTRONICS SYSTEMS DIVISION/ HARRIS CORP./ P.O. BOX 37/ MELBOURNE FL 32901/ (305) 727-4000
 32901 GEORGE E. HAYNAM/ 556 PARKER ROAD/ W.MELBOURNE FL 32901/ (904) 378-8118
 33307 BOB BRUCE/ COMPUTER SYSTEMS DIV./ MAIL DROP 15/ HARRIS CORPORATION/ 1200 GATEWAY DR./ FT.LAUDERDALE FL 33307/ (305) 974-1700 X235
 33309 ATTN: MOD COMP LIBRARY/ MS #21/ 1650 W. MCNAB ROAD/ FT. LAUDERDAL FL 33309/ (305) 974-1380
 33314 FRED L. SCOTT/ BROWARD COMMUNITY COLLEGE/ 3501 DAVIE ROAD/ FT. LAUDERDAL FL 33314/ (305) 581-8700
 35223 JEFFREY W. GRAHAM/ GRAHAM COMPUTER ENTERPRISES INC./ 3 OFFICE PARK CIR. - SUITE 106/ BIRMINGHAM AL 35223/ (205) 870-7267
 35486 DONALD B. CROUCH/ DEPT.OF COMPUTER SCIENCE/ U. OF ALABAMA/ P.O. BOX 6316/ UNIVERSITY AL 35486/ (205) 348-6363
 35758 PHILIP N. BERGSTRESSER/ 128 JACKSON AVE./ MADISON AL 35758/ (205) 837-2400
 35801 MARVIN E. KURTTI/ 1327 MONTE SANO BLVD. S.E./ HUNTSVILLE AL 35801
 35801 JOHN D. REYNOLDS/ C/O SYSTEM DEVELOPMENT CORP./ 4810 BRADFORD BOULEVARD/ HUNTSVILLE AL 35801/ (205) 837-7610
 35806 ATTENTION: DAVID MADISON/ ADVANCED SOFTWARE TECHNOLOGY DEPT./ TEXAS INSTRUMENTS INC./ 304 WYNN DRIVE/ HUNTSVILLE AL 35806/ (205) 837-7510
 35807 PEI HSIA/ COMPUTER SCIENCE PROGRAM/ U OF ALABAMA AT HUNTSVILLE/ P.O. BOX 1247/ HUNTSVILLE AL 35807/ (205) 895-6088
 37130 SAMUEL T. BAKER/ 1310 STONEWALL BLVD./ MURFREESBORO TN 37130/ (615) 896-3362 (HOME)/ (615) 741-3531 (OFFICE)
 37232 STANLEY B. HIGGINS/ DEPARTMENT OF MEDICINE/ VANDERBILT UNIVERSITY/ NASHVILLE TN 37232/ (615) 322-3384
 37916 ATTENTION: GORDON R. SHERMAN/ COMPUTER CENTER/ 200 STOKELY MGMT. CENTER/ U OF TENNESSEE/ KNOXVILLE TN 37916
 37916 CHARLES PFLEEGER/ COMP. SCI. DEPT./ U OF TENNESSEE/ KNOXVILLE TN 37916/ (615) 974-5067
 38677 ATTN: DEPT. OF COMPUTER SCIENCE/ U OF MISSISSIPPI/ UNIVERSITY MS 38677
 38677 RALPH D. JEFFORDS/ DEPT. OF COMPUTER SCIENCE/ U. OF MISSISSIPPI/ UNIVERSITY MS 38677/ (601) 232-7219 (OFFICE)/ (601) 234-0874 (HOME)
 39210 ROBERT A. SHIVE JR./ MILLSAPS COLLEGE/ STATION A/ JACKSON MS 39210/ (601) 354-5201
 39762 GAY THOMAS/ COMPUTER SCIENCE DEPT./ DRAWER CC/ MISS. STATE MS 39762/ (601) 325-2942
 40208 BRUCE DAWSON/ COMPUTER CENTER -BELKNAP/ COMPUTER AND SYSTEMS BUILDING/ UNIVERSITY OF LOUISVILLE/ LOUISVILLE KY 40208/ (502) 588-6123
 40208 SANDEE MITCHELL/ DEPT. OF APPLIED MATH AND COMPUTER SCI/ U. OF LOUISVILLE/ SPEED SCIENCE S/ LOUISVILLE KY 40208/ (502) 636-6661
 40475 JERRY LEVAN/ DEPT. OF MATH. SCIENCES/ EASTERN KENTUCKY UNIV./ RICHMOND KY 40475/ (606) 622-5782
 40506 LAVINE THRILLKILL/ COMPUTING CENTER/ 72 MCVEY HALL/ U OF KENTUCKY/ LEXINGTON KY 40506/ (606) 258-2916
 40506 M. W. VANNIER/ WENNER-GREN RESEARCH LABORATORY/ U. OF KENTUCKY/ LEXINGTON KY 40506/ (606) 258-8885
 43210 DAVID J. RYPKA/ DEPT. OF COMP. AND INFO. SCI./ OHIO STATE UNIV./ 2036 NEIL AVENUE MALL/ COLUMBUS OH 43210/ (614) 422-7402
 43220 ROY F. REEVES/ 1640 SUSSEX COURT/ COLUMBUS OH 43220/ (614) 422-4843
 43606 BRIAN NELSON/ COMPUTER SERVICES/ U. OF TOLEDO/ 2801 W. BANCROFT STREET/ TOLEDO OH 43606/ (419) 537-2511
 44106 R. B. LAKE/ BIOMETRY/ WEARN BUILDING/ UNIVERSITY HOSPITALS/ CLEVELAND OH 44106/ (216) 791-7300
 44106 FRANK OLYNYK/ CHI CORPORATION/ 11000 CEDAR AVE./ CLEVELAND OH 44106/ (216) 229-6400
 44115 T. S. HEINES/ DEPT. OF COMPUTER SCIENCE/ CLEVELAND STATE UNIVERSITY/ CLEVELAND OH 44115/ (216) 687-4762/ (216) 687-4760
 44139 TOM ZWITTER/ ADVANCED DEVELOPMENT DIV./ BUILDING B/ OHIO NUCLEAR INC./ 6000 COCHRAN RD./ SOLON OH 44139
 44306 JOHN R. LINDSAY/ 1609 SALEM AVE./ AKRON OH 44306/ (216) 784-6814
 44325 ROBERT L. BRIECHLE/ THE COMPUTER CENTER/ U OF AKRON/ 302 E. BUCHTEL AVE./ AKRON OH 44325/ (216) 375-7172
 44691 E. C. ZIMMERMAN/ COMPUTER CENTER/ THE COLLEGE OF WOOSTER/ WOOSTER OH 44691/ (216) 264-1234 X304
 45036 PATRICIA VAN DERZEE/ PROCESS CONTROLS DIVISION/ CINCINNATI MILACRON INC./ LEBANON OH 45036/ (513) 494-5320
 46202 ROBERT J. SNYDER/ GR.FL. UNION BUILDING DATA CENTER/ INDIANA U - PURDUE U AT INDIANAPOLIS/ 1100 WEST MICHIGAN STREET/ INDIANAPOLIS IN 46202
 46637 ATTN: DOCUMENTS ROOM LIBRARIAN/ COMPUTING CENTER/ U OF NOTRE DAME/ NOTRE DAME IN 46637/ (219) 283-7784
 46989 R. WALDO ROTH/ COMPUTER SCIENCE DEPT/ TAYLOR UNIVERSITY/ UPLAND IN 46989/ (317) 998-2751 X269
 47130 ANDREW S. PUCHRIK/ 1803 VILLAGE GREEN BLVD. #94/ JEFFERSONVILL IN 47130/ (812) 283-4059
 47150 DOUGLAS H. QUEBBEMAN/ COMPUTING SERVICES/ INDIANA UNIV. - SOUTHEAST/ 4201 GRANTLINE ROAD/ NEW ALBANY IN 47150/ (812) 945-2731 X287
 47306 GEORGE GRUNWALD/ DEPT. MATH. SCIENCES/ BALL STATE UNIVERSITY/ MUNCIE IN 47306/ (317) 285-6164
 47401 GEORGE COHN III/ 316 N. WASHINGTON/ BLOOMINGTON IN 47401/ (812) 337-9255/ (812) 337-1911
 47401 ANTHONY J. SCHAEFFER/ 3510 DUNSTAN DR/ BLOOMINGTON IN 47401/ (812) 334-1163/ (812) 337-9137
 47401 LAURA SNYDER/ 402 E. 17TH/ BLOOMINGTON IN 47401
 47401 HAL STEIN/ BOX 102 WRIGHT QUAD/ INDIANA UNIVERSITY/ BLOOMINGTON IN 47401/ (812) 337-7081
 47401 ALFRED I. TOWELL/ WRUBEL COMPUTER CENTER/ INDIANA UNIVERSITY/ BLOOMINGTON IN 47401/ (812) 337-1911
 47401 DAVID S. WISE/ COMPUTER SCIENCE DEPT./ 101 LINDLEY HALL/ INDIANA U/ BLOOMINGTON IN 47401/ (812) 337-4866
 47401 STEPHEN W. YOUNG/ WRUBEL COMPUTER CENTER/ HPER BUILDING/ INDIANA UNIVERSITY/ BLOOMINGTON IN 47401/ (812) 337-1911
 47902 JAMES R. MILLER/ P.O. BOX 1141/ LAFAYETTE IN 47902/ (317) 494-8232 (OFFICE)
 47906 KENNETH LEROY ADAMS/ 927 N. SALISBURY ST./ W. LAFAYETTE IN 47906/ (317) 743-9905 (HOME)/ (317) 493-9407 OR 494-8232 (WORK)
 47906 DAN DORROUGH/ 400 NORTH RIVER RD. - 1018/ W. LAFAYETTE IN 47906/ (317) 493-9408
 47907 DOUGLAS COMER/ COMPUTER SCIENCES DEPT./ 402 MATH BLDG./ PURDUE UNIVERSITY/ W. LAFAYETTE IN 47907/ (317) 493-3327
 47907 DOROTHY E. DENNING/ COMPUTER SCIENCES DEPT./ 442 MATH SCIENCES BLDG./ PURDUE UNIVERSITY/ W. LAFAYETTE IN 47907
 47907 JOSEPH H. FASEL III/ COMPUTER SCIENCES/ 442 MATH SCIENCES BUILDING/ PURDUE UNIVERSITY/ W. LAFAYETTE IN 47907/ (317) 494-8566
 47907 EDWARD F. GEHRINGER/ DEPT. OF COMPUTER SCIENCE/ MATH SCIENCES BUILDING/ PURDUE UNIVERSITY/ W. LAFAYETTE IN 47907
 48103 ALAN A. KORTESOJA/ 701 W. DAVIS/ ANN ARBOR MI 48103/ (313) 995-6124/ (313) 995-6000

48105 JOHN S. GOURLAY/ 1413 MCINTYRE/ ANN ARBOR MI 48105/ (313) 994-6645
 48106 NEIL J. BARTA/ ADP NETWORK SERVICES/ 175 JACKSON PLAZA/ ANN ARBOR MI 48106/ (313) 769-6800
 48106 CHARLES G. MOORE/ NETWORK SERVICES INC./ 175 JACKSON PLAZA/ ANN ARBOR MI 48106/ (313) 426-2620
 48106 PAUL R. TEETOR/ OPER. SYS. GROUP/ ADP NETWORK SERVICES/ 175 JACKSON PLAZA/ ANN ARBOR MI 48106/ (313) 769-6800
 48107 DAVID LIPPINCOTT/ INFORMATION CONTROL SYSTEMS/ 313 N. FIRST STREET/ ANN ARBOR MI 48107/ (313) 761-1600 EXT. 40
 48109 PAUL PICKELMANN/ 2217 CROSS/ 1440 HUBBARD ST./ ANN ARBOR MI 48109/ (313) 764-2121
 48109 LOUIS F. WOJNAROSKI/ MENTAL HEALTH RESEARCH INST./ U. OF MICHIGAN/ ANN ARBOR MI 48109/ (303) 763-1143
 48109 KARL L. ZINN/ CTR. FOR RESEARCH ON LEARNING & TEACH/ UNIV. OF MICHIGAN/ 109 EAST MADISON STREET/ ANN ARBOR MI 48109
 48127 L. RICHARD LEWIS/ 5806 COOLIDGE ROAD/ DEARBORN MI 48127/ (313) 274-6871
 48130 GREGORY J. WINTERHALTER/ 3825 NORTH ZEEB/ DEXTER MI 48130
 48202 WILLIAM GROSZY/ MATH DEPT - COMP. SCI. SECTION/ WAYNE STATE UNIVERSITY/ DETROIT MI 48202
 48221 RONALD G. MOSIER/ 17596 WILDEMERE/ DETROIT MI 48221/ (313) 956-2417
 48228 R. NEIL FAIMAN JR./ 8235 APPOLINE/ DETROIT MI 48228/ (513) 834-3065
 48823 MARK HERSEY/ 323 VILLAGE DRIVE APT. 534/ EAST LANSING MI 48823/ (517) 351-5703 (HOME)/ (517) 355-1764 (OFFICE)
 48823 THOMAS W. SKELTON/ 315 WEST SAGINAW STREET/ EAST LANSING MI 48823/ (517) 332-4368/ (517) 351-2530
 48823 THOMAS C. SOCOLOFSKY/ SYSTEMS RESEARCH INC/ 241 E. SAGINAW/ EAST LANSING MI 48823/ (517) 351-2530 (OFFICE)/ (517) 351-2530 (HOME)
 48824 JOHN B. EULENBERG/ COMP. SCI. DEPT./ MICHIGAN STATE U/ EAST LANSING MI 48824/ (517) 353-0831
 48824 STEVEN L. HUYSER/ COMPUTER LABORATORY/ MICHIGAN STATE U/ EAST LANSING MI 48824/ (517) 353-1800
 48824 MARK RIORDAN/ USER SERVICES/ COMPUTER LABORATORY/ MICHIGAN STATE UNIVERSITY/ EAST LANSING MI 48824/ (517) 353-1800
 48824 H. G. HEDGES/ DEPT. OF COMP. SCI./ MICHIGAN STATE U/ E. LANSING MI 48824/ (517) 353-6484
 48910 ALLAN MOLUF/ 3410 DAVIDSON/ LANSING MI 48910/ (517) 393-8639
 49007 MARK T. O'BRYAN/ PRESTIGE APARTMENT E/ 421 STANWOOD DRIVE/ KALAMAZOO MI 49007
 49008 MARK C. KERSTETTER/ DEPT. OF MATHEMATICS/ WESTERN MICHIGAN UNIVERSITY/ KALAMAZOO MI 49008/ (616) 383-6165
 49008 JACK R. MEAGHER/ COMPUTER SCIENCE AND MATHEMATICS/ WESTERN MICHIGAN UNIV./ KALAMAZOO MI 49008/ (616) 383-0095
 49401 GORDON A. STEGINK/ COMPUTER CENTER/ 325 MANITOU HALL/ GRAND VALLEY STATE COLLEGE/ ALLENDALE MI 49401/ (616) 895-6611 X571
 50011 GEORGE O. STRAWN/ DEPT. OF COMPUTER SCIENCE/ IOWA STATE U/ AMES IA 50011/ (515) 294-2259
 50112 TOM MOBERG/ ACADEMIC COMPUTING/ GRINNELL COLLEGE/ GRINNELL IA 50112/ (515) 236-6521
 50309 LARRY CRANE/ ELECTRONIC DATA SYSTEMS CORP./ 1200 LOCUST/ DES MOINES IA 50309
 50311 MIKE BURGHER/ DIAL COMPUTER CENTER/ DRAKE UNIVERSITY/ 24TH AND CARPENTER/ DES MOINES IA 50311/ (515) 271-3918
 52101 EDWARD O. THORLAND/ COMPUTER CENTER/ LUTHER COLLEGE/ DECORAH IA 52101/ (319) 387-1043
 52242 ATTN: SERIALS DEPT./ UNIVERSITY LIBRARIES/ UNIVERSITY OF IOWA/ IOWA CITY IA 52242
 52242 ATTN: UCC LIBRARIAN/ UNIVERSITY COMPUTER CENTER/ LCM/ UNIVERSITY OF IOWA/ IOWA CITY IA 52242/ (319) 353-3170
 53115 MICHAEL A. BEAVER/ INSTRUMENTS DIVISION/ BUNKER RAMO/ 902 WISCONSIN ST./ DELAVEN WI 53115
 53201 JAMES S. BOTIC/ POST OFFICE BOX 423 MS/51/ JOHNSON CONTROLS INC./ 507 EAST MICHIGAN STREET/ MILWAUKEE WI 53201/ (414) 276-9200
 53211 W. A. HINTON/ 3469 N. CRAMER ST./ MILWAUKEE WI 53211/ (414) 964-2671 (HOME)/ (414) 963-4005 (OFFICE)
 53211 BROOKS DAVID SMITH/ 4473 N. NEWHALL ST./ SHOREWOOD WI 53211/ (414) 963-6413
 53219 JOHN G. DOBNICK/ 3171 S. 83 ST./ MILWAUKEE WI 53219/ (414) 963-5727
 53703 HERMAN BERG/ 108 E. DAYTON/ MADISON WI 53703/ (608) 255-8545
 53705 KEVIN W. CARLSON/ 1820 SUMMIT AVE/ MADISON WI 53705/ (608) 238-3441
 53705 EDWARD H. HARRIS/ SYNNOVATION INC./ 2106 BASCOM ST./ MADISON WI 53705/ (608) 233-1984
 53706 ATTN: FRIEDA S. COHEN/ ACADEMIC COMPUTING CENTER/ U OF WISCONSIN/ 1210 W. DAYTON ST./ MADISON WI 53706
 53706 CHARLES N. FISCHER/ MACC/ U OF WISCONSIN/ 1210 WEST DAYTON ST./ MADISON WI 53706/ (608) 262-7870
 53706 FRANK H. HORN/ ACADEMIC COMPUTER CENTER/ U OF WISCONSIN/ 1210 WEST DAYTON STREET/ MADISON WI 53706/ (608) 262-9841
 53706 RICHARD LEBLANC/ MADISON ACADEMIC COMPUTER CENTER/ U OF WISCONSIN/ 1210 W. DAYTON STREET/ MADISON WI 53706/ (608) 262-0138
 54302 ED GLASER/ COMPUTING SERVICES/ U OF WISCONSIN - GREEN BAY/ GREEN BAY WI 54302/ (414) 465-2309
 54701 DAVID A. NUESSE/ DEPARTMENT OF COMPUTER SCIENCE/ U OF WISCONSIN - EAU CLAIRE/ EAU CLAIRE WI 54701/ (715) 836-2526
 54701 RUDOLPH C. POLENZ/ INFORMATION SYSTEMS AND COMPUTING SERV/ U OF WISCONSIN - EAU CLAIRE/ EAU CLAIRE WI 54701/ (715) 836-4428
 54701 BRUCE A. PUMPLIN/ DEPT OF COMPUTER SCIENCE/ U OF WISCONSIN - EAU CLAIRE/ EAU CLAIRE WI 54701/ (715) 836-2315
 55057 CARL HENRY/ COMPUTER CENTER/ CARLETON COLLEGE/ NORTHFIELD MN 55057/ (507) 645-4431 X504
 55057 TIMOTHY W. HOEL/ ACADEMIC COMPUTER CENTER/ ST. OLAF COLLEGE/ NORTHFIELD MN 55057/ (507) 663-3096
 55068 CHRIS BOYLAN/ 14620 BISCAYNE WAY/ ROSEMOUNT MN 55068/ (612) 423-1922
 55101 JOHN E. COLLINS/ BLDG 235 F247/ 3M CENTER/ ST. PAUL MN 55101/ (612) 736-0778
 55101 GLENN FISHBINE/ GCCPC/ CCP/ 444 LAFAYETTE RD./ ST. PAUL MN 55101/ (612) 296-7543
 55104 GEOFF WATTLES/ P.O. BOX 4244/ ST. PAUL MN 55104/ (612) 331-7087
 55108 GEORGE GONZALEZ/ 1435 W. JESSAMINE APT. #305/ ST. PAUL MN 55108/ (612) 647-0976
 55108 JAMES KREILICH/ 1408 ALBANY AVE./ ST. PAUL MN 55108/ (612) 644-1375
 55109 GLENN MILLER/ 2317 N. HENRY ST./ N. ST. PAUL MN 55109/ (612) 777-2483
 55112 DARRELL L. WONDRA/ ARH254/ CONTROL DATA CORP./ 4201 LEXINGTON AVE. N./ ARDEN HILLS MN 55112/ (612) 482-2542 (OFFICE)/ (612) 484-3804 (HOME)
 55112 PAUL K. HUNTWORK/ CONTROL DATA CORP./ 4201 LEXINGTON AVE. N./ ST. PAUL MN 55112/ (612) 482-2772
 55112 RUSS PETERSON/ ARH254/ CONTROL DATA CORP./ 4201 N. LEXINGTON/ ST. PAUL MN 55112/ (612) 482-2548
 55112 MARK RUSTAD/ 585 HARRIET AVE #213/ ST. PAUL MN 55112/ (612) 483-0589
 55113 KEVIN HAUSMANN/ MINNESOTA EDUCATIONAL COMPUTING CONSOR/ 2520 W. BROADWAY/ LAUDERDALE MN 55113/ (612) 376-1119
 55113 SUE PETERSON/ COMTEN INC./ 1950 W. COUNTY RD. B2/ ROSEVILLE MN 55113/ (612) 633-8130 X249
 55113 ROBERT D. VAVRA/ 741 TERRACE DRIVE/ ROSEVILLE MN 55113/ (612) 483-6123
 55113 STEVEN W. WEINGART/ MS 4753/ SPERRY-UNIVAC/ 2276 HIGHCREST DRIVE/ ROSEVILLE MN 55113/ (612) 633-6170 X3748
 55165 ATTENTION: ROBERT E. NOVAK/ DSPL DEVELOPMENT GROUP/ SPERRY UNIVAC/ UNIVAC PARK / P.O. BOX 3525/ ST. PAUL MN 55165/ (612) 456-5551

55165 ROBERT A. LAWLER/ MS U2M23/ UNIVAC PARK/ P.O. BOX 3525/ ST. PAUL MN 55165/ (612) 456-3107
55165 LEO J. SLECHTA/ DSD/ SPERRY UNIVAC/ BOX 3525 MS U1U25/ ST. PAUL MN 55165/ (612) 456-2743
55165 RAYMOND YOUNG/ M.S. U2U22/ SPERRY UNIVAC/ P.O. BOX 3525/ ST. PAUL MN 55165/ (612) 456-5517
55303 DAVID HELFINSTINE/ 1136 5TH AVENUE SOUTH/ ANOKA MN 55303/ (612) 421-8964
55337 HAROLD DE VORE/ 13401 MORGAN AVE. SOUTH APT. 321/ BURNSVILLE MN 55337/ (701) 746-6977
55343 PAUL CHRISTOPHERSON/ M.S. MN11-1611/ HONEYWELL INC./ 600 SECOND STREET N./ HOPKINS MN 55343/ (612) 542-6438
55343 GENE H. OLSON/ 421 COUNTY ROAD 3 APT 512/ HOPKINS MN 55343/ (612) 938-2454/ 941-5560 X429 (WORK)
55343 ROSS D. SCHMIDT/ MN 11-2120/ HONEYWELL INC./ 600 2ND ST. NO.E./ HOPKINS MN 55343/ (612) 542-6741
55401 MARK BILODEAU/ ENGINEERING SYSTEMS 4TH FLOOR/ NORTHERN STATES POWER/ 414 NICOLLET MALL/ MINNEAPOLIS MN 55401/ (612) 330-6749/ (612) 330-5899
55401 CHRIS EASTLUND/ ENGINEERING SYSTEMS 4TH FLOOR/ NORTHERN STATES POWER/ 414 NICOLLET MALL/ MINNEAPOLIS MN 55401/ (612) 330-6749/ (612) 330-5899
55404 RICK L. MARCUS/ 1609 11TH AVE. S./ MINNEAPOLIS MN 55404/ (612) 339-1638
55404 JOHN STANLEY/ 607 S. 9TH ST./ MINNEAPOLIS MN 55404/ (612) 339-1728
55406 BRUCE M. SORLIE/ 2810 29TH AVE. S./ MINNEAPOLIS MN 55406/ (612) 729-4435
55406 INDULIS VALTERS/ 2810 E. 22ND STREET/ MINNEAPOLIS MN 55406/ (612) 341-4430 (HOME)
55408 ABDUL RASQA BELLO/ P.O. BOX 8681/ MINNEAPOLIS MN 55408/ (612) 330-4106
55409 DON HANNES/ 4215 PLEASANT AVE. SO./ MINNEAPOLIS MN 55409/ (612) 823-3030
55413 WILLIAM C. MARSHALL/ SYSTEMS AND RESEARCH CENTER/ MN-17-2321/ HONEYWELL INC./ 2700 RIDGWAY PARKWAY/ MINNEAPOLIS MN 55413/ (612) 378-4501
55413 BELLE SHENOY/ MS MN17-1649/ HONEYWELL INC./ 2600 RIDGWAY ROAD/ MINNEAPOLIS MN 55413/ (612) 378-5418
55413 STANLEY C. VESTAL/ MS 2340/ HONEYWELL INC./ 2600 RIDGWAY PKWY./ MINNEAPOLIS MN 55413/ (612) 378-5046
55414 ATTN: KAPPA ETA KAPPA/ 330 11TH AVE. S.E./ MINNEAPOLIS MN 55414/ (612) 331-2133
55414 KEVIN R. DRISCOLL/ 330 SE 11TH AVENUE/ MINNEAPOLIS MN 55414/ (612) 331-2133
55414 JOHN FUNG/ 425 13TH AVE S.E. #1502/ MINNEAPOLIS MN 55414/ (612) 376-5464 (OFFICE)/ (612) 378-0427 (HOME)
55414 GARY M. JACKSON/ 1008 27TH AVE. SE. APT.A/ MINNEAPOLIS MN 55414/ (612) 378-2178
55414 WALT PERKO/ 727 15TH AVE. S.E./ MINNEAPOLIS MN 55414/ (612) 331-6984
55416 WARREN STENBERG/ 2012 CEDAR LAKE PKWY/ MINNEAPOLIS MN 55416/ (612) 920-7465
55417 KEITH HAUER-LOWE/ 4819 COLUMBUS AVE. SO./ MINNEAPOLIS MN 55417/ (612) 633-6170 X3362 (WORK)/ (612) 824-8026 (HOME)
55420 RICHARD HENDRICKSON/ CRAY RESEARCH INC./ 7850 METRO PARKWAY SUITE 213/ MINNEAPOLIS MN 55420/ (612) 854-7472
55421 STEVEN N. TRAPP/ 5020 MULCARE DR/ COLUMBIA HTS. MN 55421/ (612) 571-5020
55421 WILLIAM T. WOOD/ 3820 MACALASTER DR. NE #311/ MINNEAPOLIS MN 55421/ (612) 788-2390
55422 CALVIN STEVENS/ 4936 SORELL AVE. N./ MINNEAPOLIS MN 55422/ (612) 588-7724
55423 KEITH BOLSON/ 7425 17TH AVE. SO./ RICHFIELD MN 55423/ (612) 866-4658
55424 JOHN ALSTRUP/ INTERDATA/ 4620 VALLEY VIEW ROAD/ EDINA MN 55424/ (612) 854-4264
55424 ROBERT A. STRYK/ 5441 HALIFAX LANE/ EDINA MN 55424/ (612) 920-5434 (HOME)/ (612) 887-4356 (OFFICE)
55425 RON THOMAS/ DATA 100 CORPORATION/ 7725 WASHINGTON AVE. S./ MINNEAPOLIS MN 55425/ (612) 941-6500
55427 RICHARD HOYME/ 1404 KELLY DR. N./ MINNEAPOLIS MN 55427/ (612) 545-4642
55427 HUGO MEISSER/ 3021 WISCONSIN AVE. N./ MINNEAPOLIS MN 55427/ (612) 544-2349
55431 JACK ANDERSON/ HART ENGINEERING CO. INC./ 9341 PENN AVENUE SOUTH/ BLOOMINGTON MN 55431/ (612) 881-8464
55435 JONATHON R. GROSS/ CYTROL INC./ 4510 W. 77TH ST./ EDINA MN 55435/ (612) 835-4884
55437 DENNIS NICKOLAI/ SOUTHGATE OFFICE PLAZA/ CONTROL DATA CORPORATION/ 5001 W. 80TH ST./ BLOOMINGTON MN 55437/ (612) 830-6609
55440 RANDALL W. HANSEN/ HQS06B/ CONTROL DATA CORPORATION/ P.O. BOX 0/ MINNEAPOLIS MN 55440/ (612) 853-5466
55440 JON HANSON/ SYSTEM DEVELOPMENT/ DATA 100 CORP/ BOX 1222/ MINNEAPOLIS MN 55440/ (612) 941-6500
55440 GENE MARTINSON/ SYSTEM DEVELOPMENT/ DATA 100 CORP/ BOX 1222/ MINNEAPOLIS MN 55440/ (612) 941-6500
55440 DOUG PIHL/ SYSTEM DEVELOPMENT/ DATA 100 CORP/ BOX 1222/ MINNEAPOLIS MN 55440/ (612) 941-6500
55440 BILL SIMMONS/ SYSTEM DEVELOPMENT/ DATA 100 CORP/ BOX 1222/ MINNEAPOLIS MN 55440/ (612) 941-6500
55440 RICHARD SPELLERBERG/ SYSTEM DEVELOPMENT/ DATA 100 CORP/ BOX 1222/ MINNEAPOLIS MN 55440/ (612) 941-6500
55440 JERRY STODDARD/ SYSTEM DEVELOPMENT/ DATA 100 CORP/ BOX 1222/ MINNEAPOLIS MN 55440/ (612) 941-6500
55440 TOM URSIN/ SYSTEM DEVELOPMENT/ DATA 100 CORP/ BOX 1222/ MINNEAPOLIS MN 55440/ (612) 941-6500
55440 JAMES A. VELLENGA/ SYSTEM DEVELOPMENT/ DATA 100 CORP/ BOX 1222 / MINNEAPOLIS MN 55440/ (612) 941-6500 X227
55440 JIM VERNON/ SYSTEM DEVELOPMENT/ DATA 100 CORP/ BOX 1222/ MINNEAPOLIS MN 55440/ (612) 941-6500
55441 DAVID C. MESSER/ 3205 N. HARBOR LANE APT 4301/ PLYMOUTH MN 55441
55441 MIKE TILLER/ 2501 N. LANCASTER LN. #178/ PLYMOUTH MN 55441/ (612) 546-6687
55454 TIM BONHAM/ D605/1630 S. 6TH ST./ MINNEAPOLIS MN 55454/ (612) 339-4405
55454 JACK LAFFE/ 320 19TH AVE. S./ MINNEAPOLIS MN 55454/ (612) 336-4946
55454 R. K. NORDIN/ 1615 SOUTH 4TH ST. APT.M3607/ MINNEAPOLIS MN 55454/ (612) 339-5232 (HOME)/ (612) 482-3751 (OFFICE)
55455 ATTENTION: PAUL C. SMITH/ CONSULTING GROUP ON INSTRUCTIONAL DESIGN/ 205 ELLIOTT HALL/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-5352
55455 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
55455 ATTN: COMPUTER SCIENCE DEPT./ 114 LIND HALL/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-0132
55455 ATTN: REFERENCE ROOM/ UNIVERSITY COMPUTER CENTER/ 227 EXP ENGR / U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-7744
55455 SCOTT BERTILSON/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENGR./ U OF MINNESOTA/ MINNEAPOLIS MN 55455/ (612) 376-5262 (WORK)/ (612) 729-0059 (HOME)
55455 BRADFORD E. BLASING/ 1308 CENTENNIAL HALL/ UNIVERSITY OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 376-6053
55455 KEN BORGENDALE/ C.SCL. DEPT./ 114 LIND HALL/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 824-3389
55455 JEFFREY J. DRUMMOND/ UNIVERSITY COMPUTER CENTER/ LAUDERDALE/ U OF MINNESOTA/ MINNEAPOLIS MN 55455/ (612) 373-4573
55455 RON DYKSTRA/ WEST BANK COMPUTER CENTER/ 93B BLEGEN HALL/ UNIVESTY OF MINNESOTA/ WEST BANK/ MINNEAPOLIS MN 55455/ (612) 373-3608
55455 JOHN T. EASTON/ SSRFC/ 25E BLEGEN HALL/ U OF MINNESOTA/ WEST BANK/ MINNEAPOLIS MN 55455/ (612) 373-5599/ (612) 373-7525

55455 LINCOLN FETCHER/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENGR./ U OF MINNESOTA/ MINNEAPOLIS MN 55455/ (612) 376-1637
 55455 KEVIN FJELSTED/ UNIVERSITY COMPUTER CENTER/ 227 EXP ENGR/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-4181
 55455 K. FRANKOWSKI/ COMPUTER SCIENCE DEPARTMENT/ 110H LIND HALL/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-7591
 55455 SARA K. GRAFFUNDER/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENGR. / U OF MINNESOTA/ MINNEAPOLIS MN 55455/ (612) 376-5262
 55455 KRISTINA GREACEN/ C.SCI. DEPT./ 114 LIND HALL/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455
 55455 JOEL M. HALPERN/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENGR./ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-4181
 55455 BRIAN HANSON/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENGR./ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 376-5262 (OFFICE)
 55455 THEA D. HODGE/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENGR./ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-4599
 55455 TIMOTHY J. HOFFMANN/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENGR. / U OF MINNESOTA/ MINNEAPOLIS MN 55455/ (612) 926-9330 (HOME)/ (612) 376-5262 (WORK)
 55455 PETER YAN-TEK HSU/ 475 FRONTIER HALL/ U OF MINNESOTA/ EAST BAN K/ MINNEAPOLIS MN 55455/ (612) 373-7052
 55455 PATRICK L. JARVIS/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENGR./ U OF MINNESOTA/ MINNEAPOLIS MN 55455/ (612) 376-1763
 55455 GEORGE D. JELATIS/ BOX 15 MAYO/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-8941
 55455 MITCHELL R. JOELSON/ SSRFC/ 25 BLEGEN HALL/ U OF MINNESOTA/ WEST BANK/ MINNEAPOLIS MN 55455/ (612) 373-9914/ (612) 373-5599
 55455 DAN LALBERTE/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENGR./ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-4181
 55455 LAWRENCE A. LIDDIARD/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENG. BLDG./ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-5239
 55455 DENNIS R. LIENKE/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENGR./ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-1572
 55455 SHIHTA LIN/ UNIVERSITY COMPUTER CENTER/ 227 EXP ENGR/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-4886
 55455 JOHN E. LIND/ 139 TERRITORIAL HALL/ UNIVERSITY OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455
 55455 MICHAEL MEISSNER/ C.SCI. DEPT./ 114 LIND HALL/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455
 55455 ANDY MICKEL/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENGR./ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 376-7290
 55455 JAMES F. MINER/ SSRFC/ 25 BLEGEN HALL/ U OF MINNESOTA/ WEST BANK/ MINNEAPOLIS MN 55455/ (612) 373-9916
 55455 TOM MOHER/ COMPUTER SCIENCE DEPT./ 114 LIND HALL/ UNIV. OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-7746
 55455 JOHN NAUMAN/ 901 MIDDLEBROOK HALL/ U OF MINNESOTA/ WEST BANK/ MINNEAPOLIS MN 55455/ (612) 376-6596
 55455 DAVID PERLMAN/ COMPUTER SCIENCE DEPARTMENT/ 114 LIND HALL/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-7581
 55455 MICHAEL PRIETULA/ MISRC/ 93 BLEGEN HALL/ U OF MINNESOTA/ WEST BANK/ MINNEAPOLIS MN 55455/ (612) 373-4973
 55455 TIMOTHY J SALO/ UNIVERSITY COMPUTER CENTER/ LAUDERDALE/ U OF MINNESOTA/ MINNEAPOLIS MN 55455/ (612) 376-5607
 55455 BOB SCARLETT/ PHYSICS DEPT./ 148 PHYSICS/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-0243
 55455 G. MICHAEL SCHNEIDER/ C.SCI. DEPT./ 114 LIND HALL/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-7582
 55455 JOHN P. STRAIT/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENGR./ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 376-7290
 55455 JOHN URBANSKI/ WEST BANK COMPUTER CENTER/ BLEGEN HALL/ U OF MINNESOTA/ MINNEAPOLIS MN 55455/ (612) 377-3198/ (612) 373-3608 (WORK)
 55455 KAREN WAGGONER/ UNIVERSITY COMPUTER CENTER/ 129 SPACE SCIENCE CENTER - SICL/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-5768
 55455 WARREN J. WARWICK/ DEPT. OF PEDIATRICS/ BOX 184 MAYO/ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-8886
 55455 PETER H. ZECHMEISTER/ UNIVERSITY COMPUTER CENTER/ 227 EXP. ENG R./ U OF MINNESOTA/ EAST BANK/ MINNEAPOLIS MN 55455/ (612) 373-4181
 55455 ATTN: SSRFC LIBRARY/ SSRFC/ 25 BLEGEN HALL/ U OF MINNESOTA/ WEST BANK/ MINNEAPOLIS MN 55455/ (612) 373-5599
 55792 DAVID SARANEN/ 117 7TH ST. SO./ VIRGINIA MN 55792/ (218) 741-1378
 55812 ATTENTION: DAN BURROWS/ UMD COMPUTER CENTER/ 178 M.W.ALWORTH HALL/ U OF MINNESOTA - DULUTH/ DULUTH MN 55812/ (218) 726-7587
 55812 MARK LUKER/ DEPT. OF MATH SCIENCES/ U OF MINNESOTA - DULUTH/ DULUTH MN 55812/ (218) 726-8240
 55901 L. W. YOUNGREN/ 1505 N.W. 41ST ST. APT. 18F/ ROCHESTER MN 55901/ (507) 285-9696
 55987 GERALD W. CICHANOWSKI/ DEPT. COMPUTER SCIENCE/ ST. MARY'S COLLEGE/ P.O. BOX 56/ WINONA MN 55987/ (507) 452-4430 X229
 56201 JAMES F. MARTINSON/ 1210 WILLMAR AVE/ WILLMAR MN 56201/ (612) 796-2342
 56267 ANDY LOPEZ/ COMPUTER CENTER/ U OF MINNESOTA - MORRIS/ MORRIS MN 56267/ (612) 589-1665 X321
 56301 LARRY GROVER/ 330 ANGUSHIRE APTS #127 RT 7/ ST. CLOUD MN 56301 / (612) 252-0290
 56301 PAUL HELVIG/ 314 4TH AVE. S./ ST. CLOUD MN 56301/ (612) 253-8081
 56301 R. WARREN JOHNSON/ DEPT. OF MATH AND COMP. SCI./ ST. CLOUD STATE U/ ST. CLOUD MN 56301/ (612) 255-2147
 56560 C R CORNER/ 514 SOUTH 9TH ST/ MOORHEAD MN 56560/ (218) 233-1134
 58202 R. I. JOHNSON/ COMP. SCI. DEPT./ U OF NORTH DAKOTA/ BOX 8181 UNIVERSITY STATION/ GRAND FORKS ND 58202/ (701) 777-4107
 58501 GARY J. BOOS/ 517 N. 7TH STREET/ BISMARCK ND 58501/ (701) 223-0441 (WORK)
 59715 ATTN: COMPUTING CENTER/ MONTANA STATE UNIVERSITY/ BOZEMAN MT 59715
 59717 JAMES C. WILLIAMS/ COMPUTING CENTER/ MONTANA STATE UNIVERSITY/ BOZEMAN MT 59717/ (406) 994-3042
 59801 ATTN: COMPUTER SCIENCE DEPARTMENT/ UNIVERSITY OF MONTANA/ MISSOULA MT 59801/ (406) 243-2883
 60015 MARK S. NIEMCZYK/ HEWITT ASSOCIATES/ 102 WILMOT ROAD/ DEERFIELD IL 60015/ (312) 945-8000
 60030 DANIEL M. O'BRIEN/ 665 PIERCE CT./ GRAYSLAKE IL 60030
 60076 JOSEPH LACHMAN/ COMPUTER SYSTEMS AND SOFTWARE/ LACHMAN ASSOCIATES/ 8931 BRONX AVENUE/ SKOKIE IL 60076/ (312) 674-5685 (WORK)
 60201 FRED E. BALLARD/ 2139 LINCOLNWOOD DRIVE/ EVANSTON IL 60201/ (312) 491-0951 (HOME)/ (312) 822-7921 (WORK)
 60201 JOHN L. NORSTAD/ VOGELBACK COMPUTING CENTER/ NORTHWESTERN UNIVERSITY/ 2129 SHERIDAN RD./ EVANSTON IL 60201/ (312) 492-5369
 60201 ALBERT STEINER/ VOGELBACK COMPUTING CENTER/ NORTHWESTERN U/ 2129 SHERIDAN ROAD/ EVANSTON IL 60201/ (312) 492-3682
 60202 BRIT J. BARTTER/ 850A FOREST AVENUE/ EVANSTON IL 60202
 60439 MARTIN R. KRAIMER/ B221-B247/ ARGONNE NATIONAL LAB./ 9700 S. CASS AVE./ ARGONNE IL 60439/ (312) 739-7711 X3660
 60439 TRUMAN C. PEWITT/ APPLIED MATH DIVISION/ BLDG. 221/ ARGONNE NATIONAL LABORATORY/ 9700 SOUTH CASS AVENUE/ ARGONNE IL 60439/ (312) 739-7711
 60532 TERRY E. WEYMOUTH/ 4702 BEAU BIEN LANE EAST/ LISLE IL 60532
 60604 JONATHAN SACHS/ TRANS UNION SYSTEMS CORPORATION/ 111 WEST JACKSON BLVD/ CHICAGO IL 60604/ (312) 431-3330
 60625 DAVID E. CARLTON/ DEPT. OF INFO. SCI./ NORTHEASTERN ILLINOIS U/ 5500 N. ST. LOUIS AVE./ CHICAGO IL 60625
 61738 MIKE LEMON/ 168 WEST THIRD STREET/ EL PASO IL 61738/ (309) 527-4342
 61801 ATTN: CONSULTING OFFICE/ COMPUTING SERVICES OFFICE/ 116 DIGITAL COMPUTER LAB/ U OF ILLINOIS/ URBANA IL 61801/ (217) 333-6133
 61801 RICHARD BALOCCA/ 114B DIGITAL COMPUTER LAB/ U OF ILLINOIS/ URBANA IL 61801/ (217) 344-5284
 61801 ROGER GULBRANSON/ PHYSICS DEPT./ U OF ILLINOIS/ URBANA IL 61801/ (217) 344-4162 (HOME)/ (217) 333-3191 (OFFICE)

61801 M. D. MICKUNAS/ 297 DCL/ U OF ILLINOIS/ URBANA IL 61801/ (217) 333-6351
61801 CARLTON MILLS/ MILLS INTERNATIONAL/ 203 NORTH GREGORY/ URBANA IL 61801/ (217) 328-2436 (HOME)
61820 ATTN: RECEIVING CLERK/ CERL - SOC/ U.S. ARMY/ P.O. BOX 4005/ CHAMPAIGN IL 61820/ (217) 352-6511
61820 FRED P. BAKER/ 302 E. GREGORY/ CHAMPAIGN IL 61820/ (217) 344-7511
61820 AVRUM ITZKOWITZ/ 505 E. CLARK APT. 22/ CHAMPAIGN IL 61820/ (217) 359-9644 (HOME)/ (217) 352-6511 (WORK)
62025 WALT PARRILL/ MID. ILLINOIS COMPUTER CO-OP/ COTTONWOOD ROAD/ EDWARDSVILLE IL 62025/ (618) 288-7268
62708 DONALD S. KLETT/ SANGAMON STATE UNIV./ SPRINGFIELD IL 62708/ (217) 786-6549
62901 THOMAS MELLMAN/ 603-1/2 S. WASHINGTON/ CARBONDALE IL 62901/ (618) 457-2708
63110 GERALD C. JOHNS/ COMPUTER SYSTEMS LAB/ WASHINGTON UNIVERSITY/ 724 S. EUCLID AVENUE/ ST. LOUIS MO 63110/ (314) 454-3395
63110 JOAN ZIMMERMAN/ MUMPS USERS' GROUP/ BIOMEDICAL COMPUTER LABORATORY/ 700 SOUTH EUCLID/ ST. LOUIS MO 63110/ (314) 454-3364
63188 LEE POTTS/ ATTN: DRXAL-TL/ DARCOM ALMSA/ P.O. BOX 1578/ ST. LOUIS MO 63188/ (314) 268-2786
64108 LARRY D. LANDIS/ UNITED COMPUTING SYSTEMS/ 2525 WASHINGTON/ KANSAS CITY MO 64108/ (816) 942-6063
64108 JEFFERY M. RAZAFSKY/ UNITED COMPUTING SYSTEMS INC./ 500 W. 26TH STREET/ KANSAS CITY MO 64108/ (816) 221-9700
64108 ROBERT TEISBERG/ UNITED COMPUTING SYSTEMS/ 2525 WASHINGTON/ KANSAS CITY MO 64108
65401 HOWARD D. PYRON/ MATH - C.SCI./ U OF MISSOURI - ROLLA/ ROLLA MO 65401/ (314) 341-4491
66045 CHARLES J. BANGERT/ COMPUTATION CENTER/ UNIVERSITY OF KANSAS/ P.O. DRAWER 2007/ LAWRENCE KS 66045/ (913) 864-4291
66045 STEVEN S. MUCHNICK/ DEPARTMENT OF COMPUTER SCIENCE/ U OF KANSAS/ LAWRENCE KS 66045
66502 DAVID NEAL/ 1534 COLLEGE AVE #C10/ MANHATTAN KS 66502/ (913) 539-9209/ (913) 532-6350 (WORK)
67220 RODNEY M. BATES/ 4732 N. GLENDALE/ WICHITA KS 67220/ (316) 744-2847/ (316) 687-5275
68005 KEN RITCHIE/ 508 BEAMAN DR./ BELLEVUE NE 68005/ (402) 291-7224 (HOME)/ (402) 291-5400 (WORK)
68022 JERRY L. RAY/ 21320 OLDGATE RD./ ELKHORN NE 68022/ (402) 289-3381/ (402) 291-5400
68101 LYNNE J. BALDWIN/ DEPT. OF MATH/COMP. SCI./ U OF NEBRASKA/ BOX 688/ OMAHA NE 68101/ (402) 554-2836
68123 RONALD G. MARTIN/ 12430 WALKER DRIVE/ OMAHA NE 68123/ (402) 294-3253
68588 SHARAD C. SETH/ DEPT. OF COMP. SCI./ U OF NEBRASKA/ LINCOLN NE 68588/ (402) 472-3488
70118 D. B. KILLEEN/ COMPUTER LAB/ RICHARDSON BLDG./ TULANE UNIVERSITY/ NEW ORLEANS LA 70118
70122 FRED A. HOSCH/ COMPUTER RESEARCH CENTER/ UNIV. OF NEW ORLEANS/ NEW ORLEANS LA 70122/ (504) 283-0347
70125 SAM HILLS/ 3514 LOUISIANA AVE. PKWY./ NEW ORLEANS LA 70125/ (504) 821-1737
70504 ATTN: SERIALS DEPT./ U. OF S.W. LOUISIANA LIBRARIES/ 302 E. ST. MARY BLVD./ LAFAYETTE LA 70504
70504 WARREN JOHNSON/ U OF SOUTHWESTERN LOUISIANA/ BOX 4-2770 USL STATION/ LAFAYETTE LA 70504/ (318) 234-7349
70504 ED KATZ/ COMPUTER SCIENCE DEPT./ U OF SOUTHWESTERN LOUISIANA/ BOX 4-4330 USL STATION/ LAFAYETTE LA 70504/ (318) 233-6840/ (318) 233-6767
70504 STEVE LANDRY/ COMPUTER CENTER/ U OF SOUTHWESTERN LOUISIANA/ P.O. BOX 4-2770/ LAFAYETTE LA 70504/ (318) 234-7349
70504 DAVID LANDSKOV/ U OF SOUTHWESTERN LOUISIANA/ USL BOX 4-4154/ LAFAYETTE LA 70504/ (318) 234-7640
70504 A. I. STOCKS/ P.O. BOX 4-1039/ USL STATION/ LAFAYETTE LA 70504/ (318) 233-3850 X538
70504 TERRY M. WALKER/ COMPUTER SCIENCE DEPT./ U OF SOUTHWESTERN LOUISIANA/ P.O. BOX 4-4330/ LAFAYETTE LA 70504/ (318) 234-7640
72143 MIKE CHALENBURG/ HARDING COLLEGE/ BOX 4/ SEARCY AR 72143/ (501) 268-6161 X322
72143 JOHN NUNNALLY/ HARDING COLLEGE/ BOX 744/ SEARCY AR 72143/ (501) 268-6161 X440
72204 DENNIS DANCE/ COMPUTER SCIENCE DEPT./ UNIVERSITY OF ARKANSAS AT LITTLE ROCK/ 33RD AND UNIVERSITY/ LITTLE ROCK AR 72204/ (501) 569-3252
73019 RICHARD V. ANDREE/ MATH DEPT./ U OF OKLAHOMA/ NORMAN OK 73019/ (405) 325-3410
73034 MARY DEE FOSBERG/ 600 TIMBER LANE/ EDMOND OK 73034
73034 ARDOTH H. WILSON/ COMPUTER CENTER/ CENTRAL STATE UNIVERSITY/ EDMOND OK 73034/ (405) 341-2980 X321
73070 RALPH HOWENSTINE/ P.O. BOX 1327/ NORMAN OK 73070
73106 DAVID HUSNIAN/ 1731 N.W. 29TH/ OKLAHOMA CITY OK 73106/ (213) 521-1547
73110 STEPHEN A. PITTS/ 305 EAST JARMAN DRIVE/ MIDWEST CITY OK 73110/ (405) 732-4060
74171 DAVE R. ELAND/ ORAL ROBERTS UNIVERSITY/ 7777 SOUTH LEWIS/ TULSA OK 74171/ (918) 492-6161
75023 ROGER R. BATE/ 3428 MISSION RIDGE/ PLANO TX 75023/ (214) 238-3052
75042 JOE C. ROBERTS/ 1529 MEADOWCREST/ GARLAND TX 75042
75075 GILBERT J. HANSEN/ 3104 BONNIEBROOK DRIVE/ PLANO TX 75075/ (214) 423-7837
75075 BRIAN W. JOHNSON/ 1525 WESTLAKE/ PLANO TX 75075/ (214) 690-2885
75080 ATTN: COMPUTER SERVICES - FO1.3/ U. OF TEXAS AT DALLAS/ P.O. BOX 688/ RICHARDSON TX 75080/ (214) 690-2651
75080 GEORGE LIGLER/ 1000 W. SPRING VALLEY RD. APT. 263/ RICHARDSON TX 75080/ (214) 231-0825
75081 FRANK DUNN/ 1912 E. SPRING VALLEY ROAD/ RICHARDSON TX 75081/ (214) 231-3423
75081 DAVE HABERMAN/ 1806 AUBURN DRIVE/ RICHARDSON TX 75081/ (214) 238-4446/ (214) 238-5357
75081 J. P. HARVELL/ ADV. SYSTEMS DEVELOPMENT 410-260/ ROCKWELL INTERNATIONAL/ 1200 N. ALMA ROAD/ RICHARDSON TX 75081/ (214) 783-3854
75081 DOUGLAS S. JOHNSON/ 907 EDGEWOOD DR/ RICHARDSON TX 75081/ (214) 238-4092 (TI)
75081 KENNETH L. WILLIAMS/ 614 CLEARWOOD DR./ RICHARDSON TX 75081/ (214) 341-6278
75214 FRANK A. SCHROEDER/ 6451 VANDERBILT/ DALLAS TX 75214/ (214) 824-0834
75220 DEXTER COOK/ 3040 PARK LANE APT. 106/ DALLAS TX 75220/ (214) 358-3794
75222 DONNA K. DUNAWAY/ TEXAS INSTRUMENTS INC./ P.O. BOX 5936 - MS132/ DALLAS TX 75222/ (214) 238-2635
75222 TED FISHMAN/ TEXAS INSTRUMENTS/ P.O. BOX 6015 (MS 3101)/ DALLAS TX 75222/ (214) 689-4111 X330
75222 DENNIS J. FRAILEY/ COMP. SCI. DEPT./ SOUTHERN METHODIST UNIV./ DALLAS TX 75222
75229 DAVID E. BREEDING/ HARRIS DATA COMM DIV/ 11262 INDIAN TRAIL/ DALLAS TX 75229/ (214) 620-4294
75229 JERRY SCHIEFFER/ HARRIS CORPORATION/ 11262 INDIAN TRAIL/ DALLAS TX 75229/ (214) 620-4237
75234 T. W. EKBERG/ HARRIS DATA COMMUNICATIONS/ 11262 INDIAN TRAIL/ DALLAS TX 75234/ (214) 620-4208
75234 SAM LISOOK/ HARRIS DATA COMMUNICATIONS DIV./ 11262 INDIAN TRAIL - P.O. BOX 44076/ DALLAS TX 75234/ (214) 620-4225
75240 JOHN EARLS/ SUITE 509W/ ARTHUR A. COLLINS INC./ 13601 PRESTON RD./ DALLAS TX 75240/ (214) 661-2928
75240 GERALD A. SHOULTS/ 13336 MAHAM RD. APT. 138/ DALLAS TX 75240/ (214) 238-4458 (OFFICE)/ (214) 234-2182 (HOME)

75243 W. J. MEYERS/ 4-214S THE TIMBERS/ 13447 N. CENTRAL EXPR./ DALLAS TX 75243/ (214) 231-4869
75248 JOE COINTMENT/ 7709 QUEENS GARDEN DR./ DALLAS TX 75248/ (214) 387-0468
75275 JOHN J. ALLAN III/ CENTER FOR SPECIAL STUDIES/ 118 CARUTH HALL/ SOUTHERN METHODIST UNIV./ SCHOOL OF ENGR. AND APPL. SCIENCE/ DALLAS TX 75275
(214) 692-3058
75275 GARY CEDERQUIST/ SOUTHERN METHODIST UNIV./ BOX 2112/ DALLAS TX /5275
75275 JANET TAYLOR/ USER SERVICES/ COMPUTING CENTER/ SOUTHERN METHODIST UNIVERSITY/ DALLAS TX 75275/ (214) 692-2900
75961 JESSE D. MIXON/ DEPT. OF COMPUTER SCIENCE/ STEPHEN F. AUSTIN STATE U/ P.O. BOX 6167 SFA STATION/ NACOGDOCHES TX 75961/ (713) 569-2508
76011 MICHAEL SETTLE/ 751 WASHINGTON #115/ ARLINGTON TX 76011
76019 PHILIP STEPHENSON/ COMPUTER TRAINING & DEVELOPMENT/ UNIV. OF TEXAS-ARLINGTON/ BOX 19608/ ARLINGTON TX 76019/ (817) 273-3666
76114 RANDY BEST/ 5878 CALLOWAY DR. NORTH/ FT. WORTH TX 76114/ (817) 731-4974
76201 EDWARD E FERGUSON/ 1222 AUSTIN AVE/ DENTON TX 76201/ (214) 231-9736
77001 ATTENTION: COLIN G. CAMPBELL/ MS / 781/ TEXAS INSTRUMENTS/ P.O. BOX 1444/ HOUSTON TX 77001
77001 S. BALASUBRAMANIAN/ SHELL DEVELOPMENT COMPANY/ PO BOX 481/ HOUSTON TX 77001/ (713) 667-5661
77001 GINGER KELLY/ ICSA/ RICE UNIVERSITY/ HOUSTON TX 77001/ (713) 527-4965
77001 TONEY MORELOCK/ TEXAS EASTERN TRANSMISSION/ P.O. BOX 2521/ HOUSTON TX 77001/ (713) 651-0161
77027 CHARLES L. HETHCOAT III/ C/O PIPELINE TECHNOLOGISTS INC./ P.O. BOX 22146/ HOUSTON TX 77027/ (713) 622-3456 X334 (WORK)/ (713) 626-7737 (HOME)
77030 JAMES A. KENDALL/ MHMR/TRIMS/ TEXAS MEDICAL CENTER/ HOUSTON TX 77030/ (713) 797-1976
77043 JOHN EARL CRIDER/ 2918 KEVIN LANE/ HOUSTON TX 77043/ (713) 665-3016
77098 SCOTT K. WARREN/ ROSETTA ALGORITHMS/ 2414 BRANARD #D/ HOUSTON TX 77098/ (713) 526-0849
77550 RUSSELL W ZEARS/ BIOMETRY LAB/ 449 ADMINISTRATION BLDG R7/ UNIVERSITY OF TEXAS MEDICAL BRANCH/ GALVESTON TX 77550/ (713) 765-1813
77843 RICHARD HUBER/ DEPT. OF INDUSTRIAL ENGINEERING/ TEXAS A&M UNIVERSITY/ COLL. STATION TX 77843/ (713) 845-5531 X256
78284 MIKE GREEN/ DATAPOINT CORPORATION/ 9725 DATAPOINT DRIVE/ SAN ANTONIO TX 78284/ (512) 699-7345
78705 WILLETT KEMPTON/ 2512 SAN GABRIEL ST./ AUSTIN TX 78705
78712 ATTN: DOROTHY SMITH - REFERENCE LIBRAR/ COMPUTATION CENTER/ U OF TEXAS AUSTIN/ AUSTIN TX 78712/ (512) 471-3242
78712 WILHELM BURGER/ DEPT. OF COMPUTER SCIENCES/ 328 PAINTER HALL/ UNIV. OF TEXAS - AUSTIN/ AUSTIN TX 78712/ (512) 471-1902
78712 TOM KEEL/ COMPUTATION CENTER/ UNIV. OF TEXAS - AUSTIN/ AUSTIN TX 78712
78712 WAYNE SEIPPEL/ BOX 8259 U.T. STA./ AUSTIN TX 78712/ (512) 472-1773
78712 WALLY WEDEL/ COMPUTATION CENTER/ U OF TEXAS AUSTIN/ AUSTIN TX 78712/ (512) 471-3242
78721 EDWARD P. STRITTER/ V BLDG./ MOTOROLA/ 3501 ED BLUESTEIN BLVD./ AUSTIN TX 78721/ (512) 928-2600 X501
78721 DONALD G. WEISS/ 3501 ED BLUESTEIN BLVD./ AUSTIN TX 78721/ (512) 928-2600
78723 JOEL BONEY/ 6707 LASALLE/ AUSTIN TX 78723/ (512) 928-4649
78751 DAVID W. HOGAN/ 4104 AVENUE F/ AUSTIN TX 78751
78753 TERRY RITTER/ 12002B POLLYANNA AVE./ AUSTIN TX 78753/ (512) 928-2600 X532
78758 WILLIAM L. COHAGAN/ SUITE 211/ S/B/P & C ASSOCIATES/ 8705 SHOAL CREEK BLVD./ AUSTIN TX 78758/ (512) 458-2276
78767 ATTENTION: MILES RICKARD/ MS / 2201/ TEXAS INSTRUMENTS/ P.O. BOX 2909/ AUSTIN TX 78767
78769 DAVID N. GRAY/ MS 2188/ TEXAS INSTRUMENTS/ P.O. BOX 2909/ AUSTIN TX 78769/ (512) 258-5121 X2377
79015 HARRY P. HAIDUK/ DEPT. OF COMP. INFO. SYSTEMS/ WEST TEXAS STATE U/ CANYON TX 79015/ (806) 656-3966
79409 MAURICE BALLEW/ COMPUTER SERVICES/ TEXAS TECH UNIVERSITY/ BOX 4519/ LUBBOCK TX 79409/ (806) 742-2900
79409 LEONARD H. WEINER/ DEPT. OF MATH AND COMP. SCI./ TEXAS TECH. U/ P.O. BOX 4319/ LUBBOCK TX 79409/ (806) 742-2571
79601 D. A. CAUGHFIELD/ 609 E. N. 21ST/ ABILENE TX 79601/ (915) 672-1604
79601 JOHN TUCKER/ 628 E.N. 16TH ST./ ABILENE TX 79601/ (915) 673-2840
80201 GREGG E. MARSHALL/ P.O. BOX 2784/ DENVER CO 80201/ (303) 499-1000 X4482
80201 NORMAN T. OLSEN/ C/O AUTO TROL CORP./ 5650 N. PECOS/ DENVER CO 80201/ (303) 458-5900
80215 DAVID M. WARNER/ 755 VISTA LANE/ LAKEWOOD CO 80215/ (303) 238-0900
80225 ATTN: CHIEF BRANCH OF DATA SYSTEM SERV/ HSAC-POB 25367/ MINE ENFORCEMENT AND SAFETY ADM./ DENVER FEDERAL CENTER/ DENVER CO 80225/ (303) 234-3025
80225 ATTN: LIBRARY/ 67 DENVER FEDERAL CENTER/ BUREAU OF RECLAMATION/ DENVER CO 80225
80302 ATTN: KARIN & MICHELE - PASCAL DISTRIB/ COMPUTING CENTER LIBRARY/ UNIVERSITY OF COLORADO/ 3645 MARINE STREET/ BOULDER CO 80302/ (303) 492-8131
80302 HOWARD BUSSEY JR./ NATIONAL OCEANIC AND ATMOSPHERIC ADMIN/ BLDG. 1 RM 4557/ U.S. DEPARTMENT OF COMMERCE/ BOULDER CO 80302
80302 RAYNER K. ROSICH/ OT/IITS/ U.S. DEPT. OF COMMERCE/ 325 BROADWAY/ BOULDER CO 80302/ (303) 499-1000 X3109
80302 JOE WATKINS/ 2895 18TH STREET/ BOULDER CO 80302/ (303) 443-8598
80303 DENNIS R. ELLIS/ C/O CRAY RESEARCH/ 75 MANHATTAN DR. - SUITE #3/ BOULDER CO 80303/ (303) 499-3055
80303 VINCENT B. WAYLAND/ C/O CRAY RESEARCH INC./ 75 MANHATTAN DRIVE SUITE 3/ BOULDER CO 80303/ (303) 499-3055
80307 BRUCE K. RAY/ POLYMORPHIC COMPUTER SYSTEMS/ P.O. BOX 3581/ BOULDER CO 80307/ (303) 443-5362
80309 LLOYD D. FOSDICK/ DEPARTMENT OF COMPUTER SCIENCE/ ECOT 7-7/ U OF COLORADO/ BOULDER CO 8309/ (303) 492-7514
80309 GEORGE H. RICHMOND/ COMPUTING CENTER/ UNIVERSITY OF COLORADO/ 3645 MARINE STREET/ BOULDER CO 80309/ (303) 492-8131
80309 TERRY L. SPEAR/ CLIPR/ E318 MUENZINGER/ UNIV. OF COLORADO/ BOULDER CO 80309/ (303) 492-6991
80309 WILLIAM M. WAITE/ ELECTRICAL ENGINEERING DEPT./ SOFTWARE ENGINEERING GROUP/ UNIVERSITY OF COLORADO/ BOULDER CO 80309
80401 HERBERT RUBENSTEIN/ 401 GARDEN STREET/ GOLDEN CO 8041/ (303) 278-3469
80523 ATTN: USER SERVICES GROUP/ UNIVERSITY COMPUTER CENTER/ COLORADO STATE U/ FORT COLLINS CO 80523/ (303) 491-5133
80523 DALE H. GRIT/ DEPARTMENT OF COMPUTER SCIENCE/ COLORADO STATE U/ FT. COLLINS CO 80523/ (303) 491-7033
80537 JEFF EASTMAN/ CALCULATOR PRODUCTS DIV./ HEWLETT PACKARD/ P.O. BOX 301/ LOVELAND CO 80537
82071 HENRY R. BAUER III/ COMPUTER SCIENCE DEPT./ UNIVERSITY OF WYOMING/ BOX 3682/ LARAMIE WY 82071/ (307) 766-5134
83401 KYU Y. LEE/ E.C.& G. IDAHO INC./ P.O. BOX 1625/ IDAHO FALLS ID 83401/ (208) 526-0111 X321
83843 JOHN DICKINSON/ DEPT. OF ELECTRICAL ENGR./ 214 BEL/ UNIV. OF IDAHO/ MOSCOW ID 83843/ (208) 885-6554/6555
84112 ATTN: B1700 PROTEUS PROJECT/ COMPUTER SCIENCE DEPT./ 3160 MEB/ U OF UTAH/ SALT LAKE CIT UT 84112/ (801) 581-8224
84112 MARTIN L GRISS/ COMPUTER SCIENCE DEPT/ U OF UTAH/ SALT LAKE CIT UT 84112/ (801) 581-6542

84112 M. A. KLEINERT/ COMP. SCI. DEPT./ 3160 MERRILL ENG. BLDG./ U OF UTAH/ SALT LAKE CIT UT 84112
 84112 GARY LINDSTROM/ COMPUTER SCIENCE DEPT./ U OF UTAH/ SALT LAKE CIT UT 84112/ (801) 581-8224
 84112 ED SHARP/ COMPUTER CENTER/ U OF UTAH/ SALT LAKE CIT UT 84112/ (801) 581-6802
 84601 DENNIS FAIRCLOUGH/ EYRING RESEARCH INSTITUTE/ 1455 WEST 820 NORTH/ PROVO UT 84601/ (801) 375-2434
 84601 PAUL GODFREY/ 41 SOUTH 500 WEST/ PROVO UT 84601/ (801) 377-4331
 84602 THEODORE A. NORMAN/ COMP. SCI. DEPT./ BRIGHAM YOUNG UNIVERSITY/ PROVO UT 84602/ (801) 374-1211 X3027
 84602 RICHARD OHRAN/ ELECTRICAL ENGINEERING DEPT/ 459 ESTB/ BRIGHAM YOUNG UNIVERSITY/ PROVO UT 84602/ (801) 374-1211 X4012
 85061 E. W. ERRICKSON/ P.O. BOX 11472/ PHOENIX AZ 85061/ (602) 242-3420
 85260 DENNIS KODIMER/ SUITE 100/ TERAK CORPORATION/ 14425 N. SCOTTSDALE RD./ SCOTTSDALE AZ 85260/ (602) 991-1580
 85281 BRIAN D. LOCKREY/ COMPUTER SERVICES ECA-109/ ARIZONA STATE UNIVERSITY/ TEMPE AZ 85281/ (602) 965-7327
 85721 PATRICK PECORARO/ UNIVERSITY COMPUTER CENTER/ U OF ARIZONA/ TUCSON AZ 85721/ (602) 884-2901
 85726 R. W. MILKEY/ KITT PEAK NATIONAL OBSERVATORY/ P.O. BOX 26732/ TUCSON AZ 85726/ (602) 327-5511
 85726 W. RICHARD STEVENS/ KITT PEAK NATIONAL OBSERVATORY/ P.O. BOX 26732/ TUCSON AZ 85726/ (602) 327-5511
 85731 JOHN E. WAHL/ P.O. BOX 18078/ TUCSON AZ 85731/ (602) 747-0700 X307
 86301 NEAL H. CHAMPION/ 435 S. GRANITE/ PRESCOTT AZ 86301
 87002 TOM SANDERSON/ RFD 1 BOX 459/ BELEN NM 87002
 87106 BOB WALSH/ 817 LAFAYETTE DR. NE/ ALBUQUERQUE NM 87106/ (505) 268-1654
 87109 DON H. ROWLAND/ 5805 TORREON DR./ ALBUQUERQUE NM 87109/ (505) 821-9207 (HOME)/ (505) 264-9149 (OFFICE)
 87114 ATTENTION: ARMENELLA VINSON/ E.G. & G. INC./ PO BOX 10218 - ALAMEDA STA./ ALBUQUERQUE NM 87114/ (505) 898-8000 EXT 246
 87115 ALFRED J. HULBERT/ INHALATION TOXICOLOGY RESEARCH INST./ P.O. BOX 5890/ ALBUQUERQUE NM 87115/ (505) 264-2030
 87115 BRUCE LINK/ DIVISION 1712/ SANDIA LABORATORIES/ ALBUQUERQUE NM 87115
 87115 NANCY RUIZ/ ORG. 5166/ SANDIA LABS/ ALBUQUERQUE NM 87115/ (505) 264-3690
 87117 ATTN: AIR FORCE WEAPONS LABORATORY/ DYM (HARRY M. MURPHY JR.)/ KIRTLAND AFB NM 87117/ (505) 264-9317
 87544 KAY A. HANSBOROUGH/ 2377B 45TH ST./ LOS ALAMOS NM 87544/ (505) 662-9369 (HOME)/ (505) 667-5275 (OFFICE)
 87545 BILL BUZBEE/ LOS ALAMOS SCIENTIFIC LABORATORY/ C-DO MS-260/ UNIVERSITY OF CALIFORNIA/ P.O. BOX 1663/ LOS ALAMOS NM 87545
 87545 ROBERT T. JOHNSON/ C-11 MAIL STOP 296/ LOS ALAMOS SCIENTIFIC LABORATORY/ P.O. BOX 1663/ LOS ALAMOS NM 87545/ (505) 667-5014
 87545 JOHN MONTAGUE/ GROUP C11/ MAIL STOP 296/ LOS ALAMOS SCIENTIFIC LABORATORY/ LOS ALAMOS NM 87545
 87801 JAMES DARLING/ NEW MEXICO TECH/ BOX 2139 CAMPUS STATION/ SOCORRO NM 87801/ (505) 835-5455
 87801 T. A. NARTKER/ NEW MEXICO INSTITUTE OF MINING AND TEC/ SOCORRO NM 87801/ (505) 835-5126
 87801 KIM L. SHIVELEY/ NEW MEXICO TECH./ P.O. BOX 2129 C.S./ SOCORRO NM 87801/ (505) 835-5766
 88003 J. MACK ADAMS/ COMP. SCI. DEPT./ NEW MEXICO STATE U/ BOX 3CU/ LAS CRUCES NM 88003/ (505) 646-3723
 88003 ATTN: USER SERVICES LIBRARIAN/ UNIVERSITY COMPUTER CENTER/ NEW MEXICO STATE UNIVERSITY/ BOX 3AT/ LAS CRUCES NM 88003/ (505) 644-4433
 89154 ATTN: RESEARCH PROGRAMMING ADVISOR/ COMPUTING CENTER/ U. OF NEVADA - LAS VEGAS/ 4505 MARYLAND PARKWAY/ LAS VEGAS NV 89154/ (702) 739-3557
 89154 JOHN WERTH/ DEPT. OF MATH/ U OF NEVADA LAS VEGAS/ LAS VEGAS NV 89154/ (702) 739-3715
 89507 ATTENTION: ROY MAXION-PROGRAMMING ADVI/ UNS COMPUTING CENTER/ 22 WR/ U OF NEVADA/ BOX 9068/ RENO NV 89507/ (702) 784-4008
 89507 GARY CARTER/ SEISMOLOGY DEPT./ MACKAY SCHOOL OF MINES/ U OF NEVADA RENO/ RENO NV 89507
 89509 WILLIAM R. BONHAM/ SIERRA DIGITAL SYSTEMS/ 1440 WESTFIELD AVE./ RENO NV 89509/ (702) 329-9548
 90007 ATTN: ACADEMIC SERVICES/ UNIVERSITY COMPUTER CENTER/ U OF SOUTHERN CALIFORNIA/ 1020 W. JEFFERSON BLVD./ LOS ANGELES CA 90007/ (213) 746-2957
 90007 JORGEN STAUNSTRUP/ COMPUTER SCIENCE DEPT./ UNIV. OF SOUTHERN CALIFORNIA/ UNIVERSITY PARK/ LOS ANGELES CA 90007/ (213) 748-1977
 90009 FREDERICK C. COWAN/ MAIL STATION A2-2043/ THE AEROSPACE CORP./ P.O. BOX 92957/ LOS ANGELES CA 90009/ (213) 648-6482
 90020 DENNIS YOUNG/ 3311 WEST 3RD ST. APT. 1-319/ LOS ANGELES CA 90020/ (213) 383-9666
 90024 ERIC PUGH/ 632 LEVERING AVE. APT. D/ LOS ANGELES CA 90024/ (213) 479-1352
 90024 KARL H. RYDEN/ HEALTH SCIENCES COMPUTING FACILITY/ 23 DEPT OF BIOMATH/ UCLA/ LOS ANGELES CA 90024/ (213) 825-5200
 90024 BRUCE SEILER/ DEPT. OF CHEMISTRY/ UCLA/ 405 HILGARD AVENUE/ LOS ANGELES CA 90024/ (213) 825-3818
 90036 WILLIAM MOSKOWITZ/ INSTRUCTIONAL SUPPORT GROUP/ CALIFORNIA STATE UNIVERSITY/ 5670 WILSHIRE BOULEVARD/ LOS ANGELES CA 90036/ (213) 852-5780
 90048 STEVEN BARRYTE/ 6620 W. 5TH STREET/ LOS ANGELES CA 90048/ (213) 653-8697
 90064 DAVID G. CLEMANS/ 2830 SEPULVEDA APT.20/ LOS ANGELES CA 90064/ (213) 473-7961
 90066 ERWIN BOOK/ 3169 COLBY AVENUE/ LOS ANGELES CA 90066
 90068 HOWARD H. METCALF/ 2590 GLEN GREEN #4/ HOLLYWOOD CA 90068
 90230 ARTHUR I. SCHWARZ/ BLDG. 150/MS A222/ HUGHES AIRCRAFT CO./ CULVER CITY CA 90230
 90260 ATTN: LAL CHAN DANI ENTERPRISES/ COMPUTER LAND/ 16919A HAWTHORNE BLVD./ LAWDALE CA 90260
 90274 JIM HIGHTOWER/ 4947 BROWNDER LANE/ RANCHO PALOS CA 90274/ (213) 541-4662
 90274 MARK L. ROBERTS/ RYAN MCFARLAND CORPORATION/ 608 SILVER SPUR ROAD/ ROLL.H.ESTATE CA 90274/ (213) 377-0491
 90278 JOHN R. DEALY/ BLDG. R3/1072/ TRW DSSG/ ONE SPACE PARK/ REDONDO BEACH CA 90278/ (213) 535-0833
 90278 WILEY GREINER/ 90/2178/ TRW DSSG/ ONE SPACE PARK/ REDONDO BEACH CA 90278/ (213) 535-0313
 90278 J. B. HELDEBRECHT/ 2178 BLD. 90/ TRW DSSG/ ONE SPACE PARK/ REDONDO BEACH CA 90278/ (213) 535-0313
 90278 DENNIS HEIMBIGNER/ 2500 CARNEGIE LANE #B/ REDONDO BEACH CA 90278/ (213) 535-0833
 90291 RALPH L. LONDON/ INFORMATION SCIENCES INSTITUTE/ U OF SOUTHERN CALIFORNIA/ 4676 ADMIRALTY WAY/ MARINA DEL RE CA 90291/ (213) 822-1511 X195
 90403 MICHAEL TEENER/ TECHNOLOGY SERVICE CORP./ 2811 WILSHIRE BLVD./ SANTA MONICA CA 90403/ (213) 829-7411 X244
 90501 WILLIAM E. FISHER/ 2074 SANTA FE AVENUE/ TORRANCE CA 90501
 90503 JOHN R. BARR/ 22014 REYNOLDS DRIVE/ TORRANCE CA 90503/ (213) 648-8295/ (213) 540-1381
 90746 PHYLLIS A. REILLY/ 19711 GALWAY AVENUE/ CARSON CA 90746/ (213) 321-5215
 91016 CLARK M. ROBERTS/ 219 VIOLET AVENUE/ MONROVIA CA 91016/ (213) 456-3858 (HOME)/ (213) 658-2405 (WORK)
 91101 E. E. SIMMONS/ 455 SOUTH OAKLAND AVE/ PASADENA CA 91101
 91103 CHARLES L. LAWSON/ JET PROPULSION LABORATORY/ MS 125/128/ CALIFORNIA INSTITUTE OF TECHNOLOGY/ 4800 OAK GROVE DR./ PASADENA CA 91103/ (213) 354-4321
 91107 ROBERT M. LANSFORD/ 3620 GREENHILL ROAD/ PASADENA CA 91107/ (213) 351-0206

91109 ATTN: LIBRARY/ BURROUGHS CORP./ 460 SIERRA MADRE VILLA/ PASADENA CA 91109/ (213) 351-6551 X505
 91330 KEN MODESITT/ COMPUTER SCIENCE DEPT./ CALIFORNIA STATE UNIV./ 18111 NORDHOFF ST./ NORTHRIDGE CA 91330
 91335 MARK T. MARSHALL/ 18229 TOPHAM ST./ RESEDA CA 91335/ (213) 345-1739
 91702 ED KEITH/ CITRUS COLLEGE/ 18824 E. FOOTHILL BLVD./ AZUSA CA 91702/ (213) 335-0521 X313/ (213) 963-1052
 91711 GERALD BRYAN/ SEAVER COMPUTER CENTER/ CLAREMONT COLLEGES/ CLAREMONT CA 91711/ (714) 626-8511 X3228
 91711 CHRIS P. LINDSEY/ COMPUTING/ HARVEY MUDD COLLEGE/ CLAREMONT CA 91711/ (714) 626-8511 X2897
 91711 STANLEY E. LUNDE/ 890 HOOD DRIVE/ CLAREMONT CA 91711/ (714) 626-9977
 91740 DAVID C. FITZGERALD/ 652 S. CULLEN/ GLENDORA CA 91740/ (213) 335-6055
 91775 TOM GREER/ 224 N. ALABAMA ST./ SAN GABRIEL CA 91775
 92025 MARK J. KAUFMAN/ 916 E WASHINGTON APT. 108/ ESCONDIDO CA 92025/ (714) 743-5911
 92026 K. DOUGLAS JOHNSTON/ 1375 N BROADWAY APT F-2/ ESCONDIDO CA 92026/ (714) 743-5830/ (714) 485-2309 (WORK)
 92067 LANCE A. LEVENTHAL/ P.O. BOX 1258/ RANCHO SANTAF CA 92067/ (714) 755-6541
 92093 KEN BOWLES/ APIS DEPT./ C-21/ U OF CALIFORNIA - SAN DIEGO/ LA JOLLA CA 92093/ (714) 755-7288/ 452-4526
 92093 JIM MADDEN/ C-010 COMPUTER CENTER/ UNIV. OF CALIFORNIA - SAN DIEGO/ LA JOLLA CA 92093/ (714) 452-4067
 92093 MARK OVERGAARD/ APIS DEPT./ C-014/ U OF CALIFORNIA - SAN DIEGO/ LA JOLLA CA 92093/ (714) 452-4723
 92103 DAVID M. BULMAN/ PRAGMATICS INC./ BOX 33228/ SAN DIEGO CA 92103/ (714) 565-0565
 92111 WARREN EDWARD LOPER/ 6542 ALCALA KNOLLS DR./ SAN DIEGO CA 92101/ (714) 560-0718 (HOME)/ (714) 225-2480 (WORK)
 92121 LOUIS A. BENTON/ STAFF COMPUTER TECHNOLOGY CORP./ 10457 J ROSELLE ST./ SAN DIEGO CA 92121/ (714) 453-0303
 92121 CRAIG MAUDLIN/ SUITE M/ RENAISSANCE SYSTEMS/ 11760 SORRENTO VALLEY RD./ SAN DIEGO CA 92121/ (714) 452-0681
 92122 GORDON J. WOOD/ 5818 MOTT ST./ SAN DIEGO CA 92122/ (714) 453-8167
 92152 MICHAEL S. BALL/ CODE 632/ NAVAL OCEAN SYSTEMS CENTER/ SAN DIEGO CA 92152
 92152 KENNETH O. LELAND/ R & D CENTER/ NAVY PERSONNEL/ CODE 9303/ SAN DIEGO CA 92152/ (714) 225-7388/ 933-7388 (DEF. DEPT. AV)
 92324 DAVID H. WELCH/ P.O. BOX 721/ COLTON CA 92324
 92408 TED C. PARK/ SYSTEMS DEVELOPMENT/ SUITE 302/ MEDICAL DATA CONSULTANTS/ 1894 COMMERCENTER WEST/ SAN BERNARDIN CA 92408/ (714) 825-2683
 92507 ATTN: COMPUTER SCIENCES INSTITUTE/ U OF CALIFORNIA/ RIVERSIDE CA 92507
 92507 KURT COCKRUM/ 3398 UTAH/ RIVERSIDE CA 92507/ (714) 682-1907
 92626 ATTENTION: A.S. WILLIAMS/ LIBRARY/ TECHNOLOGY MARKETING INC./ 3170 RED HILL AVE./ COSTA MESA CA 92626/ (714) 979-1100
 92634 SEYMOUR SINGER/ BLDG 606/M.S. K110/ HUGHES AIRCRAFT CO./ P.O. BOX 3310/ FULLERTON CA 92634
 92653 ED HIRAHARA/ 25062 GRISSOM RD./ LAGUNA HILLS CA 92653/ (714) 8/1-3232 X3073/ OR X3989
 92663 L. M. FOSTER/ COLLINS GOVT. TELECOMM. DIV. TECH. INF/ ROCKWELL INTERNATIONAL CORP./ 4311 JAMBOREE ROAD (501-105)/ NEWPORT BEACH CA 92663/ (714) 388-4389
 92675 ROBERT L. JARDINE/ BURROUGHS CORP./ 25725 JERONIMO ROAD/ MISSION VIEJO CA 92675/ (714) 768-2370
 92701 ROBERT L. HARTMAN/ 1425 E. FRANZEN AVE./ SANTA ANA CA 92701/ (714) 646-7466
 92704 COLE A. CHEVALIER/ CONTROL DATA CORPORATION/ 3519 W. WARNER/ SANTA ANA CA 92704/ (714) 754-4134
 92704 CHARLES J. FETE/ W-14/ C/O CONTROL DATA CORP./ 3519 W. WARNER AVE./ SANTA ANA CA 92704/ (714) 754-4155
 92704 JIM FONTANA/ CONTROL DATA CORPORATION/ 3519 W. WARNER AVE./ SANTA ANA CA 92704/ (714) 754-4102
 92704 S. J. PACKER/ CONTROL DATA CORPORATION/ 3519 W. WARNER AVE./ SANTA ANA CA 92704/ (714) 754-4129
 92705 WALTER KOSINSKI/ INFORMATION SCIENCES CONSULTING/ 1654 SE SKYLINE DRIVE/ SANTA ANA CA 92705/ (714) 838-9387
 92713 GREGORY L. HOPWOOD/ MINICOMPUTER OPERATIONS/ SPERRY UNIVAC/ 2722 MICHELSON DRIVE/ IRVINE CA 92713/ (714) 833-2400
 92713 BOB HUTCHINS/ COMPUTER AUTOMATION INC./ 18651 VON KARMAN/ IRVINE CA 92713/ (714) 833-8830 X335
 92713 ERIC OLSEN/ VARIAN DATA MACHINES/ 2722 MICHELSON DRIVE/ IRVINE CA 92713/ (714) 833-2400
 92714 WILLIAM E. CROSBY/ 15381 ORLEANS CIR./ IRVINE CA 92714/ (714) 551-5632
 92714 RUDY L. FOLDEN/ 14681 COMET ST./ IRVINE CA 92714/ (714) 552-0398
 92714 STEVE LUNDQUIST/ 5142 CHATEAU CIRCLE/ IRVINE CA 92714/ (714) 871-3232 X4352
 92714 DONALD D. PECKHAM/ PERTEC COMPUTER CORP./ 17112 ARMSTRONG AVE./ SANTA ANA CA 92714/ (714) 540-8340
 92715 WILLIAM J. EARL/ 6 LEMON TREE/ IRVINE CA 92715/ (714) 552-1543
 92717 JOHN M. GRAM/ COMPUTING FACILITY/ U OF CALIFORNIA/ IRVINE CA 92717/ (714) 833-6844
 92717 JON F. HUERAS/ DEPT. OF INFORMATION AND COMP. SCI./ U OF CALIFORNIA IRVINE/ IRVINE CA 92717/ (714) 833-2400
 92805 WILLIAM L. COOPER/ ORG 4400/ INTERSTATE ELECTRONICS/ 707 E. VERMONT/ ANAHEIM CA 92805/ (714) 772-2811 X1848
 92807 DAVID W. GIEDT/ 5421 WILLOWICK CIR./ ANAHEIM CA 92807/ (714) 712-2811
 92807 D. MARCUS/ GTE INFORMATION SYSTEMS/ 5300 E. LA PALMA/ ANAHEIM CA 92807/ (714) 524-4461
 93101 JIM MCCORD/ SYSTEMETRICS INC./ 120 E. DE LA GUERRA STREET/ SANTA BARBARA CA 93101/ (805) 963-8941
 93105 ATTENTION: NANCY BROOKS/ SCIENCE AND TECHNOLOGY DIVISION/ GENERAL RESEARCH CORPORATION/ 5383 HOLLISTER AVE./ SANTA BARBARA CA 93105/ (805) 964-7724
 93109 ROBERT ALAN DOLAN/ SPEECH COMMUNICATIONS RESEARCH LAB/ 800A MIRAMONTE DRIVE/ SANTA BARBARA CA 93109/ (805) 965-3011
 93401 NEIL W. WEBRE/ DEPT. OF COMP. SCI. AND STAT./ CALIF. POLY. STATE UNIV./ SAN LUIS OBIS CA 93401/ (805) 546-2986
 93407 JAMES L. BEUG/ DEPT. OF COMP. SCI./ CALIFORNIA POLYTECHNIC STATE U/ SAN LUIS OBIS CA 93407/ (805) 546-1255
 93407 DANA A. FREIBURGER/ COMPUTER CENTE/ CALIFORNIA POLYTECHNIC ST. UNIV./ SANLUIS OBISP CA 93407/ (805) 546-2005
 93453 H. MARC LEWIS/ PO BOX 505/ SANTA MARGARI CA 93453/ (805) 546-2009
 93555 GARY BABCOCK/ 110-E RICHMOND ROAD/ CHINA LAKE CA 93555/ (714) 939-3661
 93940 ATTN: COMPUTER SCIENCE DEPT. A/ CODE 52/ NAVAL POSTGRADUATE SCHOOL/ MONTEREY CA 93940
 93940 ATTN: COMPUTER SCIENCE DEPT. B/ CODE 52/ NAVAL POSTGRADUATE SCHOOL/ MONTEREY CA 93940
 93940 GORDON BRADLEY/ COMPUTER SCIENCE DEPT./ NAVAL POSTGRADUATE SCHOOL/ MONTEREY CA 93940
 93940 SUSAN FEUERMAN/ W.R. CHURCH COMPUTER CENTER/ CODE 0141/ NAVAL POSTGRADUATE SCHOOL/ MONTEREY CA 93940
 94022 HORACE ENEA/ HEURISTICS INC./ 900 N. SAN ANTONIO ROAD/ LOS ALTOS CA 94022/ (415) 948-2542
 94022 DAVID ELLIOT SHAW/ STRUCTURED SYSTEMS CORPORATION/ 343 SECOND STREET - SUITE K/ LOS ALTOS CA 94022/ (415) 321-8111
 94025 DENNIS R. ALLISON/ 169 SPRUCE AVENUE/ MENLO PARK CA 94025/ (415) 325-2962
 94025 GENE AUTREY-HUNLEY/ 318-8/ SRI INTERNATIONAL/ 333 RAVENSWOOD AVE./ MENLO PARK CA 94025/ (415) 326-6200 X2629

94025 APRIL MILLER CONVERSE/ SEISMIC ENGINEERING BRANCH/ M/S 87/ U.S.G.S./ 345 MIDDLEFIELD ROAD/ MENLO PARK CA 94025
 94025 JEFFREY G. SHAW/ P.O. BOX 2678/ MENLO PARK CA 94025
 94035 ZAY CURTIS/ P.O. BOX 235/ MOFFETT FIELD CA 94035/ (415) 964-9900
 94035 CARL S. ROSENBERG/ AMES RESEARCH CENTER/ MAIL STOP 239-19/ MOFFETT FIELD CA 94035/ (415) 965-6436 (WORK)/ (415) 967-7000 (HOME)
 94040 J. R. BAICHTAL/ PCM SWITCHING ENGINEERING/ TRW/VIDAR/ 77 ORTEGA AVENUE/ MOUNTAIN VIEW CA 94040/ (415) 961-1000
 94041 WARREN VAN CAMP/ 178 CENTRE #14/ MT. VIEW CA 94041/ (415) 967-3170
 94086 RICH ALTMAYER/ 655 S. FAIROAKS AVE. APT. G101/ SUNNYVALE CA 94086
 94086 DENNIS S. ANDREWS/ COMPUTING SERVICES/ AMDAHL CORP./ 1250 E. ARQUES AVE/ SUNNYVALE CA 94086/ (408) 735-4011
 94086 GLENN T. EDENS/ DACONICS DIV./ XEROX/ 350 POTRERO AVENUE/ SUNNYVALE CA 94086/ (408) 738-4800 (DACONICS)/ (415) 494-4464 (XEROX/PARC)
 94086 DENNIS ERNST/ INSTITUTE FOR ADVANCED COMPUTATION/ 1095 E. DUANE/ SUNNYVALE CA 94086/ (408) 735-0635
 94086 DENNIS GRAHAM/ AMDAHL CORP./ 1250 E. ARQUES AVE./ SUNNYVALE CA 94086/ (408) 735-4602
 94086 ROBERT S. LENT/ AMDAHL CORPORATION/ P.O. BOX 5070/ SUNNYVALE CA 94086/ (408) 735-4205
 94086 GEORGE LEWIS/ R & D/ BASIC TIMESHARING INC./ 870 WEST MAUDE AVENUE/ SUNNYVALE CA 94086/ (408) 733-1122
 94086 M. H. MACDOUGALL/ AMDAHL CORP./ P.O. BOX 5070/ SUNNYVALE CA 94086/ (408) 735-4654
 94086 FLEMING M. OLIVER/ 213 WEDDELL APT. 12/ SUNNYVALE CA 94086
 94086 ARTHUR C. WILLIS/ AMDAHL CORP./ 1250 EAST ARQUES AVE./ SUNNYVALE CA 94086/ (408) 735-4011
 94086 ANDREW HARRIS ZIMMERMAN/ 550 NORTH FAIR OAKS AVE. APT. 14/ SUNNYVALE CA 94086
 94087 ADRIAN BYRAM/ 1131 S. SAGE COURT/ SUNNYVALE CA 94087
 94088 RICHARD CORE/ PO BOX 61628/ SUNNYVALE CA 94088/ (408) 735-8400 X233
 94088 T. D. TELFORD/ DEPT. 19-63/ BLDG 529/ LOCKHEED/ P.O. BOX 504/ SUNNYVALE CA 94088/ (408) 742-7301
 94088 GARY W. WINIGER/ P.O. BOX 60835/ SUNNYVALE CA 94088/ (415) 964-6982/ (408) 742-5699 (WORK)
 94111 JIM ELAM/ 150 LOMBARD #601/ SAN FRANCISCO CA 94111
 94114 RICHARD H. KARPINSKI/ 3071 MARKET STREET/ SAN FRANCISCO CA 94114/ (415) 666-4529
 94143 FRANCIS KRICKORIAN/ ADMIN. INFO. SYSTEMS/ 101 BUILDING MR 4/ U.C.S.F. MEDICAL CENTER/ SAN FRANCISCO CA 94143/ (415) 666-3012
 94304 LINDA E. CROLEY/ BNR INC./ 3174 PORTER DR./ PALO ALTO CA 94304/ (415) 494-3942 X40 OR 61
 94304 SAM GEBALA/ HEWLETT PACKARD/ 3500 DEER CREEK RD./ PALO ALTO CA 94304/ (415) 494-1444 X214
 94304 H. S. MAGNUSKI/ GAMMA TECHNOLOGY/ 800 WELSH ROAD/ PALO ALTO CA 94304/ (415) 326-1661
 94304 PAUL MCJONES/ SDD/SD/ XEROX CORPORATION/ 3333 COYOTE HILL ROAD/ PALO ALTO CA 94304/ (415) 494-4522
 94305 PAUL HECKEL/ INTERACTIVE SYSTEMS CONSULTANTS/ P.O. BOX 2345/ PALO ALTO CA 94305/ (415) 965-0327
 94305 ATTN: LIBRARY / SERIALS/ BIN 82/ STANFORD LINEAR ACCELERATOR CENTER/ P.O. BOX 4349/ STANFORD CA 94305
 94305 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)
 94305 DAVID C. LUCKHAM/ COMP. SCI. DEPT./ A.I. LABORATORY/ STANFORD UNIVERSITY/ STANFORD CA 94305/ (415) 497-4971
 94305 HUGH MCLARTY/ BOX 10291/ STANFORD CA 94305/ (415) 322-4822
 94538 BRIAN MCGUIRE/ P.O. BOX 1371/ FREMONT CA 94538
 94545 WILLIAM F. RAGSDALE/ DORADO SYSTEMS/ 20956 CORSAIR BLVD./ HYW ARD CA 94545/ (415) 783-0289
 94550 JOHN C. BEATTY/ L-73/ LAWRENCE LIVERMORE LAB/ BOX 808/ LIVERMORE CA 94550/ (415) 447-1100 X3114
 94550 S. T. HEIDELBERG/ DIVISION 8323/ SANDIA LABORATORIES/ LIVERMORE CA 94550/ (415) 455-2179
 94550 WILLIAM P. TAYLOR/ L-315/ UNIVERSITY OF CALIFORNIA/ P.O. BOX 808/ LIVERMORE CA 94550/ (415) 455-6729
 94566 J. E. POLLACK/ 435 ABBIE STREET/ PLEASANTON CA 94566
 94596 RALPH W. SWEARINGEN/ VIRTUAL SYSTEMS INC./ 1500 NEWELL AVE. #406/ WALNUT CREEK CA 94596/ (415) 254-1147
 94611 PAUL S. GERKEN/ PROGRAMMING METHODS/ INFORMATICS/ 120 RONADA AVE./ PIEDMONT CA 94611/ (415) 655-4499
 94613 RITA MAY LIFF/ DEPT. OF MATH AND COMPUTER SCIENCE/ MILLS COLLEGE/ OAKLAND CA 94613/ (415) 632-2700 X308
 94621 BRYAN L. HIGGINS/ SCIENCE APPLICATIONS INC./ 8201 CAPWELL DRIVE/ OAKLAND CA 94621/ (415) 562-9163
 94703 JAMES A. WOODS/ 2014A WOOLSEY ST./ BERKELEY CA 94703
 94704 JIM MERRITT/ P.O. BOX 4655/ BERKELEY CA 94704/ (415) 845-4866
 94720 JEFFREY BARTH/ COMP. SCI. DIVISION/ 573 EVANS HALL/ U OF CALIFORNIA/ BERKELEY CA 94720/ (415) 642-4948
 94720 BLAND EWING/ DEPT. OF ENTOMOLOGY/ 137 GIANNINI HALL/ U OF CALIFORNIA/ BERKELEY CA 94720/ (415) 642-6660
 94720 ED FOURT/ C/O LBL LIBRARY/ 134 BLDG 50/ LAWRENCE BERKELEY LAB/ BERKELEY CA 94720/ (415) 843-2740 X5293
 94720 SUSAN L. GRAHAM/ COMP. SCI. DIVISION-EECS/ 511 EVANS HALL/ U OF CALIFORNIA/ BERKELEY CA 94720
 94720 LAWRENCE A. ROWE/ DEPT. OF EE AND CS - TEOI/ EVANS HALL/ U OF CALIFORNIA/ BERKELEY CA 94720
 94903 CHRIS K. PHILLIPS/ P.O. BOX 6283/ TERRA LINDA CA 94903/ (415) 494-7900 X357
 95003 ROBERT C. NICKERSON/ 517 SANTA MARGUARITA/ APTOS CA 95003/ (408) 688-9735
 95008 THOMAS A. ROLANDER/ 1012 SMITH AVE./ CAMPBELL CA 95008/ (408) 378-5785
 95014 A. G. CARRICK/ MS970/ FOUR-PHASE SYSTEMS INC./ 10700 N. DEANZA BLVD./ CUPERTINO CA 95014/ (408) 255-0900 X281
 95014 FAY CHONG/ 10405 DEMPSTER AVENUE/ CUPERTINO CA 95014/ (408) 987-1655
 95014 R. GREINER/ MS970/ FOUR-PHASE SYSTEMS INC./ 19333 VALLCO PARKWAY/ CUPERTINO CA 95014/ (408) 255-0900 X231
 95014 DONALD E. GRIMES/ TYMSHARE INC./ 20705 VALLEY GREEN DRIVE/ CUPERTINO CA 95014/ (408) 446-6586
 95014 P. LIAO/ MS970/ FOUR-PHASE SYSTEMS INC./ 19333 VALLCO PARKWAY/ CUPERTINO CA 95014/ (408) 255-0900 X302
 95014 JOHN P. STALLINGS/ TECHNICAL DIVISION/ TYMSHARE/ 20705 VALLEY GREEN DRIVE/ CUPERTINO CA 95014/ (408) 446-6000
 95050 JOHN DENNIS COUCH/ GSD/ HEWLETT-PACKARD/ 5303 STEVENS CREEK BLVD./ SANTA CLARA CA 95050/ (408) 249-7020 EXT.2949
 95050 LARRY WALSH/ ROLM CORPORATION/ 4900 OLD IRONSIDES DRIVE/ SANTA CLARA CA 95050/ (408) 988-2900
 95051 JOHN W. BURNETT/ M/S 690/ NATIONAL SEMICONDUCTOR CORP./ 2900 SEMICONDUCTOR DR./ SANTA CLARA CA 95051/ (408) 737-5228
 95051 RONALD L DANIELSON/ DEPARTMENT OF EECS/ UNIVERSITY OF SANTA CLARA/ SANTA CLARA CA 95051/ (408) 984-4181
 95051 AL HARTMANN/ INTEL CORPORATION/ 3065 BOWERS AVENUE/ SANTA CLARA CA 95051/ (408) 246-7501
 95051 DEAN SCHULZ/ INTEL CORPORATION/ 3065 BOWERS AVENUE/ SANTA CLARA CA 95051/ (408) 246-7501
 95051 E. HAROLD WILLIAMS/ M.S. 690/ NATIONAL SEMICONDUCTOR CORP./ 2900 SEMICONDUCTOR DRIVE/ SANTA CLARA CA 95051/ (408) 737-5228

95054 FRITHJOF KOLBERG/ BOX 4802/ SANTA CLARA CA 95054/ (408) 255-0900 X2794
95060 W. TYLER/ 200 SEABORG PLACE/ SANTA CRUZ CA 95060/ (408) 925-0206
95121 DADO BANATAO/ 3060 BILBO DRIVE/ SAN JOSE CA 95121/ (408) 227-9027
95131 D. H. SPRINGER/ COMPUTER SYSTEMS DIVISION/ ANDERSON JACOBSON INC./ 521 CHARCOT AVENUE/ SAN JOSE CA 95131/ (408) 263-8520
95133 JOHN H. SPANTON/ 2351 RAVINE DRIVE/ SAN JOSE CA 95133/ (408) 258-6763
95153 TOM PITTMAN/ ITTY BITTY COMPUTERS/ P.O. BOX 23189/ SAN JOSE CA 95153
95376 TOM HORSLEY/ 1750 MELLO COURT/ TRACY CA 95376
95404 GARY LOWELL/ 2625 HIDDEN VALLEY/ SANTA ROSA CA 95404/ (707) 544-6373
95521 KENNETH A. DICKEY/ 1662 STROMBERG/ ARCATA CA 95521/ (707) 822-3986
95819 DAVID HILL/ COMPUTER CENTER/ SCI 319/ CALIFORNIA STATE UNIV. - SACRAMENTO/ 6000 J STREET/ SACRAMENTO CA 95819
95926 ORLANDO S. MADRIGAL/ DEPARTMENT OF COMPUTER SCIENCE/ CALIFORNIA STATE UNIVERSITY AT CHICO/ CHICO CA 95926/ (916) 895-6442
96822 W. W. PETERSON/ DEPT OF ICS/ U OF HAWAII/ 2565 THE MALL/ HONOLULU HI 96822/ (808) 948-7420
97005 STEPHEN A. DUM/ 16820 S.W. CAMBRIDGE COURT/ BEAVERTON OR 97005/ (503) 642-1168
97005 PETER H. MACKIE/ PHM AND ASSOCIATES/ P.O. BOX 427/ BEAVERTON OR 97005/ (503) 645-2282
97068 WILLIAM C. PRICE/ 28282 SW MOUNTAIN ROAD/ WEST LINN OR 97068
97077 ROY CARLSON/ (50-454)/ TEKTRONIX/ P.O. BOX 500/ BEAVERTON OR 97077
97077 TERRY HAMM/ M.S. 60-456/ TEKTRONIX INC./ P.O. BOX 500/ BEAVERTON OR 97077/ (503) 638-3411 X2579
97077 DON HARVEY/ MSG-WILSONVILLE/ 60-171 TEKTRONIX/ BOX 500/ BEAVERTON OR 97077
97077 NORM P. KERTH/ MS 58-736/ TEKTRONIX INC./ P.O. BOX 500/ BEAVERTON OR 97077
97077 PAUL L. MCCULLOUGH/ TEKTRONIX 60/666/ P.O. BOX 500/ BEAVERTON OR 97077/ (503) 638-3411 X2397
97077 LYNN SAUNDERS/ MS 39-135/ TEKTRONIX INC./ P.O. BOX 500/ BEAVERTON OR 97077/ (503) 644-0161 X6640
97077 ROD STEEL/ MS 60-456/ TEKTRONIX INC./ P.O. BOX 500/ BEAVERTON OR 97077/ (503) 638-3411 X2516
97123 JOHN L. RUTIS/ RT 3 BOX 292/ HILLSBORO OR 97123
97210 ATTENTION: EVAN L. SOLLEY/ THE LIFE SUPPORT SYSTEMS GROUP LTD./ 2432 NW JOHNSON/ PORTLAND OR 97210/ (503) 226-3515
97210 PAUL O-BRIEN/ P.O. BOX 10572/ PORTLAND OR 97210/ (503) 244-7538
97212 BOB PHILLIPS/ 2009 N.E. BRAZEE/ PORTLAND OR 97212/ (503) 284-8369
97217 DAVID WOLFE/ 7019 N. CHASE AVE./ PORTLAND OR 97217/ (503) 289-1228
97221 BARRY SMITH/ COMPUTING/ OREGON MUSEUM OF SCIENCE AND INDUSTRY/ 4015 SW CANYON ROAD/ PORTLAND OR 97221/ (503) 248-5923
97229 DAVID ROWLAND/ ELECTRO SCIENTIFIC INDUSTRIES/ 13900 N.W. SCIENCE PARK DRIVE/ PORTLAND OR 97229
97331 ATTENTION: WILLIAM HUNTEMAN/ COMPUTER CENTER/ OREGON STATE U/ CORVALLIS OR 97331/ (503) 754-2494
97331 DAVID F. CANTLEY/ DEPT. OF COMPUTER SCIENCE/ OREGON STATE UNIV./ CORVALLIS OR 97331
97331 KURT KOHLER/ MILNE COMPUTER CENTER/ OREGON STATE UNIV./ CORVALLIS OR 97331/ (503) 754-3474
97403 ATTN: DOCUMENTS ROOM/ COMPUTER SCIENCE DEPARTMENT/ U OF OREGON/ EUGENE OR 97403/ (503) 686-4394
97403 VERNON CHI/ ELECTRONICS SHOP/SCIENCE SERVICES/ 4 SCIENCE 1/ UNIVERSITY OF OREGON/ EUGENE OR 97403
98004 BILLY R. CASON/ 11521 NE 20TH STREET/ BELLEVUE WA 98004/ (206) 454-4846
98004 LESLIE R. KERR/ SOFTWARE DESIGN/ 10545 WOODHAVEN LANE/ BELLEVUE WA 98004/ (206) 455-3068
98006 JOHN D. WOOLLEY/ 6722 128TH AVE. SE/ BELLEVUE WA 98006/ (206) 641-3443
98043 GARY S. ANDERSON/ JOHN FLUKE MFG. CO. INC./ P.O. BOX 43210 M.S. 16/ MOUNTLAKE TER WA 98043/ (206) 774-2211 X353
98055 R. A. LOVESTEDT/ 20427 SE 192/ RENTON WA 98055/ (206) 237-1397
98055 RICHARD N. TAYLOR/ 10411 S.E. 174TH #3444/ RENTON WA 98055/ (206) 255-5856
98105 ATTN: COMPUTER CENTER USER SERVICES/ UNIVERSITY OF WASHINGTON/ 3737 BROOKLYN AVE. N.E. RM 15/ SEATTLE WA 98105
98117 ERIC SCHNELLMAN/ HONEYWELL MARINE SYSTEMS/ 5303 SHILSHOLE NW/ SEATTLE WA 98117
98124 ATTENTION: BLAIR BURNER/ MS 73-03/ BOEING COMPUTER SERVICES INC./ P.O. BOX 24346/ SEATTLE WA 98124/ (206) 773-8683
98124 ATTN: BOEING COMPANY/ 87-67 KENT TECHNICAL LIBRARY/ P.O. BOX 3999/ SEATTLE WA 98124
98124 DAVID DEMOREST/ M/S 8M-71/ BOEING COMPUTER SERVICES/ P.O. BOX 24346/ SEATTLE WA 98124/ (206) 244-6923/ (206) 773-2019
98177 CHARLES A. CASTELLOW/ 203 NW 176TH PLACE/ SEATTLE WA 98177/ (206) 546-1579
98195 HELLMUT GOLDE/ DEPT. OF COMP. SCI./ FR-35/ U OF WASHINGTON/ SEATTLE WA 98195/ (206) 543-9264
98195 JOE KELSEY/ COMPUTER SCIENCE TEACHING LABORATORY/ UNIVERSITY OF WASHINGTON/ MAIL STOP FR-35/ SEATTLE WA 98195/ (206) 543-2697
98195 JOHN S. SOBOLEWSKI/ RG-20/ LOCKE COMPUTER/ UNIVERSITY OF WASHINGTON/ SEATTLE WA 98195/ (206) 543-9275
98225 MARLIN PROWELL/ 3925 SILVER BEACH AVE./ BELLINGHAM WA 98225/ (206) 676-1554
99163 ROBERT E LORD/ COMPUTER CENTER/ WASHINGTON STATE UNIV./ PULLMAN WA 99163
2006 AUSTRALIA A. J. GERBER/ BASSER DEPT. OF COMPUTER SCIENCE/ UNIVERSITY OF SYDNEY/ SYDNEY N.S.W. 2006/ AUSTRALIA
2006 AUSTRALIA CARROLL MORGAN/ BASSER DEPT. OF COMPUTER SCIENCE/ U OF SYDNEY/ SYDNEY N.S.W. 2006/ AUSTRALIA
2006 AUSTRALIA BRIAN G. ROWSWELL/ UNIVERSITY COMPUTER CENTRE/ UNIVERSITY OF SYDNEY/ SYDNEY N.S.W. 2006/ AUSTRALIA/ 692 3491
2007 AUSTRALIA E. H. DOBELL/ COMPUTER CENTRE/ NSW INSTITUTE OF TECHNOLOGY/ P.O. BOX 123/ BROADWAY N.S.W. 2007/ AUSTRALIA/ (02) 218 9438
2033 AUSTRALIA ATTN: LIBRARIAN/ COMPUTING SERVICES UNIT/ UNIV. OF N.S.W./ P.C. BOX 1/ KENSINGTON N.S.W. 2033/ AUSTRALIA
2033 AUSTRALIA KEN ROBINSON/ DEPT. OF COMPUTER SCIENCE/ UNIVERSITY OF NEW SOUTH WALES/ P.O. BOX 1/ KENSINGTON N.S.W. 2033/ AUSTRALIA/ 663 0351
2232 AUSTRALIA JEFFREY TOBIAS/ APPLIED MATHS AND COMPUTING DIV./ AUST. ATOMIC ENERGY COMM. RES. EST./ PRIVATE MAIL BAG/ SUTHERLAND N.S.W. 2232/ AUSTRALIA/ 531-0111
2308 AUSTRALIA ATTN: SERIALS LIBRARY/ AUCHMUTY LIBRARY/ UNIVERSITY OF NEWCASTLE/ NEWCASTLE N.S.W. 2308/ AUSTRALIA/ 685745
2308 AUSTRALIA J. A. CAMPBELL/ MATHEMATICS DEPT./ UNIVERSITY OF NEWCASTLE/ NEWCASTLE N.S.W. 2308/ AUSTRALIA
2308 AUSTRALIA JOHN A. LAMBERT/ COMPUTING CENTRE/ UNIVERSITY OF NEWCASTLE/ NEWCASTLE N.S.W. 2308/ AUSTRALIA
2500 AUSTRALIA J. REINFELDS/ COMPUTING SCIENCE/ UNIVERSITY OF WOLLONGONG/ P.O. BOX 1144/ WOLLONGONG N.S.W. 2500/ AUSTRALIA/ (042) 297311
2600 AUSTRALIA ATTN: PURCHASING OFFICE/ RESEARCH SCHOOL OF PHYSICAL SCIENCES/ AUSTRALIAN NATIONAL UNIVERSITY/ P.O. BOX 4/ CANBERRA A.C.T. 2600/ AUSTRALIA/ 492143
2600 AUSTRALIA A. J. HURST/ DEPT. OF COMPUTER SCIENCE/ AUSTRALIAN NATIONAL UNIVERSITY/ P.O. BOX 4/ CANBERRA A.C.T. 2600/ AUSTRALIA/ (062) 49 4625
2600 AUSTRALIA MALCOLM C. NEWBY/ COMPUTER CENTRE/ AUSTRALIAN NATIONAL UNIV./ P.O. BOX 4/ CANBERRA A.C.T. 2600/ AUSTRALIA/ 81-6376 / 49-4216
2601 AUSTRALIA ATTN: THE LIBRARIAN/ CSIRO/ DIV. OF COMPUTING RES./ P.O. BOX 1800/ CANBERRA CITY A.C.T. 2601/ AUSTRALIA

2616 AUSTRALIA ATTN: SCHOOL OF INFORMATION SCIENCES/ CANBERRA COLLEGE OF ADVANCED EDUCATION/ P.O. BOX NO. 1/ BELCONNEN A.C.T. 2616/ AUSTRALIA
3000 AUSTRALIA G. J. KNOX/ COMPUTER CENTRE/ ROYAL MELBOURNE INSTITUTE OF TECHNOLOG/ 124 LATROBE STREET/ MELBOURNE VICTORIA 3000/ AUSTRALIA/ 341.2292
3001 AUSTRALIA ATTN: CENTRAL LIBRARY/ FLOOR 1 CASEY WING/ ROYAL MELBOURNE INSTITUTE OF TECHNOLOG/ 376-392 SWANSTON STREET/ MELBOURNE VICTORIA 3001/ AUSTRALIA
3052 AUSTRALIA PETER RICHARDSON/ COMPUTER SCIENCE DEPT./ UNIV. OF MELBOURNE/ MELBOURNE VICTORIA 3052/ AUSTRALIA/ (03) 3415225
3052 AUSTRALIA ATTN: LIBRARIAN/ SCHOOL OF MATHEMATICAL SCIENCES/ RICHARD BERRY BUILDING/ UNIVERSITY OF MELBOURNE/ PARKVILLE VICTORIA 3052/ AUSTRALIA
3052 AUSTRALIA ANTHONY P. KYNE/ DEPT. OF COMPUTER SCIENCE/ UNIVERSITY OF MELBOURNE/ PARKVILLE VICTORIA 3052/ AUSTRALIA/ 345 1844
3052 AUSTRALIA PRABHAKER MATELI/ DEPT. OF COMPUTER SCIENCE/ UNIV. OF MELBOURNE/ PARKVILLE VICTORIA 3052/ AUSTRALIA/ (03)341-6459
3083 AUSTRALIA ATTN: COMPUTER CENTRE/ LA TROBE UNIVERSITY/ BUNDOORA VICTORIA 3083/ AUSTRALIA/ 478 3122
3165 AUSTRALIA GEOFFREY A. CLEAVE/ 18 NEIL COURT/ E. BENTLEIGH VICTORIA 3165/ AUSTRALIA
4067 AUSTRALIA D. B. JOHNSTON/ DEPT. OF COMPUTER SCIENCE/ UNIV. OF QUEENSLAND/ ST. LUCIA QUEENSLAND 4067/ AUSTRALIA/ 07/3706930
5000 AUSTRALIA C. A. RUSBRIDGE/ SOUTH AUSTRALIA INSTITUTE OF TECHNOLOG/ P.O. BOX 1/ INGLE FARM S.A. 5000/ AUSTRALIA/ AUSTRALIA 260-2055
5001 AUSTRALIA ATTN: PROGRAM LIBRARIAN/ COMPUTING CENTRE/ UNIVERSITY OF ADELAIDE/ BOX 498 G.P.O./ ADELAIDE S.A. 5001/ AUSTRALIA/ 61 822 34333 X2720/X2099
5001 AUSTRALIA YOUNG J. CHOI/ DEPT. OF COMPUTING SCIENCE/ UNIV. OF ADELAIDE/ ADELAIDE S.A. 5001/ AUSTRALIA/ 223-4333
5001 AUSTRALIA B. KIDMAN/ DEPT OF COMPUTER SCIENCE/ UNIVERSITY OF ADELAIDE/ GPO BOX 498/ ADELAIDE S.A. 5001/ AUSTRALIA/ 223 4333
5001 AUSTRALIA C. D. MARLIN/ COMPUTING SCIENCE DEPT./ UNIVERSITY OF ADELAIDE/ G.P.O. BOX 498/ ADELAIDE S.A. 5001/ AUSTRALIA/ 223 4333 X2762
5006 AUSTRALIA I. N. BLAVINS/ KATHLEEN LUMLEY COLLEGE/ FINNIS STREET/ NORTH ADELAID S.A. 5006/ AUSTRALIA
6005 AUSTRALIA PETER R. SUMNER/ INTERDATA COMPUTERS PTY. LTD./ 30 KINGS PARK RD./ WEST PERTH W.A. 6005/ AUSTRALIA/ (09) 322-3391
6009 AUSTRALIA J. S. ROHL/ DEPT. OF COMPUTER SCIENCE/ U OF WESTERN AUSTRALIA/ NEDLANDS W.A. 6009/ AUSTRALIA
7001 AUSTRALIA ATTN: SECRETARY/ DEPARTMENT OF INFORMATION SCIENCE/ UNIVERSITY OF TASMANIA/ GPO BOX 252C/ HOBART TASMANIA 7001/ AUSTRALIA
7001 AUSTRALIA A. H. J. SALE/ DEPT. OF INFORMATION SCIENCE/ UNIVERSITY OF TASMANIA/ BOX 252C/ HOBART TASMANIA 7001/ AUSTRALIA/ 23 0561
A-1040 AUSTRIA HELMUT SCHAUER/ TU WEIN/ ARGENTINIERSTR. 8/ WIEN A-1040/ AUSTRIA/ 0222/6587 31 316
A-1040 AUSTRIA ADA SZER/ INSTITUT F. INFORMATIONS-SYSTEME/ ARGENTINIERSTR. 8/ WIEN A-1040/ AUSTRIA/ 65 87 31/313
A-1150 AUSTRIA KONRAD MAYER/ REICHSAPFELG 13/8/ VIENNA A-1150/ AUSTRIA
A-4060 AUSTRIA KARL PRAGERSTORFER/ EDERACKERSTRASSE 11/7/ LEONDING A-4060/ AUSTRIA
A-5020 AUSTRIA FRANZ W. MAIER/ ZENTRUM FUER EDV/ UNIVERSITAET SALZBURG/ PETERSBRUNNSTR. 19/ SALZBURG A-5020/ AUSTRIA/ 06222/44511/343
BELGIUM O. BEAUFAYS/ MATHÉMATIQUES APPLIQUÉES/ C P I 165/ UNIVERSITE LIBRE DE BRUXELLES/ AVENUE F.-D. ROOSEVELT 50/ BRUXELLES B-1050/ BELGIUM
B-1170 BELGIUM ALAIN PIROTTE/ MBLE/RESEARCH LABORATORY/ AVENUE EM. VAN BECELAERE 2/ BRUSSELS B-1170/ BELGIUM/ 673.41.90/ 673.41.99
B-2000 BELGIUM RAYMAOND BOUTE/ FRANKRIJKLEI 96A - BUS 24/ ANTWERPEN B-2000/ BELGIUM/ 031/317445
B-3030 BELGIUM JOHAN LEWIS/ AFD. TOEGEPASTE WISKUNDE EN PROGRAMMAT/ KATHOLIEKE UNIV. LEUVEN/ CELESTYNERLAAN 200B/ HEVERLEE B-3030/ BELGIUM/ 0032/16/235821
B-3030 BELGIUM P. VERBAETEN/ APPLIED MATH. AND PROGRAMMING DIV./ K U LEUVEN/ CELESTYNERLAAN 200B/ HEVERLEE B-3030/ BELGIUM
13100 BRAZIL JOSE OSVALDO FERRARI/ IMECC/ UNICAMP/ C.P. 1170/ CAMPINAS SP 13100/ BRAZIL/ 31-4555
13100 BRAZIL ROGERIO BURNIER FILHO/ RUA MARIA MONTEIRO 223/ CAMPINAS SP 13100/ BRAZIL
13100 BRAZIL PALTONIO DAUN FRAGA/ IMECC/ UNICAMP/ C.P. 1170/ CAMPINAS SP 13100/ BRAZIL/ PABX 31-4555
13100 BRAZIL FERNANDO ANTONIO VANINI/ IMECC/ UNICAMP/ C.P. 1170/ CAMPINAS SP 13100/ BRAZIL/ 31-4555
13560 BRAZIL SERGIO DE MELLO SCHNEIDER/ DEPARTAMENTO DE COMPUTACAO/ UNIVERSIDADE FEDERAL DE SAO CARLOS/ SAO CARLOS SP 13560/ BRAZIL
A1C 5S7 CANADA R. JAMES DAVE/ MATH STAT AND COMP. SCI./ MEMORIAL UNIV. OF NEWFOUNDLAND/ ST. JOHN'S NEWFOUNDLA A1C 5S7/ CANADA/ (709) 753-1200 EXT. 2767
A1C 5S7 CANADA RANDY DODGE/ COMPUTING SERVICES/ MEMORIAL UNIVERSITY/ ST. JOHN'S NEWFOUNDLA A1C 5S7/ CANADA/ (709) 753-1200 X2746
A1C 5S7 CANADA F. G. PAGAN/ COMPUTER SCIENCE/ MEMORIAL UNIVERSITY/ ST. JOHN'S NEWFOUNDLA A1C 5S7/ CANADA
G1W 2P3 CANADA JEAN CASTONGUAY/ 3140 AVENUE FRANCE-PRIME #202/ STE-FOY QUEBEC G1W 2P3/ CANADA
H1Y 3C3 CANADA CARLO LOCICERO/ 6426 MOLSON/ MONTREAL QUEBEC H1Y 3C3/ CANADA/ (514) 727-3110
H1Z 3P1 CANADA SERGE FROMENT/ 9142 QUINZIEME AVENUE/ MONTREAL QUEBEC H1Z 3P1/ CANADA/ (514) 321-2482
H3C 3J7 CANADA JEAN VAUCHER/ DEPARTEMENT D'INFORMATIQUE/ UNIVERSITE DE MONTREAL/ C.P. 6128 - STATION A/ MONTREAL QUEBEC H3C 3J7/ CANADA/ (514) 343-7092
H3C 3J7 CANADA PATRICK WARD/ CENTRE DE CALCUL/ UNIVERSITE DE MONTREAL/ C.P. 6128/ MONTREAL QUEBEC H3C 3J7/ CANADA/ (514) 343-6866
H3C 3J7 CANADA PIERRE DESJARDINS/ INFORMATIQUE/ UNIVERSITE DE MONTREAL/ C.P. 6128/ MONTREAL 101 QUEBEC H3C 3J7/ CANADA/ (514) 343-7662
H3G 1M8 CANADA DAVID KARL PROBST/ COMPUTER SCIENCE DEPT./ CONCORDIA UNIVERSITY/ 1455 DE MAISONNEUVE W./ MONTREAL QUEBEC H3G 1M8/ CANADA/ (514) 733-4921
H3G 1M8 CANADA J. W. ATWOOD/ DEPT OF COMP. SCI.:#963-10/ CONCORDIA UNIVERSITY/ 1455 DE MAISONNEUVE BLVD. WEST/ MONTREAL QUEBEC H3G 1M8/ CANADA/ (514) 879-8130
H3G 2C8 CANADA ERIC MELBARDIS/ 3467 AVE. DU MUSEE - #206/ MONTREAL QUEBEC H3G 2C8/ CANADA
H3P 3B9 CANADA IAN MACMILLAN/ P.O. BOX 128/ MOUNT ROYAL QUEBEC H3P 3B9/ CANADA
H4V 2H3 CANADA PETER GROGONO/ THE SOUND MACHINE/ 4877 ROSEDALE AVENUE/ MONTREAL QUEBEC H4V 2H3/ CANADA/ (514) 489-9995/ (514) 879-4251 (WORK)
H9R 1G1 CANADA RICHARD WEST/ SMALL TERMINAL ENGINEERING/ COMTERM LIMITED/ 147 HYMUS BLVD./ MONTREAL QUEBEC H9R 1G1/ CANADA/ (514) 697-0810 X227
H9R 1G1 CANADA EDWIN TSE/ 525 DELMAR ST./ POINTE CLAIRE QUEBEC H9R 1G1/ CANADA/ (514) 697-1320
J1K 2R1 CANADA JACQUES HAGUEL/ FACULTE DES SCIENCES/ UNIVERSITE DE SHERBROOKE/ SHERBROOKE QUEBEC J1K 2R1/ CANADA/ (819) 563-4635
K1A 0N8 CANADA BARRY SEARLE/ SECTION TASK/ TOWER C FLOOR 10C/ TRANSPORT CANADA/ PLACE DE VILLE/ OTTAWA ONTARIO K1A 0N8/ CANADA/ (613) 996-0218
K1J 6L2 CANADA D. B. COLDRICK/ COMPUTATION CENTRE/ BLDG. M-60/ NATIONAL RESEARCH COUNCIL/ MONTREAL ROAD/ OTTAWA ONTARIO K1J 6L2/ CANADA/ (613) 993-3870
K1N 6N5 CANADA LUIGI LOGRIPPO/ COMP. SCI. DEPT./ U OF OTTAWA/ OTTAWA ONTARIO K1N 6N5/ CANADA
K1N 6N5 CANADA H. TAYLOR/ COMPUTING CENTRE/ APPLICATIONS DEPT./ U OF OTTAWA/ OTTAWA ONTARIO K1N 6N5/ CANADA
K1S 5G3 CANADA SAM WILMOTT/ APT. 501/ 463 CAMBRIDGE SOUTH/ OTTAWA ONTARIO K1S 5G3/ CANADA
K2E 6T7 CANADA ATTENTION: DONALD LINDSAY/ DYNALOGIC CORPORATION LIMITED/ 141 BENTLEY AVENUE/ OTTAWA ONTARIO K2E 6T7/ CANADA/ (613) 226-1383
K2K 1K2 CANADA FRANKLIN B. DE GRAAF/ 6 CARMICHAEL COURT/ KANATA ONTARIO K2K 1K2/ CANADA/ (613) 592-5793
K7L 3N6 CANADA ATTN: REFERENCE ROOM/ COMPUTING AND INF. SCI./ QUEEN'S UNIVERSITY/ KINGSTON ONTARIO K7L 3N6/ CANADA
K7L 3N6 CANADA JACK HUGHES/ COMPUTING CENTRE/ DUPUIS HALL/ QUEEN'S UNIVERSITY/ KINGSTON ONTARIO K7L 3N6/ CANADA/ (613) 547-2800
K7L 3N6 CANADA R. D. TENNENT/ DEPT. OF COMPUTING AND INFORMATION SCI/ QUEEN'S UNIVERSITY/ KINGSTON ONTARIO K7L 3N6/ CANADA
L8N 3W3 CANADA MARK GREEN/ #904 - 123 CHARLTON AVE. E/ HAMILTON ONTARIO L8N 3W3/ CANADA/ (416) 522-2512
L8S 4K1 CANADA N. SOLTNSEFF/ DEPT. OF APPLIED MATH./ MCMASTER UNIVERSITY/ HAMILTON ONTARIO L8S 4K1/ CANADA/ (416) 525-9140 X4689
L8S 4K1 CANADA ATTENTION: CHRIS BRYCE/ APPLIED MATH. COMPUTER LAB/ MCMASTER UNIVERSITY/ HAMILTON ONTARIO L8S 4K1/ CANADA/ (416) 525-9140 X4689
M3J 1P3 CANADA GEOFFREY HUNTER/ CHEMISTRY DEPT./ YORK UNIVERSITY/ DOWNSVIEW ONTARIO M3J 1P3/ CANADA/ (416) 667-3852

M3M 3B9 CANADA ATTENTION: MARTIN TUORI/ BEH. SCI. DIV./ DEFENCE AND CIVIL INST. OF ENVIRONMENT/ P.O. BOX 2000/ DOWNSVIEW ONTARIO M3M 3B9/ CANADA
(416) 633-4240 X204 (OFFICE)/ X238 (LAB)

M5S 1A7 CANADA ATTN: M. DOHERTY/ 128 TECHNICAL REFERENCE CENTER/ UNIVERSITY OF TORONTO COMPUTER CENTER/ 10 KINGS COLLEGE ROAD/ TORONTO ONTARIO M5S 1A7/ CANADA
(416) 978-8995

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

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

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

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

N2L 3B8 CANADA DELE AYENI/ 316-102 SEAGRAM DRIVE/ WATERLOO ONTARIO N2L 3B8/ CANADA/ (509) 884-4679

N2L 3G1 CANADA W. MORVEN GENTLEMAN/ MATHEMATICS COMPUTING FACILITY/ UNIVERSITY OF WATERLOO/ WATERLOO ONTARIO N2L 3G1/ CANADA/ (519) 578-8866

N2L 3G1 CANADA KAY HARRISON/ COMPUTER CENTER/ 1088B M AND C/ U OF WATERLOO/ WATERLOO ONTARIO N2L 3G1/ CANADA

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

N6A 5B7 CANADA F. CELLINI/ DEPT. OF COMPUTER SCIENCE/ UNIVERSITY OF WESTERN ONTARIO/ LONDON ONTARIO N6A 5B7/ CANADA/ (519) 679-6051

N6A 5B7 CANADA S. WILLIAMSON/ NATURAL SCIENCE CENTRE/ 223 COMPUTING CENTER/ UNIVERSITY OF WESTERN ONTARIO/ LONDON ONTARIO N6A 5B7/ CANADA/ (519) 679-2151

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

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

R3T 2N2 CANADA W. BRUCE FOULKES/ DEPARTMENT OF COMPUTER SCIENCE/ THE UNIVERSITY OF MANITOBA/ WINNIPEG MANITOBA R3T 2N2/ CANADA/ (204) 269-3363

S7K 3P7 CANADA WILLIAM BLAMPED/ SED SYSTEMS LTD/ P.O. BOX 1464/ SASKATOON SASK. S7K 3P7/ CANADA/ (306) 244-0976 X37

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

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

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

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

T6G 2N5 CANADA C. A. MILLER/ DEPT. OF PHYSICS NUCLEAR RES. CTR./ UNIV. OF ALBERTA/ EDMONTON ALBERTA T6G 2N5/ CANADA

V5N 3X1 CANADA ALAN LILLICH/ 3477 BELLA VISTA/ VANCOUVER B.C. V5N 3X1/ CANADA

V6T 1W5 CANADA ROBERT A. FRALEY/ DEPT. OF COMPUTER SCIENCE/ U OF BRITISH COLUMBIA/ VANCOUVER B.C. V6T 1W5/ CANADA/ (604)@228-2083

V6T 1W5 CANADA BARY W. POLLACK/ DEPT. OF COMP. SCI./ U OF BRITISH COLUMBIA/ 2075 WEBBROOK PLACE/ VANCOUVER B.C. V6T 1W5/ CANADA/ (604) 228-6794

V6X 2Z9 CANADA WAYNE FUNG/ NOOTKA BLDG./ MACDONALD DETTWILER AND ASSOC. LTD./ 10280 SHELLBRIDGE WAY/ RICHMOND B.C. V6X 2Z9/ CANADA/ (604) 278-3411

V6X 2Z9 CANADA DOUG TEEPLE/ NOOTKA BUILDING/ MACDONALD DETTWILER & ASSOC. LTD./ 10280 SHELLBRIDGE WAY/ RICHMOND B.C. V6X 2Z9/ CANADA/ (604) 278-3411

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

V8P 5J2 CANADA GORDON STUART/ TECHNICAL AND VOCATIONAL INST./ CAMOSUN COLLEGE/ 1950 LANSDOWNE RD./ VICTORIA B.C. V8P 5J2/ CANADA/ (604) 592-1281 X248

CHILE ATTN: CENTRO DE CIENCIAS DE LA COMPUTA/ UNIVERSIDAD CATHOLICA DE CHILE/ CASILLA 114-D/ SANTIAGO/ CHILE/ 513548

DK-1601 DENMARK LARS EKMAN/ EDB-SEKRETARIATET/ DANISH POST AND TELEGRAPH OFFICE/ VESTER FARMAGSGADE 37.3/ KOBENHAVN V DK-1601/ DENMARK/ (01) 14 51 66

DK-2000 DENMARK PHILIP PARKER/ VAGTELVEJ 59 ST. MF./ COPENHAGEN F DK-2000/ DENMARK/ (01) 34 00 58

DK-2100 DENMARK ATTN: SOFTWARE/HARDWARE GROUP/ EMILIUS MOLLER AS-NCR/ TEGLVAERKSGADE 31/ COPENHAGEN DK-2100/ DENMARK/ (01) 29 15 55

DK-2100 DENMARK JORGEN OXENBOLL/ RC4000 APELDELING/ H.C. ORSTED INSTITUTET/ UNIVERSITETSPARKEN 5/ KOBENHAVN O DK-2100/ DENMARK

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

DK-2300 DENMARK ANDERS WEBER/ EBERTSGADE 2 - 3STH/ KOBENHAVN S. DK-2300/ DENMARK

DK-2500 DENMARK ATTENTION: JAN LAUGESEN V. 3-357/ I/S DATACENTRALUM AF 1959/ RETORTVEJ 8/ VALBY DK-2500/ DENMARK/ (01) 46 81 22

DK-2650 DENMARK G. RICHARD BLADEN/ GILLESAGER 226 ST TV/ HUIDOVRE DK-2650/ DENMARK/ (01) 75 79 15

DK-2650 DENMARK NIELS WINTHER/ REBAEK SOPARK 5-544/ HVIDOVRE DK-2650/ DENMARK

DK-2730 DENMARK JENS PETER RINGGAARD/ CHRISTIAN ROVSING A/S/ MARIELUNDVEJ 46 B/ HERLEV DK-2730/ DENMARK/ 02 91 88 33

DK-2800 DENMARK ATTN: BIBLIOTEKET/ NEUCC/ BUILDING 305/ TECHNICAL UNIV. OF DENMARK/ LYNGBY DK-2800/ DENMARK/ 02 88 12 77

DK-2800 DENMARK KELD HELBIG HANSEN/ NEUCC/ TECHNICAL UNIV. OF DENMARK/ BUILDING 305/ LYNGBY DK-2800/ DENMARK/ (02) 88 12 77

DK-2800 DENMARK GUNNAR JOHANSEN/ DEPT. OF CHEMISTRY AND CHEM. ENG./ DANISH ENGINEERING ACADEMY/ BYGNING 375/ LYNGBY DK-2800/ DENMARK

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

DK-8000 DENMARK ATTN: RECAU/ NY MUNKEGADE/ AARHUS C DK-8000/ DENMARK/ 06-128355

DK-8000 DENMARK U. HAMMELEFF/ DET REGIONALE EDB-CENTER/ RECAU/ AARHUS UNIVERSITET/ NY MUNKEGADE/ AARHUS C DK-8000/ DENMARK/ 45 6 12 83 55

DK-8000 DENMARK ATTN: RECAU (B)/ NY MUNKEGADE/ AARHUS C. DK-8000/ DENMARK/ 06-12 83 55

DK-8200 DENMARK HOLGER NIELSEN/ OSLOGADE 6 II/ AARHUS N DK-8200/ DENMARK

DK-9000 DENMARK UFFE MOLLER/ DATANOMUDDANNELSEN/ SOHNGAARDSHOLMSVEJ 57/ AALBORG DK-9000/ DENMARK/ (08) 14 12 06

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

DK-9000 DENMARK KLAUS ILLUM/ INSTITUT 4/ ALBORG UNIVERSITETSCENTER/ BADEHUSVEJ 1B/ ALBORG DK-9000/ DENMARK/ 08-138788

SF-00130 FINLAND HEIKKI KASKELMA/ OY SOFTPLAN AB/ EROTTAJANKATU 9 A/ HELSINKI SF-00130/ FINLAND/ 90-644306

SF-00330 FINLAND ANTTI SALAVA/ MUNKKINIEMEN PULSTOTIE 17 - A 13/ HELSINKI 33 SF-00330/ FINLAND/ 90-486288

SF-20500 FINLAND MARKKU SUNI/ COMPUTER CENTRE/ UNIVERSITY OF TURKU/ TURKU 50 SF-20500/ FINLAND/ 921-335 599

SF-33101 FINLAND JUHA HEINANEN/ COMPUTER CENTER/ UNIVERSITY OF TAMPERE/ P.O. BOX 607/ TAMPERE 10 SF-33101/ FINLAND/ 931-651595

F-06034 FRANCE O. LECARME/ I.M.A.N./ UNIVERSITE DE NICE/ PARC VALROSE/ NICE CEDEX F-06034/ FRANCE/ 51 91 00

F-31077 FRANCE MICHEL GALINIER/ INFORMATIQUE/ UNIVERSITE P. SABATIER/ 118 ROUTE DE NARBONNE/ TOULOUSE CEDEX F-31077/ FRANCE/ 16-61-53 11 20

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

F-34000 FRANCE ATTN: C.R.I.G./ INSTITUT UNIVERSITAIRE DE TECHNOLOGIE/ MONTEPELLIER F-34000/ FRANCE

F-35031 FRANCE JEAN BEZIVIN/ DEPARTEMENT DE MATHEMATIQUES & INFORMATIQUE/ UNIVERSITE DE RENNES/ RENNES CEDEX F-35031/ FRANCE/ 36.48.15

F-38040 FRANCE JEAN-PIERRE FAUCHE/ DEPARTEMENT INFORMATIQUE/ IREP/ BOITE POSTALE 47/ GRENOBLE CEDEX F-38040/ FRANCE

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

F-75005 FRANCE DIDIER THIBAUT/ 17 RUE GAY-LUSSAC/ PARIS F-75005/ FRANCE/ 527 16 85

F-75230 FRANCE JACQUES FARRE/ T 55.65/ INSTITUT DE PROGRAMMATION/ 4 PLACE JUS SIEU/ PARIS CEDEX 05 F-75230/ FRANCE/ 336 25 25 X58 77

D-1000 GERMANY HUBERT LEYGRAF/ INSTITUT FUR ETALLURGIE/ TECHNISCHE UNIVERSITAT BERLIN/ JOACHIMSALER STR. 31/32/ BERLIN D-1000/ GERMANY

D-1000 GERMANY ALBRECHT BIEDL/ INSTITUT FUR SOFTWARE/ DV-GRUNDAUSBILDUNG/ TECHNISCHE UNIVERSITAT BERLIN / VSH 5/ OTTO-SUHR-ALLEE 18/20/ BERLIN 10 D-1000/ GERMANY
 D-1000 GERMANY ROLF SCHUMACHER/ JEBENSSTR. 1/ BERLIN 10 D-1000/ GERMANY/ 030 393 18 12
 D-1000 GERMANY THOMAS HABERNOLL/ TURMSTR. 19/ BERLIN 21 D-1000/ GERMANY/ (030) 394 56 91
 D-1000 GERMANY WOLFGANG HAMPE/ WILHELMSHAVENER STR. 47/ BERLIN 21 D-1000/ GERMANY
 D-1000 GERMANY LUTZ CHRISTOPH/ SCHUTZALLEE 52/ BERLIN 37 D-1000/ GERMANY/ (030) 811-1743
 D-1000 GERMANY WERNER F. PRAUTSCH/ ALBERTINENSTRASSE 29/ BERLIN 37 D-1000/ GERMANY/ (030) 801 11 88
 D-1000 GERMANY THOMAS WAGNER/ AHORNSTRASSE 16/ BERLIN 41 D-1000/ GERMANY/ (030) 7925361
 D-1000 GERMANY KAY BITTERLING/ SCHOENBURGSTR. 1/ BERLIN 42 D-1000/ GERMANY/ (030) 7524517
 D-1000 GERMANY PETER NELLESEN/ MARTIN-OPITZ STR. 20/ BERLIN 65 D-1000/ GERMANY/ (030) 39393593
 D-2000 GERMANY ATTN: INSTITUT FUER INFORMATIK/ UNIVERSITAT HAMBURG/ SCHLUETER STRASSE 70/ HAMBURG 13 D-2000/ GERMANY
 D-2000 GERMANY GERHARD FRIESLAND/ INSTITUT FUER INFORMATIK/ UNIVERSITAT HAMBURG/ SCHLUTERSTRASSE 66-72/ HAMBURG 13 D-2000/ GERMANY/ 040-4123 X4170
 D-2000 GERMANY H.-H. NAGEL/ INSTITUT FUER INFORMATIK/ UNIVERSITAT HAMBURG/ SCHLUTERSTRASSE 66-72/ HAMBURG 13 D-2000/ GERMANY
 D-2000 GERMANY THOMAS BERNER/ BURGERWEIDE 77/ HAMBURG 26 D-2000/ GERMANY/ 040-2506602
 D-2000 GERMANY PETER KLAUBERG/ C/O HAMBURGISCHE ELECTRICITAETS-WERKE/ UEBERSEERING 12/ HAMBURG 60 D-2000/ GERMANY/ 040 636 2614
 D-2000 GERMANY CARSTEN KOCH/ DISTRIKT NORD/ CONTROL DATA GMBH/ UBERSEERING 13/ HAMBURG 60 D2000/ GERMANY/ 630 80 21 - 25
 D-2000 GERMANY CARSTEN KOCH (B)/ OERTZWEIG 32/ HAMBURG 60 D-2000/ GERMANY/ 6901884
 D-2000 GERMANY BERND BRUGGE/ VIELOHWEG 164/ HAMBURG 61 D-2000/ GERMANY
 D-2000 GERMANY KLAUS LIEBENWALD/ BOHMESTRASSE 8/ HAMBURG 70 D-2000/ GERMANY/ 040-686036
 D-2000 GERMANY BERNHARD NEBEL/ STEGLITZER STR. 17F/ HAMBURG 70 D-2000/ GERMANY/ 040/664911
 D-3000 GERMANY ROLF SONNTAG/ RICHARD WAGNER STR. 27/ HANNOVER 1 D-3000/ GERMANY
 D-3000 GERMANY G. MARQUARDT/ REGIONALES RECHENZENTRUM/ WUNSTORFER STR. 14/ HANNOVER 91 D-3000/ GERMANY
 D-5000 GERMANY DIETRICH KREKEL/ RECHEN ZENTRUM/ UNIVERSITAT ZU KOLN/ ROBERT KOCH STR 10/ KOLN 41 D-5000/ GERMANY/ 0221/478/5587
 D-5100 GERMANY ATE PHUNG/ KREFELDER STR. 23/ AACHEN D-5100/ GERMANY
 D-5205 GERMANY HORST SANTO/ POSTFACH 1240/ GMD MBH/ SCHLOSS BIRLINGHOVEN/ ST. AUGUSTIN 1 D-5205/ GERMANY
 D-5300 GERMANY G. ENGELIEN/ MAX-PLANCK-INSTITUT FUR RADIOASTRONOMI/ AUF DEM HUGEL 69/ BONN 1 D-5300/ GERMANY
 D-6100 GERMANY H.-J. HOFFMANN/ FACHBEREICH INFORMATIK/ TECHNISCHE HOCHSCHULE DARMSTADT/ STEUBENPLATZ 12/ DARMSTADT D-6100/ GERMANY/ (06151) 163410
 D-6300 GERMANY DIETER WEISS/ HOCHSCHULRECHENZENTRUM (HRZ)/ DER JUSTUS LIEBIG-UNIVERSITAT/ LEIHGESTERNER WEG 217/ GIESSEN D-6300/ GERMANY/ (0641) 702-2514
 D-6750 GERMANY ATTN: BIBLIOTHEK/ UNIVERSITAT KAISERSLAUTERN/ P.O. BOX 2040/ KAISERSLAUTER D-6750/ GERMANY/ (0631) 8541
 D-6750 GERMANY HANS-WILM WIPPERMANN/ INFORMATIK/ F13/ UNIV. OF KAISERSLAUTERN/ PFAFFENBERGSTR. 95/ KAISERSLAUTER D-6750/ GERMANY/ (0631) 854 2635
 D-7000 GERMANY WALTER WEHINGER/ LANGUAGES AND PROCESSORS GROUP/ RECHENZENTRUM/ UNIVERSITAT STUTTGART/ PFAFFENWALDRING 64/ STUTTGART 80 D-7000/ GERMANY/ 0711-784 2507
 D-7408 GERMANY ASHOK N. ULLAL/ GOETHESTR. 10/ KUSTERINGEN D-7408/ GERMANY
 D-7500 GERMANY KARLHEINZ KAPP/ INFORMATIK/ UNIVERSITAT KARLSRUHE/ TRANSPORT-U. VERKEHRSSYSTEME/ KARLSRUHE D-7500/ GERMANY/ (0721) 608-3170/3898
 D-7500 GERMANY ROLF G. KNOEPKER/ GESELLSCHAFT FUER KERNFORSCHUNG/IDT/ P.O.B. 3640/ KARLSRUHE D-7500/ GERMANY
 D-7500 GERMANY MANFRED SEIFERT/ INFORMATIK III/ UNIVERSITAT KARLSRUHE/ ZIRKEL 2/ KARLSRUHE D-7500/ GERMANY/ 0721/608-3982
 D-7500 GERMANY ATTN: INST. FUR ANGEWANDTE MATHEMATIK/ UNIVERSITAT KARLSRUHE (TH)/ KAISERSTR. 12 - POSTFACH 6380/ KARLSRUHE 1 D-7500/ GERMANY
 D-7500 GERMANY LUCIEN FEIEREISEN/ HAID-6-NEU-STR. 16 / W 81/ KARLSRUHE 1 D-7500/ GERMANY
 D-7500 GERMANY GERHARD GOOS/ INSTITUT FUER INFORMATIK II/ UNIVERSITAT KARLSRUHE/ POSTFACH 6380/ KARLSRUHE 1 D-7500/ GERMANY/ 0721/608-3970
 D-7500 GERMANY BRUNO LORTZ/ RECHENZENTRUM/ UNIVERSITAT KARLSRUHE/ ZIRKEL 2/ KARLSRUHE 1 D-7500/ GERMANY
 D-7500 GERMANY KLAUS R. DITTRICH/ UNIVERSITY KARLSRUHE/ DURMERSHEIMER STR. 77/ KARLSRUHE 21 D-7500/ GERMANY/ 0721-555506
 D-7750 GERMANY DIRK KRONIG/ AEG-TELEFUNKEN/ POSTFACH 2154/ KONSTANZ D-7750/ GERMANY/ 07531-862066
 D-8000 GERMANY MANFRED SOMMER/ DEPARTMENT D AP GE/ SIEMENS AG/ HOFFMANNSTRASSE E/ MUENCHEN D-8000/ GERMANY/ 089-722-61276
 D-8000 GERMANY HELLMUT WEBER/ LEIBNIZ-RECHENZENTRUM/ BARERSTRASSE 21/ MUENCHEN 2 D-8000/ GERMANY/ (089) 2105-8489
 D-8000 GERMANY PETER RAUSCHMAYER/ LUITPOLD-GYMNASIUM/ SEEAUSTR. 1/ MUENCHEN 22 D-8000/ GERMANY/ 226587
 D-8000 GERMANY MANFRED LUCKMANN/ ALEMANNENSTR. 24/ MUENCHEN 90 D-8000/ GERMANY
 D-8000 GERMANY E. DENERT/ SOFTLAB GMBH/ SEDERANGER 4-6/ MUNCHEN 22 D-8000/ GERMANY/ 089/347051-55
 D-8000 GERMANY S. ROHLFS/ SOFTLAB GMBH/ SEDERANGER 4-6/ MUNCHEN 22 D-8000/ GERMANY/ 089/347051-55
 D-8000 GERMANY P. SCHNUPP/ SOFTLAB GMBH/ SEDERANGER 4-6/ MUNCHEN 22 D-8000/ GERMANY/ 089/347051-55
 D-8000 GERMANY ATTENTION: JAN WITT/ ZFE FL SAR/ SIEMENS AG/ HOFMANNSTR. 51/ MUNCHEN 70 D-8000/ GERMANY/ (089) 722-22651
 D-8000 GERMANY WERNER REMMELE/ ZFE FL SAR 12/ SIEMENS AG/ HOFMANNSTR. 51/ MUNCHEN 70 D-8000/ GERMANY
 D-8000 GERMANY ATTN: INSTITUT FUR MED. DATENVERARBEIT/ STRAHLEN- UND UMWELTFORSCHUNG GMBH/ ARABELLSTR. 4/1/ MUNCHEN 81 D-8000/ GERMANY/ (089) 911061-68
 D-8000 GERMANY ROLAND F. BLOMER/ IMD DER GSF/ ARABELLSTR 4/1/ MUNICH 81 D-8000/ GERMANY/ 089/ 91 10 66
 D-8012 GERMANY BERNHARD H. BEITINGER/ INDUSTRIEANLAGEN-BETRIEBSGESELLSCHAFT/ EINSTEINSTRASSE/ OTTOBRUNN D-8012/ GERMANY
 D-8012 GERMANY HERBERT F. BISCHELTSRIEDER/ C/O INDUSTRIEANLAGEN-BETRIEBS GMBH / ABTEILUNG SZF/ OTTOBRUNN D-8012/ GERMANY
 D-8031 GERMANY RAINER R. LATKA/ AN DER GRUNDBREITE 1/ WESSLING D-8031/ GERMANY/ 089/229131 (CSID MUNICH)
 D-8520 GERMANY ATTN: REGIONALES RECHENZENTRUM/ UNIVERSITAT ERLANGEN-NURNBERG/ MARTENSSTR. 1/ ERLANGEN D-8520/ GERMANY/ 09131/85 7410
 D-8551 GERMANY REINHOLD WEICKER/ WEIHERSTR. 14/ HEMHOFEN D-8551/ GERMANY
 500762 INDIA ATTENTION: N. V. KOTESWARA RAO/ COMPUTER TRG. UNIT/ ELECTRONICS CORPORATION OF INDIA/ HYDERABAD (AP) 500762/ INDIA/ 71611
 2 IRELAND D. ABRAHAMSON/ DEPT. OF COMPUTER SCIENCE/ TRINITY COLLEGE/ 200 PEARSE ST./ DUBLIN 2/ IRELAND
 ISRAEL MICHAEL Z. HANANI/ COMPUTATION CENTER/ BEN GURIAN UNIVERSITY OF THE NEGEV/ BEER-SHEVA/ ISRAEL
 ISRAEL ATTN: THE LIBRARY/ MINISTRY OF DEFENCE/ P.O.BOX 962/ HAIFA/ ISRAEL
 ISRAEL MENACHEM SZUS/ ART AND SCIENCE/ BEZALEL ACADEMY OF ART AND DESIGN/ 10 SHMUEL HANAGID ST./ JERUSALEM/ ISRAEL/ JERUSALEM 32579
 ISRAEL RUTH WEINBERG/ COMPUTATION CENTER/ HEBREW UNIVERSITY OF JERUSALEM/ JERUSALEM/ ISRAEL/ 02-32011/280
 ISRAEL GIDEON YUVAL/ COMPUTER SCIENCE/ THE HEBREW UNIVERSITY/ JERUSALEM/ ISRAEL
 ISRAEL ATTENTION: M. MALKOSH/ DEPT OF APPLIED MATHEMATICS - GOLEM GR/ WEIZMANN INST. OF SCIENCE/ REHOVOT/ ISRAEL/ (03) 951721 X2124
 ISRAEL SAM LIBAI/ SDS COMPUTERS LTD./ P.O. BOX 29663/ TEL AVIV/ ISRAEL/ 53054
 ISRAEL IRVING N. RABINOWITZ/ DEPT. OF COMP. SCI./ TECHNION-ISRAEL INSTITUTE OF TECHNOLOG/ TECHNION CITY HAIFA/ ISRAEL

I-34100 ITALY MATTIA HMEJAK/ CENTRO DI CALCOLO/ UNIVERSITA DI TRIESTE/ VIA DEL RONCO 11/ TRIESTE I-34100/ ITALY/ 040-733033
I-40122 ITALY MAURO MONTESI/ TEMA SPA/ VIA MARCONI 29/1/ BOLOGNA I-40122/ ITALY/ 051-267285
I-40122 ITALY GUISEPPE SELVE/ TEMA S.P.A./ VIA MARCONI 29/1/ BOLOGNA I-40122/ ITALY/ 051-267285
I-56100 ITALY MARCO SOMMANI/ C/O CNUCE/ VIA SANTA MARIA 36/ PISA I-56100/ ITALY/ (050) 45245
113 JAPAN TERUO HIKITA/ DEPT. OF INFO. SCI./ U OF TOKYO/ BUNKYO-KU/ TOKYO 113/ JAPAN/ 03-812-2111 X2947
113 JAPAN EIITI WADA/ DEPARTMENT OF MATHEMATICAL ENGINEERING/ UNIVERSITY OF TOKYO/ BUNKYOKU TOKYO 113/ JAPAN/ (03) 812-2111 X7486
143 JAPAN TOSHIKI SAISHO/ 1-25-7 KITAMAGOME/ OOTA-KU TOKYO 143/ JAPAN
182 JAPAN MASATO TAKEICHI/ DEPT. OF COMPUTER SCIENCE/ THE UNIV. OF ELECTRO-COMMUNICATIONS/ 1-5-1 CHOFUGAOKA/ CHOFU-SHI TOKYO 182/ JAPAN
210 JAPAN SUSUMU YOSHIMURA/ INFORMATION SYSTEMS LAB./ 1 KOMUKAI TOSHIBA-CHO/ TOSHIBA RESEARCH AND DEVELOPMENT/ SAIWAI-KU KAWASAKI-CITY 210/ JAPAN
(044) 511 2111 X2489
222 JAPAN MASARU WATANABE/ 9-16 SHINOHARADAI/ KOHOKU-KU YOKOHAMA 222/ JAPAN/ (045) 401-9324
400 JAPAN MAKOTO ARISAWA/ COMPUTER SCIENCE DEPARTMENT/ YAMANASHI UNIVERSITY/ 4-3-11 TAKEDA KOFU/ YAMANASHI 400/ JAPAN/ (0552) 52-1111
500 JAPAN NOBUKI TOKURA/ DEPT. OF INFORMATION AND COMPUTER SCIE/ OSAKA UNIVERSITY/ 1-1 MACHIKANAYAMA/ TOKONAKA 500/ JAPAN
560 JAPAN ZENICHI KISHIMOTO/ 2-7-15 OKAMACHI-KITA/ TOYONAKA OSACA 560/ JAPAN
LIBYA CHARLES F. MURPHY/ COMPUTER SCIENCE GROUP/ UNIVERSITY OF TRIPOLI/ P.O. BOX 656/ TRIPOLI/ LIBYA
MALAYSIA PUAN SHARIFAH L. ABID/ DATA MANAGEMENT GROUP SDN.BHD/ 11-B JALAN BARAT/ PETALING JAYA SELANGOR/ MALAYSIA/ 564324/ 564353
NEW ZEALAND C. M. BISHOP/ COMPUTING CENTRE/ UNIVERSITY OF OTAGO/ P.O. BOX 56/ DUNEDIN/ NEW ZEALAND/ DUNEDIN 40109 EXT 890
NEW ZEALAND ATTN: DEPARTMENT OF INFORMATION SCIENC/ VICTORIA UNIVERSITY OF WELLINGTON/ PRIVATE BAG/ WELLINGTON/ NEW ZEALAND
N-2007 NORWAY ATTENTION: REIDAR AUNAN/ REGNEANLEGETT BLINDERN-KJELLER (RBK)/ POSTBOKS 70/ KJELLER N-2007/ NORWAY/ (02) 71 45 70
1 NORWAY IVAR LABERG/ COMPUTER DEPARTMENT/ UNIVERSITY HOSPITAL OSLO/ RIKSHOSPITALET/ OSLO 1/ NORWAY/ (471) 20 10 50
3 NORWAY TERJE NOODT/ COMPUTING CENTER/ UNIVERSITY OF OSLO/ P.B. 1059 BLINDERN/ OLO 3/ NORWAY/ (02) 466800
00901 POLAND LEON LUKASZEWICZ/ COMPUTATION CENTRE/ POLISH ACADEMY OF SCIENC E/ WARSZAWA PKIN 00901/ POLAND/ 200211 X2225
PORTUGAL LUIS M.M. DAMAS/ CENTRO DE INFORMATICA/ UNIVERSIDADE DO PORTO/ RUA DAS TAIPAS 135/ PORTO/ PORTUGAL/ 380313 OR 380769
SOUTH AFRICA K. J. MACGREGOR/ COMPUTER SCIENCE DEPARTMENT/ UNIVERSITY OF CAPE TOWN/ P.B. RONDEBOSCH/ CAPE TOWN/ SOUTH AFRICA/ 698531 X174
0001 SOUTH AFRICA ATTENTION: E. N. VAN DEVENTER/ COMPUTING CENTRE/ NATIONAL RESEARCH INSTITUTE FOR MATHEM/ P O BOX 395/ PRETORIA 0001/ SOUTH AFRICA/ 74-9111
4000 SOUTH AFRICA C. C. HANDLEY/ DEPT. OF COMPUTER SCIENCE/ UNIVERSITY OF DURBAN -WESTVILLE/ P/BAG X54001/ DURBAN 4000/ SOUTH AFRICA/ 821211 X138
4001 SOUTH AFRICA ATTN: COMPUTER CENTRE/ UNIV. OF NATAL/ KING GEORGE V AVENUE/ DURBAN 4001/ SOUTH AFRICA/ 352461
6140 SOUTH AFRICA M. H. WILLIAMS/ COMPUTER SCIENCE DEPT./ RHODES UNIVERSITY/ GRAHAMSTOWN 6140/ SOUTH AFRICA
12 SPAIN RAFAEL M. BONET/ PROVIDENCIA 137/ BARCELONA 12/ SPAIN/ 34-3-3257599
14 SPAIN MARTIN VERGES TRIAS/ AV. DR. GREGORIO MARANON S/N/ BARCELONA 14/ SPAIN/ (93) 333.29.49
S-100 44 SWEDEN STAFFEN ROMBERGER/ COMPUTER SCIENCE/ ROYAL INSTITUTE OF TECHNOLOGY/ STOCKHOLM S-100 44/ SWEDEN/ 08-787 7194
S-100 44 SWEDEN LARS-ERIK THORELLI/ DEPT. OF TELECOMMUNICATION NETWORKS & THE ROYAL INSTITUTE OF TECHNOLOGY/ STOCKHOLM 70 S-100 44/ SWEDEN/ SWEDEN-08-236520
S-126 25 SWEDEN BJARNE DACKER/ TN/X/TDGC/ L.M. ERICSSON/ STOCKHOLM S-126 25/ SWEDEN
S-145 72 SWEDEN SEVED TORSTENDAHL/ TOMTBERGAVAGEN 279/ NORSBORG S-145 72/ SWEDEN/ 08/719 00 00
S-161 54 SWEDEN CLAES RICKEBY/ HEDEBYVAGEN 5/ BROMMA S-161 54/ SWEDEN/ 08/37 65 37
S-172 04 SWEDEN KRISTER JANZON/ SECTION 041/ FOA 2/ BOX 416/ SUNDBYBERG 4 S-172 04/ SWEDEN
S-220 07 SWEDEN LENNERT BENSRYD/ COMPUTING CENTRE/ LUND UNIVERSITY/ BOX 783/ LUND S-220 07/ SWEDEN/ 046/12 46 20
S-220 07 SWEDEN KARL JOHAN ASTROM/ DEPT. OF AUTOMATIC CONTROL/ LUND INST. OF TECHNOLOGY/ P.O. BOX 725/ LUND 7 S-220 07/ SWEDEN/ 046-12 46 00-1500
S-223 62 SWEDEN STEN HENRIKSSON/ COMPUTER SCIENCE/ LUNDS UNIVERSITET/ SOLVEGATAN 14A/ LUND S-223 62/ SWEDEN/ (46) 46124620-1156
S-281 00 SWEDEN LARS SVENSSON/ MINIMIK KONSULT/ JAGAREVAGEN 15/ HASSLEHOLM S-281 00/ SWEDEN/ 0457-36475
S-341 00 SWEDEN BENGT THYLEN/ AB TRELLEBORGPLAST/ LVUNGBY S-341 00/ SWEDEN/ 0372/12520
S-402 20 SWEDEN BENGT NORDSTROM/ DEPARTMENT OF COMPUTER SCIENCES/ CHALMERS INSTITUTE OF TECHNOLOGY/ GOTEBERG S-402 20/ SWEDEN/ 031-81 01 00
S-431 20 SWEDEN LENNART OSKARSSON/ TELEFON AB L M ERICSSON/ FACK/ MOLNDAL S-431 20/ SWEDEN
S-431 39 SWEDEN KURT FREDRIKSSON/ RINGLEKEN 7/ MOLNDAL S-431 39/ SWEDEN/ 4631-41 05 14 (HOME)/ 4631-27 50 00-491 (OFFICE)
S-434 00 SWEDEN TOOMAS KAER/ OJERSBO PL 2719/ KUNGSBACKA S-434 00/ SWEDEN
S-461 01 SWEDEN LARS G. MOSSBERG/ VOLVO FLYGMOTOR AB/ BOX 136/ TROLLHATTEN S-461 01/ SWEDEN
S-581 83 SWEDEN ARNE BORTEMARK/ DEPT. OF MATHEMATICS/ LINKOPING UNIVERSITY/ FACK/ LINKOPING S-581 83/ SWEDEN
S-603 78 SWEDEN STEN LJUNGKVIS/ GUSTAF CLASONS GATA 61/ NORRKOPING S-603 78/ SWEDEN/ 011 - 10 80 00 (OFFICE)/ 011 - 17 02 10 (HOME)
S-751 21 SWEDEN ALF M. BRUNSTROM/ INSTITUTE OF TECHNOLOGY/ UPPSALA UNIVERSITY/ BOX 534/ UPPSALA S-751 21/ SWEDEN
S-751 21 SWEDEN HANS FLACK/ DEPT. COMP. TECHNOLOGY/ TEKNIKUM/ BOX 534/ UPPSALA S-751 21/ SWEDEN
S-751 21 SWEDEN LARS MAGNUSSON/ INSTITUTE OF TECHNOLOGY/ UPPSALA UNIVERSITY/ B ox 534/ UPPSALA S-751 21/ SWEDEN/ 018-10 04 70
S-752 23 SWEDEN OLLE OLSSON/ DEPT. OF COMPUTER SCIENCE:ADP/ UPPSALA UNIVERSITY / STUREGATAN 4B 1 TR/ UPPSALA S-752 23/ SWEDEN/ 018-138650
CH-1007 SWITZERLAND MICHEL JAUNIN/ CENTRE DE CALCUL/ ECOLE POLYTECHNIQUE FEDERALE/ 33 AV. DE COUR/ LAUSANNE CH-1007/ SWITZERLAND/ 021/26 46 21 INT. 401
CH-1007 SWITZERLAND CHARLES RAPIN/ CHAIRE INFORMATIQUE APPLIQUEE DMA EPFL/ 61 AVENUE DE COUR/ LAUSANNE CH-1007/ SWITZERLAND/ (021) 27 31 05
CH-1200 SWITZERLAND DAVID BATES/ 12 CHEMIN DE TAVERNAY/ 1218 GRAND SACCONEX/ GENEVA CH-1200/ SWITZERLAND/ 98-55-44
CH-1200 SWITZERLAND DANIEL THALMANN/ CENTRE UNIVERSITAIRE D'INFORMATIQUE/ 24 R. GENERAL - DUFOUR/ GENEVE CH-1200/ SWITZERLAND
CH-1205 SWITZERLAND URS R WYSS/ AVENUE DU MAIL 18/ GENEVA CH-1205/ SWITZERLAND/ 0041-22-28.79.61
CH-1207 SWITZERLAND DANG VAN BA/ SERVICE CANTONAL DE STATISTIQUE/ CASE POSTALE 306/ GENEVA 6 CH-1207/ SWITZERLAND/ (022) 361400 INT.19
CH-1211 SWITZERLAND ERNST MESSMER/ DIVISION D'INFORMATIQUE/ HOPITAL CANTONAL/ GENEVA CH-1211/ SWITZERLAND/ 0041-22-22 62 18
CH-1211 SWITZERLAND PATRICK CHEVAUX/ DIGITAL EQUIPMENT CORP./ QUAI ERNEST ANSERMET - B.P. 23/ GENEVA 8 CH-1211/ SWITZERLAND/ 022/ 20 40 20
CH-1211 SWITZERLAND HERVE TIRFORD/ MOTOROLA SEMICONDUCTORS/ 16 CHEMIN DE LA VOIE-CREUSE/ GENEVE CH-1211/ SWITZERLAND/ 33-56-07
CH-1211 SWITZERLAND ATTN: CERN LIBRARY/ PERIODICALS/ GENEVE 23 CH-1211/ SWITZERLAND
CH-1211 SWITZERLAND R. MOREL/ CENTRE DE CALCUL ELECTRONIQUE/ COLLEGE DE GENEVE/ GENEVE 3 CH-1211/ SWITZERLAND/ 27 22 28
CH-1211 SWITZERLAND MARYLENE WUEST/ CENTRE UNIVERSITAIRE D'INFORMATIQUE/ 24 RUE DU GENERAL DUFOUR/ GENEVE 4 CH-1211/ SWITZERLAND/ 022-20 93 33
CH-2000 SWITZERLAND NORBERT EBEL/ CENTRE DE CALCUL/ UNIVERSITE/ FBG. DE L'HOPITAL 33/ NEUCHATEL CH-2000/ SWITZERLAND
CH-3000 SWITZERLAND HILMAR GUTFELDT/ RESEARCH AND DEVELOPMENT-DEPT. 82/ HASLER LTD./ BELPSTRASSE 22/ BERNE 14 CH-3000/ SWITZERLAND/ 031 65 21 11

CH-8021 SWITZERLAND E. MARMIER/ ORGANISATION AND AUTOMATION/ SWISS CREDIT BANK/ ZURICH CH-8021/ SWITZERLAND
CH-8027 SWITZERLAND P. J. ERARD/ FIDES COMPUTING CENTER/ BLEICHERWEG 33/ ZURICH CH-8027/ SWITZERLAND/ (41) 12027840
CH-8035 SWITZERLAND ATTENTION: MAX SEVCIK/ COMPUTER ASSOCIATES INTL. LTD./ STAMPFENBACHSTR. 52 - P.O. BOX/ ZURICH CH-8035/ SWITZERLAND/ 01-60 42 52
CH-8092 SWITZERLAND URS AMMANN/ INSTITUT FUER INFORMATIK/ ETH - ZENTRUM/ ZUERICH CH-8092/ SWITZERLAND/ 01-32 62 11 X2214
CH-8092 SWITZERLAND CHRISTIAN JACOBI/ INSTITUT FUER INFORMATIK/ ETH - ZENTRUM/ ZUERICH CH-8092/ SWITZERLAND/ 41 1 32 62 11 X2217
CH-8092 SWITZERLAND SVEND ERIK KNUDSEN/ INSTITUT FUER INFORMATIK/ ETH - ZENTRUM/ ZUERICH CH-8092/ SWITZERLAND/ (01) 32 62 11 X2217
CH-8092 SWITZERLAND NIKLAUS WRTH/ INSTITUT FUER INFORMATIK/ ETH - ZENTRUM/ ZUERICH CH-8092/ SWITZERLAND
CH-8092 SWITZERLAND ATTN: RZ - BIBLIOTHEK/ ETH - ZENTRUM/ ZURICH CH-8092/ SWITZERLAND/ 01-32 62 11
CH-8092 SWITZERLAND HANS-HEINRICH NAGELI/ INSTITUT FUER INFORMATIK/ E.T.H. - ZENTRUM/ ZURICH CH-8092/ SWITZERLAND/ 01/32 62 11
CH-9470 SWITZERLAND HELMUT SANDMAYR/ NEU-TECHNIKUM BUCHS/ BUCHS CH-9470/ SWITZERLAND/ CH-085/6 45 24
THE NETHERLANDS ATTN: SARA-LIBRARY/ P.O. BOX 7161/ AMSTERDAM/ THE NETHERLANDS/ 020-5484911
THE NETHERLANDS W. DE VRIES/ C/O I.K.O./ POSTBOX 4395 - OOSTERRINGDIJK 18/ AMSTERDAM/ THE NETHERLANDS
THE NETHERLANDS D. GOSMAN/ ZEEMAN LABORATORIUM/ UNIVERSITEIT VAN AMSTERDAM/ PLANTAGE MUIDERGRACHT 4/ AMSTERDAM/ THE NETHERLANDS/ 020-5222177
THE NETHERLANDS A. C. W. LEYEN/ DEPARTMENT MSE/ C/O KONINKLIJKE/ SHELL-LABORATORIUM/ P.O. BOX 3003/ AMSTERDAM/ THE NETHERLANDS/ 070-203264
THE NETHERLANDS ANDREW S. TANENBAUM/ WISKUNDIG SEMINARIUM/ VRIJE UNIVERSITEIT/ DE BOELELAAN 1081/ AMSTERDAM/ THE NETHERLANDS/ 020 548 24 10
THE NETHERLANDS R. P. VAN DE RIET/ VRIJE UNIVERSITEIT/ DE BOELELAAN 1081/ AMSTERDAM/ THE NETHERLANDS/ 020-5482410
THE NETHERLANDS P. VAN EMDE BOAS/ ITW / VPW UVA/ ROETERSSTRAAT 15/ AMSTERDAM C / THE NETHERLANDS/ 020-522 3065
THE NETHERLANDS STEPHEN G. S. PROUT/ MGR. BUCKXSTRAAT 18/ BORN (L)/ THE NETHERLANDS/ 04498-2481
THE NETHERLANDS G. E. VAN BEINUM/ TNO-IBBC/ POSTBOX 49/ DELFT/ THE NETHERLANDS / 015-138222 EXT 250
THE NETHERLANDS ATTN: BIBLIOTHEEK 05627/ TECHNISCHE HOGESCHOOL/ POSTBUS 513/ EINDHOVEN/ THE NETHERLANDS
THE NETHERLANDS LEO C. NOORDHUIZEN/ GLOEILAMPEN - FABRIEKEN/ BUILDING VN-521/ N.V. PHILLIPS/ EINDHOVEN/ THE NETHERLANDS/ 040-783634
THE NETHERLANDS J. J. VAN AMSTEL/ COMPUTING CENTRE/ EINDHOVEN UNIVERSITY OF TECHNOLOGY/ P.O. BOX 513/ EINDHOVEN/ THE NETHERLANDS/ (040) 474547
THE NETHERLANDS C. BRON/ DEPT. OF ELECTRICAL ENGINEERING/ TECHNISCHE HOGESCHOOL TWENTE/ POSTBUS 217/ ENSCHEDE/ THE NETHERLANDS/ (031) 53 894451
THE NETHERLANDS S. D. SWIERSTRA/ TECHNISCHE HOGESCHOOL TWENTE/ P.O. BOX 217/ ENSCHEDE/ THE NETHERLANDS/ 31-53-894441
THE NETHERLANDS ATTN: DSM/ CENTRAL LIBRARY/ P.O. BOX 18/ GELEEN/ THE NETHERLANDS
THE NETHERLANDS LOU H. KRAMER/ CALLUNALAAN 8/ GOUDA/ THE NETHERLANDS/ 070 - 264221
THE NETHERLANDS D. D. DE VRIES/ LANDLEVEN 1/ REKENCENTRUM R.U.G./ P.O. BOX 800 / GRONINGEN/ THE NETHERLANDS
THE NETHERLANDS TOM VAN DER HOEVEN/ HAGEDOORNSWEG/ NIEBERT/ THE NETHERLANDS
THE NETHERLANDS P. F. KLOK/ COMPUTER GRAPHICS/ KATHOLIEKE UNIVERSITEIT/ TOERNOOIVELD/ NIJMEGEN/ THE NETHERLANDS/ 080-558833 X3201
THE NETHERLANDS L. S. C. STATEMA/ UNIVERSITY COMPUTING CENTRE/ TOURNOOIVELD 1/ NIJMEGEN/ THE NETHERLANDS/ 080-558833 X2590
THE NETHERLANDS ATTN: INSTITUTE TNO FOR MATHEMATICS/ COMPUTER CENTRE/ INFORMATION PROCESSING AND STATISTICS/ KONINGIN MARIALAAN 21/ THE HAGUE/ THE NETHERLANDS
070-824161
THE NETHERLANDS D. SANDEE/ PHYSICS LABORATORY TNO/ P.O. BOX 2864/ THE HAGUE/ THE NETHERLANDS/ (070) 264221
THE NETHERLANDS P. A. SLATS/ INFORMATION PROCESSING AND STATISTICS/ INST. TNO FOR MATHEMATICS/ KON. MARIALAAN 21/ THE HAGUE/ THE NETHERLANDS
THE NETHERLANDS P. J. VAN DER HOFF/ PIJPERSTRAAT 5/ BERKEL EN RODENRIJS/ THE NETHERLANDS
THE NETHERLANDS H. VAN LOON/ ACADEMISCH COMPUTER CENTRUM UTRECHT/ BUDAPESTLAAN 6/ DE UITHOF UTRECHT/ THE NETHERLANDS/ 030-531436
1005 THE NETHERLANDS ATTN: LIBRARY/ MATHEMATISCH CENTRUM/ 2E BOERHAAVESTRAAT 49/ AMSTERDAM 1005/ THE NETHERLANDS
2005 THE NETHERLANDS ATTN: BOEKHANDEL VERWIJS EN STAM B.V./ PRINSESSEGRACHT 2/ 'S-GRAVENHAGE 2005/ THE NETHERLANDS
2076 THE NETHERLANDS N. D. BREWER/ MATHEMATICS AND COMPUTER DIV./ SHAPE TECHNICAL CENTRE/ P.O. BOX 174/ THE HAGUE 2076/ THE NETHERLANDS/ 070-24.55.50
2231 XE THE NETHERLANDS JEAN-PIERRE BOUCHEZ/ ELZENLAAN 6/ RIJNSBURG 2231 XE/ THE NETHERLANDS
2506 THE NETHERLANDS J. A. ALANEN/ VAKGROEP INFORMATICA R.U./ BUDAPESTLAAN 6/ UTRECHT 2506/ THE NETHERLANDS
9321 THE NETHERLANDS T. J. VAN WEERT/ ELZENLAAN 28/ PEIZE GN 9321/ THE NETHERLANDS
UNITED KINGDOM C. B. KING/ PHILLIPS RESEARCH LABORATORIES/ CROSS OAK LANE / REDHILL/ SURREY ENGLAND/ UNITED KINGDOM/ HORLEY 6377
UNITED KINGDOM ATTN: THE DOCUMENTATION OFFICER/ COMPUTING LABORATORY/ UNIVERSITY OF KENT/ CANTERBURY KENT/ UNITED KINGDOM
UNITED KINGDOM STEPHEN L. BREIBART/ EASTCOTE/ 12 ELM AVENUE/ PINNER MIDDLESEX / UNITED KINGDOM
UNITED KINGDOM MAURICE O'FLAHERTY/ ANTRIM/ 444 MEVILLE GARDEN VILLAGE/ NEWTOWNABBAY N. IRELAND/ UNITED KINGDOM
UNITED KINGDOM ROBERT G. CLARK/ DEPT. OF COMPUTING SCIENCE/ UNIVERSITY OF STIRLING/ STIRLING SCOTLAND/ UNITED KINGDOM
UNITED KINGDOM N. J. FIDDIAN/ DEPT. OF COMPUTING MATHEMATICS/ UNIVERSITY COLLEGE CARDIFF/ CARDIFF WALES/ UNITED KINGDOM/ 44211 CARDIFF X2669
AB9 2UB UNITED KINGDOM DENIS M. WILSON/ DEPARTMENT OF COMPUTING SCIENCE/ UNIVERSITY OF ABERDEEN/ KING-S COLLEGE/ OLD ABERDEEN SCOTLAND AB9 2UB/ UNITED KINGDOM
AL1 1NF UNITED KINGDOM J M JENKIN/ 23 HART ROAD/ ST ALBANS HERTS. AL1 1NF/ UNITED KINGDOM/ 68026
AL1 1RZ UNITED KINGDOM M. I. JACKSON/ 165 RIVERSIDE ROAD/ ST ALBANS HERTS. AL1 1RZ/ UNITED KINGDOM/ HATFIELD 68100 X252
AL10 9AB UNITED KINGDOM BOB DICKERSON/ COMPUTER SYSTEMS GROUP/ THE HATFIELD POLYTECHNIC/ PO BOX 109 COLLEGE LANE/ HATFIELD HERTS AL10 9AB/ UNITED KINGDOM/ HATFIELD 68100
AL10 9AB UNITED KINGDOM JOHN W. LEWIS/ SCHOOL OF INFORMATION SCIENCES/ HATFIELD POLYTECHNIC/ P.O. BOX 109/ HATFIELD HERTS AL10 9AB/ UNITED KINGDOM/ 68100 X237
BH22 8HL UNITED KINGDOM DAVID SPENCER/ 29 DORSET AVE./ FERNDOWN DORSET BH22 8HL/ UNITED KINGDOM/ 0202 875571
BN1 9QT UNITED KINGDOM R. L. GRIMSDALE/ SCHOOL OF APPLIED SCIENCES/ UNIVERSITY OF SUSSEX/ FALMER/ BRIGHTON ENGLAND BN1 9QT/ UNITED KINGDOM/ (0273) 66755
BN2 4GJ UNITED KINGDOM ATTENTION: B.S. MOSSAKOWSKI/ DEPT. OF COMPUTING AND CYBERNETICS/ BRIGHTON POLYTECHNIC/ MOULSECOOMB/ BRIGHTON ENGLAND BN2 4GJ/ UNITED KINGDOM
BN2 6RD UNITED KINGDOM D. A. JOSLIN/ WOODINGDEAN/ 40 BATEMANS ROAD/ BRIGHTON SUSSEX BN2 6RD/ UNITED KINGDOM/ BRIGHTON 37772
BN3 1RA UNITED KINGDOM B. WILLIAM/ 67 DAVIGDOR ROAD/ HOVE SUSSEX BN3 1RA/ UNITED KINGDOM
BS9 4PL UNITED KINGDOM ALAN BLANNIN/ WESTBURY-ON-TRYM/ 28 HARBURY ROAD/ BRISTOL ENGLAND BS9 4PL/ UNITED KINGDOM/ (0272) 624808
BT37 0QB UNITED KINGDOM C. J. COPELAND/ SCHOOL OF COMPUTER SCIENCE/ ULSTER COLLEGE/ JORDANSTOWN/ NEWTOWNABBAY N.IRELAND BT37 0QB/ UNITED KINGDOM/ 0231-65131 X2131
BT7 1NN UNITED KINGDOM JIM WELSH/ DEPARTMENT OF COMPUTER SCIENCE/ QUEEN'S UNIVERSITY/ BELFAST N.IRELAND BT7 1NN/ UNITED KINGDOM
BT9 5EQ UNITED KINGDOM ATTN: SCIENCE LIBRARY/ QUEEN'S UNIVERSITY/ BELFAST N. IRELAND BT9 5EQ/ UNITED KINGDOM
B15 2TT UNITED KINGDOM ALAN REED/ COMPUTER CENTRE/ UNIVERSITY OF BIRMINGHAM/ BIRMINGHAM ENGLAND B15 2TT/ UNITED KINGDOM
CB2 1RP UNITED KINGDOM C. A. LANG/ PITT BUILDING/ CAMBRIDGE UNIVERSITY PRESS/ TRUMPINGTON ST./ CAMBRIDGE ENGLAND CB2 1RP/ UNITED KINGDOM/ 0223-53301
CV4 7AL UNITED KINGDOM ATTN: COMPUTER UNIT/ COMPUTER CENTER/ UNIVERSITY OF WARWICK/ COVENTRY ENGLAND CV4 7AL/ UNITED KINGDOM/ (0203) 24011 X2754
DE3 6RU UNITED KINGDOM G. OAKES/ 45 HAMILTON ROAD/ DERBY ENGLAND DE3 6RU/ UNITED KINGDOM

DH1 3LE UNITED KINGDOM H. F. TIBBALS/ COMPUTER UNIT/ SCIENCE LABORATORIES/ DURHAM UNIV./ DURHAM ENGLAND DH1 3LE/ UNITED KINGDOM/ DURHAM 64971
EC3V 1LP UNITED KINGDOM PHILIP J. MALCOLM/ C/O BANK OF ADELAIDE/ 11 LEADENHALL ST./ LONDON ENGLAND EC3V 1LP/ UNITED KINGDOM/ 01-323 0637/0
EH1 2HW UNITED KINGDOM A. BALFOUR/ COMPUTER CENTRE/ HERIOT-WATT UNIVERSITY/ 37-39 GRASSMARKET/ EDINBURGH SCOTLAND EH1 2HW/ UNITED KINGDOM
E1 4NS UNITED KINGDOM JOHN HUTCHINSON/ COMPUTER CENTRE/ QUEEN MARY COLLEGE/ MILE END ROAD/ LONDON ENGLAND E1 4NS/ UNITED KINGDOM/ 01-980-4811 X778
E14 UNITED KINGDOM ISAMU HASEGAWA/ 7 STAINSBURY ROAD/ LONDON ENGLAND E14/ UNITED KINGDOM
GU16 5HJ UNITED KINGDOM ANTHONY LESLIE GOLBORN/ SYSTEMS DESIGNERS LIMITED/ SYSTEMS HOUSE/ 57-61 HIGH STREET/ FRIMLEY SURREY GU16 5HJ/ UNITED KINGDOM
GU16 5HJ UNITED KINGDOM VIC STENNING/ SYSTEMS DESIGNERS LTD./ 57-61 HIGH STREET/ FRIMLEY SURREY GU16 5HJ/ UNITED KINGDOM
G12 8QQ UNITED KINGDOM BILL FINDLAY/ COMPUTING SCIENCE DEPARTMENT/ UNIVERSITY OF GLASGOW/ GLASGOW SCOTLAND G12 8QQ/ UNITED KINGDOM/ 339 8855 X7391
G12 8QQ UNITED KINGDOM D. G. JENKINS/ COMPUTING SCIENCE DEPT./ THE UNIVERSITY/ GLASGOW SCOTLAND G12 8QQ/ UNITED KINGDOM/ (041) 339-8855 X478/7458
G12 8QQ UNITED KINGDOM DAVID WATT/ COMPUTING SCIENCE DEPT./ UNIVERSITY OF GLASGOW/ GLASGOW SCOTLAND G12 8QQ/ UNITED KINGDOM/ 041-339 8855 X7458
HA4 9DP UNITED KINGDOM ROBERT KIRKBY/ RUISLIP MANOR/ 44 WHITBY ROAD/ MIDDLESEX ENGLAND HA4 9DP/ UNITED KINGDOM
HA6 3DZ UNITED KINGDOM N ROBINSON/ 1 THE FAIRWAY/ NORTHWOOD MIDDLESEX/ LONDON ENGLAND HA6 3DZ/ UNITED KINGDOM
HP2 5HG UNITED KINGDOM C. B. A. PRICE/ CBAP SERVICES/ 67 FIGTREE HILL / HEMEL/ HEMPSTEAD HERTS HP2 5HG/ UNITED KINGDOM/ 0442 57340
HRI 1TY UNITED KINGDOM A. J. FISHER/ 2 ELGAR AVENUE/ HEREFORD ENGLAND HRI 1TY/ UNITED KINGDOM
HU6 7RX UNITED KINGDOM B. J. CORNELIUS/ DEPT. OF COMP. STUDIES/ UNIVERSITY OF HULL/ HULL ENGLAND HU6 7RX/ UNITED KINGDOM/ (0482) 497951
KT12 5NF UNITED KINGDOM DAN C.C. HAMM/ HERSHAM/ 85 QUEENS ROAD/ WALTON-ON-THA SURREY KT12 5NF/ UNITED KINGDOM/ WALTON-ON-THAMES 43639
KY16 UNITED KINGDOM B. T. MITCHELL/ COMPUTING LABORATORY/ UNIVERSITY OF ST. ANDREWS/ NORTH HAUGH ST. ANDREWS/ FIFE SCOTLAND KY16/ UNITED KINGDOM
LA1 4YB UNITED KINGDOM D. R. ALLUM/ DEPT. OF PHYSICS/ UNIVERSITY OF LANCASTER/ LANCASTER ENGLAND LA1 4YB/ UNITED KINGDOM/ LANCASTER 65201 X4178
LA1 4YX UNITED KINGDOM ATTN: THE LIBRARIAN/ DEPT. OF COMPUTER STUDIES/ U OF LANCASTER / BAILRIGG/ LANCASTER ENGLAND LA1 4YX/ UNITED KINGDOM/ (0524) 65201 X4133
LA1 4YX UNITED KINGDOM ANN V. BARROW/ DEPT. OF COMPUTER STUDIES/ UNIVERSITY OF LANCASTER/ BAILRIGG/ LANCASTER ENGLAND LA1 4YX/ UNITED KINGDOM/ (0524) 65201
LA1 4YX UNITED KINGDOM BOB E. BERRY/ DEPT. OF COMPUTER STUDIES/ UNIVERSITY OF LANCASTER/ BAILRIGG/ LANCASTER ENGLAND LA1 4YX/ UNITED KINGDOM/ (0524) 65201
LA1 4YX UNITED KINGDOM MIKE W. CORNELIUS/ DEPT. OF COMPUTER STUDIES/ UNIVERSITY OF LANCASTER/ BAILRIGG/ LANCASTER ENGLAND LA1 4YX/ UNITED KINGDOM/ (0524) 65201 X4120
LA1 4YX UNITED KINGDOM ARTHUR FOSTER/ DEPT. OF COMPUTER STUDIES/ UNIVERSITY OF LANCASTER/ BAILRIGG/ LANCASTER ENGLAND LA1 4YX/ UNITED KINGDOM/ (0524) 65201 X4123
LA1 4YX UNITED KINGDOM BRIAN A. E. MEEKINGS/ DEPT. OF COMPUTER STUDIES/ UNIVERSITY OF LANCASTER/ BAILRIGG/ LANCASTER ENGLAND LA1 4YX/ UNITED KINGDOM/ (0524) 65201
LA1 4YX UNITED KINGDOM CHRIS D. PAICE/ DEPT. OF COMPUTER STUDIES/ UNIVERSITY OF LANCASTER/ BAILRIGG/ LANCASTER ENGLAND LA1 4YX/ UNITED KINGDOM/ (0524) 65201
LA1 4YX UNITED KINGDOM HIKMET SAKA/ DEPT. OF COMPUTER STUDIES/ UNIVERSITY OF LANCASTER/ BAILRIGG/ LANCASTER ENGLAND LA1 4YX/ UNITED KINGDOM/ (0524) 65201 X4120
LA1 4YX UNITED KINGDOM S. P. J. WAGSTAFF/ DEPT. OF COMPUTER STUDIES/ UNIVERSITY OF LANCASTER/ BAILRIGG/ LANCASTER ENGLAND LA1 4YX/ UNITED KINGDOM/ (0524) 65201
LA1 4YX UNITED KINGDOM CHI YIP/ DEPT. OF COMPUTER STUDIES/ UNIVERSITY OF LANCASTER/ BAILRIGG/ LANCASTER ENGLAND LA1 4YX/ UNITED KINGDOM/ (0524) 65201
LE1 7RH UNITED KINGDOM H. J. ROWE/ COMPUTER LABORATORY/ LEICESTER UNIVERSITY/ LEICESTER ENGLAND LE1 7RH/ UNITED KINGDOM
LS2 9JT UNITED KINGDOM TONY MITCHELL/ DEPT. OF COMPUTER STUDIES/ LEEDS UNIV./ LEEDS ENGLAND LS2 9JT/ UNITED KINGDOM
MK40 2PN UNITED KINGDOM B. HALE/ THE MERTON CENTRE/ PRIME COMPUTER INC./ ST. PETERS STREET/ BEDFORD ENGLAND MK40 2PN/ UNITED KINGDOM/ 0234-65121
M13 9PL UNITED KINGDOM A. M. ADDYMAN/ DEPARTMENT OF COMPUTER SCIENCE/ THE UNIVERSITY/ OXFORD ROAD/ MANCHESTER ENGLAND M13 9PL/ UNITED KINGDOM/ 061-273 5466 X6
M13 9PL UNITED KINGDOM RIC COLLINS/ REGIONAL COMPUTER CENTRE/ UNIVERSITY OF MANCHESTER/ OXFORD ROAD/ MANCHESTER ENGLAND M13 9PL/ UNITED KINGDOM/ 061-273-8252
M13 9PL UNITED KINGDOM M. A. PELL/ DEPT. OF COMMUNITY MEDICINE/ UNIVERSITY OF MANCHESTER/ OXFORD ROAD/ MANCHESTER ENGLAND M13 9PL/ UNITED KINGDOM/ 061-273 8241 X0-X197
M20 9QL UNITED KINGDOM GRAHAM J. WHITE/ 8 KINNAIRD ROAD/ MANCHESTER ENGLAND M20 9QL/ UNITED KINGDOM
M5 4WT UNITED KINGDOM ATTN: DIRECTOR/ COMPUTING LABORATORY/ UNIVERSITY OF SALFORD/ SALFORD ENGLAND M5 4WT/ UNITED KINGDOM/ 061 - 736 5843 X307
M60 1QD UNITED KINGDOM ATTN: THE LIBRARIAN/ DEPT. OF COMPUTATION/ UMIST/ P.O. BOX 88/ MANCHESTER ENGLAND M60 1QD/ UNITED KINGDOM/ 061-2363311 X2178
M60 1QD UNITED KINGDOM DEREK COLEMAN/ DEPT. OF COMPUTATION/ UMIST/ P.O. BOX 88/ MANCHESTER ENGLAND M60 1QD/ UNITED KINGDOM
NPT 1XG UNITED KINGDOM GEOFF V KING/ BUSINESS STATISTICS OFFICE/ CARDIFF ROAD/ NEWPORT GWENT NPT 1XG/ UNITED KINGDOM/ 0633 56111
NR4 7TJ UNITED KINGDOM S. M. JOHNSON/ SCHOOL OF MATHS AND PHYSICS/ UNIV. OF EAST ANGLIA/ UNIVERSITY PLAIN/ NORWICH ENGLAND NR4 7TJ/ UNITED KINGDOM
NW3 UNITED KINGDOM H. J. ZELL/ 14 KEMPLAY ROAD/ LONDON ENGLAND NW3/ UNITED KINGDOM
NW3 7ST UNITED KINGDOM J. B. SLATER/ COMPUTER UNIT/ WESTFIELD COLLEGE/ KIDDERPORE AVENUE/ LONDON ENGLAND NW3 7ST/ UNITED KINGDOM/ 01-435-7141 X520
N10 UNITED KINGDOM W. H. L. WILLIAMS/ 252 COLNCEY HATCH LANE/ LONDON ENGLAND N10/ UNITED KINGDOM/ 01-405-8400
N8 UNITED KINGDOM JOHN REYNOLDS/ 31 BARRINGTON ROAD/ LONDON ENGLAND N8/ UNITED KINGDOM/ 01-340-2413
OX11 0QX UNITED KINGDOM ATTN: LIBRARIAN/ ATLAS COMPUTING DIV./ RUTHERFORD LABORATORY/ CHILTON DIDCOT/ OXON ENGLAND OX11 0QX/ UNITED KINGDOM/ ABINGDON 21900 X6226
OX11 0QX UNITED KINGDOM CHRISTOPHER S COOPER/ C & A DIVISION/ RUTHERFORD LABORATORY/ CHILTON DIDCOT/ OXON ENGLAND OX11 0QX/ UNITED KINGDOM/ ABINGDON(0235) 21900 X6211
PE17 3QB UNITED KINGDOM A. R. M. WAJH/ EARLTH/ 15 SCHOOL RD/ HUNTINGDON ENGLAND PE17 3QB/ UNITED KINGDOM
RG1 7QN UNITED KINGDOM IAIN SMITH/ EUROPEAN SOFTWARE ENGINEERING/ FOUNTAIN HOUSE/ DIGITAL EQUIPMENT CORP. LTD./ BUTTS CENTRE/ READING ENGLAND RG1 7QN/ UNITED KINGDOM
(0734) 583555
RG6 2LH UNITED KINGDOM ROGER P. WRIGHT/ EARLEY/ 16 RAGGLESWOOD CLOSE/ READING BERKS. RG6 2LH/ UNITED KINGDOM/ READING 663178
SA2 8PP UNITED KINGDOM B. NIBLETT/ DEPT. OF COMPUTER SCIENCE/ UNIVERSITY COLLEGE OF SWANSEA/ SWANSEA ENGLAND SA2 8PP/ UNITED KINGDOM
SE1 0AA UNITED KINGDOM S. T. DEVEREUX/ COMPUTER CENTRE/ POLYTECHNIC OF THE SOUTH BANK/ BOROUGH ROAD / SOUTHWARK/ LONDON ENGLAND SE1 0AA/ UNITED KINGDOM/ 01-928 8989 X2327
S09 5NH UNITED KINGDOM ATTN: DEPT. OF MATHEMATICS C/O D.W. BA/ THE UNIVERSITY/ SOUTHAMPTON ENGLAND S09 5NH/ UNITED KINGDOM/ 0703 559122 X700
S09 5NH UNITED KINGDOM D. W. BARRON/ COMPUTER STUDIES GROUP/ THE UNIVERSITY/ SOUTHAMPTON ENGLAND S09 5NH/ UNITED KINGDOM/ 0703-559122 X700
S09 5NH UNITED KINGDOM J. GOODSON/ DEPARTMENT OF MATHEMATICS/ THE UNIVERSITY/ SOUTHAMPTON ENGLAND S09 5NH/ UNITED KINGDOM/ 0703-559122 X2387
S09 5NH UNITED KINGDOM JUDY MULLINS/ DEPARTMENT OF MATHEMATICS/ THE UNIVERSITY OF SOUTHAMPTON/ SOUTHAMPTON ENGLAND S09 5NH/ UNITED KINGDOM/ 0703 559122 X2387
S09 5NH UNITED KINGDOM MIKE J. REES/ DEPT. OF MATHS./ COMPUTER STUDIES GROUP/ THE UNIVERSITY/ SOUTHAMPTON ENGLAND S09 5NH/ UNITED KINGDOM
S09 5NH UNITED KINGDOM MORLEY W. SAGE/ COMPUTING SERVICE/ UNIVERSITY OF SOUTHAMPTON/ SOUTHAMPTON ENGLAND S09 5NH/ UNITED KINGDOM/ 0703-559122 X694
ST5 5BG UNITED KINGDOM K. H. BENNETT/ DEPT. OF COMPUTER SCIENCE/ UNIV. OF KEELE/ KEEL E STAFFORDSH ST5 5BG/ UNITED KINGDOM/ STOKE-ON-TRENT 621111 X410
SW11 UNITED KINGDOM DENIS LENIHAN/ BATTERSEA LABORATORY/ BRITISH STEEL CORPORATION / 140 BATTERSEA PARK ROAD/ LONDON ENGLAND SW11/ UNITED KINGDOM/ 01-622-5511 X6
SW7 2AZ UNITED KINGDOM P W R CLARKE/ CCD/ NEW HUXLEY BLDG/ IMPERIAL COLLEGE/ 180 QUEENSGATE/ LONDON ENGLAND SW7 2AZ/ UNITED KINGDOM/ 01-589-5111 X2758
SW7 2AZ UNITED KINGDOM R. A. FRANCIS/ CCD HUXLEY BUILDING/ IMPERIAL COLLEGE LONDON/ LONDON ENGLAND SW7 2AZ/ UNITED KINGDOM
SW7 2AZ UNITED KINGDOM JEFF KRAMER/ DEPARTMENT OF COMPUTING AND CONTROL/ HUXLEY BUILDING/ IMPERIAL COLLEGE/ 180 QUEENSGATE/ LONDON ENGLAND SW7 2AZ/ UNITED KINGDOM
01-589-5111 X2754
SW7 2AZ UNITED KINGDOM STUART JAMES MCRAE/ DEPT OF COMPUTING & CONTROL/ IMPERIAL COLLEGE/ 180 QUEENSGATE/ LONDON ENGLAND SW7 2AZ/ UNITED KINGDOM/ 01-589-5111 X2706
SW7 2AZ UNITED KINGDOM GREG PUGH/ DEPARTMENT OF COMPUTING AND CONTROL/ IMPERIAL COLLEGE/ 180 QUEENSGATE/ LONDON ENGLAND SW7 2AZ/ UNITED KINGDOM/ 01-589-5111 X2758

SW7 2AZ UNITED KINGDOM DAVID SLATER/ CCD/ IMPERIAL COLLEGE/ 180 QUEENSGATE/ LONDON ENGLAND SW7 2AZ/ UNITED KINGDOM
 SW7 2AZ UNITED KINGDOM IAIN STINSON/ DEPT. OF COMPUTING & CONTROL/ IMPERIAL COLLEGE/ 180 QUEENSGATE/ LONDON ENGLAND SW7 2AZ/ UNITED KINGDOM/ 01-589-5111 X2700
 SW7 2AZ UNITED KINGDOM DAVE THOMAS/ DEPT. OF COMPUTING & CONTROL/ IMPERIAL COLLEGE/ LONDON ENGLAND SW7 2AZ/ UNITED KINGDOM
 S10 2TN UNITED KINGDOM CHRIS MARTIN/ COMPUTING SERVICES/ THE HICKS BUILDING/ UNIVERSITY OF SHEFFIELD/ SHEFFIELD ENGLAND S10 2TN/ UNITED KINGDOM/ (0742) 78555 X263
 TS1 3BA UNITED KINGDOM ATTN: THE LIBRARY/ TEESIDE POLYTECHNIC/ BOROUGH ROAD - MIDDLESBROUGH/ CLEVELAND ENGLAND TS1 3BA/ UNITED KINGDOM/ 0642-44176
 TW11 OLW UNITED KINGDOM I. GOODE/ NATIONAL PHYSICAL LABORATORY/ DNAC/ TEDDINGTON MIDDLESEX TW11 OLW/ UNITED KINGDOM/ 01-977 3222
 TW20 OEX UNITED KINGDOM ROY EDWARDS/ DEPT. OF STAT. AND COMP. SCI./ HOLLOWAY COLLEGE/ EGHAM HILL/ EGHAM SURREY TW20 OEX/ UNITED KINGDOM/ EGHAM 4455
 WCIE 7HX UNITED KINGDOM J. J. FLORENTIN/ DEPARTMENT OF COMPUTER SCIENCE/ BIRKBECK COLLEGE/ MALET STREET/ LONDON ENGLAND WCIE 7HX/ UNITED KINGDOM
 WCI1 UNITED KINGDOM CHRIS LAZOU/ COMPUTER CENTRE/ UNIVERSITY OF LONDON/ 20 GUILFORD STREET/ LONDON ENGLAND WCI1/ UNITED KINGDOM/ 01-405-8400
 WCI1 UNITED KINGDOM SILVIA SUSSMAN/ COMPUTER CENTRE/ UNIVERSITY OF LONDON/ 20 GUILFORD ST./ LONDON ENGLAND WCI1/ UNITED KINGDOM/ (01) 405 8400
 WC1H OAH UNITED KINGDOM ANTHONY B. WELLER/ COMPUTER CENTRE/ UNIVERSITY COLLEGE LONDON/ 19 GORDON STREET/ LONDON ENGLAND WC1H OAH/ UNITED KINGDOM
 WIA 4SE UNITED KINGDOM ATTN: LIBRARIAN/ PO BOX 4SE/ LOGICA LIMITED/ 64 NEWMAN STREET/ LONDON ENGLAND WIA 4SE/ UNITED KINGDOM/ (01) 580 8361
 W8 7AH UNITED KINGDOM BRIAN MEEK/ COMPUTER UNIT/ QUEEN ELIZABETH COLLEGE/ CAMPDEN HILL ROAD/ LONDON ENGLAND W8 7AH/ UNITED KINGDOM/ 01-937 5411
 YO1 5DD UNITED KINGDOM D. G. BURNETT-HALL/ DEPARTMENT OF COMPUTER SCIENCE/ UNIVERSITY OF YORK/ HESLINGTON/ YORK ENGLAND YO1 5DD/ UNITED KINGDOM/ (0904) 59861
 630090 USSR S. POKROVSKY/ COMPUTING CENTRE/ USSR ACADEMY OF SCIENCES/ NOVO SIBIRSK 630090/ USSR
 41000 YUGOSLAVIA STJEPAN JARNJAK/ 13 PROLET. BRIG. 247/ ZAGREB 41000/ YUGOSLAVIA / (041) 513-822/767 (OFFICE)
 61 000 YUGOSLAVIA ROBERT REINHARDT/ FABIANIJEVA 39/ LJUBLJANA 61 000/ YUGOSLAVIA
 71000 YUGOSLAVIA SUAD ALAGIC/ ELEKTROTEHNIKI FAKULTET/ SARAJEVO LUKAVICA 71000 / YUGOSLAVIA

S. KAMAL ABDALI	12181		ATTENTION: NANCY BROOKS	93105		ATTN: DSM	THE NETHERLANDS
PUAN SHARIFAH L. ABID		MALAYSIA	ATTENTION: N. V. KOTESWARA RAO	500762	INDIA	ATTN: FRIEDA S. COHEN	53706
D. ABRAHAMSON	2	IRELAND	ATTENTION: PAUL C. SMITH	55455		ATTN: INSTITUT FUER INFORMATIK	D-2000 GERMANY
JOHN W. ADAMS	18015		ATTENTION: REDAIR AUNAN	N-2007	NORWAY	ATTN: INSTITUT FUR MED. DATENVERARBEITUNG	D-8000 GERMANY
J. MACK ADAMS	88003		ATTENTION: ROBERT E. NOVAK	55165		ATTN: INSTITUTE TNO FOR MATHEMATICS	THE NETHERLANDS
KENNETH LEROY ADAMS	47906		ATTENTION: ROY MAXION-PROGRAMMING ADVISOR	89507		ATTN: INST. FUR ANGEWANDTE MATHEMATIK	D-7500 GERMANY
A. M. ADDYMAN	M13 9PL	UNITED KINGDOM	ATTENTION: RUTH DROZIN	17837		ATTN: J. F. MCINTYRE - LIBRARIAN	22903
SUAD ALAGIC	71000	YUGOSLAVIA	ATTENTION: R. D. BERGERON	03824		ATTN: KAPPA ETA KAPPA	55414
J. A. ALANEN	2506	THE NETHERLANDS	ATTENTION: STEVE REISMAN	55455		ATTN: KARIN & MICHELE - PASCAL DISTRIBUTION	80302
JOHN J. ALLAN III	75275		ATTENTION: WILLIAM HUNTEMAN	97331		ATTN: LAL CHAN DAN ENTERPRISES	90260
DENNIS R. ALLISON	94025		ATTN: ACADEMIC SERVICES	90007		ATTN: LIBRARIAN	32611
D. R. ALLUM	LAI 4YB	UNITED KINGDOM	ATTN: AIR FORCE WEAPONS LABORATORY	87117		ATTN: LIBRARIAN	WIA 4SE UNITED KINGDOM
JOHN ALSTRUP	55424		ATTN: BIBLIOTEKET	DK-2800	DENMARK	ATTN: LIBRARIAN	3052 AUSTRALIA
RICH ALTMALER	94086		ATTN: BIBLIOTHEEK 05627		THE NETHERLANDS	ATTN: LIBRARIAN	OX11 OXQ UNITED KINGDOM
URS AMMANN	CH-8092	SWITZERLAND	ATTN: BIBLIOTHEK	D-6750	GERMANY	ATTN: LIBRARIAN	2033 AUSTRALIA
DAVID B. ANDERSON	18015		ATTN: BOEING COMPANY	98124		ATTN: LIBRARY	80225
GARY S. ANDERSON	98043		ATTN: BOEKHANDEL VERWIJS EN STAM B.V.	2005	THE NETHERLANDS	ATTN: LIBRARY	01754
JACK ANDERSON	55431		ATTN: B1700 PROTEUS PROJECT	84112		ATTN: LIBRARY	T6G 2J8 CANADA
PETER ANDERSON	07102		ATTN: CENTRAL LIBRARY	3001	AUSTRALIA	ATTN: LIBRARY	1005 THE NETHERLANDS
RICHARD V. ANDREE	73019		ATTN: CENTRO DE CIENCIAS DE LA COMPUTACION		CHILE	ATTN: LIBRARY	91109
DENNIS S. ANDREWS	94086		ATTN: CERN LIBRARY	CH-1211	SWITZERLAND	ATTN: LIBRARY / SERIALS	94305
MAKOTO ARISAWA	400	JAPAN	ATTN: CHIEF BRANCH OF DATA SYSTEM SERVICES	80225		ATTN: MATH LIBRARY	02115
KARL JOHAN ASTROM	S-220 07	SWEDEN	ATTN: COMPUTER CENTER USER SERVICES	98105		ATTN: MATHEMATICS DEPARTMENT	19085
ATTENTION: ARMENELLA VINSON	87114		ATTN: COMPUTER CENTRE	3083	AUSTRALIA	ATTN: MOD COMP LIBRARY	33309
ATTENTION: A.S. WILLIAMS	92626		ATTN: COMPUTER CENTRE	4001	SOUTH AFRICA	ATTN: M. DOHERTY	M5S 1A7 CANADA
ATTENTION: BLAIR BURNER	98124		ATTN: COMPUTER SCIENCE DEPARTMENT	59801		ATTN: M. WATKINS - TECHNICAL LIBRARIAN	21031
ATTENTION: B.S. MOSSAKOWSKI	BN2 4GJ	UNITED KINGDOM	ATTN: COMPUTER SCIENCE DEPT.	55455		ATTN: PRODUCTION AUTOMATION PROJECT	14627
ATTENTION: CHRIS BRYCE	L85 4K1	CANADA	ATTN: COMPUTER SCIENCE DEPT. A	93940		ATTN: PROGRAM LIBRARIAN	5001 AUSTRALIA
ATTENTION: COLIN G. CAMPBELL	77001		ATTN: COMPUTER SCIENCE DEPT. B	93940		ATTN: PROGRAM LIBRARY	N6A 5B7 CANADA
ATTENTION: DAN BURROWS	55812		ATTN: COMPUTER SCIENCES INSTITUTE	92507		ATTN: PURCHASING OFFICE	2600 AUSTRALIA
ATTENTION: DAVID MADISON	35806		ATTN: COMPUTER SERVICES - FO1.3	75080		ATTN: READING ROOM	02139
ATTENTION: DONALD LINDSAY	K2E 6T7	CANADA	ATTN: COMPUTER UNIT	CV4 7AL	UNITED KINGDOM	ATTN: RECAU (B)	DK-8000 DENMARK
ATTENTION: EVAN L. SOLLEY	97210		ATTN: COMPUTING CENTER	59715		ATTN: RECAU	DK-8000 DENMARK
ATTENTION: E. N. VAN DEVENTER	0001	SOUTH AFRICA	ATTN: CONSULTING OFFICE	61801		ATTN: RECEIVING CLERK	61820
ATTENTION: GARRY S. MEYER	11794		ATTN: C.R.I.G.	F-34000	FRANCE	ATTN: REFERENCE ROOM	K7L 3N6 CANADA
ATTENTION: GORDON R. SHERMAN	37916		ATTN: DATALOGISK INSTITUT	DK-2200	DENMARK	ATTN: REFERENCE ROOM	55455
ATTENTION: JAN LAUGESEN V. 3-357	DK-2500	DENMARK	ATTN: DEPARTMENT OF INFORMATION SCIENCE		NEW ZEALAND	ATTN: REGIONALES RECHENZENTRUM	D-8520 GERMANY
ATTENTION: JAN WITT	D-8000	GERMANY	ATTN: DEPT. OF COMPUTER SCIENCE	38677		ATTN: RESEARCH PROGRAMMING ADVISOR	89154
ATTENTION: JERRY W. SEGERS	30332		ATTN: DEPT. OF MATHEMATICS C/O D.W. BARRON	S09 5NH	UNITED KINGDOM	ATTN: RZ - BIBLIOTHEK	CH-8092 SWITZERLAND
ATTENTION: JO AN HUESMAN	03060		ATTN: DIRECTOR	32611		ATTN: SARA-LIBRARY	THE NETHERLANDS
ATTENTION: MARTIN TUORI	M3M 3B9	CANADA	ATTN: DIRECTOR	M5 4WT	UNITED KINGDOM	ATTN: SCHOOL OF INFORMATION SCIENCES	2616 AUSTRALIA
ATTENTION: MAX SEVCIK	CH-8035	SWITZERLAND	ATTN: DOCUMENTS ROOM	97403		ATTN: SCIENCE LIBRARY	BT9 5EQ UNITED KINGDOM
ATTENTION: MILES RICKARD	78767		ATTN: DOCUMENTS ROOM LIBRARIAN	46637		ATTN: SECRETARY	7001 AUSTRALIA
ATTENTION: M. MALKOSH		ISRAEL	ATTN: DOROTHY SMITH - REFERENCE LIBRARIAN	78712		ATTN: SERIALS DEPT.	52242

ATTN: SERIALS DEPT. 70504
 ATTN: SERIALS LIBRARY 2308 AUSTRALIA
 ATTN: SOFTWARE/HARDWARE GROUP DK-2100 DENMARK
 ATTN: SSRFC LIBRARY 55455
 ATTN: THE DOCUMENTATION OFFICER UNITED KINGDOM
 ATTN: THE LIBRARIAN LA1 4YX UNITED KINGDOM
 ATTN: THE LIBRARIAN 2601 AUSTRALIA
 ATTN: THE LIBRARIAN M60 1QD UNITED KINGDOM
 ATTN: THE LIBRARY ISRAEL
 ATTN: THE LIBRARY TS1 3BA UNITED KINGDOM
 ATTN: UCC LIBRARIAN 52242
 ATTN: USER SERVICES GROUP 80523
 ATTN: USER SERVICES LIBRARIAN 88003
 J. W. ATWOOD H3G 1M8 CANADA
 DAVID AULT 22090
 MARGERY AUSTIN 20037
 GENE AUTREY-HUNLEY 94025
 DELE AYENI N2L 3B8 CANADA
 DANG VAN BA CH-1207 SWITZERLAND
 GARY BABCOCK 93555
 CHARLES BACON 20854
 J. R. BAICHTAL 94040
 DWIGHT BAKER 01752
 FRED P. BAKER 61820
 SAMUEL T. BAKER 37130
 T. P. BAKER 32304
 S. BALASUBRAMANIAN 77001
 LYNNE J. BALDWIN 68101
 A. BALFOUR EH1 2HW UNITED KINGDOM
 EDWARD E. BALKOVICH 06268
 MICHAEL S. BALL 92152
 FRED E. BALLARD 60201
 MAURICE BALLEW 79409
 RICHARD BALOCCA 61801
 DADO BANATAO 95121
 CHARLES J. BANGERT 66045
 JOHN BANNING 94305
 WILLIAM BARABASH 11794
 JOHN R. BARR 90503
 D. W. BARRON SO9 5NH UNITED KINGDOM
 ANN V. BARROW LA1 4YX UNITED KINGDOM
 STEVEN BARRYTE 90048
 NEIL J. BARTA 48106
 JEFFREY BARTH 94720
 BRIT J. BARTTER 60202
 ROGER R. BATE 75023
 DAVID BATES CH-1200 SWITZERLAND
 RODNEY M. BATES 67220
 HENRY R. BAUER III 82071
 JOHN C. BEATTY 94550
 O. BEAUFAYS BELGIUM
 E. R. BEAUREGARD 02809
 MICHAEL A. BEAVER 53115
 MARK BECKER 06432
 BERNHARD H. BEITINGER D-8012 GERMANY
 ABDUL RASAQ BELLO 55408
 STEVEN M. BELLOVIN 27514
 DAVID A. BENNETT 13440
 K. H. BENNETT ST5 5BG UNITED KINGDOM
 LENNERT BENSRYD S-220 07 SWEDEN
 LOUIS A. BENTON 92121
 HERMAN BERG 53703
 PHILIP N. BERGSTRESSER 35758
 THOMAS BERNER D-2000 GERMANY
 BOB E. BERRY LA1 4YX UNITED KINGDOM
 SCOTT BERTILSON 55455

RANDY BEST 76114
 JAMES L. BEUG 93407
 JEAN BEZIVIN F-35031 FRANCE
 ALBRECHT BIEDL D-1000 GERMANY
 MARK BILODEAU 55401
 HERBERT F. BISCHELTSRIEDER D-8012 GERMANY
 C. M. BISHOP NEW ZEALAND
 THOMAS P. BISHOP 14853
 KAY BITTERLING D-1000 GERMANY
 GUS BJORKLUND 22091
 G. RICHARD BLADEN DK-2650 DENMARK
 WILLIAM BLAMPIED S7K 3P7 CANADA
 ALAN BLANNIN BS9 4PL UNITED KINGDOM
 BRADFORD E. BLASING 55455
 I. N. BLAVINS 5006 AUSTRALIA
 ROLAND F. BLOMER D-8000 GERMANY
 P. VAN EMDE BOAS THE NETHERLANDS
 KEITH BOLSON 55423
 RAFAEL M. BONET 12 SPAIN
 JOEL BONEY 78723
 TIM BONHAM 55454
 WILLIAM R. BONHAM 89509
 ERWIN BOOK 90066
 GARY J. BOOS 58501
 KEN BORGENDALE 55455
 ARNE BORTEMARK S-581 83 SWEDEN
 JAMES S. BOTIC 53201
 JEAN-PIERRE*BOUCHEZ 2231 XE THE NETHERLANDS
 RAYMAOND BOUTE B-2000 BELGIUM
 KEN BOWLES 92093
 CHRIS BOYLAN 55068
 GORDON BRADLEY 93940
 DAVID E. BREEDING 75229
 STEPHEN L. BREIBART UNITED KINGDOM
 RONALD F. BRENDER 01754
 BILL BRENNAN 19401
 N. D. BREWER 2076 THE NETHERLANDS
 FRANK BREWSTER 22304
 C. E. BRIDGE 19898
 ROBERT L. BRIECHLE 44325
 C. BRON THE NETHERLANDS
 ALBERT S. BROWN 01754
 ARTHUR A. BROWN 20037
 WARREN R. BROWN 02038
 BOB BRUCE 33307
 BERND BRÜGGE D-2000 GERMANY
 ALF M. BRUNSTROM S-751 21 SWEDEN
 GERALD BRYAN 91711
 DAVID M. BULMAN 92103
 WILHELM BURGER 78712
 MIKE BURGHER 50311
 JOHN W. BURNETT 95051
 D. G. BURNETT-HALL Y01 5DD UNITED KINGDOM
 HOWARD BUSSEY JR. 80302
 BILL BUZBEE 87545
 ADRIAN BYRAM 94087
 EDWIN J. CALKA 21204
 J. A. CAMPBELL 2308 AUSTRALIA
 DAVID F. CANTLEY 97331
 PETER G. CAPEK 10598
 KEVIN W. CARLSON 53705
 ROY CARLSON 97077
 DAVID E. CARLTON 60625
 A. G. CARRICK 95014
 GARY CARTER 89507
 JOHN CASEY 02115

BILLY R. CASON 98004
 CHARLES A. CASTELLOW 98177
 JEAN CASTONGUAY G1W 2P3 CANADA
 D. A. CAUGHFIELD 79601
 GARY CEDERQUIST 75275
 GERALD N. CEDERQUIST 30092
 F. CELLINI N6A 5B7 CANADA
 MIKE CHALENBURG 72143
 NEAL H. CHAMPION 86301
 GABRIEL CHANG 02139
 BILL CHESWICK 18938
 COLE A. CHEVALIER 92704
 PATRICK CHEVAUX CH-1211 SWITZERLAND
 VERNON CHI 97403
 YOUNG J. CHOI 5001 AUSTRALIA
 FAY CHONG 95014
 LARS CHRISTENSEN DK-2880 DENMARK
 LUTZ CHRISTOPH D-1000 GERMANY
 PAUL CHRISTOPHERSON 55343
 GERALD W. CICHANOWSKI 55987
 RICHARD J. CICHELLI 18103
 ROBERT G. CLARK UNITED KINGDOM
 JENNIFER CLARKE 02115
 P W R CLARKE SW7 2AZ UNITED KINGDOM
 GEOFFREY A. CLEAVE 3165 AUSTRALIA
 DAVID G. CLEMANS 90064
 KURT COCKRUM 92507
 WILLIAM L. COHAGAN 78758
 GEORGE COHN III 47401
 JOE COINTMENT 75248
 PETER COLBY 02160
 D. B. COLDRICK K1J 6L2 CANADA
 DEREK COLEMAN M60 1QD UNITED KINGDOM
 TERRENCE M. COLLIGAN 02193
 JOHN E. COLLINS 55101
 RIC COLLINS M13 9PL UNITED KINGDOM
 DOUGLAS COMER 47907
 MICHAEL N. CONDUCT 13440
 JAMES B. CONKLIN JR. 32611
 APRIL MILLER CONVERSE 94025
 RICHARD CONWAY 14850
 DEXTER COOK 75220
 CHRISTOPHER S COOPER OX11 0QX UNITED KINGDOM
 WILLIAM L. COOPER 92805
 C. J. COPELAND BT37 0QB UNITED KINGDOM
 F. J. CORBATO 02139
 RICHARD CORE 94088
 B. J. CORNELIUS HU6 7RX UNITED KINGDOM
 MIKE W. CORNELIUS LA1 4YX UNITED KINGDOM
 C R CORNER 56560
 JOHN DENNIS COUCH 95050
 FREDERICK C. COWAN 90009
 LARRY CRANE 50309
 JOHN EARL CRIDER 77043
 LINDA E. CROLEY 94304
 WILLIAM E. CROSBY 92714
 DONALD B. CROUCH 35486
 ZAY CURTIS 94035
 BJARNE DACKER S-126 25 SWEDEN
 LUIS M.M. DAMAS PORTUGAL
 DENNIS DANCE 72204
 RONALD L DANIELSON 95051
 JAMES DARLING 87801
 GENE A. DAVENPORT 10016
 ANN D. DAVIES 23284
 R. JAMES DAWE A1C 5S7 CANADA

BRUCE DAWSON 40208
 FRANKLIN B. DE GRAAF K2K IK2 CANADA
 JOHN DE ROSA JR. 01609
 HAROLD DE VORE 55337
 D. D. DE VRIES THE NETHERLANDS
 W. DE VRIES THE NETHERLANDS
 JOHN R. DEALY 90278
 RANCE J. DELONG 18018
 DAVID DEMOREST 98124
 E. DENERT D-8000 GERMANY
 DOROTHY E. DENNING 47907
 TIMOTHY DENNIS 06035
 G. D. DERHAK R3T 2N2 CANADA
 PIERRE DESJARDINS H3C 3J7 CANADA
 S. T. DEVEREUX SE1 0AA UNITED KINGDOM
 BOB DICKERSON AL10 9AB UNITED KINGDOM
 KENNETH A. DICKEY 95521
 JOHN DICKINSON 83843
 LLOYD DICKMAN 01776
 KLAUS R. DITTRICH D-7500 GERMANY
 E. H. DOBELL 2007 AUSTRALIA
 JOHN G. DOBNICK 53219
 RANDY DODGE A1C 587 CANADA
 ROBERT ALAN DOLAN 93109
 DAN DORROUGH 47906
 JOHN DOW 15261
 KEVIN R. DRISCOLL 55414
 JEFFREY J. DRUMMOND 55455
 ROGER A. DUE 02111
 STEPHEN A. DUM 97005
 DONNA K. DUNAWAY 75222
 FRANK DUNN 75081
 RON DYKSTRA 55455
 DOUG DYMENT V7W 2J6 CANADA
 T. A. D'AURIA 10027
 R. STERLING EANES 02154
 WILLIAM J. EARL 92715
 JOHN EARLS 75240
 CHRIS EASTLUND 55401
 JEFF EASTMAN 80537
 JOHN T. EASTON 55455
 NORBERT EBEL CH-2000 SWITZERLAND
 GLENN T. EDENS 94086
 HANK EDWARDS 01701
 ROY EDWARDS TW20 OEX UNITED KINGDOM
 FRED EILENSTEIN 02172
 JOHN D. EISENBERG 19711
 HOWARD EISENSTEIN 29206
 T. W. EKBERG 75234
 LARS EKMAN DK-1601 DENMARK
 JIM ELAM 94111
 DAVE R. ELAND 74171
 DENNIS R. ELLIS 80303
 HORACE ENEA 94022
 G. ENGELIEN D-5300 GERMANY
 DAVE ENGLANDER 18015
 PHILLIP H. ENSLOW JR. 30332
 P. J. ERARD CH-8027 SWITZERLAND
 DENNIS ERNST 94086
 E. W. ERRICKSON 85061
 HOWARD D. ESKIN 10025
 JOHN B. EULENBERG 48824
 BLAND EWING 94720
 R. NEIL FAIMAN JR. 48228
 DENNIS FAIRCLOUGH 84601
 JAMES N. FARMER 30332

JACQUES FARRE F-75230 FRANCE
 JOSEPH H. FASEL III 47907
 JEAN-PIERRE FAUCHE F-38040 FRANCE
 LUCIEN FELEREISEN D-7500 GERMANY
 EDWARD E FERGUSON 76201
 LINWOOD FERGUSON 22901
 JEANNE FERRANTE 02139
 JOSE OSVALDO FERRARI 13100 BRAZIL
 LINCOLN FETCHER 55455
 CHARLES J. FETE 92704
 SUSAN FEUERMAN 93940
 N. J. FIDDLIAN UNITED KINGDOM
 ROGERIO BURNIER FILHO 13100 BRAZIL
 BILL FINDLAY G12 8QQ UNITED KINGDOM
 CHARLES N. FISCHER 53706
 GLENN FISHBINE 55101
 A. J. FISHER HR1 1TY UNITED KINGDOM
 WILLIAM E. FISHER 90501
 TED FISHMAN 75222
 DAVID C. FITZGERALD 91740
 KEVIN FJELSTED 55455
 HANS FLACK S-751 21 SWEDEN
 J. J. FLORENTIN WCIE 7HX UNITED KINGDOM
 RUDY L. FOLDEN 92714
 JIM FONTANA 92704
 CHARLES H. FORSYTH N2J 4T2 CANADA
 MARY DEE FOSBERG 73034
 LLOYD D. FOSDICK 80309
 ARTHUR FOSTER LA1 4YX UNITED KINGDOM
 L. M. FOSTER 92663
 W. BRUCE FOULKES R3T 2N2 CANADA
 ED FOURT 94720
 PALTONIO DAUN FRAGA 13100 BRAZIL
 DENNIS J. FRAILEY 75222
 ROBERT A. FRALEY V6T 1W5 CANADA
 MIKE FRAME 20006
 REX FRANCIOTTI 11530
 R. A. FRANCIS SW7 2AZ UNITED KINGDOM
 K. FRANKOWSKI 55455
 KURT FREDRIKSSON S-431 39 SWEDEN
 TED L. FREEMAN 20705
 DANA A. FRETBURGER 93407
 G. FRIEDER 14226
 EDWARD R. FRIEDMAN 10012
 FRANK L. FRIEDMAN 19122
 GERHARD FRIESLAND D-2000 GERMANY
 SERGE FROMENT H1Z 3P1 CANADA
 JOHN FUNG 55414
 WAYNE FUNG V6X 2Z9 CANADA
 DAN FYLSTRA 02134
 MICHEL GALINIER F-31077 FRANCE
 SAM GEBALA 94304
 EDWARD F. GEHRINGER 47907
 W. MORVEN GENTLEMAN N2L 3G1 CANADA
 J. D. GEORGE 32806
 A. J. GERBER 2006 AUSTRALIA
 PAUL S. GERKEN 94611
 J. DANIEL GERSTEN 13201
 ROBERT A. GIBSON 22901
 DAVID W. GIEDT 92807
 N. AMOS GILEADI 01754
 ED GLASER 54302
 JOHN J. GODA JR. 30332
 PAUL GODFREY 84601
 ANTHONY LESLIE GOLBORN GU16 5HJ UNITED KINGDOM
 HELLMUT GOLDE 98195

G. H. GOLDEN JR. 14063
 DAVID A. GOMBERG 20016
 GEORGE GONZALEZ 55108
 I. GOODE TW11 0LW UNITED KINGDOM
 RALPH S. GOODELL 01451
 JOHN B. GOODENOUGH 02154
 J. GOODSON S09 5NH UNITED KINGDOM
 GERHARD GOOS D-7500 GERMANY
 D. GOSMAN THE NETHERLANDS
 JOHN S. GOURLAY 48105
 SARA K. GRAFFUNDER 55455
 DENNIS GRAHAM 94086
 JEFFREY W. GRAHAM 35223
 SUSAN L. GRAHAM 94720
 WILLIAM Q. GRAHAM 19711
 M. J. GRALIA 20810
 JOHN M. GRAM 92717
 DAVID N. GRAY 78769
 KRISTINA GREACEN 55455
 MARK GREEN L8N 3W3 CANADA
 MIKE GREEN 78284
 TOM GREER 91775
 R. GREINER 95014
 WILEY GREINER 90278
 DAVID J. GRIFFITHS 02881
 DONALD E. GRIMES 95014
 R. L. GRIMSDALE BN1 9QT UNITED KINGDOM
 MARTIN L GRISS 84112
 DALE H. GRIT 80523
 PETER GROGONO H4V 2H3 CANADA
 WILLIAM GROSZY 48202
 JONATHAN R. GROSS 55435
 STEVE GROSS 10024
 LARRY GROVER 56301
 GEORGE GRUNWALD 47306
 ROGER GULBRANSON 61801
 S. L. GULDEN 18015
 HILMAR GUTFELDT CH-3000 SWITZERLAND
 PETER GUTTERMAN 20433
 DAVE HABERMAN 75081
 THOMAS HABERNOLL D-1000 GERMANY
 MICHAEL HAGERTY 02174
 JACQUES HAGUEL J1K 2R1 CANADA
 HARRY P. HAIDUK 79015
 B. HALE MK40 2PN UNITED KINGDOM
 THOMAS HALLDORSON 18353
 JOEL M. HALPERN 55455
 RONALD J. HAM 01754
 DAVID E. HAMILTON 23666
 DAN C.C. HAMM KT12 5NF UNITED KINGDOM
 TERRY HAMM 97077
 U. HAMMELEFF DK-8000 DENMARK
 DON HAMNES 55409
 WOLFGANG HAMPE D-1000 GERMANY
 MICHAEL Z. HANANI ISRAEL
 C. C. HANDLEY 4000 SOUTH AFRICA
 SCOTT D. HANKIN 01720
 KAY A. HANSBOROUGH 87544
 GILBERT J. HANSEN 75075
 KELD HELBIG HANSEN DK-2800 DENMARK
 RANDALL W. HANSEN 55440
 BRIAN HANSON 55455
 JON HANSON 55440
 SAM HARBAUGH 32901
 T. HARDY 20234
 EDWARD H. HARRIS 53705

KAY HARRISON	N2L 3G1 CANADA	A. J. HURST	2600 AUSTRALIA	RICHARD KIMBALL	01754
STEPHEN J. HARTLEY	22901	DAVID HUSNIN	73106	MIKE KIMBER	M5V 2S9 CANADA
ROBERT L. HARTMAN	92701	BOB HUTCHINS	92713	C. B. KING	UNITED KINGDOM
AL HARTMANN	95051	JOHN HUTCHINSON	E1 4NS UNITED KINGDOM	GEOFF V KING	NPT 1XG UNITED KINGDOM
J. P. HARVELL	75081	STEVEN L. HUYSER	48824	ROBERT L. KING	13760
DON HARVEY	97077	DANIEL C. HYDE	17837	ROBERT KIRKBY	HA4 9DP UNITED KINGDOM
ISAMU HASEGAWA	E14 UNITED KINGDOM	DAVID H. WELCH	92324	ZENICHI KISHIMOTO	560 JAPAN
KEITH HAUER-LOWE	55417	M. ELIZABETH IBARRA	11973	PETER KLAUBERG	D-2000 GERMANY
KEVIN HAUSMANN	55113	KLAUS ILLUM	DK-9000 DENMARK	M. A. KLEINERT	84112
DAVID HAWK	19711	GIORGIO P. INGARGIOLA	19122	DONALD S. KLETT	62708
GEORGE E. HAYNAM	32901	ARON K. INSINGA	19711	P. F. KLOK	THE NETHERLANDS
JOHN HEATH	04103	JOHN W. IOBST	18049	J. C. KNIGHT	23665
JAMES W. HEBERT	01907	AVRUM ITZKOWITZ	61820	BRUCE KNOBE	02138
PAUL HECKEL	94305	GARY M. JACKSON	55414	ROLF G. KNOEPKER	D-7500 GERMANY
H. G. HEDGES	48824	M. I. JACKSON	AL1 1RZ UNITED KINGDOM	G. J. KNOX	3000 AUSTRALIA
CHARLES HEDRICK	08903	CHRISTIAN JACOBI	CH-8092 SWITZERLAND	SVEND ERIK KNUDSEN	CH-8092 SWITZERLAND
J. B. HEIDEBRECHT	90278	KRISTER JANZON	S-172 04 SWEDEN	CARSTEN KOCH	D-2000 GERMANY
S. T. HEIDELBERG	94550	ROBERT L. JARDINE	92675	CARSTEN KOCH (B)	D-2000 GERMANY
DENNIS HEIMBIGNER	90278	STJEPAN JARNJAK	41000 YUGOSLAVIA	DENNIS KODIMER	85260
JUHA HEINANEN	SF-33101 FINLAND	PATRICK L. JARVIS	55455	KURT KOHLER	97331
T. S. HEINES	44115	MICHEL JAUNIN	CH-1007 SWITZERLAND	FRITHJOF KOLBERG	95054
DAVID HELFINSTINE	55303	RALPH D. JEFFORDS	38677	ALAN A. KORTESOJA	48103
PAUL S. HELLER	08540	GEORGE D. JELATIS	55455	WALTER KOSINSKI	92705
CARL HELMERS	03458	J M JENKIN	AL1 INF UNITED KINGDOM	MARTIN R. KRAIMER	60439
PAUL HELVIG	56301	D. G. JENKINS	G12 8QQ UNITED KINGDOM	JEFF KRAMER	SW7 2AZ UNITED KINGDOM
RICHARD HENDRICKSON	55420	KATHLEEN JENSEN	01749	LOU H. KRAMER	THE NETHERLANDS
STEN HENRIKSSON	S-223 62 SWEDEN	MITCHELL R. JOELSON	55455	R. KRASIN	02154
CARL HENRY	55057	CHRISTOPHER K. JOHANSEN	28743	JAMES KRELICH	55108
WILLIAM HENRY	10003	GUNNAR JOHANSEN	DK-2800 DENMARK	DIETRICH KREKEL	D-5000 GERMANY
MARK HERSEY	48823	GERALD C. JOHNS	63110	FRANCIS KRICKORIAN	94143
CHARLES L. HETHCOAT III	77027	BRIAN W. JOHNSON	75075	DIRK KRONIG	D-7750 GERMANY
GEORGE C. HETRICK	02167	DOUGLAS S. JOHNSON	75081	FRANK KURKA	07757
BRYAN L. HIGGINS	94621	JOSEPH P. JOHNSON	20016	MARVIN E. KURTTI	35801
STANLEY B. HIGGINS	37232	ROBERT T. JOHNSON	87545	ANTHONY P. KYNE	3052 AUSTRALIA
JIM HIGHTOWER	90274	R. I. JOHNSON	58202	IVAR LABERG	1 NORWAY
TERUO HIKITA	113 JAPAN	R. WARREN JOHNSON	56301	JOSEPH LACHMAN	60076
DAVID HILL	95819	S. M. JOHNSON	NR4 7TJ UNITED KINGDOM	JACK LAFFE	55454
TIM HILL	22901	WARREN JOHNSON	70504	R. B. LAKE	44106
SAM HILLS	70125	D. B. JOHNSTON	4067 AUSTRALIA	DAN LALIBERTE	55455
W. A. HINTON	53211	K. DOUGLAS JOHNSTON	92026	JOHN A. LAMBERT	2308 AUSTRALIA
ED HIRAHARA	92653	RICHARD A. JOKIEL	19464	LEE LAMBERT	19426
MATTIA HMEIJAK	I-34100 ITALY	D. A. JOSLIN	BN2 6RD UNITED KINGDOM	LARRY D. LANDIS	64108
THEA D. HODGE	55455	TOOMAS KAER	S-434 00 SWEDEN	STEVE LANDRY	70504
TIMOTHY W. HOEL	55057	KARLHEINZ KAPP	D-7500 GERMANY	DAVID LANDSKOV	70504
MARILYN HOFFMAN	18018	BARBARA I. KARKUTT	18042	C. A. LANG	CB2 1RP UNITED KINGDOM
H.-J. HOFFMANN	D-6100 GERMANY	RICHARD H. KARPINSKI	94114	ROBERT M. LANSFORD	91107
TIMOTHY J. HOFFMANN	55455	HEIKKI KASKELMA	SF-00130 FINLAND	RAINER R. LATKA	D-8031 GERMANY
DAVID W. HOGAN	78751	ED KATZ	70504	JOHN N. LATTA	22210
WILLIAM C. HOPKINS	19174	MARK J. KAUFMAN	92025	ROBERT A. LAWLER	55165
GREGORY L. HOPWOOD	92713	DOUGLAS R. KAYE	10019	CHARLES L. LAWSON	91103
FRANK H. HORN	53706	VINCENT KAYSER	03301	WILLIAM M. LAYTON	03766
TOM HORSLEY	95376	TOM KEEL	78712	CHRIS LAZOU	WC1 UNITED KINGDOM
FRED A. HOSCH	70122	THOMAS A. KEENAN	20550	RICHARD LEBLANC	53706
DAVID A. HOUGH	23602	ED KEITH	91702	O. LECARME	F-06034 FRANCE
ROSEMARY HOWBRIGG	06413	GINGER KELLY	77001	HENRY F. LEDGARD	01002
RALPH HOWENSTINE	73070	TOM KELLY	18974	KYU Y. LEE	83401
RICHARD HOYME	55427	JOE KELSEY	98195	R. GARY LEE	32306
PEI HSIA	35807	WILLETT KEMPTON	78705	STEVE LEGENHAUSEN	08904
PETER YAN-TEK HSU	55455	JAMES A. KENDALL	77030	STEPHEN LEIBOWITZ	10016
RICHARD HUBER	77843	LESLIE R. KERR	98004	KENNETH O. LELAND	92152
JON F. HUERAS	92717	MARK C. KERSTETTER	49008	MIKE LEMON	61738
JACK HUGHES	K7L 3N6 CANADA	NORM P. KERTH	97077	DENIS LENIHAN	SW11 UNITED KINGDOM
ALFRED J. HULBERT	87115	ROBERT KEZELL	19122	ROBERT S. LENT	94086
GEOFFREY HUNTER	M3J 1P3 CANADA	B. KIDMAN	5001 AUSTRALIA	BENTON LEONG	16802
PAUL K. HUNTWORK	55112	RICHARD B KIEBURTZ	11794	JERRY LEVAN	40475
EDWARD W. HURLEY	22101	D. B. KILLEEN	70118	LANCE A. LEVENTHAL	92067

JOHAN LEWI B-3030 BELGIUM
 GEORGE LEWIS 94086
 H. MARC LEWIS 93453
 JOHN LEWIS 21218
 JOHN W. LEWIS AL10 9AB UNITED KINGDOM
 L. RICHARD LEWIS 48127
 A. C. W. LEYEN THE NETHERLANDS
 HUBERT LEYGRAF D-1000 GERMANY
 P. LIAO 95014
 SAM LIBAI ISRAEL
 LAWRENCE A. LIDDIARD 55455
 KLAUS LIEBENWALD D-2000 GERMANY
 DENNIS R. LIENKE 55455
 RITA MAY LIFF 94613
 GEORGE LIGLER 75080
 ALAN LILLICH V5N 3K1 CANADA
 SHIHTA LIN 55455
 JOHN E. LIND 55455
 JOHN R. LINDSAY 44306
 CHRIS P. LINDSEY 91711
 GARY LINDSTROM 84112
 BRUCE LINK 87115
 DAVID LIPPINCOTT 48107
 SAM LISOOK 75234
 STEN LJUNGKVIS S-603 78 SWEDEN
 CARLO LOCICERO H1Y 3C3 CANADA
 BRIAN D. LOCKREY 85281
 LUIGI LOGRIPPO K1N 6N5 CANADA
 RALPH L. LONDON 90291
 WARREN EDWARD LOPER 92111
 ANDY LOPEZ 56267
 ROBERT E. LORD 99163
 BRUNO LORTZ D-7500 GERMANY
 R. A. LOVESTEDT 98055
 GARY LOWELL 95404
 TIM LOWERY 32304
 DAVID C. LUCKHAM 94305
 MANFRED LUCKMANN D-8000 GERMANY
 LEON LUKASZEWICZ 00901 POLAND
 MARK LUKER 55812
 STANLEY E. LUNDE 91711
 STEVE LUNDQUIST 92714
 MICHAEL J. LUTZ 14623
 WILLIAM LYCZKO 14850
 JOHN T. LYNCH 19301
 M. H. MACDOUGALL 94086
 K. J. MACGREGOR SOUTH AFRICA
 BRUCE MACKENZIE 01730
 PETER H. MACKIE 97005
 IAN MACMILLAN H3P 3B9 CANADA
 JIM MADDEN 92093
 ORLANDO S. MADRIGAL 95926
 H. S. MAGNUSKI 94304
 LARS MAGNUSSON S-751 21 SWEDEN
 FRANZ W. MAIER A-5020 AUSTRIA
 PHILIP J. MALCOLM EC3V LLP UNITED KINGDOM
 D. MARCUS 92807
 RICK L. MARCUS 55404
 C. D. MARLIN 5001 AUSTRALIA
 E. MARMIER CH-8021 SWITZERLAND
 G. MARQUARDT D-3000 GERMANY
 GREGG E. MARSHALL 80201
 MARK T. MARSHALL 91335
 WILLIAM C. MARSHALL 55413
 CHRIS MARTIN S10 2TN UNITED KINGDOM
 RONALD G. MARTIN 68123

GENE MARTINSON 55440
 JAMES F. MARTINSON 56201
 PRABHAKER MATELI 3052 AUSTRALIA
 CRAIG MAUDLIN 92121
 P. MAURICE F-31077 FRANCE
 KONRAD MAXER A-1150 AUSTRIA
 MARK S. MAYES 01742
 JIM MCCORD 93101
 RAINER F. MCCOWN 21045
 PAUL L. MCCULLOUGH 97077
 THOMAS G. MCGINTY 02035
 M. L. MCGRAW 30328
 BRIAN MCGUIRE 94538
 PAUL MCJONES 94304
 HUGH MCLARTY 94305
 STUART JAMES MCRAE SW7 2AZ UNITED KINGDOM
 JACK R. MEAGHER 49008
 TERRY P. MEDLIN 20014
 MICHAEL MEHAN 02138
 BRIAN MEEK W8 7AH UNITED KINGDOM
 BRIAN A. E. MEEKINGS LA1 4YX UNITED KINGDOM
 HUGO MEISSER 55427
 MICHAEL MEISSNER 55455
 ERIC MELBARDIS H3G 2C8 CANADA
 THOMAS MELLMAN 62901
 JIM MERRITT 94704
 J. SCOTT MERRITT 12180
 DAVID C. MESSER 55441
 ERNST MESSMER CH-1211 SWITZERLAND
 HOWARD H. METCALF 90068
 W. J. MEYERS 75243
 JOSEPH A. MEZZAROBBA 18041
 ANDY MICKEL 55455
 M. D. MICKUNAS 61801
 R. W. MILKEY 85726
 CHARLES E. MILLER 17257
 C. A. MILLER T6G 2N5 CANADA
 DAVID MILLER 21044
 GLENN MILLER 55109
 JAMES R. MILLER 47902
 VICTOR S. MILLER 02125
 CARLTON MILLS 61801
 JAMES F. MINER 55455
 B. T. MITCHELL KY16 UNITED KINGDOM
 SANDEE MITCHELL 40208
 TONY MITCHELL LS2 9JT UNITED KINGDOM
 JESSE D. MIXON 75961
 TOM MOBERG 50112
 DAVID MOBERLY 01754
 KEN MODESITT 91330
 TOM MOHER 55455
 UFFE MOLLER DK-9000 DENMARK
 JAMES MOLONEY 14420
 ALLAN MOLUF 48910
 JOHN MONTAGUE 87545
 MAURO MONTESI I-40122 ITALY
 CHARLES G. MOORE 48106
 JAMES K. MOORE 22091
 WILLIAM C. MOORE JR. 23234
 R. MOREL CH-1211 SWITZERLAND
 TONEY MORELOCK 77001
 CARROLL MORGAN 2006 AUSTRALIA
 CHARLES ROBERT MORGAN 02138
 RONALD G. MOSIER 48221
 WILLIAM MOSKOWITZ 90036
 LARS G. MOSSBERG S-461 01 SWEDEN

STEVEN S. MUCHNICK 66045
 JUDY MULLINS S09 5NH UNITED KINGDOM
 DAVID A. MUNDIE 22903
 NEWTON J. MUNSON 13676
 CHARLES F. MURPHY LIBYA
 GERALD NADLER 02154
 H.-H. NAGEL D-2000 GERMANY
 HANS-HEINRICH NAGEL CH-8092 SWITZERLAND
 T. RAY NANNEY 29613
 T. A. NARTKER 87801
 ISAAC R. NASSI 01754
 RONALD S. NAU 22314
 JOHN NAUMAN 55455
 DAVID NEAL 66502
 BERNHARD NEBEL D-2000 GERMANY
 PETER NELLESSEN D-1000 GERMANY
 BRIAN NELSON 43606
 DAVID A. NELSON 19104
 MALCOLM C. NEWAY 2600 AUSTRALIA
 LE H. NGUYEN 32604
 B. NIBLETT SA2 8PP UNITED KINGDOM
 ROBERT C. NICKERSON 95003
 DENNIS NICKOLAI 55437
 HOLGER NIELSEN DK-8200 DENMARK
 MARK S. NIEMCZYK 60015
 JOHN NOLAN 20755
 JOHN NOLD 15701
 TERJE NOODT 3 NORWAY
 LEO C. NOORDHUIZEN THE NETHERLANDS
 R. K. NORDIN 55454
 BENGT NORDSTROM S-402 20 SWEDEN
 THEODORE A. NORMAN 84602
 JOHN L. NORSTAD 60201
 DAVID A. NUESSE 54701
 JOHN NUNNALLY 72143
 G. OAKES DE3 6RU UNITED KINGDOM
 CAROL A. OGDIN 22314
 RICHARD OHRAN 84602
 FLEMING M. OLIVER 94086
 ERIC OLSEN 92713
 NORMAN T. OLSEN 80201
 RON OLSEN 07733
 GENE H. OLSON 55343
 KENNETH OLSON 02140
 OLLE OLSSON S-752 23 SWEDEN
 FRANK OLYNYK 44106
 LENNART OSKARSSON S-431 20 SWEDEN
 ALEX OSTAPENKO 18055
 MARK OVERGAARD 92093
 JORGEN OXENBOLL DK-2100 DENMARK
 PAUL O-BRIEN 97210
 DANIEL M. O'BRIEN 60030
 MARK T. O'BRYAN 49007
 MAURICE O'FLAHERTY UNITED KINGDOM
 S. J. PACKER 92704
 F. G. PAGAN A1C 5S7 CANADA
 CHRIS D. PAICE LA1 4YX UNITED KINGDOM
 T. L. (FRANK) PAPPAS 19018
 TED C. PARK 92408
 JOSEPH A. PARKER JR. 18703
 PHILIP PARKER DK-2000 DENMARK
 WALT PARRILL 62025
 FRANK PAVLIK 10013
 PETER PAWELCZAK 10019
 DONALD D. PECKHAM 92714
 PATRICK PECORARO 85721

SHMUEL PELEG 20742
M. A. PELL M13 9PL UNITED KINGDOM 98221
HAL PERKINS 14853
WALT PERKO 55414
DAVID PERLMAN 55455
RUSS PETERSON 55112
SUE PETERSON 55113
W. W. PETERSON 96822
TRUMAN C. PEWITT 60439
CHARLES PFLEEGER 37916
BOB PHILLIPS 97212
CHRIS K. PHILLIPS 94903
ATE PHUNG D-5100 GERMANY
PAUL PICKELMANN 48109
DOUG PIHL 55440
ALAIN PIROTTE B-1170 BELGIUM
TOM PITTMAN 95153
STEPHEN A. PITTS 73110
P. J. PLAUGER 10036
S. POKROVSKY 630090 USSR
KEN POLAKOWSKI 07828
RUDOLPH C. POLENZ 54701
BARY W. POLLACK V6T 1W5 CANADA
J. E. POLLACK 94566
GEORGE POONEN 02168
L. C. PORTIL N9B 3P4 CANADA
J. L. POSDAMER 13210
LEE POTTS 63188
FRED W. POWELL 24401
KARL PRAGERSTORFER A-4060 AUSTRIA
TERRENCE PRATT 22901
WERNER F. PRAUTSCH D-1000 GERMANY
C. B. A. PRICE HP2 5HG UNITED KINGDOM
RON PRICE 07724
WILLIAM C. PRICE 97068
MICHAEL PRIETULA 55455
DAVID KARL PROBST H3G 1M8 CANADA
STEPHEN G. S. PROUT THE NETHERLANDS
MARLIN PROWELL 98225
ANDREW S. PUCHRIK 47130
ERIC PUGH 90024
GREG PUGH SW7 2AZ UNITED KINGDOM
BRUCE A. PUMPLIN 54701
HOWARD D. PYRON 65401
DOUGLAS H. QUEBBEMAN 47150
IRVING N. RABINOWITZ ISRAEL
WILLIAM F. RAGSDALE 94545
THOMAS RAMSBERGER 18015
V. LALITA RAO 18015
CHARLES RAPIN CH-1007 SWITZERLAND
WAYNE RASBAND 20014
PETER RAUSCHMAYER D-8000 GERMANY
BRUCE K. RAY 80307
JERRY L. RAY 68022
JEFFERY M. RAZAFSKY 64108
ALAN REED B15 2TT UNITED KINGDOM
MIKE J. REES SO9 5NH UNITED KINGDOM
ROY F. REEVES 43220
L. EDWARD REICH 22201
C. EDWARD REID 32303
PHYLLIS A. REILLY 90746
J. REINFELDS 2500 AUSTRALIA
ROBERT REINHARDT 61 000 YUGOSLAVIA
WERNER REMMELE D-8000 GERMANY
JOHN REYNOLDS N8 UNITED KINGDOM
JOHN D. REYNOLDS 35801

DAN C. RICHARD 07724
PETER RICHARDSON 3052 AUSTRALIA
GEORGE H. RICHMOND 80309
CLAES RICKEYB S-161 54 SWEDEN
PETER A. RIGSBEE 20375
JENS PETER RINGGAARD DK-2730 DENMARK
MARK RIORDAN 48824
KEN RITCHIE 68005
TERRY RITTER 78753
CLARK M. ROBERTS 91016
JOE C. ROBERTS 75042
MARK L. ROBERTS 90274
KEN ROBINSON 2033 AUSTRALIA
N ROBINSON HA6 3DZ UNITED KINGDOM
J. S. ROHL 6009 AUSTRALIA
S. ROHLFS D-8000 GERMANY
THOMAS A. ROLANDER 95008
STAFFEN ROMBERGER S-100 44 SWEDEN
MICHAEL ROONEY 02154
CARL S. ROSENBERG 94035
RAYNER K. ROSICH 80302
BERNIE ROSMAN 01701
R. WALDO ROTH 46989
E. L. ROWE 19301
H. J. ROWE LEL 7RH UNITED KINGDOM
LAWRENCE A. ROWE 94720
DAVID ROWLAND 97229
DON H. ROWLAND 87109
BRIAN G. ROWSWELL 2006 AUSTRALIA
HERBERT RUBENSTEIN 80401
NANCY RUIZ 87115
C. A. RUSBRIDGE 5000 AUSTRALIA
MARK RUSTAD 55112
JOHN L. RUTIS 97123
FRANK RYBICKI 19122
KARL H. RYDEN 90024
DAVID J. RYPKA 43210
JONATHAN SAGHS 60604
MORLEY W. SAGE SO9 5NH UNITED KINGDOM
TOSHIAKI SAISHO 143 JAPAN
HIKMET SAKA LA1 4YX UNITED KINGDOM
ANTTI SALAVA SF-00330 FINLAND
A. H. J. SALE 7001 AUSTRALIA
TIMOTHY J SALO 55455
CHESTER J. SALWACH 18960
A. E. SALWIN 20810
D. SANDEE THE NETHERLANDS
TOM SANDERSON 87002
HELMUT SANDMAYR CH-9470 SWITZERLAND
HORST SANTO D-5205 GERMANY
DAVID SARANEN 55792
LYNN SAUNDERS 97077
AARON SAWYER 02035
BOB SCARLETT 55455
ANTHONY J. SCHAEFFER 47401
HELMUT SCHAUER A-1040 AUSTRIA
JERRY SCHIEFFER 75229
ROSS D. SCHMIDT 55343
G. MICHAEL SCHNEIDER 55455
SERGIO DE MELLO SCHNEIDER 13560 BRAZIL
ERIC SCHNELLMAN 98117
P. SCHNUPP D-8000 GERMANY
MARK A. SCHROEDER 75214
DEAN SCHULZ 95051
ROLF SCHUMACHER D-1000 GERMANY
CARL U. SCHWARZ 01752

STEPHEN C. SCHWARM 19898
ARTHUR I. SCHWARZ 90230
FRED L. SCOTT 33314
THOMAS SCOTT 19085
BARRY SEARLE KIA ON8 CANADA
DAVID SEGAL 10003
MARK SEIDEN 06901
MANFRED SEIFERT D-7500 GERMANY
BRUCE SELLER 90024
WAYNE SELPEL 78712
GUISEPPE SELVE I-40122 ITALY
SHARAD C. SETH 68588
MICHAEL SETTLE 76011
GEORGE A. SEYFERT 32901
G. M. SHANNON 02173
ED SHARP 84112
DAVID ELLIOT SHAW 94022
JEFFRY G. SHAW 94025
JOHN M. SHAW 20014
WILLIAM F. SHAW 01754
BELLE SHENOY 55413
DAVID SHIELDS 10012
ROBERT A. SHIVE JR. 39210
KIM L. SHIVELEY 87801
BEN SHNEIDERMAN 20742
ARNOLD SHORE 22311
JAMES P. SHORES 06320
GERALD A. SHOULTS 75240
BILL SIMMONS 55440
E. E. SIMMONS 91101
CHARLES E. SIMON 06488
SEYMOUR SIMON 92634
THOMAS W. SKELTON 48823
DAVID SLATER SW7 2AZ UNITED KINGDOM
J. B. SLATER NW3 7ST UNITED KINGDOM
P. A. SLATS THE NETHERLANDS
LEO J. SLECHTA 55165
BARRY SMITH 97221
BROOKS DAVID SMITH 53211
IAIN SMITH RG1 7QN UNITED KINGDOM
JOYCE A. SMITH 20742
LAURA SNYDER 47401
ROBERT J. SNYDER 46202
JOHN S. SOBOLEWSKI 98195
THOMAS C. SOCOLOFSKY 48823
MARY LOU SOFFA 15260
N. SOLNTEFF L8S 4K1 CANADA
DAVID SOLOMONT 02155
MARCO SOMMANI I-56100 ITALY
MANFRED SOMMER D-8000 GERMANY
NORMAN E. SONDAK 01609
ROLF SONNTAG D-3000 GERMANY
BRUCE M. SORLIE 55406
STEPHEN SOULE T2N 1N4 CANADA
JOHN H. SPANTON 95133
TERRY L. SPEAR 80309
RICHARD SPELLERBERG 55440
MARTHA L. SPENCE 01741
DAVID SPENCER BH22 8HL UNITED KINGDOM
HENRY SPENCER M5W 1N5 CANADA
RICHARD D. SPILLANE 07470
D. H. SPRINGER 95131
TOM SPURRIER 32901
JOHN P. STALLINGS 95014
JOHN STANLEY 55404
L. S. C. STATEMA THE NETHERLANDS

JORGEN STAUNSTRUP 90007
ROD STEEL 97077
EDWARD STEEN 01852
GORDON A. STEGINK 49401
HAL STEIN 47401
GERALD STEINBACK 29208
ALBERT STEINER 60201
WARREN STENBERG 55416
VIC STENNING GU16 5HJ UNITED KINGDOM
PHILIP STEPHENSON 76019
CALVIN STEVENS 55422
W. RICHARD STEVENS 85726
IAIN STINSON SW7 2AZ UNITED KINGDOM
ANNE STOCCO NIG 2W1 CANADA
A. I. STOCKS 70504
JERRY STODDARD 55440
JOHN P. STRAIT 55455
GEORGE O. STRAWN 50011
JOHN N. STRAYHORN 02139
EDWARD P. STRITTER 78721
ROBERT A. STRYK 55424
GORDON STUART V8P 5J2 CANADA
PETER R. SUMNER 6005 AUSTRALIA
MARKKU SUNI SF-20500 FINLAND
EDWARD W. SUOR 14609
SILVIA SUSSMAN WC1 UNITED KINGDOM
LARS SVENSSON S-281 00 SWEDEN
RALPH W. SWEARINGEN 94596
S. D. SWIERSTRA THE NETHERLANDS
ADA SZER A-1040 AUSTRIA
MENACHEM SZUS ISRAEL
PREBEN TAASTI DK-9000 DENMARK
RICHARD TABOR 20052
MASATO TAKEICHI 182 JAPAN
RAMON TAN 18104
ANDREW S. TANENBAUM THE NETHERLANDS
DAVID TARABAR 01701
H. TAYLOR K1N 6N5 CANADA
JANET TAYLOR 75275
RICHARD N. TAYLOR 98055
WILLIAM P. TAYLOR 94550
MICHAEL TEENER 90403
DOUG TEEPLE V6X 2Z9 CANADA
PAUL R. TEETOR 48106
ROBERT TEISBERG 64108
T. D. TELFORD 94088
R. D. TENNENT K7L 3N6 CANADA
TED TENNY 13676
DANIEL THALMANN CH-1200 SWITZERLAND
R. L. THAYER 01451
DIDIER THIBAUT F-75005 FRANCE
DAVE THOMAS SW7 2AZ UNITED KINGDOM
GAY THOMAS 39762
RAYMOND E. THOMAS 20052
RICK THOMAS 20012
RON THOMAS 55425
LARS-ERIK THORELLI S-100 44 SWEDEN
EDWARD O. THORLAND 52101
LAVINE THRILLKILL 40506
BENGT THYLEN S-341 00 SWEDEN
H. F. TIBBALS DH1 3LE UNITED KINGDOM
MIKE TILLER 55441
HERVE TIREFORD CH-1211 SWITZERLAND
ALAIN TISSERANT F-54042 FRANCE
STEPHEN TITCOMB 18017
JEFFREY TOBIAS 2232 AUSTRALIA

NOBUKI TOKURA 500 JAPAN
HOWARD E. TOMPKINS 15701
SEVED TORSTENDAHL S-145 72 SWEDEN
ALFRED I. TOWELL 47401
STEVEN N. TRAPP 55421
MARTIN VERGES TRIAS 14 SPAIN
EDWIN TSE H9R 1G1 CANADA
CASEY TUBBS 32901
JOHN TUCKER 79601
W. TYLER 95060
ASHOK N. ULLAL D-7408 GERMANY
BRIAN W. UNGER T2N 1N4 CANADA
JOHN URBANSKI 55455
TOM URSIN 55440
INDULIS VALTERS 55406
J. J. VAN AMSTEL THE NETHERLANDS
G. E. VAN BEINUM THE NETHERLANDS
WARREN VAN CAMP 94041
ANDRIES VAN DAM 02912
R. P. VAN DE RIET THE NETHERLANDS
TOM VAN DER HOEVEN THE NETHERLANDS
P. J. VAN DER HOFF THE NETHERLANDS
PATRICIA VAN DERZEE 45036
H. VAN LOON THE NETHERLANDS
FRANCES L. VAN SCOY 23508
T. J. VAN WEERT 9321 THE NETHERLANDS
FERNANDO ANTONIO VANINI 13100 BRAZIL
M. W. VANNIER 40506
WILLIAM J. VASILIOU JR. 03824
JEAN VAUCHER H3C 3J7 CANADA
ROBERT D. VAVRA 55113
JAMES A. VELLENGA 55440
B. VENKATESAN T2N 1N4 CANADA
P. VERBAETEN B-3030 BELGIUM
JIM VERNON 55440
STANLEY C. VESTAL 55413
STEPHEN J VNUK 18651
EIITI WADA 113 JAPAN
KENNETH R. WADLAND 01420
KAREN WAGONER 55455
THOMAS WAGNER D-1000 GERMANY
S. P. J. WAGSTAFF LA1 4YX UNITED KINGDOM
JOHN E. WAHL 85731
M. WAITE 11740
WILLIAM M. WAITE 80309
A. R. M. WAJIB PE17 3QB UNITED KINGDOM
TERRY M. WALKER 70504
BOB WALSH 87106
LARRY WALSH 95050
PATRICK WARD H3C 3J7 CANADA
DAVID M. WARNER 80215
SCOTT K. WARREN 77098
WARREN J. WARWICK 55455
MASARU WATANABE 222 JAPAN
MARK S. WATERBURY 22152
JOE WATKINS 80302
DAVID WATT G12 8QQ UNITED KINGDOM
GEOFF WATTLES 55104
VINCENT B. WAYLAND 80303
JOHN A. WEAVER 18018
ANDERS WEBER DK-2300 DENMARK
HELLMUT WEBER D-8000 GERMANY
NEIL W. WEBER 93401
WALLY WEDEL 78712
WALTER WEHINGER D-7000 GERMANY
REINHOLD WEICKER D-8551 GERMANY

KEVIN WEILER 14850
RUTH WEINBERG ISRAEL
LEONARD H. WEINER 79409
STEVEN W. WEINGART 55113
DIETER WEISS D-6300 GERMANY
DONALD G. WEISS 78721
ANTHONY B. WELLER WC1H OAH UNITED KINGDOM
ROBERT E. WELLS 02138
JIM WELSH BT7 INN UNITED KINGDOM
JOHN WERTH 89154
JOHN P. WEST 30332
RICHARD WEST H9R 1G1 CANADA
TERRY E. WEYMOUTH 60532
WILLIAM A. WHITAKER 22209
GRAHAM J. WHITE M20 9QL UNITED KINGDOM
B. WILLIAML BN3 1RA UNITED KINGDOM
E. HAROLD WILLIAMS 95051
GEORGE H. WILLIAMS 12308
JAMES C. WILLIAMS 59717
JOHN H. WILLIAMS 14850
KENNETH L. WILLIAMS 75081
M. H. WILLIAMS 6140 SOUTH AFRICA
W. H. L. WILLIAMS N10 UNITED KINGDOM
S. WILLIAMSON N6A 5B7 CANADA
ARTHUR C. WILLIS 94086
SAM WILMOTT K1S 5G3 CANADA
ROY A. WILSKER 02154
ARDOH H. WILSON 73034
DENIS M. WILSON AB9 2UB UNITED KINGDOM
J. WILSON 13032
GARY W. WINIGER 94088
GREGORY J. WINTERHALTER 48130
NIELS WINTHER DK-2650 DENMARK
HANS-WILM WIPPERMANN D-6750 GERMANY
NIKLAUS WIRTH CH-8092 SWITZERLAND
DAVID S. WISE 47401
JOHN M. WOBUS 13210
LOUIS F. WOJNAROSKI 48109
DAVID WOLFE 97217
DARRELL L. WONDRA 55112
GORDON J. WOOD 92122
WILLIAM T. WOOD 55421
JAMES A. WOODS 94703
JOHN D. WOOLLEY 98006
DONALD L. WRIGHT 17011
ROGER P. WRIGHT RG6 2LH UNITED KINGDOM
JACOB C. Y. WU 20910
WALTER WUENSCH 13323
MARYLENE WUEST CH-1211 SWITZERLAND
NICHOLAS WYBOLT 07205
URS R. WYSS CH-1205 SWITZERLAND
CHI YIP LA1 4YX UNITED KINGDOM
SUSUMU YOSHIMURA 210 JAPAN
KENNETH YOUNG 90020
RAYMOND YOUNG 55165
STEPHEN W. YOUNG 47401
L. W. YOUNGREN 55901
GIDEON YUVAL ISRAEL
RUSSELL W. ZEARS 77550
PETER H. ZECHMEISTER 55455
H. J. ZELL NW3 UNITED KINGDOM
ANDREW HARRIS ZIMMERMAN 94086
E. C. ZIMMERMAN 44691
JOAN ZIMMERMAN 63110
KARL L. ZINN 48109
TOM ZWITTER 44139

0. Introduction

The temptation to "play" with software is more often than not too great to resist, and we succumbed. Our experience (over 12 months) with our changes has given us confidence in them, and only a few of our original mods have been retracted. We are also pleased to add that users seem more than happy to use these "extra" features, and that we often have to turn away requests for the more fanciful changes which do get proposed from time to time. We do realise that Pascal is deliberately designed to be minimal, efficiently implementable and so on. We also have some rather strong views on where the absence of certain features actually hinders many programmers, as opposed to those features which are genuinely used rarely by a small group of users (which does not include ourselves).

1. Implementation-dependent/oriented features

1.1 KRONOS-oriented changes

Necessary as always. In particular, allowing INPUT & OUTPUT to operate inter-actively under TELEX; and letting the compiler accept line numbers (sequence numbers) on source files.

1.2 Listing format

A listing format modelled after the 1972 Stanford Algol W implementation was adopted. Particularly valuable is the ability for programmers to check BEGIN-END, etc.nesting with level indicators on the left-hand side of the listing. \$-cards (lines with a '\$' in the first column) may be used to control the listing's spacing, titling, paging, etc. A more interesting \$-card is the "\$INCLUDE <filename>" which allows source text from other files to be interspersed within the main source file.

1.3 For the benefit of student users

Some more checking facilities have been added. There is a compile-time check against the assignment of a value to a for-loop control variable. The \$T+ option has the added effect of initializing the stack at run-time to 177700000000037776B's. This allows hardware checking for undefined reals and (most) pointers. An oversight in Pascal-6000-3.4. was corrected: a check is made that during a "read" a subrange-typed variable is not assigned an out-of-bounds value. The post-mortem dump was reformatted to make it a little more informative and easier to read. In addition, a procedure which invokes PMD, but then continues execution, has been added to the library ("SNAP").

1.4. Fieldlength handling

A different FL-handling discipline is implemented. The user may preset his run-time FL at compile-time by use of the (*\$FLxxxx*) control comment. This has the effect of forcing the program (at run-time) to grab an amount of core equal to its code space, plus "xxxx". The default setting is:

run-time FL = code space + size(global-data-seg) + 2000B.

These settings may, of course, always be overridden at run-time by not running in REDUCE mode.

1.5 Glitter

- a. A dayfile message has been added to provide timing information.
- b. An option (*\$Wt*) provides warnings if any of the language extensions detailed below are used by the programmer. The default setting is "on".

2. Language-oriented extensions

Most of the language "extensions" detailed below do not, we believe, run contrary to the "spirit" of Pascal. They were all implemented quite cheaply and with little or no effect on compiler efficiency. Our experience with them has vindicated them at least as far as we are concerned.

2.1 Reading strings

The standard procedure "read" is extended so that variables of type "string" (i.e. packed arrays of char) may be read. The definition is as follows: if f is a textfile, and s a string variable, then "read(f,s)" is equivalent to:

```
for i := 1 to n do
begin
s[i] := f↑ ;
if not eoln(f) then get(f)
end;
```

Note that read(f,s) never does a "get(f)" when eoln(f) is true. Hence it never causes a "readln(f)", and in addition it right-fills incompletely-read strings with blanks.

2.2 Reading and writing symbolic scalar types

It is possible in our version to read and write symbolic scalar variables (RED,GREN,BLUE;...;CAT,DOG,MOUSE;...;TRUE,FALSE etc.). This allows the language to be more generous in its treatment of scalar variables - most users complain of the absence of this feature at one time or another. An additional benefit is that the post-mortem dump can now really dump such variables.

2.3 Case-statement revamp

These extensions are arguably at odds with Pascal's "minimal language" philosophy, but turn out to be incredibly useful. They are: (i) the addition to the case-statement label list of a constant "range", and (ii) the addition of a "default" label. The first of these is surprisingly absent from standard Pascal in view of the recent addition of constant ranges in the syntax of sets (e.g. [1..9]). We have a sneaking suspicion that this was not implemented because the Pascal-6000 lexical analyser maps colon (:) and dot-dot (..) into the same internal symbol, thus making compilation of things like

```
case i of
0,2..10,12: begin ....
```

rather awkward for a one-symbol-lookahead compiler. Our (ad hoc) solution was to use the word symbol to in place of "." here. The default label is represented by else and is executed if no constant satisfies the evaluation of the case-expression. A typical example is:

```
case ch of
'A' to 'Z' : ....;
'0' to '9' : ....;
else : ....
end;
```

Articles

-3-

2.4 And for systems programmers

Two further modifications were made to the language which are not intended for use by "general-purpose programmers". They enable one to undertake systems programming from within Pascal exclusively. The extensions in this regard allow one to treat pointers as integers (and vice-versa), and to access the address of a variable. They are:

- (1) The "pointer to" operator. The use of " \uparrow " is extended so that if \langle variable \rangle has been declared thus:

```
var  $\langle$ variable $\rangle$  :  $\langle$ type $\rangle$ 
```

then the value of the expression " $\uparrow\langle$ variable \rangle " is a pointer to \langle variable \rangle , and is of type " $\uparrow\langle$ type \rangle ".

- (2) The mechanism provided by the standard functions "ord" and "chr" is extended in the following way: every type declaration allows the use of a corresponding "type-function" throughout the scope of its declaration. The type-function is of one argument, of any type; the function-result is the same argument (bit-for-bit), but with its type changed to that of the type-function.

3. In conclusion

We would like to stress that our changes to Pascal-6000 have not detracted from the overall efficiency of the compiler or its object programs. Our experience over the past year or two with these changes has definitely vindicated them, and we feel they are worth the consideration of the Pascal community at large.

(* Received (77/01/03)

Tony Gerber and Carroll Morgan are at the
Basser Dept. of Computer Science
University of Sydney *)

DISPOSING OF DISPOSE

Stephen P. Wagstaff.
University of Lancaster
England.

Abstract

This paper presents an argument for an automatic garbage collection system for dynamic variables in PASCAL, obviating the need for, and risks associated with, user-controlled de-allocation (e.g. DISPOSE). It also describes how complete protection from "dangling" pointers may be obtained.

Keywords Protection, pointer, garbage collection, dynamic variables, PASCAL.

Introduction

Consider the following PASCAL code fragment:

```
type T = record  
    x : integer;  
    .  
    .  
end;  
  
var P, Q :  $\uparrow$ T;  
    .  
    .  
new(P);  
  
Q := P;  
    .  
    .  
dispose(P);  
  
Q $\uparrow$ .X := 1;
```

The space occupied by the variable Q \uparrow has been de-allocated and yet Q has a non-nil value. This problem is mentioned in [1] and discussed in [2]. I should like to propose a solution which uses a garbage collection system based on the block structure of PASCAL. Performing the garbage collection is simple and inexpensive, and the programmer can easily arrange matters so that the space occupied by dynamic variables is not allocated for any significantly longer time than that for which the variable is actually required.

The Scope of a Dynamic Variable

Consider

```
procedure OUTER;  
  
type T = ...;  
  
var P1 :  $\uparrow$ T;  
  
procedure INNER;  
  
var P2 :  $\uparrow$ T;  
    .  
    .
```

Variables of type T cannot exist outside the scope of OUTER, and neither can pointers of type T. Thus, whenever a dynamic variable is created, the space it occupies can be maintained on a list associated with the appropriate "procedure-instance" (or in implementation terms, "stack frame").

On exit from any procedure, the whole list can then quite simply be returned to the allocation system.

The programmer can minimise his storage expenses by giving type declarations the minimum possible scope (which is good programming anyway). However, the question remains : what happens in the case where pointers reference identical structures but with differing type identifiers (and hence, possibly, differing scopes)? It seems reasonable to regard pointers as referencing types rather than structures, and whenever two types have the same structure, to regard this as a "coincidence". This gives the programmer a fine degree of control over both the lifetime and accessibility of dynamic variables. Thus, with

```

type T = ...;
  procedure OUTER;
  type T1 = t;
  var P1 :  $\uparrow$ T1;
      procedure INNER;
      type T2 = T;
      var P2 :  $\uparrow$ T2;
      .
      .
      new(P2);
  P1 := P2;

```

the distinction in the programmers mind between types T, T1 and T2 would be recognised and the final statement would be flagged as an error by the compiler, as an incompatible assignment.

Associated Protection Measures

Should it be desired to trap all possible address violations associated with pointer variables, four accompanying measures are required.

Firstly, to ensure that spurious pointer values do not exist, all pointers should be given an initial value of nil.

Secondly, (assuming that pointers are implemented as main memory addresses!), external files should not be allowed to contain components

of type pointer.

The third and fourth points concern variant records.

When dealing with access to the variant part of a record (static or dynamic), the compiler should generate code to perform a run-time check that the value of the tag-field is consistent with the variant implied (this check could perhaps be optional in general but mandatory for components of type pointer).

Finally, if variants are overlaid, there is a possibility that a dynamic change of variant would result in erroneous access to memory space beyond that occupied by the variable. This can be dealt with either by forcing all variants to be specified with NEW and disallowing any further assignments to tag-fields or by disallowing the "variant" form of NEW so that the maximum required space is always allocated (The latter would allow dynamic changes of variant).

The last two points are discussed in detail in [2].

Summary

By incorporating an automatic garbage collection system for dynamic variables in PASCAL, together with appropriate scope rules for type identifiers, the responsibility for de-allocation can be taken away from the user, and hence a class of potential address violation errors can be eliminated. Given a little programmer awareness, the cost of this added protection need not be significant. Together with the other protection measures noted all address violation errors can be wither prevented at compile time or immediately trapped at run time.

References

1. Wirth, N. Pascal Newsletter No. 5 Septcember '76 p.29
2. Fischer, C. N. and LeBlanc, R. J. "Efficient Implementation and Optimization of Run-time Checking in PASCAL". SIGPLAN Notices Vol. 12 No. 3, March '77, p 19-24

(* Received 77/05/17 *)

What is a Textfile?

The PASCAL revised report, section 6.2.4 in particular, is in serious error as to the nature of textfiles. This error arises -- or is demonstrated by -- the definition of type TEXT as FILE OF CHAR. (As a typographical convention, program fragments are presented in upper case, and the pointer operator, up-arrow, is represented by the character @). As a result of this lapse, complex special-case notions are introduced as primitive concepts. Please notice that I am not advocating a change in the language, or an abolition of existing notation: I merely propose a new, more useful, understanding and definition of the textfile notion.

First, consider the files F and G:

```
F: TEXT;
G: FILE OF FILE OF CHAR.
```

Obviously, a READ or WRITE performed on F will perform the same on G@, the "inner" file in G. Some of the auxiliary I/O constructs, however, change in a very enlightening fashion: reviewing all available literature on the semantics of PASCAL file operations, we conclude that

```
WRITELN(F) becomes PUT (G),
READLN(F) becomes BEGIN WHILE NOT EOF(G@) DO GET(G@): GET(G) END, and
EOLN(F) becomes EOF(G)! We conclude that to supply the structure
implied by WRITELN, READLN, and EOLN, a textfile is at least a file of lines, where
each line is a file of characters.
```

There is even more to a textfile: we haven't considered the PAGE statement. Let's add another declaration:

```
H: FILE OF FILE OF FILE OF CHAR.
```

Now, anywhere we used G, we can use H@: logically, however, the re-representation of READLN changes. The whole set of equivalent construct-pairs becomes, with the addition of the PAGE statement:

```
READ(F) → READ(H@@),
WRITE(F) → WRITE(H@@),
READLN(F) → BEGIN WHILE NOT EOF(H@@) DO GET(H@@):
IF NOT EOF(H) THEN
IF EOF(H@) THEN GET(H) ELSE GET(H@)
END,
```

```
WRITELN(F) → PUT(H@),
EOLN(F) → EOF(H@), and
PAGE(F) → PUT(H).
```

At this point, we have developed the structure that is necessary and sufficient to support all the standard textfile operations. As an added benefit (or is it a side effect?) we have a better appreciation of the embedded file, or file-of-file, concept. Before running off to reimplement textfiles the new way into your favorite compiler, however, let's give some thought to extensibility.

If a textfile is considered as merely a nest of files, then those implementations which would like to give access to such things as page numbers, line numbers, and vertical printer spacing ("carriage control") will have to kludge those features in as primitives: thus we would be back where we started. If, however, we consider TEXT to be predeclared as follows, we notice some nice hooks:

```
TYPE TEXT: RECORD
(*EXTERNAL FILE NAME, ETC*)
P:FILE OF RECORD
(*PAGENUMBER*)
L:FILE OF RECORD
(*LINENUMBER*)
(*SPACING*)
C:FILE OF CHAR
END
END
```

The comments point out places where interesting implementation-dependent features can reside.

William C. Price
28282 SW Mountain Rd.
West Linn, OR 97068 USA

(* Received 77/06/10 *)

WCP:pt

Generic Routines and Variable Types in PASCAL
=====

B. Austermuehl, H.-J. Hoffmann
Computer Science Department
TH Darmstadt, Germany

Abstract

Generic routines and variable types, as introduced in EL1 [1], are a means to postpone the binding time of routines and data. In this paper it is examined to what degree such features may be carried over to PASCAL without severe violation of the static type checking requirement. We conclude that generic routines fit to PASCAL, while variable types have to be subject to strong restrictions. Besides, they may be used only in connection with a special syntactic form.

Introduction

This paper is concerned with the possibility of extending PASCAL by two main features of Ben Wegbreit's language EL1 [1], namely "generic routines" and "treatment of data types as values".

In a generic routine in the sense of EL1, formal parameters are bound to a set of data types, and the type of an argument must be an element of the type set of the corresponding formal. Inside a generic different actions may be executed depending on argument types. Thus, a generic routine may be regarded as a collection of different routines for arguments of different types under a single name, i.e. as the abstraction of an operation that requires different algorithms for different input data types.

From the second feature of EL1, the treatment of types as values, follows the ability to evaluate and compute types and therefore the existence of type variables and type functions. Types are not treated statically, but in a dynamic environment. Hence, variable types, too, form abstractions, since routines are not to be bound to their data at definition time, but the structure of objects may become known only at compile or even run time.

We are well aware of the conceptual difference between EL1, which is an interpreter-based language (based on the facilities of the ECL system) where type checking may be deferred until run time, and PASCAL, where all types have to be known to the compiler. Our goal is to determine the restrictions to be posed on the EL1 features that are used to postpone the binding of procedures and data from programming time to compile or even run time by the type checking requirements of PASCAL.

In this paper, we deal with features of EL1 in PASCAL terminology, too, so we speak of types instead of modes and ignore that EL1 is an expression language, i.e. we distinguish between statements and expressions. The extended PASCAL that we investigate is referred

to as PASCAL-GVT.

A more detailed discussion (in German) of the ideas and results may be found in [2]. An experimental version of the proposed extensions is implemented based on the PASCAL P2-compiler.

Principles of the PASCAL Extension

The PASCAL design principles reliability and clarity of the language are the criteria for the extension. These principles, in the extreme, require static type checking and prohibit run time type checking of operands. In PASCAL, the compiler is able to assert the compatibility of operand types for each operation, including field selection and array subscripting. Therefore, in our extension we have to give static information about variable types to the compiler whenever we are able to. If we fail, as a consequence, there must exist interfaces to fix variable types at compile time. At those interfaces, however, we have to admit dynamic type checks to ensure the validity of the fixing at run time, and there type-dependent run time errors may occur if the run time instance of the type is not in the set of allowed types. These interfaces must be the only points where dynamic checks are required, and the user must be aware of run time errors only at those points.

Union Types and the Generic Form in EL1

In EL1, we find union types. The meaning of "union", there, is only the postponement of type choice, i.e. at run time each object and variable has during its lifetime a definite and unchangeable type. In particular, the definite type of a union-typed parameter is determined by the argument type and cannot be changed subsequent to creation.

A generic routine has parameters of union types. Inside its body, the alternatives of the union types may be singled out by means of the "generic form" that resembles a case statement in PASCAL. A generic form consists of several alternative branches and a header naming the parameters the types of which are to be fixed. The right hand sides of the branches are statements, the left hand sides are formed of type-lists (one entry for each generic parameter) and additional (optional) predicates. In the type-lists, alternatives (or unions of alternatives) of the parameters' union types are specified, to which the types of the corresponding parameters are fixed inside the branches.

The appropriate branch for a given combination of argument types may be selected at compile time, if all types in one of the type-lists "cover" (for definition of cover see [1], p.250) the corresponding argument types. Since argument types may be unions (if arguments are parameters of other routines), an argument type may be only partially covered ([1], p.250) by a type-list element, i.e. some alternatives of the argument type are not alternatives of the type-list element, while others are. Then the compiler is not able to decide whether the definite run time type fits or not, and must

generate a run time test. This, holds, too, if the additional predicate is not evaluatable at compile time.

The PASCAL-GVT Generic Extension

In Wegbreit's ECL system there exists a compiler as well as an interpreter, both fully compatible. Each may call the other as a subroutine. Therefore the compiler is able to evaluate parts of a compilation unit (routine) and to use the value instead of the form. So predicates of a generic form may be evaluated by the compiler and a compile time selection may be done. In the generic form carried over to PASCAL-GVT, predicates are not allowed. There are two reasons for this decision:

1. We have no interpreter in our system. Therefore, predicates are not evaluatable at compile-time, and a run time selection is necessary for each call of a generic routine with predicates, even if the covering of all argument types is asserted. So the number of possible run time errors increases.
2. By design, a decision in a generic form is a decision depending on the types, not on the values of the arguments. Accordingly, predicates in a generic form should be predicates on types only. The type classes that are defined implicitly by predicates, however, do not have such a specific structure that the compiler is able to handle them (e.g. all one-dimensional arrays). The compiler will not be able to determine any component statically. Therefore a static type checking will not be possible inside the branch and so the advantages of the generic form will be lost.

Union types in the sense of EL1, however, fit to the requirements of PASCAL. First, the structure of each alternative is known to the compiler, so there is no difference to a normal PASCAL type after the selection of one single alternative in a generic form. Second, the type constancy during the lifetime of a union-typed parameter allows a stack implementation of such parameters. When the procedure is entered, the definite type with its length becomes known (Since this happens at run time, run time type descriptors have to be generated by the compiler). Since the length is unchangeable, an address on the stack may be computed for each union-typed parameter and the argument values may be copied. Access to the parameter values is indirect via the compile-time-computable local address, where a pointer to the run-time-computable real address is to be stored. Since we are able to put union-typed parameters on the stack as opposed to the PASCAL heap (where flexible-length parameters would have to be put), there are no problems with RELEASE-commands of the user. So union-typed parameters do not mess up PASCAL's storage management scheme.

The demand for static type checking implies that each generic parameter is fixed to a defined, compiler-known type (including a union type) at the entry into a generic branch. If that type is a union, operations on the parameter are restricted to assignments and equality tests inside the branch, since static type checking requires fully fixed operand types for any other operation. This restriction forces a programmer to write down repeatedly similar branches for

similar, but different types (e.g. array of integer vs. array of real). The PASCAL convention of identifying a type by its name, not by its structure, disallows us to define arrays of unions and so to handle similar structures in a single branch, since we then had to have variables of type array of unions. Union-typed variables, however, will not be allowed, since (a) each variable must have an unchangeable type (there is no chance of a postponement of type choice as with parameters) and (b) union-typed variables would impose further need for run time type checks. So the disadvantage of multiple writing down can not be remedied by using unions. We will see later that there is a slight improvement by use of variable types.

With the given restrictions, the generic form is easily transferable to PASCAL. Thus, a PASCAL-GVT procedure body may be either a normal PASCAL procedure or a generic form. The only violation of static type checking by the use of a generic form may occur if only partial covering is given at compile time and thus a run time check is needed for branch selection. If at run time the combination of argument types does not fit to any of the branches, a type-dependent run time error will result, violating the principle of static type checking. This, however, only occurs at a well-defined interface, where the user may expect it. Besides, the number of run time branch selections will normally be small.

Example:

```

type INTARRAY = array [..] of INTEGER;
      UNION    = oneof (INTEGER, INTARRAY);
procedure P (PARA: UNION; PARB: oneof (CHAR, INTEGER));
  generic (PARA, PARB) of
    [INTARRAY, CHAR] : begin ... end;
    [INTEGER, INTEGER] : begin ... end;
end;

```

Types as Values

In PASCAL, types are static descriptions of the structure of a class of objects. In EL1, however, type generators are callable functions and deliver a type value. The compiler evaluates such generators under assistance of the interpreter. Consequently, user-defined type functions as well as type variables are permitted. If a type function is not evaluatable at compile time, a call to the interpreter is generated, i.e. type checking is delegated to run time. Type variables may be "frozen", i.e. evaluated in an interpreted environment of a compilation step, and their value may be used as a type constant in the compilation unit. "Unfrozen" type variables, again, require type checking at run time. The facilities of evaluating type functions and freezing type variables enables the compiler to nail down variable types. The binding of routines to types is transferred from programming time to compile time, but an interpreted environment has to be involved in this process.

In PASCAL, we do not have the facility to freeze variables, since there is no interpreted environment available. A variable type at programming time remains variable at compile time (although being invariable at run time). Static type checking, however, requires a wide range of constancy for type variables, since these have to act as representatives of the run time types: Two variables declared by the same type variable must have the same run time type, since the compiler can check their types only by means of the name of the type variable. As a consequence, a type variable in PASCAL-GVT must not be assigned a value in any other than in its declaring procedure; otherwise assignments to a type variable between declarations of two variables in hierarchically ordered procedures might result in different run time types of those variables in spite of the compile time assumption that they have the same type.

In contrast to EL1, where type checking of variable-typed variables and, if need be, insertion of operations for conversion of their values, may be done at run time, the static type checking mechanism of PASCAL requires full compile time checking of operand types for every operation other than assignment or equality check (where no specific types, but only equality of types is required). Therefore we have to introduce a facility to nail down types of variable-typed variables, similar to that we have for nailing down types of generic parameters. We define in PASCAL-GVT a new syntactic form, called "generic expression", which looks like the original generic form used as the body of a generic routine, but has expressions instead of statements as its branches. The "parameters" of a generic expression name the variable-typed variables, the types of which are fixed to allow operations on them in the branches. By this, static type checking remains possible in spite of allowing variables to be declared with (at compile time) variable types. Here, too, we have an interface, where type-dependent run time errors may be expected, if the definite combination of types at run time is not covered by one of the type-lists in the generic form. Only variables or parameters can have a variable type, since there are no operations on variable types available. Thus, a generic expression must have a unique invariable type, i.e. all branches must have the same resulting type, which must be invariable. There is only one exception to this uniqueness requirement: Assignment of an invariable-typed value to a variable-typed variable is done by use of the generic expression, too. Then the left hand variable is entered into the "parameter"-list of the generic expression forming the right hand side of the assignment statement, and each branch of the generic expression must have that result type, to which the left hand variable is fixed in that branch.

With these restrictions and syntactical aids type variables may be handled in a static type checking environment. Besides variables, however, we have to consider type functions and other type-valued expressions.

The two main advantages of type functions are the ability to define
(1) recursive data structures
(2) similar structures over different base types by one definition. As to recursion, the static type checking mechanism does not allow such a dynamic structuring, since the compiler is not able to determine the depth of the recursion statically and so cannot provide access to any component. This implies that one cannot define operations on objects declared by recursive type functions. So recursion

must be forbidden. In addition, even without recursion, type functions are not compile time evaluable because of the existence of parameters and globals. Since the compiler is unable to determine the structure of any function-defined type, those types are obviously meaningless and thus are forbidden in PASCAL-GVT.

As to other type-valued expressions, we must consider the above remark on recursion. Here the same holds for iteration. If we allow complex type-valued expressions, it will always be possible to assign: TVAR := array [...] of TVAR, which structure will not be recognizable statically. So we must forbid, too, complex type-valued expressions and allow only type variables and type constants to appear on the right hand side of an assignment statement.

Structures Over variable Types

Structures (arrays, records) over unions cannot be defined, since type union is only allowed for parameter specification; parameters in PASCAL, however, must be specified not by a type structure, but by a type name, and compatibility of actual and formal parameters is determined only by equality of the type name, not by similarity of structure (or a certain kind of covering, if unions were involved). Since we do not allow variables of union type, there cannot exist any compatible argument for a formal of type "structure of union".

Structures over type variables, however, are meaningful, since type variables may be used in any context where other types may be used. Although its overall structure is known to the compiler, the entire variable of such a type is considered a variable-typed variable insofar as its real address is determined only at run time and access is indirect. Records are physically restructured to shift variable-typed fields to the end and so to give information about the relative address of the fixed-typed fields to the compiler. Both arrays and records may be stored continuously without any use of internal pointers and thus copying may be done without examining substructures. Since the compiler has information about the overall structure of such an array or record, only component types; if variable, have to be fixed in a generic expression.

Example:

```
var TVAR : MODE;
procedure P (TPAR: MODE);
  type VARRAY = array [...] of TPAR;
  vRECORD = record VFELD: TVAR;
                VARRFELD: VARRAY;
                FIXFELD: INTEGER;
end;
```

```
var VV: TVAR; I,J: INTEGER;
begin
  I := generic (VV) of
    [INTEGER] : VV;
    [REAL]    : TRUNC (VV)
  end + J; ...
end;
```

Conclusions -----

Type variability is a means of separating data and programs. In ELI, it is a meaningful instrument, since compile and run time are homogeneous in that compile time of routines and run time of type evaluation may be the same. Thus, library routines may be written data independent and their types evaluated in an interpreted environment of their compilation, so achieving an efficient and type-secure object code in spite of data independence, since type checking may be done at compile time.

In PASCAL, where compile and run time are strictly separated, the static type checking mechanism imposes strong restrictions onto the use of type variables, making them constant in hierarchically sub-ordered procedures. We are not able to extend PASCAL's type scheme by iterated types and type functions, which may be regarded as classes of types, since these two features require dynamic treatment. Thus the type definition part remains the only place where types may be constructed.

The use of type variables in PASCAL-GVT is along two axes:

- (1) directly for declaration; this is an extension of the generic parameter mechanism to variables;
- (2) as base type of a structure; we may look at similar structures under one single type, if we represent the different base types by a type variable. However, if we want to abstract from the base type of a structure and use a type variable, we have to copy each instance of the structure with invariable base type to an instance of the structure with variable base type, since, even if we were able to fix that variable base type in a generic expression to the right invariable base type, the PASCAL convention of considering two equal structures as different types enforces copying. Such a usage may be meaningful, however, in the context of generic routines to avoid multiple writing down of similar branches. Then, we may enclose the generic in a kind of pseudo-procedure (generic, too), where the copying is done, and the generic itself may deal only with one structure over a variable base type. Especially for library routines one must consider the trade off between the copy overhead on the one hand and the possibility to use one name for one operation independently of types on the other hand.

Providing more type variability in PASCAL-GVT would have violated the above-mentioned principles of our extension.

The feature of the generic routine fits much better to PASCAL than that of type variability, since we only have to exclude the grouping of types by predicates to retain static type checking. This feature

is a meaningful extension in the direction of functional abstraction, since we now are able to denote one operation by one name with no regard to different data types with possibly different algorithms to implement the operation.

Acknowledgement -----

We thankfully acknowledge the advice of Hans Kron in improving our English.

References -----

- [1] Weopreit, B.: The Treatment of Data Types in ELI, Comm. ACM, 17(1971), 5, p.251-264
- [2] Austermuehl, B.: Zur Verwendung typabhaengiger Prozeduren und variabler Typen in PASCAL, Diploma Thesis, Computer Science Department, TH Darmstadt, File Nr. PU S055, March 1977, 84 pages

(* Received 77/08/05 *)



The University of Tasmania

Postal Address: Box 252 C, G.P.O., Hobart, Tasmania 7001, Australia

Telephone: 23 0561. Cables 'Tasuni' Telex: 58150

10th May, 1977

Dear Andy,

All is forgiven. Let's forget the past and get on with work.

Distribution Centre for Australasia: I await your suggestions. I think we can act for Australia, New Zealand and Papua/New Guinea, but Japan is probably nearer the U.S. than us.

Standards: great news! desperately needed! I am on tenterhooks!

CDC-bias, etc: May I take some time to talk around the points you make on files, program, and CDC-bias? I don't expect a reply since you are busy, but I'd like to try to convince you that I have some points here.

Your point (1): files as sequential access structures: I totally agree. Sequential files are useful; they are data objects; they are needed for the purpose you cite. What I try to say is that files are not full PASCAL-variables in that their usage in array of file or record of file, or file of file or in expressions, is undefined. They partake of few properties of full variables; about as many as procedure or function-names for example. I express a sadness that the opportunity has been lost of expressing this well in the PASCAL language.

Your point (2): arrays as random-access. I only partly agree. Sure a slow array could be implemented as a random-access file, but not all random-access files can be implemented as slow arrays. Unless you are willing to throw away PASCAL's strong typing and admit truly dynamic sized arrays. The point being that even a random-access file is a sequence of variable length. PASCAL arrays are always of fixed pre-determined length. I emphasize that random-access is a property of the access, not of the file (though CDC's standard implementation of files disguises this). Think abstractly. So I've no objection to slow arrays; they're just not equivalent to random-access files.

Your point (3): program heading. I can't see FORTRAN's identical program heading as a 'coincidence', I'm afraid. Your subsequent argument is a pragmatic one for collecting all machine-dependent information at a central place. A good practical point. The counter-argument is that based on a feeling for structuring. One of the precepts of structured programming is that information relevant to a structure should appear in one place (the point of decision) and that only. So this urges me to collect all file data at one point: the type or var declaration for it.

Besides this, several nasties creep in if the information is collected in the program heading. CDC PASCAL crudely restricts 'permanent' files to ones declared in the outermost block; if this restriction is lifted (an obvious step leading to better structuring of subprograms and their scopes), then name confusion may arise (two files called INPUT?). In addition, the program heading could become very large for complex programs, and a useful facility has been pre-empted (I mean the facility to activate a PASCAL program with genuine parameters). Now much of this is non-standard, but I hope it better illuminates what I mean when I say there is subtle CDC influence (and I mean subtle, not blatant: Wirth is a good designer).

Open Forum for Members

It may interest you to know too that it is quite possible to leave attribute information out of a Burroughs B6700 PASCAL program, and to supply it all in the Work-Flow Language (control program? JCL?) to be bound in at opening time. This rips machine- and run-dependent information right out of the PASCAL program, but would be unbearably tedious if done for every PASCAL file or run. This would be self-evident to any B6700 programmer reading our documentation, but mightn't be to others.

Your point (4): Remember B6700s and B5500s have been around a fair while: they are also today's machines. My point here is that the CDC 6000s are just about an extreme in simplicity of (i) architecture, and (ii) operating system. It is quite natural that troubles will arise at the other ends of the spectrum. It is also quite natural that systems with reasonable affinity will prove to be easy to implement PASCAL on, for the assumptions are the same. Actually the CDC conventions and operating systems are more troublesome than its architecture which is a triumph of monolithic simplicity. Examples of the (again subtle) effects are PASCAL's nonexistent attitude to interactivity, the lack of read/write scalars, and so on. Quite a long list of regrettable influences could be compiled. Many of them do not directly lead to implementation difficulty, but show up as a less-than-perfect construct. I grieve, but can do nothing about it.

Parenthetically, over the last 10 years I have had quite intimate contact with all the following systems: IBM1130, IBM360 & 370, IBM7040, IBM1401, CDC6600 (Scope) & Cyber 72, (Kronos) CDC1700, English-Electric KDF9, Elliott 503, PDP-8, PDP-11 (20,40,45 & 70), Burroughs B1700, B6700, Univac 1108, ICL 1900, Decsystem 10 (KI10), Varian and Interdata. I could add in more pre-1967 machines. I think I have managed to develop a connoisseur's nose for machines and their influences...

For's of persuasive segment.

I am interested to know that the non-academic world in the U.S. is interested in PASCAL. I'd love to know how many of those PUGN subscribers are (i) mini-computer firms, (ii) mainframe operators, (iii) software houses, or (iv) just interested individuals? It'd be interesting, yes? Thank you too for the Minnetota breakdown of usage. 5 - 10% usage rate in number of runs is indeed good progress.

Our first-year course will switch over entirely to PASCAL next academic year (a first for reactionary Australia) now our compiler is operational, and I will put on a "What's in PASCAL for you" course later this year for the general academic population. It will be interesting, as our FORTRAN usage at Tasmania has never been dominant due to some complex historical constraints. Switching Algolers into Pascallers is easier in one way, but convincing them of the merit of the switch is more difficult!

We are also organizing through Burroughs to run our compiler on a B7700 system, and probably a dual-processor B6700. If I can get to any others of the range (eg the new 6800) I'll try them too. We aim to thrash it on re-entrancy and any possible model-dependent features. Hardware documentation is very poor in Burroughs. And needed.

My best wishes. I hope the workload doesn't get you down.

Yours sincerely,

Open Forum for Members

TELEPHONE: 692 1122



BASSER DEPARTMENT OF COMPUTER SCIENCE

School of Physics (Building A28),
University of Sydney, N.S.W. 2006

24 May 1977

Andrew B. Mickel,
Editor, Pascal Newsletter,
Computer Centre, 227 Exp Engr,
University of Minnesota,
MINNEAPOLIS MN 55455
UNITED STATES OF AMERICA

Dear Andy,

Thanks for mailing my newsletter #8 air mail - as Arthur Sale points out, three months' lag is unacceptable. (It's continually annoying to receive conference notices and Calls for Papers from ACM weeks after the event.)

I am mainly writing to air a grudge. At the beginning of this year we sent you a short article dealing with changes to the Pascal-6000 compiler we had made. Although you no doubt have good reasons for not publishing any word of these changes, we are at a loss to understand why you subsequently publish proposals for changes, when we have actually implemented these changes and can attest to their worth or otherwise. We have not attempted, nor do we wish to attempt to implement features such as dynamic arrays and initializations as it is obvious that a lot of people are debating several alternative proposals. With one exception, what we have implemented entails simply weakening restrictions already present in the language. They are:

- (1) reading "packed arrays of char";
- (2) reading and writing symbolic scalar types;
- (3) allowing a "range" of labels for case-statement labels;
- (4) providing an "else" - clause for case-statements (this is the exception);
- (5) allowing functions to be of any type (except file). (This is a new one, only just implemented.)

The debate on the suitability of the else-clause in the case-statement seems to be a rather overworked one, reminiscent of the dangling-else debate for if-statements. Wirth talks of convenience as opposed to necessity in this context in PUGN #8, but I cannot help feeling that a lot of the language would disappear if this criterion was applied to the whole language (e.g. the with-statement, if-then-else is really only "case <expr> of true: ...; false: ... end", record variants). Most people here seem to be perfectly happy about using the else-clause - they include people who one could genuinely call "good" programmers.

Our other under-the-table extensions (type-functions which relax type-checking (cf. Richmond's transfer functions in PUGN #8) and the address-of operator) illustrate more closely our ideas on why we feel no regret at "extending" Pascal. These systems-oriented changes were made for purely selfish reasons: some of us wanted to carry out systems programming entirely in Pascal, despite the fact that Pascal was not designed as such. The point is that programming in "extended Pascal" is much more satisfying than programming in an assembly language. Our concern is therefore that we should make Pascal more useful than it really is, simply because the alternatives available (on the CDC Cyber) are so abhorrent. In our minds, we always maintain the distinction between "Sydney" and "Standard" Pascal, and so does the compiler - it will, unless directed otherwise, flag every use of a Sydney-implemented extension.

Surely then, our efforts should not be concentrated on standardizing Pascal at a time when Pascal is beginning to show signs of age. There are non-trivial deficiencies in Pascal which are being attacked in more recent languages (Euclid, CLU, Alphard et al). Pascal might better serve therefore as a testbed in which improved ideas may be evaluated. I have this recurring nightmare: I'm reading the UTOPIA 84 Newsletter and they're complaining about all these old-fashioned people in industry and academia who won't move from Pascal to UTOPIA 84 because of the large financial investment tied up in Pascal software ... Pascal's role is not, I believe, to serve as the next important widely-used general-purpose language. It is a credit to its design that although it wasn't designed as such, it has nearly become such. Let's keep Pascal in its proper perspective, please!

Finally, we would be grateful if you would give our modifications some publicity. They are actually implemented, they work, our experience with them (over a year) is positive, and the implementation overhead incurred is definitely acceptable.

Keep grinting,

(Tony Gerber)

(* Editor's Note: In a reply dated 77/06/07, I stated:

"I just received your letter, Tony, yesterday. John and I owe you several big apologies. I found out shortly after reading your 24 May letter that there was material on John's deck which I had never seen: a listing and some correspondence. I hope you don't get the idea that we go out of our way to hassle Australian PUG members!

"The trouble with an else on case is that it catches things you don't plan for as well as the things you do, and you can't distinguish among them. Separate compilation is a good thing. Your include feature or something like it will wind up in Release 3 [of Pascal-6000].

"Regarding Utopia 84, i've had the same thoughts, but we haven't even gotten rid of Fortran yet, and once that precedent is set, getting rid of Pascal when its time comes will be easier. No, I don't think you comprehend the politics of getting a language like Pascal widespread. So yes, Pascal's role is to be the next widely used general purpose language, and any attempts by you or I are going to fail; it simply has too much merit on its own to stop it. Languages like Euclid, Alphard, and CLU are not general purpose and therein is the rub! Besides they needlessly adopted different syntax for similar semantic constructs.

"Thank you again for all you have done...."

"P.S. What does "grinting" mean?" *)

ANPA RESEARCH INSTITUTE

1350 Sullivan Trail, P.O. Box 598
Easton, Pennsylvania 18042
(215) 253-6155
June 1, 1977

Mr. Andy Mickel
Pascal User's Group

Dear Andy,

Each Newsletter seems to be getting better. Number 8 is truly high quality both in presentation and content.

I have given lots of thought to the question of PASCAL software tools. There is no question that there exists a great need for the collection, review, and distribution of shareable software. We need to do this within PUG so that we can preserve our independence while increasing our scope.

Up until now I have collected and installed at Lehigh University a number of useful programs. I've used those to trade to get others. The problems of wider distribution have me truly worried. At Lehigh our antiquated 7 track drives and strange 63 character set make machine compatibility problems (via magnetic tape) almost insurmountable. I've even had five crates of cards (50,000) punched to import some software. Postage and other distribution costs have been paid out of my own pocket. There has got to be a better way - here's my suggestion:

I recommend that PUG Newsletter allocate a number of pages in each issue to the publication in source form of generally useful PASCAL programs. Both software tools and pedagogic examples could be published (program listings, documentation, designer commentary, and reviews) in "The Programmer's Corner" of the Newsletter. I could use my facilities at ANPA to produce camera ready copy of this material. Local non-standard usage could be clarified in text descriptions. Constructive criticism from members would be invited.

"The Programmer's Corner" has other benefits besides facilitating the sharing of programs. Good technique and compliance to standards would be encouraged. A new outlet for programmer/user ideas would be opened. Software tool distribution would be furthered by encouraging implementers/distributors to include the published programs on PASCAL distribution tapes. The tools would also form a good test base for implementors.

My personal interest in this stems from my great disappointment in the dropping of the Algorithms Section from Communications of the ACM. "The Programmer's Corner" offers a way to restore program and algorithm design to its rightful preeminent place in our profession.

I can see two disadvantages to this approach. First, it will take time before a thorough set of tools is published and, secondly, valuable

space in the already crowded Newsletter will be used. To the first objection I respond that a continually growing, universally available software set offers significant advantages. To the second I offer the following method for increasing the available space in the Newsletter. First, we set up an abbreviation scheme. E.g., SA = slow arrays, DA = dynamic arrays, DF = direct access files, FIO = formatted input/output, etc. Letters from dissidents could then be tightly compressed for publication. "W/O FIO & DF, PU & WNR: MH/ABT" could be the concise representation of "Dear Andy, Without formatted I/O and direct access files, PASCAL is useless and will never replace FORTRAN. ..."

Incidentally my own experience over the last five years with students who have learned to program using PASCAL is that if they go into a non-PASCAL environment, they quickly become an importer or implementer of PASCAL. In their minds, neither FORTRAN, COBOL, nor PL/I will ever replace PASCAL.

One final word about "The Programmer's Corner" idea. It seems to me that as our organization matures member interest will shift from implementation discussions to applications. I, therefore, look for the Newsletter to soon begin reflecting this change and membership to grow even faster because of it.

Sincerely,



Richard J. Cichelli
Research Manager
Computer Applications Department

Co-director, Computer Science Group
Department of Mathematics
Lehigh University

(* Editor's Note: I reacted negatively to this proposal at first, especially because of space considerations and who would judge what programs would be published. But Rich phoned me and talked me into it - provided he edit the section; he's really right that we should involve the interest of users much more than we have. It's been mostly implementors so far. Beginning with next issue (Pascal News #11), we should have some programs (mostly software tools) to print. See also the second page of Mike Ball's letter for his views on portable program exchange. *)

16 June 1977

Mr Andy Mickel
University Computer Center
University of Minnesota
Minneapolis, MN 55455

Dear Andy

Enclosed is a check for my membership renewal for the next year.
Please change my address to:

Michael S. Ball
Code 632
Naval Ocean Systems Center
San Diego, CA 92152

This is a change of address due to a local reorganization.

I am currently hard at work on the concurrent and sequential Pascal compilers for the Interdata 8/32. The past few months were spent on the design of the Kernel and compiler changes, so I had very little time to worry about anything else. We have an initial operation date of 15 July, so things are coming to a head. It will not be available for distribution for at least several months.

The Univac 1100 compiler is seeing increasing local use, and there are 24 known copies in the field. There are 11 at universities, 4 at government installations, and the rest in industry. I have no data on the amount of use except at a few of the installations.

I was interested in the "standard extensions" to Pascal. I would like to suggest that these be limited to those which can be translated easily into equivalent standard Pascal. For instance, Dynamic arrays can be used in ways which are much more difficult to translate than can parameter arrays. Other extensions should be limited to additional standard procedures, and perhaps minor changes to highly system dependent actions such as file declarations. This limitation should increase program portability, while at the same time providing the convenience and added efficiency which seems to be the motivation behind most of the suggested improvements.

Along that line, I would like to suggest that a standard syntax be specified for external and other language subroutine declarations. The implementation is of course highly machine dependent, but a uniform syntax would ease transfer pains.

While on the subject of extensions, I heard from Jim Shores that you have a proposed extension for initialization which Wirth liked. If this is in shape for use, I would like a copy of this, since the initialization of tables is an area of considerable inefficiency in many programs.

I would also like to urge the creation of a standard editing procedure and distribution format for Pascal programs, since in my experience much of the trouble in transporting programs comes in incorporating corrections, and then later in merging corrections with the inevitable local modifications. Something similar to Bell Lab's source code control system might provide a reasonable approach. The first job, of course, is to decide what features are needed, and what can be implemented in a portable manner. I would like to suggest the following list of features as a starting point.

1. The standard should include the full ASCII character set, but all programs should be case independent, so that they can be translated to an upper-case subset without harm.
2. Card length restrictions should be followed, since many operating systems work in card images. Serial numbers should be optional.
3. Corrections should include enough redundancy (perhaps an alphabetic checksum of some sort) so that corrections which are transmitted on paper have a reasonable chance of surviving the keypunching experience.
4. The system should provide the ability to add local changes with the local editor, then merge these corrections with new corrections from the distributor (a down-date procedure).
5. The programs which implement this should be implementable with the subset Pascals which are frequently the first step in a bootstrap. In particular, as few files as possible should be used.

More specific suggestions are easy to generate.

We are intending to implement some form of source code control system for our own use, and if there is interest in this, we will take the extra trouble to make it portable and generally useful. Let's hear from others on the subject. I am sure that I am not the only one tired of simulating other systems' editors by hand.

yours,

Mike Ball

☒ 1455 de Maisonneuve Blvd. West
Montreal, Quebec H3G 1M8

☐ 7141 Sherbrooke Street West
Montreal, Quebec H4B 1R6

Tel. 514-879-4251



June 16, 1977

PASCAL User's Group
c/o Andy Mickel
UCC:227 Exp. Engr.
University of Minnesota
Minneapolis, MN 55455

Dear Andy,

The merit of PASCAL is its simplicity. It is reasonable to expect a competent PASCAL programmer to correctly predict the effect of any well-constructed PASCAL statement, which is more than can be said of certain other programming languages. In attempting to standardize PASCAL we should attempt to tidy up loose ends, not to incorporate fancy features. When we have to extend the language, we should preserve the spirit of the initial design.

Everyone has their own ideas about what the most important defects of PASCAL are. My own pet grievance is the READ statement used to perform automatic conversion from character string to INTEGER or REAL. No user will accept a program which collapses when it encounters an unexpected character in the input stream, and no programmer wants to incorporate conversion procedures into every program he writes. Therefore, READ must have an error exit, and the problem is how to provide it in a clean way. The solution should be compatible with the existing READ, so that simple-minded conversion is available for toy programs and novice programmers.

I tentatively propose the following: the READ statement should accept an actual parameter whose type is RECORD. The record must contain two fields, one scalar or REAL, and the other BOOLEAN. For example:

```
VAR ITEM: RECORD
    DATUM: SCALARTYPE;
    FOUND: BOOLEAN
END;
```

After executing READ(FILENAME,ITEM) either ITEM.FOUND = TRUE and ITEM.DATUM has the appropriate value or ITEM.FOUND = FALSE and ITEM.DATUM is undefined. In the first case, the file pointer will have been advanced past the item read, and in the second the file pointer will be unchanged, except that leading blanks and blank lines will have been skipped. If we have formatted input, then the pointer would be advanced over the indicated field width in either case, and the program would not get a second chance to read the item. If SCLARTYPE is INTEGER and the input stream contains

^A
the string

123J5

then the first READ would find 123 and the next would fail, which might be confusing. We could insist that the item be followed by a blank, but this has obvious problems too. For example, a program reading expressions would accept 123 +456 but not 123+ 456.

The method extends naturally to user defined scalars and (note!) subranges. This is important, because I think that it would be pointless to extend PASCAL in such a way that scalars could be read but entering FLASE instead of FALSE causes a fatal run-time error.

The programmer still has to provide an error recovery routine. For an interactive program, there is no problem: issue a diagnostic and cue for corrected data. For a batch program the easy way out is to READLN, leaving the user to spot further errors on the same line. In a specific application, however, it is often possible to design a more sophisticated error recovery procedure which takes reasonably intelligent action.

The PASCAL Newsletter is doing a fine job. Keep it up!

Sincerely,
Peter Grogono
Peter Grogono



THE UNIVERSITY OF TEXAS AT AUSTIN
AUSTIN, TEXAS 78712

Computation Center
512/471-7242

June 24, 1977

Dear Andy:

Since it's renewal time, I thought it would be appropriate to bring you up to date on PASCAL related happenings here at UT.

The best news is that we finally got confirmation that the new version of DEC-10 PASCAL has in fact made it to the U.S. and DECUS. This confirmation came in the form of a copy of the files for a test installation from Carl Perkins of DEC to whom we had supplied the old version of PASCAL. He informed us that he would be the official DECUS submittor. We have the new version up and in reasonable shape. The biggest problem with it is that all programs that ran with the old version have to be changed.

On the Control Data side of things, Wilhelm Burger has left UT to take a job in Washington, D.C. Tom Keel of our staff is now looking after the PASCAL system. We are looking at installing your efficiency mods from the PASCAL Newsletter #5. Another programmer made a good start on a PASCAL interactive debugger this past semester.

UNIVERSITY COMPUTING CENTER
UNIVERSITY OF COLORADO
BOULDER, COLORADO 80309

22 July 1977

Let me turn now to the question of standardization which has been debated so thoroughly in the PN issues of the past year. It appears from the information in PN #8 that the U. S. standardization process is not well understood. I enclose a copy of a presentation made at VIM-23 by Meredith Speers which describes the process quite well. A careful review of the process will reveal that it is an extremely expensive and time consuming process. The effort in shepherding the proposal for a standard through SPARC is considerable. I would estimate that it would take a year and about \$35,000. counting personnel support to get a technical committee set up. A conscientious effort could shorten this time frame, but I doubt it. Once the technical committee is established I suspect at least 12 to 18 months will be required to formulate an acceptable standard. Assuming quarterly meetings, this translates to 4 to 6 meetings. This estimate assumes a 20 to 25 person technical committee. As you point out in PN #8, the technical committee is critical to the formulation of a standard and I doubt that the canvas approach will work with PASCAL given the acknowledged weak spots in both the Report and Manual.

The technical committee under X3 rules is a volunteer organization with strong continuing attendance requirements to assure a body of expertise behind the proposed standard. Given the strong interest in a standard expressed within PUG, I would expect a technical committee of 20 to 25 sufficiently committed volunteers could produce a standard in 12 to 18 months. The most difficult part, as you point out, would be to control extensions to the language.

If the effort through BSI does in fact result in a proposed ISO standard, then SPARC will almost certainly set up an X3 PASCAL technical committee. Consequently, I think that a U.S. X3J committee for PASCAL is probably inevitable and PUG should probably take the leadership in establishing such a committee.

Enclosed is my renewal check. Keep up the good work!

Sincerely,



Waldo M. Wedel, Manager
Programming Services

WMW:mp

Enclosures

(* Editor's Note: Wally is a member of the ANSI X3J2 Basic Standards Committee. I replied to Wally in a quick note dated 77/06/27 that: "I guess the point is that we don't want an ANSI standard that differs from an ISO standard. We are not going to go for an ANSI standard because it takes too much time and energy." I might now add that after there is an ISO standard, ANSI should adopt it as a matter of course. *)

Mr. Andy Mickel
University Computer Center
227 Exp. Engineering Building
University of Minnesota
Minneapolis, MN 55455

Dear Andy:

Please find enclosed my membership for the next academic year for the Pascal User's Group. And congratulations to you, John, and the others for producing four newsletters of exceptionally high quality. Keep up the good work.

After reading Newsletter #8 and listening to CDC present their future plans, I agree with your position that now is the time to formalize the definition of Standard Pascal by cleaning up the semantic definition and making relatively few extensions to the syntax. The important syntactical changes should include dynamic arrays, value initialization (including arrays and records), strict procedure parameter type checking and case statement alternative.

I don't expect to see the bulk of my proposals in Newsletter #8 implemented in Standard Pascal. I believe the best route for implementing extensions to PASCAL is to build a pre-processor (written in Standard Pascal) to translate extended Pascal to Standard Pascal. Such a processor is truly portable and essentially changes the compiler into a two-pass system.

Our distribution mechanism is operating efficiently with less than one week turnaround (except for vacations). Karin Bruce and Michele Dowd are doing a good job. I've enclosed some of our recently developed material. Karin feels it would be more expedient to drop the option of letting the buyer supply the tape and incorporate the cost of a tape into the minimum cost. I concur with this idea. Do you have an opinion on this change?

* * * * *

Sincerely,



George H. Richmond

ADP Network Services, Inc.
175 Jackson Plaza
Ann Arbor, Michigan 48106
(313) 769-6800

July 28, 1977

Mr. Andy Mickel
Editor, Pascal User's Group
University Computer Center
227 Experimental Engineering Bldg.
Minneapolis, Minnesota 55455

Dear Andy:

I thought your readers might like to know that we have an interesting PASCAL project in progress and that there are PASCAL related positions available here in Ann Arbor.

ADP Network Services currently operates more than fifteen DECSYSTEM-10's on an international communications network and we have the need to develop a systems implementation language to support language, monitor and other software development for DEC-10's and other hardware that may be attached to our network as our company grows. PASCAL has been chosen as the base for this language. We have embarked on a joint project with Al Kortesoja of Manufacturing Data Systems Inc., also of Ann Arbor, to develop language and code generation features that will provide us with a general implementation language that will generate good code for a variety of machines.

We began with the DECSYSTEM-10 compiler developed by H. Nagel of the University of Hamburg and are modifying it to include: random IO facilities; flexible length arrays; constant arrays and records; an exception handling facility; functions which return arrays, records and sets; and STRING handling. Through this we have endeavored to maintain the coherence and compile time checking capabilities originally designed into PASCAL by Professor Wirth.

We, ADP Network Services and MDSI, have a variety of positions open in the areas of PASCAL compiler development, systems programming, and applications development using our PASCAL. I would be pleased to receive any resumes your readers would like to send and would see that they are properly considered by Mr. Kortesoja and myself.

Sincerely,



Neil J. Barta
Manager, Programming Languages

NJB/kjs



The University of Calgary

2920 24 AVE. N.W.
CALGARY, CANADA
T2N 1N4

77-07-29

DEPARTMENT OF COMPUTER SCIENCE
TELEPHONE: (403) 284-6316

Dear Andy,

Enclosed is my renewal; if I've missed P.N. #9, would you send me a copy?

I really stand in awe of the job you've done in publishing the P.N.; nevertheless, I hereby add to your burden with the following.

If Pascal is to compete with Fortran, I believe four things are needed which I have not seen discussed as a unit in the Pascal Newsletter so far; hence, this letter.

Before I go on, I should point out that all the possibilities discussed here can be inserted into the Pascal language without much syntactic change. Better still, efficient one-pass compilation of these features is still possible, Fortran being a weak demonstration of the fact, another being found in an M.Sc. thesis which discusses these and many other interesting possibilities, "Pyxis: A language Evolved from Pascal" by E. N. Kittlitz, Department Computer Science, University of Calgary, 1977. (The author may be contacted via that department, Calgary, Alberta, Canada T2N 1N4.)

First, concerning storage mapping: I join the cry for a variable initialization facility, which in turn implies a certain amount of statically allocated storage.

Second, storage could be explicitly allocated as static either in common blocks or as "private" areas for given procedures or functions. Then one has the possibility of Pascal subroutines that do not use the run-time stack and so could be loaded with and called by a Fortran main program. A second benefit that I find personally more important is that one could then program more modularly, no longer having to use unprotected globals to implement the Algol own.

Third, there is the need for flexible array parameters; I don't suppose that is debatable any more. Of course, one must distinguish between flexible array parameters and "rubber" dynamically-allocated arrays. It strikes me as not in the spirit of Pascal to admit rubber arrays, nor would rubber arrays be at all necessary from the view of Pascal as a Fortran-replacement.

The flex array facility of Pyxis has merit; for example, it costs nothing if you don't use it. The following is a Pyxis program fragment which prints the sums of the two 6-element vectors.

```
type Flexvec = array [1 to *] of real;
var A: array [5 to 10] of real;
    B: array [-3 to 2] of real;
function Sumvec(X: Flexvec): real;
var I: integer;
    S: real;
begin S := 0;
  for I := 1 to UPB(X)
  do S := S+X[I];
  return S
end;
begin (*initialize values*);
  write(Sumvec(A), Sumvec(B))
end
```

SPECIAL TOPIC: MICRO/PERSONAL COMPUTERS AND PASCAL

Pyxis also allows one to allocate flex-typed objects of run-time-specified size to the heap, and to have a pointer which may reference any object of a given flex type, i.e. an object of a type which falls within the class of types specified by a flex type declaration.

The fourth point involves the great format debate, and variant records too. I think people are not thinking straight about these issues. A text file is not a string, nor a sequence - not even one of indeterminate length! It has funny states, e.g. the "not-opened" state; even an abstract model of a file does odd things. In Pyxis, a program interacts with a file (which is "outside" the program) via its image (which is a record of status information with a string acting as a buffer); a string is a fixed-length packed array of characters, in the Pascal sense. Thus, format operations become type coercions changing various simple data types into short strings and vice-versa; the analogy with integer-to-real coercion is quite good, and format operations are no longer the perquisite of the file handling package.

Of course, not all the foregoing viewpoint fits well with Pascal, but some fair amount does, and is worth considering. Assuming a good type coercion syntax can be designed, format operations could simply be functions which accept or return flexible arrays of characters, and their use in I/O becomes natural without being their only use. Further, if you do not use these functions, they need not consume space in your load module.

The tie-in with variant records should be clear. Variant records are used for two totally distinct and completely valid reasons. The first is that which they were designed for; the second is to pun: One must write one's own "dispose"; one needs to dump large list structures; and for a myriad of other purposes a programmer sometimes needs to get at the bits, do arithmetic on pointers and the like. Although these activities are machine dependent they are not dirty; because they must be done with great care, they must be done in a good language; and because they are so universally necessary, they ought to be accommodated in the language in a clear machine-independent way.

Rather than continuing the abuse of the variant record, let the job be done by a syntax designed for the purpose. To this end, I favor the common idea of allowing a <type identifier> to be used as a <function name> such that if its (one) argument is of a suitable type, a pun is allowed or, in certain specified cases, a coercion occurs. A suitable type for punning would normally be a type requiring the same storage as that which one is "punning it into"; and if the user doesn't know his implementation well enough to do what is required, he's still better off with the resulting error message than with the current "guess and hope" method required by variant records.

In summary, I hope most for variable initialization, private (own) variables, flexible array parameters (but not rubber arrays), and a view of type coercion to solve both formatting and variant-record problems.

Killing Fortran was presented as a motivation; more precisely I want a strong, viable language so I won't have to reprogram soon. I've done a lot of work in Pascal, in part because I hope that with just a little more strength of expression Pascal will survive; but I also believe that without that strength, it won't.

Very truly yours,



Stephen Soule,
Assistant Professor

The following four letters deal with some developments described on page two of the EDITOR'S CONTRIBUTION. See also the IMPLEMENTATION NOTES section under INTEL 8080, LSI-11 Motorola 6800, etc. And also see HERE AND THERE News section under Kenneth Bowles, Kurt Cockrum, John Collins, Creative Computing, Jack Crone, Dan Fylstra, Roger Gulbranson, Carl Helmers, Sam Hills, Aron K. Insinga, Barbara I. Karkutt, Ed Keith, Donald Lindsay, Tim L. Lowery, Bruce Mackenzie, Jim McCord, Carlton Mills, Carol Anne Ogdin, David Segal, Bruce Seiler, Michael Settle, Jeffrey G. Shaw, David H. Welch, and Richard West!

104B Oakhurst Circle
Charlottesville VA 22903
8 July 1977

Andy Mickel, Editor
Pascal Newsletter
University Computer Center
227 Exp Engr
University of Minnesota
Minneapolis, MN 55455

Dear Mr. Mickel:

(1) I have received a reply from Dean Brown at Zilog about the hypothetical Pascal machine. Zilog is not describing the machine to the public at this time---see enclosed copy. Perhaps his spontaneous use of the term "Pascal machine" is a hopeful indication however.

(2) Enclosed please find copies of letters I have sent to Byte, Creative Computing, Kilobaud and Personal Computing as my one-man campaign to stamp out BASIC and increase Pascal's visibility.

(3) Since (judging from PN 8) Pascal will soon be available for personal computers, it seems to me that a timely collection of Pascal games and hobby programs might help wean the hobbyists away from BASIC. I personally have been writing Pascal versions of Star Trek, Mastermind, Lunar Lander and so on. I would like to hear from anyone in PUG interested in sharing such programs, and also from anyone who could explain to me the copyright laws concerning Pascal translations of copyrighted BASIC programs.

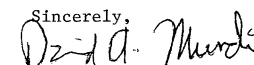
(4) I personally was aghast at the proposal to change variant record usage (PN 8-15). I think the language designer's responsibility to protect the programmer from himself stops short of that. Perhaps I have strange tastes, but I like having access to individual bits of a word by treating the word as a packed array of boolean. I like being able to declare

```
var r: record case boolean of false:(x,y,z:integer);
                                     true:(p:array[1..3]of integer) end;
```

so that for statements can be used for assignment (for i := 1 to 3 do p[i]:=something) yet clumsy array notation is avoided in other situations, for example: write(a[x,y,z]) instead of write(a[p[1], p[2], p[3]]).

(5) Could someone in PUG explain why Pascal's semicolons make Prof. Sales weep? (PN 8-33)

(6) Congratulations on the Newsletter.

Sincerely,

David A. Mundie

June 27, 1977

Dear Andy,

Thank you for the copy of your newsletter. I will put a "short contribution" extolling it in the next issue of Interface.

As Steve Legenhausen points out on page two of the newsletter, BASIC is becoming a microcomputer standard. I am very much interested in urging our members to consider other languages than BASIC, and would like to publish anything which would work to that end. An article such as a Pascal tutorial, a critique of BASIC (control structures, data types, etc.), a Pascal bibliography, a survey of micro-based Pascal activity, or a Pascal subset proposal, would be most valuable for our readers.

If you or any PUG members would be up for writing or compiling material along these lines, I would love to publish it. Like yours, our format is quite flexible, with room for short contributions as well as longer articles.

Sincerely,



Larry Press
Editor

P.S. We have an informal system of coordinators for various topics. Would you mind if I were to list you or PUG as coordinator for Pascal?

Rec'd
7/7/20

MICROCOMPUTER LIBRARY & RESOURCE CENTER

Sept 1, 1977
Aug. 1, 1977

Maria Lindsay
Coordinator
5150 Anton Dr
Room 212
Madison, WI 53719

Pascal User's Group
c/o Andy Mickel
University Computer Center
227 Exp Eng.
University of Minnesota
Minneapolis, MN. 55455

7/9/77

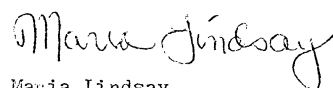
Dear Mr. Mickel:

Our Microcomputer Library & Resource Center is rapidly growing. It is a free service to the areas computerists. We maintain updated files on manufacturers & distributors of microcomputer products, including all their current catalogues, brochures, and information sheets. We already have filled 5 file drawers. We also keep a stock of extra copies of materials to give free to interested people.

We have current and back issues for ^{Now 36} 10 different computing magazines and also for newsletters and user group notes. (Most have been donated by individuals or the source, due to our minimal working budget) We would be very pleased to have several copies of your subscription forms and brochures available for our patrons. We also hope that there might be a way that we might get a copy of your current and back issues of your newsletters, especially the first issue. (Please consider our minimal budget) In return for your generosity we hope to interest our patrons in your services. (and add a few more subscribers for you) If there is a charge please inform us of it.

Thank you very much. We hope to hear from you soon.

Sincerely,



Maria Lindsay

Manufacturer Brochures & Literature · Magazines · Software · Reviews · Books

AUGUST 24, 1977

PASCAL USERS' GROUP
attn ANDY MICKEL
UNIVERSITY COMPUTING CENTER
227 EXP. ENGR.
UNIVERSITY OF MINNESOTA
MINNEAPOLIS MN 55455

Dear Andy,

I've received issues 5-8 of the PUG Newsletter, and am mightily impressed with the sheer volume of (largely usable and interesting) material you have managed to compile and publish.

In reference to my earlier offer to help promote PASCAL, you mentioned "pressing our advantage in the microprocessor area", through articles and letters to such magazines and journals as Dr. Dobb's, Byte, Personal Computing, Creative Computing, etc. While I'd be glad to swamp these and other publications with pro-PASCAL material, I really can't "press" any "advantage" because, frankly, we have none -- yet. As of today, I know of no reasonably-priced, memory-efficient generally available implementation of PASCAL (for a decent subset), in compiler OR interpreter form, suitable for use on any of the popular micros, with the dubious exception of the LSI-11, which has itself only become inexpensively available through the still brand-new Heath computer line.

Having an "advantage" entails, for me, two considerations. First, one's product or service must be inherently superior to its competition. Secondly, it must be available and easy to use. PASCAL certainly is a superior language, perhaps the worthiest I've yet encountered (despite its many flaws which I hope will be truly CORRECTED, and not merely "written around"). However, the availability of a powerful, easy-to-use micro-PASCAL remains nil, and so our "advantage" remains merely a tantalizing phantom. For the average micro-user, PASCAL is, and will remain "unreal" until someone comes up with an implementation which is, from both aesthetic and practical standpoints, more attractive than the alternatives BASIC and FORTRAN. (Micro-PASCAL will have to be "more attractive", of course, in order to lure away the vast majority of satisfied BASIC and FORTRAN hackers, and give them proper cause to learn and embrace a strange new language.)

I've been reading about the UCSD PASCAL project, and I'm filled with hope that, finally, I will be able to show my doubting friends and customers something more than the (often confusing) User Manual and Report. Perhaps I will be able to demonstrate a working compiler or interpreter, as well as the superiority of PASCAL as a programming tool. The moral victory would be even sweeter if I could point to simultaneously-available IDENTICAL versions of the language optimized for the LSI 11, Z80, 8080 and 6502! Anyway, until I hear more from La Jolla, the emergence of PASCAL into the micro-age is still my pipe dream.

Regarding media exposure for PASCAL, though, I am all for it, and suggest the formation of a steering or co-ordination committee to manage a media blitz to awaken the personal computing community to the advantages and joys of PASCAL programming. What do you think? I have noticed too many APL articles popping up lately, and suspect that either co-incidence is working overtime or an APL blitz committee (formal or informal) has been formed and is calling the shots. In either case, we'd better get something together if we intend to make any dent in the personal computing sector. APL, as cryptic as it is, is still a good language, and could very well bury us by default if we don't watch out.

Finally, in PUGN #8 (I think), you expressed interest in getting informa-

JIM MERRITT
PO Box 4655
Berkeley CA 94704
PHONE 415-845-4866

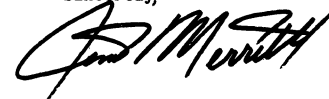
JIM MERRITT TO:
ANDY MICKEL/PUG/MINNEAPOLIS MN 55455

DATE: AUGUST 28, 1977
PAGE 2 of 2 .

tion concerning the UC Berkeley UNIX PASCAL compiler/interpreter. Enclosed is the MANUAL documentation, which should give you some help. If you need more, let me know.

Have some good times, and good luck with the next Newsletter! I'm looking forward to it!

Sincerely,



(* Editor's Note: I replied to Jim in a quick letter dated 77/08/31: "... My basic problem is time, and the hasty note I scribbled last time to you did contain some hazy thoughts. What I meant by 'pressing our advantage' was literally that in the 5 years I've been involved with Pascal, there were no areas where we had a chance to shine and the doomsayers were pretty explicit about us keeping in our place. But because microprocs/ personal computers are relatively new, there's a much less powerful establishment to overthrow. So relatively speaking Pascal µproc developments seemed to me further along than other fronts & that we should concentrate energy there (press). Oh well, I should have originally said 'enlarge the opening.'

'I agree about the APL problem. It upsets me a great deal.

"Regarding other fronts, I consider that we haven't and shouldn't yet take on COBOL, and that Pascal .vs. FORTRAN is the front I've been involved with.

"Other fronts are of course getting manufacturers just to support Pascal processors in their software line, and getting stuck up computer science departments to teach the stuff.

"I appreciate your offer of help and am glad you liked the newsletters.

"The spirit of PUG so far has been its far-sighted inability to form working committees - just loose unions...." *)

Carl T Helmers, Jr
Editor in Chief

BYTE
The Small Systems Journal

BYTE Publications, Inc.
70 Main St. • Peterborough, N.H. 03458 • 603/924-7217

BYTE Publications, Inc.
70 Main St. • Peterborough, N.H. 03458 • 603/924-7217

BYTE
The Small Systems Journal

FROM
THE
EDITOR'S
DESK

TO: Andy Mickel
Editor, Pascal Newsletter
University of Minnesota
227 Experimental Eng. Bldg.
Minneapolis, Minnesota 55455

September 6, 1977

Dear Andy,

Finally getting around to a detailed reading of PUG Newsletter #8 provides me with a theme for an editorial I will put into the December 1977, pushing PASCAL as a possible language. I picked up several Springer-Verlag books at IFIPS last month and have since spent some time discussing PASCAL with my good friend and associate Dan Fylstra.

I think that PASCAL would make an excellent choice as a successor to BASIC in the personal computing world, a thought which is echoed by several contributors to PUGN #8. Here are two points about PASCAL Personal Computing Use which will no doubt appear in the editorial I am composing this week:

* Like BASIC, PASCAL is an academically originated language with a fairly well defined set of machine independent standards. As such it has one major point in its favor: it is not a proprietary product confined to any one organization, and is thus open to the general computing public as a standard to be implemented and delivered with machines. Thinking of the general public as users requires a machine independent (or nearly so) language, and in the interests of better software techniques a structured language like PASCAL comes to mind. The large amount of activity evidenced by PUGN suggests that both the academic training and wide usage which were present in BASIC's evolution will also be available with PASCAL.

* When implemented for the personal computing milieu, PASCAL should at a minimum level of function offer an interpretive or semi-compiled interactive system which is friendly to the user in the same way that BASIC is friendly. Fully compiled and optimized code generation is not needed in a context where one high speed processor is dedicated to each user and his or her files.

As a final point in closing, we (BYTE Publications) are in the process of preparing a series of publications initially oriented to systems software books characterized by tutorial documentation of the design, complete publication of source code and any necessary intermediates, machine readable representations of the source of object text, and other information relevant to the process of getting the particular software item running in the user's personal system. (Where machine dependence is involved, we are looking for target machines which are in the following set: LSI-11, 6800, 6502, 8080, Z-80.)

I would be most interested in talking with readers of PUGN who have implementations of PASCAL available for sale which run interpretively, semi-interpretively or as compilers. Our standard form of publication agreement is an exclusive book and audio record publishing license to the software and its machine readable representations.

I'll send a copy of the editorial after it is written.

Sincerely,

Carl T Helmers
Carl T. Helmers
Editor in Chief
Byte Publications, Inc.

SPECIAL TOPIC: PASCAL STANDARDS

In Pascal Newsletter #8, we devoted many pages to a series of letters about standards. Among the actions described as being taken were: 1) we try to clarify instances of vague semantics in the Pascal Revised Report, and 2) Tony Addyman of the University of Manchester coordinate an effort to get an ISO standard certified which would amount to a 'tightened up' Revised Report with no additions.

This summer, Tony phoned that:

- 1) he had received another list of points requiring clarification from Jim Welsh in Belfast.
- 2) he wondered if there would be copyright problems with the current Revised Report already published and the proposed standards document.
- 3) he was very pleased that the June meeting of a British Standards Institute (BSI) committee on programming languages (of which he is a member) authorized a working group (headed by himself) for a Pascal standard. This is for the purpose of certifying a document as a standard, not to propose additions and changes to the language.
- 4) he envisions an appendix to the Report which would both suggest various strategies for things left to be defined by an implementation and list conventionalized extensions.

I sent him a small list of items which included:

- 1) Optional ; on last limb of a case.
- 2) Role of the word-symbols: extern, forward, and fortran (all non-standard) but in various implementations they are neither predefined, reserved, nor user-declared.
- 3) Many good definitions in the Axiomatic Definition don't appear in the Report. For example (brought to my attention by Charles Hedrick): the Report specifies the mod operator as the operation: "modulus." But the mathematical meaning of modulus gives things like: $-3 \bmod 2 = 1$. The Axiomatic Definition clearly states $-3 \bmod 2 = -1$.

On 77/08/17 I received a note from PUG member D. G. Burnett-Hall dated 77/08/10 which read: "Dear Andy,

I enclose 'Another Attention List' following Tony Addyman's Attention List in Newsletter 8: I've tried to avoid duplicating his points (and I've sent him a copy)." ...

Another Attention List

D. G. Burnett-Hall

UNIVERSITY OF YORK

1977 August 9

DEPARTMENT OF COMPUTER SCIENCE

Section

- 4 (a) Add "programs" to first sentence.
(b) Is " an illegal constant? (n = 0 characters not defined.)
- 6.1.1 type T1 = (ZERO,ONE); T2 = (ONE,TWO);
should be illegal because the type of ONE is ambiguous
(UM-5A,p34)
- 6.1.2 For Boolean type, better to make clear here that it is ordered
(false, true) than just a note in 8.1.4.
- 6.1.3 Allows lower = upper band for subrange type: UM-5B(p35) does
not. (Why?)

- 6.2.1 (a) Is array [integer] of real legal?? Note that
<index type> ::= <simple type> , and
<simple type> ::= .. | .. | <type identifier>.
(b) type T1 = array [0..9] of array [Boolean] of integer;
T2 = array [0..9, Boolean] of integer;
var A1:T1; A2:T2;
Are T1 and T2 identical types? (Assuming that "identical
types" means more than having the same type identifier.)
Specifically, is it legitimate to write A1 [5,true] or
A2[5][true]?
(cf. UM-6, p39)
- 6.2.2 (a) Field identifiers within a record must be distinct,
taking all variants of the record into account (UM-7,p46).
But one identifier can be used simultaneously as a field
identifier and the identifier of a variable (say) (UM-7,p49).
(b) Helpful if last example included an empty field list (UM-7,
p46). (*E.g. include POINT= in type SHAPE: also in
example at end of R6.3.*)
(c) Why is the conjunction of ;end
(i) illegal in the declaration record ... case ...;end , but
(ii) legal in the statement case ...;end ? (R9.2.2.2)
(*DEC-10 compiler rejects it in both instances.*)
- 8.1 This should include something along the lines of UM-4A(p21)
(and UM-10, p63/64) about whether compound Boolean expressions
are completely evaluated. (*It would change the language
to say now that they are evaluated only as far as necessary,
but I wish this had been done. So did the author of
SKIPBLANKS (UM-12A,p85) which is only dubiously legal.*)
- 8.1.2 (a) 14 div(-3) is not defined anywhere! Is it -4 or -5?
(b) mod operator is defined (in terms of div) only in
UM-2B (p13).
- 8.1.4 (In-) Equality operators for sets? (UM-8,p50). More
obvious than the set-inclusion operators the Report does
describe.
- 9.1.1 Consider also:
type R4 = record
case B : Boolean of
false : (I:integer);
true : (R:real)
end;
var X,Y,Z:R4;
(*Integer and real quantities need not be the same size*)
X.B := false; Y.B := false; Z.B := true;
X.I := 1; Y.I := 2; Z.R := 3.4;
X := Y; (*Presumably their types are identical*)
X := Z; (*Legal?? Are their types identical?*)
(*Does this imply a run-time check?*)
- 9.1.2 UM-11A (p71) says that if
procedure P (var X,Y:integer); ...
is declared, the procedure statement
P(A,A)
is illegal ("x1 .. xn should be distinct variables"). Why?

- 9.1.3 (a) Doesn't forbid duplicate use of one label in the same block!
 (b) procedure P;
 label 99;
 procedure Q; begin; 99: end (*Q*);
 begin (*P*); goto 99; ... end (*P*);
 R.9.1.3.1 does not require label 99 to be in Q, thus contradicting its second sentence. In R9.1.3.2, should "procedure" be replaced by "innermost procedure, function or program"?
- 9.2.2 (a) All the case labels within one case statement must be distinct (UM-402, p31).
 (b) One of the examples should include a list of case labels.
 (c) A case label cannot be used as the destination of a goto statement. (This is implicit, but it would be helpful to make it explicit.) In the final example, it is very tempting to write:
 1: begin x := x- pi/2; goto 2 end;
- 9.2.3.3 (a) Never says that the statement S will not be obeyed if $e1 > e2$ (to) or $e1 < e2$ (downto)!
 (b) The semantic explanations
 (i) should be enclosed in begin ... end;
 (ii) firmly state that the final value of v is e2 (*if S does not cause a jump*). And why not? (Except that UM-4C2, p24, says it is "undefined".)
- 10 (a) forward is not mentioned at all. (cf. UM-11C1, p82/83, which consistently does not treat it as a special symbol: surely this is a mistake?) Is this a case for a "conventionalised extension"?
 (b) If a procedure or function is used as a formal parameter, UM-11C2 (p83) states that the corresponding actual procedure/function must take value parameters only. (*And so it can't itself have a procedure or function parameter: for this relief much thanks.)*
- 10.1 I should favour making HALT a standard procedure (for use after disastrous errors). (*It can be done by goto 9999, where 9999: is at the end of the program, but that may involve exit from procedures, and HALT is much simpler.*)
- 10.1.2 NEW(P,T1,...,TN) : the wording does not make clear whether the tag-field values are also assigned. UM-10(p64) emphasizes that no assignment takes place.
- 11.1.3 I assumed that TRUNC(X) was the mathematical function [X]: UM-2B(p13) says TRUNC (-3.7) = -3, not -4. The present wording implies sign-and-modulus representation of numbers.
- 11.1.4. PRED and SUCC: type of argument must exclude real. (UM-2C, p14)
- 12 Can one assume, when reading a textfile F, that eof(F) becomes true only (i) after readln (f) or (ii) immediately, on reset (F)?

- (i.e. the file ends with the end-of-line marker or is empty?) If not, the program schemata in UM-9A2. 9A3 (p58/59) will fail. This assumption would considerably simplify input of data (try rewriting UM-9A2 if eof can occur at any moment!) and would be easily implemented in the run-time system. Perhaps this should be answered in R14?
- 12.1.4 (a) var R:real; I:integer; F:text;
 read (F,R,I)
 If the data are +2 -13.4,
 — is it legal to give integer datum for real variable (cf r:=+2)?
 — is it illegal to give real datum for integer variable (cf i:=-13.4)?
 (*DEC-10 system rejects both: the first is unhelpful and unnecessary.*)
 (b) When read (F,V) finishes reading a number, is F† the character which terminated the number (and not the one following it)? UM-App.F suggests it is.
 (c) What terminates (i) an integer, (ii) a real number? Does end-of-file terminate an otherwise valid number? (*My suggestion under 12 would remove this possibility*)
- 12.3.5 If e is non-negative, should it be preceded by (i) '+' or (ii) at least one space, or (iii) are no preceding spaces required? (UM-12B7,p86, suggests the last.) Should this be answered in R14?
- 12.3.6 (a) UM-12B7,12B8 (p86/87) is considerably more helpful, but still imprecise.
 (b) What, if anything, replaces the + sign for (i) a positive number, and (ii) a positive exponent? (UM-App.F answers (i) one space, (ii) '+'.) Should this be answered in R14?
- 12.3. (9) The last 3 lines of R12.2 should be moved to the end of R12.3. Should they be numbered 12.3.9? (cf last 3 lines of R12.1)
- 12.1.2, 12.1.3, 12.2.2, 12.2.3, 12.3.2, 12.3.4, 12.4.2:
 enclose equivalent statements between begin ... end.
- 13 Why is reset (input) illegal, and not just of no effect? (And rewrite (output)?, for that matter?) If one can reset (F) for a textfile F to re-read its data, why not for INPUT? (*DEC-10 system allows them: chiefly because DEC-10 operating system expects matching of internal and external filenames to be done at run-time and not by JCL commands, worse luck!*)
- 14 (a) This section should include a list of those implementation-dependent constants whose values are needed when a program is being transported. e.g. MAXINT (UM-2B, p13, but not in Report), maximum size of a set, default values of M in write (E:M) for integral and real E, etc. (*Should these be standard constant identifiers? I favour a limited number of environment enquiries, c.f. Algol 68*)
 (b) A compiler-option takes the form of a comment whose first character is \$. (*That much, pace N. Wirth, but no more. Itensures that one can put comments in portable programs which won't accidentally be taken as compiler options.*)

Implementation Notes



CHECKLIST CHECKLIST CHECKLIST CHECKLIST

1. DISTRIBUTOR/IMPLEMENTOR/MAINTAINER
(* Names, addresses, phone numbers. *)
2. MACHINE
(* Manufacturer, model/series and equivalents. *)
3. SYSTEM CONFIGURATION
(* operating system, minimum hardware, etc. *)
- * 4. DISTRIBUTION
* (* cost, magnetic tape formats, etc. *) * * * * *
- * 5. DOCUMENTATION
* (* In form of supplement to Pascal User Manual and Report
* Machine retrievable? *) * * * * *
- * 6. MAINTENANCE POLICY
* (* How long? Accept bug reports? Future development plans. *) * * * * *
- * 7. STANDARD
* (* Implements full standard? Why not? What is different? *) * * * * *
- * 8. MEASUREMENTS
* (* -compilation speed (in characters/sec. please; this is a
* meaningful measurement for compilation speed);
* -compilation space (memory required at compilation time);
* -execution speed;
* -execution space (the memory required at execution time;
* compactness of object code produced by the compiler);
* ** Try to compare these measurements to the other language
* processors on the machine, e.g., FORTRAN. *) * * * * *
- * 9. RELIABILITY
* (* stability of system (poor, moderate, good, excellent);
* how many sites are using it?
* when was the system first released to these sites? *) * * * * *
- * 10. DEVELOPMENT METHOD
* (* Compiler or interpreter? Developed from Pascal-P / hand-
* coded from scratch/bootstrapped/cross-Compiled/etc.? What
* language? Length in source lines? Effort to implement in
* person-months? Previous experience of implementors? *) * * * * *
11. LIBRARY SUPPORT
(* Libraries of subprograms available? Facilities for
external and FORTRAN (or other language) procedures
available? Easily linked? Separate compilation available?
Automatic copy of text from library into source program
available? Symbolic dumps available? *)



GENERAL INFORMATION

As this is the first issue of Pascal News in this academic year July 1, 1977 - June 30, 1978, let us explain how this section is organized:

- First a CHECKLIST to be used as a guide to distributors, implementors and maintainers for reporting the status of Pascal implementations on various computer systems.
- A SOFTWARE TOOLS section describing aids to Pascal users in developing applications.
- A PORTABLE PASCALS section reporting distribution information about kits used to produce Pascal compilers for real computer systems.
- Information on PASCAL VARIANTS.
- A FEATURE IMPLEMENTATION NOTES section describing implementation strategies and details of various Pascal features as suggestions to all the compiler implementation efforts underway.
- A list of MACHINE DEPENDENT IMPLEMENTATIONS sorted by name of computer system, giving news of Pascal compilers for real machines.
- And in subsequent issues this year, an INDEX to all the implementation information for this year.

We are essentially beginning anew this year and so in this issue we are combining summaries hand-compiled from PUG's 5-8 (last year) with the news received since #8.

IMPLEMENTORS - MAINTAINERS - DISTRIBUTORS

Please use the checklist if you are reporting information and please keep us all up to date. You might also send us a copy of your documentation and distribution forms as so many implementors have done so that we can keep up to date on the overall development of Pascal. Please send camera-ready copy, single spaced, and use wide text (we prefer 18.5 cm lines). We also will accept reports on ASCII paper tape accompanied by a listing. And please include PUG All-Purpose Coupons with each copy of your system that you send out!

USERS

Please help make us all informed consumers of Standard Pascal systems by reporting your quantitative and qualitative experiences with particular implementations.

EVERYONE

We would like to thank all the effort put forth by people who have sent in information. We regret to say that our ability to answer individual requests is limited not only by time, but by the commitment we have first to this section of the newsletter. Therefore we prefer to answer inquires through Pascal News. We print all the news that comes to our attention barring oversights and mistakes on our part!

SOFTWARE TOOLS

There has been much discussion concerning the distribution of Software Tools written in Pascal in PUGN 5-8. Please see the letter by Richard J. Cichelli in the OPEN FORUM section this issue. It was our idea that tools should be incorporated with the distribution of Pascal implementations but even this poses real problems. Starting next issue we should see news on tools greatly expand.

Examples of software tools are listed below:

1. A program to cross reference identifiers in Pascal programs.
2. A decompiler to examine the object code produced by a Pascal compiler.
3. A prettyprinter to format and indent Pascal programs.
4. Performance measurement programs to monitor execution times in Pascal programs.
5. A program to change character sets from one to another.
6. A program to compare two text files and generate a set of modifications to convert one to the other.
7. A text editor to alter and record modifications made to a source program.
8. A text formatter used to produce written documentation about software for users.

Believe it or not, right now several different programs exist in categories 1, 4, and 8, and at least one program exists in every category except 7! All are written in Pascal!

PORTABLE PASCALS

Pascal-P.

The most widely used portable compiler for creating new Pascal Implementations is Pascal-P. Basically Pascal-P is distributed from three points in the form of a kit consisting of a magnetic tape and printed documentation.

Pascal-P is a compiler written in Pascal (almost 4000 lines) which generates symbolic code for a hypothetical stack machine called a "P-machine" because it is somewhat of an ideal architecture for Pascal-P. The symbolic code is thus called P-code.

On the magnetic tape are textfiles containing:

- a sample character set collating sequence. This file is also distributed as a listing to simplify character set conversion.
- the Pascal-P compiler in Pascal.
- a P-code assembler/interpreter written in Pascal which is intended to document how to write an interpreter in an existing language on the target computer system.
- a Pascal-P compiler in P-code. In other words, the result of compiling the Pascal-P compiler on itself.

Implementation Notes

The person implementing Pascal has several choices. If there is no access to a working Pascal compiler on another machine, the implementor orders a Pascal-P kit already configured to the target machine. Configured compilers have constants inserted in them to specify, for example, the size of each simple data type. These configuration parameters are given by the implementor on the Pascal-P order form. (See below.)

After receiving the kit, the implementor can write an interpreter for P-code in another language (usually takes about one person-month), and thus immediately has access to a Pascal compiler running interpretively by using the P-code version of the compiler included in the kit.

To produce a real Pascal compiler for the target machine then requires editing of the Pascal-P compiler written in Pascal to produce code for the target machine (instead of the P-machine). After recompiling, a Pascal compiler exists in the code of the target machine.

If the implementor initially has access to a working Pascal compiler on another machine, the step of writing a P-code interpreter can be omitted.

Facts about the Pascal-P compiler:

- The current version is called Pascal-P4 and is distributed with a copy of Pascal-P3 (which is of interest to previous recipients of Pascal-P2).
- Pascal-P4 represents a major improvement over earlier Pascal-P versions because it removes data type alignment restrictions, is more efficient, includes runtime tests, and is a more complete implementation of Pascal.
- Pascal-P2 was developed from a phase in the stepwise refinement of Urs Ammann's Pascal-6000 compiler in 1974 by K. V. Nori, Urs Ammann, K. Jensen, and H. H. Nageli. Subsequent improvements were done by Christian Jacobi.
- Reliability of Pascal-P4 has been fairly good. As of Spring, 1977, it was distributed to 106 sites by George Richmond and to 37 sites by Chris Jacobi. (No distribution data has been received from Carroll Morgan.)
- Several good reports on the viability of Pascal-P were reported in PUGN #8 as well as two more in this issue: Ted Park for a Data General Nova and John C. Knight for a CDC Star-100.
- There is no promise of maintenance for Pascal-P. P4 is the final version produced at Zuerich. We at Pascal News will attempt to print bug corrections in future issues in this section.
- Documentation for Pascal-P4 consists of a 65 page report entitled The Pascal <P> Compiler: Implementation Notes (Revised Edition) July, 1976. (A 24 page correction list to the original December, 1974, edition is also available.)
- Pascal-P4 is a significant subset of Standard Pascal. Restrictions to the standard include:

- procedure and function formal parameters are not allowed.
- files are not implemented.
- goto's may not exit procedures or functions.
- a (rather small) maximum string constant length is imposed.
- the subrange form of set constants is not implemented.
- nil is not a reserved word, but rather is predeclared.
- many standard procedures and functions are not fully implemented.
- text and maxint are not predeclared by the compiler.

Pascal-P can be ordered from three places (write for prices and order forms).

In Europe, Asia, and Africa, order from:

Christian Jacobi
Institut fuer Informatik
E.T.H. Zentrum
CH-8092 Zuerich
Switzerland
Phone: 41/1-32 62 11 x2217

In North and South America, order from:

George H. Richmond
Computing Center: 3645 Marine Street
University of Colorado
Boulder, CO 80309
USA
Phone: 303/492-8131

In Australasia order from:

Carroll Morgan
Basser Dept. of Computer Science
University of Sydney
Sydney, NSW 2006
Australia
Phone: 629 1122

(* We at PUGN would appreciate new ordering information be sent to us by these three distributors for inclusion in Pascal News #11. We would also appreciate some sort of coordination on a common order form for these three distributors. *)

Pascal Trunk Compiler

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. H. H. Nageli, Institut fuer Informatik, ETH-Zentrum, CH-8092 Zuerich, Switzerland (Tel. 32 62 11).
2. MACHINE. The trunk compiler is the machine independent part (e.g., syntax analysis and error recovery) of a Pascal compiler in which the code generation has to be inserted in a certain number of empty procedures.
3. SYSTEM CONFIGURATION. Requires a working Pascal compiler.
4. DISTRIBUTION. Magnetic tape. Cost: SFr. 50.--.
5. DOCUMENTATION. In German, available in May, 1977 (77/3/3). Detailed comments in the source describe how an implementor can write algorithms for the machine dependent parts.
6. MAINTENANCE. Not defined yet.
7. STANDARD. Full Pascal is treated.
8. MEASUREMENTS. Not applicable.
9. RELIABILITY. Moderate (77/3/3). The Trunk was used with good results in 1975-76 by Teruo Hikita in producing a high quality Pascal compiler for the Hitachi 8000 series.
10. DEVELOPMENT METHOD. Started in 1975 from a phase in the stepwise refinement of Urs Ammann's Pascal-6000 compiler. The Trunk is a 5800 (indented) line Pascal source program in which the machine dependent parts are clearly marked and separated from the machine independent parts.
11. LIBRARY SUPPORT. Not applicable.

Some other machine-dependent compilers are written in such a way that they might be useful as Trunk compilers. Take for example, the current ICL 1900 compiler written by Jim Welsh, Colum Quinn, and Kathleen McShane at the Computer Science Department, Queen's University, Belfast, Northern Ireland, BT7 1NN, United Kingdom. The syntax analysis is clearly separated from the code generation in this compiler, which is written in Pascal. See ICL 1900 under Machine Dependent Implementations.

Another possible Trunk-like compiler is that implemented by Alain Tisserant,

Departement Informatique de l'INPL, Ecole des Mines, Parc de Saurupt, F-54042 Nancy Cedex, France. In this case, the compiler operates in two passes; the first pass can be parameterized and the second pass can be rewritten to generate code for different machines. This effort is explicitly oriented toward 16-bit machines. So far as we know, no other implementations have been developed from the initial compiler. See SEMS T1600 in the Machine Dependent Implementations section.

Pascal J

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. B. W. Pavenel, C. B. Mason, Software Engineering Group, Dept. of Electrical Engineering, University of Colorado, Boulder, CO 80309, USA (303/492-7204).
2. MACHINE. Pascal-J is a compiler which translates Pascal to the intermediate language Janus, a totally portable "mobile programming system" -- even to the point of defining its own character set! Janus in turn is macro-processed via Stage2 which is implemented in standard Fortran.
3. SYSTEM CONFIGURATION. ANSI Standard 1966 Fortran IV compiler. Specify character set: (a) ASCII (full 96, or 64 character subset), (b) EBCDIC, (c) CDC display code, or (d) other character sets if detailed collating sequence is sent.
4. DISTRIBUTION. 7-track magnetic tape (1200 ft. reel) \$28.00 (0.8 kg); 9-track magnetic tape (1200 ft. reel) \$39.00 (0.8 kg). Subtract \$7.00 if you supply a 1200 ft. reel. Longer reels are accepted, but more postage is charged. Overseas orders must add cost of postage and specify type of shipping.
5. DOCUMENTATION. (a) SEG-76-1 "A Preliminary Definition of Janus" \$4.00 (180 grams); (b) SEG-76-2 "PASCALJ Implementation Notes" \$2.00 (60 grams); (c) SEG-76-1 (*-3?*) "Janus Memory Mapping: The J1 Abstraction" \$2.25 (60 grams).
6. MAINTENANCE. Every six months (February and September) a new release is planned, but this is subject to manpower constraints. Attempt to fix all reported bugs.
7. STANDARD. (* no information - presumably full Pascal *)
8. MEASUREMENTS. As an interpreter, very slow, but the intent is to do a full bootstrap to a real compiler.
9. RELIABILITY. Moderate, improving with each release (Sept. 1975, Feb. 1976, Sept. 1976, Sept. 1977). As of February, 1977, the portability of the September 1976 release is deemed inadequate with implementation times ranging upwards from six person months.
10. DEVELOPMENT METHOD. Compiler originally written in Pascal to generate Janus, and used to translate itself to Janus. Janus processor written in Stage2 macros as an LL(1) system. The set of macros consists of stack operations and indexing in terms of a single accumulator and simple index register. A set of macros for multi-register machines is being implemented. The Stage2 macro-processor is implemented in Fortran.
11. LIBRARY SUPPORT. Not applicable.

PASCAL VARIANTS

Pascal-S

A description of Pascal-S comes from the abstract in the report "Pascal-S: A Subset and its Implementation", by Niklaus Wirth, Institut fuer Informatik, ETH Zuerich, June, 1975. (Available for \$6.50 from George Richmond; see address under Pascal-P.)

"Pascal-S is a subset of the programming language Pascal selected for introductory programming courses. This report describes an implementation that is especially designed to provide comprehensive and transparent error diagnostics and economical service for large numbers of small jobs. The system consists of a compiler and an interpreter and is defined as a single, self-contained Pascal program. This machine-independent formulation in a high-level language facilitates its construction and is a prerequisite for easy portability."

Standard Pascal constructs omitted from Pascal-S are: scalar and subrange types, pointers, set and file types, with and goto statements, the passing of procedures and functions as parameters, and several standard procedures. The only file operations are read on input and write on output. The report contains a complete listing of the compiler and interpreter on 34 pages!

Pascal-S is currently distributed on tape with the second release of the CDC-6000 Pascal compiler from Zuerich, Colorado, and Sydney. Pascal-S was implemented in PL/I under Honeywell Multics by the Computer Science Department, University of Southwestern Louisiana, P.O.Box 4-4330, Lafayette, LA 70504 (318/234-7640).

U. Lecarme reported on 77/03/04 that Helmut Sandmayr, Neu-Technikum, CH-9470 Buchs, Switzerland (085/6 45 24), has implemented a Pascal-S compiler (not interpreter) for the IBM 1130.

Rich Cichelli reports (77/08/31) that an incremental interactive (conversational) Pascal-S compiler was implemented at Lehigh University which is smart enough only to recompile the subprograms in which changes are made.

Concurrent Pascal

A portable pair of Pascal compilers was implemented by Per Brinch Hansen and Al Hartmann at Cal Tech in 1974-1975 for the PDP 11/45. The system consists of a "Sequential Pascal" compiler, a "Concurrent Pascal" compiler (used for writing operating systems and other concurrent programs), and a "kernel" or machine dependent set of run time routines written in assembler. The project at Cal Tech centered around writing a one-user operating system called SOLO in Concurrent Pascal. Both compilers are written in Sequential Pascal.

In 1975-1976 the system was distributed widely (252 sites) and led to the development of a machine independent version with a different kernel.

As reported in PUGN #6, distribution of Concurrent Pascal was terminated in August, 1976, when Per left Cal Tech to join the University of Southern California. On 77/07/12, Per phoned to say that distribution may resume and arrangements are being made. Details may be available for Pascal News #11.

Plans are to write a simpler kernel and I/O drivers. This may take 6 months.

Publications about Concurrent Pascal include:

- (1) "The programming language Concurrent Pascal", in the June, 1975, IEEE Transactions on Software Engineering 1:2, by Brinch Hansen.

- (2) A guest editorial and four articles by Brinch Hansen in the April-June, 1976, issue of Software - Practice and Experience 6, pp 139-205. The articles are entitled:
"The Solo Operating System: A Concurrent Program"
"The Solo Operating System: Job Interface"
"The Solo Operating System: Procedures, Monitors, and Classes"
"Disk Scheduling at Compile Time"
- (3) The book Operating Systems Principles by Per Brinch Hansen, Prentice Hall, 1973.
- (4) An article "Experience with Modular Concurrent Programming" in the March, 1977, IEEE Transactions on Software Engineering 3:2, by Brinch Hansen.
- (5) A Concurrent Pascal Compiler for Minicomputers by Al Hartmann, Springer-Verlag: Lecture Notes in Computer Science, Volume 50, 1977.
- (6) The new book The Architecture of Concurrent Programs by Brinch Hansen, Prentice-Hall, 1977.

Modula

Modula is a small language for dedicated computer systems and process control applications on small machines, developed by Niklaus Wirth and co-workers in 1975-76. It is conceptually cleaner than Concurrent Pascal in many respects. Modula is still experimental and the implementors in Zurich have insisted there are no distribution arrangements. (* We are hearing rumors of implementation efforts outside of Zurich though. *)

Published material on Modula includes:

- (1) "Modula: A Language for Modular Multiprogramming", Software - Practice and Experience 7 (1977), pages 3-35, by Niklaus Wirth.
- (2) "The Use of Modula", same as (1), pages 37-65, by Niklaus Wirth.
- (3) "Design and Implementation of Modula", same as (1), pages 67-84, by Niklaus Wirth.
- (4) "Toward a Discipline of Real-Time Programming", Communications of the ACM 20:8 (August, 1977), pages 577-583, by Niklaus Wirth.

The following is the Abstract from reference (4), above:

"Programming is divided into three major categories with increasing complexity of reasoning in program validation: sequential programming, multiprogramming, and real-time programming. By adhering to a strict programming discipline and by using a suitable high-level language molded after this discipline, the complexity of reasoning about concurrency and execution time constraints may be drastically reduced. This may be the only practical way to make real-time systems analytically verifiable and ultimately reliable. A possible discipline is outlined and expressed in terms of the language Modula."

Copyright (c) 1977, Association for Computing Machinery, Inc.
Reprinting privileges granted by permission of the ACM.

FEATURE IMPLEMENTATION NOTES

PORTABILITY NOTE

1977 February 17

SET OF CHAR

Introduction

I have recently been examining a number of PASCAL programs that are thought by their authors to be highly portable. It rapidly became obvious that it is not realized by the PASCAL community just how many problems are caused by the different character sets used on the computers we have available, nor how this problem is compounded by the set type in PASCAL. This note sets out to make the problems more widely known, and to make recommendations to implementors and programmers.

Character set collating order

There are two very common character sets in the computing industry: EBCDIC (adopted by IBM, Burroughs and ICL 2900 range), and ASCII (adopted by a number of other mainframe suppliers, and most minis), together with a few manufacturers who use their own idiosyncratic character sets (the key example being CDC). In this lot, we can assume nothing about the collating order except that the alphabets collate in ascending order; that the digits collate in ascending order and have successive ORD values; and that the lower-case alphabet collates either lower or higher than the entire upper-case alphabet (if it exists). Practically every other variant of ordering occurs.

This has always been a severe problem to programmers attempting to write portable software, and the advice that can be given only alleviates the problem: it cannot solve it.

Recommendation 1 : to PASCAL implementors

All PASCAL compilers should be able to handle objects of type char as internally represented in either the ASCII or the EBCDIC codes, and preferably both. It may be necessary to determine the char representation by a compiler option.

Recommendation 2 : to PASCAL programmers

Programmers writing code that depends on the collating sequence of objects of type char should

- (i) attempt to collect all such uses into a few routines, and
- (ii) adequately comment such uses so that the intent of the code is clear.

This advice applies particularly to programs which process PASCAL text by lexically analysing it.

Available characters

Except that PASCAL originated in CDC machines this would not be a severe problem, since ASCII and EBCDIC have a high degree of commonality in the graphics. Programmers should however be aware that the characters which can be assumed to be available on all computers are limited to the 48 FORTRAN characters. Others are available with varying degrees of probability (for example '>' and '[' are quite highly probable, but '^', '≡', '{' are extremely unlikely). The point of this is mainly felt when designing a language or sub-language or a reply system. An inappropriate choice of character may mean that there is no suitable alternative in another system, and doublet symbols will have to be used (as for example happened with the { } and (* *) in PASCAL itself).

The second major deficiency in awareness occurs in respect of the lower-case alphabet. Programmers, through long conditioning, are very proficient at reading solid upper-cased text. The general populace are not, and even programmers read normal text faster and more accurately than the upper-case we normally print. It is thus regrettable that many programs are written so as to totally ignore the existence of lower-case. Programmers should make provision for systems that can read and print lower-case alphabets to use them, even if their system cannot, by simply providing the hooks and commentary.

Recommendation 3 : to PASCAL programmers

Be aware of the essential differences between the printable graphics (and in some cases the control characters), and make allowances for these differences. They are important.

The set of char

The PASCAL set construct looks at first sight as to be heaven-sent to enable programmers to write code which is independent of character set collating order. The in operator allows testing a character for membership in a set, rather than having to do relational comparisons.

Alas, this is an illusion. Though conceptually the set construct is ideal, and it is excellent for writing such constructs for sets of more limited size, it falls down badly when it comes to a set of char.

The problem arises because sets are limited in most PASCAL systems to being contained in one or two machine words. Consequently, the size of the set is too small to contain all objects of type char in all the systems I have been able to see. The magnitude of the gap varies, and as PASCAL 6000 comes close to meeting the requirements for a set of char, PASCAL 6000 programmers assume it to be available on all computers.

To illustrate this, let me give the data I have on the set size and on the character set size for various implementations. I regret not knowing any implementation that has a true set of char, but probably one exists. I'd welcome any data on other implementations giving the character set and details as given here.

COMPILER	SET SIZE	CHAR SET
CDC PASCAL 6000	59	63 (CDC special)
ICL 1900	96	?
ICL 2900	48	64 (modified ASCII)
Burroughs B6700	48 (anysize planned)	256 (EBCDIC) 128 (ASCII)

The end-effect of this is that set of char is unreliable in CDC machines and virtually not available in other computers. Consequently programs which use this construct are highly unportable. Since the construct (if used) is likely to be used in many places around the program, it then causes considerable difficulty in rewriting the program.

Recommendation 4 : to PASCAL implementors

A set of char should give a compile-error unless the whole of the character set can participate as members of the set.

Recommendation 5 : to PASCAL implementors

If possible, implementations should permit a maximum set size which will accommodate all characters in the character set. The main problems centre around set operators, and the creation of set temporaries, if the wordsize is too small. If necessary, large sets may be restricted to the single case of 'set of char'.

Recommendation 6 : to PASCAL programmers

That despite its abstract attractiveness, programmers do not write code that contains a set of char anywhere in it.

It might be remarked that there are some programs which are double offenders in the portability stakes; those which gaily use the subrange construct in a set! Thus:

if ch in ['+' .. ';'] then

Alternatives

Having suggested that the set of char is at present a very poor type to use in a PASCAL program, I ought to indicate some alternatives. While these may lose somewhat in efficiency, it must be borne in mind that portability always has its penalties, and also that efficiency in speed usually only matters in a few critical parts of a program.

The first obvious alternative is to replace each in test (the most usual construct) by a boolean function. The machine-dependencies (if they exist) are then confined to a few places which may be well documented, and are easy to change. An alert programmer might even supply alternatives specialized for a particular computer (like the interchangeable camera lens market).

The second alternative is to examine the uses the construct is put to and see whether or not the requirement is to classify the character into one of a small number of classes (for example: alphabetic, digit, operator, etc). The desired effect may then be achieved by either a function that returns the scalar type value corresponding to the character, or an array might be set up to give the class when indexed by a char. Regrettably, PASCAL does not allow the setting up of read-only arrays, and this will have to be done in a machine-dependent initialization routine.

Examples:

```

type
  charclass = (alphabetic, numeric, operator, point, other);
var
  classvector : array[char] of charclass;
function
  classify(ch:char) of charclass;
begin
  case ch of
    'A','B','C','D', {laziness} 'Y','Z':
      classify:=alphabetic;
    '0','1','2','3','4','5','6','7','8','9':
      classify:=numeric;
    '+','-','*','/':
      classify:=operator;
    '.':
      classify:=point;
  else: {non-standard PASCAL}
      classify:=other
  end; { of case }
end; { of classify }
begin
  ...
  if classify(nextchar) = numeric then ....
  while classvector[nextchar] in [alphabetic,numeric] do ....
  ...
end.

```

Set size

The related question of what set size can be reliably assumed to be available is very difficult to answer. I would assume that 32 bits would be safe enough for large/medium computers (the usual word sizes being 32, 36, 48 & 60 bits), but mini-computers pose more of a problem. Diffidently, I suggest that 32 bits be regarded as the minimum set size limit for a compiler to be considered as implementing a compatible PASCAL. Most minis can do this with a double-word. Sets larger than this should be clearly marked in the commentary of a supposedly portable program.

Arthur Sale
 Professor of Information Science
 University of Tasmania
 (Burroughs B6700 implementor)



IMPLEMENTATION NOTE

1977 February 16

BURROUGHS B6700 PASCAL: THE FOR STATEMENT

Introduction

This note describes the implementation of the for statement of PASCAL in the compiler for the B6700/B7700 computers, as developed at the University of Tasmania.

Defining standards

The for statement is syntactically described in the Revised Report (section 9), but its semantic description in the Report (section 9.2.3.3) is plainly wrong, and in fact not compatible with the further explanations in the User Manual since it does not address several problem areas.

The PASCAL User Manual on p 24 adds more explanation, but in a loose discursive fashion which leaves many things unclear. Subsequently, on p25, a more exact definition is given in terms of equivalent PASCAL, which is the clearest of the lot. This definition will be taken as the important one.

To quote a critical section:

"A for statement of the form:

for v:=e1 to e2 do S

is equivalent to the sequence of statements:

if e1 <= e2 then

begin v:=e1; S; v:=succ(v); S; ...; v:=e2; S

end

{ at this point, v is undefined }"

Side-effects

PASCAL is generally silent on the effects of side-effects, and on the evaluation order of sub-expressions. However, from the above expression, it is clear that the two expressions are to be evaluated before an assignment is made to the control variable. Logically, this is a desirable interpretation: it implies that the limits of the loop are computed, and then only is it entered.

It follows that the definition of PASCAL ought to explicitly give this sequence, rather than leave it to be implied.

Undefinition

The definition (and the preceding paragraph) state that after the execution of a for statement (provided the statement is not left by a goto) the value of the control variable is undefined. The primary purpose of this undefinition is to allow implementors freedom to implement the loop efficiently. PASCAL programmers should not therefore presume any particular value in the control variable after it has been used in a for statement. Of particularly nasty characteristics are the compilers which may leave it set at succ(e2), since this may be out-of-range of its type.

In the Burroughs B6700/B7700 computers, it is easy to prevent programmers from doing any computation with this variable until it has been re-defined by setting it to a tag-six word (uninitialized operand). This value can be overwritten by a legal value, but causes a machine interrupt if the variable is used in an operator context. This is done for all for statements in B6700/B7700 PASCAL, and the illegal use of the variable cannot therefore be permitted.

It should also be pointed out that the definition of a for statement allows the control variable to be undefined whether or not the body of the loop is ever entered. B6700 PASCAL treats both cases the same, unlike some other compilers which take advantage of the implementation freedom to leave the control variable unchanged, or at e1, if the loop body is never entered.

Internal change to the control variable

The User Manual, on p24, explicitly forbids the alteration of the control variable by any statement in the body of the loop. Such illegal constructs are hard to detect as they may occur in the body of a procedure or function. On the B6700 computer it is possible to detect this occurrence at run-time with a small time penalty, under some circumstances.

If the loop is capable of being optimized to use the STEP-INDEX facility (which implies that e1 and e2 are in the range 0..65535, and the loop is a TO-loop), then a STEP-INDEX-WORD (SIW) is stored in the control variable v. All read-accesses of v return the (integer) value of v, without the final or increment fields, but write-accesses destroy the tag field. Thus when the loop incrementation point is reached, the STBR instruction causes abnormal termination of the loop, and a call

is made to the PASCALERROR routine to kill the program.

The detection facility is not available if the loop is a DOWNTO loop, or if it cannot be simply optimized.

Summary

In this case, and others, the B6700/B7700 PASCAL compiler enforces strict adherence to standard PASCAL. Hardware checks make this possible with negligible time penalty. Programs written in B6700 PASCAL therefore have a higher probability of being portable in this respect than would be the case for many other PASCAL compilers. There is one unfortunate effect, however: non-standard PASCAL programs are less likely to execute in B6700 PASCAL since it is such a searching test.

Implementors of PASCAL are invited to send me answers to the following questions about their compilers. The invitation is also extended to users as implementors are notoriously unreliable correspondents.

1. In what order are e1, e2, and the assignment of v carried out?
Does this differ with the form of the loop?
2. What value is left in v if the loop is never entered?
3. What value is left in v if the loop is entered?
4. What happens if the control variable is altered (or a limit-variable)
 - (a) from a piece of code compiled in the body, and
 - (b) from a procedure called in the body?
5. Are there different (optimized) forms of for-statement?
How do they differ?
6. Are there any limits on the number of repetitions or size of the limit-expressions?

If code-templates could be attached (with explanations) this might be useful too. If sufficient information is received, it may be possible to prepare a summary for PUGN.



Arthur Sale
Professor Information Science
University of Tasmania
(Burroughs B6700 implementor)

APPENDIX
B6700 FOR STATEMENT CODE TEMPLATES

CASE 1 (unoptimized)

e1
e2
NAMC(TEMP)
STOD
↓
NAMC(V)
STON

(* store into V, but leave on TOS *)

TL: →

VALC(TEMP)
LSEQ
BRFL(EL)
...
body
...
VALC(V)
ONE
ADD
NAMC(V)
STON
BRUN(TL)

(*GREQ if DOWNT0 loop*)

(*test whether to exit the loop*)

(* increment *) (* SUBT if DOWNT0 loop *)

(* store, but leave on TOS *)

(* go do the test again *)

EL: ←

ZERO
LT8(6)
STAG
NAMC(V)
STOD

(* now got a tag-six word on TOS *)

(* into the control variable *)

CASE 2 (unoptimized, but was
a potential case for optimization)

e2
NAMC(TEMP)
STOD
LIT(*e1*)

APPENDIX (continued)

CASE 3 (constant bounds)

LIT(1/*e2/e1*)
↓
LT8(4)
STAG
NAMC(V)
STOD

(* store the SIW into V *)

TL: →

...
body
...
NAMC(V)
STBR(EL)
BRTR(TL)
MKST
NAMC(PASCALERROR)
LT8(4)
ENTR

(* increment, store, and test *)

(* branches unless SIW overwritten *)

(* causes pascalerr(4) *)

EL: ←

ZERO
LT8(6)
STAG
NAMC(V)
STOD

(* undefine the control variable *)

CASE 4 (first bound constant,
second capable of optimization)

e2
DUPL
LIT(*e1*)
LESS
BRTR(EL)
LIT(*e1*)
EXCH
INSR(35:16)
ONE
INSR(47:12)

NOTE TO PUGN

INTERIM REPORT - IMPLEMENTATION OF FOR-STATEMENT - 1

The note gives some comparative details on the implementation of for-statements in two PASCAL compilers. As more information becomes available, it will be added to the list. See my earlier comments in a Note to PUGN on the Burroughs B6700 implementation.

BASIC TEMPLATE

for v:=e1 to e2 do s;

PASCAL-6000 (CDC Cyber range)

The implementation produces code which is equivalent to the following:

```
let temp1 = a register;  
temp2 = a temporary stack location;
```

```
temp1:=e1;  
temp2:=e2;  
while temp1 ≤ temp2 do begin  
v:=temp1;  
s;  
temp1:= v+1;  
end;
```

The consequences of this code on the precise action of the loop with the three questions I posed are:

- (i) the two expressions are computed before an assignment, so that v:=1; for v:=v+1 to v+10 do s; will count from 2 to 11.
- (ii) The exit value of v if the loop is never entered is its value before the loop is reached.
- (iii) The exit value of v is e2 if the loop is ever traversed.

In addition, alterations of v from within the body of the loop do in fact alter the progress of counting, if they can be achieved.

PASCAL for Burroughs B6700/B7700 (Tasmania)

More details are given in the Note mentioned before. The code is generally equivalent to:

```
let temp1 = a temporary stack location;  
temp2 = a temporary stack location;
```

```
temp1:=e1;  
temp2:=2;  
v:=temp1;  
while v ≤ temp2 do begin  
s;  
v:=v+1;  
end;  
v:=invalidtagsixvalue;
```

The answers are again:

- (i) as for PASCAL-6000.
- (ii) + (iii) In all cases the exit value of v is a special word which cannot be utilized as a value, but can be overwritten with a proper value.



IMPLEMENTATION NOTE

1977 February 17

B6700/B7700 PASCAL : ELSE IN CASE

Introduction

Many PASCAL implementations are inserting an ELSE clause in the CASE statement of PASCAL. This note puts the cases for and against, and proposes a pseudo-standard for any such implementations so that maximum compatibility between PASCAL compilers can be achieved.

Against

The case against having an ELSE clause in a case statement is that it encourages a programmer to use the clause through laziness simply to save writing a long list of alternatives. Thus when an unexpected value of the case expression occurs, it is processed erroneously by the ELSE clause, rather than being one of the 'undefined' areas of PASCAL. The arguments here rest on implementors choosing to detect values of the case expression which do not match any label, and choosing to make such occurrences definite run-time errors. Such an interpretation is not mandatory.

For

The arguments for an ELSE clause are regularity, and robustness. The regular argument comes from (i) examination of languages of similar age and utility, in most of which the feature appears, (ii) the analogy with if-then-else which may be viewed as a special version of case, and (iii) actual thought habits of good programmers.

The robustness argument derives from the need to be able to write programs which are robust against all input, and all circumstances, and from the difficulty of handling all case statements without error. Long lists of labels are error-prone, and sometimes inappropriate. If the intention is that all values other than a specified few are to be similarly treated, then it ought to be possible to specify this.

The B6700 implementation

The implementation of PASCAL for the Burroughs B6700/B7700 computers developed at the University of Tasmania contains such an ELSE facility. The semantic features of this implementation are suggested as a pseudo-standard for PASCAL implementors who also agree that this is a necessary feature.

A case without else

If no else appears in a case statement, the B6700 implementation will raise a run-time error event, and terminate the program, if the case expression evaluates so as to match no case label.

Recommendation 1
That all implementations of PASCAL regard the above as the preferred semantics of this situation arising in a case statement.

A case with else.

If an else clause appears in a case statement, then the B6700 implementation transfers control to the else clause for all values of the case expression which do not match an explicit case label. In all other respects an else clause behaves as a labelled clause.

Recommendation 2
That the above be regarded as the minimum semantic requirements of an else-clause in a case-statement. If an implementation can cause the same effect as in Recommendation 1 for values of the case-expression which are outside its declared range (as in the type), they are encouraged to do so. This is relevant only to implementations that include an else-clause.

Syntax of else-clause

In the B6700 implementation, an ELSE can appear wherever a case-label can, except that there can be at most one in any case statement. Thus an ELSE may appear in a case-label list, though it is difficult to see why this would be done. This syntax is very easy to accommodate, and requires minimal changes to the CASESTATEMENT routine in PASCAL-P4. There are no other syntactic changes.

Recommendation 3
If an implementation adopts an ELSE-clause, then the above syntax should be regarded as standard. Modified syntax diagrams are attached.

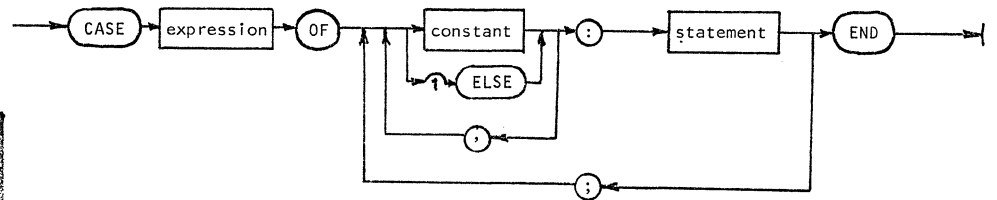
Stylistics

The preferred style for a case statement containing an else clause has the else clause last, following all labelled clauses.

Example of case with else

```
case of
  '+', '-', '*', '/':
    thing := arithmeticoperator;
  '.', '+', '[' , ']':
    thing := variableevaluator;
  ',', ';':
    thing := separator;
else:
  thing := otherthing
end;
```

Arthur Sale
Professor of Information Science
University of Tasmania
(Burroughs B6700 implementor)



MODIFIED SYNTAX CHART FOR CASE-STATEMENT IN WIRTH-FORM

Variable-parameters in Pascal

Bill Findlay,
Glasgow University, Glasgow G12 8QQ, Scotland, U.K.

The impression that variable-parameters in Pascal must be passed by reference is widespread (e.g. it appears in the books by Conway, Gries and Zimmerman and by Webster). However, I believe it to be a misconception stemming from the fact that all existing implementations have used reference passing. Many other controversies in the Pascal Newsletter arise from this failure to distinguish between language and implementation. My understanding of the matter is that (as in Fortran and for the same reason) both reference and value-result are valid mechanisms for variable-parameters.

If we look at Section 9.1.2 of the Report we find only that the formal "represents" the actual during the execution of the procedure. Name binding is disallowed (thank Heavens!) by the rule that the index of a subscripted variable-parameter is evaluated just once, but reference is not specified.

In the Axiomatic Definition, at axiom 11.2, it is stated that the variable-parameters and non-local variables accessible by a procedure-call must be distinct (no "aliasing"). Given this condition, it is not possible to determine the parameter-passing mechanism by running a legal program. I conclude that any method which satisfies axiom 11.2 is allowable.

This issue is not just of theological interest. The implementor has been given an important degree of freedom: he can copy the technique used by the Fortran system on his machine and thereby gain access to the enormous investment in Fortran library routines.

Interacting with a PASCAL program - D. A. Joslin, University of Sussex, Computer Centre, Brighton, U.K. 18/5/77

The requirement of the Revised Report that INPUT be defined right from the very start of a program (more generally: f↑ is defined immediately after READ(f), and READ(f,x) ≡ x := f↑; GET(f)) results in the first card being physically read into a buffer when the program is entered, the second card being physically read on the first READLN, and so on. An interactive program, however, normally outputs some message to the terminal before expecting the user to type his first input: it proceeds in a question/answer/response mode. This can be achieved in PASCAL provided that:

- (i) the Operating System is instructed to satisfy the program's first card-read request by any dummy record (which the program will not actually process), and second and subsequent requests by terminal input;
- (ii) the program precedes each READ, ie each request for an answer from the terminal user, by a READLN. This can conveniently be done by means of procedures - eg


```

PROCEDURE GENERAL(VAR X:REAL);
BEGIN READLN; READ(X) END;
. . . . .
WRITELN('TYPE A REAL NUMBER');      (* question *)
GENERAL(X);                          (* get answer *)
WRITELN('X =', X);                  (* response to answer *)

```

The attached sheet shows: a sample program CC2LINT written according to rule (ii) above; a George 3 macro INTERACT which performs the action of rule (i) above - the command INTERACT is given by the terminal user after he has loaded a program he is to interact with; a sample teletype session showing interaction with CC2LINT via INTERACT.

#LISTING OF IT CC21INT(1/) PRODUCED ON 4MAY77 AT 11.49.48
 #OUTPUT BY LISTFILE IN ':T.CC21' ON 13MAY77 AT 12.20.44 USING U14
 DOCUMENT CC21INT

```

PASCAL COMP, OBJECTCC21INTPRG
TYPE STRING = PACKED ARRAY [1..16] OF CHAR;
VAR S: STRING; J,K: INTEGER; C: CHAR;
PROCEDURE GETSTRING(VAR S: STRING);
VAR I: INTEGER; A: ARRAY [1..16] OF CHAR;
  BEGIN
  READLN;
  FOR I:= 1 TO 16 DO READ(A[I]);
  PACK(A,1,S);
  END;
PROCEDURE GETINTEGER(VAR I: INTEGER);
  BEGIN
  READLN;
  READ(I);
  END;
PROCEDURE GETCHAR(VAR C: CHAR);
  BEGIN
  READLN;
  READ(C);
  END;
BEGIN
WRITELN('HI THERE - PLEASE TYPE IN YOUR NAME!');
GETSTRING(S);
WRITELN('GLAD TO MEET YOU, ',S);
REPEAT WRITELN('TYPE IN A WHOLE NUMBER!');
  GETINTEGER(J);
  WRITELN('AND ANOTHER');
  GETINTEGER(K);
  WRITELN('YOUR NUMBERS ARE ',J,' AND ',K);
  WRITELN('THEIR SUM IS ',J+K,' AND THEIR DIFFERENCE IS ',J-K);
  WRITELN('SHALL WE TRY AGAIN?');
  GETCHAR(C);
  UNTIL C='Y';
WRITELN('GOODBYE, ',S);
END.
****

#?#?
PASCAL COMPILER #PASQ/ZA (SUSSEX VERSION 001) ON 04/05/77 AT 11/49/44
OPTION(S) SELECTED: NONE
  
```

```

12.19.51- LISTFILE INTERACT,NUMBER
0 #MACRO INTERACT - D.A.JOJSLIN, 04MAY77
1 (L *LP)
2 CE !
3 IN !,T????
4
5 ????
6 AS *CR0,!
7 ER !
8 RP CE,DP,LS,OL,PM
9 EN 0
10 IF FAIL(FILE *CR0), OL *CR0
11 RP FB,CM
12 RM
  
```

```

12.20.52- LOAD CC21INTPRG,CJRESK
12.21.05- INTERACT
HI THERE - PLEASE TYPE IN YOUR NAME
- DAVID
GLAD TO MEET YOU, DAVID
TYPE IN A WHOLE NUMBER
- 123
AND ANOTHER
- 67
YOUR NUMBERS ARE 123 AND 67
THEIR SUM IS 190, AND THEIR DIFFERENCE IS 56
SHALL WE TRY AGAIN?
- YES
TYPE IN A WHOLE NUMBER
- -1
AND ANOTHER
- 77
YOUR NUMBERS ARE -1 AND 77
THEIR SUM IS 76, AND THEIR DIFFERENCE IS -78
SHALL WE TRY AGAIN?
- YES, ONCE MORE
TYPE IN A WHOLE NUMBER
- 5
AND ANOTHER
- 903
YOUR NUMBERS ARE 5 AND 903
THEIR SUM IS 908, AND THEIR DIFFERENCE IS -898
SHALL WE TRY AGAIN?
- NO THANK
GOODBYE, DAVID
12.26.03 FREE *LPD ,18 TRANSFERS
12.26.06 FREE *CR0 ,10 TRANSFERS
0.03 :HALTED : OK
END OF MACRO
  
```

MACHINE DEPENDENT IMPLEMENTATIONS

---Pascal Implementations Summary---

(* This section summarizes all the information that we have on all Pascal implementations, in the checklist format. *)

Amdahl 470

(* See implementation notes for IBM 360/370. *)

Burroughs B1700

In a letter dated November 3, 1976, Tony Gerber (Basser Dept. of Computer Science, School of Physics, University of Sydney, Sydney, N. S. W. 2006, Australia; Tel. 629 1122) reported several persons who have worked on B1700 implementations. They are:

Elliott Organick's group at the University of Utah, using Brinch Hansen's Sequential Pascal.

P. Schultess and K. Hauserman at the University of Zuerich, who each worked on (separate) projects.

P. Albrich, University of Karlsruhe, Germany, was working with Concurrent Pascal.

M. Ellison at the University of Newcastle-upon-Tyne was using Pascal-P "version 1.0".

Burroughs B3700, B4700

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. R. M. Lansford; 3620 Greenhill Rd.; Pasadena, CA 91107; 213/ 351-0206. P. L. McCullough; Tektronix 60/666; P.O. Box 500; Beaverton, OR 97077 (503/638-3411 x2397). W. C. Price; 28282 SW Mountain Road; West Linn, OR 97068 (503/644-0161).

2. MACHINE. Burroughs B3700, B4700 (with Accumulator operator.)

3. SYSTEM CONFIGURATION. MCVP 5.7 and Time Sharing System

4. DISTRIBUTION. No plans at present--the need has not arisen.

5. DOCUMENTATION. Forward to program listing; in form of supplement to Pascal User Manual and Report. (* This is apparently not machine retrievable. *)

6. MAINTENANCE. None. Development has terminated. "If you find'em, fix'em."

7. STANDARD.

Unimplemented:	Extensions:
real arithmetic	segmentation
formal procedures and functions	symbolic procedure call tracing
files (except text files INPUT and OUTPUT)	stack checking and statistics
	packing is optional

8. MEASUREMENTS.

Pass 1: 4000 lines of Pascal, compiled @ 1000 lines/min.

Pass 2: 2500 lines of BPL, taking 45 sec. to generate code for Pass 1 of the compiler.

A minimum of 110K bytes is needed for a logical (reasonable) segmentation of the compiler. (* Size and execution speed of code produced not reported. How this compares to FORTRAN and other languages not reported. *)

9. RELIABILITY. Good, but not excellent. (* Number of sites using compiler not reported. Date first released not reported. *)

10. DEVELOPMENT METHOD. Compiler was bootstrapped from an early P1 compiler obtained from CalTech. The compiler consists of two passes. The first is written in Pascal and emits augmented P-code. The second pass (written in BPL, a PL/360-like assembler) generates 4700 code from the P-code. The first version of the compiler was written by Mike Mahon in 2 person-months. An additional 8 person-months have been expended in teaching the compiler about such things as optimal variable size and alignment, segmentation, etc.

11. LIBRARY SUPPORT. (* No information provided. *)

Tektronix, Inc.

June 8, 1977

Mr. Andy Mickel
PASCAL User's Group
University Computer Center
227 Experimental Engineering Building
University of Minnesota
Minneapolis, Minnesota 55455

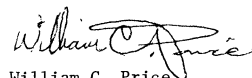
Dear Andy:

Thank you for the incredible amount of effort you have put into making PUG work. Please, however, don't use anymore of that ugly chartreuse paper.

As to the Burroughs B3700/B4700 PASCAL implementation reported by Dr. Lansford in PUG#8: Due to the efforts of Burroughs' management, the (spare-time) project has been cancelled. We understand that inquiries through Burroughs Medium Systems Plant have been answered with "Ask your local Burroughs representative." The reports we promised on certain interesting aspects of our implementation (segmentation, optimization, augmentation of P-code, etc.) have been delayed (perhaps indefinitely), as we are no longer associated with Burroughs Corporation.

Herewith, however, is a short comment arising from our attempt at understanding the full implications of PASCAL's file structure.

Truly,



William C. Price
Instrument Research Group
Tektronix Laboratories

WCP:pt
Attachment
cc: Dr. R.M. Lansford
P.L. McCullough

Burroughs B5700

Bruce A. Pumplin, Department of Computer Science, University of Wisconsin - Eau Claire, Eau Claire, WI 54701, has promised us a report on the progress of his Pascal-P based implementation for the B5700. Last we knew (76/08/25), the compiler-interpreter was working.

Burroughs B6700/7700 (Tasmania)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. A.H.J. Sale; Dept. of Information Science; University of Tasmania; Box 252C G.P.O.; Hobart, Tasmania 7001 Australia; STD 002 23-0561 x435.

2. MACHINE. Burroughs Model III B6700, B7700

3. SYSTEM CONFIGURATION. Burroughs MCP version II.8 (with few (minor) local mods). Minimal system to operate not known, but unlikely to be any B6700 that small--storage demands are low, and little else is critical.

4. DISTRIBUTION. Both 7 and 9 track magnetic tapes available. (* Cost not reported. *)

5. DOCUMENTATION. Supplement to Pascal User Manual and Report available; a dictionary-style "Reference Manual" is in preparation but is not yet complete (77/4/20). (* Not known if this documentation is machine retrievable. *)

6. MAINTENANCE. To be maintained for teaching use within the University as well as larger aims. Reported bugs will be fixed as soon as possible, with patch notices to users. Duration of support not yet determined; several other developments are also pending.

7. STANDARD. Restrictions: 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 I/O scheme is like Pascal-1. Extensions: Various reserved words, character set transliterations. Burroughs comment facility. ELSE in CASE. File attributes in declaration. Format declarations. Extensive Burroughs-compatible compiler options. (Pascal control comment option mode not implemented).

8. MEASUREMENTS.

compiles about 20% slower than FORTRAN or ALGOL, but in about 2/3 of their space (for test programs about 4-5 K 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.

9. RELIABILITY. Excellent. Only one system crash during testing attributed to Pascal. Compiler now in use at 3 sites. Compiler has been in use since 76/10. First released to outside sites in 77/4.

10. DEVELOPMENT METHOD. Compiler which generates B6700 code-files which are directly executed by the B6700 with MCP. Written entirely in B6700 ALGOL. Hand-coded using Pascal-P as a guide/model. All other paths offered much more difficulty due to special nature of machine/system. Person-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).

11. LIBRARY SUPPORT. There is as yet no BINDINFO in the code-file so that it is not possible to link Pascal to modules compiled by other language processors, but the system contains an extended set of predefined mathematical functions.

Burroughs B6700 (San Diego)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Distributor: Henry Fischer; UCSD Computer Center; University of California - San Diego; La Jolla, CA 92093; 714/452-4050. Implementors: Mark Overgaard; Jim Madden: same site.

2. MACHINE. Burroughs B6700

3. SYSTEM CONFIGURATION. (* No information provided. *)

4. DISTRIBUTION. Scheduled to start in mid-summer, 1977. (* Information on cost, magnetic tape formats, etc. was not provided. *)

5. DOCUMENTATION. (* No information provided. *)

6. MAINTENANCE. Unknown at this time.

7. STANDARD. (* No information provided. *)

8. MEASUREMENTS. Current compile speed is 5000 line/min; but expected improvements could make that 10,000 lines/min--as fast as the Burroughs Fast Algol compiler. (* Size and execution speed of code produced not reported. *)

9. RELIABILITY. Unknown at this time. (* Number of sites using this compiler not provided. Date of first release not reported. *)

10. DEVELOPMENT METHOD. Real compiler, written in Pascal which produces native code for the B6700. (* Person-hours to create compiler not reported. *)

11. LIBRARY SUPPORT. (* No information provided. *)

Burroughs B6700 (New Zealand)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Chris Bishop; Computing Centre; University of Otago; P. O. Box 56; Dunedin; NEW ZEALAND; (Tel. Dunedin 40109 x890).

2. MACHINE. Burroughs B6700

3. SYSTEM CONFIGURATION. (* No information provided. *)

4. DISTRIBUTION. Tapes can be written in any of the following formats:

- a) 1600 bpi, PE, 9 track, B6700 library tape
 - b) 800 bpi, NRZ, 9 track, B6700 library tape
 - c) 1600 bpi, PE, 9 track, USASI Multi-file tape
 - d) 800 bpi, NRZ, 9 track, USASI Multi-file tape.
- (* Costs for tapes not reported. *)

5. DOCUMENTATION. Brief notes on usage available. (* Not known if this is machine retrievable. *)

- 6. MAINTENANCE. (* No information provided. *)
- 7. STANDARD. (* No information provided. *)
- 8. MEASUREMENTS.
 - compilation space-- (* No information provided. *)
 - compilation speed--Compiles the Karlsruhe B6700 compiler in 90 sec. of processor time.
 - execution speed-- (* No information provided. *)
 - execution space-- (* No information provided. *)
 (* How this compares to FORTRAN and other languages not reported. *)
- 9. RELIABILITY. Unknown at this time. Compiler in use at 3 sites. (* Length of time compiler has been in use not reported. *)
- 10. DEVELOPMENT METHOD. Karlsruhe B6700 compiler-interpretor translated from Pascal source to Burroughs Extended Algol. Produces symbolic code for a hypothetical stack machine. This symbolic code must be assembled to produce absolute machine code which may then be interpreted. Both the assembler and interpretor are written in Extended Algol. It is planned to convert this Algol version into a true compiler for the B6700; work will start in earnest about July of 1977. (* Person-hours to create compiler not reported. *)
- 11. LIBRARY SUPPORT. (* No information provided. *)

Burroughs B6700 (Helsinki)

- 1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Antti Salava; formerly at Dept. of Computer Science; University of Helsinki; Toolonkatu 11, SF-00100; Helsinki 10, Finland; Present address: Munkkiniemen Puistotie 17-A13; SF-00330 Helsinki 33, Finland; phone: 90-486288.
- 2. MACHINE. Burroughs 6700
- 3. SYSTEM CONFIGURATION. (*Unknown *)
- 4. DISTRIBUTION. None; project not yet complete.
- 5. DOCUMENTATION. We are currently (7/7/17) preparing a report on our Pascal implementation. (* Not known if this will be machine retrievable. *)
- 6. MAINTENANCE. None, project not yet complete.
- 7. STANDARD. (* No information provided. *)
- 8. MEASUREMENTS. Unknown; project not yet complete.
- 9. RELIABILITY. Unknown; project not yet complete.
- 10. DEVELOPMENT METHOD. The compiler is written in Burroughs Extended Algol and generates B6700 machine code. (* Person-hours to create compiler not reported. *)
- 11. LIBRARY SUPPORT. (* No information provided. *)

CDC Cyber 18 and 2550

- 1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Jim Fontana, Control Data Corporation, 3519 W. Warner Ave., Santa Ana, CA 92704 (714/754-4102).
- 2. MACHINE. Control Data Cyber 18 and 2550.

- 3. SYSTEM CONFIGURATION. (* the Cyber 18 is a self contained interactive system. *) Dennis Nicolai (CDC, Minneapolis) reports that the Cyber 18 and the 2550 have similar instruction sets, and that the compiler is a cross-compiler which runs on Cyber 70's and 170's. Code is linked and 'down loaded' to the Cyber 18 and 2550.
- 4. DISTRIBUTION. Control Data Corporation.
- 5. DOCUMENTATION. CDC Manual 88988500 A. (* Apparently no machine retrievable documentation available. *)
- 6. MAINTENANCE. CDC supported Communications Front End software.
- 7. STANDARD. Unrevised Pascal language definition with extensions. I/O is hardware defined.
- 8. MEASUREMENTS. (* No information available. *)
- 9. RELIABILITY. Excellent. (* Number of sites using system not reported. Date of first release not reported. *)
- 10. DEVELOPMENT METHOD. The compiler is derived from the compiler for the CDC 2550 front end processor, which in turn was derived from the old Zurich Pascal-6000 (1972) compiler.
- 11. LIBRARY SUPPORT. (* No information available. *)

CDC 3200

A local rumor is that John Urbanski, West Bank Computer Center, 90 Blegen Hall, University of Minnesota, 269 19th Ave. South, Minneapolis, MN 55455 USA (612/373-3608), is working on an implementation of a subset of Pascal for the CDC 3200.

CDC 3300

We have not heard any news from either of the following two implementors for over two years, in spite of several attempts by us and others to reach them.

P. J. Voda, Computing Research Centre, Dubravska 3, 885 31 Bratislava, Czechoslovakia, has a version of Pascal operational on the 3300. This version includes concurrent constructs (not the same as Brinch Hansen's), and several large software projects were implemented using it.

Lou Beverino, Computer Center, California State University, Northridge, CA 91324, is known to have received Pascal-P2.

CDC 3600

This is another case of the "two-year silence" (see CDC 3300). You are welcome to try contacting Marcel Dupras, Institut de Programmation, Tour 55-65, 11-Quai Saint Bernard, F-75 Paris, France, who was listed by George Richmond as having completed an implementation on the 3600.

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER.

Distributors:

- (Europe, Asia, or Africa)
Urs Ammann
(* same address as implementor *)
- (North or South America)
George H. Richmond
Computing Center: 3645 Marine St.
University of Colorado
Boulder, CO 80309 USA
303/ 492-8131
- (Australia, New Zealand, or Oceania)
Carroll Morgan
Basser Dept. of Computer Science
University of Sydney
Sydney, N.S.W. 2006
Australia
629 1122

Implementor:

Urs Ammann
Institut fur Informatik
E.T.H. -Zentrum
CH-8092 Zurich
Switzerland
01/ 32 62 11

Maintainer:

John P. Strait / Andy Mickel
University Computer Center
227 Experimental Engineering Bldg.
208 SE Union St.
University of Minnesota
Minneapolis, MN 55455
USA
612/ 376-7290

2. MACHINE. Control Data 6000 series, Cyber 70 series, and Cyber 170 series.

3. SYSTEM CONFIGURATION. Minimum central memory-49K words. Operates under Scope 3.4 and Kronos 2.1.

4. DISTRIBUTION. Tape format is Scope 3.4 internal binary, 7 track, unlabelled, 800 bpi. Specify: person responsible for maintaining the system, your hardware, operating system, and character set (ASCII or Scientific, 63 or 64). From Switzerland cost is S.Fr. 100 (includes cost of tape; do not pay in advance, you will be billed); from Colorado cost is \$60 for new recipients (includes tape and documents), and \$35 for old recipients (includes tape but not documents); from Australia cost is \$A30 (tape and documents). New installation notes will be machine retrievable in Release 3.

5. DOCUMENTATION. Machine retrievable supplement to Pascal User Manual and Report and documentation of library support package will be available with Release 3.

6. MAINTENANCE. Will accept bug reports at Minnesota for foreseeable future. Expect to issue Release 3 in 1978.

7. STANDARD. Nearly full standard. Restrictions include: standard procedures and functions cannot be passed as actual parameters; file of file is not allowed. Extensions include: additional predefined procedures and functions; segmented files.

8. MEASUREMENTS. Compilation speed: 10500 characters per second on a Cyber 74; 54 seconds to compile the compiler. Compilation size: 46K (octal) words for small programs; 57K for self-compilation. Execution speed: see 7600 statistics, below. Execution size: binaries can be as small as 2.4K, compared with Fortran minimum of over 10K.

9. RELIABILITY. Excellent. The compiler is in use at 139 known sites. First version of this compiler was operational in late 1970. The present version was first released in May 1974.

10. DEVELOPMENT METHOD. Bootstrapped from the original Pascal-6000 compiler, but developed in a 6 phase stepwise refinement method. Approximately 1.5 person years.

11. LIBRARY SUPPORT. Allows calls to external Pascal and assembler subprograms and Fortran (FTN) subprograms. The user library supplied with the system contains many routines in addition to the standard.

(* See announcement elsewhere in this issue. *)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. This compiler is essentially the Pascal 6000 compiler modified to fit the 7600 and Cyber 76 machines. The run time system is based on that of Hans Joraanstad at CERN, Geneva, Switzerland (see Pascal News #4). Improvements by H. D. Ellison; A.P. Hayes; UMRCC; Oxford Road; Manchester M13 9PL; England, U.K.; (061-273 8252).

2. MACHINE. Control Data 7600 & Cyber 76.

3. SYSTEM CONFIGURATION. SCOPE 2.1.3, 32K SCM.

4. DISTRIBUTION. Contact R. J. Collins at address above. A distribution agreement must be signed and the cost is 50 pounds sterling.

5. DOCUMENTATION. Same as Pascal-6000.

6. MAINTENANCE. The situation is unclear at present. UMRCC will assist with bugs -- in the 7600 dependant code (runtime system) only. Currently UMRCC and Minnesota will work together on a common release for Release 3.

7. STANDARD. Same as Pascal 6000.

8. MEASUREMENTS. Compilation speed is about 57,000 characters/sec. Compiler compiles itself in less than 10 sec. Pascal execution speed has been measured by using the obvious encoding in Pascal of Wichmann's Synthetic Benchmark (see Computer Journal Vol. 19, #1). The Units are in kilo Whetstones.

compiler and optimisation level	no runtime checking	array bound checking
ALGOL 4 (OPT=5)	1996	1230
Pascal	6850	6240*
FTN (OPT=2)	945	3174**

* Using I+ option--all run time checks included.
** Forces OPT=0.

Compiler will recompile itself on a 'half-size'(32K SCM) machine. (* No information provided on size of compiler or object code produced. *)

9. RELIABILITY. 3 sites; as reliable as Pascal 6000 (Zurich). (* Date of first release not reported. *)

10. DEVELOPMENT METHOD. Cross compiled from Cyber 72 compiler. Based on Zurich 6000 compiler with necessary additions for this machine. (* Person-hours to develop compiler not reported. *)

11. LIBRARY SUPPORT. Same as Pascal 6000.

(* See implementation notes for IBM 360/370. *)

National Aeronautics and
Space Administration



Langley Research Center
Hampton, Virginia
23665

Reply to Attn of 125A

JUN 24 1977

Dear Andy:

This is to inform you that a PASCAL implementation has been completed for the CDC STAR-100. The details are:

- 1. Implementors: Douglas D. Dunlop
Dept. of Mathematics
College of William & Mary
Williamsburg, VA 23185

John C. Knight
Analysis and Computation Division
NASA Langley Research Center
Hampton, VA 23665
- 2. Language: The PASCAL P4 subset of PASCAL.
- 3. Machine: Control Data Corporation STAR-100.
- 4. Operating System: STAR O/S.
- 5. Documentation: At present, only the compiler listing.
- 6. Reliability: Compiler correctly compiles itself.
- 7. Distribution: No formal mechanism. Write if you are interested.
- 8. Implementation: The compiler was developed from PASCAL P4. Two forms exist and both compilers generate STAR machine code. They are a 6000 based cross compiler which produces object modules for input to the STAR loader, and a STAR resident compile and execute system.

Our experience with PASCAL P4 has been very satisfactory and we congratulate the developers. In less than six man weeks of effort, the PASCAL P4 compiler was modified to generate STAR-100 machine code, and the compiler which was produced successfully compiled itself on the STAR-100.

Sincerely,

John C. Knight
Aerospace Technologist
Programing Techniques Branch

- 1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Olivier Lecarme; Universite de Nice; Laboratoire D'Informatique; Parc Valrose, 06034 Nice Cedex; France (51 91 00).
- 2. MACHINE. CII IRIS 50.
- 3. SYSTEM CONFIGURATION. Siris 3 operating system. (* Minimum hardware requirements not known. *)
- 4. DISTRIBUTION. (* Unknown, project not yet complete. *) Expected to be available by end of 1977.
- 5. DOCUMENTATION. (* No information provided. *)
- 6. MAINTENANCE. (* Unknown, project still underway. *)
- 7. STANDARD. Will implement exactly Standard Pascal.
- 8. MEASUREMENTS. (* Unknown, project not yet complete. *)
- 9. RELIABILITY. (* Unknown, project not yet complete. *)
- 10. DEVELOPMENT METHOD. Various approaches tried. Tool compiler developed using Pascal-P, Pascal-E subset, intermediate machine oriented languages, and the Nagel trunk compiler used to write a true compiler. (* Person hours to implement system not reported. *)
- 11. LIBRARY SUPPORT. Will produce modules for the linkage editor. (* No information provided on external and other language subroutines, separate compilation, automatic source inclusion, or symbolic post-mortem dumps. *)

CII 10070, IRIS 80, XDS Sigma 7 (Paris)

- 1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER.
Implementor: Didier Thibault
17 rue GAY-LUSSAC
F-75005 Paris FRANCE
527 6 85
Distributor: Pierre Maurice
UER d'informatique-Universite Paul Sabatier
118 route de Narbonne
F-31077 Toulouse FRANCE
(61) 53 11 20 x300
- 2. MACHINE. CII 10070; CII IRIS 80; XDS Sigma 7.
- 3. SYSTEM CONFIGURATION. SIRIS 7 & SIRIS 8 (CII operating systems; also easily available on other operating systems, see implementation description.)
- 4. DISTRIBUTION. Compiler source and assembler code are available on magnetic tape free. Just send a tape (mini if possible) to distributor.
- 5. DOCUMENTATION. Users Manual (in French); Sept. 1975. (* Not known if this is machine retrievable. *)
- 6. MAINTENANCE. Maintained from July 74 thru Jan 78.
- 7. STANDARD. Full standard with following extensions:
-separate compilation of Pascal program
-symbolic post mortem dump of variables & procedure in case of abort at execution time
-'value' feature for initialization of variables

- 'packed' variables implemented
- extensions to 'read' and 'write' for use in an interactive environment

8. MEASUREMENTS.
- compilation speed--1800 Pascal lines/min.;
 2400 characters/sec; versus
 1200 characters/sec. for FORTRAN.
- compilation space--to run the Pascal system;
 30 K words with overlay;
 45 K words without overlay.
- execution speed--dependant on program profile; compared to FORTRAN:
- | | FORTRAN | Pascal |
|-------------------------|---------|--------|
| matrix multiplication | 1 | 1.6 |
| recursive program | 1 | 0.3 |
| character count on file | 1 | 0.2 |
- execution space--(* No information provided. *)
9. RELIABILITY. Good to Excellent. This is release 3 of this compiler. The compiler has been tested since 1974 in 30 installations.
10. DEVELOPMENT METHOD. Full compiler generating object code for the linkage editor. The compiler consists of
- a MONITOR: programmed in CII's local assembly language (2K 32-bit words). It links the Pascal program to the operating system and controls the execution of the Pascal program. All operating system dependancies are located in this monitor. To get the compiler available on some other operationing system, the rewriting of this monitor is necessary.
 - a COMPILER: written in Pascal itself, it consists of 4800 lines. It is a one pass compiler with top-down syntax analysis, separate compilation of Pascal programs, symbolic post mortem dumps, and several specific options. The compiler is fully bootstrapped so that any user may adapt it easily to a specific need (change the table sizes, specific features, etc.).
 - a LIBRARY used by the linkage editor.
- (* Person-hours to create compiler not reported. *)
11. LIBRARY SUPPORT. Separate compilation of Pascal programs implemented. (* No information on subprogram libraries. *)

Computer Automation LSI-2

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Computer Automation; Naked Mini Division; 18651 Von Karman; Irvine, CA 92713; 714/ 833-8830; TWX:910 595 1767.
2. MACHINE. Computer Automation LSI-2 (16-bit minicomputer).
3. SYSTEM CONFIGURATION. Computer Automation OS. Minimum hardware: moving head or floppy disk and 32K Memory.
4. DISTRIBUTION. Distributed on floppy disk for \$900.
5. DOCUMENTATION. User's Guide explaining the use of Pascal under CA-OS. (* Apparently no machine retrievable documentation. *)
6. MAINTENANCE. Fully supported including acceptance and response to user trouble reports. In the near future, standard Pascal I/O will be implemented.
7. STANDARD. Implements Sequential Pascal which varies from standard Pascal. Missing: reserved words file, goto, label, packed; mixed type arithmetic; standard functions: ODD, EOLN, EOF, SQR, ROUND, SIN, COS, ARCTAN, LN, EXP, SQRT. Restricted to 2 levels of static nesting. Has extended I/O and file access methods.

8. MEASUREMENTS. (* No information provided. *)
9. RELIABILITY. Very good. (* Number of sites using system not reported. Date first released not reported. *)
10. DEVELOPMENT METHOD. Seven pass compiler. (* Method of developing compiler not reported. Number of person-hours to implement compiler not reported. *)
11. LIBRARY SUPPORT. Automatic formatting option implemented. (* No information provided on separate compilation or subprogram libraries. *)

CRAY-1 (Los Alamos)

UNIVERSITY OF CALIFORNIA
 LOS ALAMOS SCIENTIFIC LABORATORY
 (CONTRACT W-7405-ENG-36)
 P.O. BOX 1663
 LOS ALAMOS, NEW MEXICO 87545

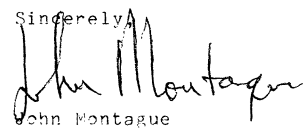
IN REPLY
 REFER TO: C-11
 MAIL STOP: 250

July 7, 1977

Dear Andy:

Despite Bob Johnson's rather discouraging letter, (PASCAL Newsletter #6), PASCAL still lives on the CRAY-1. We now have a new version based on Sassan Hazerhi's P-code Post Processor concept (P.N. #7). Current plans are to extend P-code and the P-code translator to provide code generation for the Model compiler.

I enclose an 11-point Newsletter-style description of our implementation, the User's Guide Addendum, and my check for \$4 for next year's P.U.G. membership.

Sincerely,

 John Montague

CRAY PASCAL (Version 2)

1. Implementors:

The compiler was bootstrapped by John Montague and Michael Powell. Many of the code templates were taken from Bob Johnson's cross compiler. Nearly all changes and improvements since the bootstrap was completed are due to Forest Baskett and Linda Zucconi. We can all be reached at the following address:

Los Alamos Scientific Laboratory
 Group C-11, Mail Stop 296
 P. O. Box 1663
 Los Alamos, New Mexico 87545
 (505) 667-7877

2. Machine:

Cray Research, Inc. CRAY-1

3. Operating System:

Benchmark Operating System (BOS), a LASL modified version of the CRI CRAY-OS Version 1.

4. Distribution:

Distribution is arranged on an ad hoc basis. All (both) current CRAY-1 installations have a copy.

5. Documentation:

Short write-up on the differences between CRAY PASCAL and Standard PASCAL, plus instructions for use.

6. Maintenance:

We will maintain CRAY PASCAL at LASL as long as we find it useful. The compiler is still undergoing development, and new versions will probably be complete replacements rather than updates. A project is underway to use the P-code translator as a code generator for Model, a LASL-developed language which will be used for much of our new CRAY-1 operating system.

7. Standards:

CRAY PASCAL implements the subset of PASCAL defined by the PASCAL-P compiler with a few extensions toward Standard PASCAL and several of the PASCAL-6000 predefined functions and procedures.

8. Compiler Implementation:

CRAY PASCAL is written in PASCAL and consists of two separate programs, the PASCAL-P compiler (version 2, extended by Sassan Hazeghi of Stanford University to the equivalent of P4, and further modified at LASL) and the P-CODE translator which converts P-CODE into CRAY Assembly Language (CAL.) Despite some character set problems, both programs currently run on the CRAY-1 and on a CDC 6600 under NOS. Some statistics on our implementation are:

	P-compiler	Translator
Lines of source code	4400	3900
P instructions generated	23,500	19,100
CAL instructions generated	38,100	36,500
Size of code (64-bit words)*	18,200	18,400
Compile, translate, assemble time (CPU sec.)	43	40

* this includes the run-time package

The compiler and translator run-time are currently dominated by the character by character I/O (about 50% of the total time). One of our current projects is directed toward improving the run-time support.

9. Reliability:

Most of the programs we have compiled were first debugged with PASCAL-6000, so error recovery hasn't really been tested. The P-compiler has had quite a bit of use at Stanford. No errors in the generated code have been detected for several months, and we have compiled and executed John Banning's 10,000 line PASCAL Analyzer program (PASCAL Newsletter #6).

10. Method of Development:

CRAY PASCAL was bootstrapped using PASCAL-P and PASCAL-6000. A total of 5 machines (CDC 6600, Cyber 73, 7600, Data General Eclipse, and the CRAY-1) and 3 character sets were involved in the bootstrap process. Approximately 6 man-months were required. Both implementors have previously modified batch OS/360-370 compilers to run interactively under ORVYL/370 (including

July 7, 1977

ALGOL-W, PL/C, and Sassan Hazeghi's PASCAL-P compiler) and are experienced system programmers. Neither implementor had ever used a CDC 6600 or a CRAY-1, or written large PASCAL programs before the project started.

Subsequent development has been done using a PDP-11/70 running UNIX, with a link to the CRAY-1 for compilation and testing.

11. Libraries, External Compilation, Etc.

No libraries are currently available or planned. External procedures (declared as FORTRAN, though actually requiring PASCAL calling conventions) are supported and have been used. Limited separate compilation is supported by allowing second level procedures (procedures declared in the PROGRAM block) to be entry points.

Data General Nova/Eclipse -- Introduction.

Since the announcement in PUGN 8 of a Data General implementation by R. E. Berry, we've witnessed a lot of activity this summer. As an experiment, we are going to try to get everyone together here!

Thanks to Rodney Thayer, Central Research Group, P.O.Box 451, Harvard, MA 01451 (617/772-2306) who wrote 77/07/07: "a few people in my area (myself included) are investigating R. E. Berry's U. of Lancaster PASCAL for the Data General NOVA. If I am closer than England for somebody, they are welcome to write to me to find out about Lancaster Pascal."

On 77/8/12, Gregg Marshall at the National Oceanic and Atmospheric Administration in Denver, CO 80200 (303/499-1000 x4482) wrote out a checklist for the Lancaster NOVA Pascal implementation, "in case they haven't sent one, too." (They hadn't.) Its information is included in the summary below.

Other NOVA implementations have appeared by Ted Park, A. J. Hurst, and Rafael Bonet - see below. H. S. Magnuski, Gamma Technology, 800 Welch Road, Palo Alto, CA 94304 (415/326-1661), wrote on 77/7/21 that he is trying to obtain several NOVA implementations for evaluation. Hopefully he will report his findings to PUGN. On 77/8/9, Bruce MacKenzie, Computervision Corp., 201 Burlington Road, Route 62, Bedford, MA 01730 (617/275-1800), announced that "we will be implementing Pascal on Data General's NOVA's and NOVA compatible machines running under our own operating system."

Also, Larry Walsh, ROLM Corp., 4900 Old Iron Sides Drive, Santa Clara, CA 95050 (408/988-2900) is looking at Pascal-P for the ROLM 1664, a ruggedized NOVA.

Requests for Data General implementation information have come from:
 77/07/11: Doug Kaye, Computer Services, Du Art Film Labs, 255 West 55 St., New York,
 NY 10019 (212/757-4580).
 77/07/14: Mike Tiller, 2501 N. Lancaster Lane #178, Plymouth, MN 55441 (612/546-6687).
 77/06/08: C. A. Miller, Dept. of Physics Nuclear Res. Center, University of Alberta,
 Edmonton, Alberta T6G 2N5.
 77/08/10: Kevin Driscoll, 330 SE 11th Ave., Minneapolis, MN 55414 (612/331-2133).
 77/08/16: Bruce K. Ray, Polymorphic Computer Systems, P.O.Box 3581, Boulder, CO 80303
 (303/443-5362).
 77/03/14: Wayne Seipel, James Peterson, Computer Science Dept., University of Texas,
 Austin, TX 78712 (512/472-1773).
 -Andy Mickel

Data General ECLIPSE (Loma Linda)

LOMA LINDA UNIVERSITY



LOMA LINDA CAMPUS
 LOMA LINDA, CALIFORNIA 92354

SCIENTIFIC COMPUTATION FACILITY

June 3, 1977

Dear Andy,

I thought this might be the first, but I see from the latest newsletter that at least one other Data General version exists.

However, I would like to report another Pascal P4 system solely designed for the Data General Eclipse Series computers with floating point hardware (since the Eclipse enhanced instruction set is heavily used, my Pascal will not run on a Nova).

I would be willing to disperse DG compatible dumps of the system to interested users who supply their own mag-tape. I am not in a position to supply documentation, so interested parties would still need to get the implementation kit from the University of Colorado.

To ease the implementation, I used a single size data unit -- 64-bits for everything. A virtual memory (paging) scheme is employed so that the system will run in almost any memory configuration.

The assembler for PCODE and interpreter are both written in DG assembly language. I am quite pleased with the speed of the system, it takes something over an hour to compile the compiler (~4000 lines of code). This is only 4 times slower than the vendor supplied FORTRAN compiles! (And the Pascal system is interpreted with software paging!!)

The specifications of the system are as follows:

worksize = 64 bits
 memory size = 64K words
 integer size = 32 bits used in all calculations (64 bits stored)
 real size = 64 bits

I have implemented the entire interpreter except the transcendental functions and the I/O of 'real' data. The transcendental functions are, at present, of little interest so I may be several months before implementing these. I/O of 'real' data is needed, I am working on it and will have it ready in a couple of weeks.

I have read several comments in the PUG newsletter indicating how many people perceived the bootstrapping process as being rather difficult -- indeed, the implementation kit seemed to indicate this also. I would like to offer my encouragement to those who try by pointing out that the implementation kit we

received from George Richmond at the University of Colorado was quite complete and bug free. I was able to have the compiler compile itself correctly after less than one man-month effort. All-in-all, I am very satisfied with the results.

Sincerely,

Ted C. Park
 Technical Specialist

TCP:map

cc: George Richmond

Old address:
 Ted C. Park
 Scientific Computation Facility
 Loma Linda University
 Loma Linda, CA 92354

New address:
 Ted C. Park
 Medical Data Consultants
 Suite 302
 1894 Commercenter West
 San Bernardino, CA 92408
 714/ 825-2683

The Scientific Computation Facility is a Biotechnology Research Resource supported in part by NIH grant RR 00276.

Data General NOVA (Canberra)



THE AUSTRALIAN NATIONAL UNIVERSITY

DEPARTMENT OF Computer Science
 BOX 4, POST OFFICE, CANBERRA, A.C.T. 2600

TELEPHONE: (062) 49 4625

22 June 1977

Dear Andy,

The department of Computer Science, Australian National University, is implementing PASCAL-P for a Data General NOVA from the Zurich P-4 portable compiler. The system is intended for cafeteria style student use and will require processor + 32K memory, disk, card reader and line printer as a minimum hardware configuration, and runs under RDOS. It is not intended at this stage to distribute the system, but interested people may write to A.J. Hurst, Department of Computer Science, ANU, Post Office Box 4, Canberra, A.C.T. 2600, Australia. The estimated completion date is late 1977.

John Hurst

telesincro s.a.
ordenadores electrónicos

rocafort, 98-100
teléf. (93) 385 41 00
telex 53095
barcelona-15

5 June 1977

Pascal User's Group

Dear Mr. Mickel:

I have received Pascal Newsletters #5 and #7 on the same enclosure from the University of Southampton, Great Britain. What about #6?

My company, SECOINSA-TELESINCRO is a holding owned by the spanish government for the development of the national computer industry.

We bought CALTECH'S SOLO SYSTEM to experiment it as a software tool running in the DGC'S NOVA 840 at the research & development department. The NOVA is used as a software factory.

Enclosed is a short report about our implementation. Sorry but distribution is not planned.

I have personnel interest in PASCAL, so the address in the PUG mailing list is my home address. My office address is the implementors' address below.

Sincerely yours:



Rafael M. Bonet

RMB/tg
cc: P. Brinch Hansen

1.- Implementors:

Rafael M. Bonet
Arsenio Lago
Ramón Cervelló
TELESINCRO S.A.
Departamento de Investigación y Desarrollo
Rocafort 100
Barcelona 15
SPAIN
Phone: (93) 3254100

2.- Machine:

Data General Corp. NOVA 840

3.- Operating System:

SOLO SYSTEM

Minimal Hardware Configuration:

CPU options: Floating Point Unit
Automatic Multiply/Divide Unit
Real Time Clock
Memory Map&Protection Unit (MMPU)
Memory : 44Kw, minimum
Disk : DIABLO model 33 (2.5MBytes)
AMPEX model DM448 with western Peripherals Interface
Tape : WANGCO 800 bpi, 9tracks, 45 ips
Printer : TALLY 200 lpm with Data Products Interface
Card Reader: DOCUMENTATION 600 cpm with Documation Interface
Console : Standard

Also supported by the system:

Second Console
4060 Multiplexer
as much memory as supported by the MMPU

4.- Method of distribution:

The SOLO SYSTEM and its distribution is not a company objective. Thus, we have no plans for distribution. But we shall study each request of a system copy.

5.- Documentation available:

Our system is an implementation of the CALTECH'S SOLO SYSTEM. The languages description is given in two CALTECH Manuals:

- Concurrent PASCAL report.
- Sequential PASCAL report.

The system works in interpretive mode. The NOVA interpreter, an assembly program, is documented in spanish.

6.- Maintenance:

The high level coding (CPASCAL or SPASCAL) was written at the CALTECH by Per Brinch Hansen's team. Neither CALTECH nor Per Brinch Hansen (now at the Southern California University) provide maintenance for this software. The low level coding (the NOVA interpreter) is responsibility of our team, but our structure does not allow a formal maintenance. Of course, we accept error reports.

7.- Standards:

CALTECH sequential PASCAL is not a standard PASCAL implementation as you can found in the CALTECH report.

8.- Compiler / Interpreter:

The system is interpretive. The only portion in target machine code is an assembly program, called the kernel, with a size of 5K words of 16 bits. The PASCAL interpreter, included in the kernel is 2 K words long. The SOLO D.S. runs interpretively and is coded in Concurrent PASCAL. The SOLO runs sequen PASCAL programs. The compilers speed is about 90 char/sec.

9.- Reliability:

The kernel reliability is excelent. For the compilers, some not important bugs were detected. So me of them were fixed. In general the reliability is good.

10.- Development method:

The tapes from CALTECH were used to implement a bootstrap SO-LO SYSTEM running under DGC'S Real Time Disk Operating System. Then we developed our stand-alone SOLO SYSTEM. The system can produce a backup tape. This tape is loaded into disk by means of the IPL operation and an AUToload program written at the beginning of the tape. Once the tape on disk the system is loaded by IPL. For the people interested only in sequential PASCAL: it is possible to write an interpreter (or compiler) of sequential PASCAL, changing the SYSTEM CALL instruction from a branch to concurrent code to the actual execution of the function required. As Per Brinch Hansen says, it is a 1 man month work, but it doesn't exist a documentation about how to do it.

Data General NOVA (Lancaster)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. R. E. Berry and A. Foster; Dept. of Computer Studies; University of Lancaster; Bailrigg, Lancaster LA1 4YX, U.K.; 65201 (STD 0524).
2. MACHINE. Data General Nova series (2/10, 820).
3. SYSTEM CONFIGURATION. RDOS 4.02/5.00 operating system; 32K core, disk backing store.
4. DISTRIBUTION. Cassette tape or 2.5 Mbyte cartridge disk.
5. DOCUMENTATION. A user manual is provided.
6. MAINTENANCE. No formal commitment to provide support; Release 2 under development and will subsequently be consolidating bug reports accepted on Release 1.
7. STANDARD. Pascal P4 subset accepted.
8. MEASUREMENTS. Typical runtimes compare favorably with those of other languages generally available on the Nova. P-code is generated, assembled and then interpreted. (* Compilation and execution space requirements not reported. *)
9. RELIABILITY. (* Thought to be good. Number of sites using system not reported. Date first released not reported. *)
10. DEVELOPMENT METHOD. Originally cross-compiled from a CDC 7600. The P-code assembler was written from scratch in Pascal; the P-code interpreter was implemented in Nova assembly language.
11. LIBRARY SUPPORT. (* No information provided. *)

DEC PDP-8 Minnesota

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. John T. Easton, 612/373-7525; James F. Miner, 612/373-9916; Jonathon R. Gross, 612/835-4884; Address correspondence to: Pascal Group; SSRFC; 25 Blegen Hall; University of Minnesota; 269 19th Ave. South; Minneapolis, MN 55455; 612/ 373-5599.
2. MACHINE. Digital Equipment Corp. PDP-8/e
3. SYSTEM CONFIGURATION.
OS/8 version 3. Hardware required:
-KE8-E (EAE with mode B instruction set)
-RK8-E disk, or other direct access mass storage device with at least 131K 12-bit words (e.g., DF32 or RF08).
-16 K minimum of core/RAM. 32 K is highly recommended.

4. DISTRIBUTION. Not yet ready for release.

5. DOCUMENTATION. Machine retrievable supplement to Pascal User Manual and Report (about 25 pages), in preparation.
6. MAINTENANCE. A policy has not yet been determined.
7. STANDARD. Emphasis has been placed on close adherence to the Pascal User Manual and Report. There are two major restrictions: a) Parameters may not be procedures and functions. This restriction will not be lifted without full type checking (which requires a change in the Pascal Standard). b) Files may be declared only in the main program, and files may not be components of arrays, records, or files; nor may files be allocated with the procedure NEW. Minor restrictions: set size-96 elements; maxint-8,388,607 (2**23-1). Full ASCII character set is supported.
8. MEASUREMENTS.
Execution speed--roughly comparable to FORTRAN IV (F4). I/O seems to be faster than FORTRAN, while computation seems slower.
Execution space--Interpreter takes 12K, space needed for P-code and run-time storage depends on program.
Compilation speed--much slower than F4. We hope to make some improvements in this area. About 30 characters/sec. presently (77/07/30).
Compilation space--65K 12-bit words to compile itself.
9. RELIABILITY. Fair to good and improving. The system is has been in use at 1 site since 76/11.
10. DEVELOPMENT METHOD. As with most languages on the PDP-8, Pascal makes use of an interpreter (a modification of P-code) written in PAL8. The compiler (about 5400 lines, based on Pascal-P4) and assembler are written in Pascal. All standard procedures are written in PAL8. Because of the design of the system, the implementation is not suitable for real-time applications. On the other hand, the implementation does provide 131K words of virtual memory for code and store. Effort involved has been 1 person-year for applications programmers without previous experience writing compilers.
11. LIBRARY SUPPORT. Currently (77/07/30), none.

Digital Equipment Corporation (DEC) PDP-11 -- Introduction

At one time last year (PUGN 6-7) Steven Schwarm and C. E. Bridge at DuPont wrote to say they were coordinating a DECUS SIG Pascal. We thought they would coordinate PDP-11 implementations. Well, they haven't, and they have not been communicating either. We've heard that DECUS SIG Pascal is in other hands.

Interest in PDP-11 Pascal has been high. But from our point of view there are far too many Pascals on the 11 to wade through.

A few comments: Electro Scientific Industries Pascal for the 11 has received another good report - see the letter from Wayne Rasband. Structured Systems has come up with an implementation which runs on many operating systems including UNIX. The highest quality RSX-11 system we've had reports on comes from Stockholm. Finally, we have news of UNIX Pascal from U.C. Berkeley.

Jim Shores, with the US Navy in Connecticut, phoned on 77/05/24 and reported he had a Brinch Hansen interpreter running as a task under RSX. Also he phoned Bob Lucas at NBS in Maryland and found out that Bob doesn't think too much of his own RSX implementation. With all the others around now, that's okay.

See also HERE AND THERE News section under David Miller, Matli Karinen, John Nunnally, Alfred J. Hulbert, Martin Tuori, and Aron Insinga.

-Andy Mickel



DEPARTMENT OF HEALTH, EDUCATION, AND WELFARE
PUBLIC HEALTH SERVICE
ALCOHOL, DRUG ABUSE, AND MENTAL HEALTH ADMINISTRATION

July 14, 1977

NATIONAL INSTITUTE OF MENTAL HEALTH
9000 ROCKVILLE PIKE
BETHESDA, MARYLAND 20014
AREA CODE 301 TEL: 656-4000

Dear Andy:

I suspect that readers of the PASCAL Newsletter may get the impression that there does not exist a reliable standard PASCAL compiler for the PDP-11 that is useful for production work, but from our experience this is simply not the case. We are using the compiler from Electro Scientific Industries (ESI) under the RT-11 operating system on five different PDP-11 systems (11/03, 11/04, 11/20, 11/34, 11/40) for real-time laboratory applications and image processing. We have found ESI PASCAL better suited for process control type applications than the DEC FORTRAN. It generates in-line as opposed to threaded code. It allows direct access to I/O device registers as opposed to requiring subroutine calls. It provides a more efficient interrupt handling capability and allows insertion of assembler language statements in-line.

ESI PASCAL has also proven more practical for use on small PDP-11 configurations, such as a 16K 11V03 with dual floppy disks, because it requires less memory and disk space. The ESI compiler (written in MACRO) is half the size of DEC's FORTRAN compiler and the PASCAL run-time support library is one-third the size of the FORTRAN library.

Sincerely,

Wayne Rasband
Section on Technical Development
National Institute of Mental Health
Bldg. 36, Rm. 2A-03
Bethesda, Md. 20014
301-496-4957

DEC PDP-11 (ESI)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. John Ankorn; David Rowland; Electro-Scientific Industries; 13900 NW Science Park Dr.; Portland, OR 97229; 503/ 641-4141; TELEX: 360273.
2. MACHINE. Any model Digital Equipment Corp. PDP-11.
3. SYSTEM CONFIGURATION. Minimum of 16K words. Operates under RT-11. Currently (76/11/02), an RSX-11M implementation is underway.
4. DISTRIBUTION. Compiler, support module, cross referencer, text editor and instruction manual available for \$1500. (* Tape formats, etc. not reported. *)
5. DOCUMENTATION. Over 70 page machine retrievable instruction manual. Currently (76/11/02) working on more.
6. MAINTENANCE. One year of unlimited fixes and updates, followed by annual subscription service. (* Reported by users that "vendor seems to be responsive in terms of support". *)

7. STANDARD. Full standard plus extensions: additional features for real-time hardware control; separate compilation of procedures; Macro (assembler) code in line insertion; actual core addresses of variables can be fixed (giving access to external page I/O addresses at the Pascal level.

8. MEASUREMENTS.

compilation speed--About 3500 characters /second, on the PDP-11 model 05.
compilation space--very economical-it can compile 3000 line programs in 28K on PDP-11/40. No overlays are used in the system.
execution speed--about twice as fast as the DEC FORTRAN IV and many times faster than DEC BASIC. A worst-case 'number-cruncher' example ran at 40% faster than the DEC original FORTRAN.
execution space--very economical-much of the space improvement over DEC FORTRAN is due to the smaller support module for Pascal.

9. RELIABILITY. Excellent--far better than DEC FORTRAN. In use since 75/11. (* Number of sites using compiler not reported. *)

10. DEVELOPMENT METHOD. Single pass recursive descent compiler written in Macro-11. Hand-coded based on University of Illinois bootstrap (with extensive changes) in about two person-years of effort. First compiler written by both implementors. Compiler translates source into Macro-11 which is then assembled and linked to the support module for execution.

11. LIBRARY SUPPORT. Separate compilation of procedures with load-time insertion and linkage is implemented.

DEC PDP-11 (Los Altos)



343 Second Street, Suite K
Los Altos, California 94022
415 321 8111

STRUCTURED SYSTEMS CORPORATION is pleased to announce a new Pascal compiler for the DEC PDP-11. The STRUCTURED SYSTEMS Pascal compiler (PASCAL-SS) was designed and implemented by the team of A. Ian Stocks and Jayant Krishnaswamy, who previously developed the University of Illinois Pascal-11 compiler.

The PASCAL-SS compiler is itself written in PASCAL and is self-compileable. It translates Pascal source programs directly into machine code. The language implemented is closely based on Jensen and Wirth's revised report (1975) with a number of language extensions and additional features aimed at large-scale system development in a production environment. Versions of PASCAL-SS are implemented or under development to run under the most popular PDP-11 operating systems, including DOS, RT-11, RSX-11 and UNIX.

Many features have been incorporated into STRUCTURED SYSTEMS PASCAL-SS which make it one of the most powerful and convenient-to-use Pascal systems for a production environment. Extensive compile- and run-time error checking and reporting features are incorporated in the compiler. Compiler '\$' options include an identifier cross-reference, automatic formatting/indentation of source listings, conditional compilation of sections of the source, a macro-expansion pass (similar to DEFINE in Burroughs Algol), and textual inclusion of library files in the source stream. Extensive object code optimization may be specified.

Programs and routines may be defined in separately compiled modules and linked together. User-controlled overlays permit very large programs to be compiled and executed under severe core constraints.

Anyone wishing additional information on PASCAL-SS should contact:

Martin Rattner
STRUCTURED SYSTEMS CORPORATION
343 Second Street, Suite K
Los Altos, California 94022
(415) 321-8111

Dear Andy,

The newsletters are really interesting to read, although distribution is somewhat slow.

When the May issue finally appeared in the end of July I found that I had mixed up my own address in the implementation note describing our PDP 11 compiler. I enclose an updated version.

As you can also see, I have decided to distribute the compiler myself. Mr Schwarm of du Pont have promised to distribute through DECUS, but I haven't heard anything from him yet.

Truly yours,



Seved Torstendahl

1 IMPLEMENTOR

Seved Torstendahl
Address: Telefon AB LM Ericsson
AL/X/Tdq S-125 25 Stockholm, Sweden

Phone number:
Sweden, 08 / 719 0000

2 MACHINE

DEC-10: crosscompiler that generates code for all PDP-11's.
PDP-11: model 35 and up.

The compiler generates code for floating point hardware and extended arithmetic if option switches are set.

3 OPERATING SYSTEM

RSX-11M or IAS. (DEC-10 crosscompiler under IUPS-10). Probably it is an easy task to replace the RSX interfacing routines with new ones interfacing DOS or RT-11. We do not plan to do that work here. Maybe routines to interface with RSX-11S will be made.

4 DISTRIBUTION

The compilers are available at no cost if tapes are supplied. The distribution set contains source and object modules of the compilers and the runtime library, command files for compiler generation and maintenance, user manual and compiler generation instructions.

The compiler will be distributed at no cost if tapes are supplied for one or more of the following choices:

- three DECtapes in PDP 11 DOS format (DEC10 and PDP11 users)
- one 9-track magnetic tape in DEC 10 format (DEC10 users)
- one 9-track magnetic tape in industry compatible format (users of DEC10 and other computers)
- one 9-track magnetic tape in DOS format (PDP11 users).

5 DOCUMENTATION

A user manual complementing the UNIX book.

6 MAINTENANCE

No responsibility, but if errors are found reports will be distributed to known users.

7 RESTRICTIONS AND EXTENSIONS

The compiler is a modification of the crosscompiler from Mr Bron of Twente University of Technology, The Netherlands. Two major modifications have been undertaken:

- the compiler generates standard object modules
 - the compiler gives full access to RSX file system
- The following list is mainly a copy from Mr Bron's contribution in Pascal Newsletter #7.

With regard to the definition of Pascal in Pascal User Manual and report the following restrictions hold:

- packed data structures are only implemented for character arrays (always packed, two char's/word) and for boolean arrays (packing optional, one boolean/bit). The procedures pack and unpack are not implemented.
- only local jumps are allowed.
- a pair of procedures, mark and release, to allocate and deallocate dynamic storage.

The following extensions have been implemented:

- function results can be of nonscalar type,
- arrays with unspecified bounds (but specified index-structure) can be used as formal parameters to procedures, allowing differently declared variables or constants as actual parameters,
- a string parameter type has been introduced in which one-dimensional character arrays or substrings thereof may be passed as parameters. Such strings and their constituent characters are considered as "read only",
- procedures may be compiled separately,
- separately compiled procedures can be accessed through a declaration with the procedure block replaced by "extern".

8 SOURCE LANGUAGE

The compilers are written in Pascal, and both have the same source code except for two separately compiled routines. The crosscompiler is generated when the DEC-10 Pascal compiler from Hamoury compiles the source, when it then compiles itself the PDP-11 version is created.

The size of the compiler is 50k words of code. In a PDP-11 running under RSX-11M V2 only 32 kwords are available for code and data. Through a slight modification of the overlay loading routine of RSX-11M it has been possible to segment the very recursive compiler. It now fits in a 32 kwords partition and uses about 22 kwords for code leaving 10 kwords for data. This is enough to compile fairly large programs. However, the overlay mechanism makes the compiler slow, about 200 lines / minute with PDP4-compatible disks, less with RK05 disks. with RSX-11M V3 using PLAS and a 64k partition the speed is increased 7-10 times.

9 RELIABILITY

Excellent. The compiler is now in heavy use at three sites, and is used at four more. No errors have been found during the last two months.

10 METHOD OF DEVELOPMENT

The crosscompiler for PDP-11 running on DEC-10 produced by Bron et al was used as input. As mentioned earlier, this compiler was modified to generate object code linkable under RSX-11M and to give access to the file system of RSX-11M. when the crosscompiler was finished it compiled itself and the compiler was thus transferred to PDP-11.

The implementation effort until now is about 6 manmonths. Maybe a new version which performs some optimization will be developed later.

DEC PDP-11 (Twente)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. C. Bron; Twente University of Technology; P.O. Box 217; Enschede, Netherlands; 05420-99111; TELEX: 44200.
2. MACHINE. DEC-10 cross-compiler producing code for any PDP-11.
3. SYSTEM CONFIGURATION. No operating system requirements. (* Minimum hardware requirements not reported. *)
4. DISTRIBUTION. Available on DECTape or 9 track magtape free of charge.
5. DOCUMENTATION. Machine retrievable documentation package.
6. MAINTENANCE. We intend to correct reported errors for the next few years. Error reports and updates will be sent at irregular intervals to all those who have received the compiler, unless otherwise requested.

7. STANDARD. Restrictions: Files not implemented (except input and output); jumps out of procedures not allowed; packed only implemented for one-dimensional character arrays (always packed) and one-dimensional boolean arrays (packing optional); procedure "dispose" not implemented (procedures 'mark' and 'release' will suffice for nested allocation and deallocation). Extensions: function results can be of non-scalar type; arrays with unspecified bounds can be passed as parameters to procedures; several added standard procedures, including a pair to obtain and set the value of device-register memory locations; procedures may be declared in the outer block to be associated with specified interrupt sources; a string parameter type has been introduced in which one-dimensional character arrays or substrings thereof may be passed as actual parameters (such strings and their constituent characters are considered as "read-only").

8. MEASUREMENTS. (* No information provided. *) Reported to be "quite fit for real time applications".

9. RELIABILITY. Good. (* Number of sites using system not reported. *) First distributed in 75/12.

10. DEVELOPMENT METHOD. Cross-compiler running on DEC-10 producing code for any PDP-11. Developed from Pascal-P. (* Person-hours to develop system not reported. *)

11. LIBRARY SUPPORT. (* No information provided. *)

DEC PDP-11 (Vienna)

Osterreichische Studiengesellschaft für Atomenergie Ges.m.b.H.

Lenaugasse 10 • A-1082 WIEN • Austria



Institut für Physik

Forschungszentrum Selbersdorf
 Telefon: (02254) 201.781*
 Telex: 014/353
 Telegramm: austratom wien
Bankverbindungen
 CA - Bankverein: 26-34343/02
 E. & Spar-Casse: 100-94709
 Österr. Länderbank: 106-100-432

Pascal User's Group
 c/o Andy Michel
 University Computer Center
 227 Exp Engr
 University of Minnesota
 Minneapolis, MN 55455

U S A

Ihr Zeichen	Ihre Nachricht vom	Unser Zeichen	Sachbearbeiter	Telefon (Durchwahl)	Datum
		PH/May/Hä		*	1977 06 01

We have just recently joined the PASCAL Users Group and want to tell you about the work concerning PASCAL and its applications in our data-processing group, especially

- 1.) that we have implemented P.B. Hansen's Sequential Pascal compiler in the PDP-11 Operating System RSX11-M and RT11.
 - 2.) why we choose just this compiler
 - 3.) something about the new design of the compiler
 - 4.) what we are just working on and what we plan to do.
- ad 2.) We needed a high level programming tool for process control, scanning and analyzing data and so on. Our principal concern was system security and flexibility.
 (One of the main applications being in safety control).

The advantages of P.B. Hansen's Pascal compiler are:

It is the only compiler we know, running in the original version without any bugs. Anyone concerned with compilers that are not supported by any maintainer will appreciate this.

- The concept is clear and easy to understand. This enables you to modify or extend the compiler for your special purposes.
- The concept of a "virtual machine" makes you almost independent from machine and Operating System.
- The complete interface to the Operating System is contained in a simple program prefix. This seems to be the greatest advantage.

The few restrictions to "Standard Pascal" did not matter that much to us.

ad 3.) One of the most difficult tasks was to design a suitable set of prefix routines (as an interface to RSX instead of the Solo Operating System of P.B. Hansen). These prefix routines are system functions that manage for example reading, writing and overlay loading.

Our main principle of design was to be as simple and clear as possible so that

- the programmer can learn to use the new interface as quick as possible.
- even in extreme cases it is obvious what happens
- I/O is efficient (time and core/discspace).

To put this concept through requires a lot of courage, for the users often want an I/O system as complex as they are used to from other programming languages (FORTRAN!). Moreover, RSX has a very sophisticated filesystem and it is hard to implement it in the PASCAL-system and not to use all the complex functions it contains.

As an example, look at the way files are handled by the new prefix. Only two types of files are supported: sequential textfiles and random access files with a fixed recordlength of 512 bytes.

There are three groups of prefix routines for the file handling:

1.) routines for file definition:

```
PROCEDURE PAGEFILE(U: UNIT; F:FILENAME);
PROCEDURE TEXTFILE(U: UNIT; F:FILENAME);
```

with the type definition

```
CONST FILENAMELENGTH = 30;
TYPE FILENAME = ARRAY [1..FILENAMELENGTH] OF CHAR;

CONST MAXUNIT = 4;
TYPE UNIT = 1..MAXUNIT;
```

These routines associate a page- or textfile with an unit number.

2.) file management routines:

```
PROCEDURE CREATE(U: UNIT; INITIALSIZE: INTEGER; C: CONTIGOUSTYPE);
PROCEDURE CREATETEMPORARY
  (U: UNIT; INITIALSIZE: INTEGER; C: CONTIGOUSTYPE);
PROCEDURE OPEN(U:UNIT; ACCESS: FILEACCESS);

PROCEDURE CLOSE(U: UNIT);
PROCEDURE DELETE(U: UNIT);
```

"Create" and "create temporary" create a new file and "open" opens an existing file.

"contigoustype" and "fileaccess" define the method of allocation and access.

```
TYPE FILEACCESS = (READONLY,MODIFY,EXTEND,APPEND,READSHARED);
TYPE CONTIGOUSTYPE = (NONCONTIGOUS,CONTIGOUS);
```

3.) routines for reading and writing

```
CONST PAGELENGTH = 512;
TYPE PAGE = ARRAY [1..PAGELENGTH] OF CHAR;

CONST LINELENGTH = 132;
TYPE LINE = ARRAY [1..LINELENGTH] OF CHAR;

PROCEDURE READPAGE(U: UNIT; N: INTEGER; VAR BLOCK: PAGE);
PROCEDURE WRITEPAGE(U: UNIT; N: INTEGER; VAR BLOCK: PAGE);
PROCEDURE READCHAR(U: UNIT; VAR C: CHAR);
PROCEDURE WRITECHAR(U: UNIT; C: CHAR);
PROCEDURE READLINE(U: UNIT; VAR TEXT: UNIV LINE);
PROCEDURE WRITELINE(U: UNIT; TEXT: UNIV LINE);
```

"readpage" and "writepage" are for random access pagefiles. The other routines are for sequential textfiles. Instead of the type "page" any other type with the same length can be used.

We don't claim to have invented new functions. On the contrary we have omitted as much as possible from the RSX-file system options without restricting its feasibility for the PASCAL user.

But how to work with those simple I/O routines?

A good practice will be the following one:

The programmer chooses a set of "I/O-operators" for his special needs. These operators are procedures and functions written in Pascal. The programmer takes them out of a programlibrary or writes them himself or modifies existing programs for his purposes.

An example for such a set will be:

```
procedure readinteger (var n: integer, length: integer);
procedure writeinteger (n: integer, length: integer);
procedure skipdelimiter; procedure newline;
```

and so on

The procedure readinteger does what you expect:

It reads an integer "n" with at most "length" characters from the inputstream ending with the next delimiter.

The only systemroutine used is "read one character from an inputstream". In Pascal procedures like readinteger can easily be written. If the programmer is in doubt what the program really does, one glance at the pascal source program (instead of considering twenty rules in a manual) obviously will explain it.

This method is the best one to meet the need for structured, modular, portable and flexible programs.

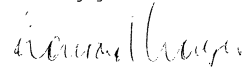
ad 4.) A Pascal version for easily programming CAMAC Systems is under work and will be running summer 1977.

The implementation of Concurrent Pascal in the Operating System RSX11M using the task synchronisation facilities of RSX11M will be completed at the end of the year.

Afterwards we are planning to use (Concurrent) Pascal and the conditional critical regions-concept for multiprocessing applications (with microprocessors Intel 8080).

If you are interested in our work, please write to us.

Sincerely yours,



Dipl.-Ing. K. Mayer

DEC PDP-11 (Belgium)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Pierre Verbaeten; K. V. Leuren; Applied Mathematics and Programming Division; Celestijnenlaan 200B, B-3030; Heverlee, Belgium; (* No phone number provided. *)
2. MACHINE. Digital Equipment Corp. PDP-11.
3. SYSTEM CONFIGURATION. UNIX. (* Minimum hardware requirements not reported. *)
4. DISTRIBUTION. (* No information provided. *)
5. DOCUMENTATION. (* No information provided. *)
6. MAINTENANCE. (* No information provided. *)
7. STANDARD. (* No information provided. *)
8. MEASUREMENTS. (* No information provided. *)
9. RELIABILITY. (* No information provided. *)
10. DEVELOPMENT METHOD. (* No information provided. *)
11. LIBRARY SUPPORT. (* No information provided. *)

DEC PDP-11 (Berkeley)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Charles Haley, William Joy, and Ken Thompson, Computer Center, Evans Hall, University of California - Berkeley, Berkeley, CA 94720. (* No phone number provided. *)
2. MACHINE. Digital Equipment Corp. PDP-11.
3. SYSTEM CONFIGURATION. UNIX. (* Minimum hardware configuration not reported. *)
4. DISTRIBUTION. (* No information provided. *)
5. DOCUMENTATION. We at PUGN have received "MANual" documentation (machine readable). Also available are: UNIX Pascal "Report Appendix", UNIX Pascal "User Manual", and "PXP User Manual". (* These are apparently machine readable. *)
6. MAINTENANCE. (* No information provided. *)
7. STANDARD. Restrictions: procedures and functions may not be passed as parameters; only the first parameter to NEW is treated - subsequent parameters are ignored. A compiler option directs the compiler to accept only Standard Pascal constructs.
8. MEASUREMENTS. (* We have been told that the system is quite fast, even though it is interpreted. No other measurements have been reported. *)
9. RELIABILITY. (* No information provided. *)
10. DEVELOPMENT METHOD. Parsing is done by a modified LALR parser. Object code is interpreted via threaded code.
11. LIBRARY SUPPORT. (* No information provided. *)

DEC PDP-11 (Portland)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Barry Smith, Oregon Museum of Science and Technology, Computing Department, 4015 SW Canyon Road, Portland, OR 97221 (503/248-5923).
2. MACHINE. Digital Equipment Corp. PDP-11.
3. SYSTEM CONFIGURATION. RSTS/E. (* Minimum hardware requirements not reported. *)
4. DISTRIBUTION. (* No information provided. *)
5. DOCUMENTATION. (* No information provided. *)
6. MAINTENANCE. (* No information provided. *)
7. STANDARD. (* No information provided. *)
8. MEASUREMENTS. (* No information provided. *)
9. RELIABILITY. (* No information provided. *)
10. DEVELOPMENT METHOD. (* No information provided. *)
11. LIBRARY SUPPORT. (* No information provided. *)

DEC PDP-11 (PAR)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Michael N Condict; PAR Corporation; On The Mall; Rome, NY 13440; 315/ 336-8400.
2. MACHINE. Digital Equipment Corp. PDP-11/45.
3. SYSTEM CONFIGURATION. RSX-11d. Minimum hardware same as for RSX.
4. DISTRIBUTION. None until at least 7/7/06.
5. DOCUMENTATION. None yet. (* Not known if documentation will be machine retrievable. *)
6. MAINTENANCE. None yet.
7. STANDARD. Full Standard, probably with extensions.
8. MEASUREMENTS. Expected to be about 5000 FORTRAN source lines and 3000 Pascal source lines. Expected to run rings around FORTRAN compiler. (* Rich Cichelli reports on 7/7/08/31 that it is a 2 pass system in which the code generated is faster than the 19 (!) pass optimizer for William Wulf PDP-11 Fortran! *)
9. RELIABILITY. Will not be distributed until it is.
10. DEVELOPMENT METHOD. One pass Pascal to FORTRAN translator. Initial version of each procedure written in Pascal and then hand translated into FORTRAN. When compiler is finished or can compile itself it will be restored to its original Pascal in a massive inverse translation, and then run through itself, thus completing the bootstrap. Currently (7/6/12/14) project has consumed about 4 person-months. Expected to consume 6 to 9 person-months in all (with 1 person devoting half-time). Implementor previously built a compiler for a subset of Pascal for a class project, but has never written any program this large before.
11. LIBRARY SUPPORT. (* No information provided. *)

DEC LSI-11 (San Diego)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Pascal Group; Institute for Information Systems; UCSD Mailcode C-021; La Jolla, CA 92093; (* No phone number reported. *).
2. MACHINE. Digital Equipment Corp. LSI-11 Microprocessor, PDP-11 any model, TERAK 8510 and 8510A.
3. SYSTEM CONFIGURATION. Comes with a one-user operating system. Apparently requires some mass storage (disk or floppy disk).
4. DISTRIBUTION. Distributed on floppy disk in two versions: 1) Complete release: including all source code and internal documentation (\$200); and 2) Code release: including system code and users manual (\$50).
5. DOCUMENTATION. For complete release: compiled listings of all source code, and user and system maintenance documentation as complete as it exists. For code release: Users manual but no detailed system documentation. Documentation is machine retrievable.
6. MAINTENANCE. For complete release: compiler updates at least 3 times during 77/8/1 thru 78/8/1. For code release: No continued support for later releases. Only minimal assistance in response to telephone inquiries. Future plans: plan to have a version of this system for the Zilog Z-80 ready. Plan to have versions for Intel 8080a ready by 77/9, MOS Technology 6502, and Motorola 6800 ready by summer of 1978.
7. STANDARD. Pascal-P subset plus strings.
8. MEASUREMENTS. 700 lines per minute compile speed. 20K byte compiler, 10K bytes for resident monitor, interpreter, and run-time support.
9. RELIABILITY. Reported good. First released on 77/8/1.
10. DEVELOPMENT METHOD. Pascal-P2 via B6700, PDP-11/10 bootstraps.
11. LIBRARY SUPPORT. Extensive graphics software, text editor, text formatter, pretty printer, all in Pascal.

DEC PDP-11 (Urbana)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER.
Implementors:
A.I. Stocks J. Krishswamy
Dept. of Computer Science Dept. of Computer Science
University of SW Louisiana University of Illinois--Urbana
P. O. Box 4330 Urbana, IL 61801
Lafayette, LA 70509 217/ 333-4428
318/ 233-3850 x538
Distributor:
Pascal-11; c/o M.D. Mickunas; 222 Digital Computer Lab; University of Illinois - Urbana;
Urbana, IL 61801; 217/ 333-6351.
2. MACHINE. Digital Equipment Corp. PDP-11/20 or up.
3. SYSTEM CONFIGURATION. Operates under our own operating system, which grew out of DEC's DOS/V4. In case you desire to install Pascal-11 on your own version of DOS, we also provide a list of DOS/V4 modifications. We believe that these modifications are sufficient for adapting DOS/V4 to Pascal-11, but we can, of course, make no guarantees. We caution that these modifications are not sufficient for installing Pascal-11 on other operating systems, but your DOS expert should be able to make the necessary modifications using our DOS/V4 modifications as a guide.

Hardware requirements are:

- PDP-11/20 or up.
28K words of addressable core store.
either a DEC RF-11 or a DEC RK-11.
(In case you have some other disk, your DOS expert should have little trouble replacing our disk driver with your own.)
a DECTape unit (we can supply the system only on DECTapes).
4. DISTRIBUTION. While our Pascal-11 system is not yet complete enough for widespread distribution, we are happy to make it available on a limited basis to interested persons. Our distribution package includes:
 - 1) Pascal-11 source of the Pascal-11 compiler.
 - 2) MACRO-11 source of the Pascal-11 run-time routines.
 - 3) Binary for both the compiler and the run-time routines.
 - 4) Binary for our operating system.If you are interested in obtaining this software, please send the following to the distributor:
 - 1) Three DECTapes (these must be in PDP-11 format).
 - 2) A statement of your intended uses.*
 - 3) One signed copy of Prof. Snyders enclosed letter.*
 - 4) A stamped, self addressed mailer for returning your DECTapes (total weight is about 900g (2 pounds)).*The Pascal-11 compiler was developed at the University of Illinois - Urbana and is copyrighted by its Board of Trustees. The work was supported in part by a grant from the National Science Foundation. Accordingly, distribution is made to any interested persons or parties who intend to use this software for "research, education, or other legitimate purposes." The NSF requires that we inform them of those receiving this software and their intended uses of it.
 5. DOCUMENTATION. Unfortunately, very sparse at present (77/01/21) but we shall include in the distribution package all that is available. (* This is apparently not machine retrievable. *)
 6. MAINTENANCE. Since the project under which the compiler was developed has expired, we have no source of funds for maintaining and upgrading the compiler. Consequently, we offer Pascal-11 'as-is', with no plans to extend it or to implement it on another system.
 7. STANDARD.
Differences:
"with" unimplemented.
types real and set unimplemented.
variant records not permitted.
procedures-as-parameters not permitted.
writeln, readln not implemented
EOL feature still included.
Extensions:
compile time options.
source level library routines.
overlays.
 8. MEASUREMENTS.
compilation speed--(* No information reported. *)
compilation space--(* No information reported. *)
execution speed--(* No information reported. *)
execution space--(* No information reported. *)
 9. RELIABILITY. (* Information on reliability not reported. Number of sites using system not reported. Date first released not reported. *)
 10. DEVELOPMENT METHOD. (* No information provided. *)
 11. LIBRARY SUPPORT. Source level library routines are implemented.

UNIVERSITÄT HAMBURG

Tele-Nr.: 214732 uni hh d

Institut für Informatik
2 Hamburg 13, Schlüterstraße 66-72

INSTITUT FÜR
INFORMATIK

Prof. Dr. H.-H. Nagel

Fernsprecher: 040-4123-4151 } Durchwahl
Belegedaten: 9.09(,)

Datum und Zeichen Ihres Schreibens

Aktenzeichen (bei Antwort bitte angeben)

Datum

Na/Ja

May 16, 1977

the idea to program our local inhomogeneous computer network (two different MINCAL-621, a PDP-11/20 and a PDP-10) in Concurrent Pascal. A code generator for the PDP-10 has just been completed and is currently being tested. In the course of writing a code generator for the PDP-10 (36 bits per word) we realized some of the shortcomings in the definition of the intermediate hypothetical machine which was originally conceived for byte-oriented machines.

Nevertheless, we have already executed a system of Concurrent Pascal processes on the PDP-10 and another one which communicated from one MINCAL-621 to another.

Our Concurrent Pascal Compiler is described in a report (in German):

CONCURRENT PASCAL Compiler für Kleinrechner
B. Brügge, B. Gisch, Th. Kahl, H. Linde, M. Mittelstein, H. Westphal
IfI-HH-B-30/76 (December 1976)

Sincerely yours,

DEC-10 (Hamburg-DECUS)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Implementor/Maintainer: E. Kisicki; H. -H. Nagel; Universtat Hamburg; Institut für Informatik; Schlüterstraße 66-72; D-2000 Hamburg 13, Germany; 040-4123-4151; TELEX: 214 732 uni hh d. Distributor: DECUS; Maynard, MA 01754; USA; 617/ 897-5111; TELEX: 94 8457; TWX: 710 347 0212.
2. MACHINE. Digital Equipment Corp. DEC-10. (Adapted to the DEC-20 by DEC).
3. SYSTEM CONFIGURATION. DEC TOPS-10 monitor using Concise Command Language (CCL). Uses KA-10 instruction set. Modifications to use KI-10 improved instruction set have been made by Charles Hedrick.
4. DISTRIBUTION. DECUS (Digital Equipment Corp. User's Society) Maynard MA 01754 (617/897-5111; TWX 710-3470212; TELEX 948457). Also DECUS Europe, P.O. Box 340, CH-1211 Geneva 26, Switzerland ((022) 42 79 50; TELEX 22593).
5. DOCUMENTATION. Machine retrievable manual included on distribution tape.
6. MAINTENANCE. No regular maintainance can be given.
7. STANDARD. Extensions: Functions FIRST and LAST for scalars; UPPERBOUND and LOWERBOUND for arrays; MIN and Max available as standard functions; procedures to determine the value of CCL options available; "OTHERS" in case statement; LOOP...EXIT IF...END statement; Initialization procedure.
8. MEASUREMENTS. (* No information provided. *)
9. RELIABILITY. Very good. First version released in 75/7. Distributed to at least 60 sites. Later version operational in 76/9. Latest version released to DECUS in 77/2.
10. DEVELOPMENT METHOD. Pascal-P2 and subsequent self bootstraps. Latest version dated 76/12/30.
11. LIBRARY SUPPORT. Symbolic post-mortem dump available. Interactive run-time source-level debugging package available. Separate compilation and inclusion in relocatable object code library of Pascal, FORTRAN, COBOL, ALGOL, and MACRO-10 assembler routines.

Dear Mr. Mickel,

as I have indicated by a letter mailed on February 14, 1977 our DECSys-10 Pascal compiler of December 30, 1976 is now distributed by DECUS. Mr. Nigel Derrett from Aarhus/Denmark pointed out one error in our PASCAL implementation of December 30, 1976: The attempt to pack a variable of a subrange type that requires exactly 35 bits - one less than an entire word - may result in an infinite loop.

Another, although minor, bug is connected with reading from TTY: in order to avoid unnecessary prompting of input during opening input from the TTY, the compiler checks whether any reading from TTY is requested during a program. The asterisk - prompting input to fill the first TTY-buffer - will only appear, if input from the TTY will be requested somewhere in the program. Unfortunately, arguments of standard procedures have not been included in this test. Therefore, if input from TTY appears within a program only as first argument to GETFILENAME, the input device TTY will not be opened automatically. An easy way around this weak point consists in inclusion of, e.g., a statement READLN(TTY).

Both errors will be removed in the next compiler version which, however, may take some time. I would like to investigate means to further optimize code generation by e.g. improving the allocation and use of registers. Since any change at such a sensitive area has to be made very carefully, it will take some time.

A PASCAL cross compiler running on the DECSys-10 and generating code for a German minicomputer Dietz MINCAL 621 is currently being converted to software paging of procedures: the pure code of procedure bodies is allocated in 128 Byte pages that may be loaded from disk to a certain core area and may be overwritten if that core area is needed. The nesting of 131 simple procedures has been successfully tested to verify the loading, overwriting and reloading of procedure bodies into core. Next we want to implement the PASCAL-S system (which is already available by a non-paging cross compiler for this MINCAL-621) by this new software paging PASCAL system and to compare its performance with paging versus the one without paging.

A compiler for Concurrent PASCAL has been developed by a group of students at our laboratory in collaboration with H. Kemen and myself. This is an implementation completely independent from that of Brinch-Hansen for the PDP-11/45. Our Concurrent Pascal compiler is executed as a PASCAL program on the DECSys-10 and generates code for a hypothetical intermediate machine which has been designed to facilitate easy code generation for Byte-oriented minicomputers. Two code generators have been implemented, one for the MINCAL-621 and one for the INTERDATA M85. Using this Concurrent Pascal implementation an assembler program to control our TV-periphery connected to the MINCAL-621 has been reimplemented as a system of Concurrent Pascal processes. The ease of designing a process system for actual applications in Concurrent Pascal has encouraged us to proceed with

Charles Hedrick wrote (received 77/07/28), "The version of PASCAL described herein answers most of the criticisms that caused me originally to declare it useless. The lack of strings and variable-size arrays is still a bother, but not serious. I chose to do all this to PASCAL because SAIL (the alternative vehicle) is too baroque to contemplate. My design goals were to give the PASCAL programmer access to all the facilities of the system - (1) in a manner that is not too badly machine-dependent nor requires him to know assembly language, but (2) in a manner that does not require a complex system of runtimes, e.g., one that simulates the OS/360 access methods. I believe the results have been successful. I'm still not sure whether it would be usable for hard-core data processing (ala COBOL), but it comes close. Mainly it is missing tape label processing and ISAM data sets. But one can now get the operating system to handle tape labels (PULSAR), and ISAM is not a primitive concept, at least for the 10."

DECsystem-10 System Programmers' PASCAL - an alternative PASCAL system for those who need full access to the facilities of TOPS-10, or who want to do data processing.

- (1) Charles L. Hedrick, Computer Science Dept., Rutgers University, New Brunswick, N.J. 08903
- (2) PDP-10, KI-10 and KL-10 CPUs only. Probably PDP-20, with minor changes.
- (3) TOPS-10 operating system. Virtual memory 6.01 or later monitor required. One minor feature requires 6.02.
- (4) The latest stable version is distributed through DECUS. The most recent experimental version can be obtained from me directly (at the above address), if you send a blank mastape and return postase.
- (5) A supplement to the Revised Report is included in the distribution in machine-readable form.
- (6) I am currently maintaining it and will continue to do so for the foreseeable future, but I probably will not do further development work (i.e. adding features). I hope this version will be superseded by an improved version from Hamburgs.
- (7) GO TO out of the current procedure is not supported. (Tricky to implement, and a terrible idea anyway.) Local files not implemented. (No ECS on a DEC-10 and simulation with randomly-named files seems unattractive.)
- (8) Compiler plus interactive debugging package (PASDDT) and a library of useful system functions. Completely integrated into CCL (COMPIL).
- (9) The compiler is quite reliable. The runtimes are reliable for standard PASCAL and the most commonly-used extensions. Some obscure corners of the extensions have not been well tested (mostly those involving user error recovery).
- (10) Modified version of the Hamburgs (Nasel) compiler. The latter was done from some edition of PASCAL-P, I believe, in several stages.

I did not start out intending to have my own version of PASCAL. Rather I wanted to test out a few ideas, with the hope that Prof. Nasel would adopt those that turned out to work, for use in the official DECsystem-10 PASCAL (which he maintains, and which is available from DECUS). I believe this will happen in the long run, but in the meantime I am publishing my experiences in the hope that

it will help other PASCAL implementors who are confronting similar problems. Thus this note is directed at fellow implementors, and is not intended as an advertisement for our version. (Indeed I recommend highly that other people use the Hamburgs version unless they absolutely require some of our features.) In the following, an asterisk (*) indicates a feature not in the edition we submitted to

DECUS. A plus sign (+) indicates a feature present in the Hamburgs version. I don't want to take credit for them, but thought other implementors might like to know about them.

(1) INITPROCEDURE(+): These specify initial values for variables. They do not compile code, but put the values in the initial core image directly. INITPROCEDURE BEGIN <assignment statements> END.

(2) OTHERS in case statement(+): OTHERS: <statement> will catch any cases not fitting anywhere else.

(3) LOOP(+): Allows a loop with exactly one exit in the middle. Note that this is still a one-in-one-out construct. LOOP <statements> EXIT IF <Boolean expression>; <statements> END

(4) Program statement(+). There was some question what the PROGRAM statement should do in interactive implementations. I believe Hamburgs' solution is a good one. If any files are listed in the PROGRAM statement, the program begins with a dialogue asking for specifications for them. It is important that this dialogue can be suppressed by not listing any files. This gives the program the option of setting the file names in some other way and specifying them to the RESET or REWRITE directly. This follows my

BASIC DESIGN PRINCIPLE: It should be possible to write a program that cannot be identified by its users as a PASCAL program. I.e. one should be able to take over error handling and file specifying if desired.

(3) "Interactive" files: RESET does an implicit GET in official PASCAL. This causes PASCAL programs to try to read from the user's terminal before starting the program when it is INPUT. That makes it impossible for the program to output a prompting message first, or to write a program that doesn't have terminal input at all. Most implementations on interactive systems allow one to specify a file as interactive. Then when it is RESET, no GET is done. Instead the buffer is filled with null (in our case) or blank (for the CYBER, which doesn't have a null), and EOLN is set. This ability is also useful for mastapes, where one might wish to issue a positioning command (space forward, rewind, etc.) before doing the first GET. The CYBER specifies files as interactive by putting a slash (/) after their name in the PROGRAM statement. We make this an option specified in the RESET statement. (See below.) Putting it in the RESET statement is helpful since not all files are listed in our PROGRAM statement. In our implementation, the user's terminal is the special file TTY, and is always interactive.

(6) End of line: The Revised Report seems to require us to set the buffer to blank at end of line. Alas, the DEC-10 has several line-terminator characters. Thus one cannot tell which one has occurred. We put the actual terminator in the buffer. A blank seems useful for those systems that do not have line-terminator characters (e.g. CYBER). Programs that use IF EOLN THEN READLN will work either way. (Our READLN also skips the line feed if the initial terminator is carriage return.) Note that it requires two GET's to skip a carriage-return, line-feed sequence, although a single READLN will do so.

(7) End of file: In PASCAL EOF is normally false for input and true for output. This lack of symmetry complicates the I/O runtimes needlessly, confuses users, and makes the implementation of update mode difficult. In update mode, one can do both GET and PUT on the same file. However, if EOF is true, GET will give an error, and if EOF is false, PUT will give an error. We have not really solved this problem. Fortunately under most circumstances PUTX is used rather than PUT in update mode.

(8) I/O to strings. STRSET(file,array[,start[,end]]) allows regular input to be done from the array, starting at element start and going through end. Similarly STRWRITE for output. This allows conversion from text to integer and visa versa using the standard READ and WRITE. It also provides a sort of poor man's simulation of files in ECS (which we don't have). I am not enthusiastic about this feature, however, and would be happy to see it go away. One must require that the array be declared at the same lexical level as the file, be global to it, or be on the heap. Otherwise one could exit the block where the array is defined and have the file pointing into nowhere.

(9) READ applied to strings: READ(file,array[:length var]) will read characters into the array until the next end of line. This is really amazingly useful for conducting dialogues with the user. The alternative seems to be to require string quotes to delimit the string, or do something like READ(file,array;set[:length]), where the user specifies a set of break characters. The idea of string quotes is very tacky, impeding the construction of simple dialogues. The break set idea is a good one, that we just have gotten around to. If a variable is specified after the colon, it is set to the number of characters read. If more characters are typed than the size of the array, the extras are ignored (but counted in length, so the program can tell what has happened). If too few are typed, the rest of the array is filled with blanks.

(10) RESET and REWRITE: Our RESET and REWRITE are REWRITE(file[,filespec[,implementation-dependent stuff]]) and RESET(file[,filespec[,interactive?[implementation-dependent]]]). Filespec is a string (PACKED ARRAY OF CHAR) of any length, including a literal in string quotes. It contains a device/file specification in the standard DEC-10 format. Interactive is true to suppress the implicit GET, as described above. The implementation-dependent stuff allows the user to control protection, version number, date, I/O mode, buffering, etc. In particular, it allows him to specify buffered or unbuffered I/O (*), and to declare that the file is blocked (in the COBOL sense) (*). It also allows him to suppress the normal runtime error messages. ERSTAT(file) can then be used to see what errors, if any, occurred.

(11) Variable size records(*): One can declare a file to consist of a type that involves variants. Then one can use GET(file[,variant],...[:size]), and similarly for PUT. As with NEW, this causes only the appropriate number of words to be read or written. :size is used when the last element in the declaration is an array. It specifies that only the first <size> elements are to be

used. *NEXTBLOCK (file, suppress GET?) gets you to the beginning of the next logical block, should you need to skip.*

(12) UPDATE(*): UPDATE is like RESET, except that it allows records to be updated in place. To use this, read the record into the PASCAL buffer with GET. The contents of the buffer can then be revised. PUTX(file) causes then new contents of the buffer to replace the original record in the file. Exactly the same number of bytes is changed as was read by the original GET. No surrounding records are affected, no matter what sort of blocking is involved (except for unbuffered I/O). PUT may also be used in update mode when unbuffered I/O is being done.

(13) Random access I/O(*): On the DEC-10 random access devices have fixed size physical blocks. No standard "access methods" are provided. (Basically we have only QSAM and BSAM with format U records.) Thus one can either simulate the COBOL runtime system, or stick with low level primitives. The latter seems consistent with PASCAL's philosophy. It has the disadvantage of being machine-dependent, however, since it depends upon the architecture of the DEC file system. (This architecture is probably quite common outside of IBM, however.) We use an index into the file. SETPOS(file,index[,suppress GET]) sets one to the index'th byte in the file. (NB: byte, not disk block. We perform the buffer manipulation needed to set to the specified byte.) CURPOS(file) returns the position at the beginning of the last record read or written.

(14) APPEND: APPEND is like REWRITE, but begins writing at the end of an existing file.

(15) BREAK, BREAKIN: BREAK forces out the buffer in buffered modes. BREAKIN clears the buffer in buffered modes. They are used before or after master positionings, etc. BREAK would be needed to force output to the terminal, except that the file TTY is handled with a special kind of I/O that does not use buffers.

(16) RENAME - RENAME(file,filespec[,implementation-dependent]) renames the file open on <file>.

(17) CLOSE(file) closes the file. This is needed to make the file safe in case the system crashes. It also releases the I/O channel for use by other files. (There is a maximum of 16 I/O channels on the DEC-10.) All files are automatically closed at the end of the program.

(18) DISMISS(file) aborts creation of an output file.

(19) UPCASE(file,Boolean expression) controls mapping of lower case to upper case.

(20) Routines are available to handle interrupts (via FSI), do interjob communication (IPCF), and call the standard DEC-10 command scanner (SCAN/WILD).

Dietz MINCAL 621 (Hamburg)

See the letter from H.-H. Nagel under DEC-10.

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. H. -H. Nagel; Universtat Hamburg; Schluterstrasse 66-72; D-2000 Hamburg 13, Germany; 040-4123-4151; TELEX: 214 732 uni hh d.
2. MACHINE. DEC-10 cross-compiler producing code for the Dietz MINCAL 621 minicomputer.
3. SYSTEM CONFIGURATION. (* No information provided. *)
4. DISTRIBUTION. (* No information provided. *)
5. DOCUMENTATION. (* No information provided. *)
6. MAINTENANCE. (* No information provided. *) Currently being converted to software paging of procedures.
7. STANDARD. (* No information provided. *)

8. MEASUREMENTS. The pure code of procedure bodies is allocated in 128 byte pages that may be loaded from disk to a certain core area and may be overwritten if that core area is needed. (* No information provided on compilation speed, or execution speed or space.*)

9. RELIABILITY. (* No information provided. *) The nesting of 131 simple procedures has been successfully tested to verify the loading, overwriting and reloading of procedure bodies into core.

10. DEVELOPMENT METHOD. Cross-compiler on the DEC-10 that generates code for the Dietz MINCAL 621 minicomputer.

11. LIBRARY SUPPORT. (* No information provided. *)

FOXBORO Fox-1

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Bob Matherne; Jim Pownell; The Foxboro Company; Foxboro, MA 02035; 617/ 543-8750.

2. MACHINE. Foxboro Fox-1 (16-bit minicomputer designed primarily for industrial process control applications).

3. SYSTEM CONFIGURATION. (* No information provided. *)

4. DISTRIBUTION. (* No information provided. *)

5. DOCUMENTATION. (* No information provided. *)

6. MAINTENANCE. (* No information provided. *)

7. STANDARD. Restrictions: sets limited to 48 members. Extensions: STOP statement, program controlled trace facility, optional profiler.

8. MEASUREMENTS.
 interpretation speed--fairly slow
 interpretation space--14K (much overlaying involved)
 execution speed--fairly slow (interpreted with software paging)
 execution space--(* No information provided. *)

9. RELIABILITY. (* No information provided. *)

10. DEVELOPMENT METHOD. Interpreter of P-code written in FORTRAN based on Pascal-P. (* Person-months to create system not reported. *)

11. LIBRARY SUPPORT. (* No information provided. *)

Fujitsu FACOM 230

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Masato Takeichi, formerly at Department of Math. Engineering and Instr. Physics, University of Tokyo, Bunkyo-ku, Tokyo 113, Japan. Present address: Department of Computer Science, University of Electro-Communications, 1-5-1 Chofugaoka, Chofu-shi Tokyo 182, Japan.

2. MACHINE. FACOM 230-38, 224K bytes.

3. SYSTEM CONFIGURATION. OS2/VS. (* Minimum hardware required not reported. *)

4. DISTRIBUTION. (* no information provided. *)

5. DOCUMENTATION. See the article "Pascal Implementation and Experience", by Masato Takeichi, Journal of the Faculty of Engineering, University of Tokyo 34:1 pp 129-136.

6. MAINTENANCE. (* no information provided. *)

7. STANDARD. Restrictions: No local file variables; no parametric procedures.

8. MEASUREMENTS. Self compiles in 309 sec. Compiler object code is 117K bytes, monitor is 8K bytes, and self compilation requires 43K bytes of data store. Execution times, relative to Fortran, are given in the following table.

	OS2/VS Fortran	Pascal
Matrix multiply	1	1.35
Sort	1	1.24
Additive partition	1	0.96
Character count	1	0.63

9. RELIABILITY. Working very well. (* Number of sites and first date of release not reported. *)

10. DEVELOPMENT METHOD. Based on H.H.Nageli's Trunk compiler (5800 lines of Pascal), with a Pascal monitor written in FASP. The initial version began working in October, 1975, after 2-3 months of work.

11. LIBRARY SUPPORT. (* no information provided. *)

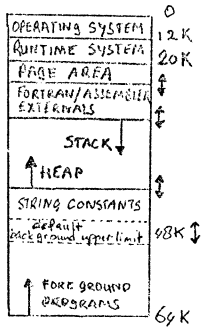
Harris/4

From O. W. van Wijk (*77/08/15*)
 TNO-IBBC, P.O. BOX 49, DELFT, HOLLAND. TEL 015-138222 TELEX 33567

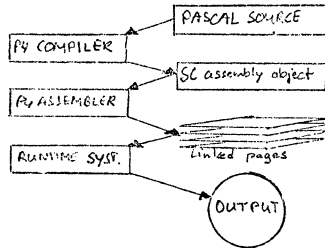
IMPLEMENTING PASCAL P4 SYSTEM ON A HARRIS/4 COMPUTER.
 At TNO-ibbc we have a Harris/4 machine with 64K of 24 bits words. The implementation was done as a student task by O.W. van Wijk from Delft University of Technology, departement of mathematics. Starting with an implementation kit obtained from E.T.H.-Zürich it took about 600 hours to get a running version of the compiler that could compile itself. The Pascal system then consisted of the p4-compiler, an assembler for the pcode instructions, written in fortran, and a runtime system of about 4K Harris assembler code. The fortran p4-assembler scans the pcode two times and generates during the second scan linked pages of subroutine calls. One page = 1024 words code + 2 words link, these pages are stored on disk. Running a Pascal program is done by loading the runtime system, which is a normal background program. This program has two parameters, one to specify the disk file containing the pages with the program object, the second to specify the number of pages held in core. The pages are allocated in the page area and swapped according the 'least recently used' algorithm. In this construction the code size of a Pascal program is not limited by the size of the machines core, and allows the use of maximum 37K of the available 42K core, for Pascal data on stack and heap. Running Pascal this way with 15 pages in core, it takes the compiler about 6 min. to compile itself. The second stage of the implementation was making some extentions to the compiler and slight changes in the code generation. Extentions were made to allow the use of external procedures of different kinds of source languages. (Pascal, fortran, Harris assembler.) External procedures are declared in a prefix to the Pascal program. The type of parameters of external procedures is restricted to the standard types. Software to create libraries was developed. External procedures of Pascal source are stored on disk in pcode form and included by the p4-assembler, now also written in Pascal, on the page file. External procedures of fortran/assembler source are stored on disk as relocatable modules, included by the p4-assembler as one record on the page file, loaded and relocated by

the runtime system. The use of external procedures also allows a kind of fortran-like direct access I/O, which was, among the use of existing programs, the main reason to make this extension.

core layout



scheme for running Pascal



so on. This now appears to be working and I have run a few hand compiled programs through it. However memory size limited the amount of ~~code~~ ^{from} I could give the CODE and STACK arrays. This is alright for running small programs but the compiler itself would not fit. I have thus taken that interpreter and split into two phases - a load phase and a run phase. The load phase now does two passes over the P4 code to produce the internal form of code on a disc file, ~~than~~ ^{rather} in an array. The run phase is then a stripped down version of the normal interpreter with all irrelevant detail (post mortem dumps, trig functions etc.) eliminated. This will have a bigish CODE array and will basically operate on a virtual storage concept. This is still in FORTRAN but I will rewrite it in HP assembly soon. Thus as soon as I can get a compiler for the HP compiled I should be able to compile programmes, but I feel that recompiling the compiler will be beyond me.

Now for the Univac. I am modifying the original FORTRAN interpreter to make allowances for the difference in architecture, I/O etc. and will move that on to the Univac soon. Then by the usual bootstrap operation I will get PASCAL up there. That done I will probably bootstrap a more effective (non ininterperative) system probably the BELFAST compiler for the 1900 series.

Thus by the end of July I should be able to compile programmes (albeit slowly) on both machines and by the end of the year have efficient systems going on each. Next year all my students will learn PASCAL of as a first language as a matter of course in their algorithms and problem solving course.

Heathkit H-11

(* This machine is based on the LSI-11 microprocessor from DEC and it is believed that the DEC LSI-11 (San Diego) implementation will run on this machine; though nothing definite has been reported. *)

Hewlett Packard HP-21MX (Durban)

See also HERE AND THERE News section under Tao-Yang Hsieh.

UNIVERSITY OF DURBAN-WESTVILLE



Telephone: 821211
Telegrams: INKOL
Ref.

Private Bag X54001,
Durban,
4000.

DEPARTMENT OF COMPUTER SCIENCE
C.C. HANDLEY.

Preliminary report on implementation of PASCAL on
HP21MX and Univac 90/30.

We bought the P4 system from Zurich early this year and after a few hassles with block sizes, end of files and character sets, managed to get the files to tape and also listed. Since then I have been attacking the problem on two roughly parallel fronts, namely implementation of the PASCAL defined by the P4 system on the two machines mentioned.

My major effort has been on the HP as I have easier access to it. I have rewritten the P4 interpreter in (of all things) FORTRAN chiefly because I could make use of its horrible features, such as EQUIVALENCing REALs and INTEGERS for the stack and

Hewlett Packard HP-2100 (Trieste, Italy)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Mattia Hmeljak; Istituto di Electrotechnica ed Electronica; Universita di Trieste; Trieste, Italy; Tel. 040-733033.
2. MACHINE. Hewlett Packard HP-2100.
3. SYSTEM CONFIGURATION. (* No information provided. *)
4. DISTRIBUTION. (* Unknown, implementation not yet complete. *)
5. DOCUMENTATION. (* Unknown, implementation not yet complete. *)
6. MAINTENANCE. (* Unknown, implementation not yet complete. *)
7. STANDARD. (* No information provided. *)
8. MEASUREMENTS. (* Unknown, implementation not yet complete. *)
9. RELIABILITY. (* Unknown, implementation not yet complete. *)
10. DEVELOPMENT METHOD. A P-code interpreter written in HP-Algol.
11. LIBRARY SUPPORT. (* No information provided. *)

Hewlett Packard HP-3000 -- Miscellaneous

See also HERE AND THERE News section under Kurt Cockrum and R. A. Lovestedt (who works at Boeing (CAD) in interactive graphics).

Also, on 77/07/25, Edward O. Thorland, Computer Center, Luther College, Decorah, IA 52101 (319/387-1043), phoned that he was ordering the P4 compiler to start an HP-3000 implementation.

Hewlett Packard HP-3000 (Santa Clara)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Ronald Danielson; University of Santa Clara; Santa Clara, CA 95093; 408/ 984-4482.
2. MACHINE. Hewlett-Packard HP-3000/Series II.
3. SYSTEM CONFIGURATION. Runs under MPE with 256K words memory.
4. DISTRIBUTION. (* Unknown, project not yet complete. *) A very rough completion date is 78/01.
5. DOCUMENTATION. (* Unknown, project not yet complete. *)
6. MAINTENANCE. (* Unknown, project not yet complete. *)
7. STANDARD. (* No information provided. *)
8. MEASUREMENTS. (* Unknown, project not yet complete. *)
9. RELIABILITY. (* Unknown, project not yet complete. *)
10. DEVELOPMENT METHOD. Via Pascal-P.
11. LIBRARY SUPPORT. (* No information provided. *)

HITACHI Hitac 8800/8700 (Tokyo)

(* See also implementation notes for IBM 360/370. *)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Teruo Hikita; Kiyoshi Ishihata; Department of Information Science; University of Tokyo; Tokyo, 113, Japan; 03-812-2111 x2947.
2. MACHINE. Hitac 8800/8700.
3. SYSTEM CONFIGURATION. OS7 (Hitachi). (* Minimum hardware requirements not reported. *)
4. DISTRIBUTION. Reluctantly.
5. DOCUMENTATION. 'Pascal 8000 Reference Manual', and 'Bootstrapping Pascal using a Trunk' are available from above address. (* Apparently no machine retrievable documentation. *)
6. MAINTENANCE. No formal support can be promised. Bug reports are welcome.
7. STANDARD. differences: standard procedures pack and unpack not implemented; files must be declared at main program level; extra loop control structures; "value" initialization part.

8. MEASUREMENTS. Compiler object size is about 100 kilobytes. compilation speed--about 350 lines/second. execution speed--comparable to FORTRAN-compiled objects. execution space--(* No information provided. *)

9. RELIABILITY. Good. (* Number of sites using system and date first released not reported. *)

10. DEVELOPMENT METHOD. A 5200 line Pascal program created by modifying Naegeli's Trunk compiler and bootstrapping it by Pascal-P. Required about 3 person-months to complete.

11. LIBRARY SUPPORT. None - the compiler produces absolute code, not relocatable modules.

Honeywell H316

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Robert A. Stryk; 5441 Halifax Lane; Edina, MN 55424; 612/ 887-4356.
2. MACHINE. Honeywell H-316.
3. SYSTEM CONFIGURATION. (* No information provided. *)
4. DISTRIBUTION. (* No information provided. *)
5. DOCUMENTATION. (* No information provided. *)
6. MAINTENANCE. (* No information provided. *)
7. STANDARD. A modified implementation of Concurrent Pascal, which varies from Standard Pascal.
8. MEASUREMENTS. (* No information provided. *)
9. RELIABILITY. (* no information provided. *)
10. DEVELOPMENT METHOD. (* No information provided. *)
11. LIBRARY SUPPORT. (* No information provided. *)

Honeywell 6000

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Implementor: W. Morven Gentleman; Mathematics Faculty Computing Facility; University of Waterloo; Waterloo, ONT. N2L 3G1; CANADA; 519/ 885-1211. Distributor: Honeywell Information Systems; 7400 Metro Blvd.; Edina, MN 55435; (* See local HIS sales office. *)
2. MACHINE. Honeywell 6000, level 66 series. Operates under GCOS (TSS). Currently (* 76/03/08 *) a DRL TASK version is under consideration.
3. SYSTEM CONFIGURATION. Honeywell level 66 or 6000 series with EIS. Minimum of 26k words.
4. DISTRIBUTION. (* No information provided. *)
5. DOCUMENTATION. From Honeywell Information Systems; Publication Dept.; MS-339; 40 Guest St.; Brighton, MA 02135: "A Pascal Product Brief", (#AW66, free), 2 pg. (marketing oriented) and "Pascal User's Guide", (#AW65, \$1.30), 30 pg. (reference manual). Machine retrievable supplement to Pascal User Manual and Report; also includes extensions, restrictions, known bugs, etc.--about 45 pages total.

6. MAINTENANCE. Supported by HIS.

7. STANDARD. Restrictions:

- Program statement not accepted, replaced by required procedure 'main'.
- No files with components of type file.
- Only files of type char may be read or written.
- Sets limited to 72 members (no sets of char).

Extensions:

- Files may be opened dynamically.
- Extended file handling is available.
- External separately compiled Pascal and FORTRAN procedures may be used.
- Various procedures and functions to provide access to operating system.
- Optional left-to-right evaluation for Boolean expressions and if statements.
- 'else' clause in case statement.

8. MEASUREMENTS.

compilation space--minimum of 26k words. Typical programs require less than 30k words
compilation speed--(* No information provided. *)
execution space--can be as small as 4-5k words depending on the program and the
Pascal support routines required.
execution speed--(* No information provided. *)
(* How this compares to FORTRAN and other languages not reported. *)

9. RELIABILITY. (* No information provided on reliability or number of sites using system. *) Distributed since 76/05.

10. DEVELOPMENT METHOD. (* No information provided. *)

11. LIBRARY SUPPORT. Separately compiled Pascal and FORTRAN routines may be saved and called from user specified libraries at run time. A post-mortem debugger is planned, but presently (* 76/10/25 *) far from being implemented.

IBM Series 1

Gus Bjorklund, 2250 Coppersmith Square, Reston, VA 22091, reported in late June that he had an IBM Series 1 implementation nearly complete and should be finished by 77/9.

IBM 360, 370 -- Introduction

As with DEC PDP-11s, requests for and news about IBM 360/370 implementations abound. Last year we tracked over ten different implementation efforts. We have news for this issue of PUGN regarding improvements to the Hitac-8000 compatible compiler which has been converted to IBM systems by the Australian AEC, as well as about the Manitoba and SUNY Stony Brook compilers. Following these, summaries are given for other known implementations based on news from last year.

Teruo Hikita's University of Tokyo Hitac-8000 compiler attracted our interest last fall when it was announced as being (1) written in Pascal, (2) very fast (as fast as the Fortran compiler), and (3) adaptable to IBM systems. Apparently the project ran short of resources and not much news developed until Joseph Mezzaroba (PUGN #8) coaxed a copy and with a team of graduate students had it running in three weeks under DOS. This summer news came from the Australian Atomic Energy Commission (AEC) that they have finished the job with respect to making Hitac-8000 Pascal available on IBM systems to non-commercial sites only. So now we list Hikita's compiler under Hitachi Hitac-8000 and replace its IBM entry with the AAEC. Joseph Mezzaroba indicates that they (at Villanova) have switched from their version of the Hitac compiler to the AAEC version.

Our thanks to W. Bruce Foulkes for sending us new and complete information on his implementation which now, we are pleased to find, is improved, upgraded, and more standard!

Also thanks to Richard Kieburz for sending new information plus an explanation as to the cost of SUNY Stony Brook Pascal. It is a credit to their dedication to Pascal that they continue to support an IBM compiler when they no longer have IBM equipment!

One final note: Thanks to Philip Malcolm (Computer Associates, Park House, Park Street, Maidenhead, Berkshire SL6 1SL United Kingdom) who phoned twice this summer to give us information about plans to evaluate many IBM implementations for the the purpose of writing production software (on 6 or 8 operating systems!). He found that:

- 1) The Technical University of Berlin has dropped their effort at a P4 implementation and has obtained Imperial College, London's version of a P4 implementation, which "runs nicely". (* Our only problem here at PUG(USA) is that no one at Imperial College has told us in writing what they are doing. Here are the names of the PUG members at Imperial College: P.W.R.Clarke, R.A.Francis, Jeff Kramer, Stuart James McRae, Greg Pugh, David Slater, Iain Stinson, and Dave Thomas. Their address: Department of Computing and Control, New Huxley Building, Imperial College London, 180 Queensgate, London, England SW7 2AZ United Kingdom (phone: 01-589-5111).

Well, how about it? *)

- 2) He was procuring the Australian AEC version of Hikita's Hitac-8000 compiler.
- 3) He rejected the SUNY Stony Brook version.
- 4) He could not get the source for the Manitoba version or the source of the run time system of the Grenoble version.
- 5) Ruled out virtual memory, and thus the Vancouver version.
- 6) Still awaited news from Oslo and from Stanford.

Philip promised a followup report and evaluation and we certainly look forward to it.

-Andy Mickel

AUSTRALIAN ATOMIC ENERGY COMMISSION

NUCLEAR SCIENCE AND TECHNOLOGY BRANCH

RESEARCH ESTABLISHMENT, NEW ILLAWARRA ROAD, LUCAS HEIGHTS

TELEGRAMS: ATOMRE, SYDNEY
TELEX: 24562
TELEPHONE: 531-0111

IN REPLY PLEASE QUOTE: JMT.mwb

Mr. Andy Mickel,
Editor, Pascal Newsletter,
University Computer Center,
227 Experimental Engineering Bldg.,
University of Minnesota,
MINNEAPOLIS MN 55455.



ADDRESS ALL MAIL TO:
AAEC RESEARCH ESTABLISHMENT
PRIVATE MAIL BAG, SUTHERLAND 2232
N.S.W. AUSTRALIA
20th June, 1977.

Dear Mr. Mickel,

On the 18th March, 1977, we received a copy of Pascal 8000 from Professor Teruo Hikita, University of Tokyo, Japan. Pascal 8000 was developed for use on a Hitachi "M" series computer, reputed to operate under an IBM370 compatible operating system. With a few modifications to the run-time system, we brought up Pascal on our IBM360/65 in only a few days.

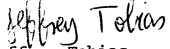
Basically, the compiler is excellent. The language implemented is very nearly standard Pascal with some very significant extensions. The compiler itself is written in Pascal 8000, and produces very efficient and, in general, compact machine code. In the majority of cases, execution speed of compiled code is faster than that of a similar program compiled under FORTRAN G. The original version of the compiler compiled itself in about 290K bytes.

Since March, we have been developing further the compiler for OS and VS on IBM360 and IBM370 computers. We have completely re-written the run time system in assembler (it now occupies 6K bytes instead of 36K) and in so doing have extended and implemented various features such as local file processing (in the true sense, not just temporary datasets), extended addressing (procedures can now be up to 24K bytes in length, rather than 4K), and various traceback and post mortem dump routines. Further, files of RECFM=F(B) (A) and V(B) (S) (A) are supported for both input and output.

Several areas of the compiler have been restructured and extended. Procedures PACK and UNPACK are implemented, so now a true superset of standard Pascal is accepted by the compiler. Internal mechanisms of code generation were changed and new functions and standard type names added. Exponentiation has been included, parts of the lexical analyzer have been rewritten, and code has been optimized in several areas. It is now possible to compile small programs in 128K, and the compiler compiles itself in 210K (corresponding figures were 176K and 286K in the original version).

We are now making this modified IBM360/370 version of Pascal 8000 available. Distribution arrangements have not quite been finalised, however, it is envisaged that support for the system will be provided. All enquiries are very welcome.

Yours sincerely,


 Jeffrey Tobias Systems Design Section
 Gordon W. Cox

19 August, 1977.

IN REPLY PLEASE QUOTE: JMT.mwb

Mr. Andy Mickel,
 Editor, Pascal Newsletter,
 University Computer Center,
 227 Experimental Engineering Building,
 University of Minnesota,
 MINNEAPOLIS MN 55455 U.S.A.

Dear Mr. Mickel,

We have now finalised the distribution arrangements of our IBM360/370 version of Pascal 8000. As you know, this system is based on Hikita's PASCAL 8000 compiler, which has been extensively modified and adapted by us for the O.S. family of operating systems. Important features of this system are small memory requirement for compilation (128K for small programs), extensive file support including local files, and full traceback and post-mortem dump facilities.

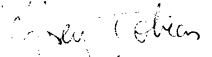

We are distributing this system, including documentation, source and object code for a fee of A\$50. This is to cover handling expenses only; no charge is being made for development of the system. We also require that an agreement be signed to the effect that the provided system will not be used for profitable purposes.

Our policy with respect to maintenance of the system is that no written undertaking can be given; we are, however, very keen to hear of any problems that may arise, and we hope to be able to provide solutions.

The system currently being distributed produces object code in a form suitable for its own internal loader; the code produced by the compiler can be saved for later execution (one example of this being the compiler itself), but cannot be linked with other modules. We are, however, in the final testing stages of a version that produces standard IBM linkage-editor compatible object decks, and linking to externally compiled Pascal, Fortran and Assembler routines is supported. This new version will be distributed as well as the original, as each has its own advantages.

We are enclosing a brief description of our Pascal System. All enquiries are very welcome, and order forms are available from us.

Yours sincerely,



 J. M. Tobias
 G. W. Cox
 Systems Design Section

PASCAL 8000 - IBM360/370 VERSION

1. Fully implements 'standard Pascal', with some very significant extensions.
2. Compiler is itself written in Pascal.
3. Total system size is relatively small. Moderately sized programs may be compiled in 128K, with the compiler able to compile itself in 210K.
4. Datasets of RECFM = F[B][S][A], V[B][S][A] or U[A] are supported.
5. Files may be external or local. Thus, structures such as 'array of files' are available. External files are named in the program statement, local files are not. Both external and local files may be declared in a procedure at any level.
6. Arithmetic is performed in double precision.
7. Control of input and output formatting is as described in Jensen and Wirth (1975). The form is

$$x[:n[:m]]$$
, where n and m are integer expressions.

Elements of type packed array of char may now be read on input. Procedures read and write have also been extended to apply to both non-text and text files.
8. Procedures have a maximum size, depending on their static nesting level. This size ranges from 4K to 24K of compiled code.
9. Some of the language extensions include:
 - (i) Constant definitions for structured types. It is therefore possible to have arrays, records and sets as constants.
 - (ii) A 'value' statement of variable initialisation.
 - (iii) A 'forall' statement of the form:

$$\text{forall } \langle \text{control variable} \rangle \text{ in } \langle \text{expression} \rangle \text{ do } \langle \text{statement} \rangle$$
 where $\langle \text{expression} \rangle$ is of type set.

- (iv) A 'loop' statement, specifying that a group of statements should be repeatedly executed until an 'event' is encountered. Control may then be transferred to a statement labelled by that event.
- (v) The types of parameters of procedures or functions passed as parameters must be specified explicitly, and this enables the compiler to guarantee integrity.
- (vi) The 'type identifier', restriction in a procedure skeleton has been relaxed to allow 'type'.
- (vii) Functions 'pack' and 'unpack' are supported, as are packed structures in general.
- (viii) Exponentiation is fully supported, and is used via the double character symbol '**'.
- (ix) A 'type-change' function has been introduced that extends the role of 'chr' and 'ord'.
- (x) Case-tag lists may now range over a number of constants, without explicitly having to list each constant.

The range is denoted by:

<constant> .. <constant>

Thus,

4,6..10,15,30..45

is now a valid case tag list.

A default exit is also supplied via

else: <statement>

i.e. else: is a valid case tag in every case statement. This path will be used if none of the other tags match.

- 10. Execution errors terminate in a post-mortem dump, providing a complete execution history that includes procedure invocations, variable values, type of error, etc.
 - 11. Object code produced by the compiler is compact and efficient. In general, execution speed of PASCAL 8000 programs is faster than that of similar programs written in FORTRAN G level.
 - 12. Maximum set size is 64 elements.
 - 13. Procedure 'new' is fully supported, obtaining the minimum heap requirements as specified by variant tags. Procedures 'mark' and 'release' are also supported.
- Procedure 'dispose' is not supported.

Reference

- 1. 'Pascal - User Manual and Report', Kathleen Jensen and Niklaus Wirth, Springer Verlag, Second Edition, 1975.

IBM 360,370 (Manitoba)

July 13, 1977

Dear Andy:

Enclosed is information about the latest release of the Manitoba Pascal Compiler. This version is more standard, more complete, cleaner, and more reliable than previous releases. I am sending complimentary copies of this release to the thirty sites which have earlier versions of the compiler. Work on the file support is now in progress.

1. Implementor and Distributor

Dr. W. Bruce Foulkes
Department of Computer Science
University of Manitoba
Winnipeg, Manitoba
Canada R3T 2N2
(204) 269-3363

2. Machine

IBM 360/370, AMDAHL 470

3. Environment

The compiler has been installed under the following operating systems: MFT, MVT, VS1, VS2, MVS and CMS (with modifications to the interface routines).
A region of 180K bytes is required. (Absolute minimum is approximately 160K bytes.)

4. Distribution

First released in December 1975 to 10 sites. July 1976 version released to 30 sites. June 1977 release is now available.
Distributed in load module and object module form with assembler source of the interface routines.
We require a "SOFTWARE RELEASE AGREEMENT" to be signed.
Cost: \$50 for June 1977 release.
Medium: 600-ft. 9-track tape which we provide.

5. Documentation

Two manuals are included in machine-retrievable form:
"MANITOBA PASCAL USER GUIDE" - 48 pages.
"MANITOBA PASCAL CODE GENERATION" - 84 pages.

6. Maintenance

I have supported the compiler since December 1975 and intend to continue, but I am not in a position to make a formal commitment.
I welcome comments, suggestions, and even bug reports.

7. Language Supported

Many non-standard features of earlier versions have been eliminated in this release. Some examples are:

- predefined types are now global.
- output formatting has been standardized.
- [and] are now allowed for arrays and sets.
- (* and *) are now allowed for comments.
- PACKED is ignored.
- program header is optional.
- etc.

Pascal Compiler Project
Dept. of Computer Science
State Univ. of New York
at Stony Brook
Stony Brook, N. Y. 11794
July 15, 1977

telephone: (516) 246-7146

IBM 360,370 (Stony Brook)

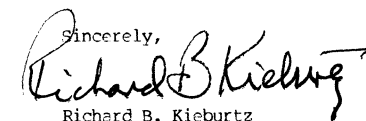
Andy Mickel
University Computer Center
227 Exp Engr
University of Minnesota
Minneapolis, Minnesota 55455

Dear Andy:

Enclosed is an announcement of the newest release of the Stony Brook Pascal/360 compilers, for publication in the Pascal Newsletter. We have distributed over 100 copies of Release 1, and under our distribution policy, those who ordered Release 1 will receive Release 2 automatically.

As some of your readers may know, Stony Brook has not had an IBM 370 for over a year and a half, and we now no longer have even the Univac Spectra 70 on which the Pascal/360 compiler could be executed. Our present mode of operation involves doing all machine-independent development work on a Univac 1110 at Stony Brook, then installing new developments and testing the operating system interface on an IBM 360/65 at Polytechnic Institute of New York. This arrangement is slightly inconvenient, but it works. Needless to say, we must pay for machine time and we are just breaking even (so far) on our \$175 distribution fee.

Sincerely,



Richard B. Kieburz

StonyBrook

STONY BROOK PASCAL/360; RELEASE 2.0 AND RELEASE 2.S

The second release of the Stony Brook Pascal compiler for IBM 360 and 370 computers is now ready for distribution. Release 2.0 is a production compiler with facilities for linkage to externally compiled Pascal program modules and Fortran functions or subroutines. The compiler generates IBM object modules which can be processed either by the OS/360 linkage editor or by the linking loader. Some language extensions have been installed in this release, while others are currently being implemented.

Release 2.S is a fast, compile-load-and-go version that implements Standard Pascal without extensions. It can be installed under HASP Autobatch for economical batch processing of small jobs. The only significant restrictions imposed by 2.S are on the maximum program size that can be compiled (dependent on the partition size allocated to the compiler). When the compiler is run under Autobatch, nonstandard files will be restricted to internal files only. This is a restriction imposed by Autobatch, not by the

Restrictions

- Only the standard input and output files SYSIN and SYSPRINT are supported. All I/O is accomplished through the use of READ, READLN, WRITE, WRITELN, EOLN, and EOF. (This is a temporary restriction. Work on file support is in progress.)
- Procedures PACK and UNPACK are not implemented.
- Branches to global labels are not allowed.
- SETs of characters are not supported (temporary restriction).
- Built-in procedures and functions are not accepted as actual parameters.
- The maximum static nesting of procedure and function declarations is 5.
- Program segments are restricted to 4K bytes of code.

Extensions

- Three additional scalar types are supported: SHORTINTEGER (SHRTINT), LONGREAL (LREAL), and STRING (n) $1 \leq n \leq 256$.
- A substring operation is provided.
- Formatted input is provided and input of BOOLEAN and STRING values is permitted.
- hexadecimal constants are supported.

8. Compiler Development

The compiler is one pass and uses a top-down parsing strategy. All semantic routines are written in PL360 (about 13,000 lines) and system interface routines in Assembler (500 lines).

The run-time routines are written in PL360 (1600 lines) with an Assembler interface (500 lines).

Compile speed averages 500-1000 lines of source per second on an IBM 370/168.

Considerable effort has been spent on localized optimizations in areas such as array subscripting, record field accessing and boolean expression evaluation with the aim of producing a compiler suitable for the compilation of application programs. (A 3100-line Assembler/Loader/Interpreter system has been written locally in PASCAL and is in production use on our student terminal.)

The compiler has been running on our express student terminal since January 1976.

I haven't run any speed tests recently, but execution speed seems competitive with the IBM Fortran G compiler.

9. Reliability

Good and getting better. (All problems which have been brought to my attention have been remedied.)

10. Method of Development

The compiler was hand coded. (Some routines were borrowed from the translator-writing system SYNTICS.)

The project was begun in the summer of 1972 and is still continuing. I have spent a total of 60 man-months on the project but was also teaching for 40 of those months, and have been distributing the compiler (copying and mailing tapes, etc.) for the last 20 months.

This is my first production compiler, but I now have five years experience.

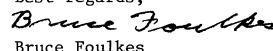
11. Subprograms

The compiler produces OS-compatible object modules and uses standard IBM linkage and parameter lists in calls of external routines (Fortran, etc.).

Separate compilation is not yet supported.

If people are interested in the compiler, they can write to me at the above address and I will send them a copy of my user manual, a description of the distribution tape contents, and a Software Release Agreement and order form.

Best regards,



Bruce Foulkes

compiler. Release 2.5 incorporates all improvements that were made during the lifetime of Release 1. In addition, the irreducible overhead to compile a trivial program has been reduced, by simulating the sequential access file I/O used for inter-pass communication with main-storage files. The current minimum storage requirement to run release 2.5 is slightly less than 150 Kbytes. Further reduction in the minimum partition size is a goal of future updates.

Release 2.0 implements Standard Pascal, including external, nonstandard files (implemented using the QSAM access method), with the following enhancements:

1. Module definition and linkage A program module, consisting of global types and data, functions and procedures, may be given the attribute external in the program heading. This informs the compiler that it is to be regarded as an ancillary compilation module, and the main program body is to be ignored. A procedure or function that is to be externally callable is given the attribute entry in its heading. External references are declared by giving a procedure or function declaration in which the body is replaced by the keyword external. A main program can also contain entry procedures and external references. Global data that are to be shared among modules are declared in a section called extvar. Otherwise, global data are private to the program module in which they are declared.

Each compilation prepares an output file called a Pascal Object Module, which contains not only code and data, but also a symbol table describing the names, types, and storage given to its entry points and external references. A new system module called the Pascal Linker accepts Pascal Object Modules, links external references, and checks them for type-correctness, outputting an IBM object module. The IBM linking loader (or the link editor) is subsequently used to perform a final relocation and to link to the standard execution environment, including any callable Fortran library routines that may have been specified.

We believe this new facility will prove to be a powerful tool for modular design and testing of large programs. The post-mortem-dump diagnostic facility is preserved throughout the linkage process, if it has been requested as a compiler option.

2. Parametric constants (to be implemented in Release 2.1)

In a const declaration, a constant may be given a declared type, by a syntax extension. A constant whose type is one of the predeclared arithmetic of scalar types can have its value part declared as external, meaning that the actual value is unknown at compile time. The value must be supplied at linkage time, either from a constant of the same name (and type) declared in a main program module, or as a named parameter to the program linker.

The principal reason for this extension is that it allows the declaration of subrange types whose bounds can be altered without recompilation; thus array and set types can have their sizes adjusted to the demands of an application merely by re-linking a program. This scheme is admittedly a compromise between the desire to allow Algol 60 dynamic arrays and the desire not to perturb the type formation rules of Pascal. Only the experience of users will tell whether or

not it is adequate. For more details on the scheme see [1].

3. Storage management (to be implemented in Release 2.1)

Effective management of heap storage has been a problem in Pascal, exacerbated by the type violation allowed with variant records. Conventional scan-mark garbage collection is frustrated by the lack of certainty as to whether or not a variant field contains a pointer. We have, therefore, implemented a storage management scheme using two-level indirection (see [2] for a general discussion) in which a pointer actually is an index into a storage descriptor table, invisible to the user. Descriptors contain a heap address, a hold count, and a pointer to a storage template. This has the advantage that the heap itself contains no absolute address constants, and therefore, most storage reclamation can be done simply by compactification. References to dangling pointers are immediately detectable, and it is therefore practical to implement the standard procedure Dispose, allowing a programmer to control the release of dynamically allocated variables.

Obviously, it is possible for inaccessible, but cyclic data structures to exist undetected and to congest the descriptor table. When this condition occurs, the inaccessible descriptor storage can be reclaimed by scan-mark garbage collection, within the array of fixed-length descriptors.

4. String processing (to be implemented in Release 2.1)

A new, built-in type, String, has been added. Values of this type are variable-length strings over type Char. Operators are all realized by means of predeclared functions; no new operator symbols have been added. These functions include:

```
function Catenate (A, B : String) : String;
function Length (A : String) : Integer;
function Byte (A : String; B : Integer) : Char;
function Substr (A : String, B, C : Integer) : String;
```

String values may be compared for equality or inequality (using lexical order) by applying the standard Pascal comparison operators. The symbol '' denotes the constant, empty string. Procedures Write and Writeln will accept arguments of type String, as will procedures Read and Readln. Strings to be input must be quoted by apostrophes.

Coercions are allowed for assignment from any expression of types array [] of Char to a variable of type String. Coercions in the reverse direction are not allowed.

String processing is provided by a Pascal-linkable library module, which need not be included in the runtime support package if it is not needed. The implementation is machine-dependent and very efficient. The maximum string length allowed is 256 bytes.

5. A predeclared type Alfa has been defined to be:

```
Alfa : packed array [1..80] of Char
```

This type is principally intended to permit efficient reading of fixed-length, 80 byte records. If the condition Eoln becomes true during an attempt to read a variable of type Alfa, and prior to reading the 80th byte, a read error occurs.

6. Constructors for typed constants

Typed, structured constants may now be declared. The constructor syntax uses parenthesization to reflect the internal structure of the declared type, in order to allow localization of possible errors in defining a constructor. This facility allows a programmer to define constant tables, or it may be used to simplify the initialization of structured variables. A constructor may be created for any types except file or pointer. For structured types, each literal value given as a component is checked for type compatibility with the declared type of the corresponding field. Parametric constants are not allowed as components of a constructor, nor can the type of a constant depend upon any parametric constant.

7. Default clause on case statements

A syntax extension now allows the end of a case statement to be optionally replaced by the clause otherwise <statement>. Also, ranges of constants are allowed as case labels. The cardinality of a label range is not limited by any implementation restriction, but only by the cardinality of its base type. The implementation uses a combination of branch trees and jump tables; a more detailed discussion of this technique can be found in [1].

8. Diagnostic aids

The two principal run-time diagnostic aids provided by the Pascal/360 compiler are a symbolic, post-mortem dump of active variables, and an execution-count profile printed on a formatted source listing. These diagnostics in Release 1 have proven to be valuable tools for analysis and debugging of Pascal programs, and for compiler debugging as well. The post-mortem dump has been extended to work with separately compiled modules, although the execution profile will only apply to the main program module.

The efficiency of run-time subrange checks has been improved slightly from that of Release 1, and a standard procedure Halt has been added, to allow a post-mortem dump to be induced under program control.

9. Cross reference listing

A new compiler option provides a cross-reference listing of all identifiers and the source program line numbers where they occur. The listing also gives the name of the block in which each identifier is declared. This option is useful in maintenance and documentation of large programs.

Future development efforts will be directed towards

- a) eliminating any bugs that may have been added along with the new compiler features;
- b) an optional code optimizer;
- c) reducing the main storage requirement of the compilers;
- d) further enhancement of the diagnostics.

References

- [1] Kiebertz, R. B., Two Generalizations of the Programming Language Pascal, Tech. Rept. No. 66, Dept. of Computer Science, SUNY at Stony Brook, April 1977.
- [2] Bobrow, D. G., An efficient, incremental, automatic garbage collector, CACM 19, No. 9, p.522, Sept. 1976.

IBM 360, 370 (Grenoble)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. J. P. Fauche, Departement Informatique, IREP, Boite Postale 47, F-38040 Grenoble Cedex, France.
2. MACHINE. IBM 360/67, 370/148.
3. SYSTEM CONFIGURATION. Runs under OS/MVT (360/67) and VS/MFT (370/148). Requires 220K for self-compilation.
4. DISTRIBUTION. Distribution is via 9 track, 800 bpi magnetic tape.
5. DOCUMENTATION. The implementation is described in a supplement to the User Manual.
6. MAINTENANCE. (* no information *)
7. STANDARD. Deviations are described in the documentation.
8. MEASUREMENTS. The standard compiler (6000 lines of Pascal) compiles in 105 CPU seconds; an enhanced compiler compiles in 84 seconds.
9. RELIABILITY. (* no information *)
10. DEVELOPMENT METHOD. (* no information *)
11. LIBRARY SUPPORT. Assembler procedures are supported.

IBM 360, 370 (Socorro, New Mexico)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Implementors: Jan V. Garwick, Paul Merillat, and Robert Knight, Computer Center, New Mexico Tech, Socorro, New Mexico 87801. Distributor: Tom Nartker, Computer Science Department, New Mexico Tech, Socorro, New Mexico 87801 (505/835-5126). Direct non-distribution questions to Robert Knight.
2. MACHINE. IBM 360, 370 series.
3. SYSTEM CONFIGURATION. OS operating system.
4. DISTRIBUTION. Released January, 1977.
5. DOCUMENTATION. (* no information *)
6. MAINTENANCE. (* no information *)
7. STANDARD. The following are not supported: gotos and labels; unpacked arrays; and sets of characters.
8. MEASUREMENTS. (* no information *)
9. RELIABILITY. Results of one month of testing were good (76/9/20).
10. DEVELOPMENT METHOD. Designed by Jan Garwick and implemented in PL360 using GPM.
11. LIBRARY SUPPORT. (* no information *)

IBM 360, 370 (Stanford)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Sassan Hazeghi, Computation Research Group, SLAC, P.O.Box 4349, Stanford, CA 94305.
2. MACHINE. IBM 360, 370.
3. SYSTEM CONFIGURATION. (* no information *)
4. DISTRIBUTION. The entire system is available to the public (as is).
5. DOCUMENTATION. (* no information *)
6. MAINTENANCE. No maintenance is promised.
7. STANDARD. Implements the Pascal-P2 (May, 1974) subset, with a few minor extensions.
8. MEASUREMENTS.

	Compiler	Post-processor
Source lines (Pascal)	4000	2500
Bytes, including I/O.	76K	52K
Time to process compiler (370/168, 16K cache)	10 sec.	5 sec.
Source lines per second	400	800

The system self-compiles in 130K bytes, 24K of which is returned to the operating system for I/O buffers.

9. RELIABILITY. (* no information *)
10. DEVELOPMENT METHOD. Developed from Pascal-P2. P-code was first translated to assembly code by macros; a P-code translator was then written in Pascal. The P-translator can produce either assembler code or a standard OS object module.
11. LIBRARY SUPPORT. (* no information *)

IBM 360, 370 (Oslo)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Ivar Laberg, Computer Department, University Hospital Oslo, Rikshospitalet, Oslo 1, Norway (471 20 10 50).
2. MACHINE. IBM 370/125.
3. SYSTEM CONFIGURATION. DOS/VS operating system.
4. DISTRIBUTION. (* no information *)
5. DOCUMENTATION. (* no information *)
6. MAINTENANCE. (* no information *)
7. STANDARD. A number of extensions are being considered, including: interface to all secondary storage access-methods; external procedures written in other languages; and "external records" (functionally equivalent to "named common" in Fortran).
8. MEASUREMENTS. (* no information *)
9. RELIABILITY. (* no information *)

10. DEVELOPMENT METHOD. Based on Pascal-P.

11. LIBRARY SUPPORT. (* no information *)

IBM 360,370 (Vancouver)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Barry W. Pollack and Robert A. Fraley, Department of Computer Science, University of British Columbia, Vancouver, British Columbia, Canada V6T 1W5 (604/228-6794 or 604/228-3061).

2. MACHINE. IBM 370/168.

3. SYSTEM CONFIGURATION. The current version runs under the MTS operating system. The monitor may be modified with minimal effort to run under VS, OS, etc. Standard OS object modules are generated. The translator requires about 320K bytes of store. Division of the compiler into overlays for non-VM systems would be possible.

4. DISTRIBUTION. The current version is available for distribution now, via 9 track magnetic tape. Costs will be limited to postage (and tape purchase, if one is not supplied).

5. DOCUMENTATION. A User's Guide describes completely the implementation's departures from the Jensen and Wirth Pascal User Manual and Report.

6. MAINTENANCE. No policy has been decided. It is anticipated that periodic upgrades and modifications will be distributed at least once a year. Reported bugs will be corrected as quickly as possible with notification to users.

7. STANDARD. The compiler provides numerous extensions and a few restrictions. A compiler option issues error messages when non-standard features are used. A complete description is contained within the documentation provided. A summary of the differences follows.

Extensions:

Strings are padded on the right with blanks.

The is a case default label: "<>".

Optional ":" allowed before else.

"(...)" may be used instead of "[...]".

The character col has been retained.

Packed is ignored.

Input of character strings using read is allowed.

Support of EBCDIC characters "&", "|", and (logical not sign). (* Sorry, we use ASCII at PUG News. *)

Use "..." for comments.

Value section exists for variable initialization.

Hexadecimal integers are supported.

FORTTRAN subroutines may be called. A return code is available in the pre-declared variable rcode.

Direct access files are supported.

Additional built-in functions include: min, max, substr (using constant length), position (direct access files), I/O interface functions and extensions to reset and rewrite, and insert for data packing.

Restrictions:

Sets are limited to 32 elements (0..31).

Program heading is not used.

Files may not be components of other structures.

Set constructors may not include <expression>..<expression>.

Input@ is initially col instead of the first character of the file. This is transparent when read is used.

Projected extensions:

McCarthy if.

Or and and lower precedence than relations.

"Usual" precedence used throughout.

Sets over the range 0..255.

Better control of input and output formats.

8. MEASUREMENTS. The compiler is written in Pascal and is modeled after the CDC 6400 implementation, but it has been extensively modified and improved. The translator consists of approximately 8000 lines of Pascal code. The run-time library consists of approximately 500 lines of Pascal code. The monitor (which contains the interface to the operating system) consists of approximately 2000 lines of IBM Assembler G code. The translator speed has not been determined, but it seems faster than our Algol-W compiler. The code produced has been timed against Algol-W code and is almost uniformly 10-15% better. This is especially true of any program using a large number of procedure calls. The compiler compiles itself in less than 60 seconds of 370/168 processor time. The compiler requires 320K bytes of core.

9. RELIABILITY. To date has been excellent. A student version of the translator has been running since September, 1976, with only one detected compiler error. The main system version has been in operation since December, 1975. All problems which have been encountered to date have been corrected.

10. DEVELOPMENT METHOD. The original translator was developed by Wirth and several graduate students at Stanford University as a partial re-write of the CDC 6400 version in 1972. The current translator and monitor have been extensively modified, a run-time library has been implemented, and a post-mortem symbolic dump package has been developed. The translator has been under continuous development at UBC since December, 1975, by two faculty members and one (* anonymous? *) graduate student.

11. LIBRARY SUPPORT. Fortran routines can be called. The compiler generates standard OS object modules.

IBM 1130

We have heard of two possible implementations, by:

(1) H. Sandmayr, Neu-Technikum, CH-9470 Buchs, Switzerland (085/6 45 24).

(2) Fred Powell, Innovative Management Systems, 865 Middlebrook Av., Staunton, Virginia (703/885-4950). (Fred was formerly at Mary Baldwin College.) "Little has been done so far," according to Fred (76/12/10).

ICL 1900 (Belfast) - MK2.

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Jim Welsh, Colum Quinn, and Kathleen McShane, Department of Computer Science, Queens University, Belfast BT7 1NN, Northern Ireland, U.K. (* No phone number provided. *) Enhancements by David Watts and Bill Findlay, Computer Science Department, University of Glasgow, Glasgow G12 8QQ, Scotland, U.K. (* No phone number provided. *)

2. MACHINE. ICL 1900

3. SYSTEM CONFIGURATION. Has been installed under George 3, George 4, Executive, MAXIMOP, and COOP operating systems. Requires 32K.

4. DISTRIBUTION. (* no information *)

5. DOCUMENTATION. A clearly written machine retrievable Supplement to the Revised Report, dated 77/2/23.

6. MAINTENANCE. (* no information *)

7. STANDARD. Primarily implements the Revised Report; exceptions include (a) files not allowed as components of structured types, and (b) non-discriminated variant records are not allowed. A six bit character set is used. Sets may have at most 48 elements. A value initialization part is implemented.

8. MEASUREMENTS. Compares favorably to Fortran, requiring about 32K to compile. Generated code is better than that produced by the old 1900 Pascal compiler. (* Compilation speed not reported. *)

9. RELIABILITY. Reported to be good. The compiler is in use at about 50 sites.

10. DEVELOPMENT METHOD. This compiler resulted from a complete rewrite of the old ICL 1900 compiler. The new compiler is designed for portability, with a clean separation between semantic analysis and code generation.

11. LIBRARY SUPPORT. Allows access to Fortran routines.

ICL 2970, 2980 (London)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. John Reynolds and Jules Zell, Department of Computing and Control, Imperial College, London SW7, U.K. (* No phone number provided. *)

2. MACHINE. ICL 2970, 2980 series.

3. SYSTEM CONFIGURATION. (* no information *)

4. DISTRIBUTION. Contact David Joslin, Sussex University Computer Centre, Brighton, Sussex, U.K.

5. DOCUMENTATION. (* no information *)

6. MAINTENANCE. (* no information *)

7. STANDARD. Presumably similar to the ICL 1900 MK2 compiler.

8. MEASUREMENTS. Code generated is fairly compact, the compiler itself producing 80000 bytes. This is better than the 2900 standard compilers. The (CDC) Pascal 6000 compiler compiles the 2900 compiler on a CDC 6400 in 82 seconds. The ICL compiler self-compiles on the 6400 in 100 secs. Running on a 2900, the 2900 compiler self-compiles in 360 seconds. John Reynolds tells us, "I've determined that almost all time required for a compilation on the 2900 is just burnt up by the system and that hardly any time at all goes in the actual act of code generation." (77/7/8) (* Execution speed of generated code not reported. *)

9. RELIABILITY. The compiler has been extensively tested and seems to work fairly well. (* Date of first release and number of sites using system not reported. *)

10. DEVELOPMENT METHOD. Based on the ICL 1900 MK2 compiler, with code generators rewritten. Poor performance of the ICL 2970 system led to development on a Control Data 7600 using Zurich's Pascal-6000.

11. LIBRARY SUPPORT. (* no information *)

Intel 8080 (INSITE)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Implemented by Thomas A. Rolander, 1012 Smith Ave., Campbell, CA 95008 (408/378-5785). Distributed by INSITE, Intel User's Library, Microcomputer Division, 3065 Bowers Ave., Santa Clara, CA 95051 (408/246-7501 x2948).

2. MACHINE. Intel 8080A using the Intel Intellec Microcomputer Development System.

3. SYSTEM CONFIGURATION. Operating system: Intel MDS ISIS-II. Hardware: 64K bytes of RAM and dual floppy disks.

4. DISTRIBUTION. The software is distributed on two soft-sectored diskettes, and includes the binaries and sources.

5. DOCUMENTATION. Consists of a short User's Guide, syntax graphs, and the source code for the virtual machine and the compiler.

6. MAINTENANCE. Bug reports will be accepted.

7. STANDARD. Implements Brinch Hansen's Sequential Pascal, except for floating point (which is under development - 77/2/22).

8. MEASUREMENTS. The virtual machine interpreter is 1300 lines of code (PL/M-80) and 10K bytes. Compilation speed is 30 lines/minute. (* Execution speed and size of generated code not reported. *)

9. RELIABILITY. Will self-compile and has been used successfully by students. (* Number of sites using system not reported. *)

10. DEVELOPMENT METHOD. An interpreter (PAS80) was written in PL/M-80, and emulates a 16-bit machine. The implementation required about "2 man-months-of-evenings" and was accomplished in the implementor's spare time. The implementor was familiar with the process of implementing the virtual machine. "Credit for the ease of implementation is due to Per Brinch Hansen who developed the virtual machine."

11. LIBRARY SUPPORT. (* no information *)

Intel 8080 (Minneapolis)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Peter Zechmeister, University Computer Center: 227 Exp Eng, University of Minnesota, Minneapolis, MN 55455 (612/373-4181).

2. MACHINE. Intel 8080.

3. SYSTEM CONFIGURATION. An operating system is included with the implementation. The minimal hardware required is an I/O device (TTY) and about 16K bytes for the compiler.

4. DISTRIBUTION. Has not been determined.

5. DOCUMENTATION. In progress.

6. MAINTENANCE. Under development.

7. STANDARD. The implementation is called Tiny Pascal (TP). It does not provide a number of standard features due to size constraints.

8. MEASUREMENTS. The bootstrap cross-compiler runs at 2400 lines/minute on a CDC 6400. The TP compiler itself loads in about 14K.

9. RELIABILITY. The reliability of the compiler is excellent. (* Number of sites using system not reported. *)

10. DEVELOPMENT METHOD. Based on the PLO compiler by Niklaus Wirth. Modifications were made to implement "variable types, Pascal statements, code generation, and register mapping." A cross-compiler running on a Control Data 6400 has been used to develop the Tiny Pascal (8080) compiler, which was not complete as of PUGN #8.

11. LIBRARY SUPPORT. None.

Intel 8080a (San Diego)

See DEC LSI-11 (San Diego), above.

Interdata 7/16

Two possibilities to check out:

Mike Ball (see Interdata 8/32 for address) has Concurrent and Sequential Pascal cross-compilers running on the U1100 generating code for the Interdata 7/16.

Rod Steel, Tektronix, MS 60-456, PO Box 500, Beaverton, OR 97707 (503/638-3411 x2516), reported a year ago that he might attempt to bring up Pascal on the 7/16. No news since then.

Interdata 7/32

See Kardios Duo 70, below.

Interdata 8/32 (San Diego)

(* See Mike's letter in the OPEN FORUM section *)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Mike Ball, Code 632, Naval Ocean Systems Center, San Diego, CA 92152. (* No phone number reported. *)

2. MACHINE. Interdata 8/32.

3. SYSTEM CONFIGURATION. (* no information *)

4. DISTRIBUTION. "It will not be available for distribution for at least several months." (7/7/15)

5. DOCUMENTATION. (* no information *)

6. MAINTENANCE. (* no information *)

7. STANDARD. Brinch Hansen's Sequential and Concurrent Pascal.

8. MEASUREMENTS. (* no information *)

9. RELIABILITY. (* no information *)

10. DEVELOPMENT METHOD. (* no information *)

11. LIBRARY SUPPORT. (* no information *)

Interdata 8/32 (Parkville, Australia)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Guy Ward, Department of Computer Science, University of Melbourne, Parkville, Victoria 3052, Australia (345 1844).

2. MACHINE. Interdata 8/32.

3. SYSTEM CONFIGURATION. (* no information *)

4. DISTRIBUTION. (* no information *)

5. DOCUMENTATION. (* no information *)

6. MAINTENANCE. (* no information *)

7. STANDARD. (* no information *)

8. MEASUREMENTS. (* no information *)

9. RELIABILITY. (* no information *)

10. DEVELOPMENT METHOD. Using Pascal-J (University of Colorado). Stage2 translates Janus into CAL (assembly).

11. LIBRARY SUPPORT. (* no information *)

Interdata 8/32 (Kansas)



Department of Computer Science
Manhattan, Kansas 66506
Phone: 913 532-6330

August 11, 1977

Dear Mr. Mickel:

As reported by Mike Ball in the Pascal Newsletter #7, we have transported the Brinch Hansen Concurrent Pascal system from the PDP 11/45 to the Interdata 8/32. This implementation in its present form uses an interpreter for a slightly modified version of the abstract code as distributed by Brinch Hansen. I am enclosing for your information a copy of the Implementation manual for the system and the implementation checklist as requested for the Implementation notes section of the Pascal newsletter.

Sincerely,

David Neal
Research Assistant

DCN:tlb
Enclosure (1)

1. Implementors:

David Neal, Gary Anderson, Jim Ratliff, and Virgil Wallentine.
Department of Computer Science
Kansas State University
Manhattan, Kansas 66506

Distributors:

Interchange (Interdata Users Group)
Interdata, Inc.
Oceanport, New Jersey 07757

2. Hardware:

Interdata 7/32 or 8/32.

3. Operating System:

OS/32-MT, minimum partition size 72.75 K, disk storage required,
floating point support necessary.

4. Method of distribution:

Nine track tape -- details available through Interchange.

5. Documentation:

KSU Implementation of Concurrent Pascal -- A reference Manual, KSU
Technical Report CS 76-16 will be provided with the implementation
package. It may be provided in machine retrievable form. The
Brinch Hansen SOLO manuals are not provided with the implementation.
The availability of these references is a necessity.

6. Maintenance Policy:

None

7. Fully implements Concurrent Pascal and Sequential Pascal (SPASCAL)
a subset of Standard Pascal.8. Sequential and Concurrent Pascal programs are executed by a code
interpreter written in Interdata CAL assembler language. This
interpreter as well as the Concurrent Pascal Kernel are provided in
source and object. The system consists of about 5000 source lines
and requires a library segment of 7.50 K for execution. Pascal
source is translated into code by the Hartman Compilers which are
written in Sequential Pascal (SPASCAL). The source and object of
these compilers are also contained in the package. Microcode
routine for virtual instruction decode are included for the 8/32.

9. Reliability:

Excellent -- all errors detected at KSU have been traced to hardware.

10. Method of development:

Transported from the Brinch Hansen PDP 11/45 implementation. The
system was moved with an approximate outlay of 4 person-months of
experienced graduate student effort.

11. Sequential Pascal programs may call one another in arbitrary,
recursive fashion using the interfaces of the SOLO operating
system (which is written in Concurrent Pascal). No provision
is made for FORTRAN or any other language. The utility programs
of the SOLO operating system include the Sequential and Concurrent
Pascal Compilers, a text editor similar to Interdata's OS-Edit, and
the source code configurer program mentioned by Mike Ball (Pascal
Newsletter #7 p. 29). All programs are maintained by the SOLO
file system and appears to OS/32-MT as a single contiguous file.-----
ITEL AS/4, AS/5

See IBM 360, 370, above.

Kardios Duo 70

See IBM 360, 370, above.

The Kardios Duo 70 consists of an Interdata 7/32 modified by Kardios Systems Corp.,
3820 Courtleigh Dr., Randallstown, MA 21133 (301/542-6826). The machine includes firmware
which emulates both Interdata and IBM 360, 370 systems. The system is designed to
concurrently execute both Interdata and IBM software. According to Kardios, most software
such as the IBM Pascal implementations will run on the Duo 70 with little or no
modifications. The changes most often required are: use Interdata CSS instead of IBM JCL;
change IBM file access calls to Interdata access calls (this is only necessary in the few
cases where the IBM file access methods are not supported by Interdata). The Duo 70 will
execute 360, 370 object modules produced by a compiler with no changes at all. Kardios
reports that their customers have reported very little trouble in modifying 360, 370
software to run on this system.

Mitsubishi MELCOM 7700.

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Masato Takeichi, formerly at Dept. of Math.
Engineering and Instr. Physics, University of Tokyo, Bunkyo-ku, Tokyo 113, Japan. Present
address: Department of Computer Science, University of Electro-Communications,
1-5-1 Chofugaoka, Chofu-shi Tokyo 182, Japan.

2. MACHINE. MELCOM 7700, 256K bytes.

3. SYSTEM CONFIGURATION. BPM. (* Minimum hardware required not reported. *)

4. DISTRIBUTION. (* no information provided. *)

5. DOCUMENTATION. See "Pascal Implementation and Experience" by Masato Takeichi, Journal
of the Faculty of Engineering, University of Tokyo 34:1, pp 129-136.

6. MAINTENANCE. (* no information provided. *)

7. STANDARD. Comparable to CII IRIS 80 implementation by Mancel and Thibault.

8. MEASUREMENTS. Self compiles in 150 sec. and 150 Kbytes (108K for code, 10K for monitor, 32K for data). Execution times, relative to Fortran, are given in the following table.

	Extended Fortran IV	Pascal
Matrix multiply	1	1.90
Sort	1	1.75
Additive partition	1	0.48
Character count	1	0.34

9. RELIABILITY. Was first released in April, 1976, with the author using for several months before that. Several compiler errors have been corrected. (* Number of sites not reported *)

10. DEVELOPMENT METHOD. The compiler is based on the CII IRIS 80 compiler by Mancel and Thibault, with modified code generation. The monitor and library procedures were rewritten to interface with BPM.

11. LIBRARY SUPPORT. (* no information provided. *)

MITS Altair 680b

(* See implementation notes for Motorola 6800. *)

MOS Technology 6502 (San Diego)

See DEC LSI-11 (San Diego), above.

Motorola 6800 (San Diego)

See DEC LSI-11 (San Diego), above.

Motorola 6800 (St. Paul)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Mark D. Rustad, 585 Harriet Ave., Apt #213, St. Paul, MN 55112 (612/483-0589).

2. MACHINE. Designed for the MITS Altair 680b, based on a Motorola 6800.

3. SYSTEM CONFIGURATION. Requires 32K bytes and a TTY. No disk needed.

4. DISTRIBUTION. (* no information *)

5. DOCUMENTATION. (* no information *)

6. MAINTENANCE. (* no information *)

7. STANDARD. The following are not supported: files (except TTY input and output), and get, put, reset, rewrite; with and goto; sin, cos, arctan, exp, ln, sqrt, pack, and unpack.

8. MEASUREMENTS. Compiler code occupies 24K bytes, the interpreter requires 3K bytes.

9. RELIABILITY. Seems to be excellent. Not yet released.

10. DEVELOPMENT METHOD. Based on Pascal-P2, cross-compiled first from a Univac 1100 (using San Diego Pascal), and later from a CDC 6400. As of 7/6/11/4, about 2 man-months had been invested. The compiler source is about 2200 lines. The cross-compiler has been designed to be independent of the host-machine's character set. The interpreter could be implemented on other 8-bit machines with minimal work.

11. LIBRARY SUPPORT. (* no information *)

Nanodata QM-1

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Dennis Heimbigner, TRW DSSG, Mail Station R3/1072, 1 Space Park, Redondo Beach, CA 90278 (213/535-0833).

2. MACHINE. Nanodata QM-1.

3. SYSTEM CONFIGURATION. 256K words nanostore; 8K words control store; 60K words main store; 9755 (55M byte) disk; TASK version 1.04.02 (or later); PROD version 2.04.01 (or later). Optional: Card Reader, Printer (very desirable).

4. DISTRIBUTION. "Release by TRW is currently under consideration. Inquiries are welcome." (77/3/17)

5. DOCUMENTATION. Brinch Hansen's SOLO manuals (not available from TRW); machine readable document describing the implementation and ways to modify it.

6. MAINTENANCE. (* no information *)

7. STANDARD. (* no information *)

8. MEASUREMENTS. Executes at about one-third the speed of the PDP11/45 (SOLO) system. (* Space requirements not reported. *)

9. RELIABILITY. (* no information *)

10. DEVELOPMENT METHOD. The Concurrent Pascal system kernel was programmed in micro-code in 6 months of part-time work. Half of that time was spent on I/O drivers.

11. LIBRARY SUPPORT. (* no information *)

NCR Century 200

Jack Laffe, 320 19th Ave. S., Minneapolis, MN 55454 (612/336-4946) tells us (77/08/30) that he is writing a Pascal compiler in Neat 3 for the Century 200.

Norsk Data NORD-10 (CERN)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. David L. Bates and Robert Caillian, PS/CCI Group, CERN, 1211 Geneva 23, Switzerland. (Tel. 41-98-11)
2. MACHINE. Norsk Data NORD-10.
3. SYSTEM CONFIGURATION. SINTRAN III operating system.
4. DISTRIBUTION. "Anyone is welcome to receive a copy of our system." (77/1/19)
5. DOCUMENTATION. (* no information *)
6. MAINTENANCE. (* no information *)
7. STANDARD. (* no information *)
8. MEASUREMENTS. It takes 15 minutes to compile the compiler. (* Space requirements not reported. *)
9. RELIABILITY. (* no information *)
10. DEVELOPMENT METHOD. From Pascal-P4. P-code is assembled and then interpreted by an assembly language program.
11. LIBRARY SUPPORT. (* no information *)

Norsk Data NORD-10 (Oslo)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Andora Fjeldsgaard, Petter Gjerull, Stein Gjessing, Jan Husemoen, Kefil Moen, and Terje Noodt. Computing Center, University of Oslo, Blindern, Oslo 3, Norway. (* No phone number provided. *)
2. MACHINE. Norsk Data NORD-10, using 2 64K memory banks.
3. SYSTEM CONFIGURATION. MOSS operating system.
4. DISTRIBUTION. (* no information *)
5. DOCUMENTATION. The implementation is described in "Rapport om implementering av Pascal pa NORD-10", University of Oslo, April 1976. A machine readable document describes changes and improvements to the implementation as they are made.
6. MAINTENANCE. It is expected that the system will be improved and changed frequently in the near future. Error reports are invited, and may be given to any member of the PASCAL group.
7. STANDARD. Files (except input, output, PRD, and PRR) and formal procedures are not implemented. Sets may have 64 elements; parameters and local variables (except arrays and records) may occupy at most 253 16-bit words in any procedure; strings may be at most 16 characters long.
8. MEASUREMENTS. (* no information *)
9. RELIABILITY. (* no information *)
10. DEVELOPMENT METHOD. Developed from Pascal-P.
11. LIBRARY SUPPORT. (* no information *)

Prime P-400

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Phillip H Enslow, School of Information and Computer Science, Georgia Tech., Atlanta, GA 30332 (404/894-3187).
2. MACHINE. Prime P-400.
3. SYSTEM CONFIGURATION. Virtual memory operating system.
4. DISTRIBUTION. (* no information *)
5. DOCUMENTATION. (* no information *)
6. MAINTENANCE. (* no information *)
7. STANDARD. (* no information *)
8. MEASUREMENTS. (* no information *)
9. RELIABILITY. (* no information *)
10. DEVELOPMENT METHOD. Bootstrapped from Pascal-P4 during 1976-77.
11. LIBRARY SUPPORT. (* no information *)

SEMS T1600

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Alain Tisserant, Departement Informatique de l'INPL, Ecole des Mines, Parc de Saurupt, 54042 Nancy Cedex, France. (Tel. (28) 51 42 32)
2. MACHINE. SEMS T1600 and SOLAR 16/05/40/65.
3. SYSTEM CONFIGURATION. BOS-D operating system. Hardware: MTS16; FHE or MHU disk; minimum 16K words of core memory.
4. DISTRIBUTION. Not yet available (77/2/2). Will be distributed by IRIA.
5. DOCUMENTATION. All available documentation is written in French. (* we don't know what is available. *)
6. MAINTENANCE. (* no information *)
7. STANDARD. "Fully implements standard Pascal; also compatible with IRIS 80 Pascal compiler." Extensions include: character strings; Loop-exit-end statement; I/O of sets and scalars; sets of any interval of integers.
8. MEASUREMENTS. (* no information *)
9. RELIABILITY. "Expected to be excellent!"
10. DEVELOPMENT METHOD. The compiler(s) is written in Pascal. A two pass scheme uses an adaptation of P-code as an intermediate language. The P-code was adapted for non-stack, 16-bit, based addressing and accumulator machines. The first pass can be parameterized, and the second pass can be rewritten to port the compiler to other machines. An automatic segmentation mechanism allows compilation and execution of large programs (such as the compiler) with small memory requirements.
11. LIBRARY SUPPORT. The implementation allows separate compilation, as well as insertion of ASM and Fortran routines.

Siemens 330.

KERNFORSCHUNGSZENTRUM KARLSRUHE

Institut für Datenverarbeitung in der Technik

Karlheinz Kapp

75 KARLSRUHE 1, 22.6.77
Postfach 3640 Jo
Fernsprecher: (07247) 821 3928
Durchwahl: (07247) 82
Fernschreiber: 7826755

Das Kernforschungszentrum wird betrieben von:
Gesellschaft für Kernforschung m. b. H., Karlsruhe, Weberstraße 5

Dear Andy:

Enclosed you find some details about our brand new Sequential-PASCAL-Machine for the SIEMENS 330 Process Control Computer, which has been implemented by me in 2.5 manmonths. The SIEMENS-Version runs under the Real-Time OS ORG330PP2 or ORG330K and is full compatible to our VARIAN-V75-PASCAL, which we received from VARIAN Munich and which was modified and extended by me in few weeks.

The language is identical to Sequential PASCAL by P. Brinch Hansen and Al Hartmann for the Concurrent PASCAL Machine on PDP11/45. So the Concurrent PASCAL Compiler, after few modifications, is now running on our SIEMENS 330 as a Cross-Compiler for the Concurrent PASCAL Machine.

We still have not so much experience with PASCAL-applications, but while implementing and modifying these two systems, we have found no Compiler-Errors at all. The Code- and Runtime-Efficiency is good compared with RT-Fortran and will allow RT-application in PASCAL.

As prerequisite for a good flexibility, the user himself is able to extend the PASCAL-system using the external procedure interface of the system.

Although the distribution and maintenance policy of my Company is not yet fixed, I enclose some information about our PASCAL-Systems.

I think that this information is interesting mainly European users, but our interest is now to receive complete PASCAL-Programs from other users, mainly a good source editor, intender and cross-referencer would be welcome. Additionally we are looking for PASCAL on a DECLSI11 or INTEL8080 Micro-Computer.

Sincerely yours

Karlheinz Kapp

Sequential PASCAL

1) Implementor/
Distributor

Phone

2) Machine

3) Op. System

Min. Config.

Core for Compiler

4) Distribution

5) Documentation

6) Maintenance

7) Standard Pascal:

8) Type of Program:

9) Code efficiency:

PASCAL-Machine
minimum core

VARIAN V75

VARIAN
Data Machines
D-8000 München
W-Germany

VARIAN V75

VORTEX II E/F

64K Memory
FPP/WCS
CR/LP/DISK/MT
TTY

20K

ask Varian

Manual in English

ask Varian

I/O-System is different
no Standardprocedures

Interpreter and I/O-System
written in Macro/Assembler
7-Pass- 8-Pass-
Compiler written in PASCAL

3 W PASCAL-Code/sourceline or
1 W PASCAL-Code/9 signif. chars

3K

SIEMENS 330

K.-H. Kapp
Gesellschaft für
Kernforschung mbH
IDT
Postfach 3640
D-7500 Karlsruhe
W-Germany
(07247) 823928

SIEMENS 330

ORG330PP2/
ORG330K
SEGSYS

64K Memory
FPP/SIM.
CR/LP/DISK/MT
TTY

20-22.5K

No-label tape,
9 track, 800 BPI

Users Guide in
German

on request

1.8K

10) Speed Compiler compiles typical PASCAL-
source-text at a rate of 150 lines/min
or 500 significant chars/sec.

For more information about Sequential PASCAL read the publications
about Concurrent PASCAL.

Siemens 4004, 7000 (Munich)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Manfred Sommer, SIEMENS AG, Department D AP GE, Postbox 70 0078, D-8000 Munich, West Germany (089-722-61276).
2. MACHINE. Siemens 4004 and 7000 series. Also RCA Spectra 70 (VMOS).
3. SYSTEM CONFIGURATION. BS2000 operating system.
4. DISTRIBUTION. Contact the implementor.
5. DOCUMENTATION. A User's Manual (German) is available.
6. MAINTENANCE. (* no information *)
7. STANDARD. Appears to conform fully with the Pascal User Manual and Report. Character set is EBCDIC. Sets may have 256 elements (allowing set of char).
8. MEASUREMENTS. "Code produced seems to be much faster than the code produced by the standard Fortran compiler." Compilation speed is 40 lines per second on a 4004/151 and 100 lines per second on a 7000/7.755 (roughly equivalent in power to an IBM 370/155 or CDC 6400). In a dozen or so benchmark programs times were comparable with CDC-6400 Pascal.
9. RELIABILITY. Over 18 sites using this version.
10. DEVELOPMENT METHOD. Based on Pascal-P4.
11. LIBRARY SUPPORT. Generated code may be put into a standard module library. Additional procedures are available for interfacing to the operating system.

Siemens 4004 (Darmstadt)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. H.-J. Hoffmann, Fachbereich Informatik, Techn. Hochschule, Steubenplatz 12, D-1600 Darmstadt, Germany. (* No phone number provided. *)
2. MACHINE. Siemens 4004/157.
3. SYSTEM CONFIGURATION. (* no information *)
4. DISTRIBUTION. (* no information *)
5. DOCUMENTATION. (* no information *)
6. MAINTENANCE. (* no information *)
7. STANDARD. (* no information *)
8. MEASUREMENTS. (* no information *)

9. RELIABILITY. (* no information *)

10. DEVELOPMENT METHOD. There are three compilers, all based on Pascal-P2: (a) a fully interpretive version; (b) a version where P-code is translated to assembly language; and (c) a version with assembly code emitters in the compiler.

11. LIBRARY SUPPORT. (* no information *)

SOLAR 16/05/40/65.

See SEMS T1600, above.

TELEFUNKEN TR-440

INSTITUT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN
Manfred Luckmann

Telex: tumue d 05-22854 · Institutseingang: Barer Straße 23, Ecke Gabelsbergerstraße

FERNRUF (089) 2105 - 8276

Institut für Informatik der Technischen Universität München
D-8000 München 2, Postfach 202420

MÜNCHEN, DEN August 8, 1977

Dear Mr. Mickel,

I send you here some information about our PASCAL implementation running on the Telefunken TR440:

The compiler was first implemented by
Hans Dieter Petersen
Universität des Saarlandes
Institut für Informatik I
Im Stadtwald
D-6600 Saarbrücken / Germany
and later extended by
Manfred Luckmann
Technische Universität
Institut für Informatik
Postfach 20 24 20
D-8000 München 2 / Germany

Machine: Telefunken TR440, operating system BS3.

Documentation: Supplement to the book: PASCAL User Manual and Report.

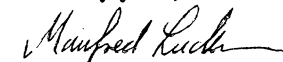
Maintenance: none.

Text-files only, no runtime checks.
External procedures allowed (written in PASCAL or in assembly language).

The compiler produces assembly language as intermediate code.

Size: ca. 40 K words, speed: ca. 10 lines / second.

Sincerely yours,



Manfred Luckmann

TERAK 8510 (San Diego)

(* See implementation notes for DEC LSI-11 (San Diego). *)

Texas Instruments TI-ASC.

Philip Bergstresser (see HERE AND THERE News section) phoned 77/05/26 to correct our information in PUGN #8. The PDL (Production Development Language) system TI implemented included a superset of Pascal and a library management system. This included software tools, a check for matching source and binary module interfaces, procedures recompiled independently with scope, complete reversible overlay process, cross reference and instrumentation code. Documentation is available from Bill Bixler at TRW Huntsville. The TI-ASC is a 650K 32 bit word machine with IBM 360-like floating point and vector and scalar hardware. It has 48 registers.

Texas Instruments 9900/4 (Vienna)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Implementors: H. Schauer, R. Nagler, and A. Szer, Institut fuer Informationssysteme, A-1040 Wein, Argentinierstrasse 8, Austria (Tel. 65 87 31/313). Distributors: ECO-Computer GesmbH&Co Kg (Fa. Langschwert), A-1010 Wein, Tuchlauben 14, Austria (Tel. 63 35 80).
2. MACHINE. TI 9900/4.
3. SYSTEM CONFIGURATION. No operating system; requires a mark-sense card reader and a line printer.
4. DISTRIBUTION. The system (hardware and software) is sold for 200.000. Austrian Schillings (about \$1500 U.S.).
5. DOCUMENTATION. Available in the form of a supplement to the Pascal Report. (* Not known if this is machine retrievable. *)
6. MAINTENANCE. "We intend to make more of it [the system] and we would like to accept bug reports."
7. STANDARD. The following are not supported: files; with and goto; formal procedures/functions. Sets of 64 characters are supported.
8. MEASUREMENTS. It is very slow compared with other systems. The system uses 12K ROM words and no external memory.
9. RELIABILITY. The reliability of the system is excellent. (* Date first released and number of sites using system not reported. *)
10. DEVELOPMENT METHOD. The system is written in Pascal and machine code (3000 source lines). It took 3 months to implement it on any microprocessor with no special experience of the implementors. The machine independent parts are bootstrapped by an existing Pascal compiler. The system is intended primarily to support programming education.
Basic concepts: the compiler translates the source program into an intermediate language represented as a tree, where each node represents one declaration and each leaf consists of the intermediate code of a PASCAL block in reversed Polish notation. This tree is the static information of the program. The compilation does not exceed the level of syntactic decomposition defined by the syntax diagrams in the PASCAL report. The interpreter performs all context-sensitive checking at execution time.

Machine independent parts of the system, i.e., the compiler and part of the interpreter are in the intermediate language. Only the nucleus of the interpreter is machine-dependent and therefore hand-coded. The input device is a mark-sense card reader accepting specially coded cards (reserved words have their own punch codes).

11. LIBRARY SUPPORT. (* no information *)

Univac 90/30.

See letter from C.C. Handley under Hewlett Packard HP-21MX.

Univac 90/70.

See Siemens 4004, 7000 series.

The U90/70 (formerly RCA Spectra 70) is very similar to the Siemens machines, both in hardware and software (VMOS vs BS2000).

Univac 1100 (San Diego)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Michael S. Ball, Code 632, Naval Ocean Systems Center, San Diego, CA 92152. (* No phone number provided. *)
2. MACHINE. Univac 1100 series.
3. SYSTEM CONFIGURATION. Exec-8 operating system; can be run in Demand mode.
4. DISTRIBUTION. As a member of USE, you may request a copy from Mike by sending a mag tape and noting any restrictions on its format.
5. DOCUMENTATION. A 29 page machine readable supplement to the Pascal User Manual and Report entitled "Pascal 1100" documents the implementation.
6. MAINTENANCE. (* no information *)
7. STANDARD. Restrictions: entry, processor, and univ are reserved words; standard procedures and functions may not be passed as actual parameters; file of file is not allowed. Sets may have at most 144 elements. The compiler accepts the full ASCII character set. A compiler option allows processing of Brinch Hansen Sequential Pascal programs.
8. MEASUREMENTS. The compiler compiles into 34K words and requires 6K words of library routines. Self-compilation requires about 15.5K for stack and heap.

Execution times for code compiled by Pascal was compared with code generated by the NUALG and ASCII FORTRAN processors. Fortran's local optimization was taken as a base value. The programs used for comparison were taken from Wirth's paper on the design of a Pascal compiler (Software - Practice and Experience, Vol. 1 (1971), pages 309-333). The results are summarized in the following table.

PART	Pascal		NUALG		FORTRAN		FORTRAN	
	no tests	no tests	no tests	no tests	local opt.	global opt.		
	(rel)	(rel)	(rel)	(rel)	(rel)	(time)	(rel)	
PART	0.62	0.61	0.85	0.84	1.00	1.00	15.10	0.99
PARTNP	1.18	1.06	3.29	3.17	0.94	1.00	0.93	0.85
SORT	1.37	1.12	1.83	1.49	1.00	1.00	18.01	0.59
MATMUL	1.82	1.43	2.05	1.70	1.00	1.00	10.26	0.39
COUNT	0.30	0.28	0.72	0.66	1.00	1.00	16.83	0.97

Univac 1100 (Madison)

**ACADEMIC COMPUTING CENTER
THE UNIVERSITY OF WISCONSIN - MADISON**

1210 WEST DAYTON STREET
MADISON, WISCONSIN 53706
608-262-1166

August 31, 1977

9. RELIABILITY. Quite good; it should approach excellent. The system has been in local use since about February, 1976, and it has been installed at 25 sites (11 university, 4 government, 10 industry).

10. DEVELOPMENT METHOD. The compiler was developed from Pascal-P2. (* Person-hours to develop system not reported. *)

11. LIBRARY SUPPORT. Generated code can be linked to subprograms written in Fortran or assembler.

Univac 1100 (Copenhagen)

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. J. Steensgaard-Madsen, DIKU (Datalogisk Institut, Kobenhavns Universitet), Sigurdsgade 41, DK-2200 Copenhagen N., Denmark. (* No phone number reported. *)

2. MACHINE. Univac 1100 series.

3. SYSTEM CONFIGURATION. Exec-8 operating system.

4. DISTRIBUTION. The charge for distribution from Datalogisk Institut is Dkr. 200. The distributors are attempting to maintain a distribution tree to reduce costs and hassles. Purchasers must sign a license agreement. The system is released only in relocatable form.

5. DOCUMENTATION. A 19-page machine readable supplement to the Pascal User Manual and Report is available. It is "A Pascal Compiler for the Univac 1100 machines", by J. Steensgaard-Madsen and Henrik Snog of DIKU.

6. MAINTENANCE. There is no promise of maintenance, but bug reports are required under the license agreement.

7. STANDARD. Deviations from the standard: Reset(f) on any textfile f will cause eof(f) = false and eoln(f) = true; Parameter types of formal procedures and functions must be specified. Restrictions: file of file is not allowed; standard procedures cannot be passed as actual parameters. Machine dependencies: Sets may have 72 elements, char is defined as (6-bit) Fieldata, ascii is an additional type; real is double precision always.

8. MEASUREMENTS. Compilation space is roughly 42K; speed is 100 lines per SUP second. Compiled programs run efficiently compared to other processors.

9. RELIABILITY. Excellent. (* Date first released and number of sites using system not reported. *)

10. DEVELOPMENT METHOD. Pascal-P with a team of 4 persons.

11. LIBRARY SUPPORT. External procedures may be written in Pascal or (ASCII) Fortran. Inclusion of assembler code is possible.

PASCAL Implementations
University Computer Center
227 Experimental Engineering Bldg.
University of Minnesota
Minneapolis, MN 55455

Dear Mr. Bonham:

Enclosed please find a description of our new diagnostic PASCAL compiler. The following will outline the development of the compiler (which isn't specifically dealt with in the description).

The UW-PASCAL compiler is the joint effort of five people (myself, Richard LeBlanc, Masahiro Honda, Steve Zeigler and Gary Holmes). It currently represents about 24-30 man months. Design was initiated during the summer of 1975 and the first test version was released to users in late 1976.

The compiler was designed from scratch, using a syntax-directed organization. The compiler uses a table-driven LALR(1) parser and an error corrector which is driven by the parsing tables. Initially the compiler was bootstrapped through a version of the P-compiler. Later, Mike Ball's N.O.S.C. compiler was used. At present, of course, we bootstrap through our own compiler. This has the added benefit of allowing our diagnostic checks to monitor our own compiler (at a very acceptable level of overhead). Indeed, the preponderance of compiler bugs are found in this manner. As a result, errors are automatically linked to the offending source statement in the compiler and readily fixed.

In case you are interested, I'm including a copy of our current User's guide (an updated version is being prepared). I'm also posting a copy of the compiler description to Andy Mickel for inclusion in the PUG Newsletter (or are you the person who handles that department?)

If you'd like further information, please feel free to write me.

Sincerely,



Charles N. Fischer

CNF:rb
enc.

Diagnostic PASCAL Compiler for Univac 1100 Series

The University of Wisconsin-Madison Academic Computing Center (MACC) has developed a diagnostic PASCAL compiler for the Univac 1100 series. The compiler is especially designed for research and instructional use. It emphasizes careful and complete diagnostic checking at both compile-time and run-time. Included are subscript and subrange checks, pointer validity checks, record variant and set range checks. When run-time errors are discovered a procedure walk back (with source program line numbers) as well as a symbolic dump of scalar variables are available. During compilation, a complete analysis of the syntactic and semantic correctness of the source program is performed. Automatic correction of minor syntax errors (e.g. missing semicolons or parentheses) is included.

The following provides detailed information about the compiler and its distribution policy.

- (1) The UW-PASCAL compiler is an ASCII processor which operates on any Univac 1100 series computer under EXEC-8. Two versions of the compiler are available. The first produces standard relocatable elements which may be collected to produce executable absolute elements. The second version operates in a "Load and Go" manner. PASCAL source programs are compiled directly into core and immediately executed. No collection step is used.
- (2) UW-PASCAL is written in PASCAL. Its source (including all versions) is about 14K lines. Compilation speed is about 4000 lines/min (on an 1110). The compiler requires about 70K words to operate (which is larger than most other Univac compilers). However overall compilation costs appear to be comparable to other Univac ASCII compilers. Code generated by the compiler is as good as, or better, than that generated by other ASCII compilers operating in a non-optimizing mode.

The Load and Go version is marginally smaller and faster than the standard compiler. For small programs, its cost (for compilation and execution) is about 60% of the standard compiler.

- (3) UW-PASCAL implements all of the Standard PASCAL language with the exception of GOTO's out of procedures. (STOP and ABORT statements are available as a partial alternative.)

In addition to the extensive diagnostic capabilities noted above, a number of other language extensions are available. These include:

- (a) A very powerful external compilation capability. Procedures which are compiled independently are always compiled in the

environment of their declaration. This allows complete compile-time checking of procedure interfaces as well as access to global variables. Linkage to externally defined assembly language procedures is also provided.

- (b) Conditional compilation facilities are provided. These include the optional compilation of code sequences enclosed by comment brackets ("conditional comments"). Further, code is not generated for unreachable program statements.
 - (c) The DISPOSE procedure is implemented. Run-time pointer checks ensure that disposed objects cannot subsequently be referenced. Heap objects may be grouped into logical units (termed "sub-pools") which may be freed in a single DISPOSE operation. This often significantly simplifies reclamation of heap storage.
- (4) UW-PASCAL has been in use since early in 1977. It has received rather heavy use and has been found to be very reliable (at present no extant bugs are known).

MACC currently maintains UW-PASCAL as a fully supported software product. The compiler, with a year of compiler support, is available for a fee of \$750. Both the source and absolute modules of both versions of the compiler as well as PASCAL support routines will be provided. Prompt distribution of corrections to compiler bugs as well as improvements to the compiler are also included. After this initial period, continuing support (including compiler improvements and extensions) is available for a fee of \$600 per year. A surcharge of \$100 will be added for users outside of the United States. A UW-PASCAL User Guide (included in machine-readable form with the compiler) which further details this compiler is available for a postage and handling fee of \$3 (\$5 foreign). Normally, the compiler is distributed on a 9-track 1600 BPI tape. However, other tape formats and densities may be available upon special request.

All inquiries should be directed to:

PASCAL Development Group
Attn: Dr. C. N. Fischer
MACC
1210 West Dayton Street
Madison, Wisconsin 53706

Information may also be obtained by contacting Dr. Charles N. Fischer at (608) 262-7870.

- (5) UW-PASCAL is an on-going research project at the University of Wisconsin. Future development plans include:
 - (a) Compiler tuning to reduce core requirement (to about 45K for

small programs) and to reduce overall compilation costs.

- (b) Inclusion of a varying-length string manipulation capability (similar to PL/1 varying length strings), with catenation and substrings operations, I/O, etc.
- (c) Addition of an interface to ASCII Fortran subprograms.

Varian (Sperry-Univac) V-70

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Distributed by the Varian Users Group (VOICE). Varian Data Machines (Sperry Univac), 2722 Michelson Drive, Irvine, CA 92713 (714/833-2400).
2. MACHINE. Varian V-70 series.
3. SYSTEM CONFIGURATION. Requires 32K+ memory, memory map, Vortex II operating system, extended instruction set, and 512 words of writable control store (WCS).
4. DISTRIBUTION. Available from Varian as VOICE #183C8.
5. DOCUMENTATION. A 120 page manual (non-machine retrievable) is available as part of distribution.
6. MAINTENANCE. (* no information *)
7. STANDARD. This is Brinch Hansen style Pascal. I/O is non-standard and oriented toward the Vortex-II I/O macros. Reference to files is by unit number. Additional restrictions: Strings must have an even number of characters. Goto's are not supported. Enumeration types cannot be defined within record declarations. Records may have at most 16 variants, and the ordinals of the variant labels (constants) must be in the subrange 0..15. Sets may have 128 elements. Uses Mark-Release.
8. MEASUREMENTS. Compiles over 1000 statements per minute. Compiler requires 17K decimal words of main memory.
9. RELIABILITY. Good. Distributed to over 10 sites.
10. DEVELOPMENT METHOD. Based on Brinch Hansen's Sequential Pascal.
11. LIBRARY SUPPORT. (* no information *)

Xerox Sigma 6, 9.

1. IMPLEMENTOR/DISTRIBUTOR/MAINTAINER. Pierre Desjardins, Universite de Montreal, Informatique, C.P. 6128, Montreal 101, Quebec, Canada (514/343-7662).
2. MACHINE. Xerox Sigma 6 and 9.
3. SYSTEM CONFIGURATION. (* no information *)

4. DISTRIBUTION. Distributed on magnetic tape (9 track, 800 bpi, structured much like a standard Xerox processor distribution tape - labelled in account :SYSGEN). Distribution cost is \$250, payable to Pierre Desjardins. The distribution includes documentation.

5. DOCUMENTATION. Program comments are in English. The following documents are distributed: "Program Description" (English) contains installation and maintenance information; "Manuel d'utilisation..." (French) is the user's manual; "METAPASC..." (French) provides macro-procedures to aid writing external procedures or functions in Meta-symbol; "Pascal 2 - Sigma: un systeme de programmation Pascal" (French) describes the functional structure of the compiler.

6. MAINTENANCE. Bug reports are welcome, and "update sheets could be sent." The distribution fee does not imply any responsibility or maintenance service on the part of the distributor, implementor, or the Universite de Montreal.

7. STANDARD. Corresponds to Pascal User Manual except: files may not be components of arrays, records or files; string constants may not occur in the const section; standard procedures and functions may not be passed as actual parameters. Sets may have at most 32 elements.

8. MEASUREMENTS. Compiler peak code size is 25K. Self-compilation takes 35K. Compilation rates are: 600 lines per minute (Sigma 6 - BPM/BTM) and 1200 lines per minute (Sigma 9 - CP-V). (* Size and execution speed of generated code not reported. *)

9. RELIABILITY. Good to excellent. (* Date first release and number of sites using system not reported. *)

10. DEVELOPMENT METHOD. The compiler source is 6200 lines of Pascal. It was produced by cross-compiling from a CDC Cyber 74. Effort was 18 person-months (without any prior knowledge of Sigma machines).

11. LIBRARY SUPPORT. The compiler produces a relocatable object module (in Xerox Standard Object Language) for each procedure and function. Provision is made for external procedures and functions written in Meta-symbol.

Xerox Sigma 7.

See also CII 10070.

The CII Iris-80 compiler (described above) has been transposed to the the Xerox Sigma 7 running under the BPM monitor by Masato Takeichi, formerly at Department of Mathematical Engineering, University of Tokyo, Bunkyo-ku, Tokyo 113, Japan. Present address: Department of Computer Science, University of Electro-Communications, 1-5-1 Chofugaoka, Chofu-shi Tokyo 182, Japan.

Another Sigma 7 project, apparently incomplete and inactive, was headed by Henry Bauer, III, Computer Science Department, University of Wyoming, Box 3682, Laramie, WY 82071 (307/766-5134).

Zilog Z-80.

Ken Bowles and co-workers, UCSD, have adapted the San Diego DEC LSI-11 implementation to run on the Zilog Z-80 running (at 2.5 MHz) about 70% as fast as the LSI-11. Release is expected by the end of 1977. See the DEC LSI-11 (San Diego) note, above.