May/June 1989

USUS NewsLetter

Copyright 1989 USUS, Inc.

All the News that Fits, We Print

William D. Smith, Editor

Volume 3
Number 5

From the Editor
by William D. Smith

Hey, guess what, I finished this NewsLetter on time. You should receive it about the first week in June. Last issue I said that I would cover my operating system interface unit this time. I didn't remember that I needed to cover my file I/O units prior to that (which are operating system and file system dependent). So next time I'll get to the operating system stuff. I would like to here from anyone who has some thoughts on what I've covered in the last few NewsLetters and any alternate ways of doing things.

Donation

USUS Member **Jean-Louis Thibaud** has donated two reference manuals which he wrote with computer science students in 1985 and 1986.

Unfortunately, these are in French! So we're not quite sure what makes up the package.

Text sources and modules are available on VAX/VMS backup tapes (and possibly DEC Rainbow 100 disks). From what we can read in the manuals, it appears the source is in fairly straight forward UCSD Pascal.

If anyone is fluent in French, please contact William Smith (CIS: 73007,173; (619) 941-4452). He has the manuals. Perhaps with your help we can find out what is involved.

Dr. Thibaud is with the Swiss Federal Institute of Technology in Lausanne, Switzerland. We're also asking him for more information about his donation.

HELP! EMERGENCY!
HELP! EMERGENCY!
HELP! EMERGENCY!

For the last six or more months USUS has been asking for some member to volunteer to be Special Interest Group Chairperson for IBM machines. No one has done so.

USUS now has a new member in London England who needs to order some material from the USUS Software Library. In addition, USUS needs to send him a "WELCOME DISK" with the SWLibe directories on it. The only problem is, USUS does not have access to an IBM PS/2 and no way to copy this data from 360K 5-1/2" 8 track disks which are the "standard" used in IBM/PC distribution!

This problem shows how much USUS needs the volunteer effort of its membership to support the needs of other USUS members. A few of us can't do it all alone folks! And we sure can't justify paying a commercial conversion firm to make these copies for us. We can pay the member who volunteers his/her time the cost of the materials they use. But also this volunteer can become a SWLibrary distributor and will then get paid for any future orders for IBM disks.

So is there someone out there reading this that can take on this SIG activity? It takes a bit of time, but gets you involved in USUS. And the more you give to USUS, the more you get out of it.

Please call me collect at (303) 526-0116 or write me at my home, 2193 Montane Drive East, Golden CO. 80401. This is urgent!

Hays Busch (USUS Administrator)

We Get Letters...

*USUS received a letter from **David Oshiro** (Nichandros Companies, 2901 Glascok Street, Oakland, CA 94601; (415) 533-6000; Fax: (415) 533-60001)*

"We are looking into the idea of hooking up an automatic switching unit to our Stride and connecting our Okidata ML 84 and a Diablo 230 to the switching unit. Does anyone know a way to send a string command to the switching unit that will select one or the other printer? Would like to do this while within programs such as Word7 and Flexware?"

If you have any interested in selling your software to the international market, the following brief excerpt from a letter USUS has received may be pertinent:

"Software USA, Inc. was formed to export computer software ... public domain, shareware and other high quality technical, medical and scientific programs ... to Austria, Switzerland, West Germany and other countries..."

USUS has made no investigation of this company and has no recommendation, either pro or con, on its offer. This notice is for information only.

If you are interested, USUS will provide information on how to contact the company.

Q & / | A? ...

Hays Busch sent the following:

What's wrong with this program? See if you can find the error. What will happen because of the error? If you give up, compile and run the program and see what happens. NOTE: The fact that this example will not terminate is NOT the error you are looking for.

```
program DoSomething;
var  Ans : char;

  procedure OutTerminal;
  begin
    writeln ('You''re in the ',
             '"terminal" proc.');
  end { OutTerminal };

  procedure OutPrinter;
  begin
    writeln ('You''re in the ',
             '"printer" proc.');
  end { OutPrinter };

  procedure DoIt;
  begin
    case Ans of
      'T' : OutTerminal;
      'P' : OutPrinter;
    end { cases };
  end { DoIt };

  procedure Decide;
  var  AnsOk : boolean;
  begin
    writeln ('Send output to T(erm',
             'inal, P(rinter ? ');

    repeat
      AnsOk := false;
      read (keyboard, Ans);
      if Ans in ['P', 'T',
                 'p', 't'] then
        AnsOk := true;
    until AnsOk;
  end { Decide };

begin { DoSomething }
  repeat
    Decide;
    DoIt;
  until false;
end { DoSomething }.
```

The Prez Sez

by Alex Kleider

Since the last NewsLetter deadline passed, USUS has had some negotiations with the Publisher of the Journal of Pascal, Ada and Modula-2 (JPAM) with the following result. All members of USUS are to receive a complimentary copy of the next issue with an invitation to subscribe at a reduced rate. I am hopeful that this will have already happened by the time this reaches you. Thus each member has the option of paying for the subscription or not. Another alternative which was discussed is to include a subscription as a membership benefit but this would not be feasible without a dues hike and was rejected by your Board of Directors for that reason. Let us (an Officer or a member of the Board) know what you think of this arrangement to guide further projects of this kind.

We are also inviting the readership of JPAM to use the MUSUS Forum on CompuServe to exchange messages since our interests overlap so much.

Harry Baya has been appointed the new MUSUS SysOp, replacing Arley Dealey to whom we all owe a debt of gratitude for his years of service.

Harry has also promised to investigate what needs to be done in terms of ByLaws changes etc. to open up USUS to a broader audience. It is the feeling of the Board and the Officers that we need to open ourselves up beyond the confines of the UCSD p-System (now the Power System) if our viability is to be maintained. There is a lot of interest amongst our members in Modula-2. We would like to include into our membership people interested in all implementations of Modula-2 and Pascal.

The subject of a USUS meeting is still of great concern and unfortunately none is coming forth with ideas and help to organize one.

As I mentioned in last month's column, USUS will only maintain its viability if there is more member participation. We are a volunteer organization and things only get done if YOU do them! Any and all comments, requests or criticisms will be gratefully received and thoughtfully pondered. If you want help with a problem, ask.

Until next time, good luck in your endeavors and Happy Computing!

Administrator Says

by Hays Busch

By now I suspect, most of you have looked at the yellow pages in the center of the March NewsLetter to check on your individual listing. (If you haven't, please do so now!) So, what did you find? Mistakes? Missing information? Or????

I found a few myself. For example, **Chris Acreman**'s name got "flipped" and is in alphabetic order with other last names beginning with "C". (Sorry Chris, you should have been the second name in the listing!) The source of this information is the Membership Data Base, so right now you can help USUS to get it more accurate by telling us what is missing or incorrect in the present listing.

If you are not listed at all, you probably checked the "do not list in USUS Rosters" option on a membership form some time in the past. Do you want that to continue? If not, let us know.

If you are a member of CompuServe, and your PPN is not shown you have not included it on a membership form (new or renewal). We would like very much to have this information for future listings.

If there is no computer name shown (or it is wrong) it's because you have not included it on a recent (new or renewal) membership form. This is VERY important information for USUS files. With it, we can sort the list to be sure you get the various SIG mailings. And we'll know what disk format you can read if we need to send you stuff on magnetic media. So, if nothing is shown, or what is shown is not the machine you use MOST with UCSD Pascal, let us hear from you. And, be specific. If you list APPLE, that's OK, but APPLE][e or APPLE ///, etc. is better. Same with other machines.

If there are errors in spelling etc., please let us know. We try to be accurate, but sometimes handwriting is hard to read and we do make mistakes. You are the only ones who can help us correct this kind of problem. Also if you have changed address and not told us, please do so.

Not trying to "brag", but if you look at my listing, it is an example of one that contains everything USUS would like to have in the data base for each member's Roster listing. (Note the "ZIP PLUS FIVE" code also. If all USUS member addresses had this, we might qualify for a special postal rate.)

I know I'm asking for extra work. But, I'll make the corrections if you send them in! Any other comments or suggestions you have about this Roster will be appreciated and will help us when it comes time to prepare the next one.

Not much else to say. Summer is on its way and the next two NewsLetters will be combined issues for July/Aug and Sept/Oct. If you send in some material, these combined issues can be big ones. How about it?

Treasurer's Report (Apr 1989)

by Robert E. Clark, Treasurer

Bank Balance	$6,318.20	03-31-89
Income - April 1989		
Dues:		(new/renew)
Student	0.00	0/0
General	70.00	0/0
Professional	0.00	0/0
Institutional	0.00	0/0
Other Income:		
Library fees	12.00	
Total Income:	$82.00	
Expenses - April 1989		
Administrator:		
CIS	8.09	
Telephone	12.95	
Photocopies	3.00	
Postage	15.59	

Other:		
Mail from La Jolla	6.20	
Library Distributor	14.00	
Printing, NL	475.00	
Mailing, NL	269.66	
Bank charge	1.00	
Total Expenses	$805.49	
Bank Balance	$5,594.71	04-30-89

Board of Directors Minutes (Apr. 18, 1989) by Samuel B. Bassett

MINUTES OF THE SPECIAL MEETING OF THE BOARD OF DIRECTORS OF USUS, INC., HELD IN ROOM 1 OF THE MUSUS FORUM TELECONFERENCING FACILITY ON THE COMPUSERVE INFORMATION SERVICE, BEGINNING AT 10:09 PM EST APRIL 18, 1989.

Present at the meeting were:

User ID	Name
76314,1364	Sam'l Bassett, Board Chaircritter
73447,2754	Henry Baumgarten, Board Member
72401,1417	Frank Lawyer, Board Member
75226,3643	A. Robert Spitzer, Board Member
76703,500	Eliakim Willner, Board Member
70260,306	A. Hays Busch, Administrator
72135,1667	Harry Baya, SysOp Designate
72747,3126	Robert Clark, Treasurer
71515,447	Alex Kleider, President
73007,173	William Smith, Assistant SysOp, NewsLetter Editor
72740,66	Howard Sweet, Secretary

The meeting came to order at 10:09 PM EST, with all Board members and all of the currently active officers present — something of a record!

Matters dealt with were:

SysOpery

Inasmuch as the current Wiz SysOp, Arley Dealey, has found himself in severe difficulties in his personal life, both he and the Board felt that it would be better for all concerned if another person were appointed Wiz SysOp. Harry Baya volunteered for the post, and Hays Busch contacted CompuServe to find out what our status was with them. According to the person who is now handling MUSUS' affairs, they are quite willing to work with us and try to make us successful — it turns out that there is even a little sentimental bias in our favor, since MUSUS is one of the original Forums.

Frank Lawyer moved that we appoint Harry Wiz SysOp, and Henry Baumgarten seconded the motion. The motion passed with an overwhelming majority.

SysOp Training

CompuServe requires Wiz SysOps to attend a free one-day intensive course in Columbus, to familiarize them with both procedures and the people they will be dealing with. The Board discussed whether to pay for Harry's transportation and lodging for one night.

Harry said that if the trip to Columbus could be arranged to coincide with one of his business trips to that area (probably in mid-May at the earliest) that he would pick up the majority of the traveling expenses. Hays opined that we need a Wiz SysOp NOW and not in mid-May, and that the training is necessary.

Frank asked for clarification of the relationship between the SysOp, the Board, CompuServe, and the President. Hays proposed that:

1) The SysOp is responsible to the President & the Board.
2) The Administrator will handle MUSUS / CIS business matters.
3) The SysOp will handle all Forum technical matters.
4) The MUSUS Contract is between USUS, Inc. and CIS (Inc.).
5) Payments are due to USUS, not the SysOp.
6) Most data from CIS may be disclosed to the Board.

When asked by the Chaircritter if anyone objected, no-one did, and most indicated that they agreed with the proposal. The Chaircritter ruled that the above six points thus embody current USUS Policy on relations with CompuServe.

Bob Clark proposed that we authorize up to $300 to defray Harry's expenses in getting to Columbus for training, and after some discussion, and it was finally agreed that Harry would work out the details with Hays, and do his best to get to

Columbus before May 3, and the authorization was passed.

JPAM

Alex Kleider reported that he had corresponded with the Publisher of the Journal of Pascal, Ada, and Modula-2:

"I've written a letter to the Editor of JPAM inviting his readers to GO MUSUS on CIS and use Sections 2 and 4 for Modula-2 and Pascal related discussions respectively. Mention was also made that an Ada section could be set up if traffic warranted it and that Section 2 could be used in the mean time.

Also have been in touch with the publisher, and the following is being proposed for BoD approval:

1. JPAM be given a list of all our past and present members.
2. All of these would receive a free sample (the next one if we move quickly) issue packaged with an offer that members of USUS can subscribe for a reduced rate of $20.00 (rather than the usual 26) together with a blurb about what USUS is and what we do etc. This needs to be put together quickly but I envision something along the line of what appeared recently in response to the query from Warren would be perfect.
3. He indicated that he'd like to indicate in his Journal that it and USUS are "affiliated". We can negotiate at a more leisurely pace the "official journal" possibility which is by no means ruled out.

"I hereby ask for a vote and if passed I'll ask Hays to provide the list of members and promotional materials to Mr. Richard P. Friedman, JPAM, SIGS Publications, Inc., 310 Madison Ave, Suite 503, New York, NY 10017. If no one else volunteers to do it, I'll amalgamate the stuff that appeared on the board in response to Warren's question and send that as well."

A long discussion ensued, with Bob Spitzer asking if we couldn't get more than one issue out of the deal, William Smith wanted at least three, while Hays and Frank thought it was a reasonably good deal. Alex didn't feel he was enough of a horse trader to go back and re-negotiate the deal, but after no-one else was willing, he agreed. Nobody was enthusiastic about Ada, but it was agreed that if anyone wanted it, it could be provided.

The Board voted to accept the offer from the Publisher of JPAM, but to try to get a couple of more issues for our members.

General Meeting

The Chaircritter pointedly asked the President if any progress had been made on arranging for a General Meeting of the USUS membership for this year. The President said no. Frank said that he had contacted the president of Washington Apple Pi, but hadn't heard anything back.

Next Meeting & Adjournment

The Board agreed to adjourn and meet again at 7 PM PST / 8 PM MST / 9 PM CST / 10 PM EST May 9th 1989 in Room 1 of the MUSUS conference facility

The Special Meeting of the Board on CompuServe was adjourned at 11:30 PM EST on April 18, 1989.

Minutes submitted by:
Samuel B. Bassett

Board of Directors Minutes (May 9, 1989)
by Samuel B. Bassett

MINUTES OF THE SPECIAL MEETING OF THE BOARD OF DIRECTORS OF USUS, INC., HELD IN ROOM 1 OF THE MUSUS FORUM TELECONFERENCING FACILITY ON THE COMPUSERVE INFORMATION SERVICE MAY 9, 1989.

Present at the meeting were:

User ID	Name
76314,1364	Sam'l Bassett, Board Chaircritter
73447,2754	Henry Baumgarten, Board Member
72401,1417	Frank Lawyer, Board Member
75226,3643	A. Robert Spitzer, Board Member
70260,306	A. Hays Busch, Administrator
72135,1667	Harry Baya, MUSUS Wiz SysOp
72747,3126	Robert Clark, Treasurer
73007,173	William Smith, Assistant SysOp, NewsLetter Editor
72740,66	Howard Sweet, Secretary

Matters dealt with were:

Annual Meeting

The Board has tasked the President, Alex Kleider, with arranging for an annual meeting, but has so far heard nothing from him on the subject. The Chaircritter injected the subject into this meeting to try to get some movement in some direction or other, without success.

Members of the Board suggested:

- That we piggyback on whatever trade shows Pecan goes to.
- That we go to MacWorld in Boston in August.
- That we go to the Washington DC Apple Pi meeting.
- That we meet at next year's Auto Show in Detroit
- Columbus Ohio — near CompuServe & cheap hotels.
- Florida.
- Puerto Rico in February.
- New England in the Fall.
- San Francisco State University.

The only consensus reached was that Harry would call Eli Willner about Pecan's plans, and the Chaircritter would build a fire under the President and write an item for the upcoming NewsLetter, asking the membership what they think on the subject.

JPAM

USUS has been talking with the new publisher of the "Journal of Pascal, Ada, and Modula-2" (JPAM) for some time about affiliating. An agreement was reached whereby USUS would supply a copy of the mailing list of current and up-to-one-year-expired members, who would receive a complimentary copy of the magazine, and an offer to subscribe at a reduced rate.

The Board got into their usual wrangle on the subject, with Bob Spitzer wanting to make JPAM the Official Journal, Frank Lawyer not wanting to have anything to do with them, and everybody in between pitching in their two cents.

They discussed how much to charge for membership with and without JPAM, whether the membership would like JPAM, and diverse other subjects. The Chaircritter, being hungry and disinclined to argue, tried to change the subject, but got dragged back by his metaphorical heels.

The upshot was a motion by Harry Bays & William Smith, directing the Chaircritter to put an item in the next NewsLetter, asking the members what they think, and tabling the discussion until we get some input. Everybody but Hays Busch voted a resounding yes, and he wanted to have a word with the Chaircritter before the item was written.

SysOpery

Harry Baya has spent his day in Columbus, learning the ropes of how to be a Wiz SysOp — he reports that it was boring, but the contacts he made with CIS people will prove helpful and useful in the future. He is also hard at work moving things around behind the scenes on MUSUS.

Both Harry and Hays (who acted as temporary Wiz until Harry was appointed) remarked that they had found that Arley Dealey had done an excellent job of organizing the structure of the Forum, but had not carried through on the implementation. A lot of what will appear in the future will be the fruit of the work Arley did, implemented by the new Wiz.

Next Meeting

The Board agreed to adjourn and meet again at 7 PM PST / 8 PM MST / 9 PM CST / 10 PM EST June 13th 1989 in Room 1 of the MUSUS conference facility

The Special Meeting of the Board on CompuServe was adjourned at 11:40 PM EST on May 9, 1989.

Minutes submitted by:
Samuel B. Bassett

The Chaircritter's Soapbox
by Sam'l Bassett

JPAM

Sometime in May or June, you should (have received/be receiving) a copy of the Journal of Pascal, Ada, and Modula-2 (JPAM). Since they are one of the few magazines still supporting

Pascal, USUS has been talking with them about affiliating for some time.

Before we get any further involved, we'd like to get your feedback:

1) What do you think of JPAM?
2) Are you already a subscriber?
3) Are you going to subscribe?
4) Would you like to have a subscription to JPAM as part of your USUS membership?

General Meeting

The Officers and Board of Directors have been going around and around about what to do about holding a General Membership Meeting this year and/or next. We have wobbled back and forth between piggy-backing on some other meeting (Washington [DC] Apple Pi, various IEEE groups, the Detroit Auto Show, etc.), going it alone (the new MUSUS SysOp, **Harry Baya** says that Columbus, OH has nice [and cheap] hotels, but there are lots of places in Florida that would love to have us), or just wandering in circles.

Now is your chance to make a difference — tell the Board where to go! If you have a good (or even not-so-good) idea, let the President, **Alex Kleider** know — his address is:

Alex Kleider
USUS President
1651 Stone Pine Lane
Menlo Park CA 94025
(415) 327-7916

Keep those cards & letters flowing, folks — unless you tell the Board what you want, we just keep stumbling around in the dark.

Apple SIG Doings
by Frank Lawyer

NEW MEMBERS..

Welcome back to **Henry Crosswhite**. Let us know if you need anything. I enjoy letters and telephone calls.

EXIT POLL....

Over the last two months, we polled Apple members who had not renewed their membership, to determine their reasons for leaving USUS. We got 25% responses, which I'm told is pretty good. There were two major trends. The first was that USUS was not providing adequate member services, partly because the member had different needs now. The second, and stronger trend (67%) of replies, was that these ex-members were still programming in Pascal or Modula-2, but were no longer using the p-System. They were using other implementations. A third trend showed that members were switching from Apples to Macs, a natural sort of trend within the Apple community.

What this tells us is that we need to take these trends into consideration, and move along with the times. Please write me and let me know if these trends are influencing you too.

WELCOME DISKS...

The latest welcome disk version is 05/01/89. As always, you can swap for the latest by sending me 2 5.25 floppies, or your old welcome disks. The bibliography, manuals and software files have been expanded for this version. As I look back a year, the welcome disk has expanded from one side which was basically a listing of all the library volumes, into four sides and has become a valuable information source which should be very helpful whether you are new to USUS or are an old hand. Most of the sections are still being expanded, and will continue to be in the foreseeable future.

LIBRARY DISKS...

There were some files added to volume APP2I01, so a new directory should appear elsewhere in this issue. If you purchased APP2I01 in 1989, you are entitled to a (free) updated copy.

Volumes APP2I05 and APP2I06 are released as of 05/01/89. Directory listings should be found elsewhere in this issue. Volume APP2I05 contains the Mac related columns and bulletins from the USUS NewsLetters. APP2I06 contains the Guerrilla Guide courtesy of Bart Thomas. Bart published this a few years back in both the USUS News and Report and the Washington Apple Pi Journal. These files were also on MUSUS for a while. For those who may be unfamiliar, Bart wrote a series of articles which covered the "gotchas" of learning Apple Pascal. Many tips and workarounds were contributed by

USUS members. Bart has suggested that this material needs to be updated through v1.3, and to that end, any volunteers would be gleefully accepted. As I read through it, I realized that many of the traps Bart pointed out are still there, so it will still be good reading.

Both these volumes are Information volumes, and contain only TEXT files, no programs.

The Apple specific library volumes currently available are:

Information volumes:
APP2I01, APP2I02, APP2I03, APP2I04,
APP2I05, APP2I06

Utilities:
APP2U01, APP2U02, APP2U03, APP2U04

Games:
APP2G01, APP2G03, APP2G04

TRENTON FAIR...

We had fun. Harry Baya and I attended and bought tons of white elephants. I am always amazed at this annual event, no matter how many times I go. I did not see too many p-System implementations this year. There were about 3 copies of Apple Pascal v1.0 for sale, most without adequate documentation, one Mac Pecan v4.2.2 implementation, and an IBM published SofTech v4.0 package complete with 5 volumes of documentation, Pascal, Fortran, Assembler and loads of utilities. I don't remember seeing any Modula packages. And if anyone needs Apple3 logic boards, please get in contact, because I got a great bargain on them.

I'm sure other USUS members were there that we didn't know. Next year I am going to start planning in December for this event, since I think it could be very worthwhile.

LETTERS...

Matt Snyder had several questions. They were all centered around his desire "to explore the workings of the p-machine interpreter itself, using my Mac and Apple2." He has a natural desire to move files back and forth between the machines. In this issue of the NewsLetter, I discuss the portability considerations in the Mac column. Somehow they seemed to fit better over there.

Matt says "I have the old Apple Pascal reference on the subject, but I have a feeling that it's not the final word." Well, you're correct on that score. A little later in this column, I'll devote some time to the documentation you will need if you decide you want to delve into the PME (the p-machine emulator).

Also, Matt would like to know how the various p-System implementations differ, how it got that way, and who sets the present standards. Reading the file UCSD.HIST1.TEXT on the welcome disk will give you some background, and I'll have some further comment on this next time.

I received a 2 page letter from **Jeremy Fields**, who is with the Biology department at UCSD. He mentions that up to now he has been "one of those USUS members who has been a consumer only." It really makes me feel good when I get a letter like this, because then I know this column is having some positive influence. Jeremy has both an Apple2+ and a 2e. The Apple2e has an Applied Engineering RAMworks 512K board and a Transwarp. He says "I used Apple Pascal to write a number of programs that analyze the data that my research generates, and was writing some programs to do simple simulations of coupled enzyme systems." Jeremy says he would be interested in communicating with anyone with similar programming interests, and especially someone who has tried writing a plotting program in Apple Pascal. He has a Mannesmann Pixy 3 plotter on his 2e, and wants to finish his library unit to draw standard shapes to the plotter. He could use some help with the generalized scaling routines.

I encourage others to write and share the kind of things they are working on, because you can never tell when others may have similar interests.

PME STUDY...

Most processors are implemented in silicon. Processor chips like the 6502 series which powered the early Apple line contain various arithmetic registers (possibly called accumulators) and status registers, and allow a certain instruction set. If you want to use a different processor, you can buy a new machine that uses that processor, or you could probably write some assembly language software (using your existing processor) which would emulate it. The software route also allows you to dream up trial machines and take them for a test drive. Implementing a new design in silicon is expensive.

The UCSD p-machine (pseudo machine) is implemented in software. (It was implemented in hardware once as the Western Digital Micro-Engine, and some USUS members have this machine). This has the desirable effect of allowing portability across different hardware platforms. It has some registers, some stacks, a heap and some user memory. P-code instructions are one or two bytes long.

For the Apple Computer implementations, the PME architecture is described in the Apple Pascal Operating System Reference Manual (OSR) in Appendix A. This is the basic document for Apple v1.0-v1.2 systems. For Apple v1.3 there is an expanded section 4 (Technical Reference) in the manual. For the Pecan (or SofTech) versions 4.0-4.2.2 the basic document is variously referred to as the Internal Architecture Reference Manual, the Internal Architecture Guide (IAG), or sometimes just THE GUIDE. You will find this the be a more thorough and valuable document than the Apple documentation. Each p-code is explained. The IAG will serve for use with either Apple2 or Mac implementations. Another reference which may be helpful is the Randy Hyde book, p-Source, a Guide to the Apple Pascal System. See the books, manuals, and software files on the Apple (or Mac) welcome disks for additional detail.

Unfortunately, Pecan tells me that the THE GUIDE is currently out of print. It will be reprinted, but they don't know when it will be reprinted, and when it is reprinted, they are not sure what version it would cover. This makes it a little tough to acquire. If anyone has an extra copy they would be willing to part with (for a fee or otherwise) please let me know. Also, those who would purchase a reprint might call Pecan and let them know.

In addition to the literature, you may want some utility programs to allow you to dissect CODE files, and probably a p-code disassembler. No doubt other needed tools will suggest themselves to the intrepid PME adventurer. I looked through the USUS library volumes, but I couldn't come up with either tool. However, I did find a segment mapper which might be useful, and on volume 17, the source code for UCSD v1.3, which is informative. "Perfect Pascal Programs" from Washington Apple Pi has some interesting utilities which may be of help. Also the "Apple Pascal in Depth" from A.P.P.L.E has both a p-code disassembler and a code mapper. Some of this stuff is out of print, so if you can't locate it, let me know and maybe I can photocopy the pertinent parts. Also, if anyone else wants to add their wisdom on this subject, please feel free.

Frank Lawyer, 126 Demott Lane, Somerset, NJ 08873; (201) 828-3616

Mac SIG Doings

by Frank Lawyer

WELCOME DISKS...

The p-System format welcome disk is done, and available if you would rather have that format than the MacWrite format. The file content is the same either way. The next update for the Mac welcome disks will be 07/01/89, and updates are planned quarterly.

I got a note from **David Babb**, who said he has recently acquired MacWrite, and so he can now read the MacWrite based welcome disk he ordered. It brings up an interesting point or two. First, when I made the choice to use a MacWrite based welcome disk, I anticipated that almost all Mac users would have a word processor which at least would READ MacWrite files. I hope that is true and that David is an exception. When he first told me he didn't have MacWrite, I remembered that I had an old version that I bought at a flea market. I thought I could send it to him and save him a few dollars. After a couple of weeks of rummaging in the dusty old software bin, I found it. I booted it and tried to read the welcome disk, and to my dismay found that it would NOT read the files. Beware the backward compatibility problem! Seems I'm currently using v4.5 of MacWrite (not the latest version), and the old copy was v2.2. v4.5 would read (and convert) v2.2 files, but not the other way around. Sorry David. I'm looking at some PD programs which will read MacWrite files, so I may solve the problem by including one of these programs on the disk. Fortunately, the p-System format remains unchanged, so any version of the p-System on the Mac reads the files on the p-System welcome disk.

LIBRARY DISKS...

There are no new Mac specific disks this month. There should be several more by the time the next NewsLetter gets out.

POWERTOOLS...

As you may know, USUS has its own software product. PowerTools provides an integrated environment of a word processor, a file manager, and a communications program. These are integrated under a menu "shell" which provides an easy way to get to any of the components. The user can also tailor the menu to execute up to nine of his favorite programs with just one keystroke.

I purchased a copy of Power Tools for my Mac Plus, and I can warmly endorse a similar purchase by any of you out there.

To me, one of the nice things about having PowerTools is that you receive a "run time" version of the Pecan Power System. This is an almost complete Pecan implementation of the p-System version 4.2.2 for the Mac. Only the program development tools like the Compiler and Linker are absent, and the system will allow you to run the regular full-screen Editor (not ASE), the Filer etc. Using the R(un command from the PowerTools menu, you can run all the p-System programs from the USUS disks which are being converted.

I learned from a number of USUS members with Macs, that they were going away from the p-System on their machines, and toward other implementations like TML, Lightspeed etc. Converting the existing USUS p-System library to Mac format seemed like asking everyone to buy a full blown development Pecan Power System. The current price of that system is $249, and an upgrade of a previous version is $129. With PowerTools, this problem is solved, because for $59.95, you can get everything you need short of the development components.

Before the RAM prices went into the stratosphere, I had my Mac upgraded to 2.5 megs, because the size of the Mac System and Filer were beginning to get into the way of application programs. If you decide to upgrade a 1 megabyte (MB) Mac to the nominal 2 MB level, you should insist on keeping the original SIMMS, because they belong to YOU, not your dealer. Most dealers will put 512 KB of your old memory right back in the machine for a total of 2.5 MB. This configuration is fully supported, and sometimes that extra 512 KB comes in handy. No, you can't use the other 512KB to bring it to 3 MB!

That was a preface to another nice feature of PowerTools. On booting PowerTools, all the p-System files are copied to RAMDISK:, utilizing all the RAM that can be found. On my Mac Plus, I wind up with 1800+ blocks. You can do wonders with that kind of space. Of course, you have to remember to copy out your files to a floppy so you don't lose them when you leave PowerTools.

There are other good features of PowerTools that I don't have time to discuss now. This wasn't intended to be a full scale review, only to discuss PowerTools from the standpoint that it provides an inexpensive way to run p-System programs on the Mac.

PowerTools was developed by USUS Professional members **Jon L. Nevins** and **Bob Clark**. This is truly a USUS oriented product because some of the source began with files available in the USUS Software library. PowerTools is available to USUS members as an added source of income for the benefit of all members, and it is a valuable benefit of membership.

If you want to use the TermTools communications part of PowerTools with your Mac, you'll need a Hayes (or other fully compatible) modem. In my opinion, PowerTools is well worth the price, even without TermTools. If you would like to order USUS PowerTools, simply print out the file PTOOLS.ORD.TEXT from the Mac Welcome disk. This file and the file PTOOLS.DOC.TEXT contain additional detailed information on the product. If you don't have a welcome disk, we can supply order forms etc.

COMPATIBILITY...

I mentioned last time that Matt Snyder had sent me some questions on trading programs back and forth between the Mac and the Apple2. Matt has the understandable need to move programs between machines, and after all, one of the big selling features of the p-System is that programs are transportable across machine boundaries. Generally, that is TRUE, but there are a few things to consider and watch out for. The problem breaks down into two areas of compatibility, source code and object code.

While I have every reason to believe that the situation is nearly identical for other hardwares, I have only ported p-System Pascal programs back and forth between the Apple2 series and the Mac, the Apple2 series and the Apple3 (trivial), and the Apple2 series and MS-DOS machines, so I'll have to confine my discussion to those environments. If others have discovered problems that I fail to discuss here, let me know. Also, although I have had some experience doing porting on p-System files, I do not claim to be an EXPERT. The discussion here is limited to UCSD based Pascal systems. The subject of portability to other non-UCSD environments is something for another discussion.

I'll attempt short answers first, with some greater detail later. The SOURCE code is almost completely portable across machines and vendor implementations, and the object code is HIGHLY portable, provided we confine ourselves to porting between the same version of the p-System. That SOUNDS simple enough.

The SOURCE files are almost completely interchangeable between versions, machines, and vendors. Of course, we need to somehow transfer the files between machines to take care of the differences in physical disk encoding techniques. I will discuss the physical transfer in a separate article, when we have more room. Once on the target machine, p-System .TEXT file(s) can be compiled and linked into an executable .CODE file, just as on the original machine. There are some slight variations in syntax and reserved words between the various compiler versions, and there may be some UNITs references (e.g. APPLESTUFF) which won't compile, but conversion is usually pretty easy. If you know you are going to port, you can plan ahead to avoid those things, and write in a common subset.

In my opinion, transferring the source files, and then re-compiling on the new machine is the correct way to proceed. I have never met a program I didn't want to change, and having the source code gives me that chance. That way you also make sure that any slight differences in syntax, now or in the future, can be taken care of. If you only have the .CODE file, you don't have the luxury.

Now to tackle the hard part. The .CODE files are interchangeable between identical p-System versions, and generally upward compatible within a series of vendor implementations. So Apple Computer Pascal v1.0 code runs on v1.1, and Apple v1.1 code runs on v1.2, and Apple v1.2 code runs on v1.3. But, for instance, certain changes were made to TURTLEGRAPHICS and other units by Apple between v1.1 and v1.2, and in v1.2 upkey and downkey values were changed, so if your program used those, it MIGHT have to be recompiled. If you used IDSEARCH or TREESEARCH in previous versions, you will have to remove them (and recompile) for v1.3 etc. You have to check the manuals specifically, but, within Apple, it's not too bad. Compiling on the new version is always best, I think, but I wouldn't hesitate to use an old code file on a new version, unless I KNEW there was a UCSD version difference.

I am a little less comfortable saying that Pecan v4.1.3 code files will run under Pecan v4.2.2 systems. I have ported a few simple programs, and had no problems. I would have to dig into the manuals and see if this were generally true. I suspect that there are some differences. Pecan would be the best authority. If Eli Willner or someone else at Pecan would like to contribute the information, I'm sure we would like to know it.

The key to porting .CODE files between machines, is that the p-System versions have to be identical. So version 4.1.3 .CODE files should interchange directly between an Apple and a Mac, in either direction. Both systems would be using Pecan v4.1.3 (since 4.1.3 is older, you might have SofTech versions). For Pecan v4.2.2 running on the Apple2GS and the Mac, the .CODE files should also interchange. Assembly code portions are obviously out, as is any program LINKED with assembly code portions. I have ported Apple v4.1.3 .CODE to IBM, and it ran fine. I have not yet attempted the Apple to Mac port of .CODE files. Even with the same UCSD version, you need to be careful in certain circumstances. For instance Apple2 Pascal and Apple3 Pascal are both based on UCSD v2.1, but we cannot interchange .CODE files. We CANNOT interchange Apple .CODE files with Pecan .CODE files, since all the Apple Computer implementations are based on UCSD v2.1.

Now the specific implementations. First, the Apple2 series. Pecan currently has a version for Apple2e and below, which is UCSD v4.1.3. And it is a stand-alone version (doesn't require support of any other Operating System files). All versions v1.0 through v1.3 by Apple Computer are stand-alone, and will run on the Apple2e and below. The Apple2GS Pecan version is ProDOS hosted, and is UCSD v4.2.2. The Pecan version for the Apple2GS is the ONLY UCSD version, so far as I know. The Apple versions and the Pecan 4.1.3 version will run on the Apple2GS in compatibility mode only (as an Apple2e). Pecan v4.2.2 DOES NOT RUN on the Apple2e and below. Although there is no official pronouncement from Apple, it seems certain to me they will NOT be bringing out a UCSD version of Pascal for the 2GS.

As far as Mac implementations, SofTech had the original Mac version which was UCSD 4.1.3, and they eventually upgraded this thru v4.2.1. Pecan now has a Mac implementation v4.2.2. All implementations for the Mac were Finder based. The Pecan implementations are the only UCSD implementations for the Mac series, as far as I know. The Mac implementations do not contain a p-System Assembler. A separate Assembler development package is available. Apple Computer has no UCSD version for the Mac series.

For greater detail on the versions, see the software file, APPLE.SOFT.TEXT on the welcome disk. The most recent version of this file is on the welcome disk of 04/01/89.

It seems to take a lot of words to explain this, but I hope the explanation is clear to everyone. If not, send me your improved versions.

Matt had one or two more questions that we'll try to get to next time, and in the meantime, don't anybody ask about FORTRAN!

Frank Lawyer, 126 Demott Lane, Somerset, NJ 08873; (201) 828-3616

WDS File Ops Unit(s)

By William D. Smith

This months column was going to cover my SysOps unit, but in preparation, I found that I needed to cover my low level file I/O units first. There is one unit which I use for my low level file I/O and two units called by that one. The two called are part of the operating system and I wrote a partial interface so that the compiler will generate calls to these system units. These two units are `FileOps` and `ExtraIo`. You will notice that neither unit has an implementation part.

How does this work? The compiler needs an interface part when it encounters the uses statement. It finds it in the file specified by the "`$U`" compiler directive (UTIL.LIB in my case, see VERSION.TEXT). On the other hand, when a program is run which need these units, the runtime system finds the implementation code of the unit in SYSTEM.PASCAL. In this way, you can have multiple interfaces to a unit **as long as the interfaces all agree with the code**. For example, Pecan has the full unit (interface and implementation of these units). They distribute the code (implementation) as part of the operating system. I used DECODE to figure out which procedures I need and what the interface had to conform to and then wrote the ones on the following pages. These are not the same interfaces as Pecan has, but the code used at runtime is.

Why did I do this? Because I wanted dynamic file declarations (ie. **`array`** `[First..Last]` **`of file`** which is not allowed in UCSD Pascal).

How does it work? When you declare a variable as type `file` (or `text`), the compiler allocates a file information block (fib) on the stack and inserts into your code (prior to any thing you write in the procedure) a call to `F_Init` (in `FileOps`) to initialize the fib (but not open the file). The compiler also puts a call to `F_Close` at

(continued next page)

```
{ IV.22 system fileops unit (fake) [1.02] --- 27 Feb 88 } { |xjm$d|nx|f8|e|. }
{$Q+}
{ File:         F.FileOps.Text              Version 1.02    27 Feb 88
  Author:       William D. Smith            Phone: (619) 941-4452
                P.O. Box 1139               CIS: 73007,173
                Vista, CA  92083
  System:       Power System version IV.2.2
  Compiler:     Power System Pascal Compiler
  Keywords:     Fake FileOps Unit
  Description:  This unit is used so that the compiler will generate calls to the
                system FileOps unit.  It is used to fake out the compiler.  It
                works with version IV.22.  It should only be called by F_Io_U.
  Change log: (most recent first)
Date       Id   Vers  Comment
---------  ---  ----  -------
27 Feb 88  WDS  1.02  Changed J to FplusSize in F_Init.
25 Feb 88  WDS  1.01  Put in interface comment.
08 Feb 88  WDS  1.00  Initial try at writing.
}
{$I VERSION.TEXT}  { Declares conditional compilation flags }
{$U-}  { Compile at system level }
unit  FileOps;
interface  {$ FileOps (fake) [1.02] 25 Feb 88 }
type  Ptr = ^integer;
  procedure F_Open   (F : Ptr;  Name : string;  OldFile : boolean;  J : Ptr);
  procedure F_Close  (F : Ptr;  CloseHow : integer);
  procedure F_Init   (F : Ptr;  FplusSize : integer;  RecSizeWords : integer);
implementation
end {$Q- FileOps }.
```

the end of the procedure to close the file (just in case you forgot to close the file). These procedures (and the other two I need, `F_Open` in `FileOps` and `F_BlockIo` in `ExtraIo`) all take the address of the fib as a parameter. They don't care if the address points to the stack or the heap. So I declare a variable as type `FibPtr` (from my globals unit declared as `^integer`), set it to `Closed` (also in `Glbs_U`, initialized to **`nil`**) and pass this variable to `OpenFile` (in `F_Io_U`) which allocates (using `new`) a fib on the heap (not on the stack) and opens the file. `CloseFile` (in `F_Io_U`) closes the file and deallocates the heap space (using `dispose`, which sets the `FibPtr` back to **`nil`**, ie. `Closed`). Since the compiler never knows that these pointers are files, it doesn't allocate space for it or insert extra code to initialize the fib or close the file. It also allows declarations like **`array`** `[First..Last]` **`of`** `FibPtr` or **`record`** `F : FibPtr;` **end**; etc. there by giving dynamic files.

Some notes of caution. You, the programmer are responsible for opening and closing the file correctly. Also since the heap is used to hold the fib, doing

```
mark (Heap);
OpenFile (F, ...);
release (Heap);
CloseFile (F, ...);
```

will not work. Your fib was released with the release statement prior to closing and you will get a heap error when `CloseFile` is called. And these units only work with version 4.22 of the p-System and only with the standard file system (SFS), not the advanced file system (AFS).

(continued next page)

```
{ IV.22 system ExtraIo unit (fake) [1.01] --- 25 Feb 88 } { |xjm$d|nx|f8|e|. }
{$Q+}
{ File:         F.ExtraIo.Text             Version 1.01    25 Feb 88

  Author:       William D. Smith           Phone: (619) 941-4452
                P.O. Box 1139              CIS: 73007,173
                Vista, CA  92083

  Description:  This unit is used so that the compiler will generate calls to the
                system ExtraIo unit.  It is used to fake out the compiler.  It
                works with version IV.22.

  System:       Power System version IV.2.2

  Compiler:     Power System Pascal Compiler

  Keywords:     Fake ExtraIo Unit

  Change log: (most recent first)
Date       Id   Vers  Comment
---------  ---  ----  -------
25 Feb 88  WDS  1.01  Put in interface comment.
08 Feb 88  WDS  1.00  Initial try.
}
{$I VERSION.TEXT}  { Declares conditional compilation flags }
{$U-}  { Compile at system level }
unit  ExtraIo;
interface  {$ ExtraIo (fake) [1.01] 25 Feb 88 }
type  Ptr    = ^integer;
      Window = packed array [0..0] of char;

  function F_BlockIo (     F        : Ptr;
                       var A        : Window;
                           Index    : integer;
                           Blocks   : integer;
                           BlockNum : integer;
                           DoRead   : boolean) : integer;
implementation
end {$Q- ExtraIo }.
```

F_Io_U contains three procedures to open, close and do block I/O on a file. There is a procedure to delete a file (PurgeFile) and a procedure to determine if the file pointer is at the end of the file (AtEof). The remaining procedures are used to get information from the fib and return information to the fib.

F_Init needs the address of the fib, the address of the end of the fib, and what kind of file it is (this unit supports block files and text files, but not record files). Only enough space for the type of file is allocated to the fib (a text file needs a 258 byte buffer). F_Open takes the fib, the name of the file and whether it is a new file or an old file and nil (a pointer which I don't know how it is used, but the compiler always passes **nil**). The p-machine instruction

```
  pmachine (^Fp, (F), Sto);
```

is used to change my FibPtr (a ^integer) to the system FibP (a ^Fib).

On page 21 I included a copy of the version file I use. From this you can see which version of the p-System I have supported (I'm no longer using IV.13 or IV.21). If I was using Apples Pascal I would include a line declaring a conditional compilation flag for those versions of the p-System.

```
{ WDS file I/O unit                  [1.04] --- 09 Mar 88 } { |xjm$d|nx|f8|e|. }
{$Q+}
{$C (c)  William D. Smith  -- 1988,  All rights reserved.                      }

{ File:          F_Io_U.Text              Version 1.04    19 Mar 88

  Author:        William D. Smith         Phone: (619) 941-4452
                 P.O. Box 1139            CIS: 73007,173
                 Vista, CA  92083

  Notice:        The information in this document is the exclusive
                 property of William D. Smith.  All rights reserved.
                 Copyright (c) 1987 to 1989.

  System:        Power System version IV.2.2

  Compiler:      Power System Pascal Compiler

  Keywords:      WDS F_Io_U File I/O Unit

  Description:  This unit contains procedures to handle files at the block level.

  Change log:  (most recent first)

Date       Id   Vers   Comment
---------  ---  ----   -------
19 Mar 88  WDS  1.04   Fixed some errors in Open, Close, PurgeFile.
09 Mar 88  WDS  1.03   Set F_Modified to false when not saving file.
28 Feb 88  WDS  1.02   Added NonFile to FileType.
27 Feb 88  WDS  1.01   Fixed error in OpenFile w/ F_Init when textfile.
14 Feb 88  WDS  1.00   Finished writing and testing.
}

{$I VERSION.TEXT}  { Declares conditional compilation flags }

{$D CK+}  { Set to "+" to check file system level at runtime }

unit F_Io_U;

interface   {$ F_Io_U [1.04] 19 Mar 88 }

{ This unit only works with the Standard File System.  OpenFile and CloseFile
  use the heap for the file information block.  Be careful if you use mark and
  release.
}

uses   Glbs_U;   { WDS globals unit }

const Vc_F_Io_U    = 2;  { 28 Feb 88 }
      Vs_F_Io_U    = 'F_Io_U';

type   FileType     = (NonFile, BlkFile, TxtFile);

var    Vv_F_Io_U    : integer;

  function BlockIo (F          : FibPtr;
                var Buf        : interface
                                   packed array [Lo..Hi : integer] of char;
                    Blocks     : integer;
                    BlockNum   : integer;
                    DoRead     : boolean) : integer;
  { This function reads or writes Blocks blocks to the file F.  BlockNum is the
    starting block in the file for the read or write.  Buf is the array which is
    read into or wrote out.  DoRead is true for reads and false for writes.  The
    function returns the number of blocks actually read.  A block is 512 bytes.
  }

  function OpenFile (var F       : FibPtr;
                         Name    : Str_23;
                         Kind    : FileType;
```

```
                           OldFile : boolean;
                       var Msg     : integer) : boolean;
{ Open file.  This function opens the file named by Name and associates it
  with the file variable F.  Kind is the type of the file and determines what
  kind of I/O can be done.  If OldFile is true, an old file is opened,
  otherwise a new file is created.  Msg returns the system ioresult.  The
  function returns true only if Msg is equal to M_NoError.  F is returned as
  Closed if there was an error otherwise it points to a file info block. Do
  not call this function between mark/release pairs unless CloseFile with the
  same F is also called.
}

procedure CloseFile (var F : FibPtr;  Save : boolean);
{ Close file.  This procedures closes the file associated with F.  If Save is
  true the file is locked closed.  The file info block pointed to by F is
  disposed of and F is returned as Closed.
}

procedure PurgeFile (Name : Str_23;  var Msg : integer);
{ Purge file.  This procedure removes the file named by Name.  Msg is returned
  as the ioresult.  If the file was found and purged, Msg is returned as
  M_NoError.
}

function AtEof (F : FibPtr) : boolean;
{ At end of file.  This function returns true if F is its end. }

function File_Is_Blocked (F : FibPtr) : boolean;
{ This function returns true only if the file F, is on a blocked device and is
  open.
}

function File_UnitNumber (F : FibPtr) : integer;
{ This function returns the unitnumber of the device which contains the file,
  F.  If F is not open, Null is returned as the unitnumber.
}

function File_Size (F : FibPtr) : integer;
{ This function returns the size of the file F.  If F is not open or not
  blocked, Null is returned as the size.
}

function File_Start (F :FibPtr) : integer;
{ This function returns the first disk address of the starting block of the
  file.  If F is not open, Null is returned as the start.
}

procedure Get_File_Tad (F : FibPtr;  var Tad : TadRec);
{ Get file time and date.  This procedure returns the time and date the file
  was last update (ie. the time and date in the directory). If the file is not
  open, Tad is returned as NullTad.
}

procedure Set_File_Tad (F : FibPtr;  Tad : TadRec);
{ Set file time and date.  This procedure sets the time and date of the file F
  to Tad.  If the file is not open, nothing is done.
}

procedure Get_LastByte (F : FibPtr;  var LastByte : integer);
{ Get last byte of last block.  LastByte is returned as the number of bytes in
  the last block.  If the file is not open, LastByte is returned as Null.
}

procedure Set_LastByte (F : FibPtr;  LastByte : integer);
{ Set last byte of last block.  The number of bytes in the last block is set
  to LastByte.  If the file is not open, nothing is done.
```

```
   }
  procedure Get_Filename (F : FibPtr;  var Name : Str_15);
  { Get filename.  This procedure returns the name of the file F. Name does not
    include the volume name.  If F is not open, Name is returned as the empty
    string.
   }
  procedure Set_Filename (F : FibPtr;  Name : Str_15);
  { Set filename.  This procedure set the name of the file F. Name must not
    include the volume name.  If F is not open, nothing is done.
   }
implementation
uses   Kernel (F_BlkSize, F_TypeText, CloseType, FibP, Fib, MemPtr,
               FileSystemLevel, PtrAdd),
       FileOps (F_Open, F_Close, F_Init),   { System file ops unit }
       ExtraIo (Window, F_BlockIo);         { System extra I/O unit }

const  Sto          = 196;
       PageSize     = 257;          { 1 block + 1 word for window pointer }

  function BlockIo { (F        : FibPtr;
                 var Buf      : interface
                                 packed array [Lo..Hi : integer] of char;
                     Blocks   : integer;
                     BlockNum : integer;
                     DoRead   : boolean) : integer };
  var A : ^Window;
  begin
    if F <> Closed then
      begin
        pmachine (^A, ^Buf, Sto);  { A := address (Buf); }
        BlockIo := F_BlockIo (F, A^, Lo, Blocks, BlockNum, DoRead);
      end { if }
    else  BlockIo := 0;
  end { BlockIo };

  function OpenFile { (var F        : FibPtr;
                           Name     : Str_23;
                           Kind     : FileType;
                           OldFile  : boolean;
                       var Msg      : integer) : boolean };
  var I, Size : integer;  P : MemPtr;  Fp : FibP;
  begin
    Msg := M_NoError;
    Size := (sizeof (Fib) - (F_BlkSize + 1)) div 2;

    if Kind <> BlkFile then
      Size := Size + PageSize;

    if varnew (F, Size) = Size then
      begin
        P .I := F;
        P .I := PtrAdd (P, Size - 1);
        F_Init (F, P .T, - ord (Kind));
        F_Open (F, Name, OldFile, nil);
        Msg := ioresult;

        if Msg = M_NoError then
          begin
            if Kind <> BlkFile then
              begin
                pmachine (^Fp, (F), Sto);  { Fp := F; }
               if not Fp^ .F_SoftBuf then
                 begin
                  P .I := F;
                 P .I := PtrAdd (P, Size - PageSize);
                  vardispose (P .I, PageSize);
                 end { if };
             end { if };
        end { if }
      else  vardispose (F, Size);
    end { if }
  else  Msg := M_NoHeap;
  OpenFile := Msg = M_NoError;
end { OpenFile };
procedure CloseFile { (var F : FibPtr;  Save : boolean) };
var Fp : FibP;  Size : integer;
begin
  if F <> Closed then
    begin
      pmachine (^Fp, (F), Sto);  { Fp := F; }
      Size := (sizeof (Fib) - (F_BlkSize + 1)) div 2;
      if Fp^ .F_SoftBuf then
        Size := Size + PageSize;
      if Fp^ .F_MaxBlk = 0 then  Save := false;  { Fix for system error }
      if Save then
        F_Close (F, ord (C_Lock))
      else
        begin
          Fp^ .F_Modified := false;  { Directory entry not changed }
          F_Close (F, ord (C_Normal));
        end { else };
      vardispose (F, Size);
    end { if };
end { CloseFile };
procedure PurgeFile { (Name : Str_23;  var Msg : integer) };
var F : FibPtr;
begin
  Msg := M_NoError;
  if OpenFile (F, Name, BlkFile, true, Msg) then
    begin
      F_Close (F, ord (C_Purge));
      Msg := ioresult;
      vardispose (F, (sizeof (Fib) - (F_BlkSize + 1)) div 2);
    end { if };
end { PurgeFile };
function AtEof { (F : FibPtr) : boolean };
var Fp : FibP;
begin
  AtEof := true;
  if F <> Closed then
    begin
      pmachine (^Fp, (F), Sto);  { Fp := F; }
      if Fp^ .F_IsOpen then
        AtEof := Fp^ .F_Eof;
    end { if };
end { AtEof };
function File_Is_Blocked { (F : FibPtr) : boolean } ;
var Fp : FibP;
```

```
begin
  File_Is_Blocked := false;
  if F <> Closed then
    begin
      pmachine (^Fp, (F), Sto);  { Fp := F; }
      if Fp^ .F_IsOpen then
        File_Is_Blocked := Fp^ .F_IsBlkd;
    end { if };
end { File_Is_Blocked  };
function File_UnitNumber { (F : FibPtr) : integer } ;
var Fp : FibP;
begin
  File_UnitNumber := Null;
  if F <> Closed then
    begin
      pmachine (^Fp, (F), Sto);  { Fp := F; }
      if Fp^ .F_IsOpen then
        File_UnitNumber := Fp^ .F_Unit;
    end { if };
end { File_UnitNumber  };
function File_Size { (F : FibPtr) : integer };
var Fp : FibP;
begin
  File_Size := Null;
  if F <> Closed then
    begin
      pmachine (^Fp, (F), Sto);  { Fp := F; }
      if Fp^ .F_IsOpen then
        if Fp^ .F_IsBlkd then
          File_Size := Fp^ .F_MaxBlk;
    end { if };
end { File_Size };
function File_Start { (F :FibPtr) : integer };
var Fp : FibP;
begin
  File_Start := Null;
  if F <> Closed then
    begin
      pmachine (^Fp, (F), Sto);  { Fp := F; }
      if Fp^ .F_IsOpen then
        if Fp^ .F_IsBlkd then
          File_Start := Fp^ .F_Header .D_FirstBlk;
    end { if };
end { File_Start };
procedure Get_File_Tad { (F : FibPtr;  var Tad : TadRec) };
var Fp : FibP;
begin
  Tad := NullTad;
  if F <> Closed then
    begin
      pmachine (^Fp, (F), Sto);  { Fp := F; }
      with Fp^, F_Header do begin
        if F_IsOpen then
          if F_IsBlkd then
            begin
              Tad .D := D_Access;
              if D_Hour > 0 then
                begin
                 Tad .T .Min := D_Minute;
                 Tad .T .Hour := D_Hour - 1;
                end { if };
            end { if };
      end { with }
    end { if };
end { Get_File_Tad };
procedure Set_File_Tad { (F : FibPtr;  Tad : TadRec) };
var Fp : FibP;
begin
  if F <> Closed then
    begin
      pmachine (^Fp, (F), Sto);  { Fp := F; }
      with Fp^, F_Header do begin
        if F_IsOpen then
          if F_IsBlkd then
            begin
             D_Access := Tad .D;
              if Tad .T <> NullTAd .T then
                begin
                 D_Minute := Tad .T .Min;
                 D_Hour := Tad .T .Hour + 1;
                end { if };
            end { if };
      end { with }
    end { if };
end { Set_File_Tad };
procedure Get_LastByte { (F : FibPtr;  var LastByte : integer) };
var Fp : FibP;
begin
  LastByte := Null;
  if F <> Closed then
    begin
      pmachine (^Fp, (F), Sto);  { Fp := F; }
      if Fp^ .F_IsOpen then
        if Fp^ .F_IsBlkd then
          LastByte := Fp^ .F_LastByte;
    end { if };
end { Get_LastByte };
procedure Set_LastByte { (F : FibPtr;  LastByte : integer) };
var Fp : FibP;
begin
  if (LastByte < 0) or (LastByte > F_BlkSize) then
    LastByte := F_BlkSize;
  if F <> Closed then
    begin
      pmachine (^Fp, (F), Sto);  { Fp := F; }
      Fp^ .F_LastByte := LastByte;
    end { if };
end { Set_LastByte };
procedure Get_Filename { (F : FibPtr;  var Name : Str_15) };
var Fp : FibP;
begin
  Name [0] := chr (0);  { Name := ''; }
```

```
      if F <> Closed then
        begin
          pmachine (^Fp, (F), Sto);  { Fp := F; }
          if Fp^ .F_IsOpen then
            if Fp^ .F_IsBlkd then
              Name := Fp^ .F_Header .D_Tid;
        end { if };
    end { Get_Filename };
    procedure Set_Filename { (F : FibPtr;  Name : Str_15) };
    var Fp : FibP;
    begin
      if length (Name) > 0 then
        if F <> Closed then
          begin
            pmachine (^Fp, (F), Sto);  { Fp := F; }
            Fp^ .F_Header .D_Tid := Name;
          end { if };
    end { Set_Filename };
  begin { F_Io_U }
    Vv_F_Io_U := Vc_F_Io_U;
    Ck_Version (Vv_Glbs_U, Vc_Glbs_U, Vs_F_Io_U, Vs_Glbs_U);
  {$B CK+}
    if FileSystemLevel <> SFS then
      begin
        write ('F_Io_U and Tx_Io_U only work with SFS <ret>');
        readln;
      end { if };
  {$E CK+}
    *** ;
  end {$Q- F_Io_U }.
```

Version.Text

```
{ System version (AOS-,IV13-,IV21-,IV22+) --- 23 Oct 87 } { |xjm$d|nx|f8|e|. }
{$D BYTE+} { Byte addressed machine }
{$D WORD-} { Word addressed machine }
{$D AOS-}  { Using Advanced Operating System }
{$D IV13-} { Using version IV13 p-System }
{$D IV21-} { Using version IV21 p-System }
{$D IV22+} { Using version IV22 p-System }
{$D IV+}   { IV13, IV21 or IV22 }
{$I-}      { No I/O checking }
{$R-}      { No range checking }
{$B AOS+}
{$H+}      { Version IV heapops }
{$E AOS+}
{$B IV13+}
{$U M13:UTIL.LIB}  { For IV13 compilation, all units must reside here }
{$E IV13+}
{$B IV21+}
{$U M21:UTIL.LIB}  { For IV21 compilation, all units must reside here }
{$E IV21+}
{$B IV22+}
{$U M22:UTIL.LIB}  { For IV22 compilation, all units must reside here }
{$E IV22+}
{$D INS_STR_FWD-}  { Used by Ins_Str }
```

New Library Disk Directories

What follows is a listing of the directory of the newly released Apple disk. The disk was developed by the Apple SIG. See the **Apple SIG Doings** column in the **August 88 NewsLetter** for more information on the naming conventions.

APP2I01:

```
UNITS.DOC.TEXT.......Info on UNITS, SEGMENTS and EXTERNAL Routines
REQUESTS.TEXT........Requests for programs and routines needed for library
UNIVERSAL.TEXT.......Suggestions for some universal routines for library
USUS.NEWS.TEXT.......News of the formation and early doings of USUS
UCSD.HIST1.TEXT......Part 1 of the UCSD p-System History
UCSD.HIST2.TEXT......Part 2 of the UCSD p-System History (in progress)
LIBR.J14.TEXT........Suggestion for a Library information program
APPLE.DISK.TEXT......Information on use, packing and shipping disks
APP.SYSP13.TEXT......Describes the SYSTEMSTUF UNIT and Apple Pascal v1.3 vars
APPLE.SOFT.TEXT......List of Systems oriented software for Apple2, 3, Mac
APPLE.MANS.TEXT......List of vendor supplied manuals for Systems software
COMAPP2I01.TEXT......Comments on the above files
DIRAPP2I01.TEXT......You're reading it
```

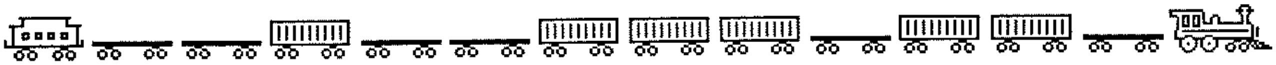
APP2I05:

```
APSIG.MAC1.TEXT......Apple SIG bulletin sent to Mac members 08/88
APSIG.MAC2.TEXT......Mac SIG bulletion sent to Mac ownders 02/89
MACSIG.01.TEXT.......Mac SIG column for NewsLetter of 02/89
MACSIG.02.TEXT.......Mac SIG column for NewsLetter of 04/89
MACSIG.03.TEXT.......Mac SIG column for NewsLetter of 05/89-06/89
COMAPP2I05.TEXT......Comments on the above files
DIRAPP2I05.TEXT......You're reading it
```

APP2I06:

```
PASIN1.TEXT..........Part 1 of the Bart Thomas Guerrrila Guide
PASIN3.TEXT..........Part 3 of the GG
PASIN2.TEXT..........Part 2 of the GG
PAS2E.TEXT...........Special part of the GG for the Apple2e only
PBOOKS.TEXT..........Bart's compiled list of books
GG.DOC.TEXT..........Documentation for the files of the Guerrila Guide
INDEX.TEXT...........Expanded index to Jensen and Wirth PASCAL USER MANUAL
COMAPP2I06.TEXT......Comments on the above files
DIRAPP2I06.TEXT......You're reading it
```

NOTE: There are no program files or code files on an Information Disk. The files all contain text of some sort, and can be printed out on your printer, or scanned with the Editor. In general, all new SIG related articles, bulletins, NewsLetter articles etc. will go on these disks. Also, some older files which appear on the Library disks will move here if they are still interesting or germane.



Macworld (May 1989) has just rated MegaScreen the *Number One Choice* in large screen displays for the Macintosh SE; quality, features and price/performance are the reasons why.

User Group pricing and our toll-free purchase assistance line mean there's no reason to wait!

Save up to 41% when you join with other members of your user group to buy selected MegaGraphics Display Systems.

Call MegaGraphics at (800) 423-0183 for details. (Inside California, call (805) 484-3799) Ask for User Group Sales.

User Group Special (Expires July 31)

MegaGraphics Product	Macintosh Compatibility	Suggested Retail Price	User Group Price*	Savings
MegaScreen SE Black & White 19″ Monitor & Controller	Macintosh® SE	$1,895	**$1,259****	SAVE **$636**
MegaScreen 2001 Black & White 19″ Monitor & Controller	Macintosh II, IIx, IIcx	$1,995	**$1,299**	SAVE **$766**
MegaScreen 2008 Greyscale 19″ Monitor & Controller	Macintosh II, IIx, IIcx	$3,290	**$1,999**	SAVE **$1,291**
MegaScreen 2008 256 Color 19″ Trinitron® Monitor & Controller	Macintosh II, IIx, IIcx	$6,090	**$3,999**	SAVE **$2,091**
MegaGraphics 2008 Controller card only, no monitor	Macintosh II, IIx, IIcx	$1,695	**$999**	SAVE **$696**

MegaGraphics offers the only full line of true WYSIWYG/75 d.p.i. displays available for Macintosh.

MegaScreen 2008 Color System for the Macintosh II uses a special Trinitron-technology monitor tuned to our uniquely high refresh rate and 1024 x 826 resolution.

MegaScreen 2008 Greyscale System offers 256 simultaneous grey tones for photographic-quality images.

MegaScreen Black & White Systems all provide rock steady, flicker-free displays for all current Macintosh models.

Each MegaScreen System offers the best performance in its category, with a specially-tuned high performance monitor to work with the high refresh rate of our MegaScreen Controllers. The result of all this technology is a stable, flicker-free display that's comfortable to use all day long.

M · E · G · A
Graphics
INC.

MegaScreen Display Systems by MegaGraphics. We make technology beautiful.

MegaGraphics, 439 Calle San Pablo, Camarillo, CA 93010 (800) 423-0183. Inside California: (805) 484-3799. Ask for User Group Sales.

*Applicable sales tax and shipping are extra. Restrictions apply: Discounts are available to current user group members when their group buys at least 3 systems.
**MegaScreen SE price includes a $20 coupon for installation by the Apple servicing dealer of your choice.
These are not discontinued items, nor do they differ from our regular retail products. Trinitron is a Registered Trademark of Sony Corporation. Macintosh is a Registered Trademark of Apple Computer, Inc.

May/June 1989

USUS NewsLetter

Volume 3
Number 5

William D. Smith, Editor

NewsLetter Publication Dates

NewsLetter	Due date Code/Forms	Due date Articles	Due date Short stuff
Sept/Oct 89	08/15/89	08/25/89	09/01/89
November 89	10/01/89	10/13/89	10/20/89
December 89	11/01/89	11/10/89	11/17/89
January 90	12/01/89	12/08/89	12/15/89

Next NewsLetter coming July/Aug

USUS
P.O BOX 1148
LA JOLLA, CA 92038

ADDRESS CORRECTION REQUESTED

[redacted]