

The Fall 1985 BMUG Newsletter

**Copyright ©1985 Berkeley Macintosh Users Group,
1442A Walnut St., Suite #153, Berkeley, CA 94709
(415) 849-9114**

BMUG BBS: (415) 849-BMUG Delphi: BMUG BMUG EAST BBS: (212) 925-4630

Editor-in-Chief

Reese Jones

Managing Editor

Dan Wood

Associate Editor

Raines Cohen

Assistant Editors

Betsy Hambrecht

Kelley Wicks

Production

Tom Chavez

Michael Lamoureux

Linda Custer

Kee Nethery

Carl S. Di Giorgio

Ron Schmidt

BMUG Project Coordinators

Sam Roberts

Steve Costa

Ty Shipman

Michael Lamoureux

This newsletter is for distribution to the membership of the Berkeley Macintosh Users Group (BMUG), a registered student group at the University of California at Berkeley. Membership dues are \$15.00 per semester (plus postage and handling) which includes the printed newsletter and the current membership disk containing a selection of choice public domain software programs. For membership information see the back of this newsletter. The material appearing in this newsletter may not be reproduced, in whole or in part, for any commercial purpose without written permission from the Berkeley Macintosh Users Group. Articles appearing in the BMUG newsletter which are not specifically copyrighted by the author may be reproduced for non-profit distribution between individuals or within other non-commercial users groups **if and only if** proper citation is made to The Berkeley Macintosh Users Group (including BMUG's full name and address), and the BMUG newsletter issue in which the material appeared. Copies of any such reproductions must be sent to BMUG. To reproduce any article specifically copyrighted by its author, contact the author in care of BMUG at the above address to obtain permission. Opinions expressed herein do not necessarily represent those of BMUG or the University of California or its Affiliates. Throughout the newsletter, titles of trademarked software and hardware products are presented. All such proper names are assumed to be trademarks of their manufacturing companies.

Table of Contents

General

About BMUG		4
BMUG After One Year	Reese Jones	7
Format? What Format?	Dan Wood	11
Dialog Boxes You'll Never See on the Macintosh	Ted Cohn/Dan Wood	12
Carrying on a Weekly Tradition	Ron Schmidt	16
New and Future Products of Note	Reese Jones	19

Hardware

News and Reviews

Hardware notes	Reese Jones	25
Mice, cables, reliability, insurance, surge protectors, fans, carrying bags, the BMUG disk review project		
Hard Disk Update	Reese Jones	31
ImageWriter II Preview	Raines Cohen	40
Hard disk drives with Removable Cartridges	Paul Gusciora	41
Typewritten Pages to Mac Files	Scott Beamer	43
The MonsterMac Two-Megabyte System	John Sawyer	46
MacMEGABYTES One-Meg Upgrade	David L. Foster	49

Tutorials

MacRecorder	Ty Shipman	51
MacRecorder II	Michael Lamoureux	58
MacRecorder Software	Michael Lamoureux	61
How to Engorge Your Mac	Paul Campbell	66
A Homebrew Uninterruptible Power Supply for the Macintosh	Jay Z. James	71
BMUGNET/AppleTalk	Reese Jones	75

Software Reviews

MacModula-2	Michael La Belle	85
Gato & Fokker Triplane	John Heckendorn	88
TurboCharger	David Foster	91
Microsoft File	Matt Williams	93
Mind Prober & Communication Edge	Storm Jenkins	95
Trivia Phillip Suh		96
Learning Microsoft Multiplan & Learning Microsoft Chart	Storm Jenkins	97
Helix 2.0	David Foster	98
Macintosh Books for the Absolute Idiot	Mike McIlwrath	100
Voulez-Vous or Vouler-Vous? A review of Le Conjuguer	Kelley Wicks	103

Software Tutorials

Managing a Research Bibliography	Linda Custer	104
Making Color Photographs with The Mac	Carter C. Collins	106
Easy 35 mm Slide Production with a Mac	Brent Fultz	109
Loan Payment Spreadsheet:		
Amortization Calculations	Kee Nethery	111
Using The Font Librarian	Dave Richey	113
Pascal Illustrated Fearlessly, Loathlessly	Scott Kronick	115
68000 Minus Fear, Loathing	Scott Kronick	120
<i>Special Section: Using the Resource Editor</i>		
How to move the Reset Button from the side of your Mac to the Keyboard	Fred A. Huxham	124
Removing Help files from MacPaint	Linda Custer	128
Font Editor Supplement	Mike May	130
Creating Your Own Icon	Fred Huxham	134
Setting Up your Letterhead and Signature	Reese Jones	137
MACsimizing MacDraw	Dan Wood	146
Using MockWrite	Gordon Haig	152
Mac Programming Languages	Linda Custer	154
Getting the most out of your Trash Can	Dan Wood	158

Telecommunications

Accessing the BMUG BBS with Free Term	Carl DiGiorgio	162
Going "On-Line" for the first time	Grace Kang	167
Apple Develops Point and Click Telecommunications	Ken Wait	172
The Melvyl™ On-Line Catalog	Mitch Turitz	175
CompuServe Update	Linda Custer	176
Delphi Update	Raines Cohen	177
"Hello, this is the SYSOP..."	Raines Cohen	178

BMUG Choice Products

179

BMUG Pinup Pages

Caution!	185
The BMUG T-Shirt winners	186
Laser fonts	188
Customized MacDraw Menu Equivalents	192
Other Macintosh Users Groups	193
Macintosh Error Codes	198
Types & Creators	199
A Dozen Basic Things Every Mac User Should Know	200
Finding the BMUG Meetings	201
The Mouseketeer	202

BMUG Software

Public Domain and Shareware Software	203
Your Membership Disk	204
BMUG Disk Catalogue	207
BMUG Software Index	218
Order Form	220

About BMUG

Who We Are

BMUG is the Macintosh Users Group located at the University of California at Berkeley. This all-volunteer group is devoted to the exchange of information about the Apple Macintosh microcomputer and related products.

BMUG's membership consists of roughly 3000 people: students, faculty, and staff of U.C. Berkeley, Lawrence Berkeley and Livermore National Laboratories and other universities, as well as members from the surrounding community and all over the world.

General Meetings

BMUG general meetings are held every Thursday at 5:30 PM in PSL (Physical Sciences Lecture Hall) on the U.C. Campus, near the East gate of campus. (In the event we can't reserve PSL for some reason, the new location will be posted there.) Meetings consist of product demonstrations, tutorials, and/or presentations by invited guests, followed by open discussion, demonstrations, and question-and-answer sessions. Attendance is typically between 300 and 400 people. We try to keep the atmosphere informal, particularly towards the end of the meeting when many people linger to exchange ideas or look more closely at the demonstration equipment. The room is well-equipped for the meetings with a HUGE video projector (a Hughes Aircraft high resolution projector directly connected to a Mac) for viewing the Mac screen or video tapes. Meetings are always free and open to the public; you need not be a member to attend. New members may register before or after the meetings. Members may purchase BMUG distribution disks (\$6 each), blank Sony/Apple disks, or other BMUG products. We also distribute promotional material and handouts from software and hardware vendors at the meetings.

Help Lines

The BMUG help line is usually staffed during late morning and early afternoon hours on weekdays, (415) 849-9114. We also have a help line on the East Coast: (212) 925-7971. On Thursdays we usually turn the answering machine on to answer general questions about the evening meeting. We can answer your questions or refer you to someone who can, via **PhoneNet**, BMUG's user help network. Members who are familiar with a particular program or topic have volunteered to answer Macintosh questions over the phone during a particular time and day (such as Sunday 4 to 5 pm). The list is available to members; please call the BMUG office for a reference or send an S.A.S.E. to BMUG for a list of PhoneNet volunteers in your area. If you can help answer questions on a particular topic or program, please volunteer for the PhoneNet.

The BMUG Newsletter

The BMUG Newsletter is published each semester and contains news, product reviews, product recommendations, tutorials, and reference information of interest to the Mac user. The BMUG newsletter does not include advertisements, as we wish to maintain a high level of objectivity. Neither BMUG nor the University of California is responsible for any damage or loss resulting from the information or software distributed by BMUG.

The newsletter consists of material submitted by the membership. Material for publication should be submitted by MODEM or on a Macintosh disk as a MacWrite document, with accompanying MacPaint or MacDraw using a standard font (Geneva, New York, Times Roman, Helvetica). Disks will be returned with something neat on them. Legible hard copy is also acceptable--if the submission is short. Please contact us for a "writer's kit." Submissions don't have to be long. Contributors will be properly credited and articles may be copyrighted by the author. **Let's share what we've found out.** The Spring 1986 newsletter is already underway!

Beginners Group

The BMUG Beginners' SIG (Special Interest Group) currently meets every Tuesday at 5:30 pm in 200 Wheeler Hall on the Berkeley campus during the school year (call to make sure). The meetings are small and informal, and are tailored to the new Macintosh user who may be lost among the often technical presentations and gossip at the general meetings.

Fall 1985 BMUG G Newsletter

Developers' Special Interest Group & Newsletter

The BMUG Developers' SIG currently meets every Wednesday at 6:00 PM in 122 Wheeler on the U.C. campus during the school year. (Call BMUG to verify the location before attending.) The Developers' Group Spring 1985 Newsletter is available to BMUG members for \$10 each. It contains tutorials and examples of programs in Consulair C from the Spring 1985 meetings. BMUG disk #18, available separately, contains source code and compiled programs that supplement the Developers' Newsletter.

Bulletin Board Systems

BMUG currently has two BBSs: one in Berkeley and one in Manhattan, running 24 hours a day, seven days a week. We are currently running on Mouse Exchange software, although we are looking for a program with a more Mac-like interface. To use the BBS, dial (415) 849-BMUG (that's 849-2684 for you digital people) on the West coast or (212) 925-4630 on the East coast at either 1200 or 300 baud. The BBS will prompt you on what to do. Until you are validated, you will be given five minutes per call to explore the system and to write to the SYSOP asking for validation. Raines Cohen is BMUG's Berkeley BBS SYSOP; Sam Roberts is SYSOP for BMUG EAST. The SYSOP will check your membership and validated you within a couple of days.

When you are validated, you will have full access to mailboxes, news (including a rumors section run by Oliver Wendell Jones III, featuring the *juiciest* gossip you'll ever see), and chatting with a SYSOP (if he/she is around/awake). You can upload submissions (new software, newsletter submissions, etc.) or download files from our massive library, which has over 10 megabytes online.

If you try calling and the line is busy, don't despair! BMUG BBS is understandably popular and is frequently in use. We're looking into expanding into a multiple-line system; call or write if you can help.

MacRecorder Kit

BMUG offers a kit called MacRecorder II, which contains a speech digitizer and A/D converter. Cost is \$45 for the kit, which includes the circuit board, all parts, software, and instructions necessary to build the device. The microphone and a case are not included. The software comes as a double-clickable application; see "The MacRecorder Software" in this issue. MacForth source code is also included. See the articles by Ty Shipman and Michael Lamoreaux for more information.

BMUGNET Kits

BMUGNET is an AppleTalk™-compatible local area network which is inexpensive because it uses ordinary phone cables. It comes in two configurations: permanent wall mount (the circuitry fits in a standard telephone jack box) or mobile (double telephone jack box). BMUGNET costs \$15 per node. See Reese Jones's article for more information.

Referrals

We often get calls from companies and individuals looking for freelance programmers, consultants, or instructors to work on short- or long-term projects involving the Mac or to teach classes about the Mac. We maintain referral lists, both of openings and of available freelance specialists from our group. If you are a company or individual looking for a consultant, programmer, or instructor/tutor, or if you are a Mac programmer or instructor who looking for a project, let us know so that we may connect you with the appropriate company or person.

Public Domain Software

BMUG has the most up-to-date public domain software library around. Disks are currently available to members for \$6 each. See our detailed library listings at the back. We always offer the latest versions available, and we are constantly updating our software library. If you have any new software for the Macintosh, we welcome your submissions. If you've discovered a wonderful public domain or shareware program, have written one yourself, have created a wonderful template, or have drawn some interesting art, upload it to the BBS or send BMUG a disk. We will return a disk full of goodies! Our software library depends on you!

The BMUG T-Shirt

BMUG T-Shirts (design shown in the pinup pages section) are available for \$6 each, on 100% Cotton (not pre-shrunk) in M-L-XL-XXL in several colors. See the order form in the back.

Newsletter Back Issues

Members can purchase back issues of BMUG newsletters: The Spring '85 newsletter is \$10, as are extra copies of this newsletter. The Fall '84 newsletter is available in the form of several MacWrite documents on disk #1. The following is a summary of the Spring 1985 Newsletter:

Notes From the Outside World (a bibliography) • Memory Upgrades: 512K and Beyond • Doing Your Own 512K Upgrade the Dr. Dobbs' Way • **Floppy Disk Brands Reviewed** • Hard Disks for the Mac • IBM PC and CP/M compatible Co-Processors for the Mac • Making Your Own Cables • Daisywheel Printers • Installing MockTerminal • Downloading a File with MockTerminal • **Transferring Files from Other Computers into the Mac** • Near Letter-Quality Printing on the ImageWriter • The OKIDATA 92 printer • Printing Multi-color Images on the ImageWriter • Microsoft Word Reviewed • Factfinder Reviewed • Microsoft Chart Reviewed • **Five C-language Compilers Reviewed** • MacFORTRAN Compiler Reviewed • Three Pascals Reviewed • FEdit Reviewed • Using the microFinder with a RAMdisk • MacTerminal Reviewed • Customizing MacTerminal Keyboard • Red Ryder • SmoothTalker Reviewed • MacCheckers and MacVegas Reviewed • Sargon III Reviewed • **Tricks of MacPaint** • Customizing MacPaint with the Resource Editor • **Creating Your Own Fonts, a Tutorial** • Keyboard ASCII layout • Macintosh & MacPaint Keyboard commands • Macintosh Error Codes • BMUG Choice Products

MacFORTH Level 3

Creative Solutions, Inc., the company that designed MacFORTH, has kindly given BMUG a copy of their Level 3 upgrade. Level 3 allows programs developed in MacFORTH to be run without the full, proprietary MacFORTH kernel. Any BMUG member who has used their own copy of MacFORTH (Level 1 or 2) to develop an application that they would like to distribute as a **public domain (or shareware)** stand-alone application is welcome to contact Linda Custer at BMUG about using Level 3 to compile their application. If desired, we will provide help in converting the source code so that it is compatible with Level 3 and will work with you to produce a finished application.

In order to write code that will be compatible with Level 3, you should not require the end user to know or use any FORTH words. You should rely completely on the Macintosh interface for all input, since the MacFORTH dictionary is completely erased when processed through Level 3. Optimally, you should develop your application completely with Level 1 or 2, and then modify it only slightly for Level 3. Then use the resource editor or an icon editor to create icons and masks for your application and any files it creates. Lastly, create as many MacPaint screens as you want to show up as **About my application...** type menu items in the Apple menu of your application. Each screen should be exactly the size of the largest picture you can see at one time in MacPaint, and should be cut into a scrapbook file. Once you've gone that far, BMUG can show you how to do the rest.

If you have an application that is long and complex, requiring memory overlay capabilities at runtime, or other features of Level 3 with which we are not completely familiar, contact BMUG, and we may be able to work with you or lend you the documentation and software to see if you can work the details out yourself. We will ordinarily not let level 3 out to users.

Of course, if your application looks good when it's done, and if you would like, BMUG will be happy to add it to our public domain software collection (see above) and help you distribute it through various on-line services and software exchanges between user groups.

BMUG after One Year

By Reese M. Jones

BMUG is now just over a year old, and has grown to nearly 3000 members spread all over the world. Our weekly meeting has increased to 3 meetings a week (General meetings on Thursdays, Developers SIG on Wednesdays, and Beginners SIG on Tuesdays). Our general meetings are typically attended by 300-400 people.

We have established 2 bulletin board systems (in Berkeley and New York). The BBS systems have become tremendously popular and are being used almost continuously for 24 hrs a day; as soon as one user hangs up, the next is ringing. Our BBS systems have become a source for the very latest information, news and software for the Mac user. We maintain an active presence on each of the major public access computer services and have established special interest areas on *Compuserve*, *The Well*, *Delphi*, and other large systems. The volume of mail and number of phone calls BMUG now receives daily have gone far beyond any conceivable expectations of a year ago.

This excessive demand for the types of services BMUG provides suggests various things about the computer industry and computer education at the University level. There is a tremendous need for a source of objective and realistic advice to the user and to the potential computer user. Most common sources of information have little incentive to remain unbiased. The dealers try to sell you the product that's most profitable for them, while magazines tailor their reviews to accommodate their advertisers, and high-priced consultants distort information to make it sound so complicated and mysterious that you couldn't possibly get along without them (you don't need consultants to buy or use a car, stereo, or refrigerator). Additionally, there are the online computer services which want to charge you for as much logon time as possible.

Some types of information resources, for instance, commercial magazines, try to be objective and unbiased, but often remove the dynamic range of their reviews (so that no one is offended). Another type of information resource, the dealer, simply is trying to sell you something. The independent consultant will present his/her educated perspective (at considerable cost) but should make it clear that it is just that - *his perspective* - and should report all of the factors he considered. When the word *custom* comes up, watch out. Commercially available products can in most cases do the job and will be up and running much faster and cheaper than custom solutions. If you hear **lots of jargon** - you're wasting your time and money.

This general lack of an honest and objective information service and the myriad false and distorted presentations made by dealers, consultants and commercial magazines have led to an atmosphere of confusion and mistrust in the microcomputer market place. Typical of today's market place is the customer, who, at his consultant's advice and after seeing a flashy ad in a magazine, goes to a flashy dealer and buys X product at **list** price, thinking that is the only alternative and assuming that he is paying for after-sales support. He takes it home, opens the paper and sees the same product advertised for a thousand dollars less at a discount dealer in the same town. He thinks "Oh well, I'm getting it from a dealer that will provide support." Then he goes to use the computer for his intended task and finds it difficult or seemingly impossible to use. He calls the dealer, they aren't very friendly (they have already got his money) and they can't really help because they have never even touched the keyboard, much less ever used the expensive speciality program they just sold him. The dealer then suggests he call the same consultant that recommended the dealer. In frustration, the consumer calls the consultant, who asks for his account number when he answers the phone, the consultant then informs him that the speciality program he just paid for will never solve his problem (although the consultant has never tried the program) and that the consultant could write a custom program that would get the job done, for a large fee of course because the application is so complex that only the consultant could possibly understand it. The customer throws his computer and consultant out the window and tries to catch up on a month lost not balancing his checkbook. This scenario is indicative of the microcomputer industry and explains why word-of-mouth through friends is the way most satisfied computer owners decided which brand of computer to buy. This trustworthy information void creates a need for the types of services that BMUG attempts to provide.

The information presented by BMUG is definitely (and intentionally) **biased** but we aren't selling anything (except perhaps the information itself or our views of the world or ourselves, but hey, we're just doing it for fun anyway). We try to provide through BMUG the information that we wanted to find out for ourselves, but couldn't, without help.

Fall 1985 BMUG G Newsletter

All of us use the Mac numerous hours **every day** and are often faced with problems like: how to do a particular real world task given the software that is available. Or, which hard disk should I buy for myself, or how do I make the cable so that I can transfer my files out of that old computer I used to use, and how do I actually do it after I get the cable. The information we provide is meant to be **technically adequate - it works OK** - and that's what is important to those of us who use the Mac to do real life work and consider our time to be valuable. It is easy to criticize and difficult to create. Any information distributed by BMUG (at the meetings, in the newsletters, on the BBSs or through other channels) may be wrong and is definitely biased, but atleast it is sometimes useful.

Our views may seem to be heavily biased towards the Mac. For many of us, however, the Mac is the 3rd or 5th computer we have owned (or still own); and if another general purpose computer comes out that is **a lot** better we'll be among the first to switch. The Amiga, Atari ST, AT&T UNIX, suped up IBM AT or anything else currently available definitely not sufficient motivation to switch... What will be? The new SUN workstation? Maybe.

Things in the Mac world have obviously changed over the last year and a half as well, clearly distinguishing the Mac as the best general purpose micro computer available for those planning to use their computer for a variety of things and the Mac is also becoming the best micro for several special-purpose applications (particularly spreadsheets, printed graphics and page layout). The software and hardware has gotten better and much more abundant.

The closed hardware architecture that the uninformed still complain about was proven to be imaginary by the numerous companies offering add-on multi-megabyte memory upgrades, internal DMA hard disks, video output ports, RGB and NTSC **color** video interfaces, high resolution color graphics, coprocessors, array processors, LAN hardware and software (that really works to do real work!), A/D-D/A and digital I/O interfaces, modems, color plotters, graphics capable laser printers, Photo Typesetters, and so on. Many don't realize that the Mac offers all of this hardware expandibility while maintaining the incredibly important **standard user interface** and maintaining superior software compatibility across all possible hardware configurations. This combination is found nowhere among IBM's, Apple IIs, VAXen, and all other "*open*" machines, namely "those with slots."

With all these improvements, the Mac is becoming a much more productive tool for real world jobs (pun intended). A good example of this advancement is the contrast in the ease of production of the BMUG newsletters.

The Fall 1984 BMUG newsletter was written with MacWrite 2.2 and MacPaint 1.0 on a single drive 128K Mac with the aid of a single partition Davong (RIP) hard drive. The newsletter was printed on the Imagewriter (one) for paper distribution and for distribution on disk, separated into 5 files (because of the MacWrite limit of 5-6 pages/file on the 128K Mac).

The Spring 1985 BMUG newsletter was written in an early beta-test versions (3.987) of the disk based version of MacWrite (allowing ~50 pages/file) along with the 1.5 version of MacPaint and an early version of MacDraw was use for the cover only. At this point we were using 512K Macs and a Tecmar Hard disk (also RIP) that with experimental software allowed for partitioning and print-spooling. We printed all the rough drafts on an Imagewriter and considered distributing the Spring NL on disk also, but when we went to print out the final draft, it was >600K and took more than 5 hours to print on the Imagewriter, so we decided to distribute it on paper (not as high-tech, but imagine 3000 people spending 5 hours each to print out their own copy).

About this time the Apple LaserWriter was announced, but, as usual, you couldn't find or buy one anywhere. We decided to try and print it at Apple on one of their early LaserWriters (with beta test software). We plunged through the numerous bugs, system crashes, and problems we expected using beta test software under a beta test operating system, on beta test hardware, preparing a file to print on an unreleased printer that we had never tried before. We packed up our hard disk and trekked over to Apple and ,after numerous additional hours of re-formatting, bug-avoidance maneuvers, and minor mistakes, we finally got a draft printed out.

The cover of the Spring NL had been done entirely in MacPaint (which couldn't print on the version of the laserwriter software we were using at the time) but since we still had 10 minutes access to a LaserWriter remaining, we wanted to take advantage of the nice high resolution ROM-based fonts for the title. We used the Thunderscanner software to copy our MacPaint cover into MacDraw and then went through several of the early versions of MacDraw on hand until we found one that could both accept bitmaps and handle the font properly. We chose **Print**, and, 5 minutes later, the

Fall 1985 BMUG G Newsletter

LaserWriter spit out a blank page. But the processing light continued to flash as our hosts were standing around waiting for us to get out, and, finally, out came the cover. So with a little liquid paper, and some cut and paste, the Spring NL was off to the copy shop.

Producing this Fall 1985 BMUG Newsletter has been totally different kind of experience than putting together the previous publications. The hardware and software limitations are no longer a major problem in the production process. The software is much more reliable, more powerful, and most of the bugs have been worked out (or we have learned to work around the bugs!). More software tools like the MacWrite4.5, MacPaint1.5, Paint Cutter, and MacDraw are available, as well as Switcher to easily integrate them in a large-memory machine. We have since acquired a LaserWriter, several large memory Macs, a Hyperdrive Mac, MacXL 2/10 (with 1 megabyte of RAM and an internal hard disk, the central place where the newsletter articles were finally compiled), and wired up the whole house with AppleTalk (and BMUGNET). With this setup the primary goal of the integrated office is realized: where several people can easily work together on the same large project, and when other people bring their Macs to help out, they can easily be plugged into the network as needed. The Mac is becoming a very powerful tool transparent to the job at hand.

Public Access Computing Facilities

Several new Macintosh/LaserWriter based *Public Access Computing* service businesses have opened in the Berkeley area. If you want to produce resumes, documents, or graphic artwork, these new businesses fill the gap between self-service copy shops and full-service printers, graphic art shops, typesetters, and publishers.

Several copy centers have installed a floppy-based Macintosh hooked to a LaserWriter where you can bring in your own floppy disk and pay an hourly usage fee and a charge per page to print them on the LaserWriter (**Krishna Copy Center** (415) 540-5959, **Cleo's Copy Center** (415) 843-6000, **Tolman Micro Facility** (UCB students only)).

Another new business is a combination self/full-service Macintosh Work Center (**William Tell Computing** (415) 849-9993) where you can go and rent time on any of several Macintoshes networked over AppleTalk to a LaserWriter (printing included, no "per page" charge), all running off a large hard disk/file server with all of the major programs accessible by each machine (programs like: Excel, Word, PageMaker, Spellcheckers, Statistics, Database and other expensive specialty software products). You can also use a variety of speciality devices at W.T., like a Video Digitizer, Modems (for downloading your files from another computer like UNIX) and soon a high resolution photo typesetter that you will be able to print your Mac files on. The store has someone on staff to provide help as necessary and also offers some additional services like training/tutoring, word processing, database work, slide production, and file transfers from other computer's floppy disks to and from Mac floppy disks. You can sign up for BMUG membership at William Tell Computing and BMUG members can obtain our disks and newsletters there.

There is also new Full Service laser printing company in Oakland (**The Lasers Edge** 835-1581) that will take your Mac disk (or IBM-PC or KayPro disk) and print your files on a laser printer for \$1.00 per page.

Rent-a-Mac in San Francisco (415) 824-7740 and San Jose (408) 292-1292 will rent you a Mac system by the day, week or month that you can use at home.

There is now Apple credit available through the ASUC Student Union Computer Store for UC Berkeley students to **buy a Mac on credit**. You can also apply for the Apple credit card, available through any authorized Apple dealer, that allows you to buy any Apple product on credit (at relatively high interest rates).

Starting a Local Users Group

One of the most valuable aspects of BMUG are the weekly meetings. At these meetings timely information of interest is presented in a personalized manner with enough flexibility to have extended open discussion. At the end of the meeting, you can ask specialized questions, see and play with the products that were demonstrated or meet and talk with other people who are working with the same types of applications that you are. For those who can't make it to the BMUG meetings in Berkeley, and don't already have access to any local users group, we suggest that you start one. Here are some suggestions regarding starting and maintaining a users group based on our experience. You will really find it is well worth your time and effort to meet with other Mac users in your area.

- Get together with a couple of friends in your company, community, or school that own Macs that you think might be interested in helping.

Fall 1985 BMU G Newsletter

- The primary goal of a users group is to promote the exchange of information. Try to define an orientation for the group and the type of information to be exchanged (eg computing for handicapped people, architecture, thesis writing, or general use of the Mac)
- Write up a very simple constitution or set of by laws (we will send you ours as an example if you request it)
- Open a checking account for the group in order to separate the finances of the group from your own
- Get a federal tax I.D. number (you usually get one when you open a business account. Ask the bank.)
- Set-up and keep good financial records
- Establish a special phone number for the group, preferably with an answering machine and call forwarding.
- Establish a post box for a **permanent** mailing address, not your home address if you value your privacy.
- Try to set up regular meetings (at the same time of the month or week) in the same meeting place if possible. Schools and local colleges will probably have meeting facilities available and even welcome your presence. You might try having your first few meetings at someone's house.
- Post notices at local hangouts, laundry mats, dealers, and local businesses.
- Try to always have at least one Mac at the meetings for product demonstrations.
- Keep the meetings as informal and unstructured as possible. Don't worry about getting a speaker and don't invest a lot of time preparing a presentation unless you are very comfortable with public speaking.
- Schedule time for informal discussion after the meeting, simple BYO refreshments are good way to lighten things up a little.
- Try to get local dealers to support the group, it will promote sales and relieve their need to support users.
- Encourage members bring their friends, and **don't charge people** for attending the meetings.
- A newsletter is a lot of work and not necessary to make a good group.
- Try to establish a group PD software library. Many small groups make a *disk of the month* that is copied by anyone who wants it at the meeting. Take special care to discourage software piracy, and educate members about the software piracy issue and how it hurts us all.
- If several members subscribe to computer related newsletters, and throw them away after reading them once (as they should) encourage them to donate their magazines to a group library. The library could be maintained by a particular member and limited check-outs could be arranged at meetings.
- Remember a users group has many social aspects and benefits of a club.
- Try to have regular goal-planning meetings with the core active members.
- Don't **waste** a lot of time with organization, elections, titles, structure, etc. Spend your efforts on exchanging information, as we do, and the group will be much better.
- Don't try to make money by starting a users group. Your rewards will come in other ways and in due time.
- Call BMUG! If you need any help getting started, we will gladly provide it.

Format? What Format?

The Guts of the Fall '85 Newsletter

By Dan Wood

We get many questions about how the BMUG Newsletters were written, formatted, printed, and published. This newsletter was produced almost entirely on the Macintosh without using any special software. The backbone of the newsletter was good ol' MacWrite 4.5 (with the Assimilation Process *Mac•Spell•Right*), with much help from MacDraw, MacPaint, Switcher, and either Bill Atkinson's *Paint Copier* program (available on BMUG disk #23) or Silicon Beach Software's *Paint Cutter*. The newsletter was compiled from individual articles and pictures, submitted by the authors on disk or by modem, producing several large MacWrite files. The articles were loaded onto a Macintosh XL 2/10 and converted to 10-point Times Roman, with bits of Helvetica and Courier. The 1 megabyte RAM of the XL was extremely helpful, although not entirely necessary for production.

The header used throughout the newsletter was created in MacDraw using layers of grey patterns with shadowed Helvetica bold oblique text on top. It was then pasted into the Header field of each MacWrite document. The Newsletter was put together in several 10-40 page Macwrite documents for safety, speed, and convenience. The titles for each article were stretched by typing the title in MacDraw (in Times Roman 24 pt. bold), selecting the field, cutting, switching to MacWrite, and pasting. In MacWrite they were then stretched to various dimensions. Using this method, it's possible to expand any LaserWriter font to any imaginable size, proportion, or distortion, while using the ROM-based LaserWriter fonts.

Most diagrams in the articles are MacPaint images, primarily screen dumps. If the figures were to be shown actual size (as in "Font Editor Supplement") they could be pasted into the Macwrite file directly from MacPaint. Usually, however, the MacPaint images were pasted into MacDraw, shrunken (keeping the correct proportions as outlined in "MACsimizing MacDraw") and then pasted into MacWrite. Two MacPaint images could be placed side by side by using MacDraw. See "Creating Your Own Icon" for an example. The pictures are easy to read because of the smoothing that the LaserWriter performs on MacPaint images (example: "MacRecorder"). Other diagrams were created in MacDraw ("MacRecorder II," the table in "Going OnLine...", etc.). Some diagrams are hybrid: "Easy 35mm slide..." uses text in MacDraw over the MacPaint image. Figure 8 on "How to Engorge your Mac" adds text fields to the side of the picture, an impossible feat when using only MacWrite and MacPaint. Most articles were printed with smoothing; a few, however, were left unsmoothed to show square dots on the MacPaint images when printed. Examples are "MACsimizing MacDraw" and the picture of the old church after "Mac Programming Languages."

A few sections were printed from MacDraw. In the MacWorld parody, "Getting the most out of your Trash Can," the text was entered in MacWrite, then pasted in as paragraph text. Pictures were drawn in MacDraw or MacPaint, shrinking or expanding as needed. The Madonna picture was scanned with Thunderscan and shrunken fifty percent or so. The cover was printed in MacDraw in two passes (by running through the LaserWriter twice): the top half, consisting of the lettering, the reduced Campanile, and the ribbon, was printed with enlargement for the large letters; the bottom half, consisting of the lettering and a MacVision-digitized picture (of Kee Nethery's hand on a mouse) was printed normal size.

The pinup pages were the most difficult to produce. The Font charts, for example, were created by printing template pages then pasting (by hand!) the copy that Adobe sent us. For the "Caution" picture, we pasted the picture from MacPaint into MacWrite, using *Paint Copier*, and pasted the text from MacDraw, expanding in MacWrite. The T-shirt picture was printed in two passes: first by printing a page with just the header/footer and then by printing the MacDraw document on the same page. This was done because the document is reduced 50% in order to achieve thinner lines. Several sections were also printed in two passes to keep the header and footer. The Forth source code for MacRecorder II was printed, using Microsoft Word, over a blank header/footer page from MacWrite in order to get several columns. The photos for "A Homebrew U.P.S...." were digitized with MacVision, laid out in MacDraw (reduced 50%) and printed over a header/footer sheet. The multi-column disk index was printed using OverVUE over a header/footer page.

Fall 1985 BMU G Newsletter

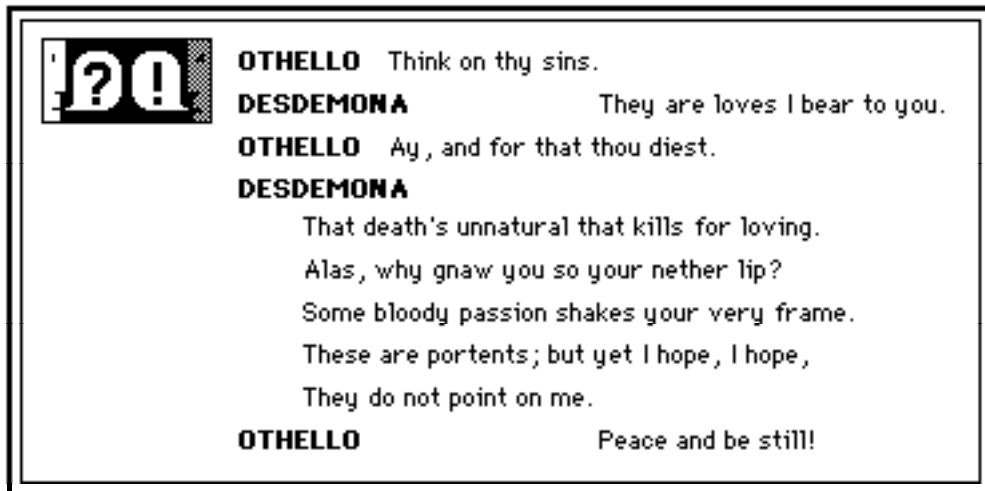
Now you, too, can make your own Newsletter!

Dialog Boxes You'll Never See On The Macintosh

© 1984, 1985 by Dan Wood and Ted Cohn

While rumours persist that Apple Computer is setting up a Macintosh Consortium Program for third-world nations and members of the Teamster's Union, Professors Tedintosh and Macindan are on the move probing Mac mysteries. Recently, after months of prying and snooping (and gobs of illicit coercion), we have acquired several unreleased prototypes of Macintosh dialog boxes from someone at Apple whose identifier shall remain anonymous for microseconds to come. These boxes represent state-of-the-art technology in user-friendliness. But, to our great surprise, the Mac Development Team rejected these dialogs just before Macintosh production began. We can only surmise that these dialogs were trashed on the basis of being too frank. One of our studies shows that humans in this high-tech, high-minded, high-resolution, high-protein, high-octane, high-strung and generally high society, often become uncomfortable and defensive when confronted with the truth. This, most likely, is the rationale for Apple's impulsive decision. Thus, the boring dialogs you normally see have been toned down for the average user. But we know that you, as faithful BMUGgers, can handle a sampling of Apple material in its virgin form.

Ted Cohn and Dan Wood are students at U.C. Berkeley. They are currently working on a project to prove that the signatures on the inside of the Macintosh case are forged.







OK

Hi, I'm Steven Jobs. Sorry to interrupt your MacWrite session, but I just wanted to let you know that I'm the creative force behind the Macintosh you are now using so well... Catch you later on some other application!



Do you think I ask too many questions?

never




We're sorry; you have reached an application that is disconnected or is no longer in service.



RED ALERT



The Macintosh has been surrounded by  personal computers. What shall we do, Captain?

PANIC!

Use corbomite device recently installed



You know, you have really been overworking the Macintosh this last week. The computer needs a long vacation NOW.

ALL MAC activities must be postponed; the current disk has been erased. Choose now where the Mac will vacation.

Virgin Islands

Tahiti

Hawaii



Congratulations! You have just succeeded in **destroying your Macintosh! Take a close look at the screen, because this the last picture it will ever display. Come to think of it, take a close look around. You see, the Macintosh is leaking radiation all over you...**

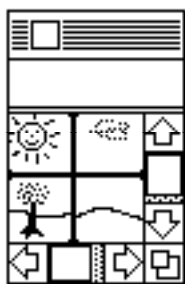
Restart

Resume

You know, I really enjoy working with users like you. You light up my phosphors every single day and feed me yummy disks with yummy software on them. I don't mean to be critical, but you are the worst typist I have ever seen, I mean felt! Oh, also, I'm really getting bored of this Multiplan garbage. Would you mind putting in MacPaint so we can draw some pretty pictures? Thanks.

OKAY...

Aww, shut up



STUCK WINDOW ERROR ID = 23

Ah, excuse me, I don't mean to bug you, but I'm having a bit of trouble opening this window. It seems to be jammed. What do you think I ought to do? Again, I'm really sorry to disturb you. Please forgive me...

Keep trying

Get the WD-40

Sledgehammer

DANGER DANGER

This operation might **DESTROY** the Macintosh.
Are you sure you want to spin the disk drives
at *90,125* revolutions per second?

YES

of course!!

sure thing



Attention: you have just invoked error number 642, which means that you will not be able to recover all files used in the font manager unless they have been set to an ID less than 1024 by the resource mover which has been modified under the previous Quickdraw managers by the event interrupt handler system file finder disk which has only 200K available under the current file protection scheme.



You dingbat, you're a complete washup. You just succeeded in ruining your ENTIRE application.

You did such a stupid thing. You know you weren't supposed to press that key.

Macintosh knows how to deal with people like you. Please select the punishment you'd prefer.

erase disk

electrocute

void warranty



OK

Hi there, it's me again! Sorry to interrupt your MacSlots session, but I was noticing that you've been betting less than you ought to be. Try betting more at each whack. Remember, I know what I'm talking about because the Macintosh is *My* creation! I don't care what John Scully says!

Carrying on a Weekly Tradition

Running a Macintosh Users Group

by Ronald A. Schmidt

What comes to mind when you think of BMUG: The award-winning Spring '85 newsletter? The informative and entertaining weekly meetings on the University of California campus? The most up-to-date public domain software disks around? The BMUG BBS?

What doesn't come to mind are the thousands of hours of creative energy donated by BMUG volunteers which enable this users group to provide members with a valuable and necessary Macintosh resource at the lowest possible cost. Take a look at what goes behind the scenes at the world's largest non-commercial, non-profit Macintosh users group.

The BMUG Disks: BMUG has assembled over two dozen unique and up-to-date public domain disks which are distributed to members at a low cost. BMUG disks include utilities, desk accessories, games, templates, graphics, programming aids, tutorials, etc.

Behind the Scenes: Public Domain (PD) software comes from many sources. Many of the PD & Freeware programs are collected and submitted by members. Some are downloaded from CompuServe, Delphi, the Source, the Well, and other bulletin boards. Some PD software is obtained from other user groups. Many original PD software programs are produced by BMUG programmers. Some copyrighted software distributed by Apple, and others, is put onto BMUG disks only after BMUG requests and receives special permission to distribute it.

Each PD software program is evaluated and tested, and the best programs are organized and jam-packed on a BMUG disk. BMUG has made a special effort to keep the number of disks in the library down to a minimum. The BMUG disk library is generally organized by topic. Each BMUG disk roughly follows a theme and is assigned a number. A new BMUG disk is added to the library when volunteers collect sufficient *new* PD software of one topic to fill up a disk. BMUG disks do not contain system folders. We update or replace the software on each disk as newer versions of programs come along.

BMUG disks are copied with a Mountain Computer disk duplicator and one member's Mac. The duplicator saves volunteers time, not to mention wear and tear on their own Macs. After duplication, custom designed diskette labels are attached to each disk by hand, then stamped with the appropriate disk number and stored. BMUG keeps from 100 to 1000 ready-to-distribute BMUG disks available.

Blank Disk Sales: Each week BMUG sells 300-500 blank disks to members at the general meetings.

Behind the Scenes: The BMUG staff is constantly working with distributors to get low prices. Disks are ordered in lots of 2000. BMUG sells only selected brands which members have thoroughly tested for reliability. The price BMUG charges for the disks reflects the sum of the cost of the disks, shipping, tax and related expenses.

Guest Speakers: BMUG holds weekly general meetings in the Physical Science Lecture Hall on the U.C. Berkeley campus which feature a variety of guest speakers. Presentations include new software and hardware demonstrations, tutorials, commentary and lively question and answer discussions.

Behind the Scenes: BMUG hears about potential speakers through word of mouth, press releases, trade shows, personal contacts, magazines, etc. BMUG contacts the speakers, reserves a date, and arranges any necessary equipment. BMUG welcomes ideas for presentations.

Scheduling Meetings and Events: BMUG currently holds three types of weekly meetings: General, Developers, and Beginners meetings. All three weekly meetings are held on the Berkeley campus. General meetings are usually held in the Physical Science Lecture Hall (1 PSL), which features a University owned state-of-the-art video projection system.

Beginners Group: The beginners group meets weekly and holds sessions on the basic Mac applications.

Fall 1985 BMUG G Newsletter

Behind the Scenes: We are always searching for volunteers familiar with Mac programs.

Developers Group: Meets every week to work on collaborative projects and to exchange programming ideas. Meeting notes are compiled and edited to produce the Developers Group Newsletter.

Macintosh Gossip: News about Macintosh and the computer industry are shared at the meetings.

Behind the Scenes: Volunteers wade through the volumes of computer journals and other publications, sign onto bulletin boards and join conferences, attend user group meetings and computer trade shows, read press releases, hang around computer stores and talk with key industry employees.

BMUG Newsletter—Behind the Scenes: Creating the award-winning Newsletter involves hundreds of hours of volunteer effort. Articles are collected from members all over the world. BMUG volunteer members write most of the Newsletter articles. The editors and production assistants also do a great deal of writing. Newsletter articles are grouped into similar topics to conform to the planned outline. The Newsletter also contains selected reprints from other published and unpublished sources (with permission). We try to write about and collect the information we find most useful.

The newsletter draft is printed on a member's Laserwriter (as is most other BMUG paperwork), then proofread for errors. The final draft is also printed on the Laserwriter, then copied, using a high speed photo duplicating machine, and bound in large quantities.

The completed Newsletters, along with membership disks, are distributed at the General meetings and bulk mailed all over the world. Complementary issues are mailed to other Macintosh user groups (in exchange for their newsletter), selected industry individuals, and to major Macintosh-related publications for review, evaluation, and information exchange.

Registering New Members—Behind the Scenes: Volunteers designed the BMUG sign-up form using MacDraw and printed it on the Laserwriter. A volunteers designed the membership card, printed it on the LaserWriter, copied it onto acetate, and cut each card with a bulk slicer.

BMUG receives completed sign-up forms at meetings and by mail. Paid sign-up forms are dated and entered into the membership database, currently in OverVUE, Version 2.0b. BMUG uses the membership database only for non-commercial purposes: to send out notices, mail newsletters, and tabulate membership statistics.

BMUG Bulletin Board: Members can dial (415) 849-BMUG on the West Coast, or (212) 925-4630 on the East Coast to read/post messages, exchange mail, upload/download public domain software, and access other bulletin board features.

Behind the Scenes: BMUG's Berkeley BBS operates on a 1 Megabyte Macintosh, Hayes SmartModem, and a dedicated hard disk drive. The bulletin board requires daily attention by the system operators. Maintenance includes uploading current public domain software, authorizing new users, monitoring and troubleshooting the system, and participating in online conversations with callers. The SYSOPs are also currently working on a project to link the news and mail on BMUG's East and West coast BBS systems.

What else goes on at the BMUG office? Some of the tasks volunteers tackle:

Opening Mail and Answering the Phones: BMUG receives dozens of letters, packages, parcels, and phone calls daily. About half of the mail consists of membership requests and/or BMUG disk orders. The other half consists of a wide assortment of correspondence: software and hardware for evaluation, newsletters from other user groups, press releases, member inquiries, advertisements, magazines and journals. The helpline is staffed from 10-2 (Pacific Time).

BMUG staffers spend many hours opening, sorting and processing the daily mail. Here's how some of it is processed:

Press Releases: Taken to meetings and announced. **Membership Requests:** Processed as soon as possible and entered into the membership database. **BMUG Disk Requests:** Processed immediately. **Invoices, Bills:** Paid as soon

Fall 1985 BMUG G Newsletter

as possible. **Software and Hardware:** Tested, evaluated, returned (if requested). **Magazines and Newsletters:** Read, then filed in the BMUG library. **Inquiries, Other Mail:** Processed individually.

Computer Shows: BMUG participates in the major Macintosh-related shows nationally, and in selected computer trade shows in the San Francisco Bay Area. For example, BMUG staffed booths at the San Francisco MacWorld Expo, Boston MacWorld Expo and West Coast Computer Faire.

Preparations for each computer show begin with registration months in advance. Arrangements for the booth include electricity, ordering tables and chairs, and making booth decorations (one example is 3 X 3 foot giant BMUG disk designed by Dan Wood, Carol Cohen, and Raines Cohen for the Boston MacWorld Expo). Volunteers must also arrange for the following: several Macs, a VCR, thousands of blank disks, BMUG disks, current and past newsletters, flyers, and demonstration models of BMUG hardware products. Trade shows provide BMUG with the opportunity to make contacts with user groups, developers, vendors, distributors, press people, and many interesting individuals.

Here's a glimpse of what volunteers put into the Boston MacWorld Expo:

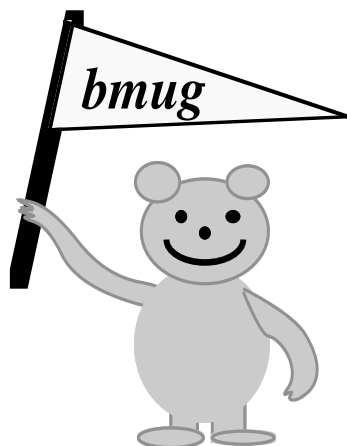
Several volunteers flew and stayed in Boston for a week at their own expense. BMUG literally set up headquarters at one member's home. Several carloads of BMUG equipment and materials were shipped from Berkeley. Volunteers spent many hours copying BMUG disks. Volunteers also found time to film a 20-minute video during the Expo which, thanks to overnight mail, was shown at a general meeting in Berkeley the next evening.

Other BMUG Products

Volunteers have recently developed kits for two creative Mac accessories: BMUGNET, an inexpensive alternative to AppleTalk; and MacRecorder, hardware and software which enables a Mac to record voice or any other sound. BMUG encourages development of innovative hardware and software.

A special thank you to the following volunteers who have done so much for BMUG in the recent past:

Dave Burnard, Tom Chavez, Annie Cohen, Carol Cohen, Jerry Cohen, Raines Cohen, Steve Costa, Linda Custer, Carl Di Giorgio, Dave DeNuzzo, Rick Eisner, Betsy Hambrecht, Fred Huxham, Reese Jones, Janice Kang, Mike Lamoureux, Tim Morris, Kee Nethery, Sam Roberts, Ty Shipman, Jim Takatsuka, Kelley Wicks, and Dan Wood.



New & Future Products of Note

By Reese M. Jones

On September 17th, Apple announced several new products for the Macintosh and Apple II lines. The announcement was overshadowed in the news by Jobs's concurrent announcement that he would leave Apple. Most of the products that were just introduced are specifically for the Apple //e and //c (like a 1 meg upgrade, double-sided 3.5 inch floppies for the //, and some nice composite video color monitors for the //'s). Here are a few notes about the new products that the Macintosh can take advantage of:

The Apple 1200 Modem

The new modem has an interesting design. It is quite a bit smaller than most of the other modems on the market and is designed to plug into a wall socket with a long cable going from the modem to your Mac's phone port. This new product is supposed to be **truly** Hayes compatible, unlike the current Apple Modem by USRobotics which is **almost** completely Hayes compatible. It also has two phone plugs so you can plug the modem to the wall phone connector and then plug the phone into the modem (unlike many modems).

One problem I see with this modem is its list price of \$400. You can get a USR for \$175 discount. Another problem is that if you want to plug your modem into a power strip, a multi-plug surge protector, or an old non-grounded wall socket, the design of the modem will cause you problems. Lastly, Apple has used yet another type of plug for the modem interface. The round five pin connector they chose is non-standard and will be difficult to find if you want to make your own cable.

Apple Hard Disk 20

Apple finally released its personal hard disk for the Mac at the surprisingly low price of \$1495 for **20 megabytes** (~\$1000 university/discount price--considerably cheaper than any other drive on the market). The drive connects through the external drive port (at 500 kbaud). It also has a second drive port on its back for plugging in your external floppy drive (which you will no longer need) or for daisy-chaining more hard drives to get additional storage. To use the hard disk, you need the new hierarchical file system, so the drive comes with a new system, finder and HD drivers. You can convert your files from Finder 4.1-used floppy disks to the new Finder/file structure, but not the reverse. The HD20 is about 3 inches high and fits right underneath the Mac. The drive seems relatively fast and is easy to set up and use. The HD20 improves both the capabilities and ease of use of the Mac and is essential for anyone who uses their Mac extensively.

The HD20 is a little noisy, but no more than other 3.5 inch hard drives for the Mac. One disadvantage of the HD20 is that the cable supplied with the drive is relatively short, so you can't put it on the floor or in the closet to minimize the noise like you can with other brands. The noise from the drive is relatively unobtrusive (nothing compared to the LaserWriter, and less than the sound of your floppy drive running all the time). One anticipated problem is potential incompatibilities between existing software and the new file structure. We don't have enough experience with this drive to say more yet. One of the nice features of the Hyperdrive that is conspicuously lacking in the HD20 is the ability to add security passwords to different partitions on the hard drive. Also lacking is a backup utility for handling single files larger than 400K or quickly backing up all files modified after a particular date.

The ImageWriter II

see Raines Cohen's article on this product in the **Hardware News & Reviews** section.

Other new products of note recently released by third parties:

MacCharlie

The MacCharlie IBM PC-compatible coprocessor for the Mac is finally available in the stores. MacCharlie allows you to run IBM PC programs on your Mac screen and keyboard. It sits along side the Mac and has its own dual 5 1/4 inch floppy drives, 640K memory, CPU, and an overlay for the Mac's keyboard that adds the IBM function keys. The MacCharlie seems to work fine, but you can't access any of the graphics features on the Mac screen or use the Mac's mouse to make the mouse commands in some IBM software. It will run Lotus1-2-3 but you cannot see any plots on the

screen. The most useful application for MacCharlie is its file transfer capabilities: to read your data files off of your old IBM PC floppy disks into the Mac, where you would then convert the files to MacWrite, MSWord, or Excel for use on the Mac. If your work requires you to prepare files in a particular PC format (Wordstar, 1-2-3, etc.) you could use the MacCharlie to go the other way. The MacCharlie costs around \$1800. I'm not sure that it offers any advantages over buying a used PC of new PC compatible for ~\$1000 and transferring your files back and forth through the serial port with MockTerminal on the Mac (as described in detail in the Spring '85 BMUG newsletter). Buying a regular PC would allow you to use the PC graphics if you needed them, plug in expansion cards, and make use of some of the hardware dependent, multi-format disk reading programs to handle 5 1/4" floppies from almost any computer. (Examples include MediaMaster, \$39.95, from Spectre Tech [(818)716-1655] to read, write, and format everything but Apple II disks, and the AppleTurnover card, \$149.00, from Vertex systems (213) 938-0857 for read, write, and format hard sector Apple II disks on a PC. Both of these products are very hardware-sensitive and probably won't run on MacCharlie.)

The Bech-Teck Chromatron

A new product is available from Bech-Teck (in Berkeley at (415) 548-4054) that converts the Mac's video output signal to an NTSC composite video signal that can be viewed in **COLOR on a standard TV**, recorded on a VCR or mixed with other video signals. The color is produced by an interaction between the pixel patterns and the way color is produced on a TV (methods similar to those used on the Apple II). The process works with any Mac software, and different color patterns can easily be defined by the user in any program that allows pattern editing like MacPaint, by using the resource editor. The Chromatron is expensive at \$2900, but should be considered by anyone doing video work with their Mac, interactive video, video disk work, video projection of the Mac screen for meetings or demonstrations, etc. The combined price of the Chromatron and a normal video projector would be much less than a high scan rate projector capable of handling the Mac screen directly.

Aldus PageMaker

PageMaker is the best of the page layout/publishing tools currently available, but is just a hint of what's coming over the next year or so. PageMaker is like a combination of MacWrite, Word, and MacDraw. The program puts an image of a sheet of paper in the center of the screen, with a lot of desk space around it. You can read MacWrite, MacPaint, MacDraw, and Word files directly from the disk and place them on the page in any fashion you like. This method is great for producing small publications, presentations, newsletters, slides, posters, and the like, containing text and graphics: multi columns, text and graphics side-by-side, and overlaid. PageMaker is powerful and reasonably easy to use, but works best only with some extended RAM, a hard disk (critical), and a LaserWriter (or Postscript-driven phototypesetter). It is also limited to less than 16 pages per file, and doesn't yet work with custom Laserfonts. However, the cost of all this is still much cheaper and more powerful than traditional publishing equipment. Expect to see more powerful, larger-capacity publishing programs later.

LaTex- and troff-to-Postscript Translators

For those who want to use an Apple LaserWriter for printing LaTeX or troff files from UNIX or VAX VMS, a couple of translation programs are now running on various Berkeley and LBL machines. These programs convert to Postscript format and then spool the file to the LaserWriter. You can also print your MacWrite files remotely by saving the PostScript file to disk (type *Command-F* after **ok** in the LaserWriter print dialog box). You then upload the PostScript file to any mini/mainframe that has a LaserWriter by using MacTerminal and a modem. Then ask the mini/mainframe to send your file to the LaserWriter. (The mini must also send the LaserPrep file to the LaserWriter.) You will, of course, have to go in and pick up your printout, but this is reasonable for final drafts.

Microsoft Excel

Excel, the state of the art in the evolution of the spreadsheet, now combines a spreadsheet, database, plotting, page formatting, macros, matrix manipulation, and a powerful programming environment. Excel is far superior to any other spreadsheet program available on any computer because of its combined ease of use, power, depth, flexibility, and programability. Now there is no reason to even consider buying any other spreadsheet.

Excel is fast because it recalculates only dependent cells when a change is made rather than recalculating every cell in the worksheet. Excel is easy to jump right in and use, with an interface similar to Multiplan's. Excel makes use of up to one megabyte of RAM per worksheet, and multiple worksheets and charts can be open at the same time and combined with powerful ways of linking to still more worksheets on disk.

Ranges of cells can be set up as databases or tables. This organization allows you to look up a record, sort on a range, or refer to a table from equations in other cells. You also have the capability to define and make calculations with arrays and matrices including matrix multiplication, inversion, and other procedures that are impossible to do in other spreadsheets. This opens the ability to incorporate more sophisticated mathematics into your spreadsheet models and allows complex simulations in Excel.

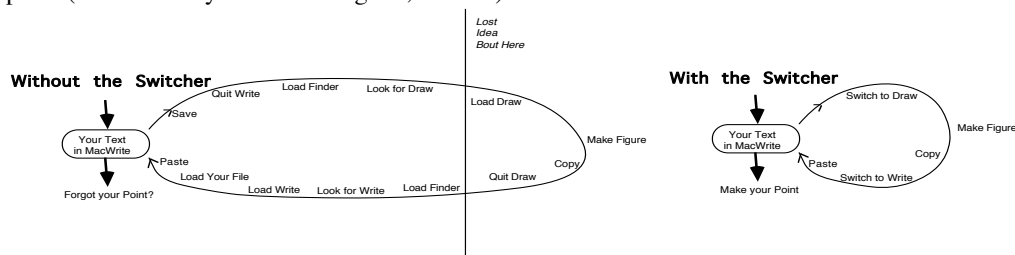
Other advanced features of Excel are its plotting and printing capabilities. It can produce elaborate graphs, formatted reports, invoices, and similar forms. A nice **show page** feature allows you to preview a page layout before you print--much like show page in MacPaint. Look for this feature to be incorporated into Microsoft Word 2.0 and Microsoft Typographer versions to be released soon.

The macro **recorder** feature allows you to easily create complicated macros by just showing the computer what you want it to do by pointing, clicking and typing through the procedure once yourself. You are actually writing a program using a very high level language. This is the way programming should be--*watch me, now you do it*. You can also go far beyond the simple recorded macros by adding things like input dialog boxes, user prompts, or custom formulas by using Excel's rich programming language. Sophisticated custom applications for vertical markets or complicated models and simulations can be quickly programmed in Excel, so the nonprogrammer can concentrate on the concepts behind the model or simulation and not waste so much time in implementation. We are currently preparing a BMUG disk with Excel templates and macros for a variety of applications.

Excel and PageMaker will sell a lot of Macintoshes to the skeptics because they are far beyond anything currently available on the IBM PC, XT, or AT (although a version of an Excel-like program will probably be coming out soon to run under the new Microsoft Windows operating system on the PC). Excel sits squarely in the area where the PC is generally considered superior--financial number crunching. The only things left to recommend the PC are the specialty programs already written for it and the large number of PCs already installed. The Mac/HD20/LaserWriter combination is now clearly superior for both word processing/page-layout applications and financial spreadsheets, the two most common reasons for using microcomputers.

Andy Hertzfeld's Switcher

Excel and PageMaker are only hints of what is to come for the Mac in making it a more powerful tool. Switcher with multiple applications loaded at the same time adds a new dimension of functionality to the Mac, particularly with 1-4 megabytes of RAM and a hard disk available. Having that other software tool right at hand makes it much easier to incorporate that drawing, graph, calculation, or picture into your document. The minute it would take to quit the word processor, load MacDraw, make your diagram, copy it, quit, and reload the word processor is too much of a disturbance in your train of thought to bother with. So your documents lack those extra touches that would communicate your point (that *is* what you are writing for, isn't it?)



The Switcher, as good as it is, is only an intermediate step toward a true multitasking operating system (coming soon) where several different applications can be running in different windows on the screen at the same time (much like worksheets and charts in Excel,) perhaps with linking and macros.

Apple Resource Editor for the Mac

One of the most important features of the Mac is its *user interface standard* where the basic command structures and data file structures are the same for any application from any company that chooses to follow the rules. This standard interface means that if you've learned one program you will already know all the basic functions in any new program, because these basic functions (open, copy, save, quit, etc.) will be the same as in Write or Paint. Thus you can jump in and start learning the subtle and advanced features of the new program, rather than spending your time learning yet

another new set of function keys assignments and commands. Those familiar with the IBM PC or CP/M machines will immediately appreciate the tremendous value of this.

In what seems at first to be contrary to the standard interface idea, the Mac is designed to let you customize both the operating system and any application to suit your own needs. You are familiar with the way you can use the Control Panel to adjust many aspects of the Mac such as the keyboard sensitivities, mouse sensitivity, and desktop pattern. The Resource Editor allows you to modify many aspects of applications like Excel, MacPaint, or MacDraw. This capability is essentially unknown on most other computers and is made possible by the highly structured programming environment offered by the Mac. The environment encourages every programmer to code menus and other features in the same way. One benefit of this standard is that a single utility can be used to edit these common aspects (resources) in any program. The Resource Editor from Apple is the utility that allows you to customize the resources in any Mac application. See the several articles below describing some of the many uses for the Resource Editor.

Future Products to look for soon (AKA: Some Rumors)

The new ROMs for the Mac

A new set of ROM chips for the Mac are close to being released from Apple. The current ROM chips have 64k of special sub-programs (including QuickDraw and various Managers for controlling things like windows, I/O, drawing to the screen, and keyboard/mouse input.) Each application makes use of the code in the ROMs rather than doing everything itself. The new ROMs will be faster and will contain 128K of subroutines including the AppleTalk drivers, the new hierarchical file structure, the new list manager, perhaps a system font, and a whole host of other things. The current holdup seems to be Apple deciding what programs *not* to include in the limited space of the new ROMs.

You will be able to upgrade your Mac from its current ROM chips to the new ones for about \$50 (the chips cost \$35) by taking your Mac to your dealer. The dealer will open the case, pull your old ROMs out of their sockets and plug in the new ones (10 minutes total). Look for these in January.

A Multi-Tasking Operating System for the Mac

Also coming relatively soon is a true multitasking operating system for the Mac. Multitasking allows you to have several different applications open in different windows on the same screen and running at the same time. You can activate the menus for one or another application by clicking on the window or by selecting the application's name from an additional pull-down menu. The feel is much like going between charts and worksheets in Excel except that you can be running *any* Mac application in each window. The different applications share clock cycles on the Mac's 68000 CPU and can each be assigned a different priority for control. This program looks fantastic and leads the Mac to a whole new level of functionality. Look for it in January.

Object-Oriented Programming on the Mac

Several new object-oriented software development tools for the Mac are under development at Apple. A new object-oriented MDS development system, object-oriented C compiler, and object-oriented Pascal compiler will all be native to the Mac. The common object-oriented programming environment will allow developers to simply piece applications together much like you can do with resource editor. This should make the life of the Macintosh developer easier because the object-oriented development systems are much more flexible than their procedure/record-oriented standard counterparts.

Apple II Emulator and Apple 3.5-inch disks

We have seen a very competent Apple II Plus (6502)-emulator running on the Mac. It seems relatively fast and very robust in its ability to run Apple II software on the Mac, and accurately performs screen graphics and I/O. The 3.5-inch drives announced for the Apple II are the same variable speed Sony drives used in the Mac (although they are double sided). Thus it is only a matter of software to be able to read the Apple II-formatted floppies in the Mac, and, with this emulator, run many of your Apple II programs directly on the Mac. Look for this around December or January.

TOPS and AppleTalk Filing Protocol

Centram Systems in Berkeley has implemented the TOPS network protocol on AppleTalk. (TOPS is actually a file protocol running on the AppleTalk network protocol. Apple will finalize its own standard *filing protocol AFP* and

session protocol ASP by the end of the year. The *AFP/ASP* probably won't be the same as the TOPS system but will also be accessible from MS-DOS). Under TOPS, machines with different operating systems and different data structures can each convert their files to TOPS format files and send the converted files over the network. Each machine is also capable of converting a TOPS format file back to its own format. Thus you can have a machine-independent local area network where any type of computer can access the network assuming the machine has a file translation program.

Centram has a translator that runs on a Mac, a PC, an AT, and on UNIX soon. They are also producing an AppleTalk card for the PC/AT. You will connect the computers together with AppleTalk (or preferably BMUGNET) and any machine will be able to access files on any other machine's disks. For example, a Mac running Excel could open (and use) a Lotus 1-2-3 file on the floppy disk of an IBM AT machine in another room. Or, conversely, the AT could access a MacWrite file on the floppy disk of the Mac. With this software, you could use that XT or AT as a hard disk/file server/print spooler for a LaserWriter and a network of Macintoshes all hooked together with BMUGNET/AppleTalk. The Centram software will allow any device on the network to act as the **file server**, will handle print spooling to the LaserWriter, and has an optional utility to help the PC to use the special fonts and graphics capabilities of the LaserWriter.

Infosphere makes *XLServe*, a program for the MacXL that uses the XL's hard disk as a **disk server** for an AppleTalk network. Infosphere recently released *MacServe* that makes a Mac/hard disk combination into a disk server for AppleTalk. (MacServe works with the Hyperdrive, Apple HD20, Tecmar, and Quark). As soon as Apple finalizes its AppleTalk filing protocol, Infosphere will be coming out with the software to make a true AppleTalk **file server**.

The Mac as a front end for UNIX

Several companies are working on UNIX front-ends for the Mac where you use the Mac's user interface as a terminal for running jobs on the UNIX machine. The UNIX machine will be running a program that controls the Mac with QuickDraw commands. This way a lot can be happening on the Mac end while minimizing the volume of information transferred between the two machines. The user will be able to send UNIX commands with the Mac's pull-down menus, and have different UNIX tasks running in different windows on the Mac screen. These programs are indicative of the general trend towards smarter terminals taking over more of the front end load from the central mainframe.

Various Rumored Hardware Upgrades for the Mac

On October 1st, Apple dropped the list price of the 128k to 512k upgrade to \$450. Several other hardware upgrades for the Mac are rumored to be coming soon. One reasonable rumor is for a board swap upgrade to a fast (12-17MHz) 68000 or 68010 CPU along with 1 megabyte of RAM, expandable to four (or more), and perhaps a deeper case to accommodate a larger board. Another reasonable rumor is for the release of an add-on expansion bus with slots (who needs it?) with cards for video stuff, modems, etc. A new keyboard for the Mac is also rumored. The new keyboard would incorporate both a numeric key pad and a trackball (like the Assimilation Process' Mac•Turbo•Touch add-on trackball). Other machines rumored have a component design (box, monitor, keyboard) and will likely have a larger monitor, expansion slots, more memory, etc. etc. Don't expect Apple to release the much rumored 68020/68881/many megabyte machine very soon given recent events (see the comments on the 3M machine below).

Laser/Video Disk Interface

A video disk interface for the Mac is inevitable. It is only a matter of time before the drives come down in price and are standardized enough for one to be released for the Mac. You can already control some of the laser/video disk drives on the market (like the Pioneer) from the Mac's serial port, but these drives store and replay only video signals at the moment. Something to look for in mid-'86 are read-only digital laser disks for the Mac that you would use to access large libraries of data like dictionaries, encyclopedia/databases, and large program libraries. The new hierarchical file structure for the Mac is a logical step towards being able to use the read/write laser disk mass storage devices that should become generally available next year.

The 3M/4M Machine and the future

The university workstation machine is a set of design specifications proposed by the Information Technology Center (a consortium organization centered at Carnegie-Mellon University with several participants from U.C.Berkeley). The specifications are commonly referred to as the 3M machine because it is slated to have at least 1 meg of RAM, a display with at least 1 million (1024 by 1024) pixels, and be capable of performing more than 1 million instructions

per second (some people add a couple extra Ms in there for things like floppies, etc). The machine will most likely have a UNIX as the low-level operating system with a better high-level user interface. The specifications for this machine were proposed more than a year ago, along with the restriction that the machine must cost less than \$3000. The principal intended application for the machine is for use as a standard workstation for the university environment and several schools would require all entering students to purchase one as part of their tuition. By establishing a standard with a specified minimum installed base, hardware manufacturers could recover their development costs and the universities could devote major efforts towards developing specialized software for that design.

The specifications and price point were ambitious but considered reasonable for an expected development completion by early 1986 with quantity shipments available by fall of '86. Several companies, including IBM, DEC, TI, Sun Micro Systems, and Apple, were involved with the contract proposals and started working on designs to meet the specifications. It seems unlikely that any of the companies involved will be able to produce a machine satisfying the specifications and price by early '86 as required. SUN recently released a 68020-based machine that would meet the design specs but not the price point. IBM certainly has several possibilities available, and DEC has the microVAX line.

The specifications for the 3M machine are close to those of several machines under development at Apple, including a design being worked on by Richard Page. That machine was based around a 16 MHz 68020 with several megabytes of RAM, a 68881 math coprocessor, and a large high resolution display - a 3M machine. When Jobs left Apple to form his new company called Next, several key Apple employees also left Apple to join the new company. Page was among the first 5 Apple employees to leave with Jobs (since then several more key people have followed.) When Next described the product that they were planning to produce, it was the 3M machine, to be directed at the university market. Apple filed suit against both Jobs and Page claiming that they were taking Apple trade secrets when they left and the suit was to prevent Next from incorporating those trade secrets into their new machine and competing with Apple in the same market- the universities. The machine that Page had been working on at Apple had no doubt incorporated Apple trade secrets but was built around the same basic components as other companies working on the 3M machine, as you might expect, because all are designed for the same specifications. Thus although Next will be restricted from using some things from Apple, they will certainly be free to develop a very powerful machine. Unfortunately, Mac compatibility will be very difficult for Next to incorporate into their new machine, but perhaps they will produce something much better. Apple, of course, will continue to work on the 3M machine projects already under way.

The concept behind the 3M machine is for the hardware to be powerful enough to support extremely complex layers of programs. The 3M machine is not necessarily meant to be a stand-alone device but should be able to interact intimately with a larger machine and a layered network. In the interaction with the larger machine, the 3M should handle most of the user interface and be as intelligent as possible in order to access the network as little as possible. The 3M would be asking the big machine for help (information) only when needed, rather than the current situation of the dumb terminal being completely controlled by the big machine - thus bogging down the big machine with trivial overhead and slowing down the network due to the large traffic of low level commands. The smart 3M machine could manage almost all of your work without having to go through the network (or modem) except to get vital information or send final reports.

The 3M machine would also be a standard around which high level development tools could be developed. These tools would allow nonprogrammers to use the power of the machine to interpret their requests and write the low level computer routines for them. Each academic specialist could interact with the computer in the symbols and language of their own specialty when programming the computer to analyze their results, simulate their model, or search the literature for abstract concepts. Of course, these tools could also be used to develop sophisticated courseware.

The researchers would not have to concern themselves with the details of programming the computer but concentrate on the formulating, simulating, and analyzing their problem in the language they are most comfortable in- the language of their speciality, not C, FORTRAN or Assembly language. The current conditions, where you are forced to adapt your thinking to the computer rather than the computer working to interpret your request, is only a transient phase limited by the current state of the technology. We are already seeing hints of what is to come with programs like Excel on the Mac, where to *write a program* that will perform a particular task, you only have to show the computer what you want it to do and leave the computer to work out and remember the logistical steps required for doing the task.

Hardware Notes

by Reese Jones

Notes on Various Cables of Interest

Cables are computer items that often seem (and sometimes are) more complicated than they should be. Apple contributes to this confusion by frequently changing their designs, using with each change a more obscure design of connector and plugs. Finding the right cable to connect two devices is often the most difficult aspect of getting them to work. If a device isn't working right, odds are good that the problem is the wrong cable or a loose connection somewhere. Cables are basically **very** simple and inexpensive to make, but often are pricey to buy. Most stores will sell you the mysterious cable of your choice for \$50-\$100, yet you could make it yourself for less than \$10. The **only** trick is getting the correct parts and knowing which color wire goes in which hole. Getting the parts is explained below. Finding the wiring information can be more difficult. (Apple hides it in the \$100 *Inside [this] or [that]* manuals.) The following are notes we have collected on several cables you may want make for your Mac or Lisa/MacXL. See also the article on the BMUG AppleTalk kit.

The cable that came with your Imagewriter (One) printer can be used as a minimal *RS-232 null-modem cable* for connecting your Mac directly (or via a modem) to another computer. Computers we've connected to are: Cray, Vax, Osborne, RadioShack Model 100. You can also use the Imagewriter cable to connect to a serial-driven daisywheel printer, dot matrix printer, serial-driven laserprinters (Apple/HP) or to a phototypesetter. The device you connect to must expect a simple RS-232 "null modem" connection. Many modems can be set to either null modem or direct; see your modem owner's manual. If you plug the Imagewriter cable into the modem port on the Mac and into the RS-232 serial port on your other computer, odds are that this (and BMUG disk #20) is all you will need to transfer files between the two computers. (See **Transferring Files Between Computers** in the Spring BMUG Newsletter for more details.)

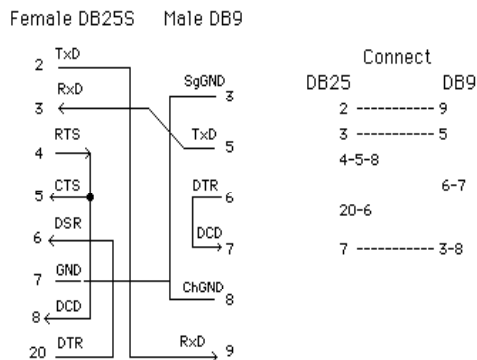
If the other device expects a non-null-modem connection or you would like to make another cable, Radio Shack has a great deal on cables. Their **Joystick Extension Cable** (Cat. #276-1978) costs \$3.99 and includes a 10-ft. length of round cable with two cast plastic DB-9 (Mac type) connectors. **K-Mart** also sells a 20-ft. cable for \$2.99. This cable can be used as is to move your printer further away, or it can be modified to connect to an RS-232 serial device such as a modem, daisywheel printer, or another computer. You will also need two other parts from Radio Shack: a female (usually, although you may want a male) 25-pin solder-type connector (Cat. #276-1548; \$3.99), and a connector hood (Cat. #276-1549; \$1.99). You will also need a little solder and an iron.

To make an Imagewriter (RS-232) cable, cut the Radio Shack cable, and select the appropriate color wires. Solder them to the appropriate pins (see chart below; note that the wire color changes marked with a '*' were in error in the Fall newsletter). These are really easy to make. Don't be scared off -- the most you can lose is the ten bucks for the parts. Our "make your own cable night" at a recent meeting was a success and we plan to do it again soon. If you have any questions, call the BMUG office at (415) 849-9114.

Mac DB-9 pin	Radio Shack Cable Wire Color	Mac Function	RS-232 pin	Connect Cable Wires	RS-232 Function
1	Green*	Frame ground	1	Green (Opt.)	Frame ground
2	Red	+5V			
3	Orange	Signal ground	7	Orange & Grey	Signal ground
4	Yellow	Trans TXD +			
5	Brown*	Trans TXD -	3	Brown	Receive data
7	Black	HSC in	20	Black	Data Terminal Ready
8	Grey (or Purple)	Rcv RXD +	7	Orange & Grey	Signal ground
9	White	Rcv RXD -	2	White	Send data

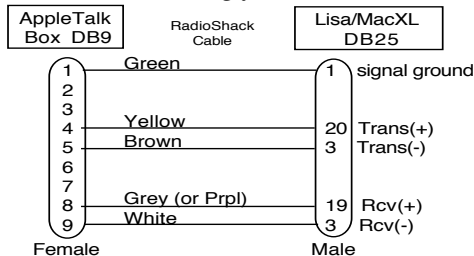
For some printers, some modems, and some direct connections to other computers, you will need a **non-null modem** cable. Solder DB9 pin 5 (brown wire) to DB25 pin 2, DB9 pin 9 (white wire) to DB25 pin 3, DB9 pin 7 (black wire) to DB25 pin 6, and DB9 pins 3 **and** 8 (both orange **and** grey wires) to DB25 pin 7. Note that wire no. 8 may be either grey or purple depending on when or where it was made. The other color codes have been consistent from RadioShack for more than a year and a half.

The IBM PC uses a slightly different RS232 pinout than most everybody else (surprised?). The connections for an IBM PC RS232 to the Mac DB9 are shown on the left.



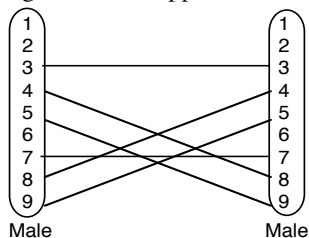
Lisa/MacXL DB25 to AppleTalk Box DB9

This is for connecting your Lisa/Max XL DB25 port to the standard AppleTalk boxes.



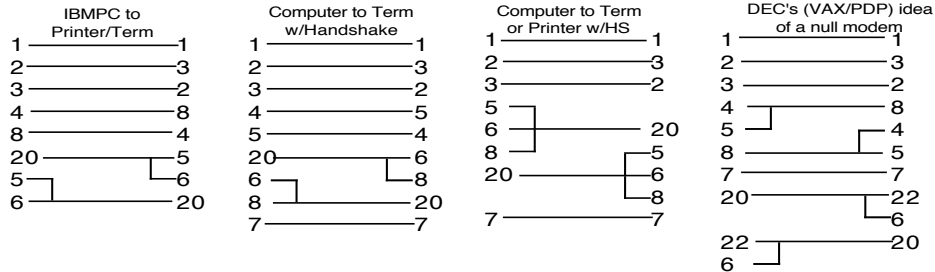
Mac-Mac Cable (DB9 null modem) AppleTalk Substitute

Used to transfer files between two Macs or use the *Macsbug* two-Mac debugging program (one Mac runs the program being tested, the other watches what's happening through the cable). This also works for connecting two devices together over AppleTalk, such as a LaserWriter and a Macintosh or two Macintoshes.



Other DB25-RS232 Null Modem adaptors you might want to make:

[Null modem refers to cables where the send-related wires in one plug are connected to the receive wires on the other plug.]



Sound Output Cables

You can also get ready-made cables at RadioShack that can be used for connecting your Mac to your stereo (so you can **really** play *Airborne*). Get the one with a mono phone jack plug (Walkman size, 1/8") on one end and two female RCA connectors on the other end (RadioShack Flexible "Y" Cable, Cat. No. 42-2154, \$1.99). Also get a regular stereo RCA cable to go from the Y adaptor to your stereo. Plug this setup into the **tape in** or **aux** on your home stereo and you will be amazed with what your Mac can do using programs like *Concertware* or *VideoWorks*. Incidentally, the cable provided with Sony portable compact disk players also work for connecting a Mac to a stereo.

Recently several companies have come out with little self-powered speakers for running a Walkman. These also work well for boosting the Mac's sound output and usually come with the proper cable to plug them right into a Mac. (\$20-80 discounted).

AppleTalk for one Mac and a LaserWriter

If you want to connect only two devices over AppleTalk, such as a Mac and a LaserWriter, you **do not** need to buy two AppleTalk connectors (\$100). You can simply use the Mac-to-Mac DB9 null modem cable described above. Apple should include such a cable with every LaserWriter, or at least tell dealers this so the customer buying a Mac and a LaserWriter doesn't have to spend \$100 for AppleTalk boxes they don't need. If you want to connect three or more devices, you only need the AppleTalk boxes (isolating transformers). DB9 null modem cables can again be used to connect the devices.

AppleTalk Cables

When setting up an AppleTalk network you often find that the cables that come with the AppleTalk connectors aren't long enough to go from room to room. Apple sells a 10-meter extension connector for \$50 (>\$1.50/ft). However, you can buy similar, fully functional shielded-twisted pair cable (2 conductor) for around 20¢/ft.

Don't be afraid to cut an AppleTalk cable in half and insert a longer piece in the middle. When you cut the AppleTalk cable you will find a woven shield layer with 2 insulated wires in the middle, one blue wire and one white wire. To make an extension cable, simply peel back the woven shield, pull out the blue and white wires, and strip off the plastic insulation on each of the two cut ends. Now take your roll of shielded 2 conductor wire and run it through the walls, ceiling or wherever you need to go to get the wire from one Mac to the other. All that remains is to connect the blue to blue, white to white and shield to shield through your extension wire. You can twist the ends together and wrap with tape, or add a little bit of solder to the twist, then tape it up.

This, of course, is probably not approved by Apple or your local building inspector for a permanent installation, as the wire you add in the middle must be specially coated for fire codes. This is, however, very easy to do even by someone who has never touched a piece of wire before, and the cables work well. If you have a long distance to cover, this will save you a lot of money (an AppleTalk network is rated to go a total of 3000 ft.).

The Reliability of the Mac

At the 1985 Apple shareholders meeting, someone yelled from the back of the auditorium "What is Apple doing about reliability problems with the Mac?" Jobs responded that the failure rate for Macs was smaller than parts per thousand which I presume meant at shipping time. This may be so, but once you get it home and the first 90 days have passed,

the failure rate seems to be much higher. By our estimates, based on shows of hands at meetings, the failure rate seems to be more like 1 or 2 in 10 (10-20%). I've gone through 2 keyboards, 3 power supplies, and 2 logic boards. One of the replacement power supplies failed 3 days out of warranty and was not covered (\$200 for repair). As I was walking out of the dealer, the strap buckle on my Apple brand carrying case broke - sending my Mac tumbling to the ground. (The bag was also 93 days old.) We have heard numerous instances of Macs dying 2-10 days out of warranty and not being covered by the dealer or by Apple.

The moral is: get a service contract on your Mac, external drive, and printer. Also, develop a relationship with a dealer who can fix things properly and promptly (he must keep all Mac parts **in stock**). You should get the *AppleCare* service contract at a certified Apple dealer. This contract allows you to have your Mac fixed **free** at **any** certified Apple dealer. The contract should clearly and explicitly state that it comes from Apple. Only consider getting a service contract offered by a particular store, as opposed to an Apple contract, if you know the store and store owner very well and are confident that you will be dealing with that store for a while (i.e. Chapter 13 is not on the horizon for the store). Reliability with the Mac **is** a concern, but, based on the experience of members with several different types of computers, the Mac's reliability is roughly the same as IBM and CP/M machines under the same use load.

Insuring your Mac

In addition to a service contract, you should also insure your hardware, software and data files against theft and disaster. Often when a Mac system is stolen, disks sitting along side it are taken too. After a while, the value of your software can easily be worth more than the hardware, not to mention the value of your data and the time it would take you to recover from the loss.

You should keep backup copies of your most valuable programs and data in a separate place and get an insurance policy that covers the Mac, commercial and custom software you own, and an estimate of the expense of recovering your data after a major loss. You of course pay the rates for the dollar value you estimate but that's how insurance works.

An insurance company offering a reasonably-priced policy that covers hardware, software, and data is Data Security Insurance (DSI) 1877 Broadway, P.O.Box 9003, Boulder, Colorado 80301, 1-800-621-8385, x494. About \$50/year covers a system.

Surge Protectors

If you use your Mac in a place where the power is irregular, a surge protector is probably a good idea. There is some controversy as to which is the best one, and even over what a surge protector should do. One brand people generally find acceptable is the Curtis Ruby, available from MacConnection (1-800-MAC-LISA, about \$60). This model has the advantage of having 6 grounded plugs. It is just as important (or more so) to surge protect your ImageWriter and LaserWriter as well as your Mac, so your surge protector should have **at least** 3 plugs.

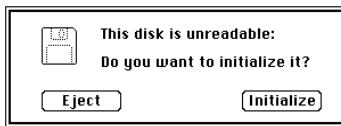
If you live in a place with particularly unreliable power, where blackouts, brown outs, or lightning are common, check out the uninterruptable power supply designed and described by Jay James in this newsletter. Because of the design of the Mac's power supply, a square wave inverter (which is cheaper than sine wave inverters) can be used for running your Mac off a car battery for up to 12 hours. With an inverter (~\$60), a car battery, and a trickle charger for the battery (\$35), you can make an uninterruptable power supply for your Mac. These usually cost several hundred dollars for most other computers which require sine wave inverters. These power supplies also work in any country; all you have to do is find the appropriate battery charger for the local conditions (220v or charged by your Land Rover). The BMUG BBS system is now running off of such an UPS with good results. You should also consider putting a surge protector between the inverter and the Mac for extra safety.

Fans for your Mac

One source of reliability problems for the Mac power supply is heat. Unlike most other computers, the Mac was designed to not require a fan, making it quiet and "allowing you to think". This is a nice concept, but in hot or poorly ventilated environments the Mac probably runs too hot. (As indicated by the power supply and logic board reliability problems in heavily-used Macs.) One solution is to add a fan. The fan should move air throughout the Mac, particularly over the analog/power supply board on the left side and over the floppy disk drive.

Some debate rages over whether a fan should blow or pull air (a blowing fan can filter the air, a pulling fan can't). In my opinion, a sucking fan placed at the top is more efficient for cooling the Mac because of the convection paths available. The main criterion for me is noise, as I sometimes use my Mac in a very quiet location. Two types of fans are available for the Mac: rotary/ball bearing or piezoelectric. The piezoelectric - which has butterfly-type "wings" - is quieter but not as powerful. Levco (619) 755-7827 has a piezoelectric fan for \$29. You have to install it internally yourself, but they also sell the tools necessary for opening your Mac case. Beck-Tech (415) 548-4054 has a combination rotary fan and surge protector that fits under the handle on the top of your Mac so you don't have to open your case to install it; it's also easily removable.

On the subject of heat, Henry Spragens of Beck-Tech pointed out the susceptibility of floppy disks to internal heat in the Mac. He described a scenario several of us have experienced. After using a Mac for several hours with the same floppy disk in the internal drive, the disk heats up and expands. You don't realize this and continue writing stuff on it. You shut down for the day and turn off your Mac. When you come back in the morning, turn on your Mac, and put in the disk you get:



What's this #3∞∅?! Well, the day before when your floppy disk was hot, it was a little bigger and now it has returned to normal size. When you saved your file to the hot floppy the tracks were written in their normal places. As long as the disk stayed hot, the tracks were always in the same place. After the floppy cooled and shrunk, the data tracks moved closer together. When cold, the disk drive read head positions itself over track #3 - it's not there! It's a little closer to the center since the disk has shrunk!

You can remedy this by heating the disk back up to the approximate temperature it was the day before. This can be accomplished by leaving the disk sitting for a while on the vents above the power supply board (on the left side) until it heats up a while. You **should** then be able to read the read the disk (unless something else is wrong). This can all be prevented by installing a fan or by eliminating the constant need to use floppies by getting a hard disk or multi-megabyte upgrade.

Mac Carrying Bags

On the subject of reliability, the Apple brand Mac carrying bags fail miserably. The buckles that hold the sholder strap on are just not strong enough to carry a Mac. The buckles have broken on several Apple Mac bags, sending the Mac crashing to the ground, in one case down a flight of stairs. In these cases Apple replaced the buckles, not the Macs. People who have Apple brand cases for the Mac should immediately replace the sholder strap buckles. You could use 3/8" rope. **Both** of the buckles on my Mac bag have broken (6 months apart). In addition, several people have reported seams on their Apple Brand Mac bags splitting. There also seems to be a large static electricity discharge when you remove your Mac from several brands of Mac bags. The brand I like the best is the MacTote, although I have experienced static discharges through the vents above the power supply board with this brand also.

Air Travel with your Mac

When flying, you should **never check your Mac in a carrying bag**. We have seen **many** Macs destroyed, including the first ever 1 meg RAM-expanded Hyperdrive Mac. If you absolutely must check your Mac, put it in its original shipping box, covering all labels that advertise to a thief what is in the box. Insure your Mac and don't put anything else that you don't want to loose in the box with your Mac. If the airline won't let you carry on your Mac in its case, it is better to take it out of the bag and carry it on by itself rather than to check it.

If you travel a lot, consider getting one of the rigid foam-padded trunks for shipping your Mac. Also look into the MacColby modification. For ~\$600, this modification intalls a Mac into a small portable metal cabinet specifically designed for airline travel. It has room for Hyperdrives, multi-megabyte upgrades, modems, and a cellular telephone. Also available soon from Colby is a Mac with a flat plasma screen installed in a leather brief case .

Also note that some people have reported that metal detectors have destroyed data on their disks. We haven't heard of problems with airport x-ray machines, but in theory **any** ionizing radiation such as x-rays could effect data on disks. Just for safety, have security hand-inspect your disks at the airport security station. Remember that checked baggage is often x-rayed also so you shouldn't check your disks.

BMUG Floppy Disk Review Project

Our floppy disk brand review in the Spring BMUG newsletter received a great deal of attention and was re-published in the New York Times and several computer publications. We got calls from several purchasers and all of the disk manufactures mentioned. Each of the manufactures claimed that their product is now **much** better. We also heard from a couple of the manufactures that several very large contracts were lost (or won) based on the results of the report.

Our current impressions about the relative quality of the different brands have changed somewhat since the Spring newsletter was written. Most of our experience has been with the Sony disks (blue and tan). BMUG members have used an estimated 25,000 disks by now. We still feel that Sony makes the best floppies. Our findings are further reinforced by the comments made by three local high-volume disk duplicating companies. These companies use mostly Sony disks now. Based on the experiences of one of these duplicating companies, we must lower our rating of the Maxell disks. They reportedly wore the heads on the company's disk duplicating machines at a much higher rate than several other disk brands (Sony or Fugitsu). Our opinion of the low quality of Memorex disks has been reinforced over and over by reports from users at meetings and over our phone help lines. BASF has modified their manufacturing process and has been actively trying to demonstrate that their disks are more reliable now. We will be interested to see the results for BASF in our upcoming test.

Since there is obviously a great deal of interest in the issue of relative disk quality and because it is very difficult to quantitatively measure the reliability of floppy disks we have decided to embark on a major project. BMUG plans to try to objectively evaluate the quality of all of the brands of 3.5" floppy disks currently on the market. As more and more computers are using the 3.5 inch disks, the results of this test will be of interest to many people.

The failure/error rates for most brands of floppy disks are in the range of 1:100 to 1:1000. For the test, this fall we plan to track at least 1000 disks of each brand. We intend to test the disks in everyday situations rather than in an artificial test paradigm. The results of an *actual use* test may not be as well controlled as an automated test but the results should be more relevant to performance during actual use.

To accomplish this testing protocol we will get 1000 disks from each manufacturer. 10 to 15 brands will be represented. The disks will be repackaged into sets with one of each brand in each set. These repackaged sets will be distributed to our membership, with a limit of one set per person. We will also distribute evaluation forms, instructions about the test, and a request to return **all** disks with which the user experiences problems. Thus approximately 10-15 thousand disks will be distributed and tested at over 1000 test sites of all user types.

Each month the returned forms will be compiled in a database and the preliminary results made available to anyone who requests them. The short-term results compiled for each brand will include: failure incidence, type of failure, mean time before failure. The short-term results will be published in the Spring 1986 BMUG newsletter. We will try to follow the test disks as long as possible and publish long-term reliability results in the Fall 1986 BMUG newsletter.

Any interested party wishing to help with the project (as a test site, preparing the disk sets, preparing or scoring automated reporting forms, or providing financial or equipment support) should contact BMUG by mail addressed to "BMUG disk project" at: BMUG, 1442A Walnut St. #153, Berkeley, CA 94709. Reliable test sites will also be considered for participation in other upcoming large scale product tests. Companies or others interested in providing financial support for maintaining this project will receive interim monthly test results mailed directly to them as the results are compiled. Remember BMUG is a nonprofit, volunteer-run entity and depends on member support. Contact BMUG at (415) 849-9114 for more information.

Hard Disk Update

By Reese M. Jones

Why buy a Hard disk now

For those of you who have been waiting to "take the plunge" and buy a hard drive for your Mac, your decision should be much easier. The prices of hard drives have fallen drastically over the past months. There are several brands available in the range of \$1000 (discount pricing) for 10 or more megabytes of storage. Apple has finally released their single-user hard drive for the Mac (more about this later); the third party manufacturers are producing second generation designs and releasing software upgrades that address many of the deficiencies in their original products.

Futhermore, the long expected double-sided 3.5 inch floppy drives are still not available for the Mac (as of October 1985) and will probably be expensive if the price for the new 3.5 inch Apple II drives is any indication. By the time the double-sided floppies become generally available, you should be able to buy a discounted Apple or other brand of hard disk for about the same price as the 800k floppies, and get 50 times the storage capacity and 5 times the speed. You should get a hard disk now. Next year this time they will be twice as big at half the cost but you will have had the pleasure of using your own for a year by then.

What can a Hard Disk do for you

If you haven't used a hard disk before, you will find that it improves the functionality of the Macintosh more than you would expect. Once you have used a Mac with a hard drive for a little while you can never go back to a standard 2 drive 512k (or more) Mac without frustration.

The main advantage of the hard drive is having all of your favorite applications, tools, fonts, desk accessories, templates, data files, and previous work *instantly* accessible at your whim. This frees you from searching through piles of floppies or interrupting your thoughts when switching programs. On a floppy based system you will often find yourself not including that little graph, diagram or figure in your text because it was too much effort; as a result, the quality of your finished work suffers.

Things to Consider

Hard disks vs Super RAM upgrades

The question often comes up, *should I get a hard disk or a 2-Megabyte RAM upgrade?* One answer, of course, is "Get both." But realistically, most of us can only get one. The advantage of getting the larger RAM is that you will be able to run a few of your programs much faster ("blindingly fast" if you are accustomed to floppy-only systems) with the various programs running from a large RAM disk and still have extra RAM available for the program to use. The disadvantage of being restricted to the RAM-only system is that when starting up you will need to swap in 3-5 floppies as you load up your RAM disk. You will have to deal with all the problems invoked by copy protection when working with RAM disks, and if you happen to want to copy a paragraph out of that letter you wrote a month ago, you will be forced to have the right floppy at hand and load it in to get the right file. The worst thing is that all of the information in your RAM disk is volatile. When you turn off your machine, or if you lose power, everything is lost.

A hard disk is not as fast as a RAM disk (but fast enough for most applications) and has the major advantages of being nonvolatile and capable of storing many more programs along with *all* your old files and data. With each program and piece of data right at hand, you don't have to worry about where your floppies are and which one has which file or program on it. Another advantage of a hard disk over a RAM disk is that copy-protection problems are much easier to solve. Most Major applications can be installed on hard disk such that the program will no longer require a master floppy disk to run.

A hard disk, therefore, is a better choice for most people; the large RAM/RAM Disk is faster if you only use a couple of applications and don't regularly refer to very many files. If you can afford both, you can install a Beck Tech 1 megabyte board in a Hyperdrive 20, or use the Apple HD20 in conjunction along with a 2 megabyte Levco board. The rumored 1 megabyte/fast 68010 Apple Mac upgrade is also coming soon; so it might be wise to hold off on getting a large RAM upgrade until seeing Apple's product.

Apple or non-Apple

The Apple brand hard disk will become the standard hard drive for the Macintosh; developers will make sure that their programs work right with the Apple drive first. If they have time and money they will test their products on the other brands. For this reason a third party product should be either **not** compatible with Apples and **much** less expensive (>30% less) or offer **substantial** advantages in performance over the Apple product in order to be considered for purchase. At the moment, the only single-user hard disk worth considering other than Apple's is the HyperDrive because of its increased performance (equally fast, internal, easier to use, with several additional features as well as being a second standard for other software products). Other brands of single user Macintosh hard disks, many of which are very fine products, are not worth considering unless they become **a lot** less expensive (much less than \$1000).

Partitions vs. Tree Structure

The Apple HD20 uses the Mac's new hierarchical file structure. In the new Turbo File System (TFS) files can be *archived* into a folder such that they are no longer maintained in the directory in the RAM. You will thus be able to have unlimited directory sizes in with out needing tremendous RAM just to store the directory. This becomes useful when you have more than 128 files per disk or more than 600 files on the storage device. For storage devices larger than 5 megabytes the new file system is a necessity, and has great application when the large capacity laser/video disks become available for the Mac.

The advantages of the TFS are that it is faster, more memory efficient, and the new Apple standard to which all major new devices will conform. There are still some advantages to the partitioning possible with many of the hard drives (you can have the TFS in the partitions, but the Apple drive does not currently support partitioning). Partitioning is conceptually nice to deal with allowing for logical task separation, since the floppy disk or book volume is easy to deal with. Passwording and security levels are easily understood at the partition level rather than the folder level, and the ability to fool older applications (that are incompatable with the TFS) into thinking they have multiple floppy disks is useful.

Speed/Bus Interface

The different Macintosh hard drives interface to the CPU through several means. The Hyperdrive connects directly to the data bus (the fastest method); the Apple and Quark hard drives connect through the external drive port at 500K Baud (baud = bits transfered per second); most of the other single user drives connect through the serial ports at between 500K baud and 900K baud depending on whether they are internally or externally clocked; some drives like the MicroDesign connect through AppleTalk which runs at 230K Baud. However, contrary to expectations, each of these drives run at *roughly* the same speed in day-to-day use. This suggests that the speed limitations in hard-drive performance is not hardware-limited. This turns out to be the case even at the intermediate speed of the external drive port. The transfer rate through the external drive port is 500K baud or 500,000 bits per second; the RAM contains 512K of 8-bit words or 4 million bits; so at the full speed of the hardware you could fill the whole RAM in eight seconds. But to load MacWrite, which fills a fraction of the 512 K RAM, it takes between 5 and 20 seconds. The point is, the software has much more effect on the perceived speed of the different hard disks than the method of interface, so don't make prejudgements on the speed of a particular hard drive based on which port it goes through.

Noise

Until you bring home a hard drive, you may not appreciate how **quiet** the Macintosh is. The Mac is cooled by convection alone and does not require a fan. Most hard disks are relatively noisy; some are noisier than others. You should try the drive in a quiet location in order to see how loud it is. You will find that the cooling fan in the drives contributes less to the noise than the sound from the drive itself. The 3.5 inch hard drives tend to be noisier than the older 5.25 inch hard drives. If you use your Mac in a very quiet environment the sound of a hard drive should be a consideration.

Price and Size

Given the list price of the Apple Hard Disk you should not consider paying more for **any** other drive except the Hyperdrive 20, and then not more than \$1995 (a Hyperdrive20 with a 1meg Beck Tech board installed is \$2645; try Software-for-Less in San Francisco at (415) 753-1066). You should be able to buy an Apple HD20 at a variety of discount stores for 10-20% off list price (don't just pay list price!). The special discount pricing for the Apple HD20 varies depending your circumstance: Berkeley ASUC Store, \$1295+tax; UCSF, \$1080+tax; Stanford, \$980; Developers, \$750.

Don't buy a larger drive than you actually need. 20 megabytes is adequate for almost every single-user application. You can always add more storage **when you need it** by daisy-chaining multiple HD20s together or by adding an external drive to your Hyperdrive. By adding additional storage only as needed you can expand while taking advantage of the rapidly decreasing prices.

For network installations you can also add multiple **file** servers to the network as needed, using a single large drive, particularly a disk server type (like the 127 meg Corvus), will make the network much slower than using multiple small file servers (like MicroDesign Keepers or MacServe/HD20 combinations). The very large drives will be cheaper per megabyte. But with the decreased network performance, and if you don't **absolutely need all** the storage **immediately**, the building-block approach is much better.

Print Spooling and AppleTalk Compatibility

All of the major hard disks available for the Mac are AppleTalk-compatible (except those that can **only** interface through the printer port). It will be important to maintain AppleTalk compatibility to be able to use the host of AppleTalk products coming out over the next year. Print spooling to the ImageWriter is a nice feature for a hard disk to be able to handle, but most don't speed you up by more than 50% because much of the print time is due to MacWrite or MicroSoft Word formatting the print file to sent to the buffer. Several companies are coming out with RAM-based print spoolers that will work with any system. And with a multitasking operating system for the Macintosh, you can have MacWrite operating as a printing job in the background of what ever else you might be doing.

Software Compatibility and Copy Protection

With the new hierarchial filing system, new finder, and new Mac ROMs coming out soon, we expect to see a whole host of software incompatibility problems with the new Apple HD20; third party drives will soon take advantage of the new file system. Copy protection (the nemesis of the legitimate purchaser and an adventure game for the pirate) is a definite problem if you want to use a hard drive. However, most good business applications come with a means of installing them on a hard disk, or can be installed by following the instructions described in detail in the Spring '85 BMUG Newsletter.

A couple of utilities are available for that can make a backup copy of any program within one month of the program's release. **Copy2Mac** (which comes with MacTools and Copy2Hard Disk) from Central Point Software in Oregon is the best and a definite *must* for every hard disk owner. Some additional utilities can actually deactivate the parts of many programs that control their copy protection. The Copy2Hard Disk utility that comes with Copy2Mac can remove the copy protection from several of the most common applications. Another program, **Hard Disk Utility**, a very easy-to-use program that can deprotect hundreds of Macintosh programs, is available from FWB Software; see the "Choice Products" section. Registered owners of Hard Disk Utility can obtain perpetual upgrades to handle new programs by calling and joining *The Bay*, a Bulletin Board in San Francisco at (415) 775-2384 for \$30. Both of these utilities work with every hard drive on the market; every Macintosh hard disk owner should have them.

Single User Drives

Summary Recommendation

For personal-use hard drives for the Mac: Hyperdrive 20 is best if price is no object; the Apple HD20 is the most cost-effective. Don't buy more storage than you need immediately because you can always add more on later.

Apple Hard Disk 20 (Finder 4.9 or greater)

This is the one to get from the point of view of cost, size, general performance, expandability and compatibility with future Apple and third-party software and hardware products. The HD20 has twenty megabytes of storage on a sealed (nonremovable, more reliable) 3.5 inch hard drive. The unit sits squarely under the Mac and is about 3 inches high. There are numerous third-party software products that don't work properly under the TFS on the HD20. Apple claims that in almost every case the problem can be traced to some part of the program that is implemented incorrectly according to *Inside Macintosh*. Most of these problems will be routed out in time; we will try to compile a list of the problem programs and fixes as soon as we encounter them (and as soon as we get some of the drives to work with). So if you must use a particular program, be sure it runs under the TFS on the HD20 before buying. The specs on the power requirements seem fairly robust (85 to 270 volts AC, 47 to 64 Hz, 30 watts draw). We haven't found out for sure if the HD20 can be run off the square wave inverter as used in our battery-powered uninterruptible power supply (see the UPS article) but it looks promising.



A few other features that are lacking in the Apple HD20 are a backup utility, print spooling to the LaserWriter and ImageWriter, passwording different access levels for the different folders, and an additional partitioning capability similar to the hyperdrives (not necessary for use but conceptually it is nicer to be able to have the flexibility for using two different *methods* for dividing up your storage. The optional partitions could also be used to fool the incorrectly implemented programs into thinking there are different floppy disks present).

One other minor problem is that the HD20 is noisy (but no more than any of the other hard drives). The noise could be minimized if you had the flexibility of putting the drive on the floor (or even in a closet). The cable is much too short to move the drive any distance away from the Mac yet longer than the minimum necessary, so you have a clump of cable to deal with in the back. With most of the other external drives that run through the serial ports, you can use a \$2.99 cable to move them up to 20 feet away (see hardware notes); a cable to move the HD20 further away would be more difficult to make because the HD20 interfaces through the external drive port with a DB19 connector. All in all, the HD20 looks very nice and should be easy to use; the price is right.

General Computer Corp. Hyperdrive (Version 2.0 Software)

The Hyperdrive has several nice features not found in the Apple HD20: The Hyperdrive is installed internally on the 68000 data bus, which should make it faster in theory (in practice it's roughly the same speed as the Apple drive in day to day use, although both these drives are faster than most of the other drives on the market). The recent Hyperdrives are installed by using a Motorola clip-on debugging cable to clip to the 68000 chip on the Mac, connecting the ribbon cable to an add-on controller board. Because installing a Hyperdrive does not require permanently modifying the Mac's logic board, the installation does not void the Apple warranty.

One of the best features of the Hyperdrive is that you can boot directly from the hard drive (no other hard disk can do this), so you don't need to carry around any floppy disks at all. The software for the Hyperdrive is far superior to that on any other hard disk for the Mac, including Apple's. The drive can be divided into multiple volumes that are mountable and unmountable from a desk accessory, the partitions are dynamically sized as you add or remove files, the Hyperdrive is AppleTalk-compatible and compatible with the new turbo file system, and it leaves *all* Mac ports free. In addition, the Hyperdrive software provides an easy method for password-protected volumes and files. The 2.0 software version provides a disk-caching buffer and a print-spooling utility. The Hyperdrive works well with every major application on the market (unlike the Apple HD20) and can be used as an AppleTalk disk server with the MacServe software.

There were a few bugs in the 2.0 version of the software including a problem updating the desk top of a partition when it is unmounted either by dragging it to the trash or by using the Drawers desk accessory. Some of the features in the current version of Central Point Software's MacTools (v4.2) don't work with the Hyperdrive 2.0 software; it works on several of the other hard drives. An excellent backup utility is also provided. Because GCC is closely involved in the development process, they have guaranteed that the new Mac ROMs will work with the Hyperdrive when they are released. The biggest complaint about the hardware is the noise produced by the drive itself. The noise is right at the resonant frequency of my ear and is a little grating when working in a very quiet environment. An external hard drive could be put on the floor or in the closet with a longer cable; the hyperdrive must be no further away than your Mac itself.

The Hyperdrive is definitely the most pleasant and easiest-to-use hard disk for the Macintosh. Our biggest concern with the Hyperdrive is its reliability. The Hyperdrive is mounted inside the case of the Mac, one of its nicest features, but the Mac has enough problems cooling itself, much less an additional circuit board, power supply and disk drive. GCC addresses this problem by installing a mechanical fan inside the case to draw off the extra heat. The other reliability concern is that your Mac gets banged around a lot more than might an external hard drive, so the chance of damaging it are higher. One of the best features about the Hyperdrive is your freedom to take along only one box and have

everything with you. Setting up the HyperMac in the new place is even easier than setting up a regular Mac with an external floppy drive (which you will never need as long as you have a Hyperdrive, or any hard disk for that matter).

Our personal experience with the reliability of the original solder-on Hyperdrives has not been good. Out of a group of 8 of us that got some of the first units, 6 have failed for one reason or another (failing *twice* for two of us). The new clip-on Hyperdrive 20s are supposed to be much more reliable, but I would definitely consider getting a service contract on both the Mac and the Hyperdrive. The Hyperdrive20 is definitely my drive of choice personal use assuming price is no object; otherwise get the Apple HD20.

We have heard several rumors that GCC is working on some more powerful add-on boards for the Mac that may include larger memories, 68010 or 68020 CPU replacements, and/or floating point coprocessors.

Paradise Systems Mac-10 drive: The Paradise drive was released in August. The Paradise Mac-10 uses an external 3-1/2 inch hard disk in a small case that sits along side the Macintosh, and an industrial-looking power supply that sits on the floor (especially at trade shows and in their ads). The Paradise drive is small, fast, and quiet. The main software is elegant and very nice to use (second only to the Hyperdrive's), providing multiple volumes (variable-sized, nondynamic) that can be mounted and unmounted from a desk accessory. Partitions can be dragged to trash for unmounting (Updating the desktops and clearing the directories from RAM, unlike the Hyper 2.0 software).

The Paradise uses a very simple method of setting up the bootup configuration; creating volumes is straightforward and easy; the software seeks to automatically find the port through which the drive is connected. I have seen the Paradise drive running with the new Finder and the new Turbo File System in multiple partitions. The Paradise drivers are easily installed into a system without conflicting with the AppleTalk drivers, the Hyperdrive drivers or the drivers of several other hard drives or other desk accessories. This was refreshingly unusual compared to the other hard drives I've dealt with. One particularly nice feature is that the drive senses if the 5V signal is present at the Mac's serial port. If the Mac is off and the signal is not present, the drive turns off; when the power of the Mac is switched on, the 5V signal switches a relay in the Paradise and the drive turns on.

One complaint with the software is that to change the size of the disk based print spooler buffer you must reformat the entire disk. This would be ok if the default configuration for the spooler size was maximum, but on the drive I tested, the buffer was set a 200k out of the 400k. The print spooler works but as with all of the print spoolers for the Mac, it doesn't help much because most printing time is used to create bitmaps.

My big complaint with the Paradise is with the hardware design. When several of us took ours out of the box, the front cover of the drive fell off, with internal guts spewing forth and wires swinging. A little Scotch tape holds the case together nicely, but it's not included. The cables that connect the power supply box to the drive case and the power supply to the wall plug are **very** stiff and cumbersome. They should be removable from both the drive case **and** the power supply box, making packing and transporting the drive much easier. After some fussing around, our preferred configuration was to attach the drive case to the top of the power supply with a bunch of rubber bands (making the combination half the height of the Mac) setting both parts alongside the Macintosh. During the evaluation the drive I had stopped working. It was a bad power supply, which I managed to fix after disassembling the power supply and re-soldering all the suspect looking connections on the board. It has been working flawlessly for several weeks since. All in all, not a bad drive with much nicer software than hardware, but given the price of the Apple drive, the Paradise should only be considered if it is available in the \$600-900 price range for the 10 magabyte unit.

PCPC Macbottom

Our experiences with the PCPC company have not been particularly good. On several occasions they have said they were sending or had sent an evaluation unit (Federal Express); it never arrived. Other people complained that after ordering units direct from the company that it took numerous calls and complaints to finally get their drives. These types of interactions with a company color our impressions of what is otherwise a very nice-looking product. I have not had a chance to do any extended evaluation of the MacBottom relative to the other drives on the market except at shows. The MacBottom is similar in appearance to the Apple HD20, interfaces through either port, although it must be told which port to use. The MacBottom has some disk-caching RAM built in, but the current version of the software doesn't take advantage of it yet. The software provides desk-accessory mounting of partitions and drag-to-trash

unmounting. Creating new partitions is a little cumbersome compared with the Hyper or Paradise. The MacBottom is worth considering if priced in the range of \$900-\$1000 for 20 megabytes.

Ty Shipman wrote some additional comments about the MacBottom.

I found that the MacBottom drive is excellent and easy to use. The driver software is very sound and works with all versions of the Finder currently available to the public. The utilities at present do not work from a desk accessory, but the volume inserter does. The volume inserter is smart so when you bring a volume to the desk top the inserter window is not covered up. The present configuration of the driver is only 10M bytes. This is enough for the average Mac user. The current retail price for the 10M byte is \$1595.00.

The drive is small and compact. As the name implies **MacBottom** goes under your Mac to save space on your desk. The drive itself is smart. It has a 68008 brain that allows for print spooling, 200K byte by default. There is a small noise coming from the drive, caused by the cooling fan and drive motor—it's not loud and annoying. The software package is good. It includes utilities to backup the entire disk, from a date only, or a specific file. When you configure the drive the utilities allow you to select which volumes you want to appear on the desktop on bootup. The drive is partitioned in 400K byte at a time; this is the only complaint I have about the driver software. A nice feature about the driver software is that after bootup, the floppy disk ejects. The documentation is poor to adequate.

Tecmar (Version 2.1 Software)

This drive has many of the features we would look for in a hard drive: It connects through the printer port leaving the modem port free for a modem. It's small, reasonably quiet, has both a 10MB fixed drive and an optional 5MB removable cartridge, good for rapid backup. The software drivers allow for variable-size, ejectable volumes. Only four partitions can be mounted at the same time on a 512K or 1-meg Mac, a limitation when using the Tecmar for an application where many files are to be accessible from mounted partitions at all times. A desk accessory mount manager would be nice; it's rumored to be in the works. One nice touch is that the microflop boot disk ejects immediately on startup. The MacServe software will also work with the Tecmar drive allowing you to use it as a disk server for a small AppleTalk network.

Corvus Omnidrive

Corvus has a longstanding reputation for making outstanding microcomputer hard drives and has been involved in local area networks for some time. Their single-user Macintosh drive works well and allows the drive to be partitioned into variable-size volumes with a desk accessory mounter. The Omnidrive connects to the Macintosh through the modem port only, making it difficult to use most Macintosh terminal-emulation software with a modem. See below for comments on the OmniTalk disk server software.

Lisa/Mac XL 2/10

The Lisa/MacXL 2/10 has been discontinued, but can still be purchased new or used for a good price. With the new MacWorks software and the hardware upgrade to fix the screen pixel shape, the MacXL will run almost every major software program for the Mac. The internal hard disk and 1 to 4 megabytes of RAM that you can add make the machine a powerful workhorse for most applications, a necessity for using Apple's current development tools (although new object-oriented native development tools will be available for the Mac from Apple in mid-1986). The 68000 in the Lisa/XL runs at a slower speed than the CPU in the Mac so don't expect the XL to run quite as fast as a Mac. With the larger screen, all that RAM, the hard disk and other features it is still a powerful machine worth considering.

IOMEGA Bernoulli Box

There has been a lot of fanfare about this product for some reason we don't understand. People somehow are given the impression that the Bernoulli Box is much better, faster, or easier to use than other Macintosh hard drives. It is basically the same as all the rest, lacking several features provided by all of the other drives. The Bernoulli Box works on a slightly different principle than the other cartridge drives but is functionally identical to the other cartridge drives. Because the Bernoulli Box can be configured only as a single cartridge system it is more limited than either the Tecmar or Micro-Design hard drives which can be configured either with dual cartridge drives, dual fixed drives (which are

faster and more reliable than a cartridge because they are sealed), or a combination of a fixed and a cartridge drive in the same box (giving both the extra speed, reliability, of the fixed disk and the open-endedness and fast backup of a cartridge system all in the same unit). See the article on hard drives with removable cartridges by Paul Gusciora.

The Iomega drive seems adequate for the Macintosh, but several owners have reported problems, particularly in printing with some versions of the Iomega software. The Bernoulli Box is very popular in the IBM PC world and has proven to be reliable as a from a hardware point of view; people rave about it as the second coming. The control software for Macintosh hard drives is much more of a factor in the drive's overall performance than hard drives on the IBM PC. The two versions of software for the Bernoulli boxes that I have tried have been inadequate, relative to the other hard drives available for the Macintosh. I haven't had the opportunity to use the Bernoulli box for an extended period in day-to-day work (long enough to learn how to work around its quirks and limitations). The new turbo finder and TFS should replace the need for special hard disk software and bypass some of the limitations in the Bernoulli's software.

Quark QC-10

Quark was unwilling to provide a QC-10 for evaluation. Because we haven't been able to find one anywhere, we can't say very much about their drive. The drive interfaces through the external drive port, allows partitioning, and works with the MacServe disk server software for AppleTalk.

Cider

The MacCider drive is expected to be released in January and will incorporate turbo finder and the new file system. The drive is supposed to be 20 megabytes and have a list price of less than \$800!. The company has been making hard drives for the Apple II for some time and sells only direct mail order direct (no dealers), thus keeping overhead down and prices to a minimum. Several owners of the Apple II Cider drive speak well of the product; the company has a reputation for providing friendly knowledgeable service and support.

Davong (version 2.0 Software)

Davong Systems recently filed bankruptcy and sold all of its remaining hard drives at auction. These drives have been reappearing in various reincarnations, used or repackaged in other company's cases for very reasonable prices. The drive was not bad with its version 2.0 software that allowed for partitioning the drive and setting bootup configurations. The drive works adequately and if you can find one for the appropriate price it is worth considering. (The drives were auctioned off for around \$300-\$400 for 10 megabytes and for less depending on quantities; quite a few 33 megabyte drives were also auctioned.) One company in San Jose bought many drives and has been repackaging them into new cabinets and selling them as complete units or selling the software, case and controller electronics for very cheap. You can buy a standard drive mechanism yourself and plug it in. The controller handles up to at least 33 megabytes.

Others in the wings

Several companies including Levco in Del Mar, California, are working on producing SASE controller boards for the Mac (as an add-on to the Levco 2 megabyte internal RAM upgrade board). With these boards you will be able to plug any PC style hard drive mechanism into the controller to have a very inexpensive hard drive (less expensive because of the large market for SASE drives). The new Apple Turbo Finder, turbo file system and new ROMs to be released in early '86 should minimize the need for special software to interface with these drives, and perhaps make them worth considering if they are inexpensive enough.

Multi User Drives

Summary Recommendations

For multiuser drives, stick with the AppleTalk network protocol for the Mac. **File Servers** are much better than **Disk Servers**. A single big disk server for a network application is inefficient and may cost more than adding storage as needed. Several small file servers (or several small disk servers) are more efficient for network applications, more flexible, and you can always add storage as you need it by adding additional servers to the network. The MicroDesign Keeper (File Server) is a good expandable choice for a small network/office system (LaserWriter and 3-6 Macs or MacXLs).

Apple's File server

The file server that Apple announced at the 1985 shareholders meeting seems to have been put on permanent hold. As soon as Apple settles on the AppleTalk file structure, expect several products to come out that use a Mac with one or two Apple HD20 drives as a network file server. InfoServe recently released MacServe, a program that runs on a Mac with a HD20 (or Hyperdrive, Tecmar or Quark) and acts as a disk server for an AppleTalk network and allows several servers can be on the same network. As soon as Apple settles the AFS, InfoServe plans to introduce a true File Server that runs on a similar configuration. Centram Systems is about to release their TOPS file system and file server software that works with AppleTalk and can make any supported device a File Server for the network including a Mac floppy, Mac hard disk, IBM PC, XT or AT and eventually UNIX (see the discussion in new and future product notes).

MicroDesign Keeper

MicroDesign markets their drive as a File Server, as which it works very nicely (although we haven't had the opportunity to do extensive testing yet). The drive itself has all the advantages of the Tecmar drive and looks physically very similar with two slots for fixed and removable cartridge drive combinations. As the first true AppleTalk **File Server** on the market, the MicroDesign does much more. It has a 68008 CPU and 512K of RAM built in to handle the File service; it can run applications locally (like print spooling, mail or other background network tasks). The drive appears to each user as a giant floppy. Multiple people can run the same application simultaneously. The MicroDesign handles file locking so multiple users can work with the same datafile on the disk (the first user can read/write, additional users can read only. The drive can be partitioned into multiple, dynamically-sizing volumes which can be password controlled for security. Up to 32 volumes can be active at once (good for BBS applications); a nice backup utility is provided. The MicroDesign interfaces to the Macs through AppleTalk and can thus easily be connected to a Lisa/MacXL (unlike most of the other hard disks). The drive can be plugged into the network at any point; multiple drives can be added to the network if you want to expand your storage capabilities. The disadvantage of running off of AppleTalk is that the transmission speed is lower than direct connect. The MicroDesign connects at 230K Baud (for the single user) as compared with 500K Baud for the Apple HD20. The Micro Design looks like it's well thought-out and is definitely worth considering for multi-user applications.

Corvus (OmniTalk diskserver Software)

Corvus sells the OmniTalk disk serving software for using an OmniDrive a Disk Server for an AppleTalk network, where one Mac is used to control the disk service for the network. The controlling Mac runs at half its normal speed, and if it is used for anything, the whole network slows down substantially. The OmniTalk software provides nice security and passwording features but it is somewhat buggy and difficult to set up. A network of 8 to 10 Macs each running off of a large Corvus OmniTalk as a **disk server** runs fairly slowly. The slowness is intrinsic to a disk server running over AppleTalk. A disk server is much easier to develop but requires much more traffic to be transmitted over the network. A disk server is acceptable if most of the Macintoshes on the network are running off of a devoted hard disk or RAM disk and only accessing the disk server infrequently. With everyone running programs off of the disk server drive, the whole network slows way down. In this case a file server (or multiple) would be preferred. If you must run off a disk server, you should only do so for small networks 2-5 Macs and LaserWriter or include several disk servers on the network, not currently possible with Corvus. See the comments on the Corvus drive above.

Sunol Disk Server

Sunol systems was unwilling to provide an evaluation unit, so our only experience with one was at the San Francisco Macworld Exposition and at the Berkeley MacFest (where the company was unable to make its drive work with AppleTalk and the Apple LaserWriter after three days of attempts) and by talking with owners. The Sunol drive is currently being marketed as a disk server system for the AppleTalk office system, but Sunol will be announcing a File Server system at the fall COMDEX show. Their file server will not be using the standard AppleTalk File System until it is standardized sometime in January. The file server will probably be upgradeable to the AFP when finalized. The drive is large, noisy, and more expensive than the other systems. The Sunol system allows for multiple volumes but they must be mounted and ejected by using the Sunol volume manager application (volumes may not be ejected or controlled with a desk accessory as we would prefer) The Sunol has an optional large-capacity streaming tape cartridge backup system that fits in the same cabinet as the drive, but the streaming tape is much slower for backups than the cartridge drives. You will be able to daisy-chain multiple drives on the same network; it will provide some sort of file locking for multiuser applications. They are also working on converting to the new turbo finder and file structure. We hope to look into the Sunol further for the Spring 1986 newsletter.

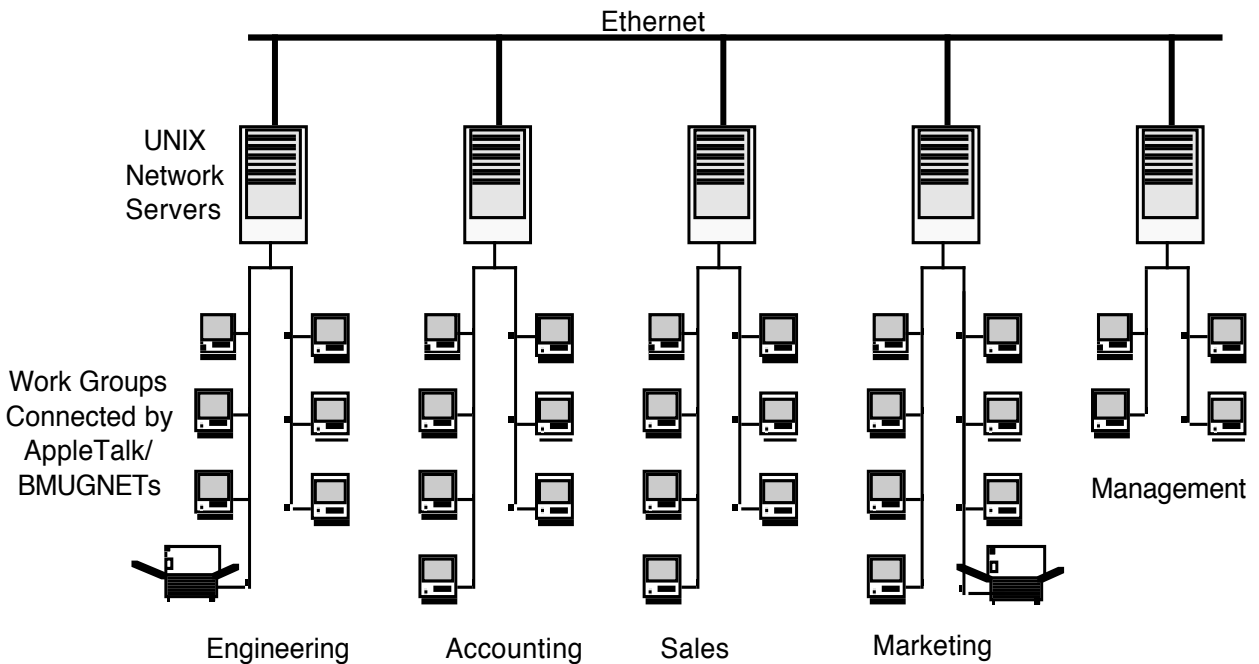
3 Com

This is a relatively expensive file server system for a network of Macs. We haven't had much experience with it yet, but we plan to include it in our review of network servers in the Spring '86 BMUG newsletter.

Lutzky-Baird

This is a sophisticated distributed networking configuration, where a UNIX minicomputer is used as a server for a network of Macintoshes running off of AppleTalk (they actually use a setup like BMUGNET). The UNIX machine provides a central hard disk, file management, access control, E-mail, print spooling, DBase management; it manages record locking for multi-user network applications (like accounting, order-taking etc). Lutzky-Baird provides custom installations for particular specialized businesses requiring multi user setups. Lutzky-Baird is currently working on converting their system software from the current Zilog 8000 UNIX machines to work with DEC VAXs and work with a SUN UNIX supermicro as the network server. If you want a larger setup, several UNIX machines can be networked (through Ethernet), servicing hundreds of Macs organized in work groups under their UNIX server, but able to network (through AppleTalk and the Ethernet) with any other Mac served by any of the other Machines (see figure). The current setup is fairly expensive but can be custom-configured to your particular application. Turnkey systems should be forthcoming.

A Distributed Network with AppleTalk and Ethernet



ImageWriter II Preview

By Raines Cohen

The ImageWriter II's case is similar to that of the Apple //c and the LaserWriter: solid white, slotted, with thin diagonal buttons, and designed (presumably) by Frog Designs. Two large "feet" lift the back of the unit several inches off the table, while the front corners are not elevated, lending a sharp tilt to the unit. This tilt is somewhat disconcerting to the eye at first glance, but is helpful in the operation of the printer. The IW II seems to weigh about as much as the original ImageWriter, and is about the same size, although the "feet" in the back make it seem larger. In addition, the large round areas encasing the roller make it look clunky, rather than elegant. Adding the automatic sheet feeder, which extends several inches into the air, makes the whole thing look much larger than it is. The overall design, however, seemed solid and sturdy.

The printer uses an unusual type of connector: an 8-pin mini-D round male connector, unlike any I've seen before, although I've heard that Apple will be using this type on other products in the future. Of course, a cable to hook up to the Mac is included, but now you can't use your cable to connect to other RS 232-C devices without a special adapter. The cable components are identical to those used in AppleTalk, including the smaller, smoother thumbscrews.

The instruction/reference manual included with the unit is very friendly, indeed. It tries to be fair to users of all types of Apples: Macs, Lisas, the //c and //e, and even the Apple ///! For each computer, it includes setup instructions, sample printing, software examples and screen dumps, as well as other detailed information. All of the various escape codes that allow control over the individual aspects of printing are shown, but there is not much other technical data, such as schematics or pinouts for the expansion bus. The sheetfeeder manual is in four languages, and brief, yet thorough.

Powering up the machine results in a strange sequence of noises, something like: Buuuzzzzz <whirr!> chucka-chucka click-click! Hearing it reminds me of C3PO. This is caused by the machine testing out printhead movement and paper feed movement, as well as moving the ribbon briefly up and down to test the color printing capability. Turning it on with the FormFeed button held down initiates a self-test, which is like that of the IW I, but somewhat improved: It prints a few initial title and status lines revealing the configuration of the internal DIP switches and any expansion options installed before printing the "barberpole" of self-test. You can adjust the print quality during self-test printing, to conveniently switch between draft, medium, and Near Letter Quality (NLQ) modes.

The IW II seemed very IW compatible. "Draft" mode looks just like the standard draft-printing mode on the IW, but runs at 250 CPS (twice as fast!). Standard and high-quality printing on the Mac uses graphics to form characters, so the increase in speed and quality is not as obvious, but it is there. The sheet feeder operates very nicely. The easily-attached device holds up to 100 sheets of paper, which are automatically fed in as needed. From my limited experimentation, however, it seemed that I needed to manually select "Form Feed" between pages in order to feed in paper properly. I had some mechanical difficulties with it at first, but nothing that would affect a regular user.

The tractor-feeding of paper felt much better than on the IW I, as the tractors seemed to grip the paper more securely. There is a slot inside the case (for "dealer installation") for an AppleTalk or 32K memory expansion card, but the manual warns that the latter will not improve performance significantly on the Mac.

As mentioned above, the ribbon can move up and down under software control, for printing different colors on the multi-layer color ribbon. The current Mac software does not support any control of the color, but enhancements are allegedly forthcoming. It was interesting to discover a contact switch in the middle of the ribbon area that was *not* activated by a normal ribbon --- perhaps color ribbons' cases are larger?

We had a few problems printing in Draft mode because the sheet feeder doesn't reverse-feed properly. Envelopes allegedly work well; I was unable to test this in the time I had it. The tractors can fit closer together than they can on the IW I, making it easier to use smaller labels. They still can't come all the way together; the limit is about 2 inches.

Overall, I'd say the IW II is a pretty nice device. Once the Mac software takes full advantage of its capabilities, I'd give my full whole-hearted recommendation to it; for now, though, I'll say "look around and compare."

Hard Disk Drives with Removable Cartridges

A Cautionary Note

© 1985 by Paul H. Gusciora

The Macintosh is limited by the speed and capacity of two floppy disk drives. To expand beyond this, you can now purchase one of several brands of hard disk drive. These hard disks provide fast access to ten megabytes or more.

Copying ten megabytes onto floppies for backup can be a formidable task. It takes 25 single-sided floppies to store ten megabytes. Any information that would be difficult to recreate should be copied to two or three sets of floppies for backup.

At least three manufacturers (Iomega, Micro-Design, Tecmar) sell hard disk drives with removable cartridges for the Macintosh. Each cartridge stores 5 megabytes. The cartridges are removed from the hard disk drive in the same way that the floppy disks are removed from the floppy drives.

At first glance, removable cartridges make backup easier. It takes only two cartridges to store ten megabytes. Unfortunately, there are some hidden problems, prompting this cautionary note.

My experience is with a mini-computer system. The system has a 5 megabyte fixed (non-removable) disk, and a 5 megabyte removable cartridge, but no floppy disk drive. The system is nine years old, but many of the problems are the same with more modern systems. (For hardware buffs, the mini-computer is a Data General NOVA 830 with a Diablo 4234A 15" cartridge disk)

Reliability is the first problem with removable cartridges. No matter what the manufacturers try to imply, hard disk drives with removable cartridges are less reliable than sealed hard disk drives or floppy disk drives. In a hard disk drive, the heads fly above the surface of the disk on a thin film of air. If the heads encounter any dust or dirt, they bounce up, and then crash, causing catastrophic destruction of the disk and the heads. Cigarette smoke particles are large enough to cause head crashes.

Cartridge disks are vulnerable because they expose the disks to a relatively dirty environment. Sealed hard disk drives reduce the risk of head crashes by isolating the disk from contaminating particles. Floppy disk drives do not suffer from head crashes because the heads are in constant contact with the slowly rotating floppy disk.

Compatibility is the next problem with removable cartridge disk drives. Hard disk drive heads are aligned to a fixed reference point relative to the edge of the disk. If the heads shift, the alignment is lost, and the cartridges become unreadable. The heads of a cartridge disk drive can be re-aligned with a special alignment cartridge.

If you are lucky, the heads were correctly aligned when you created your cartridges, and re-alignment will allow it to read all of your cartridges. Otherwise, you will be informed that your disk drive was always out of alignment, but now that it is correctly aligned, it can not read any of your cartridges. Sealed hard disk drives can usually be re-aligned so that it can read all of the data. Floppy disk drives are less critical about alignment.

Because of alignment differences from drive to drive, you should not depend on transporting cartridges from one drive to another, or one system or another. It will probably work most of the time, but floppies will be more reliable.

If your disk system combines a sealed disk drive with a removable cartridge disk drive, you should assume that if **any** part of the system needs repairs:

1. All of the information on the sealed disk drive will be lost. (routine diagnostic tests)
2. The heads will be re-aligned, making all of your cartridges unreadable.
3. If the cartridge drive is replaced, the new drive will not be able to read your cartridges.

In other words, *you can kiss your data goodbye*. If the manufacturer tries to tell you otherwise, ask them to back it up with a warranty that provides \$75/hour for a technician to adjust your drive so that it can read your cartridges.

Maintenance costs are higher on a cartridge disk drive. Many cartridge systems use filters to remove dust from air circulated past the disk. The filters must be replaced periodically.

Cost is another problem. Table 1 shows that removable disk cartridges cost more per megabyte than floppy disks. Of course there is a savings in time, which is an important consideration. But backup copies are useless if they are not perfectly reliable.

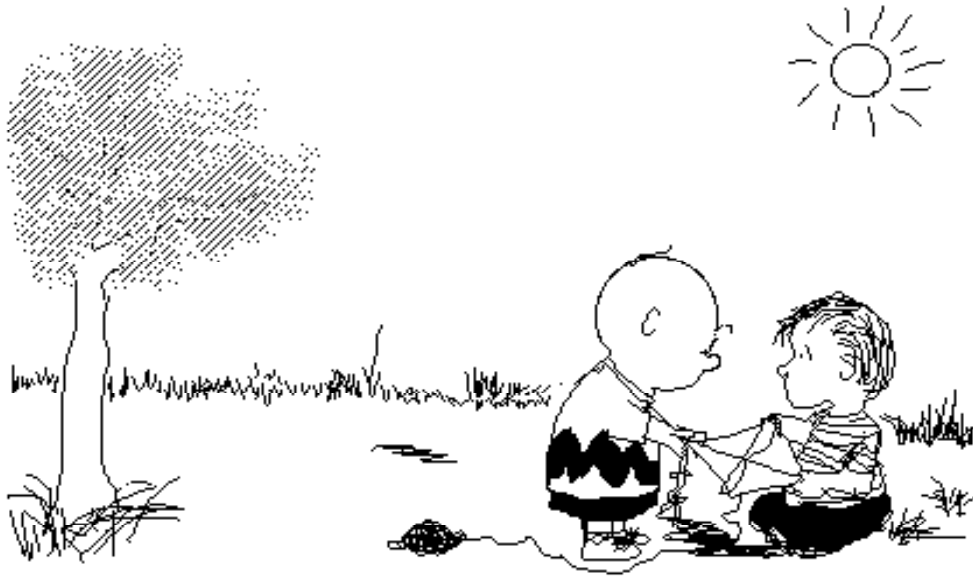
Hard disk drives with removable cartridges can be cost-effective for applications where computer crashes, power failures, or accidental deletion are the most probable loss of data. The removable cartridges provide a convenient way of backing-up and restoring files. For total reliability, make a second copy onto floppy or magnetic tape, in case the hard disk fails.

Most individuals should probably purchase a sealed hard disk, which is more reliable. Use the Macintosh floppy disk drive to make two copies of the files on the hard disk for backup.

Table 1. Comparison of Backup Costs

Disk	Capacity M bytes	Number per 10 Mbytes	Cost \$	Unit Cost \$/M byte
3.5" SS Floppy	0.4	25	3.00	7.50
3.5" DS Floppy	0.8	13	4.00	5.00
Removable Cartridge	5.0	2	55.00	11.00

Paul H. Gusciora is a Ph.D. candidate in Chemical Engineering at the University of California, Berkeley. The title of his dissertation is Adaptive Control of an Autothermal Reactor. He currently has 12 disk cartridges which cannot be read on other disk drives.



Yi Chong Terng

Typewritten pages to Mac Files

Optical Character Readers "For The Rest Of Us"

by Scott Beamer

The purpose of this article is to save Mac users some time if they want to read typewritten or printed material into their computers for editing, database, or whatever. I have spent several months trying to get a thousand page manuscript of a book into my Mac so I can edit it with Word. I've spent a lot of time trying to do it. In fact, it still isn't done, but now I know how to do it, and have a fairly good idea of what it's going to cost.

First of all, forget Thunderscan and Omnireader for the time being. Thunderscan does graphics, not text, and the Omnireader is a nice idea but at this stage of its development it is a balky gadget that is difficult to get any useful work out of.

There are several machines, called OCRs (Optical Character Recognition), that do a surprisingly good job of reading material in, but they all have drawbacks. The usual one is price. The so-called "medium priced" ones (\$5,000 to \$15,000) are fast and fairly accurate, but they only read typewritten material and in a limited number of fonts. The best machine (\$36,500) reads everything...newspapers, magazines, books, you name it, and if it doesn't know the font it'll learn it fast.

Obviously, in this price range, the only chance the casual user has is to find a service bureau that will do the paper file to electronic file conversion for you. Luckily, there are some, though still not many. Another problem is rates. Currently, the rate these companies charge seems to work out at just about the rate a word processor/typist charges. I found typists asking about \$15 per hour and claiming a speed of about ten pages an hour. This gives about \$1.50 a page which is about the rate the OCR service companies are asking. This would suggest that the only reason to opt for an OCR service over a typist is speed. The OCR could deliver my 1,000 page manuscript on disk in less than two days. The typist would be lucky to be done in two and a half weeks. As this seems a very new and rapidly-growing field, both sides are willing to negotiate the rate. By digging through the yellow pages under word processing, data processing, secretarial services, etc. you will find a variety of places offering OCR service, direct word processing entry, or a combination.

As the competition heats up with these OCR service companies, I expect the prices will drop. It pays to phone around. In phoning around, I found the rate range for both OCR services and typists varying by 300%. Think about it, too. Maybe there's an OCR you can get access to and just haven't noticed. Law firms, universities, and typesetting facilities are the most likely places to look. OCRs are available for rent locally as well, but the best deal I've found is a one month rental of a Dest for \$1,000.

Let me run through the hardware in a little more detail. The Thunderscan factory technician told me they feel their machine is not useful for reading text into MacPaint files. Before the text in the MacPaint file will be legible, the original text must be a very large size, way out of the range of typewritten material, and even then you will be faced with the problem of trying to edit text with MacPaint.

The Omnireader is a nice enough concept. But I've had it demonstrated to me on two separate occasions, and it just doesn't work, at least not the way the advertisements suggest it does. What surprises me even more is that the reviews I've read in computer magazines talk about it as if it does work. In MacWorld (August '85, p.112), the author says "You can enter a twenty page manuscript in about five minutes." **What was this guy smoking?** In an hour's worth of fiddling with it, neither of the demonstrators showing me the machine managed to read in a whole typewritten page, not even their demo sheet. I would estimate an experienced operator would be lucky to average five minutes a page, and there would be no chance of keeping that rate up to enter twenty pages in even one hundred minutes. The error rate was horrible as well, probably averaging something like twenty per cent. The machine accepts only four font/point combinations (albeit the four most popular in business). If you're curious or adventurous, or have some special use in mind, go give it a try.

But what I'm waiting for is the Omnireader sensor head to be attached to the Imagewriter in the way Thunderscan has done it. To cut costs and make a machine people could afford, Oberon made the Omnireader without an electric paper feed system. The mechanical system they have devised is, I feel, inadequate and is the principal cause of their high error rate. Using the Imagewriter for an automatic, electric feed ought to reduce the error rate and increase the scan speed dramatically. Yet, when I was talking to the factory by phone, the P.R. person refused to admit that any such thought was being considered by them. (Thunderscan said the same thing). If neither of these companies is developing this product, then someone else is sure to do it. The potential is just too obvious.

The most popular of the "mid-priced" copiers is the Dest. It comes as a desk top unit, and is the height of simplicity to use. It reads a dozen fonts and does so without your having to coach it. The company claims its various models read between 140 and 240 pages per hour of double spaced, typewritten text. This is the machine you are most likely to find being used for OCR in an office or service bureau.

The Kurzweil is the Rolls Royce of OCR systems. In fact, it doesn't even call itself an OCR, preferring instead the moniker "ISS"(Intelligent Scanning System). I feel it earns the title. While OCRs digitize the character they wish to identify, then compare it to their font library for identification, the Kurzweil uses its artificial intelligence software to analyze the shape of each character. This enables it to automatically read almost any typed or printed material, including material with varying fonts and type sizes. It handles italics, bold, underline, etc. and in 6 to 24 point type. If it is uncertain about a character, it highlights it in context on the CRT, prints a pixel map of the character about two inches high, and shows its best guess. It then awaits your prompting of how it should read the character, and will remember your prompt the next time, too. While the machine awaits your prompt, it is not slacking off. It is scanning on ahead, taking advantage of its 30,000 character buffer.

The machine is large, about the size of a mini computer, and includes its own keyboard and CRT. There is also an electronic tablet that allows you to trace the parts of a page you wish scanned. This is essentially a way to allow you to read text that snakes around a picture. It also allows you to set up column formats for scanning, or to scan part of a page. The machine also prescans every page so that it does not waste time scanning large white areas (including margins). It is not as fast as the Dest, but its claimed speed of about 65 pages per hour should be fast enough for most.

The biggest problem with the Kurzweil is finding one. The Western sales representative gave me the names of the three in Northern California owned by service oriented companies, that is, companies that will take in outside work for a fee. All of them appear to have other services, such as typesetting, as their principle businesses. Imagemakers in Sunnyvale has a Mac in the office, so they can deliver the final product directly on Sony disks. They will work by mail or modem as well. The second, in San Francisco, is still in a start up phase, but have their Kurzweil networked with their Macs. The third, in Grass Valley, I haven't contacted.

For those who would like some meatier reading on this subject, *PC* magazine's July 9, 1985 issue has a thirty page series of articles covering the field of OCRs.

There is one other type machine worth mentioning. These are represented by the DATACOPY model 700 Word Image Processing System (WIPS) and the EIT personal scanner. These are actually a collection of hard and soft ware that are currently only partially released. They are designed to be run interfaced to an IBM PC XT or AT. The final product intends to provide the capability of optically scanning a page of mixed text and graphics (or either separately) into electronic files for editing and reassembly. They do this by scanning the page in through a desk top scanner (a kissing cousin of a desk top copier). The replica of the page is on the screen in a few seconds, but is not legible. First, the operator must electronically lift the graphics off the screen, then pass the text to the word processing program. The image can be edited separately. The software provides an expanded range of Macintosh like features including a mouse, cut and paste, zoom, rotate, scaling, etc. I saw the Datacopy machine demonstrated in this form. The operator was agile with it. The list price is about \$4,000 excluding the PC. But more interesting is the promise that the currently inadequate OCR unit (it only recognizes two fonts) will be upgraded, first with the addition of ten more optional fonts at \$195 each. Then, promised for "later this year" is a learning mode character recognition system. This will cost another \$2,000 but will have the ability to learn any (typewritten) font. It will also allow adjustment of the program in which the scanning speed can be increased, while sacrificing some accuracy. The demonstrator insisted his

company had no plans of becoming Mac compatible. This I can understand as most of the features are already available on the Mac, but some of the others I would very much like to have.

After I had submitted this article for the first time, I was contacted by a factory representative for another OCR, released August 1, a competitor of DATACOPY. This is the EIT (Electronic Information Technology) Personal Scanner. I have not seen it, but the factory claims its superiority to the Datacopy in price (2,495 vs. \$3,500 and "soon to be released" font learning software \$500 vs. \$1,995 for Datacopy), fonts (six come with the unit vs. 2 for Datacopy), speed (one page read and recorded to disc in 30 seconds vs. 4 1/2 minutes for Datacopy), minimal size of PC required (256K PC vs. XT or AT for Datacopy). Among the other advantages of the EIT mentioned were its ability to compress data, and do it while scanning to disk, and it can scan for a target word processing program such as Wordstar and retain some formatting. The biggest disappointment about the EIT besides not seeing a demonstration, was to learn that it will not be released in a Mac compatible form. The factory representative stated this while admitting that he has a Mac in his office and it is his preferred computer. He went on to explain that even with Apple's help, he was unable to discover enough market to justify the reworking costs of his product.

OCRs are no longer gadgets (see exception above) or novelty items at computer shows. They are now a standard business machine found in an ever increasing number of offices. However, because of price, OCR "for the rest of us" remains a matter of dealing with a local service bureau. At current rates, typists often have more to offer Mac users, being able to deliver the material formatted and with pencilled corrections or notes edited in. Also, OCRs are sensitive to copy quality, even more so than typists. All OCRs for the Mac deliver text only files with hard returns at the end of every line.

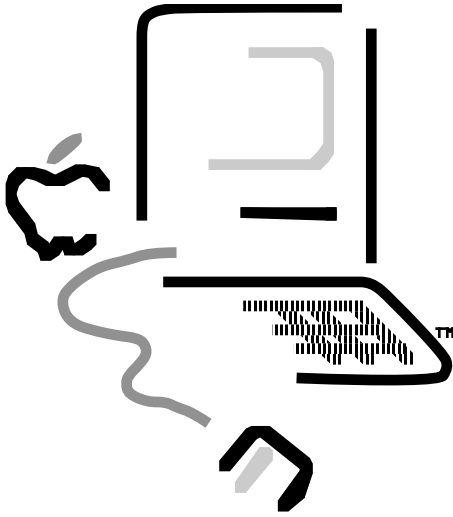
Yet, with all the array of OCR software and hardware on the market and in development, it seems only a matter of time until a really spectacular package is available for the Mac, able to read pages of words and images together at high speed and reasonable cost. Certainly all the features of Datacopy are available to the Mac developer. Datacopy's trademark is "The Eye of the Computer." Thunderscan gave the Macintosh eyes. OCRs make it literate, but who's going to come up with the reasonably priced package for the Mac that will read in text and graphics together?

SERVICE COMPANIES (that have a Kurzweil):

Imagemakers, Inc. (408) 245-2660
Sunnyvale
Contact: Tom Yarr

Quadratype (415) 621-0220
San Francisco
Contact: Mani Feniger

American Microtel (916) 346-8148
Grass Valley



The MonsterMac™ Two Megabyte System

by John Sawyer

The MonsterMac™ two megabyte system for the Macintosh, designed by Levco Enterprises of San Diego, is an internal add-on board which brings the Mac's total memory, as the name suggests, to two megabytes, and also adds some capabilities that are new to the Macintosh. The extra memory is fully recognized by any version of the Macintosh Finder, with no modifications required to it or to the System. It conforms to all of Apple's guidelines for memory expansion beyond 512 kilobytes, and there is no incompatibility with either the present Macintosh system ROMs or with future versions.

The question is sometimes asked, "What do you need two megabytes for? Shouldn't 512 kilobytes be enough, and at most, shouldn't one megabyte be more than enough?" The answer is that whereas for most applications you don't need two megabytes (in fact, there are no programs for the Mac at present which require two megabytes), you can make good use of two megabytes in many programs. For some programs, having two megabytes can mean the difference between painfully sluggish operation, and lightning-like (by comparison) performance. Some database and spreadsheet programs use all the memory they can get to work at their fastest when they do a sort operation. Microsoft's File™, for instance, has been clocked at 150 seconds for one complex sort operation on a large database (200K), while accessing the data file in 512K mode from a floppy disk; that time was reduced to 50 seconds when working in two meg mode, still working from the floppy disk, and further reduced to 30 seconds when the data file was placed in a RAMdisk in two megabyte mode. The benefits of two megabytes become even more apparent when you're working with more complex configurations consisting of several programs in memory at once (either in a RAMdisk or under Switcher) along with each program's data files in a RAMdisk. Placing the system file in a RAMdisk is also useful because the system utility routines and desk accessories will run faster since you don't have to wait for them to load off of a physical disk; it also allows the installation of many more fonts, and larger fonts, than can physically fit on a floppy disk. In two meg mode, Assimilation Process's Mac•Memory•Disk™ allows you to create a RAMdisk of up to 1.852 megabytes. While one megabyte, and to a limited extent, even 512 kilobytes, will also allow you to create similar configurations and speed up operations, they do present a ceiling which for many users is easily reached, preventing them from doing all they might wish to do. Some people might find that even two megabytes presents an inconvenient ceiling, and that still more memory would be desirable, but at present there is a space limitation on how many more memory IC's can be practically added to a Mac's internals. When memory IC's with greater storage capacity are available, adding memory beyond two megabytes will become practical.

Other big users of memory are compiled languages such as C, and Lisp, a language for artificial intelligence. Lisp's capabilities expand as the available memory expands. In any compiled language, the compilers and libraries are often so large that repeated access to them on disk, even hard disk, slows down compilation. Compilation can take several minutes when the utilities are accessed from a floppy. Placing them in a large RAMdisk speeds things up tremendously. One timing test done on a typical compilation, comparing speed with all utilities on a Hyperdrive™ installed on a separate Mac, then in a 512K RAMdisk in a MonsterMac, yielded these results: Hyperdrive: 60 seconds; RAMdisk, 30 seconds. This speed increase comes in handy when doing many compilations during a typical programming session.

One new feature that the MonsterMac board adds is the first parallel bus expansion connector for the Mac, which allows circuit cards of various functions to be plugged in, such as coprocessors, hard drive interfaces, more memory (within limits), color graphics generators, and so forth. This connector is a single 64-pin, 2-row standard DIN card connector, mounted on the Monster board, into which one expansion card can be plugged. It contains all the necessary signals from the 68000. At present it is accessible only inside the Mac--its signals are not brought outside the Mac's case--though hardware-oriented users may choose to figure out the best way to do that for themselves if they need external access.

Another new feature is sockets for EPROMs (Erasable, Programmable Read-Only Memory) which can contain programs that will not be erased when power is turned off. These EPROMs can contain up to 128 kilobytes. In the future, Levco will make available EPROMs containing utility programs, though users may also be able to place their own programs in EPROM. One program the EPROMs can contain is the boot software for hard drives, which normally

comes on a floppy disk. This allows any hard drive to be made auto-bootable without the need to insert its startup disk. Another utility currently being worked on is a program that creates a RAMdisk of up to 1.5 megabytes which does not lose its contents if the Mac crashes and forces you to hit reset. It's possible that this program will also allow you to boot from this RAMdisk. This is of particular benefit to program developers who sometimes experience crashes while working on a program--it would allow them to keep a copy of the last version of their program in the RAMdisk, along with utilities used in program development, so that the developer doesn't have to go through the tedious process of reloading those files after each crash, and also would do away with the need to reboot from a floppy each time--the programmer can thus recover quickly and continue with the creative process with minimal interruption. An EPROM-based debugger is another possibility. There are sockets for four EPROMs; in each MonsterMac, two of these sockets are used for the Monster's own system EPROMs, which at powerup or reset move the screen portion of memory to the appropriate place in RAM, depending on the mode chosen (two meg or 512K mode), as well as other memory management chores. They also provide a visual indication of the two megabyte installation--under the 'insert disk' icon in the center of the Mac's screen at powerup, they place three small square boxes to indicate how many banks of extra RAM are installed. The full two meg uses three added banks of 512K, so all three boxes contain a black dot in a fully-configured MonsterMac. If only two or one of the extra banks are installed, only two or one of the boxes will contain a black dot. Also, when you boot a disk, the smiling Mac icon now has a pair of fangs as a further visual indicator for those times when you boot a disk so quickly that the initial disk icon doesn't appear, and also for comic relief.

Another change made by the Monster's system EPROMs is that 'mouse-eject' (the method for ejecting a disk from the drive by shutting the power off, then turning it back on, or by hitting reset, while holding the mouse button down) now works for the external drive as well as the internal drive, so that the disk in the external drive doesn't boot up when you don't want it to. The MonsterMac's EPROMs make these changes after the Mac's system ROMs have done their work at powerup--in fact, they are triggered by the Mac's own ROMs: after the Mac's ROMs have done their startup work, they pass control to a location in ROM space which is normally used only during the testing process at the Macintosh assembly plant by a special set of test ROMs which are temporarily connected to the motherboard. This is where the MonsterMac's system EPROM space begins--once control is passed to them, they can accomplish their tasks. If the remaining two EPROM sockets--the user EPROM--are occupied, control can then be passed to them, so that any programs they contain, such as the utilities described above, can be run. Finally, control is passed back to the Mac's operating system.

The MonsterMac's extra 1.5 megabytes runs at the maximum clock speed of 7.8 MHz, which is faster than the Mac's motherboard 512K, which runs at about 6 MHz. The reason is that it doesn't have to operate in synch with the screen portion of memory--in the Macintosh, the video circuitry and the microprocessor share the same RAM banks, so that they alternate in their accesses of that RAM. While the video circuitry is accessing that RAM, the microprocessor cannot--it has to wait its turn--and so the microprocessor's effective speed is slowed to about 6 MHz. Since the bank of 1.5 megabytes that the MonsterMac adds is not needed for video memory, the microprocessor can access it without waiting for video accesses; this increases the effective speed to about 7.8 MHz. The MonsterMac system EPROMs also rearrange memory map pointers so that the application heap begins in the extra, faster 1.5 megabytes instead of in the motherboard memory. This speeds up many programs which are calculation-intensive, such as spreadsheets, databases, compilers, and so forth.

One problem encountered with memory upgrades beyond 512K is the proper placement of the portion of memory which contains the screen image. Apple has stated that screen memory should be located at the top of RAM memory, no matter what that total memory may be (from 128K to four megabytes), instead of at a fixed location. This is the case with 128K and 512K Macs--screen memory is located at the top of their respective total RAM space. One way to think of it is that screen memory should float like a cork to the top of whatever memory is installed in the Mac, and programs should find out where the screen is by referring to pointers maintained by the system software. This creates a contiguous application space--one that is unbroken from its top to its bottom. However, some memory upgrades beyond 512K don't move the screen memory to the top of RAM--they keep the screen at the 512K mark, and place the extra memory above this, so that the total application space straddles the screen memory--the screen sits fixed in the middle of the application space. This split application space is more complicated for programs to deal with than a contiguous block of memory; some programs use their own memory manager which can't deal with fixed blocks. Most programs have no problem dealing with this split arrangement since they occupy only the lower 512K anyway, but some others do have problems since they make use of the extra memory, and they expect to see the proper memory

arrangement when they're working in an environment greater than 512K; if they don't find it, they don't work correctly, if at all.

These programs must be run in an environment in which the screen has been properly moved to the top of memory. One such program is Filemaker™. Yet another problem arises when the screen has been moved to the top of memory: some programs don't run correctly in this properly-arranged environment, since they were written without taking into account memory sizes larger than 512K and their associated relocated screen. They violate Apple's guidelines--they do a simple test to see if the Mac is a 128K, and if it's not, they assume it's a 512K, and thus assume that screen memory is at the 512K mark. They then place some or all of their screen image data into that specific location. That screen data will thus not be displayed properly on Macs with memory greater than 512K which look for the screen at the top of memory. These programs must be run in an environment in which the screen is fixed at the 128K or 512K mark. Most programs now obey Apple's guidelines regarding screen memory, but many have been written that violate them in this manner. Some of these are Dollars and \$ense™ (at least its opening screen), the Macintosh Development System (commissioned by Apple but not written entirely to Apple's guidelines), and Mazewars. Pre-release versions of some programs such as Jazz™ and Excel™ have other problems running in such an environment, but remember that you take your chances with pre-release--unfinished--software. The actual release versions of these programs work properly in environments above 512K.

The desk accessory, 'The Bug', which creates an animated image of a small bug which repeatedly crawls from the bottom to the top of the Mac screen, also doesn't work--it will still create a bug, but not one that is visible since it is placed where it thinks the screen is supposed to be, which will now possibly contain program code or data instead. This invisible bug thus walks across your program or data, and may eventually cause a bomb. The solution to this is provided in the MonsterMac system, which lets you configure the Mac as either a true two megabytes, with the screen at the top of two megabytes, or as a true 512K, with the screen at the top of 512K. In the 512K mode, the extra 1.5 megabyte is not presently available for use (though there are plans to make it available in the form of a protected RAMdisk as described earlier), thus making the MonsterMac's 512K mode a true Apple 512K mode on which all 128K/512K-only programs will run. This mode is entered by pressing the interrupt button (provided on the misnamed "programmer's switch") at the moment you hear the "bong" when you turn the Mac on, or after you hit the reset button (either the physical button or a reset button in an alert box onscreen). First-release versions of some programs, such as Crunch™, OverVue™, and Microsoft File™, don't make use of any memory greater than 512K. However, the second-release versions of these programs do make use of the extra memory, though a few still make direct use of only one megabyte even when their memory usage meters indicate that 1.9 megabytes is available for use. These programs can still benefit from two megabytes by the use of RAMdisks, disk cache programs such as Turbocontrol™ and MacBooster™, and by being installed under Switcher with other programs, to fill up the remaining memory.

The MonsterMac board is installed by clipping out and discarding the Mac's 68000 processor and putting a socket in its place, into which mounting pins on the Monster board are inserted. A replacement 68000 is provided on the Monster board. This method of connection allows the Monster board to gain access to the parallel address, data and control buses of the microprocessor so that they can be brought out to the expansion connector and to the board's extra memory and control circuitry. A small, silent piezoelectric fan is installed between the disk drive and the power/video board to handle the extra five degrees F. generated by the Monster board. In fact, the MonsterMac runs as cool as, and in some cases cooler than a 128K Mac. The Monster board does not require the use of a separate power supply since its current drain on the Mac's power supply is within the Mac's safety margins. Total current drain of the Monster board is 500 milliamps average, 600 milliamps maximum. This presents no problems in driving any peripherals that may be attached to the Mac's I/O ports as long as those peripherals obey Apple's guidelines for current drain. Two months of daily use of a MonsterMac at CJS Systems, with an external Apple floppy drive (sometimes a Corvus 45 megabyte hard drive), an Imagewriter, and sometimes an Apple modem connected and turned on at the same time have not uncovered any power supply or overheating problems. Four programmable array logic IC's ('PAL's) handle various functions such as timing for the new memory. On the MonsterMac board, they are labeled Groucho, Harpo, Chico, and Zeppo. A fifth space is reserved for Gummo, but as usual, he's not really there. He may be put to use some time in the future to add more functions to the MonsterMac.

John Sawyer is a partner in CJS Systems in Berkeley, CA, which is the San Francisco Bay Area dealer for the MonsterMac and other Levco products.

MacMEGABYTES™ One-Meg Upgrade

by David L. Foster

Now why would anybody want or need an entire megabyte of internal memory (RAM) in their Macintosh? Frankly, until this summer there really wasn't that much need (or supply for that matter). However, this year's torrent of new Macintosh software has included the debut of a new generation of programs that were designed with 512K of memory in mind. Many of these programs load both themselves as well as their data files into memory, thereby sidestepping the Mac's performance bottleneck — its agonizingly slow disk access. If you've had the opportunity to see the amazing speed of RAM-based programs like ProVUE's OverVUE™ or Filemaker™ from Forethought, you'll understand the kind of performance boost that can be attained using this approach. This increased speed is accompanied, however, by a new limitation. The more data you want to deal with, the more memory you'll need. For example, although second generation Mac spreadsheets like Excel and Crunch promise millions of cells for data entry, the limited amount of memory available after these programs are loaded into a 512K Mac will prevent you from using even a single percent of these programs' theoretical capacity. Integrated or multi-functional programs like Lotus's Jazz are also severely limited within the memory constraints of 512K, and programs as large as Jazz or Crunch cannot be usefully integrated with other programs using Switcher.

Upgrading your Macintosh to a megabyte of internal memory quickly eliminates these restrictions. Furthermore, having this much memory provides you with greater flexibility in the way you can configure the Macintosh to match your specific needs or working habits. Beck-Tech, located in the Claremont Hotel in Berkeley, was one of the first firms to offer a true one megabyte expansion option. Their upgrade, MacMEGABYTES, is installed by Beck-Tech certified dealers by mounting a single daughter board holding an additional 512K of memory on top of the main logic board. The upgrade in no way affects the external appearance of the Mac and it does not require a cooling fan. Although the upgrade currently voids Apple's 90 day warranty, Beck-Tech offers a warranty of equivalent length for their upgrade. Once the Beck-Tech board is installed a contiguous megabyte of RAM becomes available for program use. The fact that the memory is contiguous or uninterrupted is important, because some programs require this in order to operate correctly. Switcher is one example. Beck-Tech accomplished this feat by modifying the Apple ROM memory chips so that the Mac could recognize and properly use the full megabyte of memory on boot-up. No user actions are necessary to make the entire megabyte of RAM available to the system or to programs. Thus you can boot the Macintosh with any disk and special software is completely unnecessary. Earlier this year, MacMEGABYTES was packaged either with or without the ROM enhancement — if you buy this upgrade make sure you get the modified Apple ROM with it.

The MacMEGABYTE upgrade includes some RAM disk software, but it is buggy and easily outperformed by a program that is in the public domain (RAMStart on BMUG disk #3). Several commercial RAM disks are also available, but few offer much more than RAMStart.

Compatibility with Software

I tested the Beck-Tech upgrade with a large number of programs (Table I) over the course of a three month period and found only a handful of programs which refused to run properly (Table II). Most of these are games, demonstrations, or in the case of the bug desk accessory, jokes. Most games run fine, but if any are especially important to you, try them out on an upgraded machine before buying. All of the major business programs like Word, Excel, Helix, and Jazz performed without any problems (or at least any related to their being run in a 1024K machine). Furthermore, these programs made full use of the extra memory. For example, OverVUE could manipulate a 850K data file. This corresponds to an address file with about 15,000 entries. Jazz has more than twice the amount of memory available for its use than it does in a 512K machine. Pre-release versions of Microsoft Excel handled nearly 40,000 cells!

With a megabyte of memory Switcher can be used to load up to eight programs into memory at once. In practice few people ever need this many programs loaded into memory at one time; however, an extra 512K of memory allows you to simultaneously load several large programs requiring lots of memory — a situation that is often very desirable. For example, Excel lacks word processing and telecommunication capabilities that can easily be provided in a one megabyte Mac by using Switcher. Although you can load Excel and Word together on a 512K machine, very little memory remains for building spreadsheets, graphs, or documents.

An added bonus for those without a hard drive is the fact that RAM cache programs like TurboCharger and MacBooster are fully compatible with Switcher. RAM cache programs like these monitor your disk usage and keep an image in RAM of the disk sectors that are used most often. Data and programs held in RAM can be accessed much faster than from floppy disks. Consequently, the performance of each program loaded with Switcher is enhanced in proportion to the amount that you use it. Thus, by using Switcher together with a disk cache you can work with several programs and have them all perform nearly as fast as if they were being run from a RAM disk. Keep in mind that until double-sided disk drives are available you may be forced into a bit of disk swapping.

Reliability

It would appear that Beck-Tech's upgrade is highly reliable. A MacMEGABYTES equipped Macintosh served as BMUG's BBS for a period of nearly three months running nearly 24 hours a day with no problems or down time related to the upgrade. Currently this Macintosh is used for conducting software demonstrations at BMUG's weekly meetings on the Berkeley campus. Likewise, I experienced no problems with my machine over roughly the same time period. In addition, several friends of mine with MacMEGABYTE upgrades have also had no problems. In their August 12 issue *InfoWorld* also reported no reliability problems in their review of MacMEGABYTES.

MacMEGABYTES is compatible with the Hyperdrive.

Beck-Tech announced an adapter kit at the Boston *MacWorld* Expo that makes MacMEGABYTES compatible with General Computer's Hyperdrive — currently the only internal hard drive available for the Mac. The kit only works with the new style "clip-on" Hyperdrive product and will be \$100 - \$200. I had a chance to use Beck-Tech's first prototype for several days and found their 1024K Hyperdrive to be the Macintosh I've been longing for.

Compatibility with Future Apple ROM Updates

A widespread apprehension in getting your Macintosh upgraded by third party suppliers like Beck-Tech is: will the expansion be compatible with the expected ROM upgrade from Apple? The answer is that no one can be certain one way or the other until the new ROMs are released. People who *should* know tell me that the upgrade will seek out and recognize whatever memory is available. Steve Beck, president of Beck-Tech, says his company has tested early versions of the new ROMs and found them to recognize all 1024K of their upgrade in a contiguous fashion. Whatever the case, Beck-Tech intends to make their upgrade compatible should it fail to work with the new ROMs.

While on this point, it is widely rumored that Apple is not exactly happy with Beck-Tech modifying the Mac's ROM. According to Steve Beck, Beck-Tech is currently negotiating with Apple to obtain an official endorsement, much like that given to General Computer's Hyperdrive, that allows users to install MacMEGABYTES without voiding the Apple warranty. As of mid-October, this issue still remained unsettled. [Note: any modification to the Macintosh Beard voids the warentee.] Apple has apparently been too busy over the summer reorganizing itself to develop any consistent policy with respect to third party developers like Beck-Tech. In my opinion, Apple's discontinuation of the one megabyte Macintosh XL (Lisa) without providing a high performance machine to take its place was a clear invitation for others to provide the Mac with more memory. Apple should suck in its pride and selectively accommodate upgraders with high quality products like MacMEGABYTES. They filled a vacuum that Apple knowingly created.

Conclusion

I recommend MacMEGABYTES as a reliable, high quality one megabyte upgrade. Not everyone needs a full megabyte of memory. Whether you need it or can simply afford it, you'll certainly enjoy your Mac more once you have it. This upgrade (particularly in combination with the Hyperdrive) gives the Mac the muscle to take the PC head on and win.

Table 1: software that works fine with MacMEGABYTES

MS-BASIC 2.0, CHART 1.0, CONCERTWARE PLUS 2.0, COPY II MAC 4.2, CRUNCH 1.0, EDIT 1.0, EXCEL, FACTFINDER 1.0, FILE 1.0, FILEMAKER 1.0, HELIX 2.0b, JAZZ 1.0, MAC DISK CATALOG 1.0, MAC MEMORY DISK 1.0, MACDRAFT 1.0, MACDRAW 1.7, MACPAINT 1.5, MACPROJECT 1.0, MACTERMINAL 1.1, MACWRITE 4.5, MULTIPLAN 1.02, MUSICWORKS 3.0, OVERVUE 2.0b, PAGEMAKER 1.0, RESOURCE EDITOR 0.7, SIDEKICK 1.0, SMOOTHALKER 2.0, SPELLER 1.2, STATWORKS 1.0, SWITCHER 4.4, THINKTANK 1.1, TURBOCHARGER 1.11, VERSATERM 1.52, VIDEOWORKS 1.0, WORD 1.05.

Table 2: software that doesn't work with MacMEGABYTES:

Fall 1985 BMU G Newsletter

BUG (desk accessory), DELTA SQUADRON, FROGGER, MAZEWAR, MONKEY BUSINESS, TUMBLING MAC DEMO.

MacRecorder

A Speech digitizer for the Macintosh

by Ty Shipman

SMASH!!!! BOOoom!!!! OOF! RRoorrr....

These would be great sounds coming from the Macintosh in those great "run-time" games everyone is coming out with. Better yet, it would be great just to say **Open**, start talking, and have the Mac get the latest version of your life story you have been writing (hoping to make a million). At any point, why type? You could just speak into a microphone and have it become readable. Well, it's not too far into the future. Speech synthesis and voice recognition systems are hot in the electronics industry today. Did you realize that if you call you local information operator, a synthesized voice will say the number you wish—after a human tells the computer what number to say. But, in time, I think the person will be gone as well.

Back to those great sound effects you want in your next version of Kill the Enemy or Capture the Flag. Who wants to spend hundreds of hours (or dollars) trying to code those sounds? — NOT I. Well, with the help of the MacRecorder voice Digitizer, you can add them in only the time it takes to say SMASH!!!! BOOoom!!!! OOF! RRoorrr....

For a long time programmers, teachers, scientists, and many others have wanted to drive computers and their applications by voice and receive voice output instead of black and white dots on a 9-inch screen. MacRecorder allows you to do it. You can make the Macintosh respond through a voice command system and/or other general analog waves. You may also avoid those beeps telling you your disk is full. Instead, you computer will say "please change disks: this disk is full."

This project was developed to take sound (music, voice, or noise) and put it into a form the Mac can understand. Then digitizer can replay the sound, or any segment of it, utilizing the Mac's sound chip. You can also display the wave form on the screen.

HOW IT WORKS

The conversion of sound, which is an analog wave, to a digital sequence is called (you guessed it) analog to digital conversion (A to D). See Figures 1 and 2. A to D conversion is done by taking a reference point on the analog wave, usually the lowest point and calling it zero (0). The highest point on the wave is assigned a number, for example 255 (you'll see later why I choose that number). This assignment from high to low gives the computer the ability to recognize 256 different values.

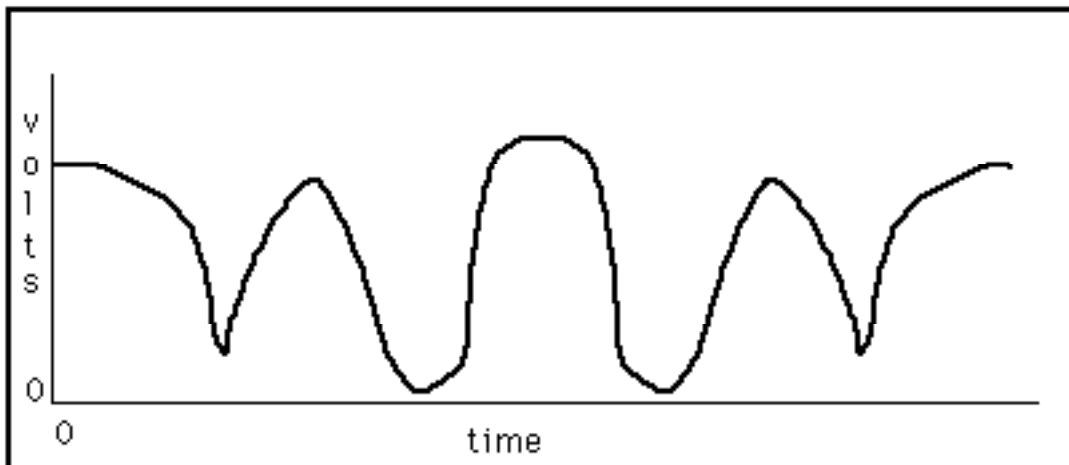


Figure 1: Simple analog wave starting at time zero.

When the computer sees a number like five (5), it sees a sequence of 1's (on) and 0's (off). Thus 5 to us looks like 0000101 (8 bit format) to the computer. This is known as a binary number, because it has only two values for each digit: one and zero.

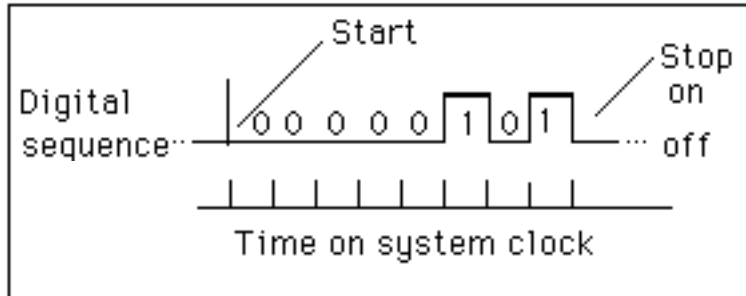


Figure 2: A digital sequence which represents the number 5 in binary (8-bit format)

Sampling an analog wave with a digitizer yields an intensity value consisting of a number ranging from 0 to 255 (Figure 3). With these numbers the computer can reconstruct both what the wave looked like, on the screen, and what it sounded like through the sound chip. The only limitation to a perfect reconstruction of the wave is the sample rate and the range of numbers that represent the intensity at the sample point. Essentially, the more samples taken, the better the reconstruction; the fewer samples, the poorer the reconstruction of the wave.

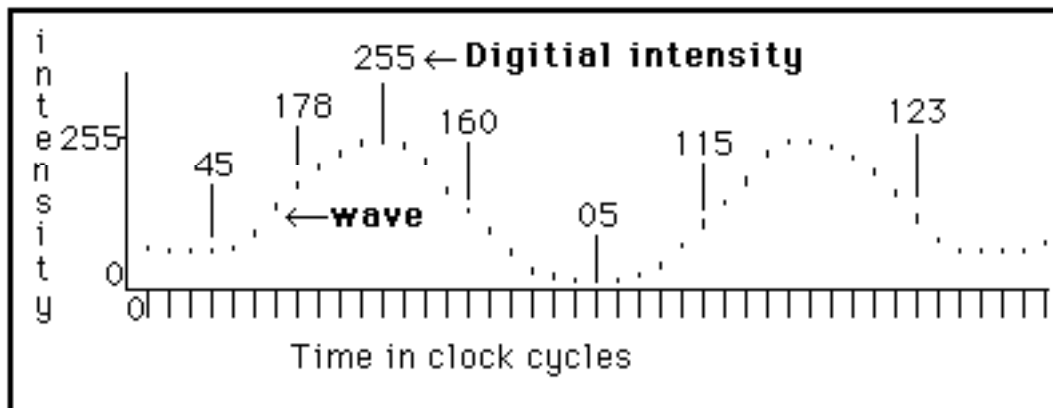


Figure 3: A sample of an analog wave showing digital conversions

Our project, using a sampling rate of 9321 samples per second and an 8 bit format (digital values 0 to 255) turn the Macintosh into a low fidelity tape recorder. This allows the Mac to reproduce sounds similar to those coming from your car's AM radio. The amount of time and quality that is recorded is very dependent on the speed of the software and memory size in your Macintosh. The 128K Mac with the current software records about 3 seconds of audio data. The 512K Macintosh records about 45 seconds of sound. The software is written in MacForth from Creative Solutions. A version written in C is planned.

THE HARDWARE

The MacRecorder is designed around an analog to digital converter made by National Semiconductor, the ADC0831 (Figure 4). This chip samples the sound wave intensity at V_{in+} and converts it into an 8-bit format (0-255). The digital output depends on the difference between V_{in-} (ground in this case) and V_{ref} (5 volts in this version). The A to D conversion starts when the chip select (**CS** U6;pin1) goes low. See Figure 5. This causes a start bit (low) to be sent out **D0** (U6;pin6). Following this start bit the eight data bits are sent, most significant first. The end of the transmission for that sample point is when the **CS** goes high, sending a stop bit (high). This is a forced high stop bit,

which is accomplished by the pull up resistor (R22). From **D0** the data is sent to the RS422 serial driver (U7;pin2), μ A9638c. This line driver is used to buffer both the data and the synchronizing signal, supplied by U4 pin 9.

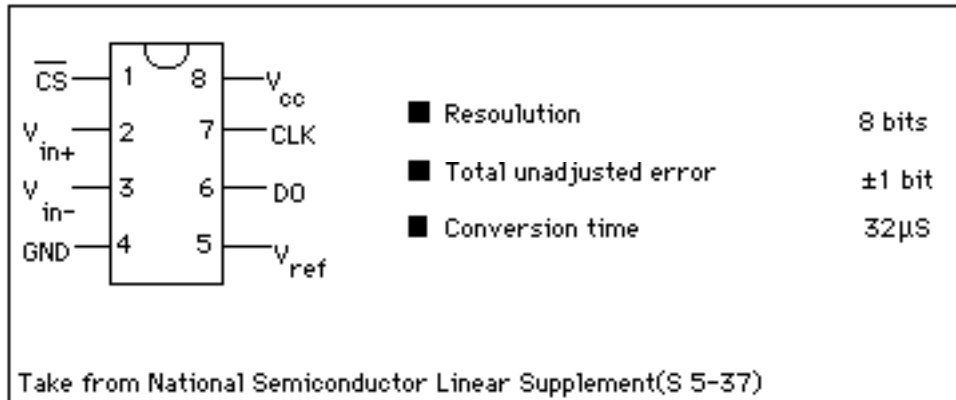


Figure 4: The heart of the MacRecorder

The data is sent to the Macintosh at a rate of 111,860 baud. This is so fast that we have also supplied a synchronizing signal to the Mac through pin 6;U7. The remaining circuitry is divided into two functions. The clocking portion (in Figure 5) is used to control the ADC. The portion used to preprocess the sound (in Figure 6) presents its output to the ADC (U6;pin 2).

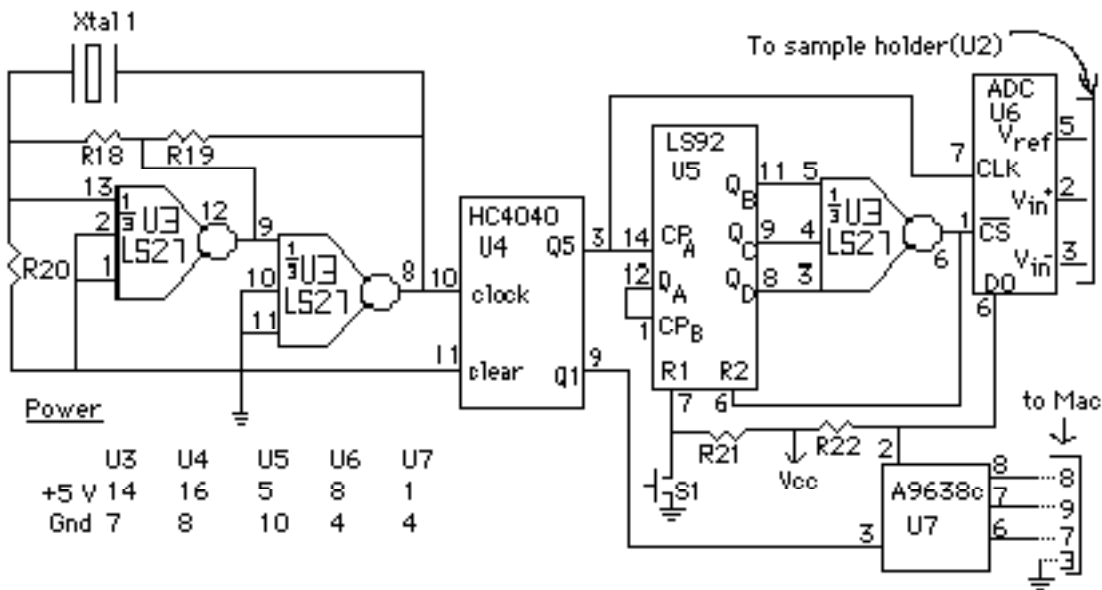


Figure 5: High speed serial A/D converter

The clock is made up of two NOR gates in the 74LS27 (U3) and a standard U.S. color burst crystal, 3.5795 MHz. This master clock frequency is divided by 2 in the 74HC4040 (U4) and output is at Q1 (pin 9;U4). This is the synchronizing signal that is used to drive the Serial Communications Controller (SCC) in the Mac (16 times the baud rate). Again the Master clock frequency is divided, this time by 32 with output through Q5 (pin3;U4). This signal is the actual transmission rate of the data. The signal at Q5 is also sent to the 74LS92 (U5), a divide-by-twelve counter. The output from this chip is sent to the last NOR gate on U3 which controls the **CS** line to the ADC. The actual **CS** line controls are as follows: two cycles high and ten cycles low. One low cycle is used for setup, another is for the start bit (low) and eight cycles are used for the eight data bits. One high cycle is used for the stop bit (high) and the other for setup. Thus the sample rate at this speed is about 9321 samples per second or 74,568 bits per second.

The analog preprocessing (Figure 6) is done by the LM324 (U1), a quad op amp. The input is supplied by a standard Electret microphone and amplified by two op amps. Then the analog signal is sent to a third op amp that is configured as a low pass filter. This filter cuts out all frequencies above one-half the sample rate to prevent frequency aliasing. The high gain of the preprocessor is achieved by having the signal swing from 0 volts (V_{in-} is ground) to 5 volts (V_{ref}). Because of the swing from 0 to 5 volts the power supply to the LM324 must be 9 volts. This is a constraint of the LM324. The artificial ground for the voltage divider ($V_{ref}/2$) is supplied by the two 10K 1% resistors (R16,R17). From the low pass filter the signal is passed to an analog switch, the CD4066 (U2). This chip forms a sample and hold switch with the .022 μ f capacitor (C9).

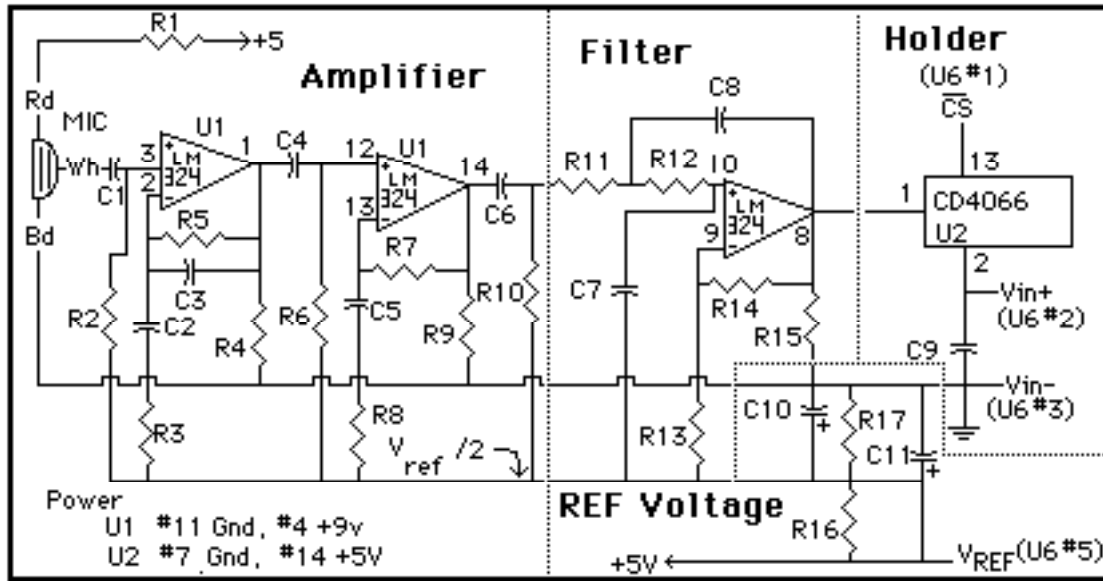


Figure 6: Amplifier, filter, and sample holder for voice digitizer

You might decide to replace all chips with their CMOS counterparts to run off the Mac's power. However, I advise against this! If you short out **any part** of the board, you may ruin your entire computer.

In case you were confused by of this "techie" stuff, a simple block diagram (Figure 7) explains it all. Really, it is this simple.

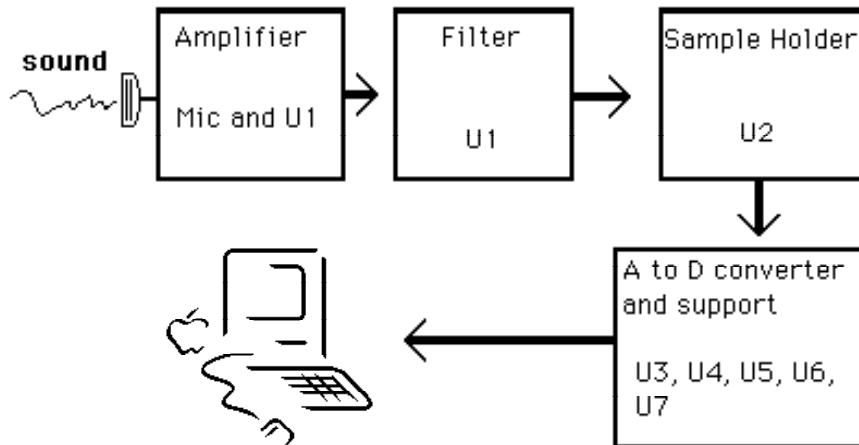


Figure 7: Block diagram of MacRecorder

The power supply (Figure 8) for the MacRecorder is a simple 9 volt DC wall transformer (P1), similar to a calculator power supply, plus some onboard regulation to supply 5 volts to the digital chips. A 220 μ F capacitor (C12) helps keep down those voltage ripples that would interfere with the A to D conversion. At this point, the line is split to supply the LM324 (U1) and the 5 volt regulator (U8), a 7805.

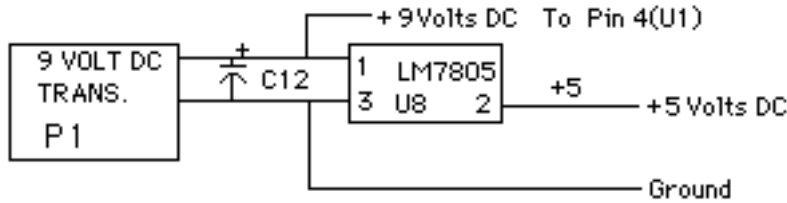


Figure 8: Power supply for MacRecorder

Parts list for MacRecorder

Resistors:

1/2 watt 5% carbon unless specified

R1, R21, R22	1 K Ω	R10,	10 K Ω
R2	5.6 K Ω	R11	11 K Ω
R3	1.1 K Ω	R12	6.2 K Ω
R4, R8, R9, R15	4.7 K Ω	R13, R20	22 K Ω
R5	220 K Ω	R14	68 K Ω
R6	100 K Ω	R16, R17	10 K Ω 1%
R7	91 K Ω	R18, R19	1.8 K Ω

Capacitors:

50 WVDC

C1, C7	0.01 μ F	C9	0.022 μ F
C2, C4, C5, C6	0.1 μ F	C10, C11	100 μ F 25V
C3	200 pF	C12	220 μ F 25V
C8	0.0033 μ F		

Semiconductors:

U1	LM324	U5	74LS92
U2	CD4066	U6	ADC0831
U3	74LS27	U7	μ A9638c
U4	74HC4040	U8	LM7805

Miscellaneous:

Xtal1	U.S. color burst crystal 3.5795 MHz
P.C. board	Available from BMUG
P1	9 volt DC power supply (at least 300mA)
Mic	Electret Microphone (Radio Shack #270-092) or equiv.
S1	Normally open momentary contact Switch
S2	STSP toggle switch (not shown power on/off)
Connectors	Power supply type, DB9 Male (better Joy Stick)

Fall 1985 BMU G Newsletter

IC Sockets	extender by Radio Shack), Microphone connector(s)
Cabinet	if your microphone is in a housing
Supplies	2 eight pin, 1 sixteen pin, 4 fourteen pin
	at least 20cm(L) x 14cm(W) x 8cm(H)
	solder, iron, cutters, wire (for jumpers),...etc.

Other ideas for the MacRecorder

If you're the adventurous type you may want to hook up straight into the ADC0831. Here is how. You must put 10KΩ pots on both the V_{in-} and the V_{ref} (Figure 9). You should not let V_{in+} fall below 1 volt. If this happens the ADC will incur a lot more error than the ± 1 bit. You will have to amplify the input through an op amp. The reprint of the data contains ideas on how to hook up sensors to the ADC0831.

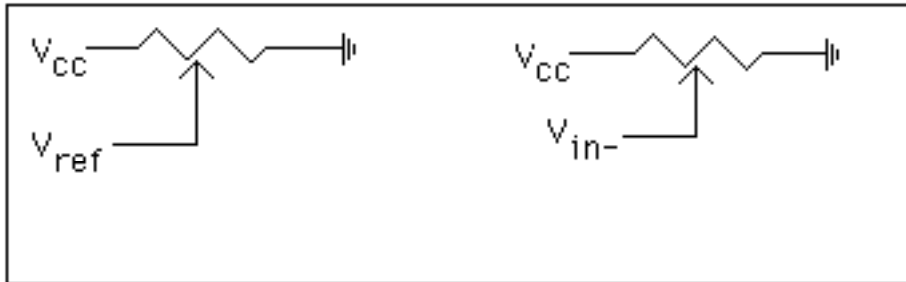
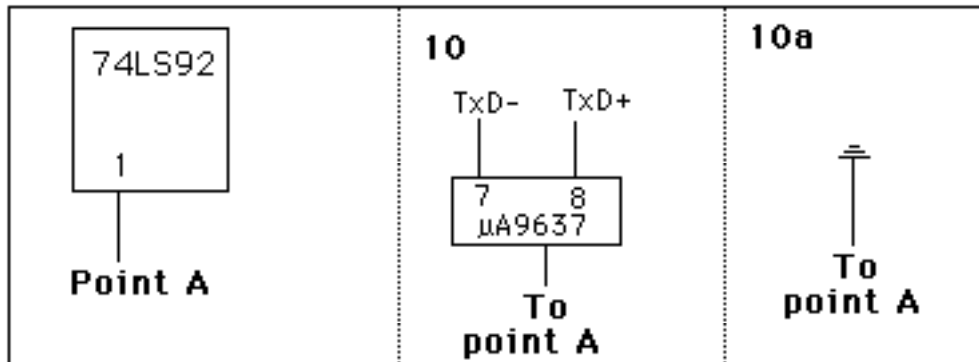


Figure 9: Hooking 10KΩ pots into ADC0831 for adjustable analog inputs

Also, if you want remote control over the sending portion of the system, do it through the software, taking only the sample you want. If you're really hard up you can add a $\mu A9637$, which is an RS-422 receiver. See Figure 10 for the exact procedure. Then you just send a 00H (hex) out the serial port and the receiver will do the rest. I must mention that this hasn't yet been tried, so you may have to experiment. To set up continuous data transmission, see Figure 10a.



Figures 10 and 10a: Remote polling and continuous polling of MacRecorder

Another improvement on the circuit is to replace the crystal with one of a higher value, such as 4.9152 MHz. This increases the sample rate to 12800 samples per second. You would also have to modify the software to handle the increased speed (modify Snd.rate in the FORTH version). Figure 11 is a list of the pinout for the Macintosh's serial port. If you decide to modify the circuit for auto polling you'll need it.

M A C	1	2	3	4	5	6	7	8	9
U S E	GND	GND		TXD-		HSK		RXD-	
		+5V		TXD+		+12V		RXD+	

HSK = sync line for external timing

Figure 11: Macintosh RS-422 serial port configuration

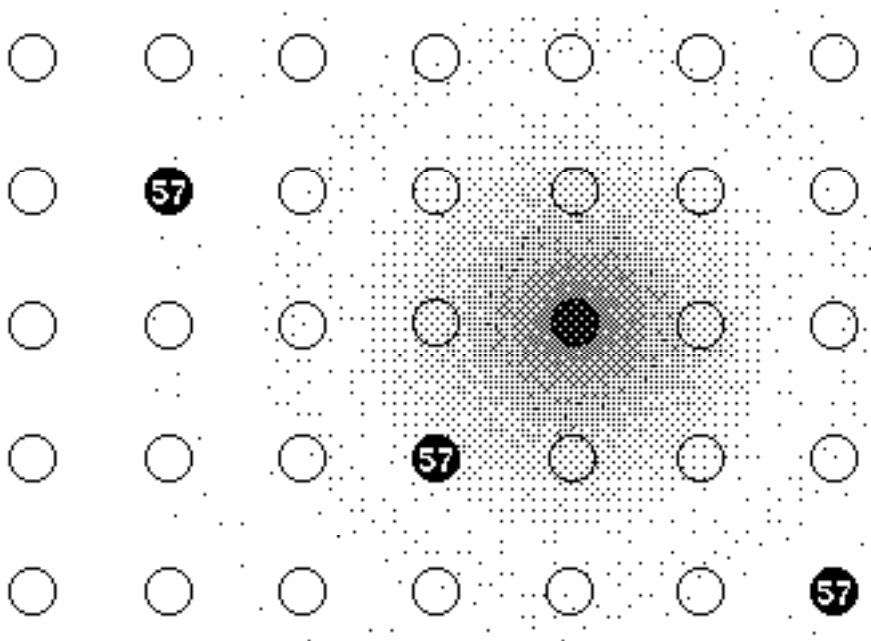
Conclusion

If you are a nontechnical person, this kit is for you too. It is not hard to build. With its step-by-step construction book, you need only a steady hand and some patience. I encourage all who want to experiment with a voice recognition system to buy the kit and build it. Once again: no technical knowledge of electronics is necessary.

The MacRecorder expands the range of applications for the Mac. One could modify the operating system to take input from the MacRecorder. The MacRecorder is also an option for disabled people currently not benefitting from the computer revolution because it is too time consuming to type all the commands or to control the mouse.

The original designer of the MacRecorder is Michael Lamoureux. Michael is currently at the University California, Berkeley working towards a Ph.D. in mathematics.

Michael and I are designing a full 4 or 8 channel A to D converter, along with a 4 or 8 channel D to A converter with controls. Look for it in the Spring '86 BMUG Newsletter.



Ruderman-Kittel-Kasuyda-Yosida Electron Spin Density Disturbance

Brent Fultz

MacRecorder II

An improved voice digitizer for the Macintosh

© 1985 by Michael Lamoureux

Warning: This article is unabashedly technical. Briefly, it describes an improved version of the MacRecorder voice digitizer that was introduced in Ty Shipman's article. This circuit is simpler than the one Ty described, with added features such as volume control and improved noise characteristics. By the time you read this, BMUG should be offering a complete kit for the MacRecorder II. At this point, the software for the MacRecorder turns your Mac into a two thousand dollar audio cassette recorder. (That's \$2,000 for the Mac, not the kit!) We're hoping that by distributing this inexpensive circuit, other applications will be developed, such as voice recognition systems, video digitizers, specialized lab equipment and so forth. Moreover, by using this as a standard interface for A to D conversion, wide distribution of many applications is possible.

Introduction

The MacRecorder voice digitizer described in the last article by Ty Shipman is in fact just one circuit in a long series of digitizers I've built for the Macintosh. That particular version has a number of problems, most notably the need for a dual power supply (i.e. 9 volts to the op amp chip, and 5 volts to the digital chips), the use of a low performance sample/hold circuit (the 4066 with capacitor), and the overall complexity of the circuit. In addition, the low pass filter circuit cuts off useful frequencies while the A to D converter chip (the ADC0831) introduces noise to the sampled waveform due to its "total sampling error" of 1 significant bit. (Total sampling error is a characteristic of the converter chip, and can be thought of as random noise.) Nevertheless, it is a useful and instructive circuit with remarkably good fidelity.

However, I couldn't let a good thing sit, so I redesigned the MacRecorder to come up with the MacRecorder II. This circuit is based on a Texas Instruments chip, the TLC549, which is an 8 bit serial A to D converter, similar to the ADC0831 used in the original MacRecorder. However, the TI chip has a total sampling error of only .5 significant bits, which means less noise. It also has a built-in sample/hold circuit which eliminates the 4066 chip from the circuit. Moreover, the microphone amplifier has been redesigned to include a gain adjustment (i.e. volume control) and the filter now has the right frequency cutoff near 4800 Hz, that is, at one-half the sample rate of 9600Hz. Looking ahead to other applications, this circuit can easily be upgraded to a faster sampling rate simply by changing the crystal. In fact, I've tested it at a sampling rate of 19200Hz and found no problems.

By the way, if you haven't read Ty's article on the MacRecorder yet, I recommend you do so now, as I won't repeat here the basics of A to D conversion and the MacRecorder hardware which he covers very well.

About The Circuit

Figure 1 shows the digital part of the MacRecorder II. The 74HC4060 is a 14 stage binary counter and oscillator which provides the master clock signals for the circuit. Pin 9 of the 74HC4060 sends a clock signal to the Mac serial port through the μ A9638 line driver (i.e. buffer). The 74HC20 generates "start bits" and "stop bits" for producing the standard asynchronous communications format to the Mac's serial port. The TLC549 is the A to D converter which accepts analog data at pin 2 (Analog In), and outputs binary data at pin 6 (D0), which is buffered by the μ A9638 and sent to the Mac. R4 and R5 are 10 turn potentiometers (variable resistors) that are adjusted to set the reference voltages Ref+ and Ref- on the TLC549. For proper operation of the circuit, R4 must be adjusted so that Ref- is at 1.5 volts, and R5 adjusted so that Ref+ is at 3.5 volts. (The TLC549 measures the Analog In voltage relative to these two reference voltages. The values mentioned are ones I've found useful.)

Figure 2 describes the analog portion of the voice digitizer. Resistors R7 and R8 form a voltage divider to give an artificial ground for the op amps. The first op amp is a high gain amplifier which boosts the signal from the microphone. Note that the potentiometer R11 is the gain control, which is adjusted to set the sensitivity of the microphone (to set for recording either whispers or only shouts). The second op amp is configured into a low pass two pole Chebyshev filter with a cutoff frequency of 4660 Hz and a pass band ripple width of .5 dB. This stage only passes

sound with frequencies less than roughly one-half the sample rate while blocking higher frequencies. The .5 dB PRW is a measure of how uniformly this stage passes the lower frequencies. (.5 dB is pretty good.)

One thing missing on the MacRecorder II is a push button to start the conversion process. In fact, the circuit is continuously sending information to the Macintosh, and it is the job of the software to accept or ignore the data. The advantage of this method is that the hardware is now under software control, which means everything can be controlled with the mouse: user friendly!

By the way, this circuit is easily adapted to digitizing other voltage sources. Just use the circuit in Figure 1, and send the voltage to be sampled to Pin2 on the TLC549 (Analog In). Use R4 and R5 to set Ref- and Ref+ to the minimum and maximum voltage that you expect to measure on Pin2. Keep in mind that Ref- and Ref+ must lie between Gnd and Vcc (5 volts), and for accuracy in the conversion, Ref+ should be at least one volt greater than Ref-.

Finally, in case you wish to try other sample rates, note that we have the formula:

$$\text{SAMPLE RATE} = \text{CRYSTAL FREQUENCY}/256$$

and so by simply changing the crystal, you can change the sample rate. However, the data sheets for the TLC549 suggest that you can't expect it to work at sample rates over 29000Hz, while the 74HC4060 may not oscillate reliably with crystal frequencies much over 4MHz.

Circuit Availability

BMUG offers the MacRecorder II as a kit, which includes a printed circuit board, power supply and cables to connect it to your Mac. It costs \$45, including the "cassette recorder" software. If you have any questions or suggestions about the circuit, you can easily contact me by mail (care of BMUG), or send messages to my electronic mailbox in the BMUG BBS. In later newsletters we will cover further applications of the MacRecorder.

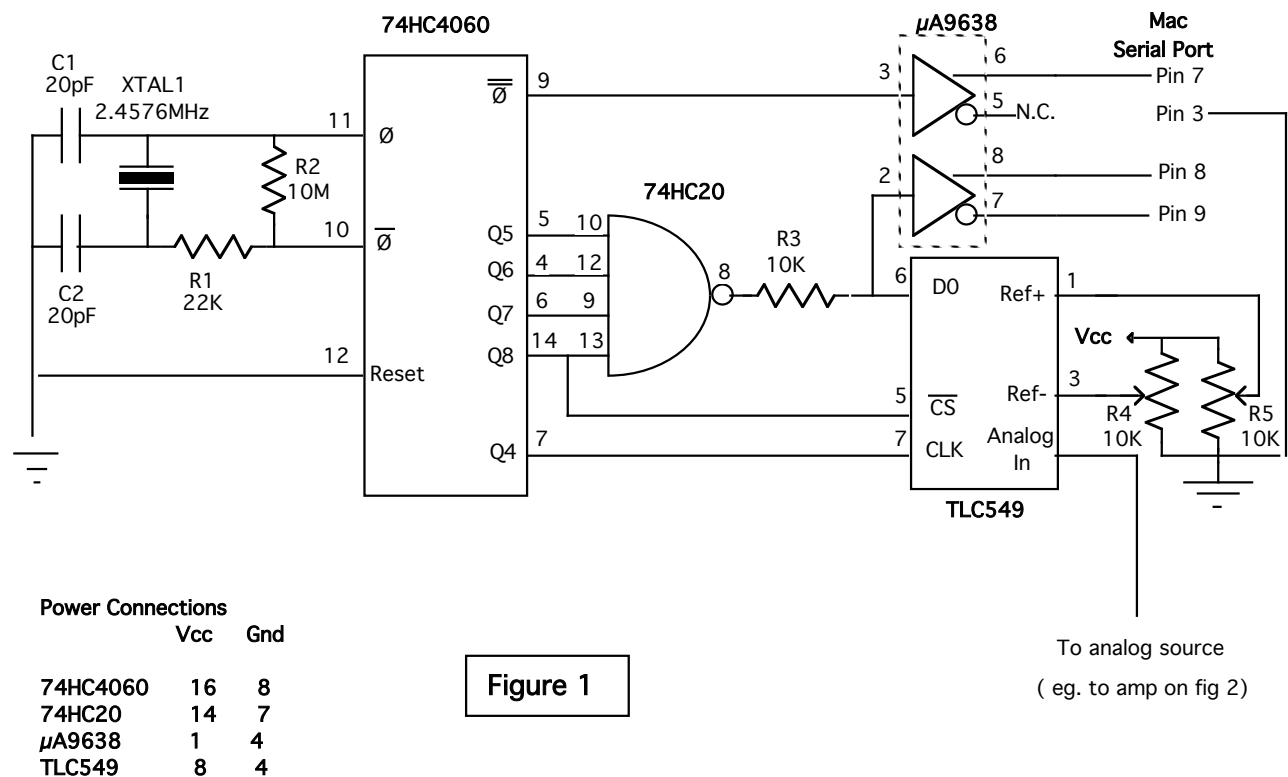
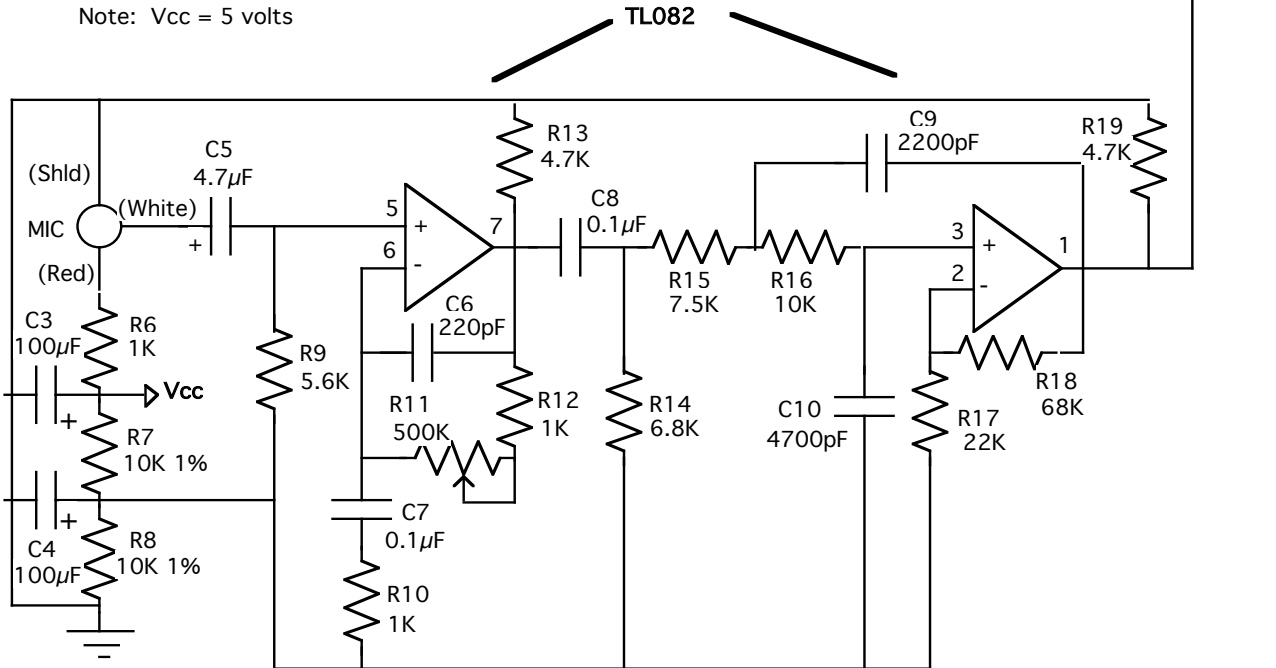


Figure 2

Power Connections
 Vcc Gnd
 TL082 8 4

Note: Vcc = 5 volts



Parts List for the MacRecorder II

Capacitors (all at least 9 WVDC)

- C1,C2 20 pF
- *C3,C4 100 μ F electrolytic
- *C5 4.7 μ F electrolytic
- *C6 220 pF
- *C7,C8 0.1 μ F
- *C9 2200 pF (use a good quality Mylar or similar)
- *C10 4700 pF (ditto)

Integrated Circuits

- 74HC4060 14 stage binary counter with oscillator
- 74HC20 Dual 4 input NAND gate
- μ A9638 Dual RS-422 line driver
- TLC549 8 bit serial A-D converter (Texas Instruments)
- *TL082 Dual biFET op amp
- 7805 5-volt power regulator (not shown)

Resistors (all 1/4 watt, 5% unless otherwise specified)

- R1,R17 22K
- R2 10M
- R3 10K
- R4,R5 10K, 10-turn potentiometer
- *R6,R10,R12 1K
- *R7,R8 10K, 1% precision resistor
- *R9 5.6K
- *R11 500K potentiometer
- *R13,R19 4.7K
- *R14 6.8K
- *R15 7.5K
- *R16 10K
- *R18 68K

Miscellaneous

- XTAL1 2.4576 MHz crystal
- *MIC Radio Shack #270-092 Electret condenser mike element
- DB-9 connector (male)

Note: parts with a * are those used only in the microphone amp and filter (Figure 2).

The MacRecorder Software

by Michael Lamoureux

Introduction

The MacRecorder II is a hardware device used to digitize voice waveforms and pass them to the Macintosh through the serial port. The software that accompanies the MacRecorder II basically turns your Macintosh into a cassette tape recorder, allowing you to record sounds, play them back and save the recordings to disk. In addition, some editing features are provided, to allow you to "doctor" the recording by adding or deleting sounds, or rearranging their order, in a manner similar to MacWrite. This concept of the Mac as a mixed tape recorder/word processor will make the controls more intuitive to use.

Description

I'll begin by describing the typical Macintosh screen displayed by the MacRecorder, Figure 1 below. This screen was reached by doubling clicking on the MacRecorder icon, and opening (using the FILE menu) a data file called Lamoureux.9600. There is one window on the screen, which is given the title of the file opened. The most prominent feature of the window is the graphics display, which is an amplitude verses time representation of the voice waveform stored in the opened file. The horizontal axis is the time axis, while vertically is the amplitude of the wave at each instant. Usually the stored waveform is bigger than what can fit on the screen, so the scroll bar along the bottom of the window is used to scroll along the entire waveform, just as one scrolls through a document in MacWrite. Above the graphics area are five controls. **Play**, **Record** and **Stop** work exactly as you would expect them to (more on **Record** later), while **RePitch** and the sliding control next to it are used to adjust the playback speed of the recording, just like a speed control on a tape recorder. **RePitch** resets the sliding pitch control to the normal setting.

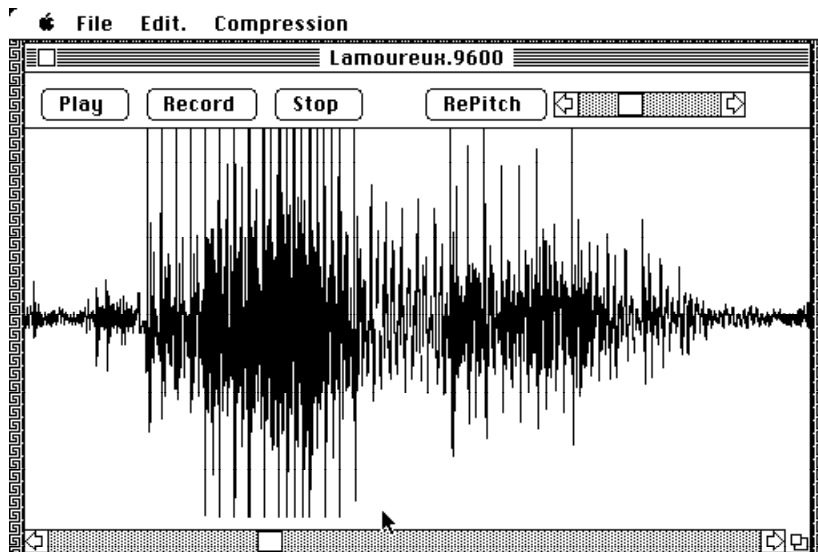


Figure 1

A user can select a portion of the waveform by clicking and dragging with the mouse, just as one selects text with MacWrite. Of course, the waveform scrolls along as you select by dragging off the screen, while mouse clicks with the shift key held down extends the region of selection. Figure 2 shows a selected region of the waveform, as indicated by highlighting. Pushing the **Play** button now would play only the selected region. Pushing **Record** would record over the selected region, erasing it in the process. If no region is selected, the insertion point for the next **Record**, or beginning point for playback is indicated by a vertical bar on the screen.

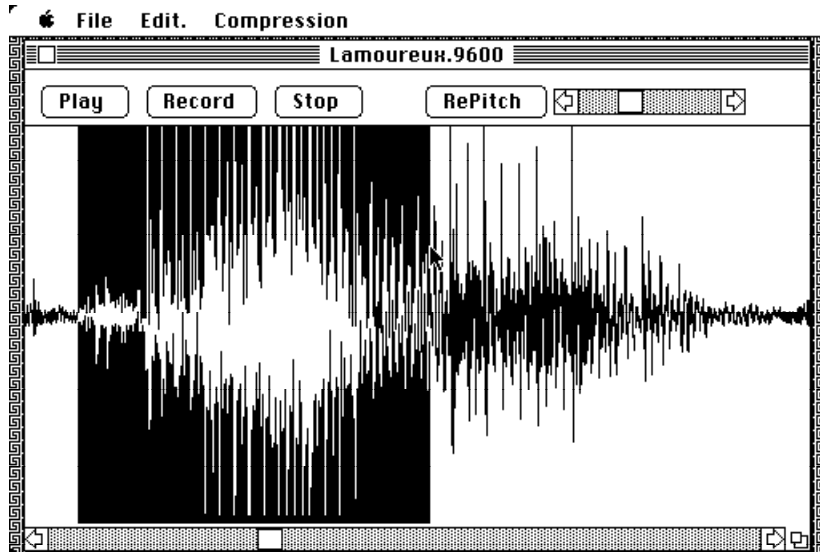


Figure 2

The **Record** button has some special features now which allow the user to set the total length of recording time. When you hit **Record**, a new window opens, titled **Recording**, as shown in Figure 3. On it are two push buttons, **Start** and **Stop**, a message indicating the total length of sound to be recorded, and a sliding control for you to set this length. (On a 128K Mac, the maximum recording time is just over 3 seconds, while the 512K ("Fat") Mac can record over 40 seconds worth of sound.) Once the recording length is set, you hit **Start** to begin the actual recording. Thus, the **Record** button only sets up the machine for recording, while the **Start** button actually "gets the tape rolling".

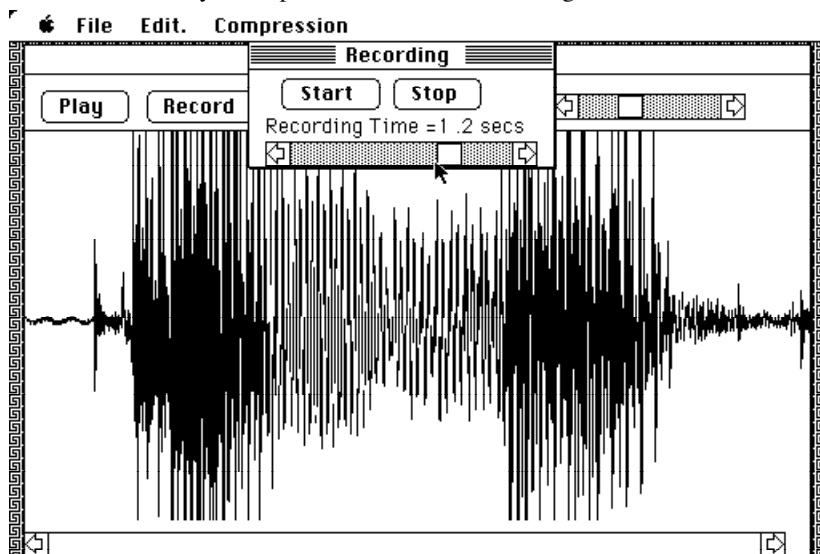


Figure 3

Of the four menus at the top of the screen, three are familiar. The **Apple** menu contains the usual desk accessories, plus some information about the program. **File** allows you to save and retrieve waveforms on the disk, and to quit the program. **Edit** provides for cutting and pasting operations. Under the **Edit** menu is also a **Reverse** item, which reverses the order of a selected waveform, so at playback, the sound comes out backwards (my brother's suggestion, and it's fun). The fourth menu is the **Compression** menu, which simply formats the graphics display in order to condense the waveform so that more of it fits on the screen. Note that **Compression** does nothing to change the sound of the waveform. Think of it as changing the font size in MacWrite, so that more of the document can be seen on the screen.

Using the Software

The easiest way to become familiar with the software is to open a data file that is already on the disk, and experiment with **Playing** the waveform, scrolling along it, selecting and **Reversing** sections, **Clearing** out selected sections, and adjusting the pitch. You don't need to have the hardware installed to do this, and since this is Shareware, you are free to use it to send and receive spoken messages to and from other people.

To use the **Record** command, you must have the MacRecorder hardware. First, close any file you now have open and select **New** under the **File** menu, to start fresh. Attach the 9 pin plug from the MacRecorder to the modem port on the Mac, and plug in the power supply for the MacRecorder. Hit **Record**, and then under the **Recording** window, set the **Recording Time** to a few seconds by sliding the thumb on the slide control. Then hit **Start**, and speak into the MacRecorder microphone. The **Recording** window disappears when the recording is done, and the newly entered waveform appears on the screen, highlighted to show it is the selected waveform. Now hit **Play** to hear what you said. At this point, you can edit the waveform, save it to disk, or add more sound by hitting **Record** again (but keep in mind that a selected region is erased and recorded over. Think about how MacWrite replaces selected regions with typed-in text and you will see why this program records over selected regions).

General Problems

Since the whole waveform you wish to play must be kept in memory, the amount of memory in your Mac is a big restriction. It doesn't take long to fill up all the memory at this recorder's 10K Bytes per second. On a 128K Mac, 3 seconds can be recorded, which is about a sentence, while on a Fat Mac, the 40 seconds you have amounts to a lot of talking. Doesn't this remind you of good old MacWrite version 1.0?

Program Listing

The following pages are listings of the MacRecorder software for reference. The software comes with the kits, and is also available on BMUG disk #4 (MacForth Disk). The listing is the latest version as of this writing — 0.8 — which doesn't implement all editing features and has a few problems.

Conclusion

At this point, a software reviewer usually tries to give an objective summary on the good and bad points of a product, and a rough idea of how useful it is. However, as the designer of both the software and the hardware for the MacRecorder, I don't feel the least bit objective. However, I do know that at both the MacWorld Expo in Boston this summer, and the PC Faire in San Francisco this fall, people were very excited about this "Macintosh tape recorder". I also know it's nice having your own digitizer in your home, especially if you have any interest at all in linguistics, voice recognition or voice synthesis, and this device is a convenient replacement for the minicomputers usually used for those jobs. Games programmers will find it useful in adding special sound effects to their software. Computer dating services could use it to keep a sound recording of prospective partners! Probably the most useful thing I've found is that the **MacRecorder** has made me very aware of how I speak, letting me hear right away which syllables I drop, how my voice sounds when I'm tired, and so on. I'm sure you can think of applications too: as another Mac speech program concluded in its demo, "just imagine ... the possibilities! "

How to engorge your MAC

Going Beyond 512K

© 1985 by Paul Campbell

Introduction

What follows is a brief outline on how to expand your Mac from 512K to 2 or 4Mb. I am not publishing complete do-it-yourself instructions. This process is not for the technically unsophisticated. If you have done your own FatMac upgrade and can follow what I present below, this article should provide you with enough information to increase your Mac's memory further. For background, I recommend you read references [2] and [3] at the end of this article.

At current prices of memory it becomes economically feasible to replace normal disk drives with dynamic RAM; 512K of RAMdisk costs less than \$100 (compared to a floppy drive at \$400-500). Of course, the improvements in performance are remarkable. As someone who uses systems with hard disks all the time, the Mac floppies are depressingly slow. Trying to use a commonly available software development systems on a standard 512K Mac can be a real exercise in frustration. You haven't seen anything until you see your compiler running off a RAMdisk!

At present, my Mac has 1Mb, consisting of the normal 512K plus a 512K bank of memory RAMdisk. I have two RAMdisk programs on different disks. At boot time, one loads the driver, and creates and initializes the RAMdisk like a normal RAMdisk. The other just loads the driver without initializing the disk. When I have a crash during software development I use the second RAMdisk to reboot my system. This way I don't lose my RAMdisk. Restart takes only a minute or so. This method does have the risk of a runaway program trashing the RAMdisk, but in practice this has not yet been a problem (more about this later).

Basic Dynamic RAM Theory

Before getting into the details of the Mac RAM system I would like to go over the basics of how dynamic RAMs are used. Understanding the Mac innards requires this background. Those who already understand this may skip this section.

In order to save on pins, the RAM designers multiplex the address lines into a DRAM chip. This means that the system designer must provide logic to do this multiplexing. This is what the 74AS253 in the Doctor Dobbs Journal (DDJ) upgrade article is for, it multiplexes the two new address lines (A17 and A18, as well as make a video address) for the extra pin used on the 26K DRAM chips.

This multiplexing, as well as the other timing of the DRAM chips, is controlled by two lines, row address (RAS, actually not RAS) and column address (CAS, not CAS), so named because the internal architecture of a chip is square, arranged in rows and columns. During a memory access cycle RAS goes low first. This transition clocks in the row addresses from the address lines. Next, after a minimum setup time CAS, also goes low clocking, in the other half of the address. The CAS line in essence performs the chip select function. Outputs will remain tri-stated unless a CAS transition is seen. After a further time period the data lines become valid and remain so until CAS goes high again. The R/W (read/not write) line determines whether the cycle is a read or write cycle. If it is high during the downwards transition of CAS, then a read is done and the data appears on the output line. If it is low, then data is latched in from the data input line. Different timings of these three lines (RAS, CAS and R/W) allow one to do a number of things other than a normal read/write cycle. One can do a fast refresh cycle without accessing memory, a read/modify/write cycle, or page mode read/write (all on the same row). An example of a read cycle is given in Figure 1.

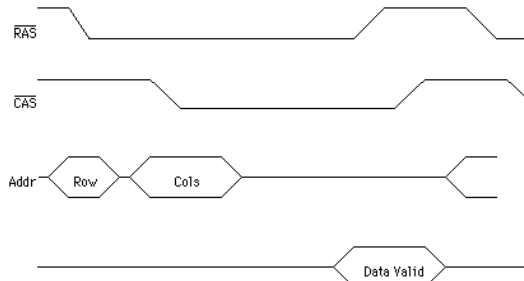


Figure 1. A typical DRAM cycle.

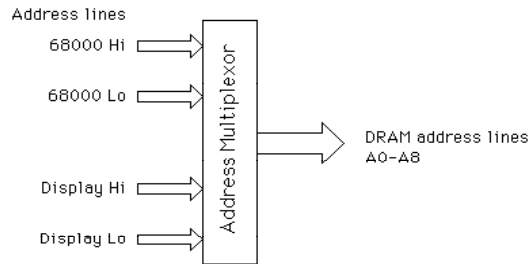


Figure 2: Addressing: block diagram

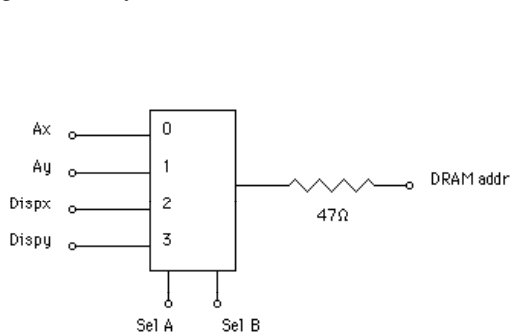
Because of the speed required to get all this data to the chips in time (esp. the RAS, CAS, R/W and address lines) it is normal to drive these lines through resistors (47Ω) in order to reduce signal reflections.

Dynamic memory needs to be refreshed. This means that regularly (every 2ms or so) each row has to be accessed in some way. Special fast refresh cycles are provided for this (RAS without CAS) which are usually sneaked in between normal cycles.

Figure 5 shows the pinout of a 256 DRAM chip. Further information on this subject can be found in Reference [1] and other similar data books.

Mac Internals

Now let's look at how the Mac memory system works. The basic memory map is given in [3], page 40 and is worth studying. The addressing is sparse with many things being mapped more than once. The RAM is mapped into the bottom 4Mb (8 times). The ROM maps into the next 2Mb and various peripheral chips above 8Mb. For the moment the RAM is the interesting piece of hardware. It is accessed when the CAS line is activated during a processor or video access to memory. This occurs for any access into the bottom 4Mb (and other special circumstances, more later). In fact, there are two CAS lines, one for each byte. This way the 68000 can read/write single bytes. The CAS signals are generated by a PAL and delivered to the RAMs through 47Ω resistors.



Sel A	Sel B	Function	DDJ example
0	0	RAS - 68000 addr	A17
0	1	CAS - 68000 addr	A18
1	0	RAS - display addr	1
1	1	CAS - display addr	1

Note: Sel A = $\overline{\text{RAS}}$
Sel B = select 6800/display

Figure 3: Typical address multiplexer a la Doctor Dobbs (1/2 74S253). Figure 4: Logic table for SelA/SelB.

Mac RAM is accessed by the 68000 and by the video circuits. These circuits generate an address at the top of the first 512K. This is where the Mac software expects to put the screen buffer. In order to generate addresses for the RAMs, circuits are needed to multiplex four things: the 68000 high 9 bits of address, the 68000 low 9 bits, the video high 9 bits and the video low 9 bits. See Figure 2. The DDJ [2] upgrade involves adding a 4 to 1 multiplexer chip to generate the extra address bit used by the 256K chips. The Mac designers provide two signal lines that are used for controlling this multiplexing. Figures 3 and 4 show how the circuit from [2] works. The signal line called SELA is low during the setup time for the RAS transition, it changes high to setup for the CAS transition. In our case the more important line is SELB. It is low during 68000 accesses and high during video accesses.

Figure [6] shows the basic memory access paths used in the Mac. Figure [3] has a much more detailed block diagram of the Mac hardware. A useful exercise would be to identify the various chips on Fig. [3].

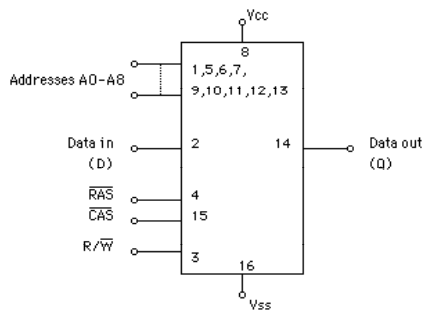


Figure 5. 256K DRAM chip pins.

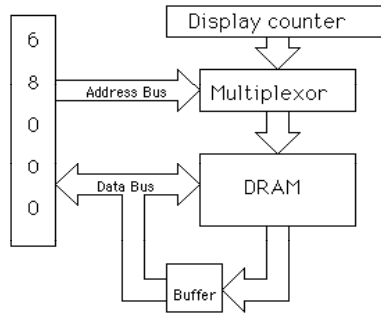


Figure 6. Mac Memory block diagram.

Adding more memory

In order to add more memory to our Mac, the main thing we must do is change the memory decoding so that different banks of RAM chips respond when different addresses in the bottom 4Mb are addressed. While doing this we must make sure that when video accesses are made (SELB is high), the correct bank is selected. Figure 7 shows such a circuit. The rightmost three chips are 1 of 8 decoders (74S138); they decode 1 out of 8 banks of 256K chips. The three nand gates (74S00) driven by the inverse of SELB force bank 0 to be selected during video accesses. The two (upper and lower byte) CAS lines are gated to the selected output pins. These pins are high when not selected. Because the nand gates ones complement the three high address lines, the selected lines come out of the pins the wrong way around. The R/W line is also ored with SELB and selected with another 1 of 8 decoder. (I fail to see why this is necessary, but my system does not run without it. It generates writes to memory during video accesses. This is probably a timing problem and/or may be due to the long signal lines I'm using. Figure 8 shows the chip pinouts and where the various signal lines should be connected.

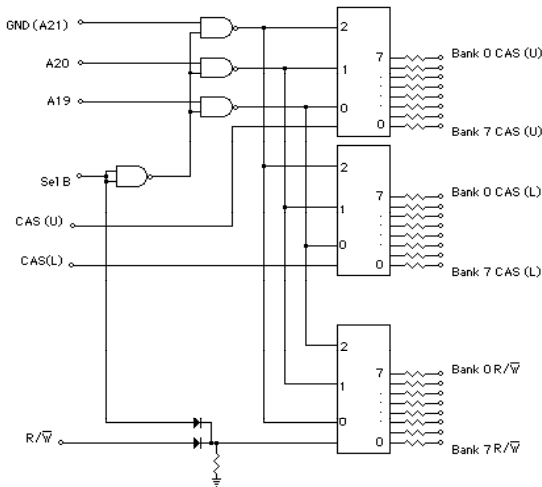


Figure 7:
Full circuit diagram

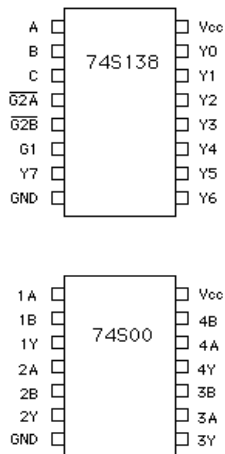


Figure 8: Chip pinouts and connectons

Chip	Chip Pin
A19	C
A20	B
GND (A21)	A
GND	$\overline{G2A}$
GND	$\overline{G2B}$
CAS - R/ \overline{W}	G1
BANK 0	V7
BANK 1	V6
BANK 7	V0

The important lines CAS(U), CAS(L) and R/W must be obtained from main logic board. The CAS lines are obtained by cutting the traces on pins 12 and 13 from the chip labeled TSM (Timing State Machine) at D1. Follow these traces to see where they go. They end up at the low end of the SIP resistor pack between G4 and G5. The bank 0 outputs should be connected to the other side of this SIP where the lines go to the RAMs CAS pins. The R/W line is extracted by clipping the resistor R33 between D3/E3 and D4/E4. Connect the D3/E3 end to the input of the multiplexer and the bank 0 output to the other end.

At boot time the Mac comes up with the ROM mapped in at 0 and the RAM mapped at 6Mb. If you wish to have more than 2Mb of RAM, or map it in above 2Mb, you must add extra circuitry to make RAM accesses. When A22 is high, force a selection of bank 0 (just one more chip). As a simple alternative, one can connect the line to A21 to GND instead. This maps in the bottom 2 Mb twice, once at 0 and once at 6Mb. This also means that accesses to RAM at 6Mb will access bank 0 during booting. There is a pin on the VIA chip // port that controls the memory mapping. This too is left as an exercise for the reader.

The RAM chips MUST be 150ns chips. The new CAS decoding adds delay to the memory cycles and the extra speed is required. Since most people do not have access to 4 layer boards with Vcc and GND ground planes, Vcc and GND wiring is VERY important. It should be short, direct and of low resistance. Because of the heavy precharge currents on 256K RAMs each chip should have at least a 0.33 μ F disk ceramic capacitor across it, as close to the supply leads as possible. If possible use a larger one. Be careful, as the power supplies of the RAM chips are the opposite direction from normal TTL (+5 to pin 8 and GND to pin 16). This is an easy way to fry chips!

The address and data lines to the RAM should be buffered if more than two banks are to be used. These should be FAST for the address lines and RAS (74S). The output lines from the RAMs are already buffered on the board, although if you have amazingly large amounts of memory this could become a problem. The input data lines should be buffered in the same way as the address lines.

Physical considerations

There are two main physical considerations to keep in mind. The first has to do with the Mac environment. The Mac gets hot enough as it is, especially above the logic board, and adding more heat is just asking for things to get too hot. I recommend one of two actions: either run it with the back off (as I do at the moment) or buy a silent fan. I am told the Piezos are good. A fan could be tapped off the logic board. The Mac power supply is designed to provide enough power for the Mac and both an external floppy and a modem. You may run into trouble adding extras to these.

The other consideration is how to put all these components together. I simply bought a hacked up board and connected it to my Mac via a spaghetti mess. I have three ribbon cables: one for data in, one for data out and one that is connected over the top the pins on a RAM chip on the main board. This setup provides most of the signals I need. The other ones are the CAS and R/W lines mentioned above, the address lines A19-A21 (from the 68000 pins 47, 48 and 50 resp). You

should try to keep signal lines down below 45cm in length, especially the CAS lines. This board should fit inside the case (however, mine is lying on the table beside the Mac at the moment). I intend to rebuild it later.

Another problem is how to get all the signal lines off the main logic board (bus). I see 3 basic solutions:

- 1) Do as I have: build a daughter board and hook it up via ribbon cable.
- 2) Piggyback the chips on top of the first bank. This may work for 1 extra bank. The CAS lines for each row and all R/W lines of the piggybacked chips would have to be wired separately. There could be heat problems and decoupling caps would probably have to be mounted on top. The 4 selection chips would be on a small daughter board.
- 3) Build a board that plugs into the ROM sockets. This gives you access to the data lines (32 of the 50-odd lines you need to move). You would need data buffers. If you get most of the address lines this way then you have to do your own address multiplexing, although this means the processor runs faster. Also the video could not be placed on memory accessed here as the video address and data bus are not connected to the ROM socket.

Debugging

A 'scope is essential. Mine is 10000 kilometers away so I had to do without; I could have saved hours of work otherwise. The usual sign of an error is a bizarre buzz when the system comes up. This indicates that the sound DMA done by the video is wrong and is getting garbage from somewhere. I.e., it means the CAS signals are wrong. I wrote a set of simple memory test programs that step through memory reading and writing patterns. Timing errors can show up as snow on the screen, nasty noises, the floppy motor going crazy (also done by video DMA) or data being written on the screen as the testing program goes through the memory being tested. You should also run the normal memory test diagnostics on your existing 512K.

Software

The current Mac ROM only understands Macs with 128 and 512K. I hope this will be fixed with the new ROMs due to come out soon. The system provides no way to free any memory above the screen image without patching the ROMs. The best solution is to use a RAMdisk. Using FEDIT, I have patched the publicly available RAMdisk to work with my system, in two different versions.

The Future

I intend to put my system onto a PC board. I plan to build one with room for 2Mb. I also want some form of simple write protection that allows me to protect banks of memory so that they cannot be written without my enabling them first. This way I can build a RAMdisk that can't be trashed by runaway programs! This should take another 3 chips.

Conclusion

Hopefully this is enough to get you started. If you want the RAMdisk programs and/or my simple test programs please send me mail at my network address above or send a diskette and a SASE.

Disclaimer/Warning

You are on your own. I take no responsibility if you damage your Mac. This document and the attached illustrations are copyrighted by myself and Tanewha Systems. Permission is given for reproduction and distribution, provided these same conditions apply for the noncommercial use of these circuits. If you want to make money off them please contact me (its a VERY small charge ...).

References

- [1] Motorola Memory Data Manual. *Motorola Inc* 1982.
- [2] "Fatten Your Mac", Doctor Dobb's Journal, January 1985.
Tom Lafleur and Susan Raab.
- [3] "The Apple Macintosh Computer", Byte, February 1984.
Gregg Williams.

A Homebrew Uninterruptible Power Supply For The Macintosh



by Jay Z. James

Say you're computing up a storm on your Mac, putting the finishing touches on your Macdraw masterpiece. And the lights go out. Bye-bye, masterpiece. It could ruin your whole day.

Mainframe computer installations protect against such data loss with "uninterruptible power supplies" (UPS) that switch over to a bank of storage batteries whenever the utility power is lost or degraded. You could buy a UPS for your Macintosh, too, if you don't mind paying \$330 and up. Or you can build the homebrew UPS described here for about \$135.

Theory of Operation

First let's define some terms. A power inverter is an electronic assembly that changes a direct current (DC) power input signal, as might be supplied by a backup battery, into an alternating current (AC) power output for the Macintosh power cord. A square wave power inverter produces an AC output that jumps abruptly between a constant positive value and an equal negative value, instead of sweeping gradually between positive and negative as a sine wave inverter would do.

The secret of cheap uninterruptible power to the Macintosh is giving up signal conditioning. The Macintosh's switching power supply doesn't require a sine wave AC input, so a simpler and cheaper square wave power inverter will do. What's more, the frequency of the AC input is unimportant, since the Macintosh's internal AC requirements (for the screen and the disk drive motor) are internally generated. So there's no need for expensive UPS frequency control.

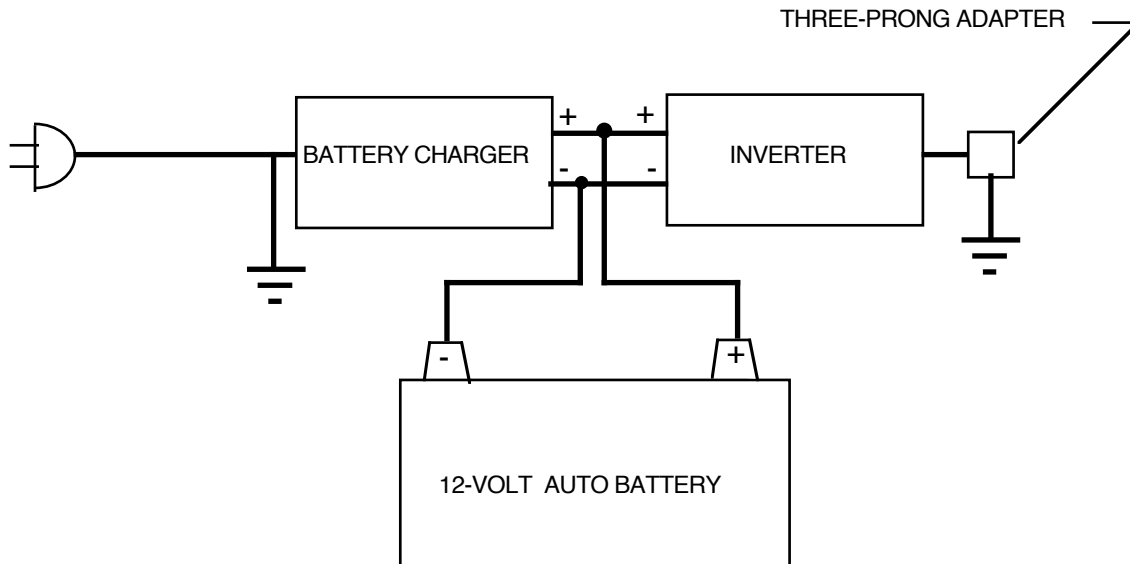


Figure 1: Uninterruptible Power Supply Block Diagram

Tripp Manufacturing Company of Chicago makes an inexpensive non-frequency-controlled square wave power inverter -- the Tripp Lite PV-100. The PV-100 takes up to 9 amperes (amps) of 12-volt DC current and puts out .85 amps of 117-volt square-wave AC current, for a total power output of 100 watts. The Macintosh is rated at 60 watts, but requires only about 40 watts during normal operation. The PV-100's one shortcoming is in the area of cleanliness of power. The PV-100's output is not a sharp-edged square wave; there is a voltage overshoot when the signal switches polarity that can be significant at no-load (up to 180 V peak). But the overshoot drops rapidly as load is applied (135 V at 25 watts, and 120 V at 60 watts).

There are two types of UPS duty ratings -- continuous and standby. A continuous-duty UPS normally costs more because it requires heavier-duty components, but it works better for two reasons. First, there is no switching transient between the utility power and the standby power, since the computer works off the UPS all the time. Second, the "electrical mass" of the UPS transformers serve to damp switching surges and electromagnetic interference during non-emergency operation. The UPS described here is a continuous-duty design; the Macintosh requires so little power that the larger-size components cost less than the power switching that would otherwise be required.

Finding the Components

As shown in Figure 1, a continuous-duty UPS has only three major components -- a storage battery, a power inverter, and a battery charger. For the battery, buy the cheapest good-quality model you can find. In the Berkeley area, Go Battery Company at 1441 Rumrill Boulevard, San Pablo gives good value. Their standard-size battery, rated at 55 amp-hours (that's more than **sixteen hours** of normal Macintosh operation) costs \$34.95. Zack Electronics in San Francisco sells the Tripp Lite PV-100 for \$67.00. Finally, for continuous duty you need a battery charger that can match the average current drain of the Macintosh. Kragen Auto Parts has a 4-amp model for \$28.99, though battery chargers in the 3- to 5-amp range are often advertised on sale at auto parts stores. Remember that you don't need to have a fully-charged battery at all times, and that any shortfall in the charging rate will be made up when the Mac is turned off. Make sure that the charger has a three-wire power cord.

You will need some minor components too: a pair of battery terminals (\$.99 each at any auto-parts store), a three-prong receptacle adapter for the Macintosh cord (\$.49), and some hookup wire and electrical tape (scrounge). Total cost = \$133.41 for the parts shown in Figure 2.

For tools you'll need wrenches to tighten the wires onto the battery terminals, pliers, a screwdriver for screwing the charger and inverter cases together, a soldering iron, and an optional AC/DC volt-ohmmeter.

Building the UPS

Construction consists of nine steps, eight of which are shown in Figures 3 through 10:

1 (Figure 3). Cut off the cigar-lighter plug on the inverter's input, pull the two conductors apart, and strip the ends back about an inch. Wrap both conductors with masking tape or Scotch Magic tape, mark the tape around the ribbed conductor with a "+", and mark the tape around the smooth conductor with a "-".

2 (Figure 4). Cut off the clips on the battery charger's output wires, and strip the ends back about an inch. Wrap the wires with a short piece of tape as in Step 1, mark the wire that was connected to the red clip with a "+", and mark the wire that was connected to the black clip with a "-".

3 (Figure 5). Connect both the inverter's negative input wire and the charger's negative output wire to one of the battery terminals, one under each of the two crimping bolts. Connect the positive wires to the other battery terminal in the same way.

4 (Figure 6). Wrap the two positive wires together with electrical tape, and do the same for the two negative wires (for the sake of neatness).

5 (Figure 7). Attach the inverter to the charger so that they make one unit. I found that the inverter could be turned upside down and screwed onto the top of the charger box through holes in the inverter's steel-mesh top plate, using two existing charger case screws.

6 (Figure 8). Solder a foot-long piece of hookup wire (20-gauge minimum) to the grounding screw hole at the bottom of the three-prong adapter.

7 (Figure 9). Plug the three-prong adapter into the inverter receptacle, and attach the adapter's ground wire to the case of the charger. Test the continuity of ground from the charger plug's round prong to the three-prong adapter's round hole, using the volt-ohmmeter or a continuity tester.

Fall 1985 BMU G Newsletter

8 (Figure 10). Tighten the negative leads' battery terminal to the battery's negative post, and the positive leads' terminal to the battery's positive post. The inverter should start to buzz softly. At this point you might want to test the battery voltage and the inverter output voltage with a voltmeter, if one is available.

9. Plug the charger into a wall outlet. It should start to hum softly, and the pitch of the inverter's buzz should change. The output voltage should stay the same.

There you have it! Now just plug the Macintosh into the inverter adapter receptacle, turn on the computer, and stop worrying about blackouts. The UPS we built is now powering the Macintosh that runs the BMUG BBS!

AppleTalk for the rest of us

By Reese Jones

BMUGNET—the alternative to AppleTalk

This article describes a design for some AppleTalk-compatible local area network hardware that we call BMUGNET. The design maintains (I think) complete hardware and software compatibility with the AppleTalk hardware available from Apple. This design is an evolution and synthesis of similar designs; I have drawn on design concepts used by various consulting companies involved in installing custom AppleTalk-compatible local area networks for business and educational applications, including Lutzky-Baird Associates (Culver City, CA (213) 649-3570), INFOTEK Inc (Patchogue, NY (516) 289-9682) and William Tell Computing (Berkeley, CA, (415) 849-9993).

BMUGNET was designed to make use of the same modular components used extensively by the telephone company. The standard design established by AT&T and the fact that almost all telephones use the modular connector systems has lead to an economy of scale which makes the parts very inexpensive.

AppleTalk was designed to be easy to hookup and use, but was not designed for minimum cost, reliability or flexibility in installation configuration. AppleTalk also uses some very unusual connecting plugs that are custom-manufactured for Apple. Apple buys all the connectors that are produced, so the parts are not available on the open market. The unusual custom parts and overdesign has lead to a system that is more expensive than is really necessary.

Here is a comparison of some of the costs of AppleTalk components and BMUGNET

Component	AppleTalk	BMUGNET
Connector Kit	\$50.00	\$15.00
Connectors	\$3.60ea	\$0.16ea
Connecting Cable	\$1.50/ft	\$0.11/ft (multiply that by the network length!!)
Wall Boxes	n/a	\$1.40
Maximum length for Network	1000ft	5000ft

As you can see the costs for BMUGNET are substantially lower than for AppleTalk, while maintaining full compatibility. How can this be? It's primarily possible by using the modular connector components but also cutting some corners in design where we think AppleTalk is overdesigned like with the shielded cables, overbuilt shielded connectors, and single source manufacturing.

What we are looking for by publishing this design for general use is to promote the development of more AppleTalk-compatible software and hardware. AppleTalk is a well designed, flexible and powerful standard to work with. It is at a good cost/benefit point in terms of speed; the cost for much faster hardware increases sharply as you go up in speed only slightly. Assuming you setup your network correctly, AppleTalk is *Fast Enough* to work well for numerous networking applications in the real world. AppleTalk is also simple and much cheaper than anything else that comes close to being called a full networking design. We expect to see numerous products coming out for AppleTalk in the near future.

An Overview of AppleTalk

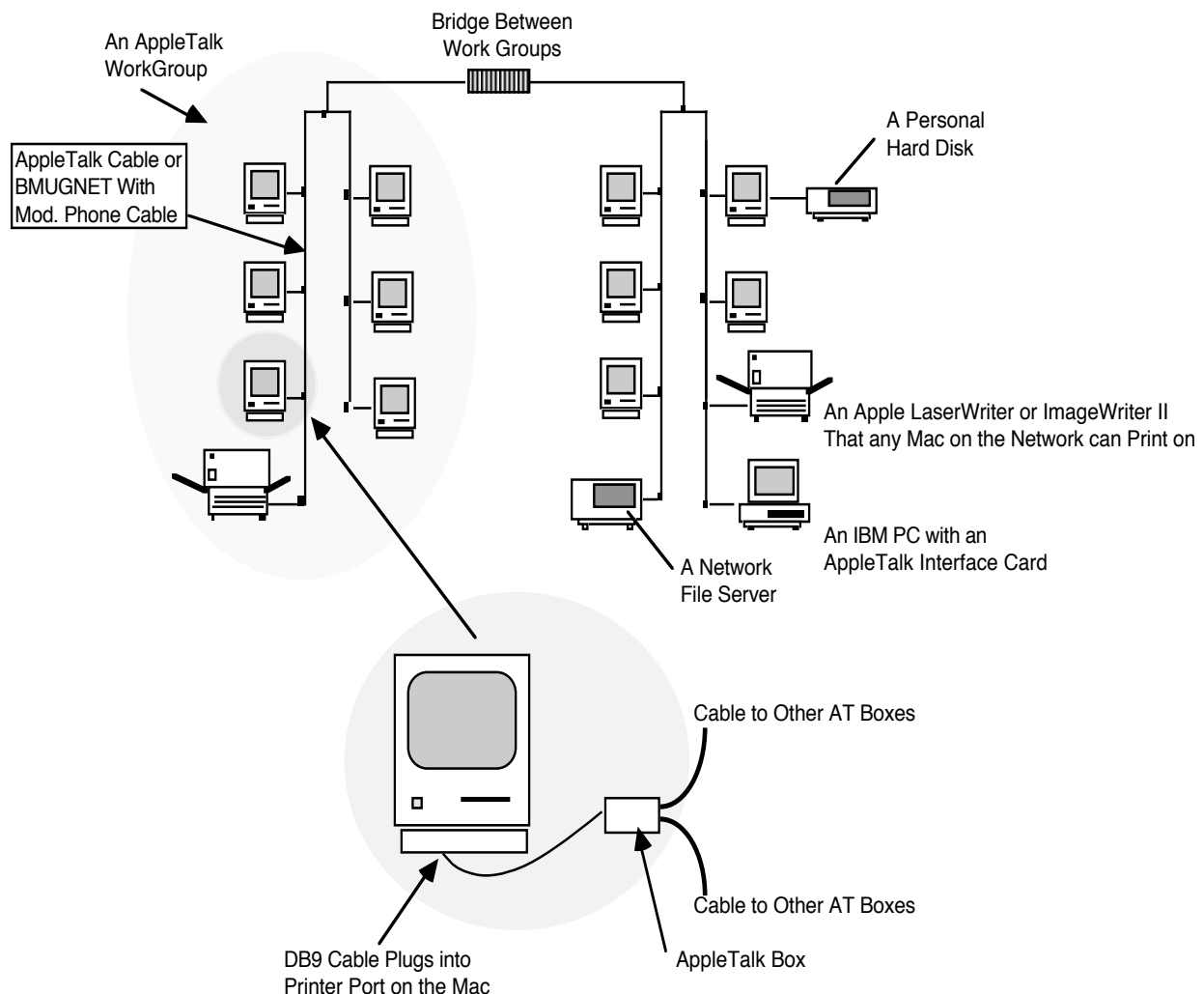
AppleTalk is a combination of software (communication protocols and driver standards) and some special cables to connect several Macintoshes together such that they can exchange data with each other.

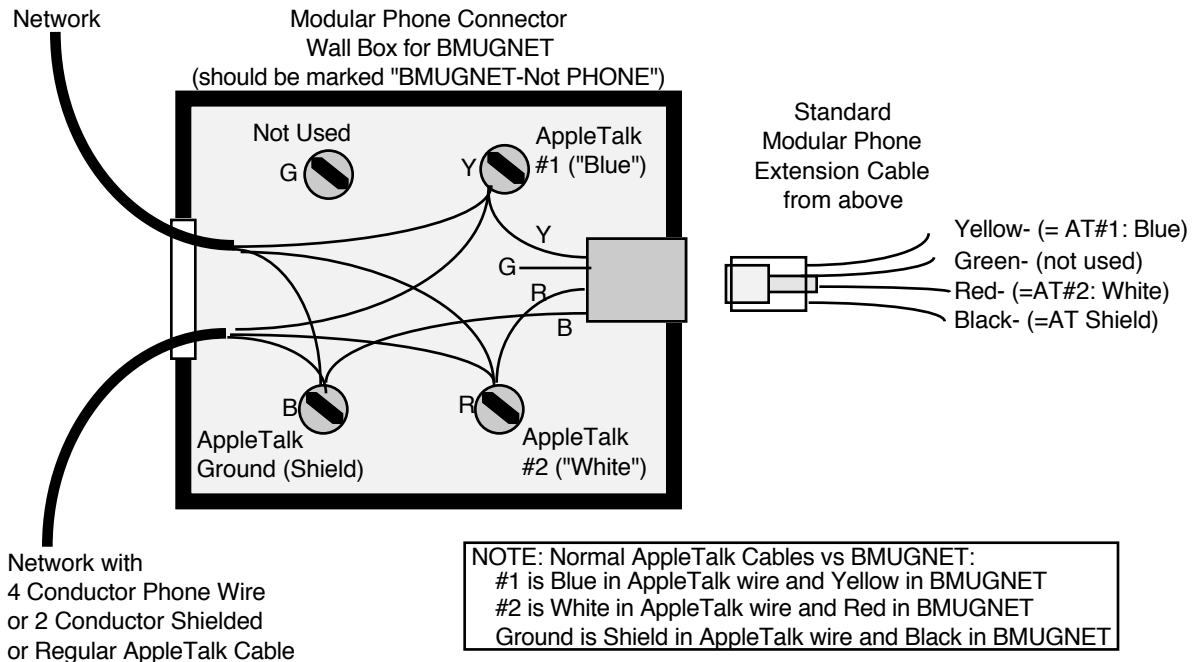
Why connect Macintoshes together? This allows you to easily transfer data (programs, files, mail etc) from one machine to the next conveniently. These data transfers can also be made between different types of computers with the aid of a standard file system for the network, so a Mac and IBM PC can easily share data given the right software for the network. A network allows several Macs to share expensive or infrequently used peripherals like modems, printers and mass storage devices. Sharing one Apple LaserWriter between several Macintoshes is the most common use for AppleTalk right now.

With a *disk server* (capable of working with only low level disk driver calls like read or write a given sector) for the network, several Macs can have access to a large hard disk storage device. With the addition of a network *file server* (more intelligent than a disk server, usually with its own CPU and RAM, capable of running programs locally, and working with a much higher level of file and system calls between the program and the disk) several Macintoshes can use the same program and even work on the same data file. This has great value for things like multiuser databases, accounting, laboratory, or educational applications.

If you only want to transfer data between 2 machines you could simply use your Radio Shack cable to plug the machines together and use MockTerminal or FreeTerm to transfer files from one machine to the next through the cable. If you wanted to connect more than two Macs together in this way you could daisy chain them with more cables and Y adaptors. You could even transfer data between any two of the Macs assuming that only one Mac was sending and the receiver was ready to get the data. If an additional Mac wanted to send some data at the same time, it could, but then two computers would be sending data over the same wire at the same time. You can see that unless something special is done, the two messages would overlap one another and become confused. The AppleTalk software protocol keeps the messages separate and makes sure they are delivered to the intended receiver. The AppleTalk hardware uses a transformer to isolate electrically each Mac from the wires running between the various Macs on the network while still allowing it send and receive data over the wires. The rest is all software (which we will be discussing in the Spring '86 BMUG newsletter).

A Local Area Network Layout Using AppleTalk (or BMUGNET)





Instructions for Rev B board

Assembly of the Rev B design is very similar to the Rev A design, with several simplifications. The Rev B design will be much easier to assemble because it uses only PC-mount and modular components. Thus, you don't have to solder any loose wires onto the printed circuit board. With the Rev B design, the board is narrower and uses a PC mount DB9 connector that is positioned such that it can plug directly into the female DB9 on the printer port. A Radio Shack Joystick Extension cable (see "Hardware Notes"/cables) can be used if you want to add a flexible wire between the BMUGNET board and the Mac. If you are using a Lisa/MacXL, the RS Cable can be modified as described in "Hardware Notes" to go between the DB25 port on the MacXL and the DB9 port on the BMUGNET. The DB9 on the BMUGNET board is pin compatible with the standard AppleTalk Connector Box, so the BMUGNET board can be used with the LaserWriter or any other AppleTalk device that uses the DB9 connector.

Soldering the other parts onto the board is very simple, even if you have never done any soldering before. We recommend starting with the three resistors. Gently bend the wires to fit into the appropriate holes and push them through. Once the wires are through, bend them out to hold the resistor close to the board, turn the board on its back and heat a small drop of solder onto the holes where the wire goes through. Next, add the capacitor (the pancake-shaped part), then the transformer, PC mount DB9 connector, and finally the PC mount modular phone connector. You're done!

Installing a Temporary (Mobile) BMUGNET Installation

For temporary installations, you can use a modular phone extension cord (wall-type, not handset-type) to go between two Macs or a Mac and a LaserWriter. As noted above, for only two devices a null modem DB9 cable will work alone. If you want to add more Macs in a temporary installation, you can use telephone "Y" connectors to plug two modular connectors into one BMUGNET board. (Y adaptors are used to connect two phones to the same line.) You can add more devices by daisy-chaining them together as shown above. You do need one BMUGNET kit for each Mac or other device that you're connecting to the network. BMUGNET is designed to use **exactly** the same cables used for a phone. You can get the pre-made cables and modular Y adaptors at most hardware stores.

Permanent BMUGNET Installations

If you are doing a *permanent* installation, you will want use the modular phone wall boxes described above. The boxes used are identical to those used for your phone; you can get them anywhere phone parts are sold. Install the BMUGNET wall boxes just as you would install extension phones, by running the wire along or through the walls from

one box to the next. Be sure to attach the same color wires to the screws in each box. We recommend cutting off the green in the box to protect the system in case someone accidentally plugs a phone line into one of the boxes. After you install the BMUGNET wall boxes it is a good idea to write clearly on the box "BMUGNET ONLY - NO PHONES".

Adding Terminating Resistors for Larger Installations

You might need to add a terminating resistor to the first and last Wall Box of a large BMUGNET installation. This is easily done by connecting a 100 ohm resistor between the Yellow and Red screws (AT#1 and AT#2) and a 1M ohm resistor between the Blue and Black screws (AT#1 and Ground) in the last wallbox on each end of the network. Twist the wires from the resistor around the screw along with the other wires. This is not always necessary, but should reduce noise on the network. It is a more conservative design that loads the terminal ends and reduces reflections which cause noise.

Wiring the Wall Boxes Together

You can use regular 4 conductor telephone wire to connect the BMUGNET boxes together. If you are obsessive/compulsive, you can spend a little more and use shielded two-conductor cable (similar to standard AppleTalk cable) between the wallboxes. If you are installing a lot of boxes, we recommend getting a Modular Crimping Tool (Radio Shack 279-388A, \$7.95), a roll of Telephone Cable (Radio Shack, Modular Type 4 Conductor, 278-366, \$10.98 for 100 feet), and a pack of Modular Connectors (Radio Shack 279-384, \$2.29 for a 10-Pack). If you don't use these for BMUGNET, you can always use them for your phones.

Interfacing with Regular AppleTalk Hardware

You can easily interface to regular AppleTalk by making an adaptor cable with an AppleTalk plug on one end and a modular connector on the other end. This is easiest if you get one short AppleTalk extension cable, such as the one that comes with an AppleTalk box. Cut it in half and expose the white and blue wires inside by pulling back the shielding. The Shield (Ground) needs to be connected to the black wire in BMUGNET, the blue wire (AT#1) should be connected to the yellow wire in BMUGNET, and the white wire (AT#2) connects to the red wire in BMUGNET.

We don't recommend having an exposed AppleTalk plug anywhere on a network. Plugs not attached to anything add noise and cause problems (with either regular AppleTalk, or with BMUGNET). Since it is difficult to attach the round AppleTalk cable to the modular connector, we suggest attaching a short length of modular telephone cable to a connector and soldering the appropriate color wires in the telephone cable to the AppleTalk cable. Apple has said that they will soon be selling the standard AppleTalk plugs singly. If you can get one, attach the standard AppleTalk plug directly to the telephone cable with the modular connector on the other end. You can then put the modular plug into any BMUGNET wallbox and plug the other end of the adaptor cable into a regular AppleTalk box.

The Contents of a BMUGNET Kit (\$15 each through BMUG)

1	Custom printed Circuit Board
1	Custom Transformer
3	1K Resistors
1	0.1 μ fd Capacitor
1	PC Mount DB9
1	PC Mount Modular Phone Connector

Other things you might need:

- Soldering Iron, Solder, Wire Cutters/Strippers, Screwdriver
- A Radio Shack Joystick Extension Cable
- Pre-made Modular Phone Extension Cables
- Modular Phone Wall Connector Box
- Modular Phone Cable and/or Shielded 2 Conductor Cable
- Modular Crimping Tool
- Modular Connectors
- BMUG AppleTalk-Related Disks (#15 and #25)

Ordering a Kit from BMUG

When you order a BMUGNET kit from BMUG it will include the parts listed above, instructions for putting it together, and other relevant information. The Rev B kits will be available in quantity by mid-November and the Rev A kits are available immediately but are in limited supply. Please remember that we are providing this information and these kits for fun. Thus, **none of our information is guaranteed to be correct** and the kits are not guaranteed to work. You are using BMUGNET at your own risk. We have tested BMUGNET and it has worked reliably in several installations for some time now. This is an experimental project for BMUG, so the BMUGNET kit prices, designs, and specifications are subject to (and will likely) change at any time without notice.

We are distributing both the *BMUGNET* and the *MacRecorder* kits at cost in the hope of stimulating more software and hardware products (public domain or commercial) that will take advantage of them. The BMUGNET configuration can also save you hundreds of dollars compared to the prices for regular AppleTalk connectors and cables when wiring up an office.

The BMUGNET kits are **very** easy to put together as well as being fun and emotionally satisfying. So if you have never touched a soldering iron, give it a try. For those who don't want to deal with assembly, a couple of members have offered to assemble some kits for a small charge; if you would like to get pre-assembled kits add \$5 each when ordering. For those interested in a large number of kits or who need help with assembly or installation, simply call or write BMUG.

Deeper Inside Apple Talk

By Hal Adkins, Infotek Inc.

INFOTEK INC, 56 CAMILLE, E. PATCHOGUE, N.Y. 11772 (516)-289-9682

Appletalk is a good, low cost computer data networking system, but like most computer system elements, you should know more than just how to plug it in. The design and literature implying any idiot can instantly be running 32 Macs, laserwriters, file servers and mainframes on the networks probably accounts for many of the problems and rumored problems with Appletalk. We will try in the following notes to give some background and recommendations based on years of communications, design, development, experience and a few months use of Appletalk.

The transmission of digital information is usually by hard wire or over voice networks via modems. We are specifically covering hard wired and twisted pair differential communications (RS422) as used in Appletalk. Most people have the idea that coaxial or shielded, single ended transmission is the best, but years of experience shows that balanced differential twisted pair can have much lower error rate per megabit mile than coaxial or shielded pair if it's done right. Millions of telephones have been running voice frequencies or 256,000 bits/second voice data for years with unshielded twisted pair and I wonder why people disregard these facts when designing. Apple states that unshielded cable can be an unreliable transport medium but Dartmouth college, Corvus, DNA, Orchid and the telephone industry seems to say different. Corvus often runs at 1 megabit per second rather than the Appletalk under .25 megabit/second rate. Flexible Coaxial cable depends on a braided shield that is usually called 85% shielded. Coaxial cable is excellent for 10 megabits/second baseband or RF carrier uses but working baseband with a signal relative to ground brings up the question of what is ground. A good broadband ground is perhaps the most illusive thing in field engineering. The laboratory worker avoids the question by testing on a thick copper bench with a central ground point. A lab design is then optimized with a near perfect ground and the engineer then wonders why field results are so bad. Is the green wire on the home or office wall receptacle really the good, low noise ground that designers rely on? I've had much better results over the years assuming a high impedance, noisy ground was all that's available in "real life".

Looking quickly at VSWR (voltage standing wave ratio) or Viswar as it's usually pronounced, I find people apply 100 megahertz or one gigahertz thinking to 230 KHz and end up compromising in the wrong direction. Transmission line impedance or characteristic impedance is another factor to be considered. Attenuation or loss can be excessive if 500 megahertz power is sent on a mismatched transmission line but at 250 KHz it's not that serious and it pays to experiment. Unbalance on twisted pair transmission can be crucial if multiple repeaters are used but since no repeaters are used, this is not critical if the right transformer is used.

Lets look at Appletalk specs say and what experiences has shown;

Appletalk- a bus. network. (Theory)

- (1) Retail per connection \$50.00
- (2) Service - Laser printer, file server, bridge to other Appletalk networks, ethernet gateway, disk servers, IBM PC interface, 3270 server, serial device (modem or printer) device server, unix server, electronic mail, etc.
- (3) 32 devices per network.
- (4) Installation - easy.
- (5) Baseband with no tuning.
- (6) Network software - built into applications, architecture - Layered open.
- (7) Max. length 1000 ft.
- (8) Transmission speed - 230 Khz.
- (9) Access method CSMA/CA.
- (10) Cable type. Twisted pair, precut cables in 2 meter and 10 meter lengths or 100 meters of bulk cable.

Experience; (2) Apple has a connection to the Laserwriter and third parties are just coming out with a few of the remaining items with all software needed. (7) Max length seems to be under 400 ft. for any reliability and it's difficult to most applications to get 32 devices into 400 ft. (10) Cable type does not mention shield twisted pair which is all any of us has seen from Apple. We have not seen Apple furnished or specified fire code approved Teflon cable.

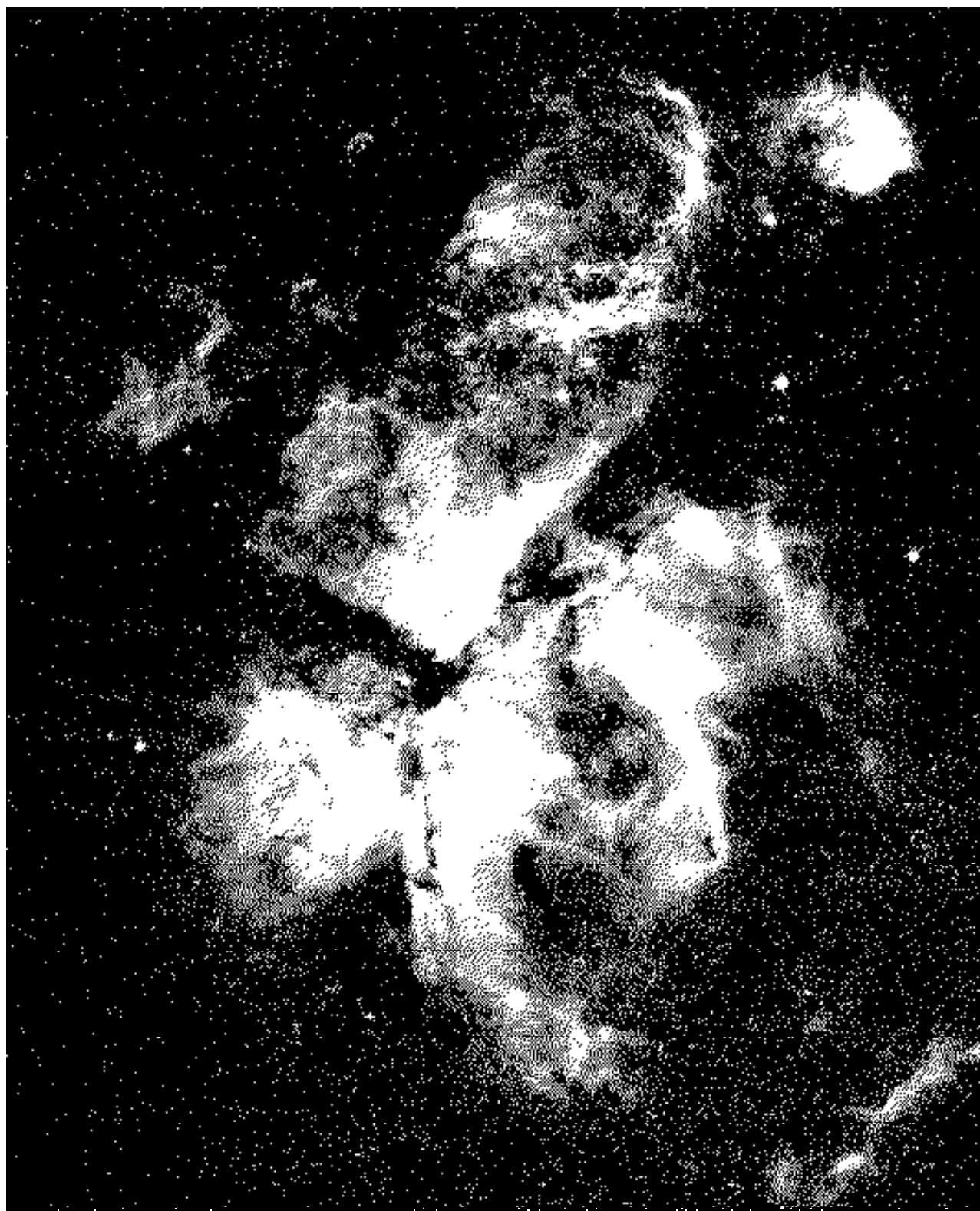
To find the reasons for the 400 ft. and to find how to reliably extend this to our presently available over 5000 ft. we measured the internal Mac circuit, came up with an equivalent circuit then an equivalent circuit for the system . Optimized the system using this data, we built up several long distance (to 6000 ft.) and multi Mac networks and tested them while injecting typical electrical noise as found in an office. We also installed some networks in a business invironment.

The real world and theory agreed within less than 10% for the ungrounded Infotek systems but varied by 300% and more for apple connectors. It appears that beauty won out over function and the nice chrome plate ground connections on the Apple version develops noise far better then any of our labs noise generators, and leaving out the ground through the 3rd pin makes it worse. The crazy glue used to hold these Apple production connectors together makes this poor shell connection far worse on some boxes thus some networks work and others dont. Our reliability experience over the years with telephone modular connectors has been excellent, so we usually design with these field proven connectors to avoid being the pioneers with the arrows in our back.

To run a network without grounds or at least without shielding, you make isolation transformers with less than 10 picofarads of primary to secondary capacitance rather than have about 36 times this value and depend on a good ground to solve the problem.

At some future date we can answer questions like (a) what length of 100 ohm cable has as impedance of 100 ohms, (b) What makes up characteristic impedance, (c) How to layout a good Appletalk network, (d) What is the effect of a nonsymmetrical driver on digital communications lines, (e) Termination and (f) What is noise to a digital filter.

The surprising result of the new and different hardwares is an easier to install, lower in cost, 5000 ft. + line reliable network that allows the full potential of a really good low cost networking idea (Appletalk) to be used. **GET ON LINE.**



Eta Carinae

A Review of MacModula-2

by Michael La Belle

Price: \$150.00

Available from: Modula Corporation
950 N. University Ave.
Provo, Utah 84604

Background

Modula-2 was developed by Niclus Wirth, the creator of Pascal. In fact, the Modula-2 language contains most of the features found in Pascal. But unlike Pascal, Modula-2 was designed to be a "real" programming language and to allow a programmer or team of programmers to implement large complex programs (such as operating systems). In addition, it also allows you to perform low-level operations, which in Pascal would require the use of assembly language.

As the name implies, a program written in Modula-2 is composed of modules. There are two types of modules: client modules and library modules. A client module consists of a group of (usually) related procedures. A major feature is the use of the IMPORT list to import "facilities" (commonly used procedures such as those for input/output to disk or screen, various math functions, string handling, etc.) into the client module from various library modules.

A library module exports facilities to client modules and consists of both an implementation module and a definition module. The implementation module contains statements which perform the modules actual work and looks much like a client module, while the definition module contains a list of what items from the implementation module the library module can export and what items it can import from other library modules. From an executing programs point of view, a library module is largely a black box with only a few visible features, the rest being hidden from the program. This allows you to change the way a library module does its work without having to rewrite whatever client modules depend on that library module. As you might guess, it is fairly easy to turn a client module into a library module. All modules can be separately compiled and a module can contain other modules.

Modula-2 has a number of other differences from Pascal, most of which are changes in syntax, however, there are also a number of new statements, data types, and operators. Due to space and time limitations I will not go into them. Anyone who wants further information about Modula-2 should read either "Programming in Modula-2" by Nicolas Wirth or "Modula-2 for Pascal Programmers" by Richard Gleaves. Both books are available in the ASUC bookstore on the Berkeley campus. I recommend the book by Gleaves. It isn't quite as complete as the book by Wirth but it is a much better teaching manual.

MacModula-2

MacModula-2 is a full implementation of the Modula-2 programming language which has been integrated into the Macintosh working environment. It is supplied on two disks, neither of which is bootable. Instructions for building bootable disks for either a 1-drive or 2-drive system are contained in the System Reference Manual which is supplied with the disks. The two disks, which are not copy protected, contain the Modula-2 compiler, linker, M-code interpreter with support files, various support files for the compiler, compiled library modules, two ToolBox folders containing files for interfacing with the Mac's ToolBox Utilities (one folder is for 128K Macs, the other is for 512K Macs), a screen editor, a Resource Maker application and a number of Modula-2 demonstration programs.

The System Reference Manual is 548 pages long and contains chapters dealing with an introduction to programming in Modula-2, how to use the text editor, compiler and linker, a description of each of the supplied library module facilities and how to call the Macintosh's ROM routines. The manual claims that MacModula-2 allows access to over 400 of the Mac's ToolBox Routines. While I didn't count them, a look through the section of the System Reference Manual dealing with calls to the ToolBox was convincing. The manual also contains suggestions on methods for using the Mac's memory efficiently, which is a must for those with 128K Macs. In addition, there are appendices dealing with the

Resource Maker, programming tips and error messages. While the manual states plainly that it was not written to teach you how to program in Modula-2, it is so well done that after reading it and examining the demonstration programs included on the disk, you will be able to write at least simple Modula-2 programs. In addition, the documentation of the ROM routines is so good that I tend to look in this manual before I go to Inside Macintosh.

To create a working program in Modula-2 you follow a four step process. Of course, the first step is to write the source code. For this you can use the Modula-2 text editor which, while not the equal of Macwrite, is certainly adequate for writing and editing a program. The Modula-2 editor uses a Menu bar with standard pull-down menus (File, Edit, etc.). If for some reason you find that the Modula-2 editor just won't do, then a program can be written using Macwrite as long as it is saved as text only and is written using only the Monaco font. One flaw in the editor is that the transfer option hasn't been implemented for the editor, so that when you have finished writing the program you must save it, quit the editor and then run the compiler rather than being able to transfer directly from the editor to the compiler.

The second step is to compile the source code to M-code. The Modula-2 compiler is a four pass compiler, that is it takes 4 passes through your program to produce the corresponding M-code. The first 3 passes consist of various (and extensive) types of error checking, syntax analysis, initializing local modules, etc. The fourth pass produces the actual M-code. If a mistake is found during any of the passes the compilation is terminated and the ErrorLister is called. The ErrorLister will present the program line which contained the error plus the preceding line. In addition it will also describe the type of error encountered and a caret (^) pointing to the place where it thinks the error occurred. If you are either just learning Modula-2 or how to program the Macintosh, you can expect to make extensive acquaintance with the ErrorLister. As with the editor, you cannot transfer directly from the ErrorLister to the editor to make corrections, but must go back to the desktop and then into the editor.

Once you have a compiled program, the next step is to link the program with all imported library modules. A nice touch is that now you can transfer directly from the compiler to the linker. The linker is also fairly fast, so if an error is found, not too much time is lost. After linking the program, you can then execute the program directly without having to return to the desktop.

The compiled and linked M-code programs can also be executed from the desktop by double clicking on the program icon, just as with any other Mac program. The programs are actually executed by the M-code interpreter, which must be present in a disk drive if the program is to execute.

Conclusions

I am of divided opinion on MacModula-2. Certainly MacModula-2 is a very powerful programming language, one which I judge would be very useful for writing large programs, particularly if various people write different sections of the program. In addition, the current version supports floating point operations and the new Finder and updated System appear to work well with MacModula-2. If you plan to write some really substantial programs then you should definitely consider MacModula-2. However, I would not recommend using MacModula-2 to learn to program on the Macintosh or even for writing short to medium programs because there are some problems. In my view one of the major problems with MacModula-2 is that the 4-pass compiler is very slow. Since virtually all programs of any substance contain some trivial (and perhaps some not so trivial) errors when originally created, a slow compiler combined with having to go through the desktop to get into the editor makes even relatively trivial errors deadly. Further, unlike Pascal, Modula-2 is case sensitive, that is, it differentiates between upper and lower case. By itself this is not necessarily bad, but Modula-2 requires that all reserve words (FOR, WHILE, IMPORT, etc.) must be typed in upper case. The upper case requirement was done to increase program readability, but I have found that 1) it makes understanding the program more difficult since the upper case reserve words interspersed with the lower case variables, etc. interrupts the flow of reading, 2) it increases the difficulty of typing in the program and 3) in the programs I have written, failure to capitalize all the reserved words has been a source of the deadly trivial errors mentioned above.

Another problem involves choosing from which of the supplied modules to import a given facility, since many of the facilities can be imported from more than one module. Some of the supplied modules are self-contained, but many import facilities from other modules which results in a module hierarchy. At the bottom of this hierarchy are the self-contained modules, while the top of the hierarchy consists of those modules which import the most facilities. If your program imports from a module at the top of the hierarchy, the amount of memory consumed would be quite large compared to the amount used by one of the low (or lower) level modules. Although the System Reference Manual does

thoroughly index both the ROM ToolBox Interface and Modula-2 library procedures and types, along with their respective exporting modules, no list of module hierarchy is included in the manual and this is definitely a flaw in an otherwise excellent work.

Since MacModula-2 is compiled to interpreted M-code, rather than to 68000 native code, the M-code interpreter must be present for a program to run. This means that your program will not run as fast as a program compiled to 68000 native code, although it will probably run faster than a program written in an interpreted language such as Microsoft BASIC. The table below shows how fast MacModula-2 runs some short programs compared to Microsoft's BASIC (version 2.0) and MacPascal. Also, if you intend to sell any program that you have written in MacModula-2, you are probably going to have to pay a licensing fee to Modula Corporation. I do not know how much that might be (the manual doesn't seem to include any references to a licensing fee), but with the interpreter required to run a program you can bet there is a fee.

Finally, even though you really can develop a program on a 128K Mac with only a single drive, I would strongly recommend a minimum of two disk drives and for serious programming a hard disk drive and a 512K Mac are a must.

Comparison of Execution Speeds *

PROCEDURE	TIME (SECONDS) TO RUN THE PROCEDURE IN:			
	MODULA-2	MS-BASIC (binary)	MS-BASIC (decimal)	MACPASCAL
Empty For Loop <	0.3	4.7	6.8	9.9
Mathtest I	10.6	69.6	108.4	156.2
Find Greatest Common Denominator	26.7	498.2	515.5	458.2
Graphics	30.0	35.1	35.7	33.8
Mathtest II	215.6	212.6	1745.0	1037.4

* Mathtest I (addition, subtraction, division, multiplication) and Mathtest II (sin, cos, atan, ln, e^x) were run for 10,000 iterations in a For loop. The empty For loop was also run for 10,000 iterations, as was the program for finding the greatest common denominator. The graphics program involved drawing a fairly complicated design, was run for 20 iterations and involved some addition and subtraction calculations, in addition to the graphics commands. These applications were run on a 512K Macintosh equipped with 2 drives.



Gato & Fokker Triplane

A Review

by John Heckendorn

If you enjoy graphic simulation games such as *Flight Simulator* and *MacChallenger*, and yet find yourself enjoying some of the faster action of *Airborne*, then you'll probably like *Gato*, a simulation of World War II submarine warfare by Spectrum Holobyte. *Gato* is fairly complex software, not only providing you with a preselected group of combat scenarios, but also enabling you to create your own with a variety of editing tools which allow you to affect the relative qualities of the ships and submarines (strength, speed, aggressiveness, number), navigation routes, mission descriptions, and level of play. You can create scenarios which are basically no-win situations, with adversaries stronger, faster, and more numerous than your paltry sub. On the other hand, if you're the type that likes to cheat at solitaire on long, ennui-filled nights, you can design what is essentially a shooting gallery. Furthermore, the game's fidelity to the Macintosh interface is complete: it is one of the few games which is entirely mouse-driven, though its creators have supplied keyboard command equivalents for most of the often-used menu items. It even comes with a Morse code tutorial (which you may need when you get to the highest levels, where radio messages no longer show up as text, but only as audible dots and dashes).

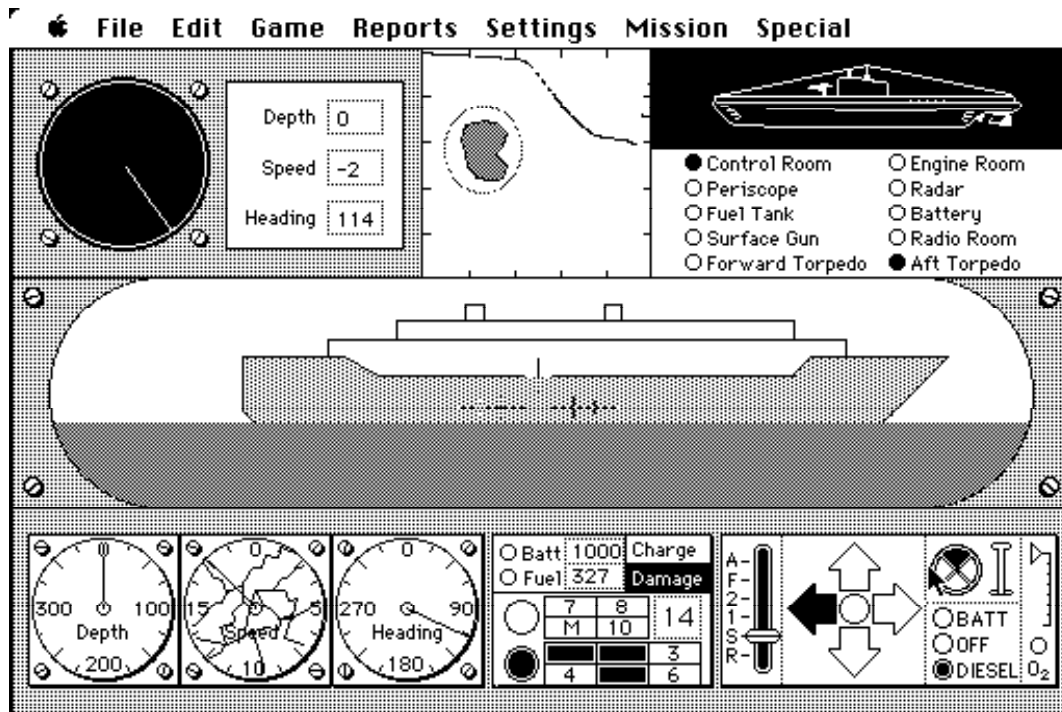
Gato's complexity means that it takes some time getting into the game: the documentation is very good, but there is a great deal to digest before you know how to avoid such common pitfalls as running out of air while underwater, running into ships or islands, or getting clobbered by depth charges. The normal screen shows a control panel with depth, speed, and heading indicators, periscope control, power source switch, torpedo/mine tube control, rudder and diving control, a radar screen, damage report window, and a speed control. In addition there is a periscope window and a navigation chart. All of these come into play and require coordination during a mission, so you need to take some time to learn what they do unless you like watching the water inundate your screen as you sink slowly into the depths. In the early going you'll get sunk about as often as you crashed the plane in *Flight Simulator* the first week you ran it.

The typical mission is either a search-and-destroy or a rescue of stranded airmen or sailors. Your objective on the former type are freighters, troop carriers, and tankers, with destroyers and P.T. boats running interference. Depending on the level of play you select, the attack boats can be nearly insurmountable foes or merely challenging. Not only will they attack with guns and depth charges, but they will ram your sub if they can. Damage to your sub is cumulative, and cannot be repaired except by your sub tender, which normally stays in safe waters because it is so vulnerable to attack. If you lose your tender through an enemy attack or your own blundering (it happens...), you're sunk. The action in the game can range from slow and patient to fast and furious if you're under attack on one of the higher levels. A time accelerator will speed up getting from Point A to Point B if there is no enemy contact along the way, however, so you can avoid long hours at sea and get to the juicy stuff without delay. With practice you will find that well-conceived attacks require planning and patience, since escort vessels such as P.T. boats and destroyers will try to zap you if they detect you, giving the real prizes (troop and cargo ships) the opportunity to disappear as you dive hoping you'll be able to surface again. In short, the game can be challenging and exciting, and has enough variables you can set that it stays interesting for a long time.

It does have a few bugs and shortcomings, however. It seems to have some problems running on 128k machines. I managed to coax out a few ID = 25 bombs (out of memory). Sometimes the screen will freeze leaving no alternative but to reboot. And curiously, the periscope window sometimes fails to refresh, which means that while your radar tells you a ship is moving off to port, its image does not move on the screen (in such circumstances clicking on the screen a couple of times rights matters). The graphics are fine if simple, though the animation of ships turning is anything but realistic. A couple of touches would have made the game better. It may be so only in the movies, but I expect to see a wake when a torpedo is fired. It would certainly explain why some of mine have missed broadside at near point-blank range. Also, when a ship is hit, a pseudo water-spout rises, the screen blanks, and the scene returns with the ship having disappeared. I would have preferred the overwhelmingly satisfying sight of the stricken vessel going down on its bow. It would be enough, even without terrified sailors jumping into the water. As it is, I feel just slightly cheated.

Fall 1985 BMU G Newsletter

The sound (which can be toggled on and off) is only adequate (burps and blips from the back of the Mac). Up to five games can be saved and reloaded.



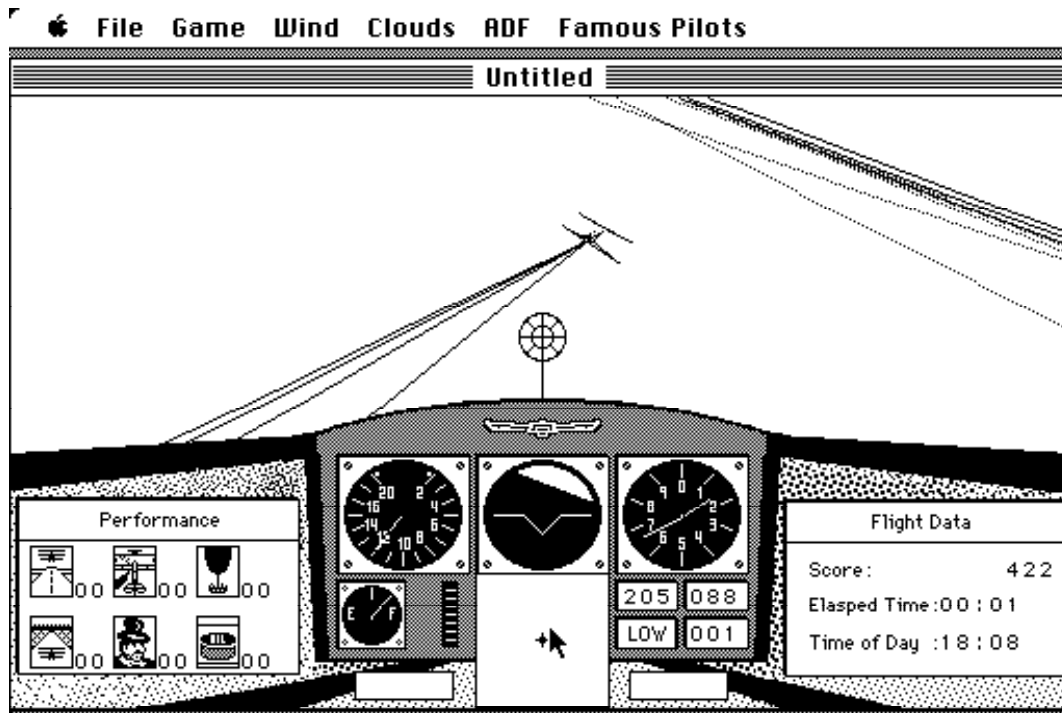
Gato

If *Gato* is a good game, then *Fokker Triplane*, distributed by PBI Software, just may be the game of the year. In terms of creating a realistic sense of flying, this game is better than even *Flight Simulator* in most ways save the complexity of the graphics, which were sacrificed somewhat in return for terrific animation. In a word, the game is *fast!* As the title implies, *Fokker Triplane* is a recreation of the World War I air war, in which you as an aviator are given the most maneuverable plane in the war, some good documentation teaching you how to fly it, and unlimited deaths as you learn. The cockpit is equipped with some non-period instruments to help you (such things as airspeed and fuel gauges, altimeter, artificial horizon, and directional indicators). You can draw from a variety of sample scenarios which will put you in the middle of a dogfight or close to an enemy target, or simply practice landing your plane. Even so, it will take you several hours before you can take off, perform turns and rolls, and land with consistent success. After that, the game is a matter of completing missions; unlike *Gato*, missions are not well-defined, but more a matter of dogfighting, pursuing objectives, and amassing points until you are shot down or you run out of fuel away from a friendly airstrip (although the action is more free-wheeling than *Gato's* deliberate pace, you still need to be aware of your fuel and ammunition supplies and plan accordingly).

The great achievement of *Fokker Triplane*, which cannot adequately be described with words, is the uncanny realism with which it captures the sensation of flying. While such functions as throttle and rudder are keyed, the mouse is the joystick, and every movement has an immediate effect on the plane's control. There is even a menu item for enhancing or diminishing the sensitivity of the stick (er, mouse). Coming out of a steep turning dive, you'll find yourself using body english to help regain control of your plane from gravity, momentum, and centrifugal force. Screen refresh is quite rapid, and a nearly three-dimensional illusion exists, though the graphics are quite simple.

Apart from these wonderful environmental effects, the game action is very entertaining and challenging. You will find yourself not only attacking stationary targets such as fuel dumps and reconnaissance balloons, but dogfighting with enemy planes which become more formidable as the game progresses, until they can match you move for move. It is worth getting shot down at least once (and don't worry, you will) just to see and hear your stricken plane spinning down to earth. The documentation includes detailed instructions on how to perform such skillful maneuvers as a barrel roll, knife edge flying, loops, and Immelmann turns. Knowing these and being able to execute them crisply will mean

the difference between winning a dogfight or experiencing the terrifying fall to earth. It is a game which perhaps cannot be won, but it is above all a *player's* game. Donald Hill, Jr., the author, notes in the documentation that he wrote it as a labor of love, and it shows.



Fokker Triplane

Though it's hard to fault this superb game, *Fokker Triplane* could be improved in future versions. One way would be making the bullets visible as they are fired. In the current game, you must have the enemy plane dead in your sights to shoot him down. A more realistic version would make possible leading your shots, and having him fly into them. Another enhancement would be to give the sky a light grey tone, to make it easier to differentiate it from the ground. This would make for even finer control during complex turns, rolls, and dives. The monotone sound of the plane's engine can grow monotonous after a while, and while it can be toggled off, it would be nice to have a volume control such as *Airborne's*. And perhaps the ultimate version of this game would run on Appletalk, with individual networked Macs flying separate planes. This is getting beyond the point of simple enhancements, but one can only hope...

Gato and *Fokker Triplane* are both fine games in their genre. The average gamefreak should derive many hours of enjoyment from both of them. After experiencing the joy of flying, however, you might find the deliberate pace of submarine warfare somewhat like returning to one floppy drive after having worked with a hard disk for a while.

TurboCharger

From Nevins Microsystems, Inc.

Requirements: ≥512K

Not compatible with hard disk drives, not copy-protected.

Macintosh.Comments:

Reviewed by David L. Foster

If you love your 512K Macintosh but have come to hate the wristwatch, TurboCharger™ from Nevins Microsystems will bring new fire to your relationship. TurboCharger is a simple yet powerful software package that makes your Macintosh work from two to five times faster. This enhanced performance is accomplished by making the Macintosh utilize its memory and floppy disks more efficiently using cache memory to dynamically create a RAM disk as you use a program, instead of tediously placing an application in a RAM disk at the start. Once you've tried TurboCharger, you'll relegate your traditional RAM disk utilities to the back of your disk holder and systematically turbocharge every application you own.

The performance of many programs on the 512K can be enhanced by fooling the Macintosh into thinking that some portion of its internal memory (RAM) is a disk. However, traditional RAM disks have some disadvantages. Current software only permits setting up a 316K -350K RAM disk on a 512K machine. A RAM disk of this size really cannot hold much more than a single large program in addition to your system files. Furthermore, traditional RAM disks must be built up from scratch each time they are used, and reconfiguring one after it's formed can be both tedious and time-consuming. Finally, data saved to a RAM disk is precarious indeed; a crash or loss of power will permanently banish your hard work from this world. TurboCharger can be configured to circumvent the shortcomings of traditional RAM disks, while still providing that sought-after increase in speed.

TurboCharger works by designating a certain portion of the 512K Macintosh's internal memory (RAM) to act as a buffer to store sections of floppy disk memory that are frequently accessed by the software you're using. As you work, TurboCharger transparently monitors your disk usage and keeps an image in RAM of the disk sectors used most often. You're never bothered by this "behind-the-scenes" work...you just see substantial speed improvement. The second time any information is needed by the Macintosh it is quickly fetched from RAM instead of from the disk again. For example, it normally takes 27 seconds to open a new MacWrite file. With TurboCharger, the first opening requires 22 seconds, and subsequent openings take only 12 seconds! The advantage of TurboCharger over a traditional RAM disk program is that TurboCharger can save everything to disk instead of to RAM, eliminating the risk of losing your work if the program crashes or your power fails! There's no need to set up a RAM disk, then save it to disk later. Optionally, you can configure TurboCharger to save to RAM just as a RAM disk program would, gaining additional speed at the risk of losing your work.

Easy installation, automatic operation

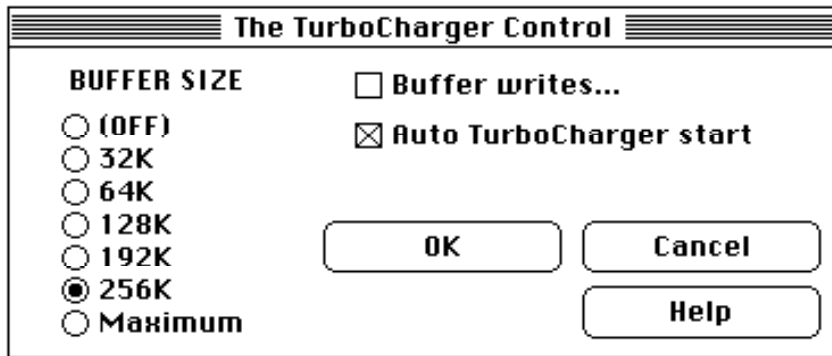
After you insert the TurboCharger master disk in either disk drive, it automatically ejects and asks for an application disk (using a dialog box). Upon insertion of a disk containing a System and a Finder, TurboCharger installs itself on that disk. The Mac is then reset, causing the modified disk to boot unless the mouse button is held down immediately following the reset. Because the TurboCharger installation program must then be reloaded in order to modify another disk, this last feature is a bit of a hassle if you wish to turbocharge several disks sequentially. However, TurboCharger can also be installed by copying a utility program called TurboControl onto any System disk using the Finder, so users with two floppy drives do have a faster alternative. Once you've added TurboCharger, it automatically installs each time you boot the application, revealing a TurboControl icon in the lower left corner of your screen.

Tailored to fit

The size of the TurboCharger memory buffer is fully configurable using the TurboControl utility program that is installed on each disk modified by the TurboCharger installer disk. When this program is launched you are presented with a window containing several radio buttons corresponding to the buffer sizes available.

Choices range from 32K to "Maximum," which on a 512K Macintosh corresponds to somewhere around 330-350K, essentially leaving you with a turbocharged 128K Macintosh. When TurboCharger is installed using the installer disk,

the modified disk is initially configured to maintain a 256K RAM buffer, a reasonable compromise between using too little and too much RAM. When memory-hungry programs such as Lotus's Jazz™ or Microsoft's Excel™ are run, smaller buffers are required so that sufficient RAM is available for the operation of these programs. Two boxes permit you to select automatic installation of a RAM buffer whenever that disk is booted and to decide whether you wish to fool application programs into writing data to the RAM buffer instead of to a disk. Obviously, writing data to the RAM buffer is as dangerous as saving a file to a RAM disk. In fact, it is even more dangerous because a system crash could scramble your floppy disk as well as lose your work. However, this configuration is optional with a turbocharged system; you don't have to use it if you don't wish to. Of course, this particular option should never be used with any software configuration that might crash. The program does remind you of these hard facts each time this option is selected, and the authors should be credited for providing such a useful feature even though its use entails some risk. The selection of this option really speeds up programs that constantly write data to disks. Microsoft's File™ and Telos's Filevision™ are typical examples of such programs.



Works with Switcher

Turbocharged systems can be used with Switcher™ provided you choose a RAM buffer small enough to leave sufficient room for other applications (typically, 128K or less). Also, it's a good idea to set up the RAM buffer before you enter Switcher, not after. With a turbocharged system it is possible to work with two or three programs using Switcher and have them perform nearly as fast as if they were being run from a RAM disk. If you're fortunate enough to have a 1- or 2-megabyte machine (upgrades to such memory sizes are currently available from several third party suppliers, such as Beck-Tech in Berkeley, Calif.) you can load more programs as well as use larger RAM cache buffers. Unfortunately, you are currently limited to using a RAM cache no greater than 256K, but you'll need 2 megabytes before that becomes much of an irritation.

TurboCharger is remarkably free of bugs, working with almost every Macintosh program available. The only program out of over fifty tested that refused to work with TurboCharger was ThinkTank™ Version 1.1. TurboCharger and TurboControl require only 14K of disk space, allowing installation on most application disks. Keep in mind when turbocharging that you must reduce cache memory size with memory-hungry programs such as Jazz™ before they'll work without crashing. Desk accessories requiring a lot of memory may refuse to work or may even crash when TurboCharger is used in its maximum size configuration. As a general rule, such desk accessories also have trouble working on a 128K Macintosh. The version of TurboCharger available at the time of this review (Version 1.1) was incompatible with the Macintosh XL or any hard-disk drive, but Nevins Microsystems claims to be developing versions that will be usable on such systems in the near future.

TurboCharger lets you use all of the memory in a 512K Macintosh even when your program doesn't need it or ask for it. TurboCharger is a far cheaper alternative to speeding up the Macintosh than acquiring a hard-disk drive, and it offers the security of saving to disk if you choose, unlike traditional RAM disks. Also, it installs and works "behind the scenes" automatically-- everytime you boot up a turbocharged disk. I highly recommend it as an indispensable time saver for any 512K Macintosh user.

Dr. David Foster is a staff scientist at Lawrence Berkeley Laboratory and an organizer for the East Bay Macintosh User's Group. Icon Review, a quarterly magazine which reviews Macintosh software, kindly granted permission for the inclusion of this review in the BMUG Newsletter.

Microsoft File

A Review

by Matt Williams

I bought my Macintosh in February, 1984 for two reasons: 1) to write business letters, and 2) to keep information (a data base) on business contacts. For my correspondence I have relied upon MacWrite, which I have found to be a quite serviceable word processing program. To meet my data base needs I bought Haba Systems' Habadex, a program which falls between the A's 1984 pitching staff and the Tokyo earthquake on the list of the greatest disasters to befall man, but, as they say, that is the stuff of another review.

To put File through its paces, I typed 480 records (here, names and addresses of persons) into a new data file. Before typing any data, however, the user must tell File the order in which data will be entered. For example, I could type in 480 consecutive first names, then 480 zip codes, etc., or I could type in the complete address (names, address, city, state and zip code) for one individual at a time. An entry sequence can easily be changed before all data has been entered, and the length of each line of information can be quickly shortened or lengthened, too. For my purposes I chose to enter data in this form:

FIRST	LAST	COMPANY	STREET	CITY	STATE	ZIP
NAME	NAME		ADDRESS			CODE

After all the records were entered, I set up a new form for mailing labels by dragging the field headings to their appropriate areas:

FIRST NAME LAST NAME
COMPANY
STREET ADDRESS
CITY, STATE ZIP CODE

Inserting the comma between city and state was done quickly, and without the need to type a comma 480 times. The spacing between first and last name, and state and zip code was perfect: because I ticked its "mailing label" dialog box, File knows the above format is for mailing labels, and will not place inappropriate spaces between fields.

Next, and after some delay (discussed below), I printed out mailing labels for all 480 individuals with an Imagewriter. Again, everything came out fine, and in the order I specified (numerical order by zip code--to save on postage-- and then alphabetically by last name).

File can do other tasks, of course. Pictures from MacPaint can be pasted into a record; calculations can be made automatically within a field (for instance, the user can multiply the contents of one field by that of another, something like what Multiplan can do; a data base can be smoothly merged into another; reports can easily be created, say, to show how many model X widgets Jones sold in June to buyers in Tuolumne County, etc. Furthermore, data from other Microsoft programs and from MacWrite can be transferred to and from File (useful with form letters). Help can be had while running File by pressing first the command key and then the question mark key, and then by clicking the mouse when the cursor is over the part of the screen where you need help.

How fast does File perform its functions? Alas, there is no one answer to a question such as, "How long does File take to sort 480 zip codes?" I am unsure, but File may be slower at sorting if there are more fields than my seven, and File may be faster if it only had to crunch five digit numbers, rather than some five and some nine digit codes as it did in my tests. As for hardware, I put File through the hoops using a standard 128K Macintosh. I ran the tests using first two disk drives and then again using only one drive. My findings using two disk drives:

Fall 1985 BMU G Newsletter

TASK	TIME REQUIRED
Open the data file	25 Seconds
Scroll from beginning to end of data file (first scroll)	30 Seconds
Subsequent scrolls	5 Seconds
Sort on one field which has been indexed	25 Seconds
Sort on one field which has NOT been indexed	1 Minute 45 Seconds
Sort on two indexed fields	50 Seconds
Sort on two fields which have not been indexed	3 Minutes 30 Seconds
Find a single record using an indexed field	5 Seconds
Find a single record using a non-indexed field	1 Minute
Print 100 address labels using highest quality print	23 Minutes

My findings using one disk drive (one application disk and one data disk):

TASK	SWAPS REQUIRED
Open the data file	10
Scroll from beginning to end of data file (first scroll)	1
Subsequent scrolls	1
Sort on one field which has been indexed	6
Find a single record using an indexed field	8
Change the form of displayed data	3
Print a single page	10

File can be used successfully with only one disk drive, it seems. It takes longer to perform a task with only one drive than with two, of course, in part because the hand is not quicker than the eye. But the time differences are not great (for instance, it took 90 seconds to open the data file, including the time needed for 10 swaps).

File is an easy program to use, but it was frustrating to learn how to use it. Sample programs are included on the application disk, but the instruction booklet often refers to two non-existent sample programs. Once I had the 480 names and addresses entered on the data disk, I wanted to print out mailing labels. Right on page 2 of the manual is a picture of address labels. "Easy", I thought. The section on printing has two pages of information about printing address labels just like the ones I wanted to print. "Piece of cake", I figured. Was I wrong! After spending five hours in the production of labels destined for the fireplace, I finally figured how to keep the names from creeping up on the labels as they were being printed. There are mistakes in the manual--the dimensions for "international fanfold" and for "A4 letter" are switched. Further, the choice (again, under Page Setup) for "computer paper" is not mentioned in the manual. Are mailing labels computer paper, fanfold, US letter, or what? Should there be top and bottom margins? There is no way to tell from the manual. Don't the folks at MS read their own instruction sheets? To print labels (File can only handle one label across the sheet of labels), chose Page Setup, then select the page size which can be divided evenly by the sum of the height of one label and one of the gaps between labels. Also, do not have any top or bottom margins. Oh, yes, "computer paper" is only to be used for 15 inch ImageWriters, but it does not say so in the manual. As you may have guessed, a lot is not in the manual. After I successfully printed out about a dozen labels, I asked it to print the remainder at one go. It started to print them, all right, but at the rate of one line every minute. At that rate, my work would have taken 32 hours! I asked that the printing be stopped so I could eject the application disk to unlock it. After whirring around in the Macintosh for 10 minutes I decided to risk trashing the disk, and turned off the machine. Once I unlocked the application disk all was well: 100 labels every 23 minutes. At no place in the instructions is mention made of locked disks.

I know how to use the program now, like it and will use it in future, but I do have concerns.

They are:

- 1) Make the instructions better. They are very readable, which is a blessing. But the examples in many places do not match the program;
- 2) The top-to-bottom height of forms (such as mailing labels) can only be changed in increments of 1/18th of an inch. 1/16th is a much more practical increment;
- 3) When pictures from MacPaint are contained within a data base, the time needed for sorting is excruciatingly long;
- 4) Disk Copy Protection. DCP is a gripe with nearly every program, so I won't expound mine here;
- 5) Using the 480 records as a sample, and with two fields indexed, a floppy disk probably will hold less than 5,000 names and addresses. A hard disk of some sort will thus be essential for some users;
- 6) Twice in one week File seized-up, and I lost control of the Macintosh. On both occasions, with disks whirring furiously and seemingly forever, and having exhausted all other alternatives, I turned the Macintosh off. The second time this happened, I lost some 200 names from the data disk. Make back-up copies! This bug needs to be eradicated.
- 7) The required formatting of each individual field (picture, number, date or text), once data has been entered, is irreversible. Remember the zip code field used above? If I chosen "number" for that field, all would have been well for US postal codes. However, for Canadian postal codes, the number field will not accept a valid code such as H3A 2J6. Think ahead!

Bottom lines: A good data base program, but it could be made an excellent one. Typically thorough MS instructions with typical MS mistakes. The Macintosh interface is a significant improvement over that of Multiplan. Deserves close examination by those interested in data base programs for their Macintosh. For a somewhat different view of File, see the review by M. Veljkov in the May 1985 MACazine.

Mind Prober Communication Edge

A Review

By Storm Jenkins

Mind Prober, and Communication Edge by Human Edge Software struck me as quite "cute" by not very substantive. They both require you to answer a number of true/false questions about the person's mind you wish to probe, or about both the person with whom you wish to communicate and about yourself. The problem struck me as two fold. First, the questions were too black and white, and a number of them could not be answered accurately by simply a true or false response. Thus you are forced to make judgments, thereby reducing the accuracy of the input. The other problem was that some of the questions related to such unusual circumstances that if you knew that much about the person (or even yourself for that matter), then I doubt you would need these products in the first place. I even tested the two products by filling out the question file on myself as well as two of my closest friends. The readout was just not accurate in either case. The products are "fun" but I would have to be absolutely certain about the way I responded to the questions before I would rely on the readout.

Trivia

By Philip Suh

available from Mirage Concepts Inc., 4055 W. Shaw - Suite 108, Fresno, Ca 93711

List Price: \$24.95

Trivia, as the name implies, is a "computerized" trivia game for the Macintosh, which comes on two disks: one program disk, and one data disk with the questions on it. Other "question disks" are being made available. Trivia allows up to six people or teams to play at a time, answering question in five different categories: Sports, History, Geography, Literature and Entertainment. Various options can be set, allowing players to choose a "winning point total," a time limit for answering questions, and the number of players playing at a time. There are also options for looking at on-line instructions or loading a previously saved game. Adding your own questions and answers are possible, but this involves a long and tedious process.

When the number of people to play is chosen, the program asks for the names of the players. One problem with this feature is that there is no way to correct mistakes once the names have been entered without repeating the whole process of selecting players and entering all the names over.

To play the game, a level of difficulty corresponding to varying point values is selected in one of the categories. The question is displayed on the screen. The player must verbally answer the question, and then someone else must check the answer with that displayed on the screen. If the answer is correct, the "correct" box would be clicked, if not the "incorrect" box would be clicked. When time runs out, a beep is sounded, and the "incorrect" box would have to be clicked.

For a computer game, I would like to have seen more "computer" functions such as automatically ending a person's turn when time ran out. I particularly would have liked to be able to answer question directly into the computer i.e. be able to type in an answer or chose from a list of answers. Trivia has no way to deal with this situation, and therefore, many things such as noting whether an answer was right or wrong must be handled manually.

Basically, the people at Mirage Concepts have just taken a board game and put it directly onto the Macintosh. What you have is simply a board game which replaces the board with the Macintosh screen and adds some "cute" music to add interest. There are no real special features that you would expect to see in a "computer" game. After a while, the same repetitious music gets annoying and unless you are truly fascinated with trivia, I suspect you would soon get bored. I was not impressed by this program and cannot recommend it.

Learning Microsoft Multiplan Learning Microsoft Chart

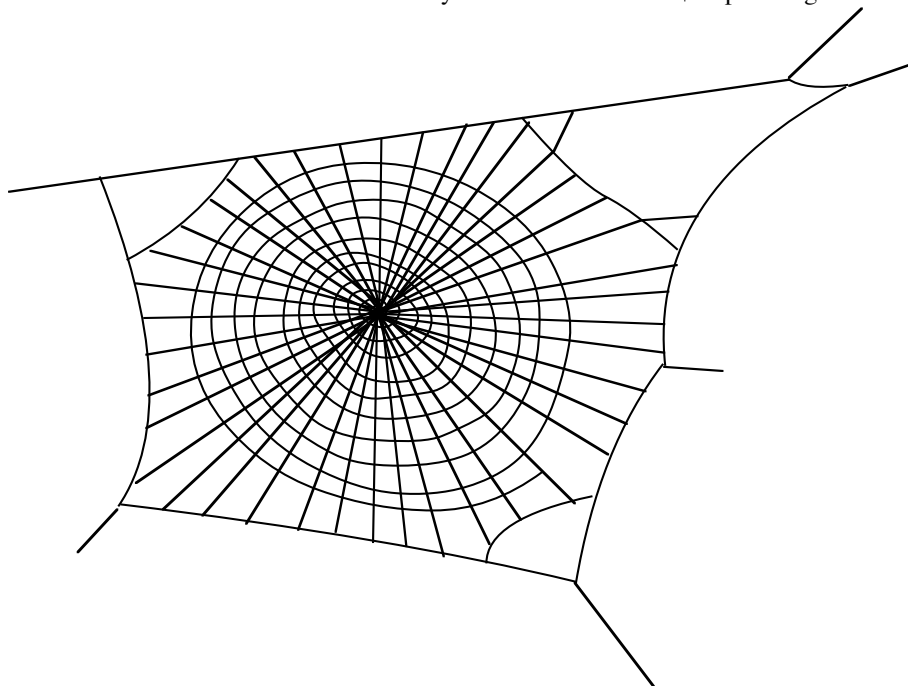
A book review by Storm Jenkins

Microsoft has begun to publish software products under the Microsoft MacLibrary label. These are products which were written by somebody other than Microsoft, but published and distributed by them.

Learning Microsoft Multiplan and *Learning Microsoft Chart* are two such programs. They come bundled together for around \$60. They are interactive tutorials for Microsoft's Multiplan and Chart programs. Both tutorials take you step-by-step, through the respective programs and check off each step as you go so that you always know which lesson(s) you have already covered. You can clear the checkmarks if you wish to start over fresh. Another feature which comes in handy is that you do not have to start with lesson one to get to lesson ten for instance. You can review any part of the tutorial alone without having to rehash areas you already know.

First I worked with the tutorial for Multiplan. I found it very easy to follow. It took me by the hand as we built a spreadsheet. The tutorial goes into ample detail on the basics and covers linking well. The major drawback is that I felt a little let down when the tutorial ended, it got me to the point where I had a good working knowledge of Multiplan's features and then when I was ready to get down to using functions, there was nothing. The tutorial simply did not cover any of the functions beyond SUM and LINK. The functions are really where I needed the help. Hopefully Microsoft will hear enough about this that they will write or commission somebody to write a tutorial which really gets into the nitty gritty. Even with this major flaw, I am glad that I got the tutorial because it did make some important aspects of Multiplan far more understandable. The salad was great but the chef quit before the entree was prepared.

Learning Microsoft Chart was far more complete. I never could get enthusiastic about learning how to use Chart until I purchased the tutorial. It truly made Chart easy and fun to use. It covers all the chart types available and how to link them with Multiplan so that whenever there was a change in the Multiplan's numbers, the change would automatically reflect itself in Chart. This one was really worth its half of the \$60 price tag.



MacDrawn by Dan Wood

Helix 2.0

from Odesta Corporation

Requirements: 512K Macintosh and external drive.

Recommended: Hard Disk (Not copy protected.)

Reviewed by David L. Foster

Although the graphic capabilities of the Macintosh have been used to considerable advantage by nearly all of the many database programs that have appeared for the Mac, none has utilized the Macintosh user interface more extensively or effectively than Helix. A marvelously innovative database manager, Helix packs the punch of a full-featured relational database without requiring you to first master a query language or build complicated algebraic formulas. Thanks to the Mac interface and its Mac-like design, Helix succeeds in providing both flexibility and ease of use in a powerful package that can be easily tailored to your own specific needs.

Although Helix was immediately recognized as an innovation in database management upon its release last January, its initial acceptance was hampered by its sluggish performance and incessant disk access. These problems have been largely swept aside by Helix 2.0, a new version that Odesta intends to ship by late summer. Helix 2.0 provides many enhancements over the first release, including several-fold faster performance and the ability to use and store pictures in both forms and data files.

One of the most unique aspects of the Macintosh is its use of icons or small pictures to represent various objects or functions. For example, copying a file from one disk to another is accomplished with a single fluid movement of the mouse rather than with a series of keystrokes. In a very similar fashion, the manipulation of icons by pointing, clicking, and dragging the mouse is used throughout Helix for nearly every aspect of managing your data files. The keyboard is really only necessary to generate labels for forms and for directly entering data.

Much of the initial effort in learning Helix really involves accustomizing yourself with the program's half-dozen icons. Once these icons and their functions become familiar, the creation, editing, and use of Helix data files becomes straightforward. The learning process is assisted by a manual which is amongst the finest I've ever encountered. In addition, a tutorial disk and cassette tape helps to guide the first-time user on a tour of Helix in a manner similar to the tutorial disk supplied by Apple with the Macintosh, thus giving the uninitiated an excellent overview of the program. A third disk contains five example databases that aptly demonstrate how Helix can be put to work managing inventories, checking accounts, form letters, and even a book library. By examining the structure of these databases, you can quickly deduce most of what you need to know to build similar databases on your own.

Helix is a fully relational database that can maintain a large collection of data files that can be accessed through an endless variety of entry and data retrieval forms. You can create these forms with the same ease that rectangles are drawn in MacPaint or that tabs are designated in MacWrite. Once entered into Helix, information can be easily accessed and examined from within any data file. Changes to data stored within Helix are automatically reflected in every dependent file or record throughout the entire database collection, making it easy to keep budgets, lists, and status reports current.

Helix not only stores data, but also provides the tools required to extensively manipulate and evaluate data in almost any matter imaginable. Helix 2.0 supports almost eighty different types of calculations ranging from simple addition and subtraction to complicated statistical functions. Even trigonometric and logarithmic functions are included, permitting the design of databases useful for scientists and engineers. Furthermore, complex calculations can be incorporated into your application without building complicated equations. All calculations are built in a flow sheet style one step at a time with answers routed from one part of an equation to another using arrows that are dragged across the screen with the mouse. This contrasts with the complicated algebraic formulas usually required by other database programs. Considered together, the relational capabilities and tools provided by Helix are sufficient to make this program far more than just a database manager. Indeed, Helix could quite conceivably manage the affairs of an entire company.

Fall 1985 BMU G Newsletter

The ability to use and store bit-mapped pictures created with MacPaint or brought into the Mac with image digitizers greatly expands the type of data that can be manipulated with Helix. Forms for data entry and display can be annotated with graphics ranging from company logos to pictures that could be used instead of numbers or text to reflect the program's analysis of data as you enter or modify it. Pictures of employees can also be stored right with their personnel records; this feature helps make the computer a lot less impersonal and would be very useful to a manager who has trouble connecting faces to names. It also represents one more small step towards the paperless office of the future.

If you ever wanted to get involved with database analysis and design, Helix is the best tool I can think of to get you started. You can model any sort of input form, display list, or report imaginable and then build relationships between forms and data. You can build it to interactively prompt data entry personnel with messages or information from other files. If you don't like the way you designed it the first time around, you can simply change it without feeling like you've wasted weeks of work building what you have. This is really the single most distinguishing characteristic of the program.

Helix is also distinguished amongst major Macintosh applications in that it is one of the very few that aren't copy-protected. In fact, Odesta encourages legitimate users to make unlimited backups of its disks and to store the originals in a safe place. The absence of a copy-protection scheme is definitely an asset because Helix can easily be copied onto and run from a hard disk (which is where most serious database programs belong anyway). Most other Macintosh programs demand the insertion of a signature disk whenever they are launched from a hard drive or from a copy of the original master disk.

Helix is not for everybody. In order to work with large databases you will need a hard disk drive; only about 250K of data can be stored using a two-drive floppy based system. If you can't afford a hard drive or if your needs aren't too complex, several other database programs for the Macintosh will provide better performance for a smaller investment. Good examples are OverVUE™ if you need fast sorts and don't need to store pictures, and Microsoft File™ or Forethought's Filemaker™ if relational capabilities aren't required.

In summary, Helix is a powerhouse database management system that has capabilities far beyond those of most database programs. Helix makes full use of the icon based interface of the Macintosh and although it can be confusing at first, it is very easy to learn and use. This is an exhilarating program not only because it is so different, but because it is one of the first database managers whose limitations more likely stem from a lack of imagination on your part than from a list of any built-in restrictions. If your needs are anything beyond the ordinary, I recommend you take a good hard look at Helix 2.0.

Dr. David Foster is a staff scientist at Lawrence Berkeley Laboratory and an organizer for the East Bay Macintosh User's Group. Icon Review, a quarterly magazine which reviews Macintosh software, kindly granted permission for the inclusion of this review in the BMUG Newsletter.

Macintosh Books For The Absolute I diot

a book review by Mike McIlwrath

The more the microcomputer becomes established as a tool for the layperson, the more that books designed to help the "computer illiterate" and "average" users understand their computers will find their way to print. In the past three years there has been a virtual explosion of "An Introduction to Your..." and "How to Use Your..." computer books written for every micro on the market. It is hard to find a bookstore these days that doesn't have a new "Computers" section with dozens of these books spilling over the racks.

Given the somewhat complicated, and often subtle, operation of an IBM PC or even an Apple II, it is not hard to understand why people might buy these books. Most computers are not easy for novices to master simply because they demand that the user learn some amount of technical, or at least technical sounding, information. Publishers, no doubt, made a killing on books written to help new computer owners feel at ease with their purchases.

But the Macintosh, one would expect, would virtually eliminate the market for these books. After all, the Macintosh interface is so easy to use that it caused a redefinition of the term "user-friendly" in the microcomputer public and the industry. Remember the commercial?

"In order to use this IBM PC you need to memorize these huge, complicated and tremendously heavy instruction manuals." (Clunk!!)

Fade into light piano music: "But to use this Macintosh all you need is this skimpy leaflet..." (plink.)

And the great thing about the Macintosh is that this claim is basically true; in fact, the leaflet itself is hardly necessary for mastering the computer. One of best clichés about the Macintosh interface is that it makes learning how to use the computer more of an intuitive process - meaning that you often don't need to know how to do something in order to do it. This is what makes the Macintosh so accessible to the person who has never before used a computer.

So, logic would seem to jump up and shout, publishers would be reluctant to print introductions to a computer which needs no introduction, right?

Not right. Probably because they had such lucrative experiences with books for all the other microcomputers, publishers have happily provided us with a more than ample supply of texts explaining the "how to" of "point and click."

Disregarding the argument that half the fun of the Macintosh is teaching yourself to learn it, these texts are always far less capable tutorials than the free Guided Tour that comes with the computer, and usually less informative than Apple's original skimpy leaflet. In order to pretend at pedantry, however, these books go beyond the assumption that the reader is a computer idiot; they act on the assumption that the reader is just an idiot. Penetrating deep into the obvious, these texts hardly go beyond what even the most incompetent Macintosh owner would eventually blunder through and discover for him or herself. For example, did you know that when you first insert a disk into the computer you should get a smiling Macintosh on the screen?

There are more than a dozen books now available, not counting the books of rewritten third party software documentation, to help the Macintosh owner understand and use the computer and its two most common and easy to use applications, MacWrite and MacPaint. These worthless books are usually expensive (in the \$10.00 to \$20.00 range) and criminally padded to fill the pages.

One explanation of why there are so many of these useless books is that they are not meant to be purchased by the Macintosh owners themselves. If these endeavors wind up making any money, it will be from gift sales generated by the friends and relatives of Macintosh owners. For example: I would never think of buying *Macintosh for College Students* (see below), but my well-meaning parents don't know that, and I figure it's only a matter of time before it winds up decorating my bookcase.

In fairness to the authors of the texts reviewed below, the abundance of introductory MacBooks might in some cases have resulted from the exuberance of Macintosh owners who just wanted to share their excitement with the rest of the world. In that case, let the rest of the world buy and read these books; they are clearly not needed by "the rest of us." The following books are merely a sample of what is currently being marketed to help the Macintosh owner:

Getting Started with Your Mac* (If You've Never Used a Computer Before)

Tim Hartnell and Rohan Cook (\$12.95, Ballantine)

This book is really for the person who never figured out how to use a tape recorder and therefore missed the Guided Tour to the Mac. While there is some very basic explanation of what the Macintosh is (this part is called the keyboard) that might be interesting to the book's stated audience, nobody needs this book to get started with the Macintosh. The Apple documentation is in fact very good at getting the new computer user quickly set up and going.

101 Ways to Use A Macintosh: A Practical Guide for the Rest of Us

David Thornburg (\$14.95 from Random House)

I first thought this book might be a good candidate for the dumbest book written for the Macintosh owner. After giving it a second look, however, I realized that it could actually be one of the dumbest books written for anything. "Produced on a Macintosh," the book is printed in double spaced New York 12 point text, but it isn't like you wish Thornburg had tried to make room for another 20 ways to use the Macintosh. Included are examples of how to make bumper stickers, ID tags, raffle tickets, crossword puzzles, a "family newsletter", etc. Written for the person who bought a Macintosh and needs to find a use for it. Reading through it I made me think of 101 things I would rather do with \$14.95...

Einstein's Illustrated Guide to the Macintosh

Jeff Einstein (Harcourt Brace, \$9.95)

There is nothing special about this book; it's just another shameless rewrite of the Macintosh and Write/Paint documentation that comes with the computer. The only interesting thing about this book is that author Einstein thinks he is up to something truly profound. In his pretentious preface to the book he suggests that books by computer scientists and programmers are confusing and poorly written. "Relax," Einstein boasts, "I'm a writer." This book is not a major contribution to world literature.

MacWrite Made Easy

Robert Wolenick (CBS Computer Books, \$16.95)

MacWrite *is* easy.

MacWork MacPlay

Lon Poole (Microsoft Press, \$18.95)

Yes, this is a book from the "Get Info" guy, MacWorld's respectable Macintosh tutor. Surprisingly, however, *MacWork MacPlay* is not nearly as informative as Poole's magazine column. The book is simply (groan) yet another general introduction to Macintosh. Probably one of the better MacBooks in that Poole's writing is clear and effective, but there really isn't anything beyond sheer volume to make it worth the 20 dollars. Had this book come out at the same time as the original MacBooks by Cary Lu and Doug Clapp, I might have been inclined to think more highly of it. But it didn't, and much of what Poole covers can be found in those earlier texts.

MacWrite: Guide for Students and Business Professionals

Alan Neibauer (CBS Computer Books, 17.95)

Notice how this book sounds like it's intended for specific, sub-groups of Macintosh owners, but really includes almost everyone who owns one? It should appeal to neither group, however. Just a rewrite of the original MacWrite documentation, only this costs \$17.95.

Macintosh Notebook

John Heilborn (two volume set from Prentice Hall, \$16.95 for *MacWrite*, \$17.95 for *MacPaint*)

Some people argue that if God had not intended for there to be books like this, he would not have made coffee tables. Expensive.

Practical Applications for the Macintosh

Eben Brown (Arthur Brown, \$9.95)

When you buy a book for its practical instruction, as opposed to buying literature for its spiritual or entertainment value, it is not demanding too much to require more than just quality from it, as in this case. Regardless of whether or not Brown's book really has anything new to say to Macintosh owners, it simply is not long enough to warrant the price. Via the Macintosh production technique (10 point, single spaced New York text), this \$1.50 pamphlet (at best) managed to become a ten dollar book without additional material.

Macintosh for College Students

Bryan Pfafenberger (Sybex, \$14.95)

This is another guide written for the person who already has a Macintosh but no particular use for it. While this sort of book might be useful to the college student who wants to know how a computer (Macintosh or otherwise) can help with school, it will not benefit the Macintosh owner, particularly the college student.

MacCats: 99 Ways to Paint a Cat on Your Macintosh

Floyd Flanagan (Scott, Foresman & Co, \$9.95)

As the title suggests, Flanagan uses the 99 cat illustrations to demonstrate the various ways to use MacPaint. Perhaps this book could be used to teach young children to use MacPaint, except that children are usually more willing to explore MacPaint's varieties on their own. And it might even have been interesting if Flanagan had just edited and published a book of MacPainted cats by different artists instead of going into the whole tutorial thing using his own illustrations. But at least the book has the charm to approach the market from a unique angle and it isn't as expensive as most others.

Voulez-vous or Vouler-vous?

Le Conjuguer: A Desk Accessory that Conjugates French Verbs

A review by Kelley Wicks

Le Conjuguer

list price \$49.95

Éditions Ad Lib. inc., 970 av. Salaberry, Québec, QC, Canada G1R 2V3, (418) 529-9676

Le Conjugueur is the type of Macintosh program which will interest only a few people, but for those who **do** have a use for it, it will be a provocative discovery. If you are learning French, wish to review the French you once knew, or if you commonly write in French, consider this gem.

Le Conjugueur was written by a university professor at the Université Laval in Québec. It is installed under your Apple menu as a desk accessory. It is fairly large for a desk accessory: 59K, which is a consideration if you do not have a hard disk, but Le Conjugueur will run on either a 512K or 128K Mac. This review was completed using a demo version. The program is not copy-protected and is easily installed on a hard disk. Since Le Conjugueur was written for native French speakers, the pull-down menus, on-line Help menus, and written documentation are in French. This feature is charming if you already have a working understanding of French, but could be problematic for novices. Actually, true to Mac programming standards, little use of documentation is necessary to make this application work for you.

The main function of Le Conjugueur is to provide an on-line conjugation resource while word-processing. Say you've composed a phrase and are unsure of the correct form or spelling of the verb. You pull down Le Conjugueur, insert the verb's infinitive, click ("cliquer"!) on the desired person and tense, and the verb form is displayed. Close the desk accessory, and the form you chose is inserted into your document at the cursor or selection. You may turn off the "Auto insertion" if you wish. I did not find a verb which the program couldn't recognize, providing I had spelled the verb correctly. In fact, Le Conjugueur takes anglicized verbs, such as "cliquer", as long as you provide a standard infinitive ending. Le Conjugueur works very quickly as you click among the various pronouns and tenses. Also, the Key Caps menu on a standard Mac system contains all the French accents (é, â, ç, etc.). Like the best of Mac applications, it allows you to get on with your thoughts, and doesn't get in your way.

This program would be excellent for people who are studying French. One could review conjugations by guessing the form which goes with a particular pronoun, then clicking on the pronoun to see the correct answer. This would certainly be more fun than falling asleep over an open grammar book! Verb form and usage in French is complex enough that years were required to compile the rules and exceptions which this program implements. Some verbs have different spellings depending on their context: Le Conjugueur prompts you for the context so it can give you the right verb form.

Francophiles, alert! Le Conjugueur is a charming and functional program well worth its price.

Using Microsoft File and Microsoft Word

by Linda Custer

Lots of people, especially around a university like UC Berkeley, need to manage fairly large files of references for their research groups. We have often had requests at meetings for information about anyone who has released software which can handle this kind of database. Maybe a program dedicated specifically to this task will be available someday, but odds are it won't have the flexibility found in combining a few programs already out on the market. Almost any commercially available database will do an adequate job of keeping track of your listings, but using Microsoft *Word* and *File* together allows you to take advantage of some special benefits. You will be able to print out your references in a form (that you design) acceptable for your thesis or paper directly from the database with no retyping required. Also, you will be able to create fields of any length (great for listing that one article that has sixteen authors when all the others only have a few).

Start by designing a form in MS *File* that contains all the information you want to know for each article. The one I designed is shown below as an example:

Then, do the most difficult part of the whole procedure. Type in all the information for each reference. If your database gets to be more than about fifty files, you may want to break it up into subsections for efficient print merging. However, *File* will be more than happy to handle hundreds of records in the same file quickly and easily. Make sure you make a few fields "indexed" fields. These should be the fields you will want to sort with most often. Probably the author and date fields will receive this treatment. When you are satisfied that your file is complete and once it is in the sorted order you want, choose the **Save As...** command from the File menu and check the box labeled for **MS Word merge**. Now quit *File* and run *Word*. Create a merge document following the following example.

```
DATA ()  
(now, design the way your bibliography will look, such as:)  
DATA  
( ) .      :
```

You'll need to repeat this as many times as you have records in the merge document, with a instruction between each line of the direction after the first (use the Paste command liberally):

```
( ) .      :  
( ) .      :
```

Fall 1985 BMU G Newsletter

() . :
() . :
() . :
() . :
() . :

...and so on. This will, when printed using the Print Merge command, give you output that will look like

Brynda E, Drobník J, Vacík J, Kálal J (1978) Protein sorption on surfaces measured by fluorescent labels. J Biomed Mater Res 12(3):55-65

Walton AG, Maenpa FC (1979) Application of fluorescence spectroscopy to the study of proteins at interfaces. J Colloid Interface Sci 72:265-278

By altering the form of the merge commands, you can alter the font and size of each element in the listing. You also can purge all extra blanks. And you can design a bibliographic form with which you are comfortable. When it comes time to print out your thesis, add a record to the database which asks whether or not each reference should be included in your final bibliography. Select only those references which say "yes" in that field and create a merge document as described above. Few (if any) databases will interface so easily with a word processor.

One note: I did notice one glitch in the system. When the files you bring over to *Word* are very large (say more than one hundred records), the necessary print spool file created will overflow a 400K disk. If you have a hard disk or auxiliary print spooler, you're all set. If you don't, you may need to break that merge document up into pieces about fifty records long. Try it and see. The printing process does take a while, but is worth the time you save by only entering your reference list once.



Making Color Photographs With The Mac

by Carter Compton Collins

Now you can make full color photographs on the Imagewriter. On page 49 of the Spring 1985 BMUG newsletter David de la Vega described a method for making color prints. I will describe how to make color *photographs*, particularly color *portraits*, which are especially difficult to produce because of the extreme sensitivity of the eye to slight variations in the color balance of skin tones. I will describe in some detail a method for achieving a reasonable color balance in portraits using Imagewriter printer ribbons with the subtractive primary colors, the so-called process colors, cyan, magenta and yellow.

With MacVision™ (or another video digitizer) one takes three pictures, each through a separate color filter over the TV camera lens. These color separation images are then printed out on the Imagewriter, each with a separate color ribbon. The paper is rolled back each time to exactly the same starting point so that the three color images are printed exactly in register on top of each other. The result is a three color photograph.

The materials you will need include a rigid tripod, light source, a television camera, an adapter ring and filter holder, and color separation filters, including Kodak Wratten #25 (red), #58 (green), and #47 (blue). Of course, a video digitizer is necessary to convert the TV image to a bit mapped image to be stored on disk. MacVision™, produced by Koala Industries, is quite easy to use. The pivotal items for true colors are the cyan, magenta and yellow process color ribbons which are available from Esoft Enterprises (P.O. Box 179, Owasso, OK 74055). Although one can separately store image files and register them on paper manually, Esoft also offers application software (Color Print™) which makes color printing much easier, faster and automatic.

In operation, install the MacVision driver software and copy the Color Print program and a system folder onto a working disk. You might also want to include MacPaint for touch-up work. This leaves room for two or three sets of color separation images, so you will probably eventually want to move the images to a separate storage disk in order to be able to take more pictures.

Using MacVision, portraits should be copied from a good color photograph since three separate *long* exposures are necessary. Mount the camera on a solid tripod so that it cannot shift when you change filters between exposures. For close copy work, close-up (portrait) lenses are useful. A combination of plus 2 and plus 3 portrait lenses permit copy work at about 8 inches. Arrange the light from about a 45 degree angle to avoid bright spots resulting from direct reflections of the source light into the camera.

It should be possible to utilize the scientific method employing filter factors and a neutral gray step wedge to calculate exposures for each of the three color separation images. However, one must also balance the settings of the brightness and contrast control knobs of the video digitizer which makes it tricky to find a satisfactory compromise for all settings which hold for all three color separation images. Instead, I will describe a fast converging trial and error method involving a bit of artistry to home-in on the appropriate settings.

Initial Settings

Attach the green filter and open the camera lens to f:2.0. Initially set the MacVision contrast and brightness controls to 60% and 75% of full rotation, respectively. Open MacVision, select **Adjust** and adjust the brightness and contrast controls so that the image tracing spans a brightness range from about 30% to 80% of the full "meter" scale on the screen. Next, select **Scan Window** and make final brightness and contrast adjustments for the "best" picture quality by eye. This is where the artistry is involved. With the green filter (which will make the magenta printing image), adjust the controls so that the highlights are not too broad, the pixels in the highlight regions should be drastically thinned out, but not deleted altogether.

Next attach the red filter and close the diaphragm one stop, to f:2.8. With **Scan Window** examine the image (which will become the cyan printing image). Make changes of the brightness and contrast controls until the broader and

flatter areas of the face (cheeks and forehead) are essentially devoid of pixel points in this image. Only the edges of major facial features should be modeled in cyan. (Shadows and darker colors, which may include the background, may also show).

Now attach the blue filter and open the diaphragm two stops to f:1.4. Examine the blue filter image in the scan window (this will become the yellow printing image) to see that there are sufficient pixel points in the yellow regions of the picture (perhaps hair and skin) and not too many points in the highlight and white regions. Make adjustments in the controls only if necessary.

Take Pictures

Once again attach the red filter, close the diaphragm two stops (e.g. to f:2.8), and examine the **Scan Window** image for the original criteria of the cyan print. Make small changes of the brightness and contrast controls only if necessary. Note the final settings for use in making future color photographs.

Select **Scan Screen** to capture a full sized image. Then select **Save Screen** and the cyan printing image will be stored on disk as a file called "Screen 0". Press the space bar to return to the MacVision screen.

Attach the green filter, open the diaphragm one stop to f:2.0 and select **Scan Screen**. You may make a small (perhaps 1%) adjustment in brightness to fulfill the criteria of the magenta image. **Save Screen** will store this image as a the file "Screen 1". Press the space bar for the MacVision screen.

Attach the blue filter, open the diaphragm one stop to f:1.4, select **Scan Screen** and the yellow printing image will fill the screen. Again, you may wish to make a 1 or 2% adjustment of brightness or contrast. Click **Save Screen** and the image will be stored as disk file "Screen 2". Press the space bar and the MacVision screen will appear for the last time. Close MacVision by selecting **Quit**. The contents of the Color Print disk will appear as icons. You can now change the names of your color separation images to whatever you want, but postscript each with its printing color. For example, you could call Screen 0, Diana.C; Screen 1, Diana.M; and Screen 2, Diana.Y.

Make Prints

Before Making color prints, turn off the Imagewriter and clean the printhead as instructed in the Imagewriter instruction manual, page 28. Ink of darker colors accumulates around the ends of the printhead wires and can be transferred to the lighter colored ribbons to discolor them. Also clean the paper guide rollers. Next, insert the yellow ribbon cartridge into the Imagewriter. Make sure it is completely seated.

Turn on the Imagewriter, open "Color Print", and select the name of your printer, (Imagewriter). Select the "Blue" box on the screen, and then click "Select Blue Document" and your named color separation images will appear in an overlapping window. Select the cyan image by double clicking, and a small replica of your cyan image will appear.

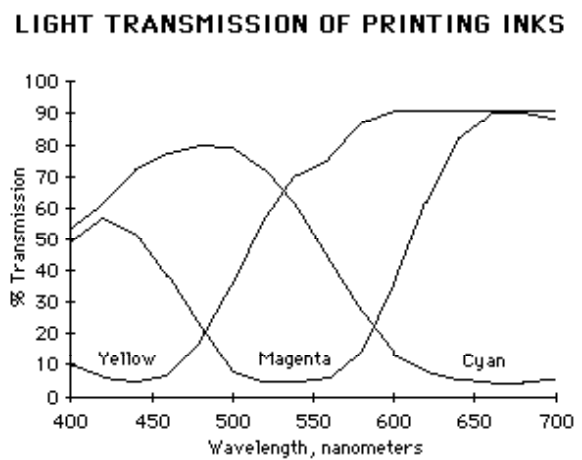
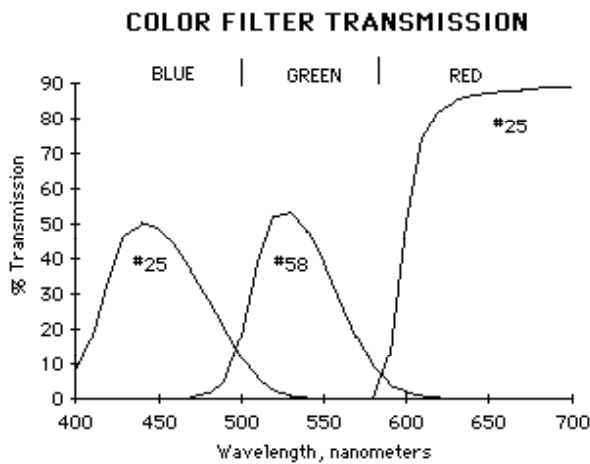
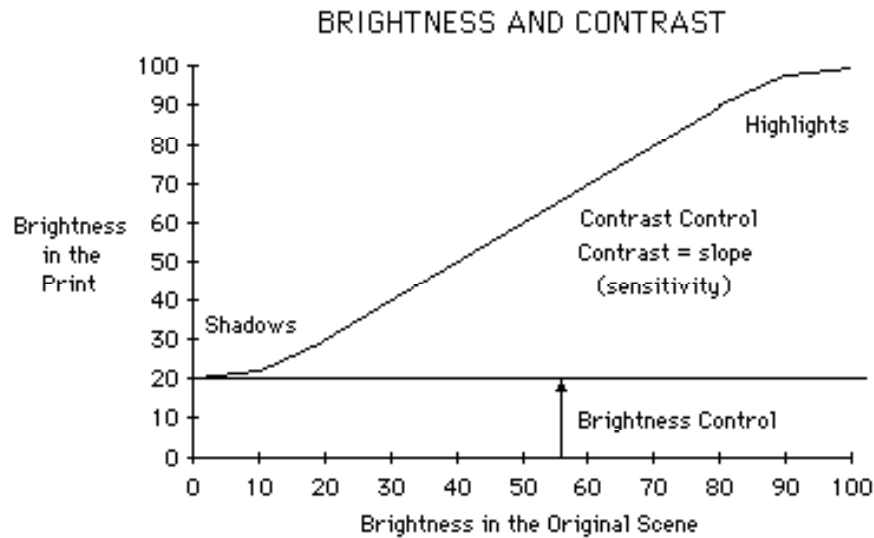
Close the currently displayed window and Select the "Red" box, click in the "Select Red document" box and select (double click on) your magenta image. Close the currently displayed window and select the "Yellow" box, click in the "Select Yellow document" box and select your yellow image.

Now select **Print** from the File menu. Paper movement will be automatic. Since the yellow ribbon is in place you may click the "OK" box and the yellow color separation image will be printed. Now change ribbons putting in the magenta color cartridge, click "OK" and the magenta image will be printed in register with the yellow image. Again change ribbons inserting the cyan cartridge, select "OK" and the cyan image will appear in perfect register with the previous two separation images. Select **Page Setup** from the **File** menu and then click "Feed the paper one page forward" and the Imagewriter will then roll your completed color portrait out of the machine. Congratulations! You have successfully made a full color photographic portrait with the Mac.

In subsequent prints the color ribbons will be depleted unequally, the yellow ribbon first, magenta next and cyan last. It is difficult to compensate for these changes. Use fresh ribbons (particularly yellow) for critical work.

For tutorial interest I include a graph of the functions of the controls: brightness, a zero level control, *adds* light; contrast, or sensitivity, *multiplies* the light level at each picture point sampled by the TV camera. Also of interest may

be the wavelength transmission characteristics of color separation filters and printing inks. Finally, Table I may serve as a handy reference summary of the color separation process.



COLOR SEPARATION FILTERS			PRIMARY COLOR ABSORBED	SUBTRACTIVE PRIMARY PRINTER RIBBON USED
KODAK WRATTEN NUMBER	PRIMARY COLORS	COMPENSATING F: STOP		
# 25	RED	REFERENCE e.g. F: 2.8	GREEN & BLUE	CYAN
# 58	GREEN	OPEN 1 STOP e.g. F: 2.0	RED & BLUE	MAGENTA
# 47	BLUE	OPEN 2 STOPS e.g. F: 1.4	RED & GREEN	YELLOW

Table I

Easy 35 mm Slide Production with a Mac

By Brent Fultz

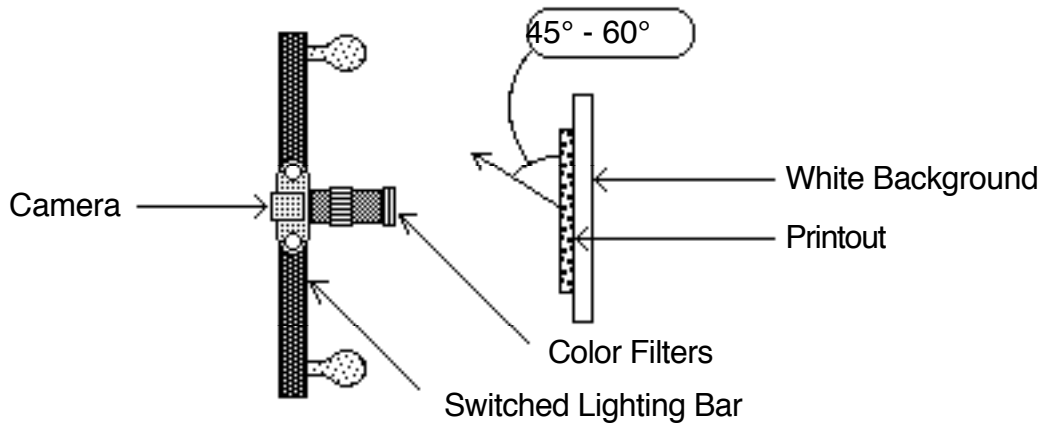
It takes both thought and hassle to prepare good visual materials for technical talks. Here I'll explain how I maximize the thought and minimize the hassle of preparing 35 mm color slides.

Strangely, in my field of materials science, I have not yet met another technical speaker who uses Macintosh-generated images. It seems that many people are prejudiced against the jaggies which are characteristic of digital images, and place a premium on conventional pen and ink technical artwork. However, the success of any technical slide is determined by how efficiently it conveys the concept being explained. I believe that Macintosh-generated images are superior in this respect. The Mac almost invites me to reconstruct the image, and while I am redrawing I am looking at the figure, familiarizing myself with it, and thinking about the concept that it helps to explain. A technical speaker who submits rough sketches to a draftsman cannot be as intimately involved with his visual materials. So in addition to low cost and efficiency of production, I find that working with my images on the Mac has a more important advantage: it helps me organize my talk.

I try to prepare all images within the 3 1/4 x 5 1/2 inch window of MacPaint. I prefer MacPaint even for text-only slides because MacPaint is so convenient for repositioning text, and motivates me to enclose concepts in little boxes. To minimize the jaggies I use the largest font which is practical. Once in a while I'll photograph the Mac screen directly. Although this gives a pleasant, streaky-blue background, it requires good camera alignment to avoid the borders of the MacPaint window. Also, the curvature of the Mac screen aggravates the image curvature inherent in photographing flat subjects with amateur equipment. I prefer to print the images on my Imagewriter with a medium-dark ribbon before photography. Of course, the LaserWriter produces a superior image, but self-sufficiency saves time and money.

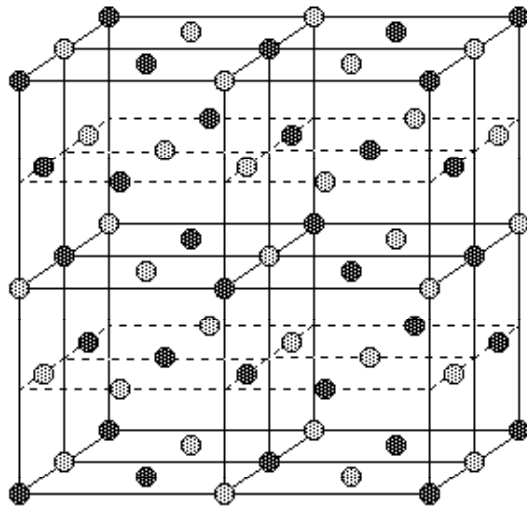
The lens must focus sufficiently close so that the image will nearly fill the frame. Longer lenses are preferred because they minimize the curvature of the photographic image. Although I have used a standard 50 mm lens, a 100 mm telephoto gives a much more rectilinear image. Fast lenses are unnecessary; the depth of field at f-8 or greater is necessary for keeping the entire image in focus.

I use ordinary Kodak Ektachrome 100 color slide film, and I get black writing on color backgrounds by using color filters in front of the lens. Ektachrome 100 film is color balanced for sunlight, and on windless days I often just take the printouts outdoors and tape them on white posterboard so that the background is pure white. I select a color filter and shoot the figures with the camera held in my hand or mounted on a tripod. If the exposure is set properly, the filter will impart a very deep color saturation to the background. Unfortunately, such a vivid background will distract from the image. An overexposure by 1 f-stop is about right; the background will be lightly colored, and the text will still be black. It is important that the camera is positioned perpendicularly to the center of the image, or the slide will look lopsided. It is also necessary to have the sunlight incident on the printout at a high angle, or textural features of the paper will be quite noticeable in the finished slides. These latter two requirements are somewhat incompatible because they require that the photographer nearly shades his subject. Nevertheless, good results can be obtained this way with a minimum of equipment.



A more controlled lighting scheme is shown in the figure. The subject-to-camera distance will depend on the lens, but I usually use a distance of 1 1/2 to 3 feet. Tungsten light bulbs with a 3400 K blackbody temperature (such as G.E. 88A-No.1) will cause a reddening with Ektachrome film. Type A Kodachrome film (ASA 40) is color balanced for this light, and gives outstanding results. However, I usually use Ektachrome 100 and compensate for the reddening with an 80B blue filter over the lens. This arrangement produces a near-white background from the paper, so a second color filter can be used on top of the 80B to get colored backgrounds. The lamps are mounted symmetrically about the camera with an incidence angle of 45 to 60 degrees to reduce glare. The G.E. lightbulbs last only a few hours, and change their color even sooner, so you can't burn them continuously. A lighting bar with its own switch is a major convenience. A second lamp with household tungsten bulbs is also useful for setting up and focusing, but it must not be left on during the photograph or observable reddening of the image will result (it isn't a pretty red either -- more like a rusty brown). With 3400 K tungsten lamps and the 80B filter the required amount of overexposure depends on the type of second filter. I usually underexpose 1.7 stops for an orange O(G), 1 stop for a second blue 80B, 1 stop for a green X1, and 0.3 stops for a purple FL-W. To some extent the vividness of the background is a matter of taste; bracket these exposures if you have extra film.

My favorite feature of Ektachrome 100 film is that there are many firms which offer 3 hour processing and mounting services for it, even on Saturdays (check your yellow pages). Frequently I have in my possession the 15 or so slides I need for a talk in less than 24 hours after I first sit down with MacPaint. This allows me to rethink and redo my subject material until it is right. Unfortunately, I have noticed a tendency to prepare my slides closer and closer to the departure time of my airline flight. This is dangerous. As I reduce the hassle in slide preparation, I must be careful not to reduce the thought. The slides leave images with many people which not only show something about me, but something about the Macintosh as well.



Brent Fultz

Loan Payment Spreadsheets: Amortization Calculations

by Kee Nethery

Real estate agents and car dealers carry around little books that tell you how much your monthly payment will be when you borrow money at some percent interest and pay it back over so many years. If you are doing any spreadsheet calculations requiring the monthly cost for borrowing money, the following equations will be much more accurate and flexible than any little book. The equation calculates monthly loan amortization payments. The general form of the equation is:

$$R = \frac{Pi}{1-(1+i)^{-n}}$$

Where:

- R = the loan payment per period (month)
- P = the principle (the amount borrowed)
- i = the interest fraction per period
(if the interest rate is 14% with a payment every month (12 months per year) then:
 $i = 0.14 / 12 = 0.01167$)
Note: Car ads quote APR financing. Annual Percentage Rate financing is discussed last.
- n = the total number of payment periods
(30 year loan with a payment each month then:
 $n = 30 * 12 = 360$)

So much for theory. For the following Macintosh spreadsheets (and equation processor) the exact entries for monthly loan amortization payment calculations are shown below.

- A1 or R1C1 will be the amount borrowed (P)
- B1 or R1C2 will be the interest rate in percent (%)
- C1 or R1C3 will be the payment period (years)
- D1 or R1C4 will be the monthly payment (R)

Multiplan	R1C4	=R1C1*R1C2/1200/(1-(1+R1C2/1200)^(R1C3*12))
Excel	D1	=PMT(B1/1200,C1*12,A1)
Crunch	D1	PMT(A1,B1/1200,C1*12)
Jazz	D1	=PMT(A1,B1/1200,C1*12)
Quartet	D1	-PMT(A1,B1/1200,C1*12)

TK!Solver

___St	Input	Name	Output	Unit	Comment
	8000	amt		\$	amount borrowed
	19	int		%	interest rate in percent
	4	yrs		years	payment period
		payment	{answer}	\$	monthly payment

S *Rule*
 payment=amt*int/1200/((1+int/1200)^(yrs*12)-1)

Fall 1985 BMU G Newsletter

Test your spreadsheet with these numbers to insure that you didn't make any typing errors.

borrowed	interest	years	payment
8000	19.0	4	239
25000	11.0	30	238
113355	5.6	11	1152
113355	12.2	50	1155

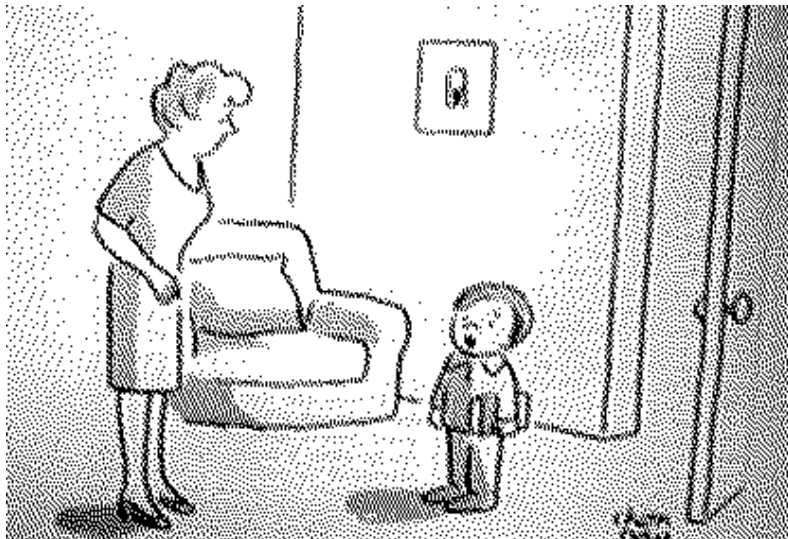
APR (Annual Percentage Rate) financing is used in advertisements because most people don't understand compound interest. I take out a loan and make one interest payment each year. If I borrow \$1000 at 20% and the interest is calculated once each year, the amount of interest paid is twenty percent of \$1000. $\$1000 * 0.20 = \200 interest. If the interest is calculated each month then on the next month I pay interest on what I borrowed plus the previous month's interest . . . so that the total amount of interest would be \$219.39. The interest rate is 20% but they would advertise 21.93% APR financing.

1000.00 *	0.20/12 =	16.67	end of month 1
1016.67 *	0.20/12 =	16.94	end of month 2
1033.61 *	0.20/12 =	17.23	end of month 3
1050.84 *	...		
1179.74 *	0.20/12 =	19.66	end of month 11
1199.40 *	0.20/12 =	19.99	end of month 12
1219.39	total amount owed at the end of month 12		
219.39	interest owed at the end of month 12		

To calculate the interest rate from an APR rate:

$$\text{interest rate} = \{((1+\text{APR})^{0.002})-1\} * 500$$

Note: All the equations use the decimal equivalent of percentages.
For example 0.15 equals 15 percent.



***“Well, Mom, I'm off to
the BMUG meeting!”***

Using The Font Librarian

Notes From The Author

By Dave Richey

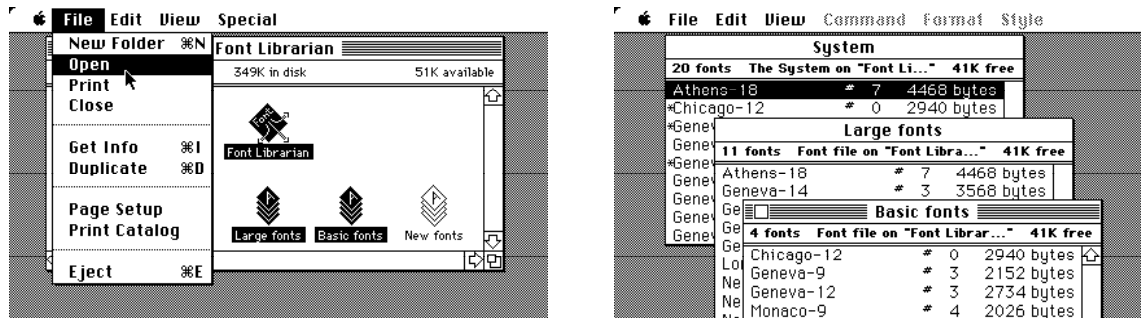
The Font Librarian (BMUG disk #21) is a shareware product that can replace Apple's Font Mover. This was a fun program to write, since I knew that the result would be a font moving utility that was a pleasure to use. I've always felt that one of the greatest things the Macintosh offers us is the huge choice of fonts, type sizes, and styles. There are lots of folks out there creating hundreds of new fonts, but it's hard to experiment with them using Apple's Font Mover since you can't really see how a font looks until after you install it. So... the Font Librarian was born.

What The Font Librarian Does

The Font Librarian does everything that the Font Mover does, but in a more convenient way. For instance, you can see what a font looks like and experiment with it, before you decide whether to install the font in your system file. This is great for working with custom font disks. In addition, The Font Librarian also has a number of advanced features for true font hackers. You can change the name or number of a font to anything (within reason!) you want. You can customize your font menus, or make a system file which uses a font other than Chicago-12 as the system font.

Getting going

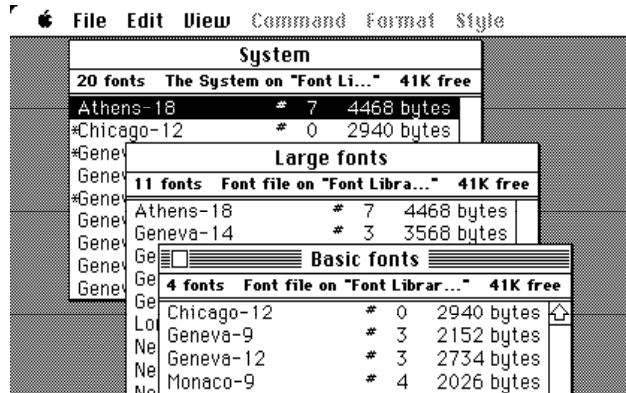
When you start up The Font Librarian and get past the initial screen, you get a desktop with a window showing the fonts currently installed in the system file. As you open up other files, additional windows are displayed to show what fonts are installed in those files. The Font Librarian will let you open up as many files as you want. If you select a bunch of files from the Finder, then shift-select The Font Librarian and choose Open, the selected files will be opened.



In the example, two font files were selected from the Finder in addition to The Font Librarian. When the program starts, it automatically opens the font files called "Large fonts" and "Basic fonts" for you.

Handy hints

Another example shows several fonts from different files displayed. Many people use the Macintosh screen dump feature to get a printout of their fonts to keep in a notebook. Another handy feature here is that a font can be displayed in many different formats at once. With the ASCII chart format, it's easy to compare two different fonts. You can also figure out which characters are which in a picture font such as Cairo using an ASCII chart display.



User-supported software

The Font Librarian is a "shareware" product, meaning that it's distributed directly from one user to another. (You may also hear the term "freeware" -- it means the same thing.) There are many programs of this type represented on the BMUG disks; some of the more famous ones are FEDIT, Red Ryder, and MockPrinter. The way it works is this. You copy the program from a friend, or from a users group disk, and try it out. If you like it and want to keep using it, the author of the program expects you to pay for it according to details supplied with the program.

There are many advantages to users in having good quality software distributed this way. The primary advantage is that it's cheaper. Since there's little or no advertising, and no retail packaging or mark-up, the price an author asks for user-supported software is generally about one-third of what a comparable program would cost at retail. So, be cool and support shareware!

Development notes

For you interested programmers, The Font Librarian is written entirely in Aztec C. It's heavily segmented so it will run smoothly even on a 128K Macintosh. This is important since large fonts can be up to 32K; it's easiest to move them from one file to another in a single chunk, so there must always be enough memory somewhere. On a 128K system, there's only about 40K left over as scratch. (If you don't want to deal with large fonts, it should run ok in a small partition under the Switcher.)

Fonts are always stored as resources in a resource file, but the way the resource is stored depends on the use of the font. For instance, the system font (usually Chicago-12) will always have the "system heap" flag set so that the system font will stick around after you leave an application. Conversely, the system heap flag must be cleared when copying a font from a foreign system file, since the program must place it on the application heap to do the copying. There are a number of similar issues that each have to be handled as special cases.

With a program written in C, some care has to be taken to remain compatible with the Pascal conventions built into the Macintosh. The many text strings displayed in the program's windows are handled as combined C and Pascal strings, with a leading byte count and a trailing null byte. This makes it possible to use UNIX-style string-handling from C, and still be able to pass the same strings to the Toolbox.

Where to find more information

For many Macintosh font fans, moving fonts around with The Font Librarian will be the ticket to happiness. But for those who want to know even more about fonts, Fred Huxham wrote an excellent article in the Spring 1985 BMUG Newsletter which describes how to modify or create your own fonts with the Font Editor. That article also has a good deal of general background information about fonts. For the technically-minded, Inside Macintosh devotes an entire chapter to the nitty-gritty of how the Macintosh deals with fonts.

Pascal Illustrated Fearlessly, Loathlessly

A Taste of Programming for Pascal Virgins

by Scott Kronick

The following was adapted from the beginning chapters of *Macintosh Pascal Illustrated: The Fear and Loathing Guide*, published by Addison-Wesley, June 1985.

The first half of this article shows you the shell of a Pascal program. With this shell you can begin writing your own programs. You will see that programs can be written without months of monastic discipline, deprivation, dispossession or, worse yet, attending school. Later, you will see Quickdraw and the Toolbox in action.

Program AirHead;



A Savage Journey into the Head of Pascal

All Pascal programs must begin with the reserved word **program**, followed by a name chosen by the programmer, and a line-ending semicolon. This is what you see in the first line of the program above:

program AirHead;

Pascal doesn't give a fig about lines or carriage returns, but semicolons are taken seriously. Semicolons separate statements.

All Pascal programs must contain at least one **begin** and one **end**. The last word of every Pascal program is **end** and it is always followed by a period like this: **end**.

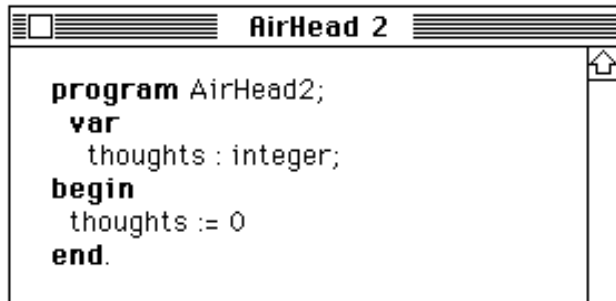
That is all that Pascal requires. In summary, a Pascal program must have no less than:

- The reserved words **program**, **begin** and **end**
- A made-up name chosen by the programmer.
- A semicolon after the made-up name.
- A period after the reserved word **end**.
- All of the above inserted in the same order as shown in **program Airhead**.

Declarations

Most programs have a declaration section. **program** AirHead has none. If **program** AirHead used any variables, they would be declared beneath the line **program** AirHead; and above the line **begin**.

Just for the heck of it, take a look at **program** AirHead2 with a variable. Notice the reserved word **var**. **var** says to a program, "Hey, pal, listed below are the names and types of variables that I want to use in this program."



```
program AirHead2;  
  var  
    thoughts : integer;  
begin  
  thoughts := 0  
end.
```

The variables listed below **var** might be thought of as a grocery list. The variable name is equivalent to an item's brand name. The variable type is equivalent to an item's kind. Name and type are separated by a colon. Every listing, even the last, ends with a semicolon.

Here is an example of a possible variable declaration:

```
var  
  rockyRoad : ice cream;  
  snickers : candy;  
  grandmaDora : cookies;
```

Programs often will use Pascal's predefined types such as *integer*, *real*, *char*, *string*, *text* and *array*. Here is a sample variable declaration using predefined types:

```
var  
  chocolateFix : integer;  
  sugarIntake : real;  
  bloatedFeeling : string;  
  constipation : packed array[1..26] of char;
```

You can read up on each of Pascal's predefined types in the Fear and Loathing Guide. For now you should know that:

- a. Each variable needs to be declared as to its **type**.
- b. The declaration occurs following the reserved word **var**, in the format:

```
var  
  variableName1 : typeA;  
  variableName2 : typeB;
```

The Main Body

You might have guessed by now that a computer program is nothing more than a list of instructions. When you write a computer program you are giving orders in the same manner that a parent gives orders to a child. A parent might say:

Go to the supermarket. Buy a loaf of whole wheat bread and a half gallon of low-fat milk. Here is three dollars, bring back the change.

The order of the instructions becomes important. You cannot buy the bread and milk until you go to the supermarket. You cannot get the change until you buy the bread and milk.

The order of a Pascal program is also important. You have got to know where your instructions **begin** and **end**. And guess what: Pascal does not execute a program in a strict, linear, top-to-bottom order.

In what order does Pascal perform instructions? The answer is illustrated by following the *pointing hand*. Drooling teachers and leaden books can try to explain program flow till hell serves Haagen Dazs, but not as well as the pointing hand of MacPascal's **Step** commands. (Explained in the Fear and Loathing Guide, the **Step** command is a MacPascal menu option that runs a program one line at a time.)

Whenever the order of program execution becomes confusing, use the **Step** command to inch your way through the program. The **Step** command will perform your program, putting output in the Text and Drawing windows the same as the **Run** command, yet at a line-by-line pace.

Step through program AirHead. Though the program does absolutely nothing, you can see that it is a actual program. The main body of AirHead consists of two words: **begin** and **end**.

The main body has no special name or reserved word to state its presence. The main body requires only Pascal's delimiters, **begin** and **end**. These reserved words serve throughout the Pascal program as bookends to hold together two or more instructions as a unit. Only in the main body do they serve the added purpose of beginning and ending a Pascal program.

The main body resides at the end of the program, following all other parts of a program. In forthcoming chapters you will be introduced to the building blocks of a program called **procedures** and **functions**. A block is simply a group of instructions which have been lumped together under an assigned name in order to accomplish a task.

In **program** AirHead, running the program with **Step** makes the pointing hand point to **begin**. The next **Step** points to **end**. Remember, the pointing hand points to the instruction which will be performed next. One more call to **Step** will end the program.

All programs begin with the first **begin** in the main body. All programs end with the last **end** in the main body. The instructions between **begin** and **end** of the main body instigate all other activity.

From this chapter on, you will be instructing a computer to go to the supermarket, buy bread and milk, and return with the change'. If the computer comes back with ice cream and cookies, you will just have to shrug and say four of the most satisfying words in the English language:

At least I tried.

What's Next

This program flips a rectangular coin when the cursor arrow is on top of the coin. Move the mouse off the coin and you get heads if the coin is black or tails if the coin is white. The key command for a program to read the mouse is *getMouse*, a predefined procedure from the Macintosh Toolbox. (The subsequent two chapters of the Fear and Loathing Guide show more elegant ways of programming this same task.)

```
program CoinFlip1;
  var
    x, y : integer;
begin
  frameRect(50, 50, 100, 150);
  moveTo(65, 70);
  writeDraw('BlackHeads');
  moveTo(65, 85);
  writeDraw('WhiteTails');
  repeat
    getMouse(x, y);
    if (x >= 50) and (x <= 150) and (y >= 50) and (y <= 100) then
      invertRect(50, 50, 100, 150)
  until button
end.
```

Declarations

Two variables, *x* and *y*, are declared to be of type *integer* under the reserved word **var**. The variable *x* will hold the horizontal coordinate and *y* the vertical coordinate of the tip of the mouse's cursor arrow.

Main

The instruction *frameRect(50, 50, 100, 150)*; draws a rectangular outline in the Drawing window. The numbers in parentheses, known as *parameters*, correspond to the top, left, bottom and right sides of the rectangle, in that order.

moveTo(65, 70); puts the Quickdraw pen into position for text to be written. (65, 70) is a coordinate point of the Drawing window and is contained in the rectangle drawn above.

You never actually see the Quickdraw pen. It is an invisible tool which positions and draws Macintosh graphics. What you *can* see is the ink from the Quickdraw pen when you execute a drawing or *writeDraw* command.

writeDraw('BlackHeads'); inserts the text of its string parameter beginning at the pen location. The string parameter is the letters between the single quote marks.

moveTo(65, 85) repositions the pen location to a point directly below where 'BlackHeads' was written.

writeDraw('WhiteTails') inserts its text at the new pen location.

Now begins a **repeat** loop. All the instructions following **repeat** up to the bold-lettered, similarly-indented **until** are contained in the loop.

The loop will be repeated until the boolean Toolbox function named *button* is true. Boolean means true or false. A Toolbox function is a group of instructions defined inside the MacPascal language. The Toolbox function *button* tests the status of the mouse button and returns a true or false value wherever the word *button* occurs.

button returns the value *true* when, and only when, the mouse button is pressed.

getMouse(x, y) is the first instruction in the **repeat** loop.

getMouse(x, y) is a Toolbox procedure which reads the integer coordinates of the mouse's cursor, and returns the horizontal coordinate to its first variable parameter and the vertical coordinate to its second variable parameter. In program CoinFlip1, the parameter variables are named *x* and *y*.

A Toolbox procedure, similar to a Toolbox function, is a group of instructions defined in the MacPascal language. The instructions are performed by calling the procedure's name. In the case of *getMouse(x, y)*, the Toolbox procedure assigns the integer coordinates of the mouse to the parameter variables *x* and *y*.

Following *getMouse* is an **if..then** statement. If the condition following **if** is true, then the action following **then** is performed. Otherwise, nothing is done and the program continues at the statement beyond the **then** action, which in *CoinFlip1* is: **until** button.

The **if** condition is:

$(x \geq 50) \text{ and } (x \leq 150) \text{ and } (y \geq 50) \text{ and } (y \leq 100)$.

The reserved word **and** means that the conditions on both sides of **and** must be *true* in order for the entire condition to be *true*.

Since *x* and *y* represent coordinate integers that were assigned in the *getMouse(x, y)* procedure, the **if..then** statement says: **if** the mouse's coordinates are within the specified rectangular area **then** perform the action *invertRect(50, 50, 100, 150)*.

invertRect is a Quickdraw procedure that inverts the dots within the parameter rectangle. If they were black they become white; if they were white they become black. The rectangle's parameters are the same as the parameters of the *frameRect* procedure.

The area specified by the **if** condition happens to be the same area enclosed by the *frameRect* box drawn in the first statement. Matching the *x* and *y* coordinates with the top, left, bottom and right parameters of *frameRect* deserves a few minutes of your time.

The **repeat** loop ends at: **until** button. Assuming the button is not being pressed, the *getMouse* procedure is performed again, the location of the mouse is checked by the **if** statement, and if the cursor is within the rectangle, the rectangle is inverted.

The best way to see how fast MacPascal performs this loop is to run the program with the mouse pointing inside the rectangle. Watch how fast the rectangle changes back and forth from white to black. Point the mouse outside the rectangle to stop the rectangle from inverting.

Are you quick enough to make the coin flip always stop at BlackHeads? Press the mouse button to exit the **repeat** loop and end the program.

The mention of BlackHeads makes Mr. Moss (The Fear and Loathing Guide's mentor) think of his adolescence, where hope and frustration stand out years after the pimples have passed. Hope turns to resolve, frustration becomes sadness; Mr. Moss's adolescence lasted too long. You have got a computer language to learn, but damned if Mr. Moss is going to let a chapter pass without a kindred message on priorities and patience.

Priorities? Patience? Lofty aims for a book on Pascal. Like the blind man playing impeccable Warrior at the video arcade: "Blind man", says Mr. Moss's girlfriend, "How in heaven do you play these machines?"

"Very poorly", he answers. Quarters exhausted, he steps back, bumps her. "But I'm improving."

68000 Minus Fear, Loathing

A Primer for Using the Macintosh Development System

by Scott Kronick

The following was adapted from *Macintosh 68000 Assembly Illustrated: The Fear and Loathing Guide*, to be published by Addison-Wesley, Spring 1986. A companion book, *Macintosh Pascal Illustrated: The Fear and Loathing Guide*, also by Kronick and Addison-Wesley, is currently available.

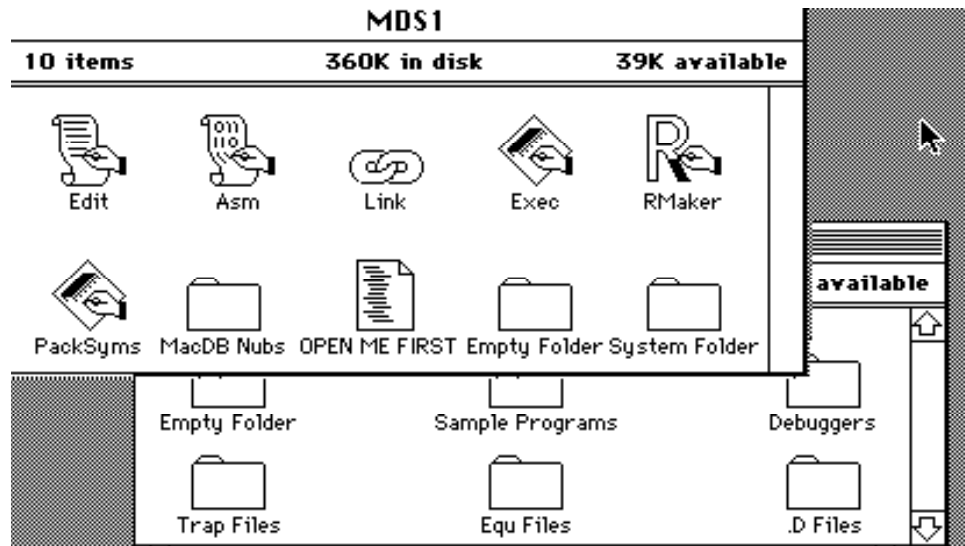
Programming is not simple, though many programmers are. If you are new to assembly programming, or all programming languages, you can learn by practicing with small programs such as the one in this article. The grammar and structure of assembly are easier to learn after, not before, you have practiced running small programs.

Apple's Macintosh Development System (MDS, for short) is an assembly language construction set contained on two disks. On these disks are numerous programs that help you build your own stand-alone application program.

Program development is easier and more efficient when using a Macintosh with two disk drives. If you do not have an external disk drive, you will have to swap files at various stages in a program's development. This is a pain, yet it is possible to create small assembly programs on a single drive, 128K Macintosh.

The program presented here can be created and run on a 128K Macintosh. Users of a 512K Macintosh will have access to a better error-fixing utility and faster program compilation. However, the kilobytes of intrigue and ingenuity in your brain are more important to good programming than the K in your computer.

Those Who Can, Do; Those Who Cannot, Teach; and Those Who Cannot Teach, Teach Programming



If someone else has used the MDS disks before you, or if you are using an updated version, the Desktop window described in the remaining paragraphs of this section might be different.

MDS1 initially displays ten icons. The text file OPEN ME FIRST contains information on the contents of the MDS disks. After you have read this document, you might want to make a paper copy and attached it to your User's Manual. Then, drag the icon to the trash can.

As the OPEN ME FIRST document suggests, certain programs and folders should be dragged to the trash in order to make more room on your MDS disks. But before you trash anything, you should make copies of both MDS disks, and only use the copies as your work disks. The original MDS disks should be safely stored in a cupboard with the picture of your first boyfriend or girlfriend, and used only in the event that your work disks are lost.

Edit opens a window for inputting the text of your assembly program. Asm and Link are programs that will make your assembly programs run as independent applications. RMaker, Exec, PackSyms, and MacDB Nubs are development system programs that beginning programmers will not need.

The System Folder contains operating information that Macintosh needs. You will not need to directly access anything in the System Folder, yet it must stay on the disk.

MDS2 initially displays six folder icons. All of these folders will be useful to you as your programming knowledge increases. But when you need more empty space on MDS2 for the programs that you will be developing, throw away the Sample Programs folder from your work disk. Besides, once you have written a couple of sample programs of your own, you will say of the MDS demos, "Gag me with a PC Jr." Heck, your Fear and Loathing programs will knock the sills of their windows.

You should always have the original disks or, better yet, third copies of the original disks from which to recover the programs and files that you throw out of your work disk. Errors may be frustrating, but they never ought to be catastrophic. At Apple, people say, "When the going gets tough, the tough turn pro." The Fear and Loathing Guide says:

When the going gets tough,
spend your nights with someone who isn't.

A Tiny Tease of a Program

In this section you will type, assemble, link, and run an assembly program that fills the Macintosh screen with a pattern. To do this, you will use three programs on the MDS1 disk: Edit, Asm, and Link. The whys and what for's of assembling and linking can be learned later. For now, just do it.

Creating an assembling language program takes four steps:

1. Entering the program in the Editor.
2. Assembling the program using the Assembler.
3. Entering a short linking program in the Editor.
4. Linking the newly assembled program using the Linker.

Double click on the Edit (short for Editor) icon. Then type the program as you see it below. (The assembler ignores comments--all text preceded by a semicolon--so you can omit them if you want.)


```
===== MDS2.3:Fourplay.Asm =====
INCLUDE MacTraps.D           ;define Toolbox traps
INCLUDE SysEqu.D             ;define ScrnBase

        MOVE.L   ScrnBase,A1   ;load screen base address
        MOVE.W   #5471,D0      ;screen size in long words
        MOVE.L   #$44444444,D1 ;screen pattern 01000100

Night   ;fill screen marker
        MOVE.L   D1,(A1)+      ;put pattern on screen
        DBRA     D0,Night      ;loop until size exhausted

Wait    ;user response marker
        SUBQ     #2,SP         ;make room on stack
        _Button  ;call button trap
        TST.B    <SP>+        ;test status from button
        BEQ.S    Wait         ;loop if status 0-no press

        _ExitToShell          ;return to desktop
.END                               ;code end directive
```

The indentation and spacing of the program code is important. Use the Tab key to make the code line up in columns. The upper case letters serve only as a convenience for readability.

Any errors you may have typed will not be noticed by the Editor. Only in the later stages of assembling, linking, or running a program will errors be noted.

Select **Asm** (short for Assembler) from the Transfer menu. A dialog box will appear asking you to give a name to save your program's text file. You can also select which disk drive you want the file to be saved on. You will usually find that the external drive has more room for your program files.

Type Fourplay.Asm, then click the Save button (or press Return). After the disk whirs and the file is saved, a minifinder appears asking you which text file you wish to assemble.

You might have to press the Drive button to find Fourplay.Asm, depending on which disk drive you have saved Fourplay. Take note of the suffix .Asm that you appended to the text file name Fourplay. These suffixes play an important role in identifying the files used in creating an assembly language application. Certain MDS programs create files with suffixes already attached. Other suffixes are the programmer's responsibility.

Double click on the file name Fourplay.Asm (or select the name and click the Assemble button). The screen will flash a few windows stating the condition of the assembly process, then return to the minifinder.

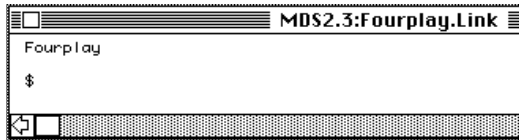
If the Macintosh presents an error message in a dialog box, you will have to reenter the Editor (by choosing **Edit** from the Transfer menu), and fix the incorrect code. Because of this error or errors, the program was not assembled. After you find and correct the error in your code, you should again choose **Asm** from the Transfer menu, and reattempt the assembly. The code should appear exactly as you see it in the previous illustration.

The Rewards of Linking

Now, choose **Edit** from the Transfer menu. You will now enter a very small text file that will link your assembled program into a stand alone application.

You might be wondering what is being linked to what, since the idea of a link normally involves more than one part. For now, assume that your assembled program is being linked to internal files that result in a stand alone application. Though the Linker is capable of linking more than one assembled programs into a single application, here you are linking a single file.

Select **New** from the File menu and type the following short code.



Choose **Save** from the File menu and type the name Fourplay.lnk in the dialog box. This link file should be saved on the same disk as Fourplay.Asm.

Now you are ready to create the stand alone application. Choose **Link** from the Transfer menu. A minifinder will appear. Double click on the name Fourplay.Lnk (or select the name and press the Link button).

A progress report of the link flashes on the screen. If all goes well, the application will have been created. Otherwise, an error message will inform you that the link was unsuccessful. In this case, you will have to correct the error in Fourplay.Lnk, and run the Link program again.

Once you have successfully linked your program, you are ready to test the stand alone application. If you exit from the minifinder by selecting **Quit** from the File menu, you will return to the Desktop, and find your application with its own icon. Better yet, go to the Transfer menu and select Fourplay. This will run your new application directly.

The disk purrs. The screen fills with lines. Move the mouse. Hmmm. Then press the mouse button.

(In the Fear and Loathing Guide you will type and run an animated graphics program called Cornered Coin, but for the BMUG newsletter, you get teased with only Fourplay. Check out *Macintosh Pascal Illustrated* for a Pascal version of Cornered Coin. Anyway, here's the punchline.)

It is over. You are no longer a programming virgin. Maybe you expected rockets and fireworks. Instead you got the Cornered Coin. The Fear and Loathing Guide offers, not for the last time, this hurtful, troubling wisdom:

Programming is less fun than being in love.

Once you are back to the Desktop, you will find that in addition to the application icon, the assembly process will have created two intermediary files, Fourplay.Rel and Fourplay.Map. When you transferred to the Asm program, Fourplay.Rel--a binary compilation of Fourplay.Txt--was created. When you transferred to the Link program, it was actually Fourplay.Rel that was linked into an application. In the link, the text file Fourplay.Map was created as a programmer's tool for following the symbolic names of an assembly program.

The intermediary files, as well as the text files which created them, are not necessary for the application to run. You can drag your application icon to any disk, and the program will work the same. That is what is meant by a stand alone application. However, you'll always want to keep around your text source files in the event you want to alter the program.

Attitude: Now that You've Got the Moves, Learn the Language

Ahead is your discovery of the instructions and grammar of assembly language. With the Toolbox at your command, Macintosh assembly offers many of advantages usually found only in higher level languages like Pascal, C, and BASIC. Much of the nitty gritty work is done for you.

With the Fear and Loathing Guide in hand, examine and experiment with its short programs. You don't need to take courses, go to seminars, or, worse yet, ask questions at computer stores. Programming artists write programs; programming talkers trade faulty information on subjects they do not understand.

When a book or article says *do this*, do it differently. When a program goes berserk because of a change you have made, laugh or curse, snicker or stomp. Yet never let a computer daunt you. Allow yourself to be daunted only while you are alone with your mate and the door to the room is closed.

A two byte Macintosh program

or

How to move the reset button from the side of your Mac to the keyboard

by Fred A. Huxham

I don't know about the majority of Mac users, but I use the reset button on my Mac all the time. If you're like me, and are constantly fumbling around the left side of your computer for that magical "programmers switch," then read on. You may even find what I say here to be useful.

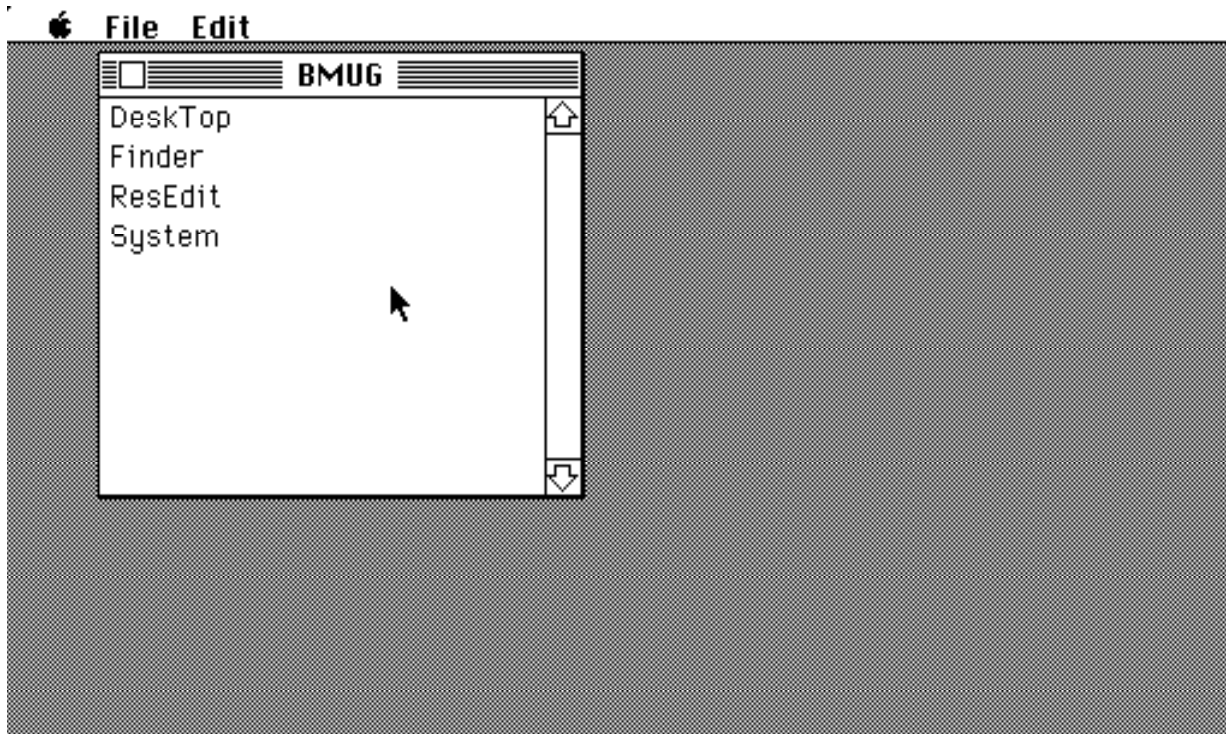
I'm sure you're familiar with the keyboard commands *Shift - Command - 3* which prints an image of the screen to disk and *Shift - Command - 4* which prints an image of the screen to your ImageWriter. Well what really happens when you hit *Shift - Command - #key* is the following:

The computer goes into the system resource file and looks for the FKEY resource that has resource ID #key. When it finds this particular resource, it executes the piece of 68000 machine code stored in it.

We can now deduce that in FKEY 3 is a piece of code that tells the computer to print the screen to disk and in FKEY 4 is a piece of code that tells the computer to print the screen to the printer. What I'm going to show here is how to create an FKEY resource that has a piece of code in it that tells the computer to reset.

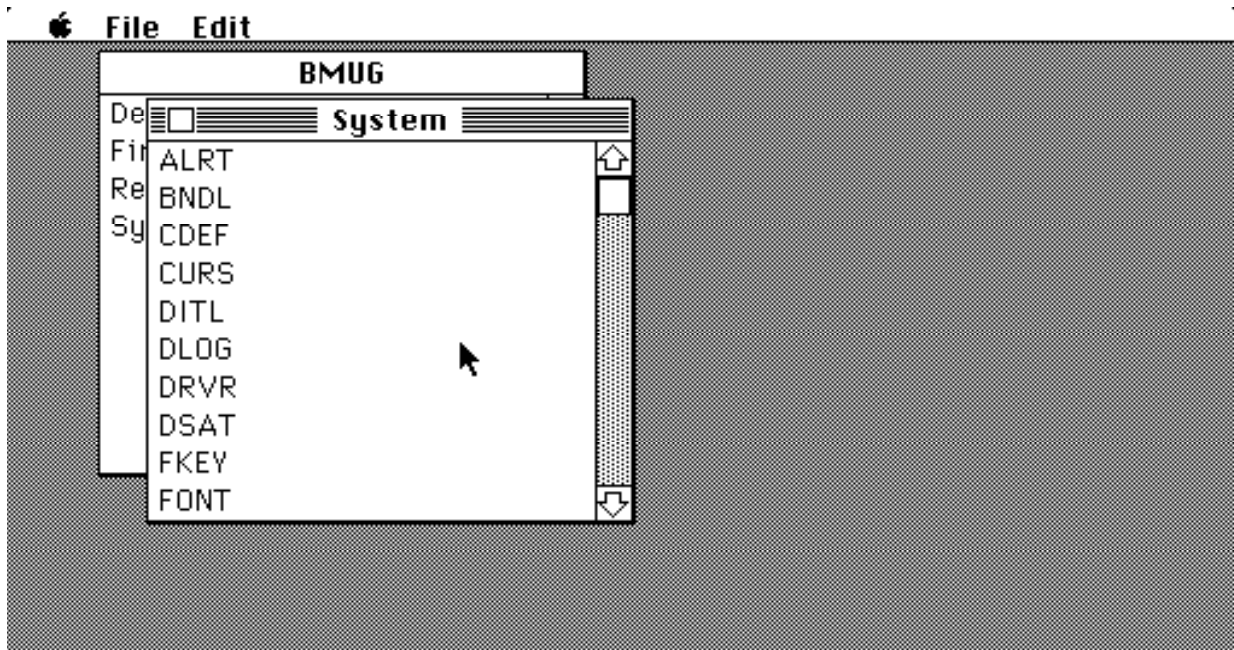
To do this you don't need an expensive compiler or assembler. All you need is the public domain program Resource Editor. Set up a disk with a System, Finder, and Resource Editor on it. Until you get good at using the Resource Editor you'll always want to play with it on a back-up disk.

Put the disk in your Mac and start up the Resource Editor. You should see something like the following:

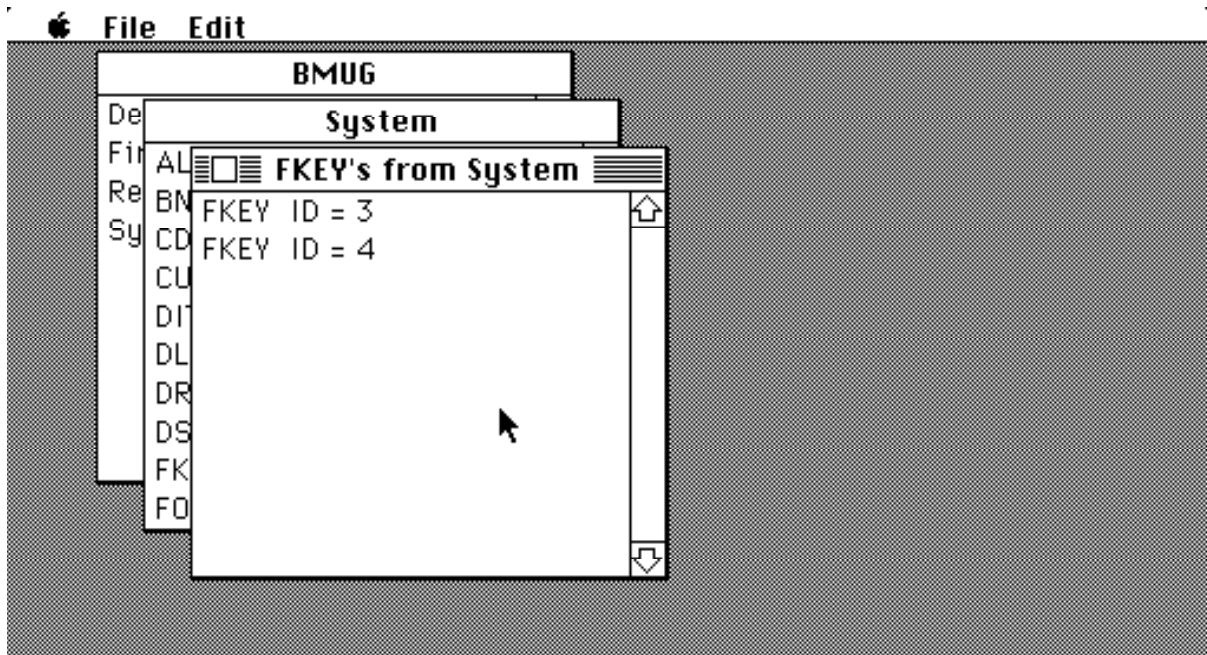


Fall 1985 BMUG G Newsletter

Double-click on the file name "System". You'll see another window open up like the following:

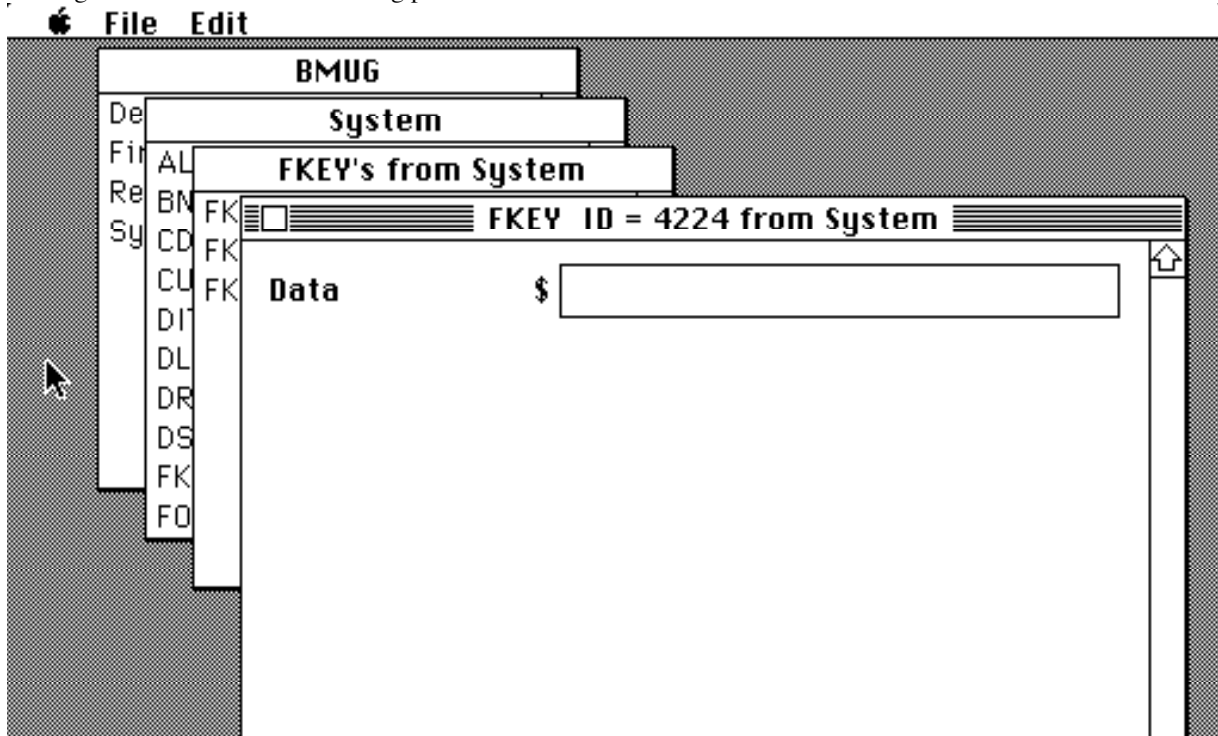


Now double-click on the resource type FKEY. Another window should open and look like the following:

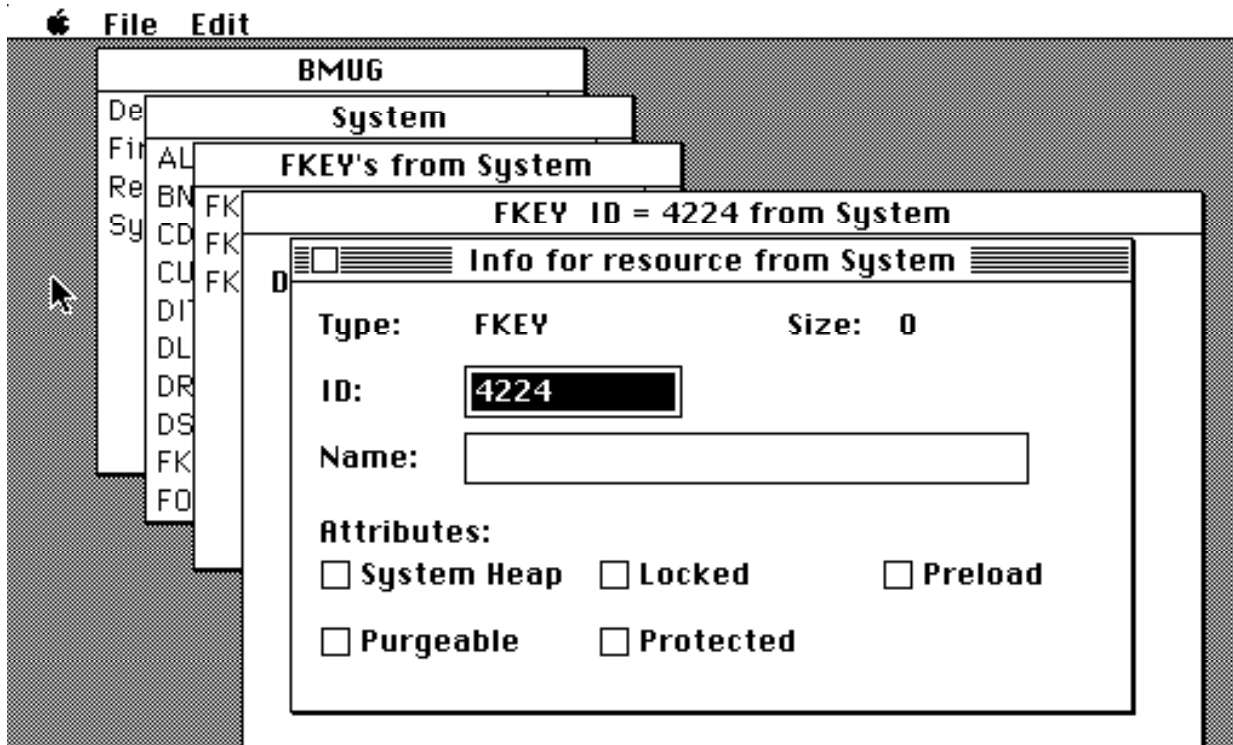


Fall 1985 BMUG G Newsletter

Go up the "File" menu and select the menu item "New." Yet another window will open up and your Mac should bear a striking resemblance to the following picture:



Now its time for us to decide which key we'd like to assign the reset button to. It has to be assigned to one of the number keys 0 - 9. Keys 1, 2, 3, and 4 are already used for other functions so we must choose from the keys 0 and 5 - 9. I think of resetting my Mac as "zeroing it" so I assign the reset function to key 0. To do this we select the menu item "Get Info" from the "File" menu. A final window will appear just like the one you see below:

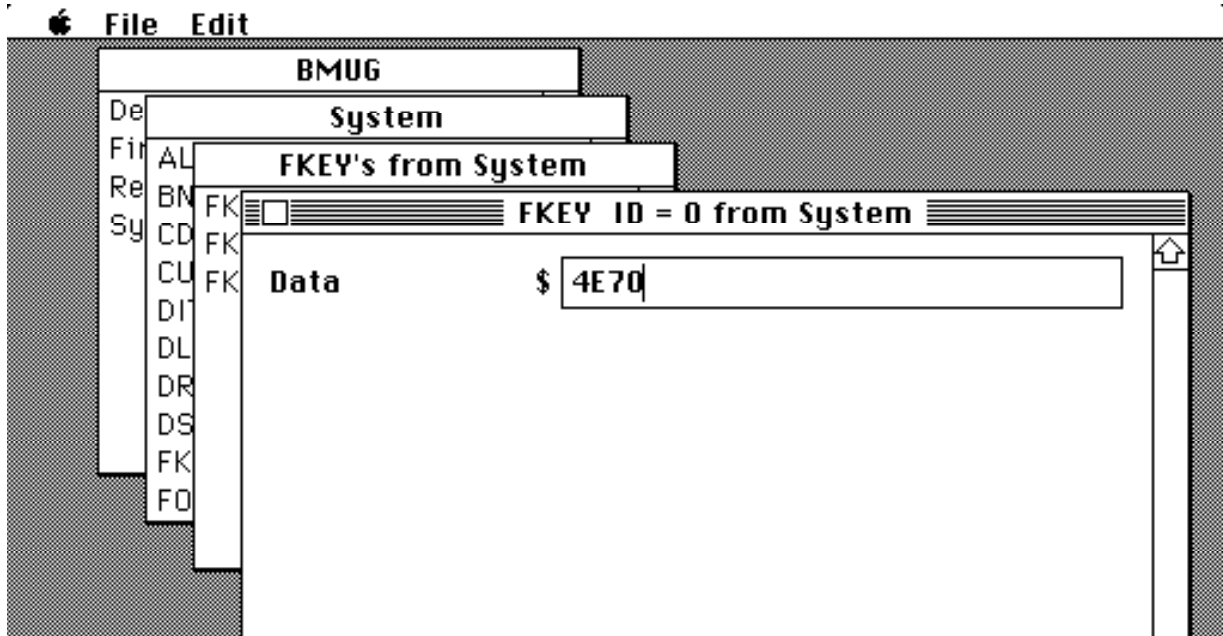


We just need to set the "ID:" value. Simply type "0" (that's zero), into the ID field of the previous window. That will assign the reset function to the *Shift - Command - 0* keystroke sequence. Now close the window. The topmost window should now have the title "FKEY ID = 0 from System".

Now it's time to do some serious hacking. We're going to write a program, inside the FKEY resource that we've just created, that tells the computer to reset. And we're not going to write it in BASIC, or C, or any other wimpy programming language. We're going to write our program in 68000 MACHINE CODE!! Lucky for us there just happens to be a single 68000 instruction - RESET (take a guess what that instruction does). This is the instruction we are going to use. All we need to do is to type in the characters 4E70 into the FKEY (See the picture below). The characters 4E70 are simply the hexadecimal representation of the instruction RESET. Now close up all the windows and quit. When the computer asks if you want to save the changes say yes. That's all there is to it. Now whenever you use the system you just modified and you hit *Shift - Command - 0*, the computer will look in the FKEY resource that we just created, see the instruction 4E70 (RESET) and will reset.

Now you can brag to all your friends that you've written your own Macintosh application (even though it is only 2 bytes long), and that you wrote it entirely in 68000 machine code. Good luck and have fun.

Fall 1985 BMUG G Newsletter



Scanned/drawn by Brent Fultz

Removing help files from MacPaint

by Linda Custer

You can easily reduce the size of the program *MacPaint* from 60K to 46K by using the resource editor to remove the help files--the two screens that you see when you choose the Introduction or Shortcuts menu options. Although these two screens are helpful when you first learn to use Paint, they are reproduced in the Paint documentation. Moreover, if you use the features listed in these screens often, you will quickly have learned which keystrokes perform which functions and no longer need to consult them.

First, take a disk with a system folder and MacPaint and copy it onto a blank disk. (Although I've tested what I'm doing here, the versions of resource editor out now *can* bomb and ruin the data on your disk. Try it on a copy first, and if it works, copy it again.) Copy the Resource Editor from BMUG disk 10 or 20 onto the MacPaint disk. Double-click on the Resource Editor, and get a cup of coffee while it loads. You should see a screen much like figure 1. Now double-click on the resource group called MacPaint. You'll get a screen resembling figure 2. Scroll down until you find the resource labeled *PICT*. Click on it once to turn it black, and cut! Congratulations. You just made MacPaint 14K smaller. Click on the close box of all the windows, answering "yes" to all the dialogs asking if you want to save your changes. And then quit the resource editor. When you return to the finder, you'll note that MacPaint is smaller.

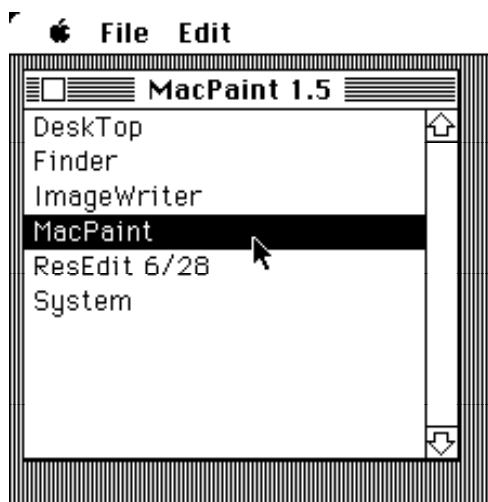


Figure 1. Choosing resources belonging to MacPaint.

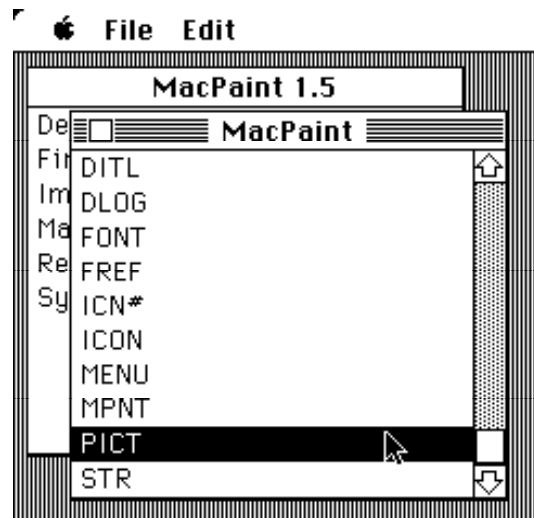


Figure 2. Choosing the PICT resources from MacPaint.

What actually happened when we cut the *PICT* resources? We removed the bit-map image of the help screens, since the help screens are merely bit-image MacPaint pictures. Now, when you select Introduction or Shortcuts, nothing happens. MacPaint looks for the image to display in the resource fork of its file, finds that it is missing, and decides to do nothing.

If you are daring, you can paste in pictures to replace the ones that Apple supplies. (This may not be particularly useful, but will give you practice with the resource editor.) Draw any screen in MacPaint and copy it into the clipboard. Leave enough space for the "Cancel" button that shows up in the lower right-hand corner of MacPaint with the help screens. Now enter resource editor just like before and choose MacPaint. Open the scrapbook, copy the image you desire, close the scrapbook, and hit Paste while the window for MacPaint is active. Double click on the new *PICT* resource which you created automatically, and verify that the picture you clipped made it into the program resource file. (See figure 3.) Next, choose Get info (refer to figure 4) and give the picture resource number 2401 (to be brought

up if you choose the Introduction menu option) or 2400 (to be brought up with the Shortcuts menu option). Try it! Note that if the screen you clipped is not the full size of a MacPaint screen, MacPaint resizes it to fill the whole picture area.

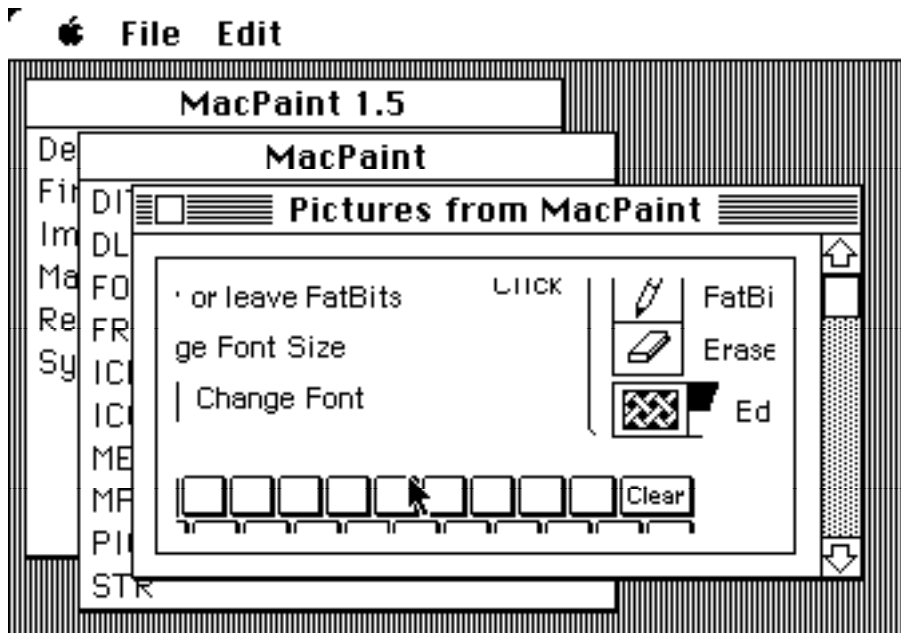


Figure 3. After double-clicking on the PICT resources, the pictures are revealed. Note that this picture has a light box around it indicating that it has been selected. Any cut or paste commands will apply to this picture.

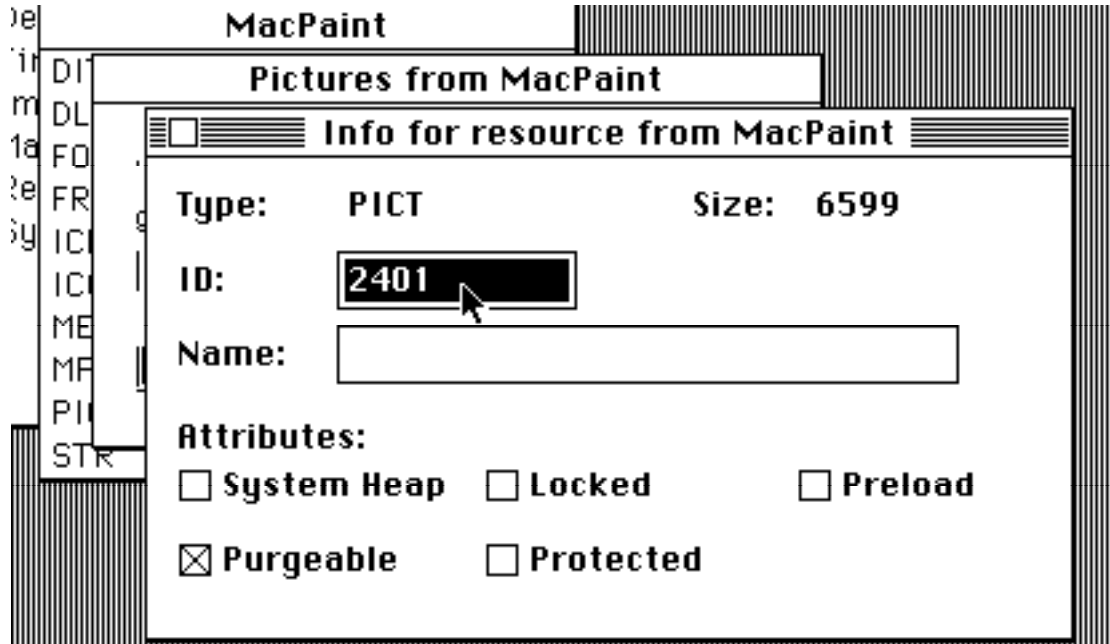


Figure 4. Choosing Get Info after selecting a picture brings you this dialog box. Notice that the ID number is 2401 and the picture alone takes up 6 or 7 K of space. The purgeable bit is chosen, so if Paint needs RAM space, it will unload this picture and get it from disk again later if it needs to.

Font Editor Supplement

by Mike May

The following is a supplement to Fred Huxham's font tutorial, available in the Spring 1985 BMUG newsletter. Described here are techniques for font editing, using the Resource Editor.

Extra Keys Available

In addition to the 188 keys from the standard keyboard you have access to 34 more characters by typing Option-```, Option-`e`, Option-`i`, Option-`u` or Option-`n` followed by a,e,i,o,u, sometimes y or n or repeating the Option-key. In the standard fonts this produces foreign language characters such as ñ,à, or É. The chart below gives the character number for these 34 characters.

Character	a	e	i	o	u	y	n	rpt
Option e	135	142	146	151	156	-	-	171
Option `	136	143	147	152	157	-	-	-
Option i	137	144	148	153	158	-	-	-
Option u	138	145	149	154	159	216	-	172
Option n	139	-	-	155	-	-	150	-

Lowercase Option accessible keys

Character	A	E	O	U	N
Option e	-	131	-	-	-
Option `	203	-	-	-	-
Option u	128	-	133	134	-
Option n	204	-	205	-	132

Uppercase Option accessible keys

Zero Width Characters

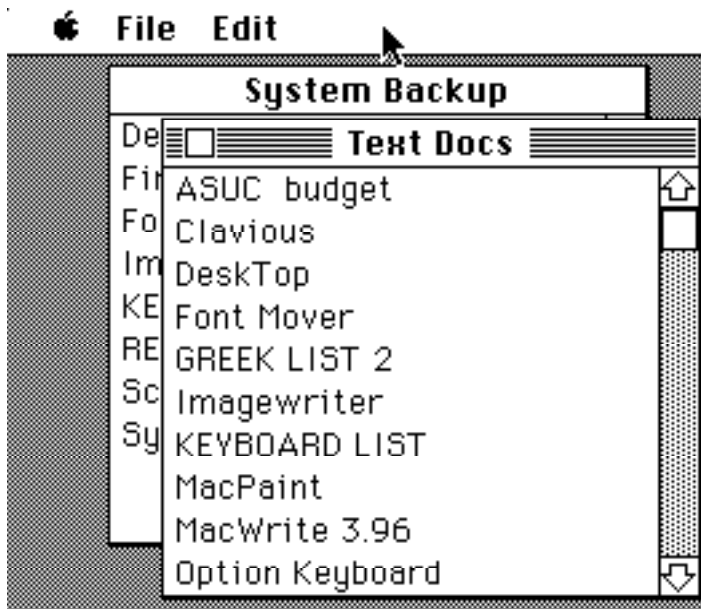
It does not seem to be possible to get zero width characters (thus the ability to underline or overline words or characters while in MacWrite) from the Font Editor. Equally bad is the fact that Font Editor will undo zero width on any font it is used on. This means you will have trouble if you try to make a ten point version of the Princeton Font (BMUG Disk # 6). This difficulty can be bypassed by using ResEdit from BMUG Disk #20.

CAUTIONS

- 1) ResEdit on the disk is a prerelease version of a very powerful program. I have crashed the system with it several times. It is thus even more important to follow the standard procedure of making a backup copy before you work on anything.
- 2) For most of the work of creating or editing a font Font Editor is easier to work with. Thus I suggest only using ResEdit for adding zero width characters.

To use ResEdit put it on a disk with a copy of the system file containing the fonts you want to add overstrikes to:

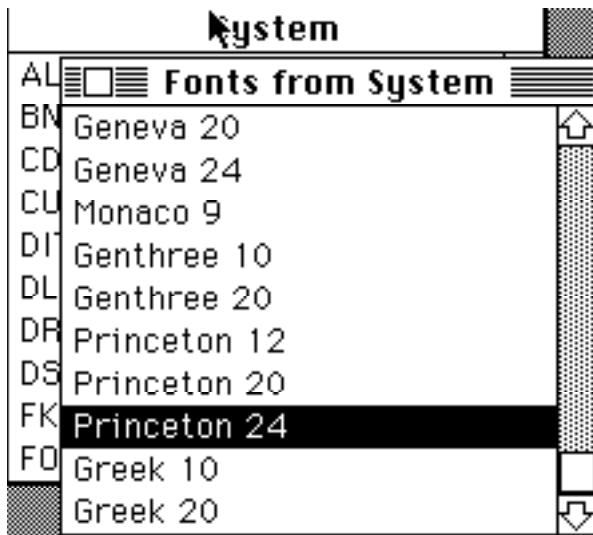
- 1) Double click ResEdit to start that program. You will see a box that lists the files on each disk in the machine.



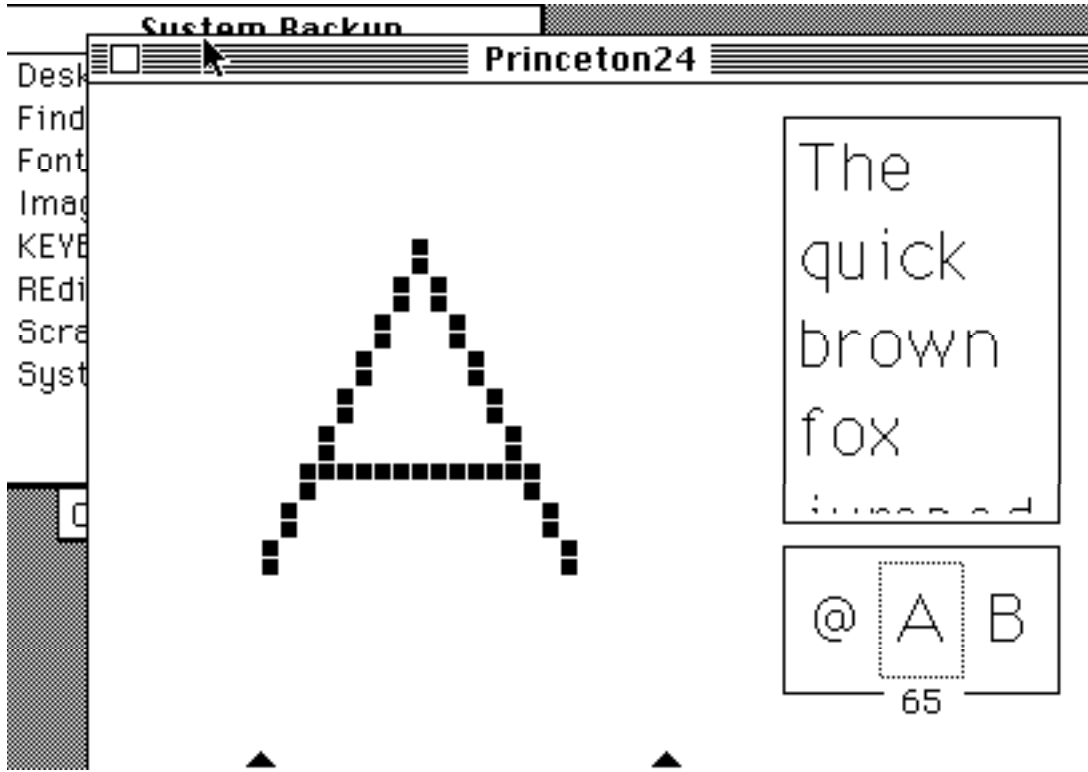
- 2) Activate the box corresponding to the disk containing your fonts (this is not necessary if you only have one disk in the machine).



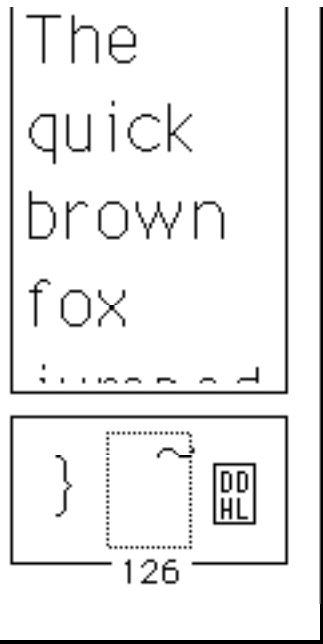
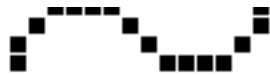
- 3) With the mouse select the file SYSTEM. Open the file by selecting open from the "File" menu.
- 4) In turn select and open FONT from the SYSTEM box and the font you are interested in from the font box.



- 5) At this point you have opened ResEdit's own font editor. You should see a picture something like the one below. The upper right box is for sample text so you can see the finished product. The lower right box shows the character on the screen, its number and the characters before and after it. New characters can be chosen either by clicking the lower right box or by typing the character you want to see. Notice the two small triangles beneath the fat bit map of the character. This determines the width of the character. These triangles can be moved in the same way you change tabs in MacWrite. By moving them together we get a zero width character.

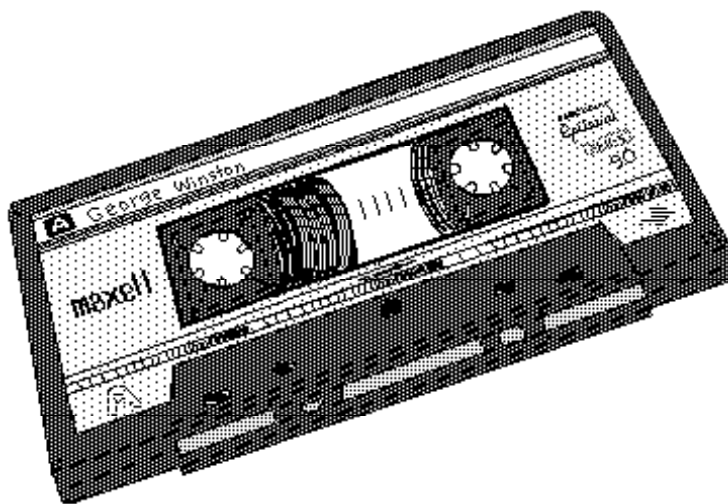


6) An example of a zero width character is given below. It is the ~ from the 24 point Princeton font. The designers of this font have followed Apple's lead by having overstrikes go after the cursor. This has the effect of typing Ñ as ~N. Since I prefer to think of it as N~ I move my triangles to the right so my overstrikes go before my cursor. The only inconvenience is that an overstrike will not show up as the first character after either a *return* or a *tab*.



7) An advanced point: since I set up my overlines and underlines for a normal width character I needed to add a forward overline and underline for wide characters like w and m.

Mike May is a graduate student in mathematics at Cal Berkeley and a Jesuit Scholastic.



Fall 1985 BMU G Newsletter

Dan Wood

Creating Your Own Icon

© 1985 by Fred Huxham

The following is adapted from the forthcoming book, Using the Macintosh Toolbox, a book on writing Macintosh applications using the C programming language, by Fred A. Huxham, Dave Burnard, and Jim Takatsuka, published by SYBEX, available late October 1985.

Half the fun of writing your own application is creating and assigning its desktop icon. For an application to have a custom icon, it must have a unique signature, and, in its resource fork, a number of Finder-related resources. In this section we will examine file signatures and all the Finder-related resources an application needs in order to have a custom desktop icon. While we are discussing each item required, we will be creating all the necessary resources to assign a custom icon to a sample application named SoundCODE. To create each of the Finder-related resources, we will use the Resource Editor application. The application SoundCODE is shown in Figure 1 with its generic application icon.

NOTE: To create a custom icon, you will need two applications, the Resource Editor, and an application that will allow you to set a file's creator and bundle bit (for example, the SetFile 2.0 desk accessory or the Fedit application). A working knowledge of both the Resource Editor and the other program you choose to use is assumed.

File Signatures and Bundle Bits

In order to have a custom icon, a Macintosh application must have a unique signature that the Finder can identify it by. An application's signature, often referred to as its creator, is a unique four-character sequence. The creator must be unique in the respect that no other application on a currently mounted disk can have the same four-character signature. In this example, we'll set the creator of our application SoundCODE to be the four character sequence FRED.

In addition to having a unique signature, an application must also have its bundle bit set. The bundle bit, which is described briefly in Chapter 12, is one of the Finder flags. When an application's bundle bit is set, the Finder copies the application's, ICN#, FREF, BNDL, and version data resources, if they have been created, into the Desktop file. The version data, ICN#, FREF, and BNDL resources are the Finder-related resources we will learn how to create shortly. The Desktop file is the invisible file on each disk that keeps track of all the custom icons for files and programs on the same disk. Once an application has a unique signature and its finder-related resources have been copied to the desktop file, its custom icon will appear.

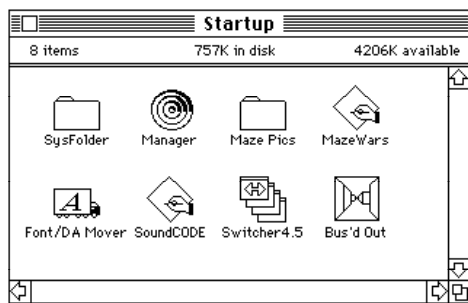


Fig. 1: SoundCODE with a Generic Application Icon

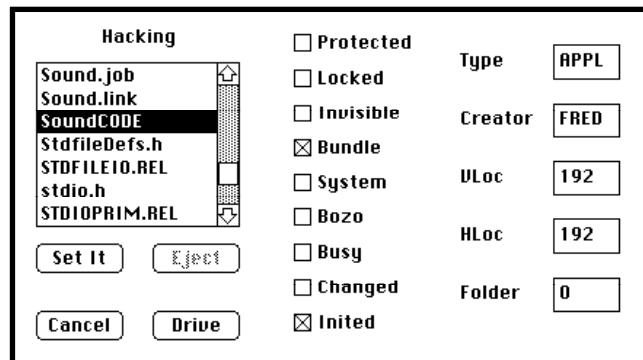


Fig. 2: Setting a File's Creator and Bundle Bit

There are many applications and desk accessories that allow you to set a file's creator and bundle bit. Fedit, a file and disk edit utility, and SetFile 2.0, a desk accessory, are two programs that do the job. Figure 2 shows SetFile 2.0 with the application SoundCODE selected, with its creator set to FRED and its bundle bit set. It is important, when setting a file's creator and bundle bit, not to change any of its other fields or bits unless you know what you're doing, since doing so can often cause lots of problems.

Now let's move onto the four Finder-related resources.

Version Data Resources

Each application must also have a version data resource, a special resource that has the application's creator as its resource type. Thus, our application, SoundCODE, will have a version data resource type of FRED. We can create a resource of type FRED with the Resource Editor application. When the resource type selector box of the Resource Editor appears on the screen, as shown in Figure 3, instead of selecting one of the standard resource types, type in the four-character sequence FRED. The resource doesn't need any data entered into it and should have a resource ID of 0, as shown in Figure 4.

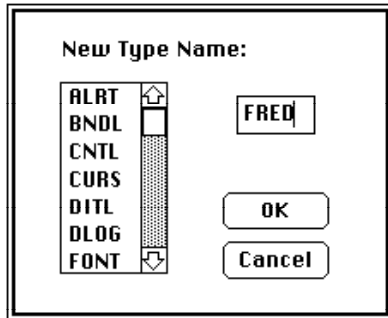


Figure 3
Resource Type Selector

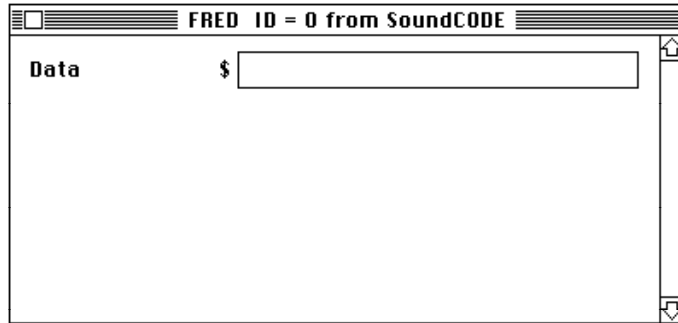


Fig. 4: Creating the Version Data
Resource FRED with ID number 0

ICN# Resources

The ICN# or icon list resource is the actual custom icon for the application. Creating an icon list resource is as easy as using fat bits in MacPaint. Figure 5 shows the Resource Editor template for creating ICN# resources. The icon on the upper-left is the application's desktop icon, and the icon on the upper-right is its icon mask. The icon mask determines how the desktop icon will appear when it is clicked with the mouse or moved across different backgrounds. A common situation is to have the icon mask the same as the icon. For SoundCODE we set the ICN# resource ID to be 128.

FREF Resources

A FREF or file reference resource needs to know only the type of the file the custom icon is being assigned to and the local ID of the custom icon. We can determine a file's type with either Fedit or SetFile 2.0. If we look back at Figure 2, we see that directly above the Creator field is the Type field. The file type for SoundCODE, which is also the file type for any application program, is APPL. Thus, for our sample program SoundCODE, we fill into the FREF resource pictured in Figure 6 the type APPL and the icon local ID of 0. The icon local ID can be any number so long as it is consistent with the local ID that we specify in the BNDL resource, which we will do next.

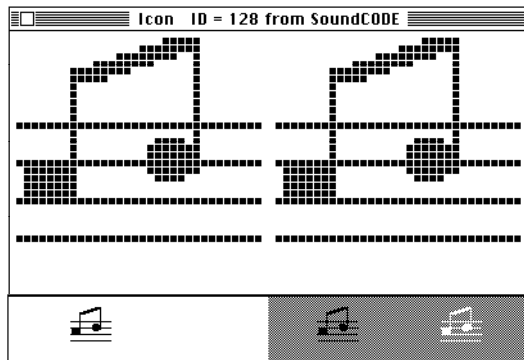


Fig. 5: Creating an ICN# Resource
with ID number 128

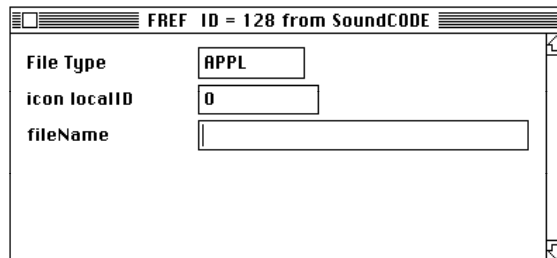


Fig. 6: Creating a FREF Resource
with ID Number 128

The fileName field of a FREF indicates any file that should accompany the application if it is copied to another disk. SoundCODE doesn't need any associated files, so we have left the field blank. We have set SoundCODE's FREF resource ID to be 128.

BNDL Resources

The last resource type an application needs for a custom icon is a BNDL, a resource that serves to bundle everything together. Refer to Figure 7, which shows the BNDL resource for SoundCODE, as we go over its fields one by one.

The first field, OwnerName, has to be set to the file's creator. For SoundCODE, of course, the creator is FRED. Next, the ownerID is set equal to the resource ID of the file's version data resource. The resource ID of SoundCODE's FRED, or version data resource is 0, so we fill in the ownerID field with 0. We don't set the numTypes field, the Resource Editor sets it automatically.

Next let's set up the mapping between resources' local IDs and resource IDs. These IDs can be set to whatever we want so long as we are consistent throughout all of the Finder-related resources. First, we will set up the mapping for the icon list resource of SoundCODE. We set the type field to ICN# for icon list, and then set the local ID to 0, as we did in the FREF resource. Finally we set the resource ID to 128 as we did when we created the ICN#. We then do the same thing for the file reference resource, setting the type field to FREF, the local ID to 0, and resource ID to 128. The value for the # of this type field is not set by the programmer. We set the BNDL resource to be 128.

Wrapping things up

Once SoundCODE's creator and bundle bit are set and all of the Finder-related resources are created, its new custom icon should appear as in Figure 8.

If you follow the general directions we have just given for creating a new icon and the new icon does not appear, it may be the case that the file or application you are creating the icon for previously had a custom icon assigned to it. To force the new icon to appear, double-click the mouse on the Finder while holding down the Command and Option keys. This throws out the old desktop file and recreates a new one, making your custom icon appear.

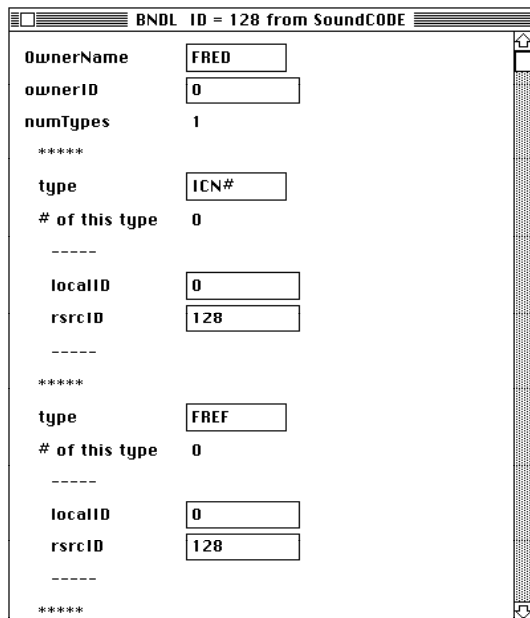


Fig. 7: Creating a BNDL Resource with ID Number 128

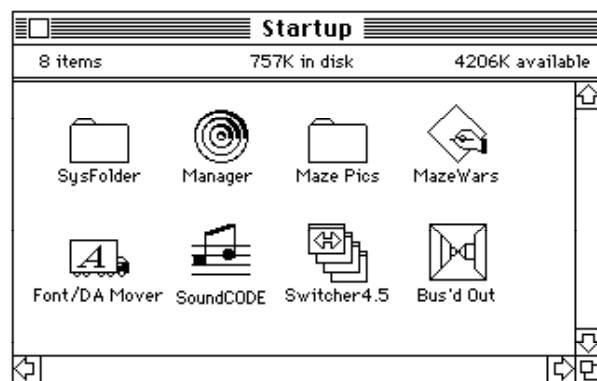


Fig. 8: SoundCODE with a Custom Icon

Setting up your Letterhead and Signature with MacWrite and MacDraw

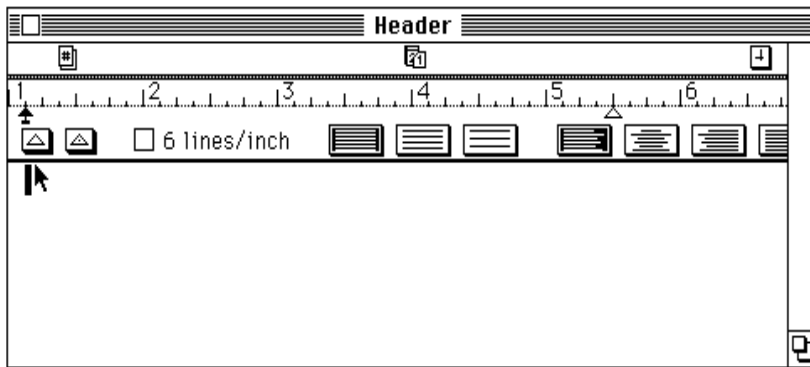
by Reese Jones

Introduction

Very often the major effort in writing a letter involves setting up the format including the letterhead, address, opening, closing, signature, and envelope. The following tutorial describes one way to set up a skeletal letter in MacWrite that can be used as a template for much of your correspondence and covers several of the more subtle features of using MacWrite and MacDraw.

Setting up your Letterhead

Begin by opening a new MacWrite file. Select the **Open Header** command under the **Format** menu and you will get a new header window.

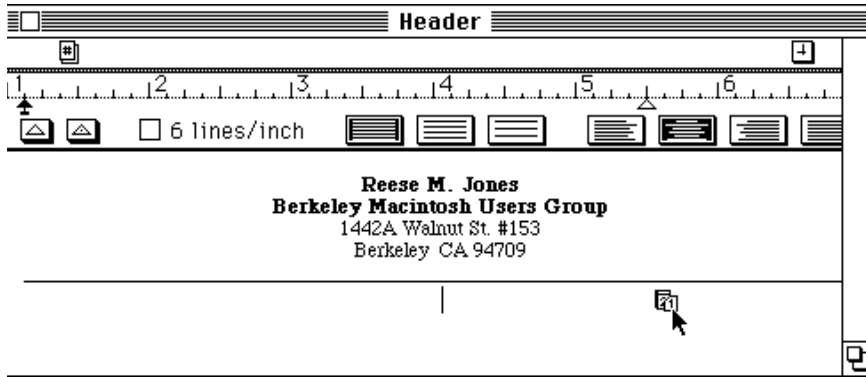


This is where you will enter your name, company name, and return address. The first character in the header (a space or a letter) determines the default font to be used for the date feature that we are about to use. You should select the font that will be used when printing the date. Usually this will be the same font as the rest of the letter. If you are printing on an ImageWriter, try Geneva or NewYork. If your letter is to be printed on a LaserWriter, you should make the first header character a laser font (e.g. Times). The default font for the system is usually Geneva (the LaserWriter automatically substitutes Helvetica for Geneva if you have selected *font substitution* in the **Page Setup** dialog box).

Select the *Center Adjust* option on the ruler in the **header** window. Type in one space and one return. Now type in your name, company name, address and whatever other information you might want to include in your letterhead (like phone number or account number) use Return to enter each line. You may want to include some additional formatting in your letterhead such as putting your name and/or company name in bold face and a separation line between the letterhead and the body of your letter.

We will make the name and company name bold face here and add a bold line below the address. Note the center alignment icon is selected in the ruler. Be careful when you make your name bold face that you don't also make the first space in the header bold (the first space is typed in just before the R in the example here). The bold separation line is made by making 3 carriage returns after the address, clicking on the second one down and typing a few long dashes. Then, select the whole line and choose bold format. Then click in the middle of the dashed line and hold down the shift/dash keys until the dashes appear on the line below, then backspace until the dashes are only on the one line. Finally click on the date icon just above the center of the ruler and drag the icon down to just below the dashed line and slightly to the right. Your header should look something like the one following.

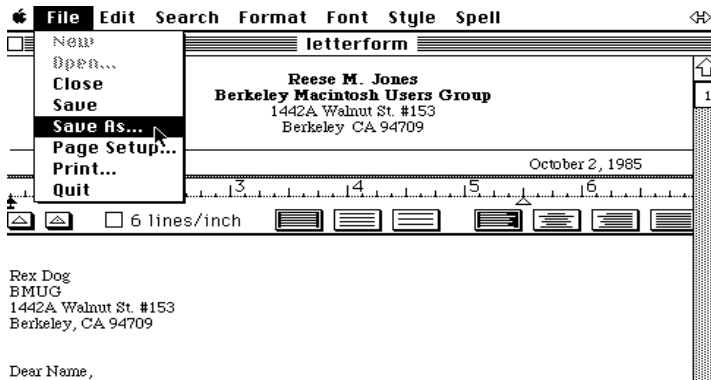
Fall 1985 BMU G Newsletter



This header will be printed at the top of the your letter (if your letter is longer than one page the header will appear on each page in which case you would want to put your address in the letter and not use a header). Each time you print the letter the date will be printed where you placed the date icon. MacWrite gets the current date from the system clock; you can set the system clock in the control panel desk accessory. The date will be printed in the same font as the first character in the header (in this case a space typed on the line above the name).

Letter layout and rulers

Next you want to set up the format for the recipient of the letter. Since this is a form letter we will use a generic addressee. Open the main window for your letter by clicking below the header window. You should see your header at the top of the page including the current date. You should add two more returns, choose **Insert Ruler** from under the format menu, and then type in a generic addressee as shown in the next figure. After the addressee, add two more returns and a generic salutation (Dear Name), followed by a return and another **Insert Ruler**. Now save a copy of your work to this point as the file *letterform*. Your letter should look something like the following figure.



The text of your letter

The text of your letters obviously will vary from one letter to the next but often you will have a couple types of letters where the basic content does not change much. With your *formletter* you can enter the general text of a letter that you find your self writing often, such as *my paper is late because my Mac ate the file* and other common letters. A good book, that has lots of examples of letters written for various life occurrences is: Lassar A. Blumenthal, *The Art of Letter Writing: The Guide for Writing More Effective Letters for all Occasions* Grosset & Dunlap Inc. NewYork 1977.

Enter the text of your letter below the ruler that you inserted under the "Dear Name." If you set up the ruler as shown below with the left indent marker mover at 1.5 inches, the first line of each of your paragraphs will be automatically indented one half inch. The line spacing will be one-&-a-half and the paragraphs will be left justified with a ragged right margin. Finish the main text of the letter with a return and another inserted ruler. You can change the font of your whole letter at one time by clicking at the beginning of the page then using the slide bar to move down the page then hold down the shift key and click at the bottom of the page. This is a general procedure for selecting a range on the Mac: *click, move, shift-click* will select every thing between the two clicks. This procedure is preferred over dragging to select a range when the range to be selected is large.

Fall 1985 BMU G Newsletter

Rex Dog
BMUG
1442A Walnut St. #153
Berkeley, CA 94709

Dear Name,



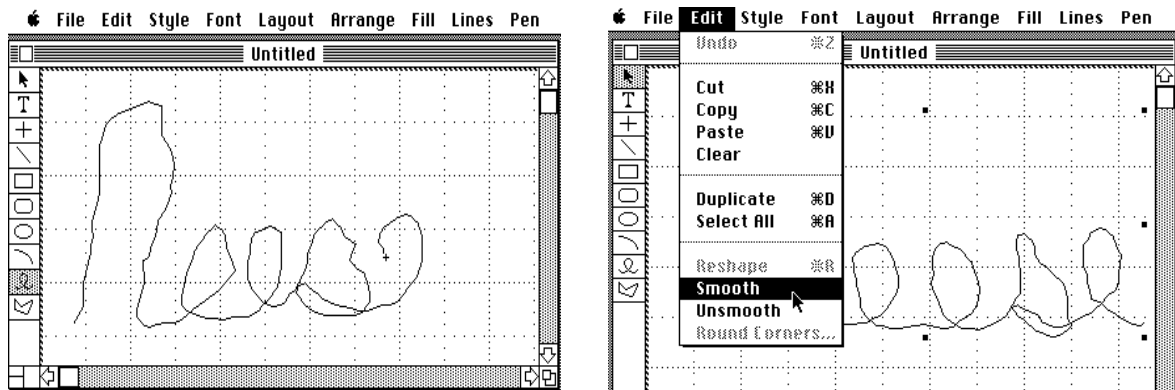
Thanks for your *nice* letters that we have been receiving over the last months. I'd been meaning to get back to you sooner but it was too much trouble to type each letter from scratch each time so I never got around to actually replying to your letters.

Adding your Signature

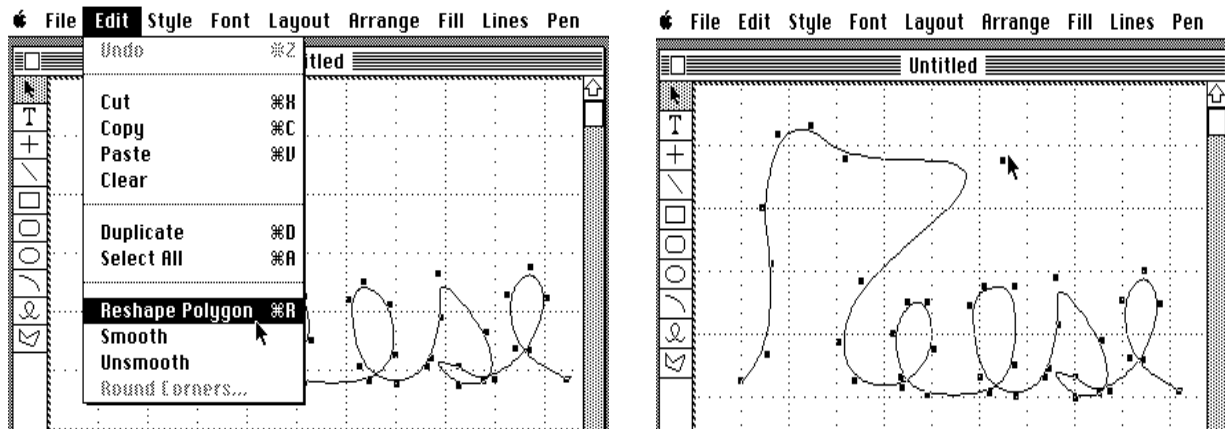
One fun touch is to let your Mac sign your letters for you. This works best if you are printing on a Laserwriter but can be used with the Imagewriter I or II. You will create your signature as a large MacDraw object and shrink it to paste into your letter.

Making your Signature in MacDraw

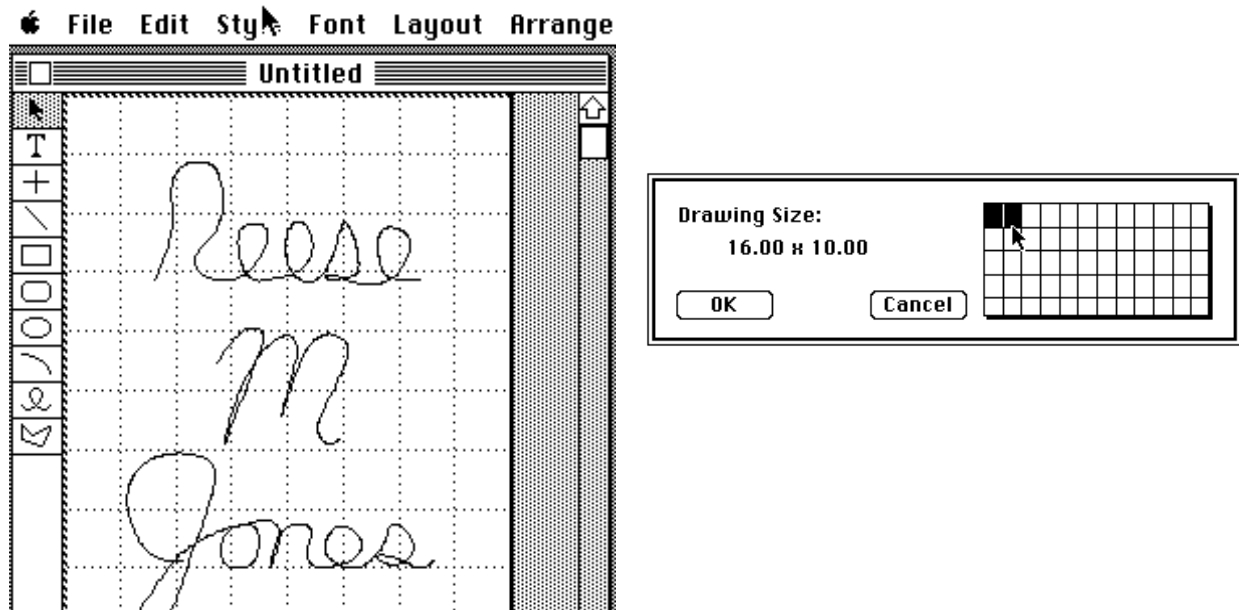
Open a new file in MacDraw and choose the line tool to draw the first part of your signature.



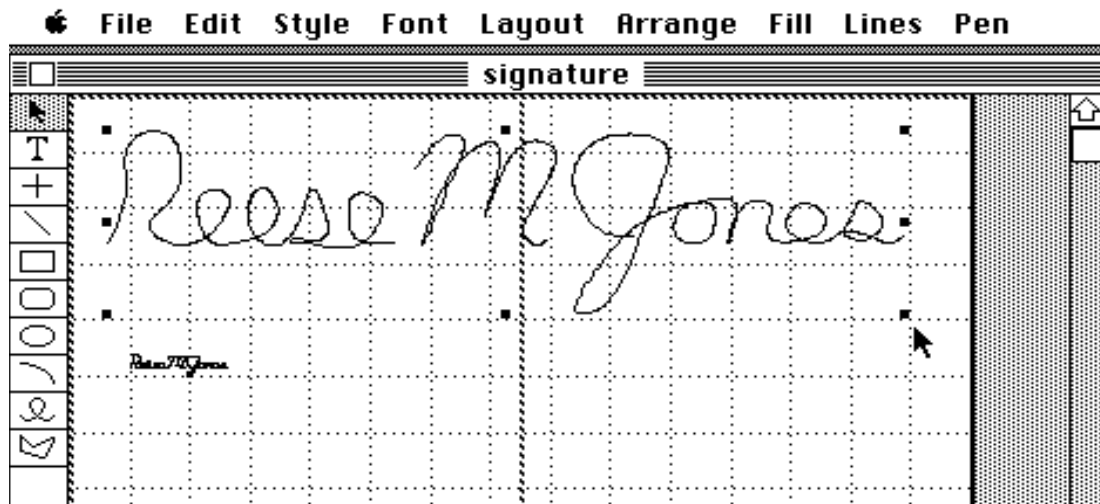
After drawing the first part of your signature with the line tool (it must be one continuous line segment) choose **Smooth** from under the **Edit** menu then choose the **Reshape Polygon** command.



You can now adjust the subtle shapes of the letters to more closely match your actual signature. It helps to sign on a piece of paper so that you have a reference. You can adjust the shapes if you click on one of the many object marker squares along the line, hold the mouse button down and drag. Give a little time for the computer to catch up with your change. It is worth spending some extra time at this point to get the smoothing right. You may need to re-sign, smooth, and reshape several times before you get a nice, realistic looking signature (especially if you are using a head mouse). Finish signing smoothing and reshaping each part of your signature as a separate object on the large scale. Then choose **Drawing Size**, under the **Format** menu to re-size the document to be two pages wide.



Choose **Reduce to Fit** under the **Layout** menu to get both pages onto the screen at the same time. Now select and drag each object so that your signature lines up. Depending on your signature you might find it useful to line the parts up roughly, and then use the **Align Objects...** command under the **Arrange** menu. Select all the objects by dragging a box around all of them at once (when you let go of the mouse: each object should have a set of select boxes around it) then choose **Align Objects** and select the most reasonable alignment criterion (align tops works best for this signature).



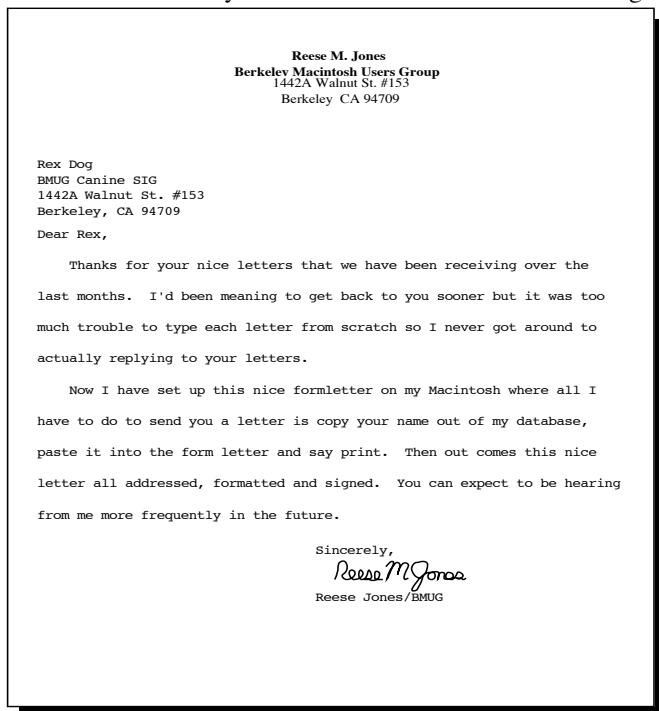
Now, while all of the objects are still selected, choose **Group Objects** from under the **Arrange** menu and you should get the eight selection boxes around your whole signature. Save this MacDraw file as *Big Signature*. Finally to shrink

Fall 1985 BMUG G Newsletter

your signature down to normal size on the page grab the lower right selection box and drag it towards the upper left corner box. Shrink it until the size looks right relative to the page break makers in MacDraw. You might want to save this file as *Shrunk Signature* and print one out to see whether you want to make any adjustments.

Finally, select the shrunken signature and copy it on to the clipboard. Next open your Scrapbook desk accessory and paste in your signature (remember that your signature is now stored with the Scrapbook Icon in the System Folder on the disk you have just been working from. You might want to copy this Scrapbook file from your MacDraw folder over to write disk). You may be surprized to see your signature sign when you open your scrapbook, and your friends will certianly be amazed. This is because your signature is remembered by the Mac as a MacDraw object two pages big, that has been shrunk down to be 1 by 4 cm. Thus the computer remembers the resolution of the larger object so that the printed object will look much nicer than an actual-size signature created in MacPaint.

Now all you need to do is paste your signature into the appropriate place in your letter. When you paste the graphic into MacWrite it will initially end up left aligned. To move it over to the right, select the object by clicking on it and releasing, then click on the middle of the right side of the box surrounding the object and drag the object to the right. The overall result of your letter form should look something like the following when printed:



Preparing a Letterhead with pictures using MacPaint and MacDraw

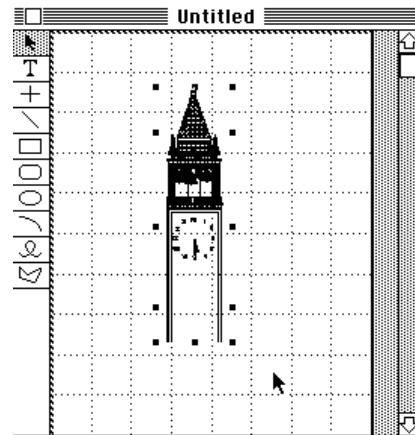
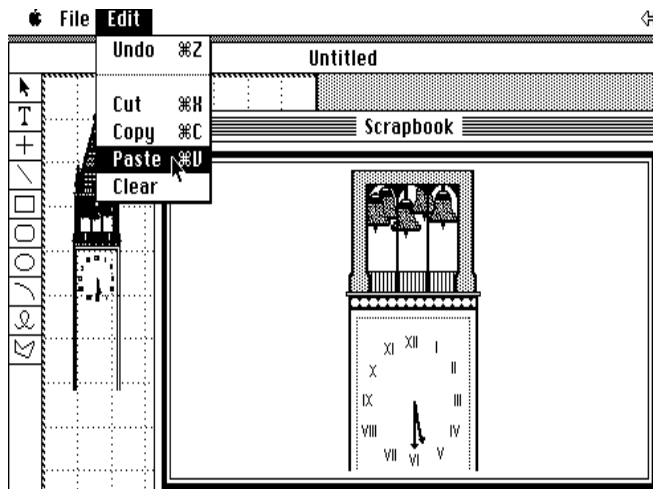
When you want to include a picture in your letterhead you usually want the picture side by side with the text. In MacWrite you can't paste pictures on the same lines as text. So to get a picture like your company logo to the left of your address you need to create a picture to paste into MacWrite that has both the graphic and the text in the same picture. You could make your letterhead figure in MacPaint with the text and logo side by side then copy the picture into the clipboard (and/or scrapbook) and paste it into the header of your letterform. You can copy a MacPaint picture that is larger or wider than the normal MacPaint window by using special programs and paste as a single object into MacWrite. The *Page Mover* program (on BMUG disk #23), the *Paint Grabber* desk accessory (*Art Thief* DA from Macromind, comes with *Videoworks*), the *Thunderscanner* software and the *Paint Cutter* software from Silicon Beach are all programs that can be used to select a paint object larger than the screen.

The disadvantage of preparing both your text and logo/graphic in MacPaint is that the whole figure including the text is stored and processed as a bit map. So it is more difficult to go back and change the text or font style of your address

without re-typing it or rearrange the positions of the objects in your header. In addition if you are using a LaserWriter and have a bit mapped font in your header, you will not be taking advantage of the special high resolution laser fonts.

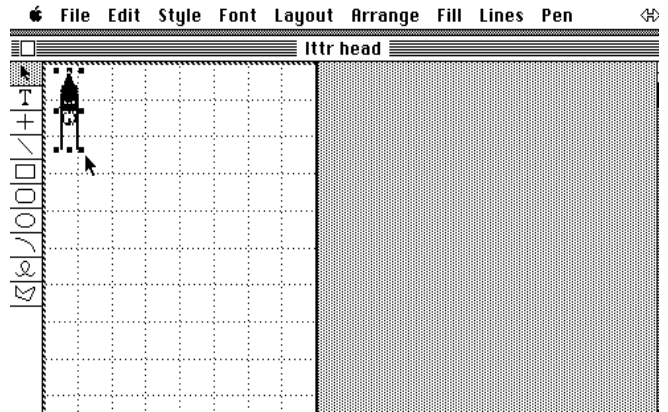
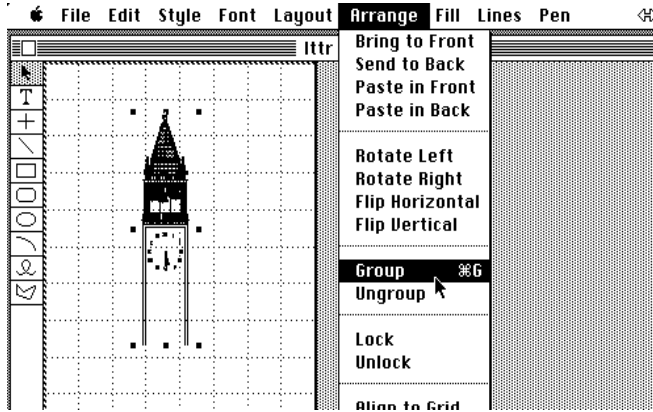
An alternative approach is to use MacDraw to prepare your letterhead with the your logo/graphic and using the fonts for your address put together in the MacDraw. You can create your logo completely in MacDraw, this will have the advantage of being easily scaled is size and, because it is all objects, will be printed at the best resolution of the printer (ImageWriter, LaserWriter, or Postscript-driven PhotoTypesetter like the Allied linotype) where as a bit mapped image will be printed as a construction of nice sharp squares. But creating a nice logo in MacDraw is often more difficult than in MacPaint or by using a digitizer. The solution is to use the combination of both bit mapped objects and MacDraw objects in your logo. The problem of bitmaps not taking advantage of the laser printers resolution can be overcome by treating the bitmap as a very large object, with fine detail, and use the computer to figure out how to scale it down to the final size.

In the following example we will incorporate a bitmapped graphic, a MacDraw object, and some some text in LaserWriter fonts into one figure that can be pasted into MacWrite for a fancy letter head. We will use the great graphic of the campanile on the Berkeley campus that was created completely in MacPaint by David de la Vega for the cover of the BMUG spring newsletter. In the original MacPaint picture the tower is the full 11 inch height of the page with lots of intricate detail, but we want to shrink it down to be only 2 inches high while preserving all the fine detail. This is similar to the approach we would take with a digitized image created with MacVision through a video camera or with the Thunderscanner. We would want digitize the image to be as large as possible (full screen or full page), clean it up in paint with fatbits, and finally reduce the desired size in MacDraw. Using any of the aforementioned cutting programs such as *Page Mover*, the object is selected, copied to the clipboard, pasted into the scrapbook, then pasted into MacDraw.

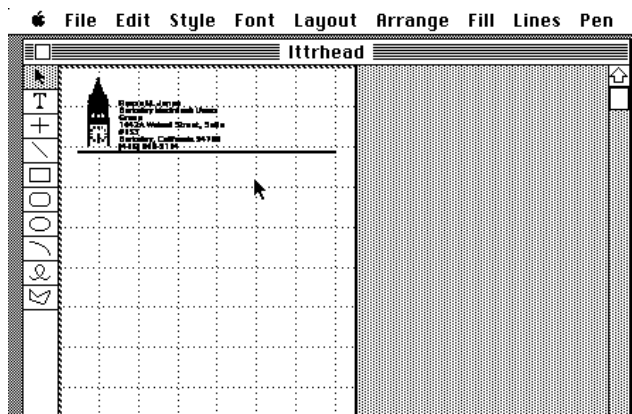
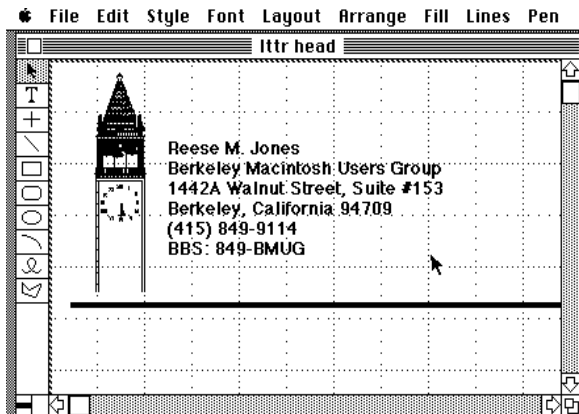
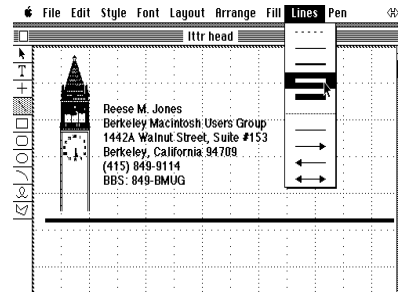
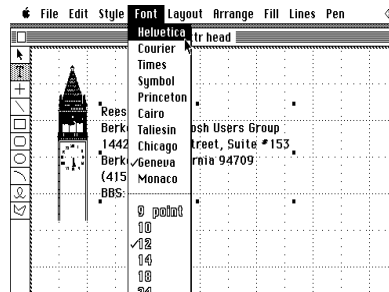
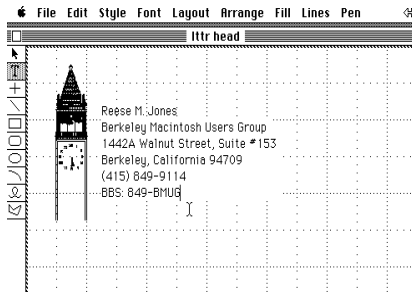


Note how the object has several selection indicator boxes. This is because MacDraw is thinking of the picture as several different objects rather than one. (This frequently happens when you paste large bit map pictures into MacDraw or when pasting from older versions of MacPaint). **Immediately** after pasting, choose **group** (*command G*) from under the arrange menu. This will tell MacDraw that you want to treat this picture as a single object. Grab the lower right selection indicator box and drag it to shrink the size of the object on the page to the relative size that you want for your letterhead (just as you did with your signature above).

Fall 1985 BMU G Newsletter

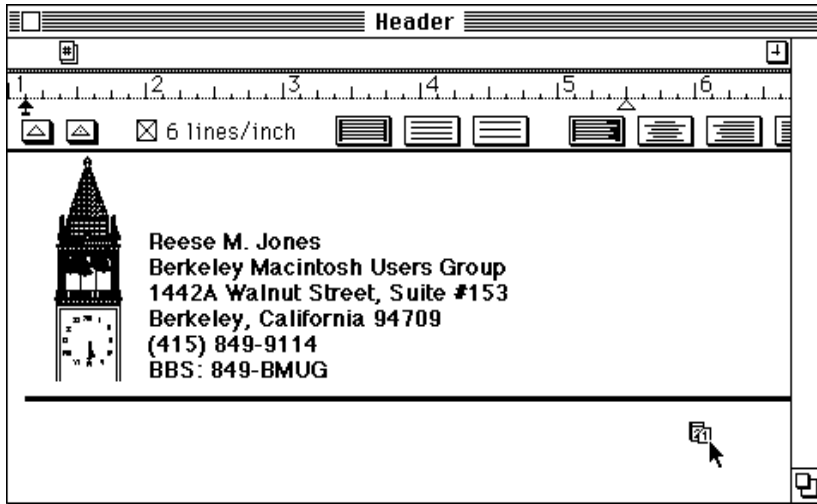


Now select the A (text entry tool) in MacDraw and type in your name and address to the right of the picture. With the text selected choose the font and style for your address; be sure that you choose a LaserWriter font (Times, Helvetica, etc.) if you want to print your letters on a LaserWriter. If you want your name to be in one style and font and you want your address to be in a different font or style you need to enter them as separate pieces of text. You may also want to add other embellishments to your letter head like a line below the text to separate the letterhead from the text of the letter. The line here was drawn all the way across the page in the **Reduce to Fit** size and the line was selected to be bolder.



The final result of your letterhead in MacDraw will look something like the one above. Now save your MacDraw document, select all with *command-A* and choose **copy**, quit MacDraw and open MacWrite; or, if you are running MacWrite and MacDraw in the switcher, move over to MacWrite while converting the clipboard. Open the header in MacWrite and paste in the picture. Position the date icon to the position that the date is to be printed on the header.

Fall 1985 BMU G Newsletter



Your final letter with the letterhead, graphic/logo, laserfonts, and signature should end up looking something like the following:



Reese M. Jones
Berkeley Macintosh Users Group
1442A Walnut Street, Suite #153
Berkeley, California 94709
(415) 849-9114
BBS: 849-BMUG

October 10, 1985

Rex Dog
BMUG Canine SIG
1442A Walnut St. #153
Berkeley, CA 94709

Dear Rex,

Thanks for your nice letters that we have been receiving over the last months. I'd been meaning to get back to you sooner but it was too much trouble to type each letter from scratch so I never got around to actually replying to your letters.

Now I have set up this nice form letter on my Macintosh where all I have to do to send you a letter is copy your name out of my database, paste it into the form letter and say print. Then out comes this nice letter all addressed, formatted and signed. You can expect to be hearing from me more frequently in the future.

Sincerely,

Reese Jones/BMUG

MACsimizing MacDraw

© 1985 by Dan Wood

Introduction

The following article is *not* a tutorial or manual for MacDraw, nor is it a "Gee, isn't this program woonderful" article. Rather, it is a collection of tips that I have found that make MacDraw easier and faster to use.

Use of the *command* and *enter* keys

One way to optimize MacDraw is to make full use of the *command* (cloverleaf) and *enter* keys. If the arrow pointer is selected, holding down the *command* key will return MacDraw to the previous drawing tool as soon as the mouse button is depressed. In other words, you can continue using a drawing tool by holding down the *command* key and proceeding as if that drawing tool were selected.

To use a tool several times in a row, just hold down the *command* key as you repeatedly draw with that tool. When you release the key, all of the objects you have drawn are selected, and the arrow pointer is selected.

Pressing *enter* will return the tool to the arrow pointer, with the object you were working on now selected. This is most useful when you wish to select the text box as a whole after entering text. Then, to return to the I-beam cursor, hold down the *command* key and click on the insertion point.

Text fields

In MacDraw there are two kinds of text fields: *caption* and *paragraph*. Caption text is the easiest to produce, but it is the least flexible. To enter headline text, select the T tool, click an insertion point with the I-beam cursor, and start typing. There is no wraparound in caption text: new lines are entered with the *return* key. If you are typing several lines and you wish to change the size/shape of the field, you will have to do so manually by removing and inserting carriage returns.

Paragraph text is entered in a less obvious fashion. To enter paragraph text, create or select some object (such as a rectangle); then just start typing. The text will be entered starting at the top of the object's rectangle bounds, and automatically wrap around to stay within these bounds. As you continue typing, the text may extend beyond the bottom of the object. If you select the text field as a whole (by pressing *enter* or by selecting the text when the arrow pointer is active) you can *reshape* the text box by dragging the lower-right corner square. The text will word-wrap to conform to the new proportions.

There are actually two formats in which to cut or copy text from MacDraw. One is by selecting the entire text box and cutting. The other way is to select actual text with the I-beam and cut that. Dragging and double-clicking works the same way as in MacWrite. Triple-clicking in a text selection with the I-beam will select all the text in the field. Text is pasted from the clipboard as caption text by default. Of course, this is a major problem when the text is from MacWrite, the notepad, etc., where most lines are not separated by carriage returns. To paste as paragraph text, select an object and type a couple of characters to prepare MacDraw for paragraph text. The text being pasted should be text only (by selecting with the I-beam from MacDraw rather than by cutting a selected text field). Then, delete the first characters you typed at the beginning. By using this procedure it is possible to convert between caption and paragraph text: with the I-beam, triple-click on the text field; cut; select a rectangle, type a couple of letters; paste; then delete the first characters.

By using paragraph text, you have the capabilities of a primitive word processor. The drawbacks are the lack of full justification and no mixing of type sizes, styles, or fonts within a text selection. You can always kludge mixed styles by making another field with the different style, filling the text field with white (or whatever background pattern the other field uses) and sticking it over the other field. It's not too elegant, but it works. Watch out for proper alignment!

It is a good idea to be sure that the **Page Setup...** is set to “Tall Adjusted” (the default) if you are printing on the Imagewriter. When “Tall” is used, text will be reformatted to appear narrower than it is on the screen, so what you see is not what you get. Using “Tall Adjusted” is essential especially when using the technique for mixed styles as mentioned above.

If you switch between different system disks on your MacDraw documents, you may notice that the fonts sometimes change as if by magic. The reason is that MacDraw only allows eleven fonts to be used. The font menu is restricted to showing eleven fonts; any fonts in the document that are not one of these eleven will be automatically converted to the topmost font in that list. Geneva is the default font, but if it is beyond the bottom of the list (ordered in the reverse order of installation), the default will be the topmost font. If you are not planning on using Geneva in MacDraw, just install the font you want to be the default *last*.

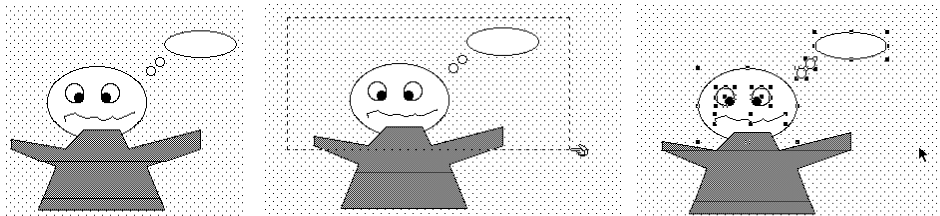
Using the shift key

As in MacPaint, the shift key provides a number of constraining features. Holding down the shift key while creating objects produces the following results:

- Ovals are constrained to circles
- Rectangles and rounded rectangles are constrained to squares and rounded squares
- Arcs are constrained to quarter-circles.
- Lines are constrained to 45-degree increments
- Letters are constrained to upper case (Groan! Blame Bill Atkinson for this one)

As in most other applications, the shift key is used for multiple selections. This is indispensable for selecting groups of objects to reshape, group, align, etc. In text, you use the shift key to extend a text selection just as in MacWrite.

Note that holding down the shift key and pressing the mouse button while in selection mode turns the arrow into a selection “hand,” even if you are clicking on top of an object. This is extremely useful when you wish to select (draw a box around) a group of objects that lie on top of another object. Ordinarily you would end up selecting and dragging the larger object. In the included example, we wish to select the face and thought-bubbles so that we can bring them to the front. To select the small objects, do the following:



Selecting on top of a large object

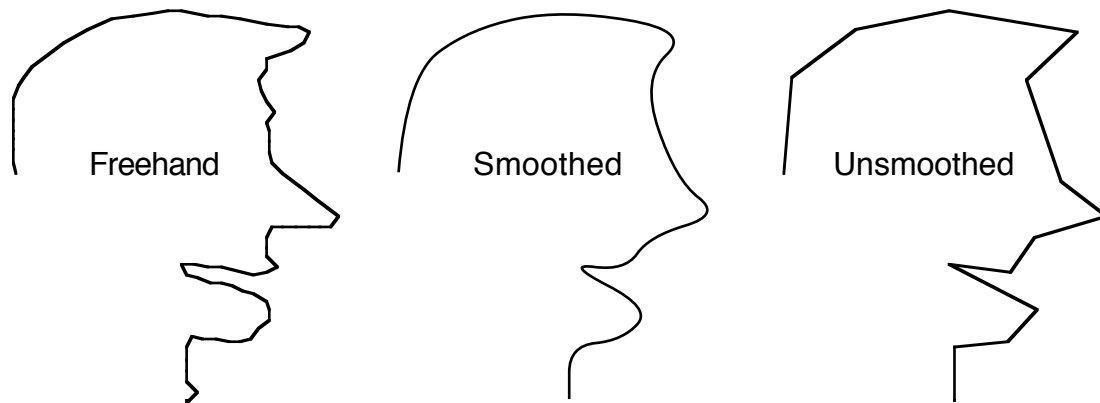
- Be sure that nothing is currently selected in your document (left figure). This is important because you will be using the shift key to “extend” the selection. Unless you do wish to extend your current selection, click on an empty area in the document so that no objects are selected.
- Hold down the shift key. The arrow pointer will turn into a hand.
- Draw a box around the objects (center). They should now be selected with the object they lie on intact (right).

Polygons and Freehand shapes

Polygons and freehand objects allow MacDraw to be used for much more than just diagrams, charts, and layouts. With the underrated **Smooth** function and reshaping capability, you can create any shape or blob imaginable.

The freehand shape tool may not seem to be flexible at first. One mistake, and it's all wrong. However, the object can be reshaped by choosing **Smooth**, or **Unsmooth**. A freehand shape that is smoothed or unsmoothed will look different

than the original shape;. There will be no way to return to the original shape because smoothing or unsmoothing converts it into a polygon.



A freehand shape and its permutations

Smoothed polygons are polygons with curves drawn rather than corners. To create cusps (pointed corners), place two reshaping handles of the polygon in the same place. (Two handles in the same position are not visible since the squares cancel each other out.) No smoothing is visible if two handles are in the same place.

Clicking on a point where a corner already exists will close up the polygon and finish the shape. To keep the polygon active, hold down the option key before clicking. Use the option key if you want to place two polygon corners in the same place.

Reshaping is used on smoothed or unsmoothed polygons and also on arcs. After you begin reshaping, the object will be drawn with no fill so you can see through it. You can drag the corners around arbitrarily while reshaping a polygon. If you grab onto part of the object that is *not* a corner, you will drag the whole shape. Watch out for this, since you may want to undo the move.

Setting MacDraw defaults

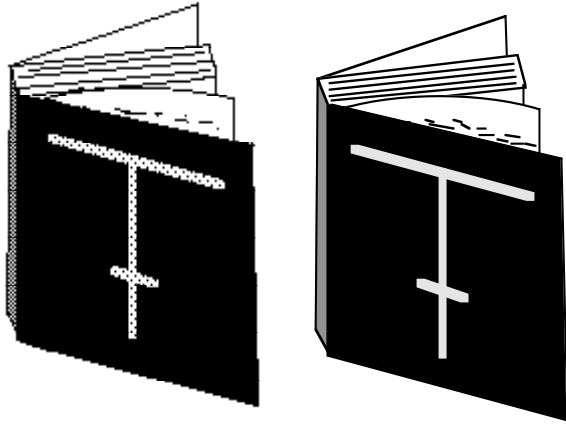
To select defaults for text, just choose the desired font, size, style, etc., when nothing is selected. To select defaults for fill pattern, pen width, arrows, or pen patterns, select a tool such as rectangle, then select the desired defaults in the Fill, Lines, and Pen menus. The lower-left corner of the window will show the defaults you have selected. Note: arcs and polygons are always drawn initially as unfilled, regardless of the fill pattern default.

You may wish to save a template document with these defaults and other program settings (show/hide size, page breaks, grids, rulers, page setup, etc.).

Bitmaps

MacDraw allows bitmaps (images from MacPaint) to be pasted onto its documents. When the bit image are produced by some versions of MacPaint and the Thunderscan Software, MacDraw will paste a set of horizontal slices of the picture. After pasting, therefore, immediately choose **Group** so that the image can be worked with in its entirety. If the image gets taken apart, you cannot put it back together again perfectly (especially noticeable when using the LaserWriter).

With the bitmap facility, it is possible to “convert” a MacPaint picture to a MacDraw picture. Just draw the MacDraw picture on top of the bit image (enlarging first if desired), then delete the bit image underneath.

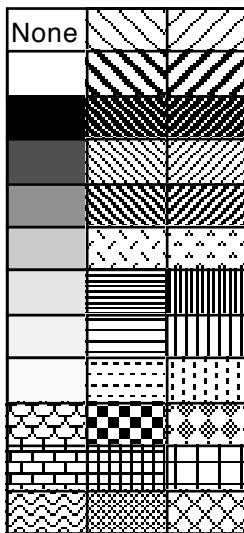


Original MacPaint picture (left); conversion to MacDraw (right)

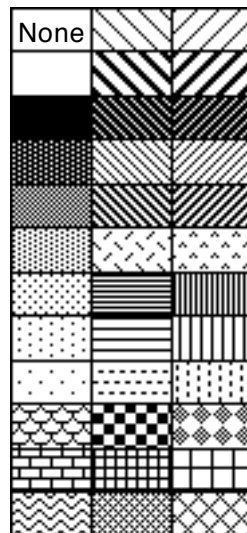
Using the LaserWriter

Since the LaserWriter has much better resolution than the ImageWriter, you may be in for some surprises when you print your documents on the LaserWriter for the first time. Many objects, although they look fine on the ImageWriter, will look different when laser-printed.

- One nice feature is the true halftoning that the Laserwriter produces with the first few patterns down the left side of the Fill and Pen menus. Rather than printing the sparse dots you see on the screen, the LaserWriter prints corresponding shades of gray. See the book pictures and the pattern examples.
- On some MacDraw documents or systems, the LaserWriter prints patterns rotated 90 degrees.



LaserWriter patterns



Bit image patterns

Patterns on the LaserWriter vs. the screen

- Patterns stay the same size regardless of reduction.
- Patterns are never smoothed (unless it is a bit image of a pattern and “Smoothing?” is checked in the page setup).
- You can get the effect of a tapering line by covering a line up with another object (such as a white polygon) at a gradual angle.
- Unfortunately, the single-thickness lines in MacDraw come out pretty thick on the LaserWriter. They're probably equivalent to one pixel wide, but it would be nice to have a bit more control.

- When printing shadowed or outlined text, select the menu options with the command-key equivalents, or hold down the *command* key when selecting from the menu. This will insure correct spacing of the letters.

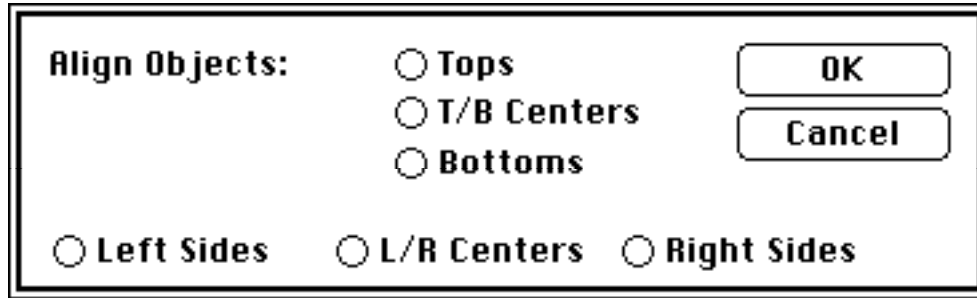
Miscellaneous tips

- To adjust the scroll bars so that the selected objects appear in the center of the screen, choose **Enlarge** then **Reduce**, or **Reduce** then **Enlarge**.
- When a window first opens on the screen, a *lot* of space is wasted. You can see another vertical inch of your document by dragging the window up to the top (slipping it partway below the menu bar) and by resizing the window as far down as it will go.
- To zoom in on a particular part of a large object, draw a small object on top of the part you wish to examine, enlarge it, then hit *backspace* to delete the object. Your screen will now display the area you wish to work with.
- By using the **Show Size** option with large objects (when you cannot see the borders on the screen) you can often tell if you have selected the correct object by looking at the measurements as you click on it. For instance, if you have a 48-inch long line and a 20-inch by 10-inch rectangle, you cannot tell which object is selected if you are looking at the center in normal size because the selection rectangles are not displayed. If you try to select the line, you can verify that you actually did select it by noticing the measurements. If it shows 48", you've selected the line; if it shows 20" by 10", you've selected the box by mistake.
- Use grouping and locking extensively. Remember that grouping items together makes one object. That item can in turn be grouped into other items, and so on. When items are in the correct relative position, group them together so that they aren't disrupted. When items are in the correct place, lock them. When locked items are grouped with other objects, they are unlocked. When items are grouped, the group is placed on the stacking order of the frontmost object in that group.
- Rectangles can be converted to rounded rectangles and vice-versa. Select the rectangle and set a rounding measurement with **Round Corners...**
- Draw objects bigger than they will end up being, then shrink them down when they are correct. This is vital (especially when using the LaserWriter) to be sure objects are properly aligned and shaped. If the object is double size, you can even scale it using custom rulers (i.e. by displaying 2-inch increments marked as 1-inch).
- To shrink or expand objects while keeping the proportions intact, draw an unfilled square around them. Select the square and everything inside of it, then (using the grid and **Show Size**) enlarge or shrink the group of objects so that the square remains a square. Then remove the square.
- Although MacDraw works on a 128K Mac, it's slow and very limited, especially when using memory hogs like bitmaps and polygons. The program takes up so much memory, there's not much left for the document. So if you don't have one, go get a 512K Mac!
- If an alert box saying "The print command was not completed" appears when you try to print, you probably don't have enough space on your disk, or you may have to use **Choose Printer**.

Customizing MacDraw

The most important technique for improving your performance on MacDraw is to alter the program. *Don't Panic!* All this means is to change some of the Menus and dialog boxes to help speed things up. Using the Resource Editor and/or Menu Editor and/or REdit (all of these available on BMUG disks), it is simple to customize MacDraw's menus and dialogs. I have added so many keyboard equivalents for MacDraw that every letter key has some function when used with the *command* key. Additionally, many keys have related or opposite functions assigned to *command* WITH option. The MENU, DITL and DLOG resources which I have changed are available on BMUG Disk # 20); they can be easily installed using ResEdit. Of course, this does not mean that MacDraw "should" have these keyboard equivalents and other changes; I've found them, however, to be the most convenient.

The only dialog box which I have found necessary to change was the **Align Objects** display. The original was not easy to read; this rearrangement (I used REdit) makes it easier because it is graphically oriented. In combination with a keyboard command, what was painful is now a pleasure.



New dialog box for Align Objects...

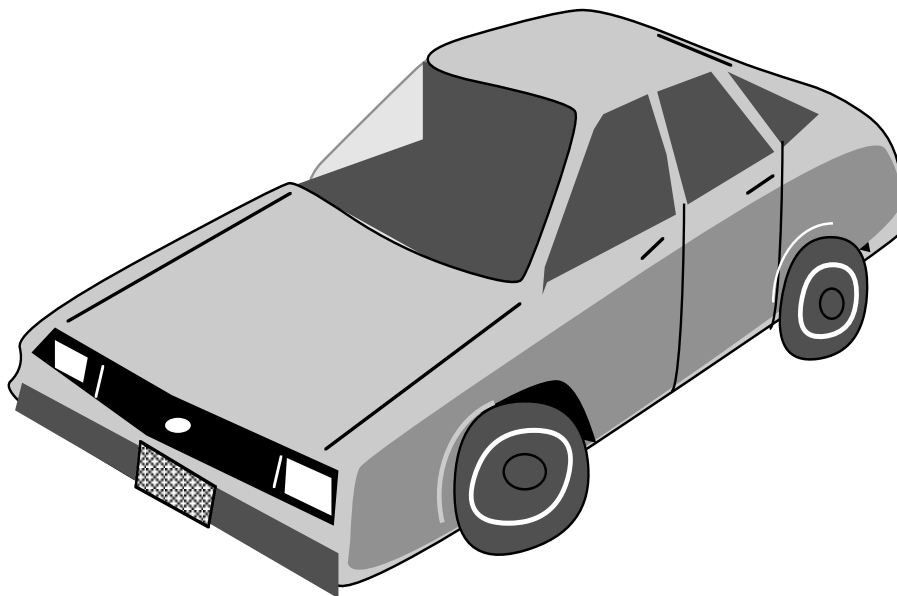
The menus make extensive use of option-commands. Keyboard equivalents for menu commands can be accessed not only with command-character; they also work with command-option-character and even command-option-shift-character! Looking at the menu lists does not make it easy to recognize, since the letters represented are often a foreign symbol or a box. However, the option sequence always shows up in my menus with its ordinary counterpart for clarity. For instance, **Bring to Front** is command-f, **Send to Back** is command-option-f. Occasionally the command-option sequences do not work, probably due to a bug in MacDraw.

The patterns in MacDraw are stored in a PAT# resource in the MacDraw program. You can customize your own patterns by using a recent copy of ResEdit (older versions often crash when editing a PAT#).

Finale

These discoveries, mostly my own, have made drawing much easier for me. The keyboard equivalents are the most valuable asset. The tricks and shortcuts, some discovered by accident, are the result of constant use of the program. I hope this article will inspire others to streamline their copies of MacDraw and other programs as well. It's so easy to do!

Dan Wood is a student at Cal, a freelance consultant, graphic artist, and programmer for the Macintosh, and is involved in a myriad of activities too numerous to remember.



Dan Wood

Using MockWrite

by Gordon Haig

Just for fun I decided to write something on multi-tasking with the desk accessories. I have a 512K Mac; I just got the Switcher. I suppose this technique will be less useful to me after I get used to it, but I've been switching back and forth amongst types of applications for some time, and I am loathe to give up my accustomed method. For quick access to a file, to write a letter rather than a disquisition, or for those who have a 128K Mac, desk accessories are the way to go. I use MockTerminal, MockPrinter, Delete File, and most especially, MockWrite. Using MockWrite I can have two windows open at once - or two files, or two types of applications, or two parts of one file that I'm working on.

If you have tried to use MockWrite, you may have been dismayed to find out it can't open MacWrite documents. I certainly was, and after some initial fooling around, I gave it up as one more weird puzzle supplied by BMUG. Sometimes attempting to puzzle things out from a user-supported disk makes me feel like an archaeologist holding half of a pot, trying to figure out what the rest was like—and is there a handle?

However, I left MockWrite on the disk figuring, somewhat grudgingly, that I might want to use it as a kind of variant notepad. (Anyway, with the kind of mastery of the desk accessory mover that I had at that stage, I considered I was lucky to have gotten it installed at all. The new one is **much** easier). Because of this I discovered that it could open the files that I fed in from my Radio Shack Model 100 laptop portable, which I use as a kind of scratch pad. Since most of my files start out as scritch-scratchings made on the train or in the coffee house on the Model 100, I soon had a set of current files that could be opened by MockWrite.

You can get the same kind of formatting and make your MacWrite files openable from MockWrite. There is a text only option in the "Radio Buttons" at the bottom of the **Save As...** dialogue box. Choose **Save As**, click on **Text only**, and click on the **Save** button. On a working disk, Text-only format is more economical of disk space. One document I tried took up 3K as a regular MacWrite document, 2K as text only. Another example is a set of notes with no special format: 11K ordinarily, 6K as text only. Another advantage of text only files is that the Mockprinter desk accessory will enable you to make a quick printed copy of them while you continue to work on something else on your Mac. I like to move to my file listings and see what I'm going to do next.


However, saving text-only documents will lose the formatting information you had in the original; for example, if you reopen it in MacWrite, you will have to reinsert the tabs. A document with extensive tabs, tables, font changes, and multiple different rulers *cannot* be saved as text only. Also, if you reopen the document in MacWrite, adding font or tab formatting, and want to save it, you must choose **Save as** once more; the **text only** file you start from doesn't save the fonts or even ask you if you want to. I mention these things for completeness, but for myself there hasn't been much of a problem. If you are in the habit of just selecting **New** and starting to pound the keys you'll never even notice the difference. Even though saving my notes as text-only documents reduced them from 11K to six, there was no difference when I reopened them in MacWrite. I am now getting into the habit of routinely saving rough notes or a working draft as text-only files. If you do this the first time you save you can use MockWrite rather than MacWrite, where it opens it as "Untitled" every time. When I find my file starting to evolve towards a finished document, I save by using **Save as...** and *not* clicking on **Text only**.

I find this system particularly convenient when I want to refer to the notes in one file while working in another. MockWrite is ideal for this. It opens faster than MacWrite. It uses a very legible font, single-spaced nine point Monaco. This is very convenient for consultation: it is easy to see at a glance a decent sized chunk of text. If you want to look back and forth to the underlying window, MockWrite's wordwrap follows the dimensions of its window, so you can resize the window without losing the end of every line.

Have an off-beat inspiration? One related to your present work by a path only a squirrel could follow? If I have an brainstorm, I just jot it in down in Mockwrite, confident I can pull it back anywhere: in another file, in another application, or with MacWrite proper. Since MockWrite creates a regular text-only file, rather than using a special

notepad file in the System, on a two-drive machine you can open your notes even if you change the startup disk. This is much better than the notebook: I removed it from my disk after I got used to MockWrite.

Want to cut and paste? Just open one file in MacWrite, another in MockWrite and you can have them both on the screen at the same time. You choose the phrase or paragraph, that you want to transfer while looking right at the text you are choosing it for. You cut, click on the destination window, then paste in the usual manner. To repeat the process, just click on the first window to make it active again. If you didn't have MockWrite and wanted to paste several different selections you'd have to cut, then paste into the scrapbook, each piece in turn. Then you'd close the file you were in, open the new one and start cutting and pasting out of the scrapbook into the file. And, of course, while cutting out of one file you can't see the other. The really marvelous thing about doing cut and paste from Mockwrite is that you always get just the right phrases, because you're looking right at the passage they'll be in.

For quick glances at a file I open the MockWrite window, but when I've looked I don't use the close box, I just click in the MacWrite window to make it active. Select MockWrite from the  menu and it quickly pops up exactly the way I left it. For a little more extensive work you can shrink the MacWrite window vertically, and spread out across the bottom a short wide MockWrite window (like the "Show Clipboard" window). However, I think that's ugly, so when I'm planning a lot of back-and-forth work, I change the font in MacWrite to 10 point Geneva or 9 point Monaco, set the right margin about one-third of the way in towards the left and shrink the MacWrite window. That way I can have the two windows side by side, see just as much of the MacWrite text, with about the same line length and pagination .

The same approach will let you open windows on two different parts of the same document. Cutting and pasting with MockWrite, or making a few extra notes are so easy and wonderful that you may want to try this, but if you do, be careful. This is a use that isn't really allowed for in the way the applications work and if you get casual, you won't get a dialogue box; you'll get a disaster. If you make changes in the MacWrite window and save it, and then make changes in MockWrite, and save them, only the changes in made from MockWrite will end up on the disk. I save the file under two names, say 'File1' and 'File1.b', then open one in MacWrite, the other in MockWrite. This can get confusing, so start off with a playful spirit, and first back up your file on a separate disk.

I have used MockWrite mostly with MacWrite but there is no reason not to use it with other applications. It will open a MacPascal program as text, enabling you to take a quick look at how something was done in another program. If you want to consult a text while in MacPaint don't close, open, close, open: just open MockWrite. You can open the Rolofile from Bill Atkinson's public domain Rolodex application on BMUG disk #1. It will of course open files captured from MockTerminal, like the ones I uploaded from my model 100. For other applications, you are on your own.

I use 'Set Startup' to get a fast startup directly into MacWrite. I don't often go back to the desktop. If you work like this or start to use some of the techniques I've recommended, you'll find things can get pretty overgrown. With *Extras, Disk Info* (both on BMUG Disk #20), or *Delete File*, you can delete files without leaving MacWrite or any other application you are using. If you decide that version 4 is shaping up well and that you'll never go back to #2, delete it as you work. If you cut and paste all of a small file that you opened with MockWrite into your current opus, then get rid of the old file. If you ever get to really whaling and then after half an hour of inspiration, go to save only to get a "not enough Space on the disk" message, one of these desk accessories can be a life saver. I weed the garden with *Delete file*.

I'm a desk accessory fan. Desk accessories let me start anywhere and go anywhere, just as I do in thinking. All I need now is a Think tank desk accessory, and an Etch-a-sketch - and then a hard disk so I won't have to make those hard choices as to what to include in the system file.

Mac Programming Languages

By Linda Custer

with help from Larry Rosenstein, Brent Fultz, and Kirk Austin

You are interested in programming the Mac, for fun or profit. Which language should you choose? Originally, your choices were very narrow, but by now most popular languages are available. This article will help you sort through them and decide which are best for you. It is admittedly only an introduction into the products available, so ask your computer dealer and your friends for their opinions.

INTERPRETED VS. COMPILED

Most languages operate in one of two ways. They either interpret what you type in every time you run the program or they compile your program into code that the processor understands once and store that machine code to run again and again. Interpreted languages are often easier to use in developing a program because they will run until they find an error, politely tell you the nature of the error, allow you to fix it, and resume running. In many interpreted languages, you can examine the contents of your registers as the program executes and be confident that the program is operating the way you intended.

Compiled languages, on the other hand, are fast. Interpreted languages require that every symbol in your program be translated piece by piece each time it runs. Compiled languages translate your code only once. From then on, you run the compiler-produced code. Each time you find a bug in your code, you have to recompile. Once you remove all the bugs, though, the program will run much faster than interpreted programs.

Another major difference between interpreters and compilers is that interpreter-based languages cannot generate code which can run all by itself. The language disk, purchased from the manufacturer, must always be present. So you cannot use an interpreter-based language to write programs that can be distributed freely without expecting that people obtaining the program own the language it is written in. Of course, this also means that you must distribute your actual source code (the words and symbols you put down) so everyone can see how you wrote the program and can modify it. This is great if you want to have a program for hackers to take apart, but awful if you are trying to sell your program and keep the source code proprietary.

Most compilers allow you to produce a file which you can rename, give your own icon to, and distribute as a stand-alone application. The recipients will then only need the standard Macintosh system folder (system, finder, etc.) to run it. And your source code will remain forever a mystery.

Somewhere in-between interpreted and compiled languages are languages that use an incremental compiler. You enter in short pieces of code that do a particular thing, compile them, and test them before going further. Once you get to the end, you have built a whole compiled program in memory, but debugging does not require recompiling the whole program each time you make a change. Just recompile the short procedure. These languages can allow you to produce a stand-alone program, but often that requires some extra work on your part and on the part of the software company. Typically, you buy a developers' add-on to the language which takes you step-by-step through the procedure for removing their language from your program.

THE LANGUAGES

BASIC. BASIC (Beginners All-Purpose Symbolic Instruction Code) is probably the easiest language to learn. Almost every computer built can understand it. Two different versions, both interpreted only, are available: one from Apple called Macintosh BASIC, and one from Microsoft called MS BASIC. MS BASIC can run programs ported over from other computers. Neither, in their current versions, require line numbers any more, but both are fairly slow. Macintosh BASIC boasts an unusual degree of user-friendliness and is good for beginning programmers. Macintosh BASIC, however, may never be released. Apple was trying to decide whether just to dump it into the public domain to avoid the high costs of supporting it, to sell it to another software company to distribute and support, or to just forget it exists. It looks like it will never be released, due to legal problems and competition with Microsoft. Both BASICs, though,

allow access to windows, menus, and some graphics, but not to all toolbox routines. Since Microsoft already markets good, fast, compiled BASICS for other computers, maybe they'll eventually upgrade their Macintosh version too.

Pascal. Pascal is fast becoming the teaching language now that the College Board uses it almost exclusively on the Computer Science Achievement Test. It is a highly-structured language that attempts to make programming more error-free by catching errors as not making sense to the language. You have to define exactly what variables you will use and what values they can take on before you use them, a task not usually required in BASIC. Currently, one version from Apple and Think Technologies, called MacPascal, is available. It is an interpreter only, and is highly user-friendly. It supports graphics, windows, and menus, but not the whole toolbox. Good for beginners. It is rumored that a full stand-alone development system by Apple for the Mac (using all LisaPascal toolbox calls) will be available soon. There are also a few other Pascal compilers on the market which can be compiled directly into runtime code or into a somewhat more inefficient code called p-code (a Pascal system originally developed at UCSD). As far as I can tell, these have not received rave reviews.

C. Somewhat like Pascal, but a little more difficult, C allows more access to lower-level routines than Pascal does. This language was developed by Bell Labs and is the favorite of UNIX programmers. Currently, at least five or six different Cs are available (see reviews in the Spring 1985 newsletter), and all compile code. (Some compile to assembly language, then compile that to machine code.) Their major differences lie in how they interact with the programmer. Some ignore the standard Macintosh interface for the most part and present the programmer with a command-line format much like UNIX. Others make extensive use of windows and menus during programming and debugging. C is obviously the favorite language of BMUG developers. Ask any of them about the various products out there.

Modula-2. Modula-2 was developed by Nicolas Wirth, the person who dreamt up Pascal. It is very similar to Pascal, but offers some enhancements which clean up input-output commands and procedure-nesting. "MacModula-2" from Modula Corp. of Provo, Utah, is a complete implementation of Wirth's language and includes support for most toolbox routines. Like some of the Pascal products, MacModula compiles into a pseudo-code, this time called m-code. The m-code is then interpreted at run-time, but is very efficient. Programs using many toolbox calls are fast; those doing a lot of number crunching are slower. A public domain version of Modula-2 is currently being distributed through computer networks, and BMUG distributes it.

FORTH. FORTH, by nature, is an incrementally-compiled language that produces very fast programs. It does everything backwards from other languages and relies on a last-in, first-out stack (similar to a Hewlett-Packard calculator) more than it does on named variables. Code is written in blocks of 64 characters by 16 lines and is compiled block by block. Originally developed to control radiotelescopes, it is excellent for robotics and analog-to-digital processing applications. It is also a decent Macintosh programming environment. Currently, two FORTH systems are available: one by Creative Solutions, and one by MicroMotion. The Creative Solutions system can, with its Level Three development system, create standalone applications. It allows access to all of the toolbox routines via their 4-digit hex codes but also supports most of the graphics, menu, window, and control routines directly with FORTH words. If your code is still not fast enough, CSI's FORTH supports in line assembly language. The CSI FORTH is not compatible with FORTH-83, the standard. If you need to run programs in FORTH-83, MicroMotion FORTH can do the job.

LISP. LISP, a list processing language, is the favorite of the artificial intelligence community. It allows easy definition of an manipulation on data types, especially lots of items. Even programs themselves can become data so programs can modify themselves or create new programs. Any program requiring a significant amount of graphics or sentence-parsing should probably be written in this language. In the world of mainframes, LISP is usually run on computers called LISP-machines, essentially fast, sophisticated interpreters. However, on the Mac it is usually compiled. In the serious LISP created by ExperTelligence, it is incrementally compiled, like FORTH. A developers system should be available soon from ExperTelligence. BMUG distributes a public domain version of LISP which presents you with a fully white screen and accepts commands. I'm not sure how complete a version it is, but it comes with a file that allows some PROLOG programs to be run on the Mac. (PROLOG is another AI language that has gained widespread popularity in Japan but is not as popular in the US.)

LOGO. A baby of LISP, the LOGO programming language is especially adept at manipulating graphics. Many children are learning this language on microcomputers, writing procedures to make bunnies or turtles move around the screen, leaving graphic images behind them. The LOGO marketed by ExperTelligence, although simple to learn, is the most extended LOGO available. ExperTelligence says they eventually intend to market a full LOGO development system!

Smalltalk. Smalltalk is a language noted for its ability to create and operate on objects using messages. Developed at Xerox PARC, Smalltalk is very powerful for creating models of systems where events depend on other events and on the state of the "things" or objects producing events. In response to requests from universities, Apple has released a Smalltalk-80 system which will run on Mac-XL's and Macs. One version, Level 0, will run on 512K Macs (and on all XL's), but will be limited in available memory and in the types of classes it supports. The other version, level 1, will run on XL's and Macs which are at least 1 meg (with the memory screen *at the top*). Apple will not be supporting Smalltalk as a regular product, but will simply be charging duplication costs. If you are interested in Smalltalk, write: Smalltalk request, c/o Eileen Crombit, Apple Computer Inc., 20525 Mariani Ave., Cupertino, CA 95014.

Neon. Neon by Kriya Systems is, as far as I know, a language unique to the Mac. It is a mixture of FORTH and Smalltalk, and is incrementally compiled. I haven't heard too much about Kriya's system since it is fairly new. But it can create standalone programs.

Assembler. If you need speed, try writing in 68000 assembly language, marketed directly by Apple. Assembly language is notoriously complicated as it details on a low level every step the processor will perform, just one level above the actual ones and zeros going to the processor. But for programmers working on projects like Jazz and OverView, it was the only way. Advice: start simple and build from there. Of course, all toolbox routines are fully supported -- the hard way.

FORTTRAN. FORTRAN, or FORmula TRANslation, has been around as long as scientists and engineers have been programming. The only reason to use FORTRAN on the Mac is because you already have programs written for other computers that you want to run on your Mac. Absoft's FORTRAN is pure FORTRAN IV (or FORTRAN 66 if desired as an option when you compile) with most toolbox routines added in, and will compile fast, efficient standalone applications directly. The manual is horrid, but the debugger and linker are excellent. Rumors abound that Microsoft wants to market this as their FORTRAN.

THE BOTTOM LINE

Which language you choose to program in really depends, as you might expect, on what you want to do with your Mac. As Reese says, most of us will never really have to program the Mac since most of the useful programs have already been written (database, word processing, page layout, spreadsheet, statistics and graphics packages, etc., etc.). In fact, MacLion, a database program, comes with a language called Leo that allows you to manipulate data; Microsoft Excel allows you to write "macros", or procedures that you can repeat again and again simply by saying "record what I'm doing now"; Odesta's Helix database system comes close to being an icon-based programming system. The line between applications and the languages they are written in begins to blur, as it should. As good as they are for their specialized functions, and as easy as they are to learn, MacLion, Excel, and Helix are not **general purpose** languages.

If you are new to computers, the obvious first choice is BASIC or Pascal. Purchase one of these languages, along with a good book or two on the language from your local bookstore. Try the examples and read the manuals thoroughly. Soon, you will have your Mac drawing hundreds of pretty pictures and calculating everything in sight. You will, most importantly, learn how to state problems in a way that makes them easy to translate into a computer program.

If you do lots of scientific calculations, and need to get numeric speed and easy programming out of the Mac, consider FORTH or FORTRAN. (Don't be confused by the fact that the names of these languages sound similar -- they are very different.) Both languages are easy to learn and make dealing with numbers simpler than some of the other languages.

For programs that require a large variety of graphics effects, consider using Logo or Lisp. Many simple commands already exist in these languages that make manipulating figures on the screen very easy. If most of the graphics can be done easily using toolbox calls, C may be a good choice.

Fall 1985 BMUG G Newsletter

For programs that you seriously want to get to market and make some money with, consider C. Currently, more small developers are using C systems than anything else. There are a large number of good C development systems in the marketplace, and BMUG's Developers' subgroup is a valuable resource in learning how to make the Mac perform well in C.

If you are particularly adventurous, you might want to try Kyria's NEON or a version of Smalltalk. These languages are the beginning of a new breed of computer languages written for people who think more about ideas than numbers and subroutines. However, these systems have not been around long enough to make a great deal of criticism or praise available ... yet.

Whichever language you choose, have patience, and make sure that you distribute any programs you develop either commercially, or for more personal satisfaction, through the BMUG bulletin board and other public domain and user-supported channels. Let us all see the creative things you do!

Happy programming.

Linda Custer is a graduate student in Chemical engineering at UC Berkeley, and files SYSOP for the BMUG BBS.



"On the Great Plains, Near Winner, South Dakota" Photograph by Dorothea Lange (1938)

For information on the MacRecorder hardware and software, contact:

Michael Lamoureux
3024 A Fulton St.
Berkeley, CA 94705
or c/o the Berkeley Macintosh Users Group

Copyright © 1985 by Michael Lamoureux

All rights reserved.

(Copyright 1985 by Michael Lamoureux)

INCLUDE" ADC1"

: fopening (-- | dimming unuseable menu items)

1 false file.menu.id item.enable
2 false file.menu.id item.enable
3 true file.menu.id item.enable
4 true file.menu.id item.enable
5 true file.menu.id item.enable ;

: fclosing (-- | dimming unuseable menu items)

1 true file.menu.id item.enable
2 true file.menu.id item.enable
3 false file.menu.id item.enable
4 false file.menu.id item.enable
5 false file.menu.id item.enable ;

cr ." Including windows and controls"

35 CONTROLS.VSIZE ! (vert size of control region)

NEW.CONTROL PLAY.BUTTON (play button control)

A.PUSH.BUTTON PLAY.BUTTON C.TYPE

" Play " PLAY.BUTTON C.TITLE

10 10 PLAY.BUTTON C.POSITION

NEW.CONTROL RECORD.BUTTON (record button control)

A.PUSH.BUTTON RECORD.BUTTON C.TYPE

" Record " RECORD.BUTTON C.TITLE

76 10 RECORD.BUTTON C.POSITION

NEW.CONTROL STOP.BUTTON (stop button control)

A.PUSH.BUTTON STOP.BUTTON C.TYPE

" Stop " STOP.BUTTON C.TITLE

156 10 STOP.BUTTON C.POSITION

NEW.CONTROL REPITCH.BUTTON (repitch button control)

A.PUSH.BUTTON REPITCH.BUTTON C.TYPE

" RePitch " REPITCH.BUTTON C.TITLE

250 10 REPITCH.BUTTON C.POSITION

NEW.CONTROL PITCH.CONTROL (pitch button scrolling control)

```

A.SCROLL.BAR  PITCH.CONTROL C.TYPE
12 330 28 450  PITCH.CONTROL C.BOUNDS
100          PITCH.CONTROL C.VALUE
50 200       PITCH.CONTROL C.RANGE
NEW.WINDOW RECORDING.WINDOW      ( setup recording window
)
" Recording " RECORDING.WINDOW W.TITLE
37 150 97 340 RECORDING.WINDOW W.BOUNDS
NOT.VISIBLE  RECORDING.WINDOW W.ATTRIBUTES
NEW.CONTROL START.BUTTON          ( start button definition )
A.PUSH.BUTTON START.BUTTON C.TYPE
" Start "    START.BUTTON C.TITLE
20 5        START.BUTTON C.POSITION
NEW.CONTROL RSTOP.BUTTON          ( recording stop button
                                definition )
A.PUSH.BUTTON RSTOP.BUTTON C.TYPE
" Stop "     RSTOP.BUTTON C.TITLE
90 5        RSTOP.BUTTON C.POSITION
NEW.CONTROL TIME.CONTROL          ( set recording time scroll
                                control )
A.SCROLL.BAR TIME.CONTROL C.TYPE
44 10 60 180 TIME.CONTROL C.BOUNDS
0       TIME.CONTROL C.VALUE
0 0     TIME.CONTROL C.RANGE
: PLAYBACK. ( -- | plays between the margins, with error
             checks )
LT.MAR @ RT.MAR @ OVER - DUP 0>
IF  PLAY.BYTES          ( play if rt.mar > lt.mar )
ELSE DROP RCD.DSIZE @ OVER - DUP 0>
  IF  PLAY.BYTES          ( else play if dsize > lt.mar )
  ELSE 2DROP
  THEN
THEN ;
: RECORD. ( -- | deletes selected region, calls up
           RECORDING.WINDOW )
  TRUE changes ! fopening
  LT.MAR @ RT.MAR @ OVER - DUP 0>
  IF  RCD.DELETE DROP LT.MAR @ RT.MAR ! ( delete region )
  THEN RECORDING.WINDOW SELECT.WINDOW ;
: @PITCH.CONTROL ( -- current value | checking pitch control )
  PITCH.CONTROL @ GET.CONTROL ;
: !PITCH.CONTROL ( value -- | set pitch control, and snd.rate )
  50 MAX 200 MIN DUP SAMPLE.RATE @ * 100 / 65536 * 22257 /

```

```

SND.RATE !
PITCH.CONTROL @ SWAP SET.CONTROL ;
: INC.PITCH.CONTROL ( increment -- | increments pitch control )
  @PITCH.CONTROL + !PITCH.CONTROL ( inc control )
  LAST.PART UPDATE.CONTROL      ( highlight the part )
  0      UPDATE.CONTROL ;      ( unhighlight the part )
: DO.PITCH.CONTROL ( increment -- | handles mouse to
                    increment pitch )
  BEGIN MOUSE.BUTTON
  WHILE DUP INC.PITCH.CONTROL
  REPEAT DROP ;
: TRACK.PITCH.THUMB ( -- | tracks thumb on pitch scroll control )
  FOLLOW.MOUSE DROP
  @PITCH.CONTROL SAMPLE.RATE @ * 100 / 65536 * 22257 /
  SND.RATE !
  GET.WINDOW SHOW.CONTROLS ;
: REPITCH. ( -- | resets pitch to normal )
  100 !PITCH.CONTROL
  GET.WINDOW SHOW.CONTROLS ;
: DO.PITCH.PARTS ( -- | handles the different parts of
                 scroll bar )
  LAST.PART
  CASE IN.THUMB OF TRACK.PITCH.THUMB ENDOF
    UP.BUTTON OF -1 DO.PITCH.CONTROL ENDOF
    DOWN.BUTTON OF 1 DO.PITCH.CONTROL ENDOF
    PAGE.UP OF -10 DO.PITCH.CONTROL ENDOF
    PAGE.DOWN OF 10 DO.PITCH.CONTROL ENDOF
  ENDCASE ;
CREATE TIME.MAX 0 , ( maximum time recorded on host Mac )
: SET.TIME.MAX ( -- | sets time.max in tenths of a second )
  ?HEAP.SIZE 5000 - 0 MAX 10 * SAMPLE.RATE @ / TIME.MAX ! ;
: @TIME.CONTROL ( -- current value | reads value of time
                control )
  TIME.CONTROL @ GET.CONTROL ;
: !TIME.CONTROL ( value -- | set time control )
  0 MAX TIME.MAX @ MIN TIME.CONTROL @ SWAP SET.CONTROL ;
: DISP.TIME ( --- | show amt of time to record )
  10 39 MOVE.TO ." Recording Time =" @TIME.CONTROL 10 /MOD
  .." .." .." secs" 10 spaces ;
: START. ( -- | starts recording )
  START.BUTTON @ IN.BUTTON HILITE.CONTROL
  RSTOP.BUTTON @ 255 HILITE.CONTROL

```

```

RT.MAR @ @TIME.CONTROL SAMPLE.RATE @ * 10 /
RECORD.BYTES
@TIME.CONTROL SAMPLE.RATE @ * 10 / RT.MAR +!
START.BUTTON @ 0 HILITE.CONTROL
RSTOP.BUTTON @ 0 HILITE.CONTROL
FLUSH.EVENTS
RECORDING.WINDOW HIDE.WINDOW ;
: RSTOP. ( -- | stops recording )
    RECORDING.WINDOW HIDE.WINDOW ;
: INC.TIME.CONTROL ( increment -- | increments time control )
    @TIME.CONTROL + !TIME.CONTROL ( inc control )
    LAST.PART UPDATE.CONTROL    ( hilight part )
    0    UPDATE.CONTROL    ( unhighlight )
    DISP.TIME ;
: DO.TIME.CONTROL ( inc -- | handles mouse to increment
                    time control )
    BEGIN MOUSE.BUTTON
    WHILE DUP INC.TIME.CONTROL
    REPEAT DROP ;
: TRACK.TIME.THUMB ( -- | follows thumb on time control )
    FOLLOW.MOUSE DROP
    GET.WINDOW SHOW.CONTROLS
    DISP.TIME ;
: DO.TIME.PARTS ( -- | handles the different parts of scroll bar )
    LAST.PART
    CASE IN.THUMB OF TRACK.TIME.THUMB ENDOF
        UP.BUTTON OF -1 DO.TIME.CONTROL ENDOF
        DOWN.BUTTON OF 1 DO.TIME.CONTROL ENDOF
        PAGE.UP OF -10 DO.TIME.CONTROL ENDOF
        PAGE.DOWN OF 10 DO.TIME.CONTROL ENDOF
    ENDCASE ;
: RECORDING.SELECT ( -- | checks which selection is pushed )
    LAST.CONTROL DUP TOGGLE.CONTROL
    CASE START.BUTTON @ OF START. ENDOF
        RSTOP.BUTTON @ OF RSTOP. ENDOF
    ENDCASE ;
: DO.RECORDING.MOUSE ( -- | handles mouse driven controls )
    IN.CONTROL?
    IF LAST.CONTROL TIME.CONTROL @ =
        IF DO.TIME.PARTS
        ELSE FOLLOW.MOUSE
            IF RECORDING.SELECT THEN
                THEN

```

```

    THEN ;
: RECORDING.PROG ( flag -- | routine for the RECORDING.WINDOW )
  IF SRCCOPY TEXTMODE
    0 !TIME.CONTROL
    SET.TIME.MAX
    TIME.CONTROL @ 0 TIME.MAX @ SET.CONTROL.RANGE
    RECORDING.WINDOW SHOW.WINDOW DISP.TIME
    BEGIN DO.EVENTS MOUSE.DOWN =
      IF DO.RECORDING.MOUSE THEN
        AGAIN
      ELSE RECORDING.WINDOW HIDE.WINDOW
      THEN ;
: SCROLL.GRPH ( -- | scrolls along waveform )
  strt.gph @
  @hpos comp.fact @ * 10 *
  dup strt.gph !
  - comp.fact @ /
  dup abs gph.hsize @ <
  IF gph.box over 0 get.window 122 + @ scroll
    dup 0> IF 0 strt.gph @ rot graph.test
      ELSE dup gph.hsize @ + dup comp.fact @ *
        strt.gph @ + rot negate graph.test
      THEN
    ELSE drop graph.redo
    THEN ;
: INC.SCROLL.CONTROL ( n -- | increment control by n )
  @HPOS + !HPOS
  LAST.PART UPDATE.CONTROL
  0 UPDATE.CONTROL
  SCROLL.GRPH ;
: DO.SCROLL.CONTROL ( inc -- | handles mouse to increment scroll
  control )
  BEGIN MOUSE.BUTTON
  WHILE DUP INC.SCROLL.CONTROL
  REPEAT DROP ;
: TRACK.SCROLL.THUMB ( -- | handles the thumb control )
  FOLLOW.MOUSE DROP
  GET.WINDOW SHOW.CONTROLS
  SCROLL.GRPH ;
: DO.SCROLL.PARTS ( part.code -- | performs the part function )
  CASE IN.THUMB OF TRACK.SCROLL.THUMB ENDOF
  UP.BUTTON OF -1 DO.SCROLL.CONTROL ENDOF
  DOWN.BUTTON OF 1 DO.SCROLL.CONTROL ENDOF

```

```

        PAGE.UP    OF -10 DO.SCROLL.CONTROL ENDOF
        PAGE.DOWN  OF 10 DO.SCROLL.CONTROL ENDOF
    ENDCASE ;
: MAIN.SELECTED ( -- | toggle control and do it )
    LAST.CONTROL DUP TOGGLE.CONTROL
    CASE PLAY.BUTTON @ OF HUSH PLAYBACK. ENDOF
        RECORD.BUTTON @ OF HUSH RECORD. ENDOF
        STOP.BUTTON @ OF HUSH ENDOF
        REPITCH.BUTTON @ OF REPITCH. ENDOF
    ENDCASE ;
: REGION.SELECTION.1 ( -- | selects a data range, and scrolls )
    @mouse.dn pt>offset dup last.mar !
    with.shift? if extend.mar
        else reset.mar
        then
    begin still.down
        while @mouse dup
            point>xy drop
            dup 0 > not if -10 inc.scroll.control then
                gph.hsize @ < not if 10 inc.scroll.control then
                    pt>offset update.mar
            repeat ;
: DO.MAIN.MOUSE ( -- | handles mouse driven controlsa )
    IN.CONTROL?
    IF IN.SCROLL.BAR?
        IF SWAP DROP DO.SCROLL.PARTS
        ELSE LAST.CONTROL PITCH.CONTROL @ =
            IF DO.PITCH.PARTS
            ELSE FOLLOW.MOUSE IF MAIN.SELECTED THEN
                THEN
            THEN
        ELSE @mouse.dn point>xy swap drop controls.vsize @ >
            IF REGION.SELECTION.1 THEN
                THEN ;
include" fileio blocks"

```

(** NOTE:

fileio blocks is a public domain routine distributed
 by NMFUG -- National MacFORTH users group -- and
 available through PSG4 on COMPUSERVE

it is a standard Macintosh file IO interface

it contains the following definitions

```
data.open ( opens data file )
data.create ( creates data file )
```

and it creates the following variables

```
Ofile ( output file )
Ifile ( input file )
Ifile ( input file name )
Oname ( output file name )
```

Parts of the code in small letters are likely to be optimized slightly in release version.

**)

```
: fopen ( -- | opens a data file and retrieves data )
-1 ' ifile ! data.open ifile -1 > if
  rcd.dsize 4 0 ifile read.virtual
  sample.rate 4 4 ifile read.virtual
  repitch.
  0 rcd.dsize @ ?heap.size 5000 - 0 max min
  0 rcd.dsize ! rcd.insert drop
  rcd.handle @@ rcd.esize @ + rcd.dsize @ 8 ifile
    read.virtual
  ifile ' ofile ! ( so that saves will go to this
    file)
  fopening
  iname sys.window set.wtitle TRUE named !
  FALSE changes !
  sys.window show.window sys.window window
  sys.window select.window then ;
: fsaves ( -- | names a file and saves data to it )
  named @ IF ( one file already open )
    ofile ( save ofile on stack )
  -1 ' ofile ! data.create ofile -1 >
  if ofile rcd.to.file ofile close ofile remove then
  ' ofile ! ( restore ofile )
  ELSE ( no file yet open )
```

```

-1 ' ofile ! data.create ofile -1 >
if ofile rcd.to.file oname sys.window set.wtitle
true named ! false changes ! then THEN ;
: fsave ( -- | saves file )
changes @ IF
ofile -1 > if ofile delete
ofile create.file file.error?
ofile open file.error?
ofile get.file.info
"data ofile >fcb +fcb.type !
"maca ofile >fcb +fcb.appl !
ofile set.file.info
ofile rcd.to.file
false changes !
else fsaveas
then THEN ;
: fnew ( -- | opens a new file )
" untitled " sys.window set.wtitle FALSE named !
-1 ' ifile ! -1 ' ofile ! TRUE changes !
fopening
sys.window show.window sys.window select.window ;
: fclose ( -- | closes a file )
( fsave ) ( for now, no saving done here )
ofile -1 > if ofile close ofile remove then
ifile -1 > if ifile close ifile remove then
0 lt.mar ! 0 rt.mar ! 0 last.mar !
0 rcd.dsize @ rcd.delete drop
fclosing
sys.window hide.window ;
: fquit ( -- | quits program )
fclose bye ;
: file.menu ( -- | menu creation )
file.menu.id delete.menu
0 " File " file.menu.id new.menu
" New(;Open...;Close;Save;Save As...;-(;Quit"
file.menu.id append.items
file.menu.id menu.selection:
case 1 of fnew endof
2 of fopen endof
3 of fclose endof
4 of fsave endof
5 of fsaveas endof
7 of fquit endof

```



```

endcase 0 hilite.menu graph.redo ;
: MAIN.PROG ( flag -- | main program for main window )
  IF GRAPH.REDO
    BEGIN DO.EVENTS
      CASE IN.CLOSE.BOX OF hide.mark fclose   ENDOF
          IN.SIZE.BOX OF GRAPH.REDO         ENDOF
          MOUSE.DOWN OF hide.mark DO.MAIN.MOUSE
          show.mark ENDOF

      ENDCASE
    AGAIN
  ELSE
    THEN ;
: edit..menu ( -- | menu creation )
  edit..menu.id delete.menu
  0 " Edit. " edit..menu.id new.menu
  " Undo(;Cut(;Copy(;Paste(;Clear;-(;Reverse"
    edit..menu.id append.items
  edit..menu.id menu.selection:
    case 5 of delete.selection  endof
        7 of flip.selection  endof
    endcase
  TRUE changes !
  0 hilite.menu GRAPH.REDO ;
: comp.menu ( -- | menu creation )
  comp.menu.id delete.menu
  0 " Compression " comp.menu.id new.menu
  " x1;x10;x100;x1000" comp.menu.id append.items
  comp.menu.id menu.selection:
    1 FALSE comp.menu.id item.check ( remove check marks )
    2 FALSE comp.menu.id item.check
    3 FALSE comp.menu.id item.check
    4 FALSE comp.menu.id item.check
  dup TRUE comp.menu.id item.check ( add a check mark )
  case 1 of 1  comp.fact ! endof
      2 of 10  comp.fact ! endof
      3 of 100  comp.fact ! endof
      4 of 1000  comp.fact ! endof
  endcase 0 hilite.menu GRAPH.REDO ;
: MAIN.CONFIG ( wptr -- wptr | reconfigure sys.win )
  >R
  " MacRecorder " I W.TITLE
  CLOSE.BOX SIZE.BOX SCROLL.LEFT/RIGHT + +
  I W.ATTRIBUTES

```

```

40 10 340 500      I W.BOUNDS
I ON.ACTIVATE MAIN.PROG
R> ;
: APPEND.WINDOWS.ETC ( -- | appends windows, sets up menus
                        and controls )
APPLE.MENU file.menu edit..menu
COMP.MENU DRAW.MENU.BAR
GET.WINDOW PLAY.BUTTON  APPEND.CONTROL
GET.WINDOW RECORD.BUTTON APPEND.CONTROL
GET.WINDOW STOP.BUTTON  APPEND.CONTROL
GET.WINDOW REPITCH.BUTTON APPEND.CONTROL
GET.WINDOW PITCH.CONTROL APPEND.CONTROL
RECORDING.WINDOW ADD.WINDOW
RECORDING.WINDOW START.BUTTON APPEND.CONTROL
RECORDING.WINDOW RSTOP.BUTTON APPEND.CONTROL
RECORDING.WINDOW TIME.CONTROL APPEND.CONTROL
RECORDING.WINDOW ON.ACTIVATE RECORDING.PROG ;
: MAIN.SETUP ( -- | level 3 MacFORTH standalone turnkey
              token )
GINIT
0 CONTROLS.VSIZE @ XYOFFSET
APPEND.WINDOWS.ETC
INIT.ALL ;

(*** PROGRAM ADC1:: ***)

FORTH DEFINITIONS
20000 MINIMUM.OBJECT ( establish minimum object & Vocab )
5048 MINIMUM.VOCAB
( INCLUDE" Controls" )
( INCLUDE" Assembler" )
( "controls" and "assembler" blocks files are standard
  with Level 2 MacFORTH )
create snd.rate 0 ,      ( sound rate for playback)
create sample.rate 0 ,  ( rate of the digitizer )
CREATE flip.table 256 allot ( a lookup table )
create comp.fact 10 ,    ( compression factor for graphics )
create rcd.handle 0 ,   ( handle to sound record data )
create rcd.dsize 0 ,    ( sound record data size )
create rcd.esize 6 ,    ( extra size -- six bytes -- for
                        std sound record )
create lt.mar 0 ,      ( left margin in sound record
                        data )

```

```

create rt.mar 0 ,      ( right margin in sound record
                        data )
create last.mar 0 ,   ( last margin selected in sound
                        record dta )
create gph.hsize 500 , ( horizontal size of graphics
                        region )
create gph.vsize 280 , ( vertical size of graphics
                        region )
create strt.gph 0 ,   ( offset for start of graph )
0 0 0 0 rect gph.box  ( size of graphics box )
create controls.vsize 0 , ( vertical size of controls )
11 constant file.menu.id
12 constant edit..menu.id
13 constant comp.menu.id
create changes FALSE , ( boolean for whether changes
                        made )
create named FALSE , ( boolean for whether file has
                        been named )
create init.length 0 , ( initial length of file )
hex      ( various system constants for the SCC chip )
9FFFF8 constant SCCRBase ( SCC base read address      )
BFFFF9 constant SCCWBase ( SCC base write address     )
    6 constant aData    ( offset for A channel data    )
    2 constant aCtl     ( offset for A channel control  )
decimal
: sccpeek ( n -- read[n] | to read value in read register n )
    sccwbase actl + c! sccrbase actl + c@ ;
: sccpoke ( n1\n2 -- | to write value n1 into write reg. n2 )
    sccwbase actl + c! sccwbase actl + c! ;
: SCC.INIT ( -- | initializes the SCC chip )
136 15 sccpoke ( turn off the CTS interrupt )
    3 1 sccpoke ( turn off the Rx interrupt )
    0 14 sccpoke ( null command )
48 11 sccpoke ( use TRxC as the receiver clock only )
    0 10 sccpoke ( null command )
    8 9 sccpoke ( initialize master reset )
    98 5 sccpoke ( initialize transmitter )
    68 4 sccpoke ( x16 clock mode, trans. 1 stop bit, no par )
193 3 sccpoke ( initialize receiver, 8 bits ) ;
( creating a purgeable assembler )
CREATE OTHER.DP 0 , ( creates alternate dictionary pointer )
: SWAP.DP ( -- | swapping dictionary pointers, see Level 2 doc )
    DP @ OTHER.DP @ DP ! OTHER.DP ! ;

```

```

HERE 4096 + OTHER.DP ! SWAP.DP
: ?ABOVE ( -- | see Level 2 doc )
  LATEST 2-
  BEGIN 2DUP >W@< DUP
  WHILE TOKEN>ADDR <
    IF >W@< EXIT THEN
      2+ COUNT 31 AND +
  REPEAT 2DROP NOT ;
: PRUNE ( -- | removes excess definitions )
  BEGIN ?ABOVE DUP
  WHILE DUP DEALLOT BEHEAD
  REPEAT 2DROP ;
INCLUDE" Assembler"
SWAP.DP ( swapping dictionary pointers )

```

```

CODE MAKE.FLIP.TABLE ( addr -- | makes the lookup table at addr)
  A0 POP, ( addr )
  256 # A0 WORD ADDA, ( starts at top of the table )
  255 # D0 WORD MOVE, ( counter for outer loop )
  BEGIN,
    D0 D1 BYTE MOVE, ( d1 gets untransformed value )
    07 # D3 WORD MOVE, ( counter for inner loop )
    BEGIN,
      D1 01 # BYTE LSR, D2 01 # BYTE ROXL,
      ( d2 gets transformed value )
    D3 WLOOP,
    D2 A0 -) BYTE MOVE, ( store value in table )
  D0 WLOOP, NEXT END-CODE

```

```

CODE RECORD.ASS ( addr\cnt\addr1 -- | load cnt bytes from the
                serial port to addr; addr1= lookup table )
  A1 POP, D0 POP, A0 POP, ( table addr, count, addr )
  D1 LONG CLR,
  D0 01 LONG SUBQ, ( pre-decrement count )
  BEGIN, ( begin the input loop )
  BEGIN, ( begin the wait for Rx char loop )
  SccRBase aCtl + @#L 00 # BTST, NE ( check Scc for Rx char )
  UNTIL,
  SccRBase aData + @#L D1 BYTE MOVE, ( RX char goes to D1 )
  D1 00 WORD A1 @I) A0 )+ BYTE MOVE, ( translate via lookup tab )
  D0 LOOP,
  NEXT END-CODE

```

```

CODE GRAPH.COMP ( scrn\addr\cnt\fact -- | graphs min to max )
DO POP, D6 POP, A0 POP, D4 POP, D0 PUSH, ( save fact on stack )
BEGIN,
    SP ()      D0 LONG MOVE, ( get fact on stack )
    D0         01 LONG SUBQ, ( predecrement counter )
    255 #      D1 LONG MOVE, ( initial min )
    0 #        D2 LONG MOVE, ( initial max )
    BEGIN,    D3 LONG CLR, ( clear upper bytes )
              A0 )+ D3 BYTE MOVE, ( get value at addr )
              D1 D3 WORD CMP, ( compare for a min )
    MI IF, D3  D1 WORD MOVE, THEN, ( save min )
              D2 D3 WORD CMP, ( compare for a max )
    PL IF, D3  D2 WORD MOVE, THEN, ( save max )
    DO LOOP,          ( do min\max loop )
    A0 D5 LONG MOVE,  ( Preserve A0 in D5 )
    D4 PUSH, D2 PUSH, D4 PUSH, D1 PUSH, ( X\Ymax\X\Ymin )
    IP RP -) LONG MOVE, ( protect IP )
              ( D4 D5 D6 automatically protected )
    >FORTH VECTOR >CODE ( graph min to max )
    RP )+ IP LONG MOVE, ( restore IP )
    D5 A0 LONG MOVEA, ( restore A0 )
    D4 01 LONG ADDQ, ( increment scrn )
    D6 01 LONG SUBQ, EQ ( decrement count )
UNTIL,
    D0 POP, ( discard fact, which was still on stack )
NEXT END-CODE

CODE FLIP.ASS ( addr1\addr2 -- | reverse order of data bytes )
A2 POP, A1 POP,
BEGIN, A1 () DO  BYTE MOVE,
    A2 () A1 )+ BYTE MOVE, ( A1 is incremented )
    DO  A2 () BYTE MOVE,
    A2  1  LONG SUBQ, ( A2 is decremented )
    A1  A2  LONG CMPA, ( A2 - A1 )
    MI UNTIL, ( stop if A2 < A1 )
NEXT END-CODE
." Purging assembler"
CREATE MARKER ' ASSEMBLER @ PURGABLE ' FLIP.ASS PRUNE (
purges
                                assembler )
: graph1 ( scrn\addr\cnt -- | graphs cnt bytes, starting at addr )
    3 pick 3 pick c@ move.to ( position pen on screen )
    over + swap do ( loop from addr to addr+cnt )

```

```

dup i c@ draw.to 1+ loop ( draw, and increment scrn )
drop ; ( clear stack )
: rcd.delete ( d\cnt -- Bool | will delete cnt bytes from the data
rcd starting at offset d.
Space returned to the heap. )
over rcd.handle @@ rcd.esize @ + + ( offset d )
2dup + swap ( offset d + cnt )
rcd.dsize @ 5 roll - 4 pick - cmove ( delete cnt )
negate rcd.dsize +! ( decrement dsize by cnt )
rcd.handle @ rcd.dsize @ rcd.esize @ + resize.handle
( return extra room to the heap )
if FALSE else TRUE then ;
( TRUE if successfully returned to heap )
: rcd.insert ( d\cnt -- Bool | will insert cnt bytes into the data
rcd starting at offset d.
Returns TRUE if successful )
rcd.handle @ over rcd.esize @ rcd.dsize @ + +
resize.handle if ( increase space to handle by cnt )
2drop FALSE ( unsuccessful )
else
over rcd.handle @@ rcd.esize @ + + ( offset d )
2dup + ( offset d + cnt )
rcd.dsize @ 5 roll - cmove> ( make space )
rcd.dsize +! ( increment dsize by cnt )
TRUE ( successful )
then ;
hex
A403 OS.TRAP A.WRITE ( buf.ptr -- | executes device write )
decimal
: long.aplay ( lnth\addr -- | pass the addr and lnth of a sound
synth. record to the sound driver
and plays it asynchronously )
sound.fcb ( a file control block in FORTH )
0 over 12 + ! ( ioCompletion ptr )
65532 over 24 + w! ( ioRefNum = -4 )
swap over 32 + ! ( ioBuffer ptr = addr )
swap over 36 + ! ( ioRegCount = lnth )
0 over 44 + w! ( ioPosMode )
0 over 46 + ! ( ioPosOffset )
a.write ( execute the device write call ) ;
: record.bytes ( d\cnt -- | record cnt bytes at offset d in data
rcd)
dup 0= if 2drop ( if cnt=0, done )

```

```

else 2dup rcd.insert          ( insert space )
  if swap rcd.handle @@ rcd.esize @ + + ( offset d )
    swap flip.table record.ass    ( input cnt bytes)
  else 10 39 move.to ." insufficient heap space "
  then then ;
: play.bytes ( d \cnt -- | plays cnt bytes from data rcd, at
              offset d)
  swap dup 2 mod - swap          ( make d even )
  over rcd.handle @@ + pad rcd.esize @ cmove ( save 6 bytes )
  0      rcd.handle @@ 4 pick + w!    ( freeform mode )
  snd.rate @ rcd.handle @@ 4 pick + 2+ ! ( sndrate longword)
  rcd.handle @ lock.handle
  6+ rcd.handle @@ 3 pick + long.aplay ( play the record )
  rcd.handle @ unlock.handle
  rcd.handle @@ + pad swap rcd.esize @ cmove ;
: graph.test ( scrn\offset\cnt -- | graph starting at scrn from
              offset in rcd. Goes up
              to cnt, or end of rcd. )
rcd.dsize @ 3 pick - comp.fact @ /
min dup 0>          ( compare cnt to end of rcd )
IF 3 pick 3 pick 3 pick ( scrn\off\cnt\s\o\c )
  swap rcd.handle @@ rcd.esize @ + + swap ( s\o\c\s\addr\c)
  comp.fact @ 1 = IF graph1
  ELSE comp.fact @ graph.comp
  THEN
    ( scrn\off\cnt )
  rt.mar @ 3 pick - comp.fact @ / min 3 pick +
  lt.mar @ rot - comp.fact @ / 0 max rot + swap
  2dup < IF 0 swap 260 invert rectangle
  ELSE 2drop THEN
ELSE 2drop drop THEN ;
: graph.clear ( -- | clears out the graphing area )
  gph.box erase.rect ;
: round ( n -- n' | round down to a multiple of comp.fact )
  comp.fact @ / comp.fact @ * ;
: to.hpos ( n -- n' | converts an offset to a hbar value )
  comp.fact @ / 10 / ;
: pt>offset ( pt -- n | translate point on screen to offset)
  point>xy drop 0 max gph.hsize @ min ( keep x value on screen)
  comp.fact @ * strt.gph @ + 0 max rcd.dsize @ min ;
: offset>scrn ( n -- n' | translate an offset to scrn point )
  strt.gph @ - comp.fact @ / 0 max gph.hsize @ min ;
: with.shift? ( -- Bool | returns true if shifted mouse down )

```

```

    mouse.down.record 16 + w@ 512 and if TRUE else FALSE then ;
: offset.invert ( offset1\offset2 -- | inverts on graph )
    offset>scrn 0 rot offset>scrn 280 invert rectangle ;
variable mark.state mark.state off
: invert.mark ( -- | change the mark on screen at lt.mar )
lt.mar @ dup comp.fact @ + offset.invert
    -1 mark.state @ - mark.state ! ;
: hide.mark ( -- | remove mark line at lt.mar )
mark.state @ if invert.mark then ;
: show.mark ( -- | place a marker line at lt.mar )
lt.mar @ rt.mar @ = mark.state @ not AND if invert.mark then ;
: graph.redo ( -- | redoes the entire graph )
    hide.mark    ( finds the current graph area size )
    GET.WINDOW +WCBOUNDS DUP 4+ W@    ( vertical window size )
    CONTROLS.VSIZE @ 0 MAX - gph.vsize ! ( vert gph region size )
    6+ W@ gph.hsize !    ( horizontal window size )
    CONTROLS.VSIZE @ 0
    gph.vsize @ 3 PICK + gph.hsize @ gph.box !rect ( gph region )
    100 gph.vsize @ 100 * 260 / xyscale ( scale to gph region )
    0 -1 gph.hsize @ -1 vector ( line below control region )
    strt.gph @ rcd.dsize @ >
IF lt.mar @ strt.gph ! THEN ( enforce strt <= dsize )
    strt.gph @ round strt.gph !    ( round down )
    GET.WINDOW +HBAR @ 0 rcd.dsize @ to.hpos SET.CONTROL.RANGE
    strt.gph @ to.hpos !hpos ( update the scroll thumb )
    GET.WINDOW SHOW.CONTROLS ( redraw all controls )
    graph.clear 0 strt.gph @ gph.hsize @ graph.test show.mark ;
: reset.mar ( offset -- | resets lt.mar and rt.mar )
    lt.mar @ rt.mar @ offset.invert ( uninvert last selection )
    dup lt.mar ! rt.mar ! ;
: extend.mar ( offset -- | extends the appropriate margin )
    dup lt.mar @ < IF lt.mar @ over lt.mar !
        ELSE rt.mar @ over rt.mar ! THEN
        offset.invert ;
: update.mar ( offset -- | updates the margins )
    dup last.mar @ offset.invert ( invert from last offset )
    dup rt.mar @ last.mar @ = if rt.mar ! else lt.mar ! then
    last.mar !
    rt.mar @ lt.mar @ < if rt.mar @ lt.mar @ rt.mar ! lt.mar ! then
    ;
: FLIP.SELECTION ( -- | reverse data between lt.mar and rt.mar )
    LT.MAR @ RT.MAR @ <
    IF RCD.HANDLE @@ RCD.ESIZE @ + DUP

```



```

    LT.MAR @ + SWAP RT.MAR @ + FLIP.ASS
    THEN ;
: DELETE.SELECTION ( -- | delete data between lt.mar and rt.mar )
    LT.MAR @ RT.MAR @ <
    IF LT.MAR @ RT.MAR @ OVER - RCD.DELETE drop
    lt.mar @ rt.mar !
    THEN ;
: INIT.ALL ( -- | initializes SCC chip and various constants )
    SCC.INIT
    9600 sample.rate ! ( initialize input rate )
    sample.rate @ 65536 * 22257 / snd.rate ! ( playback rate )
    flip.table make.flip.table
    rcd.esize @ from.heap rcd.handle ! ( initialize space for
        rcd ) ;
: RCD.TO.FILE ( file# -- | saves the record in file # )
    rcd.dsize 4 0 4 pick write.virtual
    sample.rate 4 4 4 pick write.virtual
    rcd.handle @@ rcd.esize @ +
        rcd.dsize @ 8 4 pick write.virtual
    drop
    ( should make a check here to see it recorded ) ;

```

region)
create strt.gph 0 , (
Ät7w14 ^ Äà™ èz1N Äà™! z-& ^ (úÄWord Rescue ¶ @8f ~ º

Äfi g @n ...n` **Error!**

: rcd.delete (d\cnt -- Bool I will delete cnt bytes from the data

□ □□6= rcd starting at offset d.

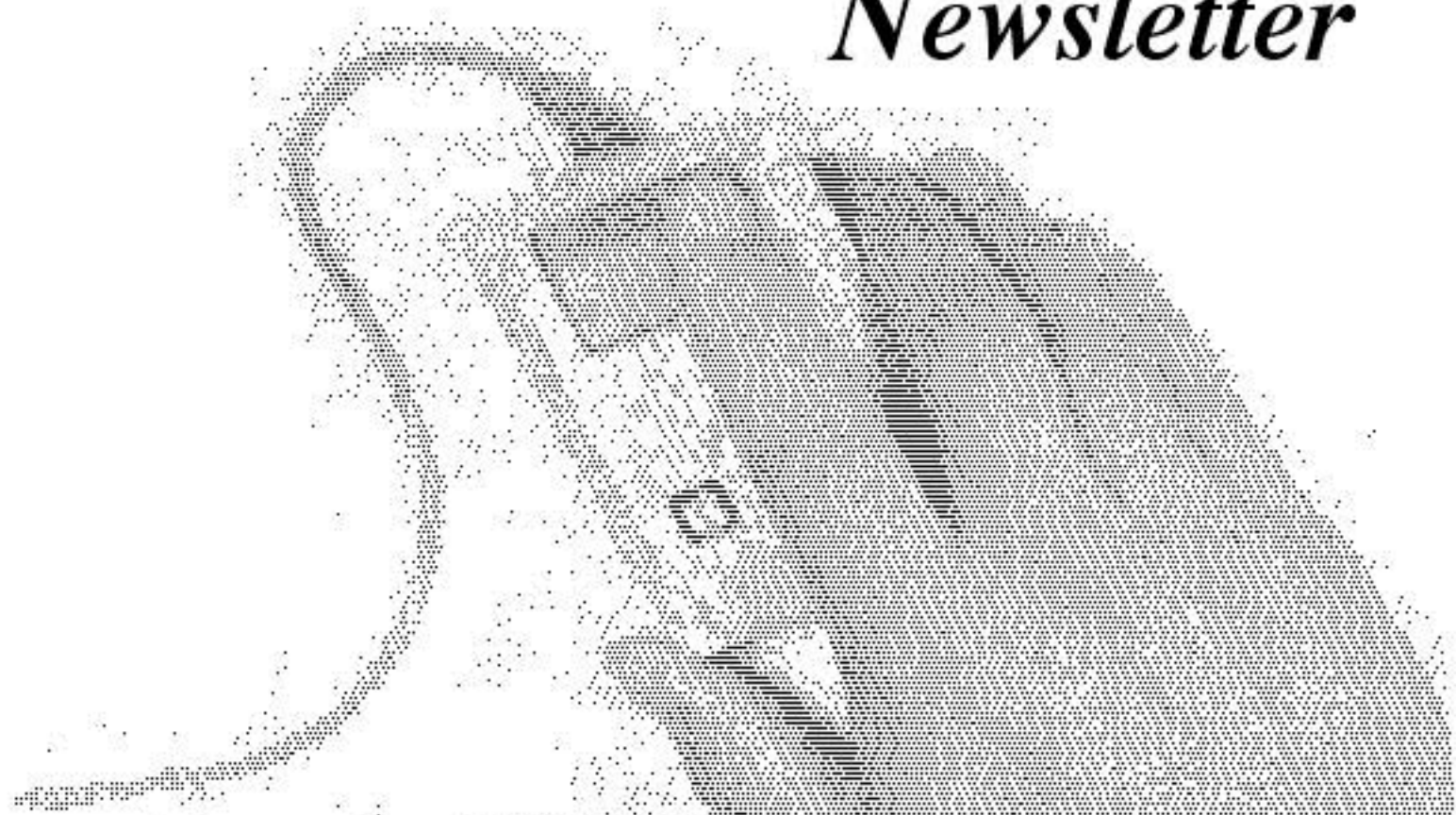
Space returned to the heap.)

over

***Berkeley
Macintosh
Users
Group***



*Fall 1985
Newsletter*



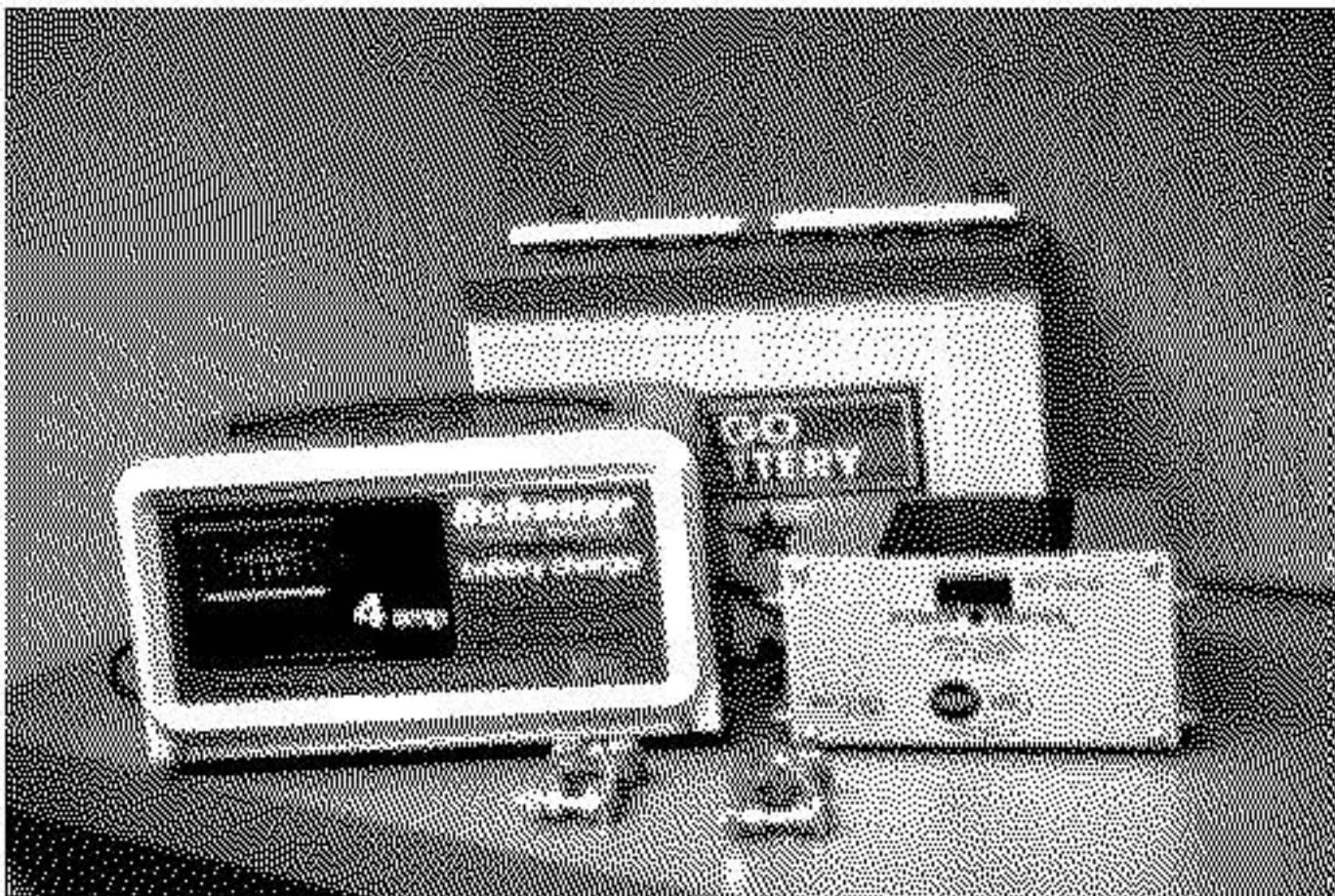


Figure 2. The main components of the uninterruptible power supply -- (clockwise from left) the battery charger, the battery, the power inverter, and the battery terminals.



Figure 3. Clipping and stripping the power inverter supply wires from the cigar lighter plug.



Figure 4. Clipping and stripping the charger power-out wires from their battery clips.

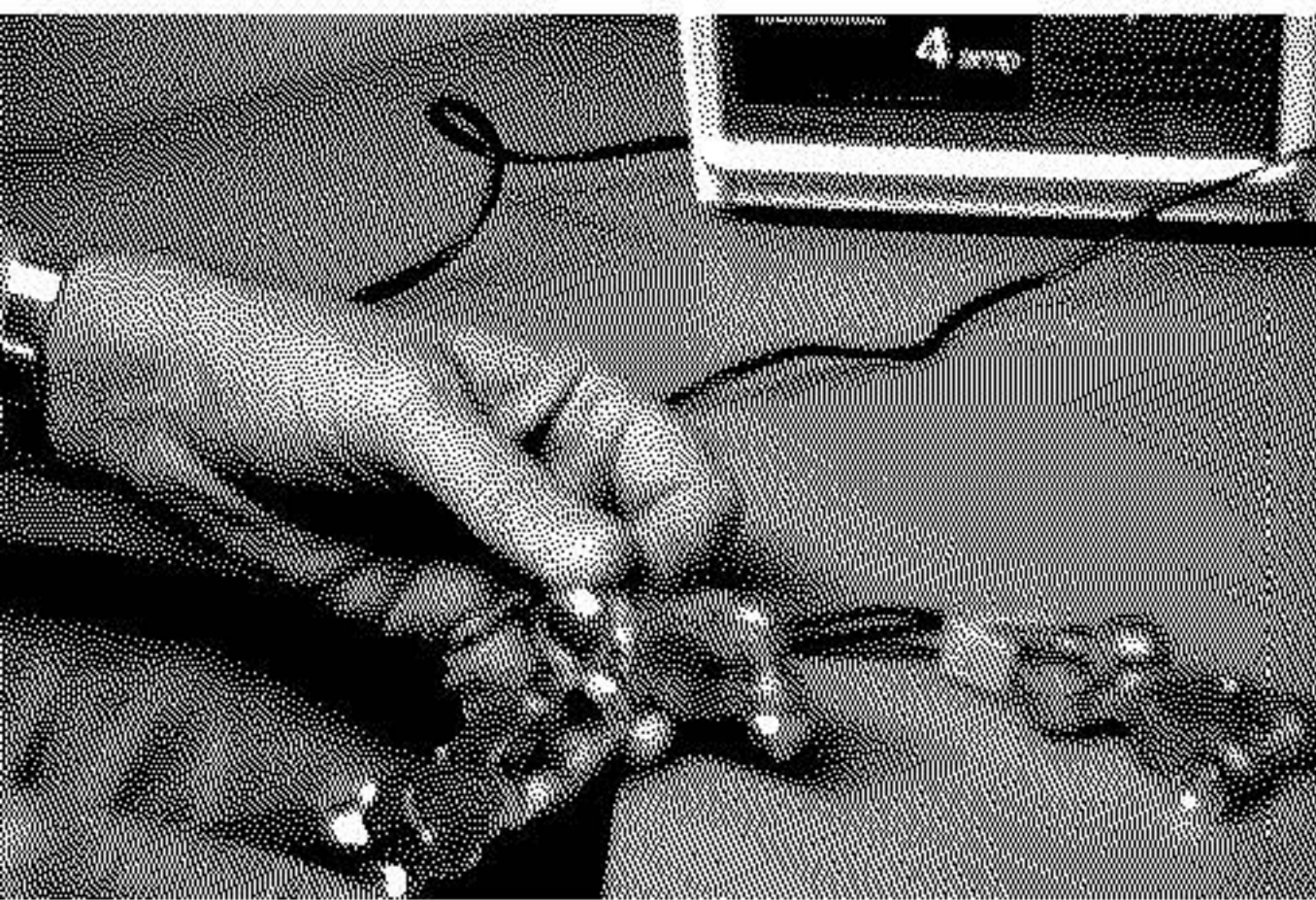


Figure 5. Connecting the inverter and charger negative leads to a battery terminal (the positive leads have already been crimped on to the other terminal).



Figure 6. Wrapping the two negative leads together, into a single cable to the negative battery terminal (the positive leads have already been wrapped).

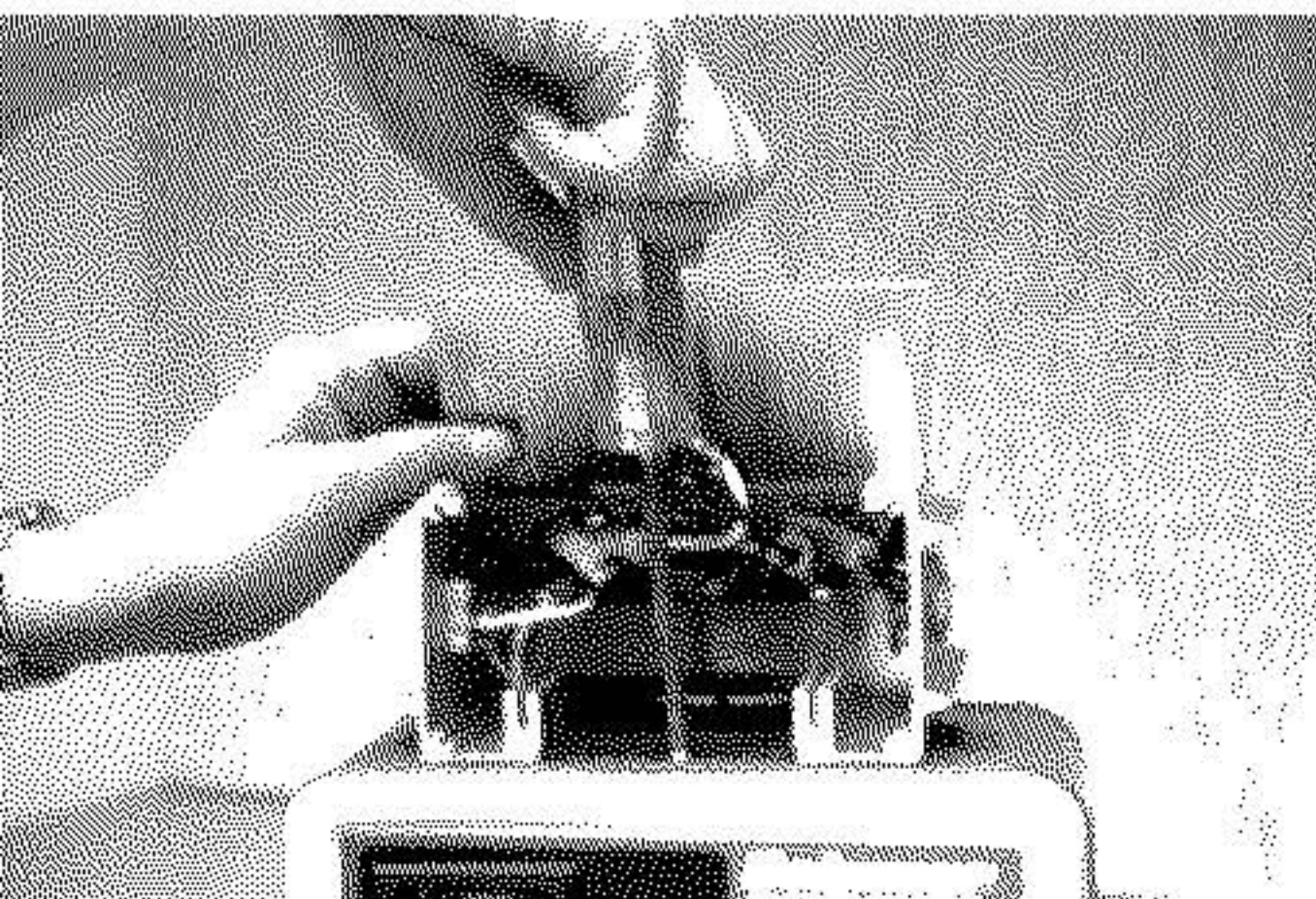


Figure 7. Attaching the inverter case to the charger case with two of the charger case screws through mesh holes in the inverter's top plate.

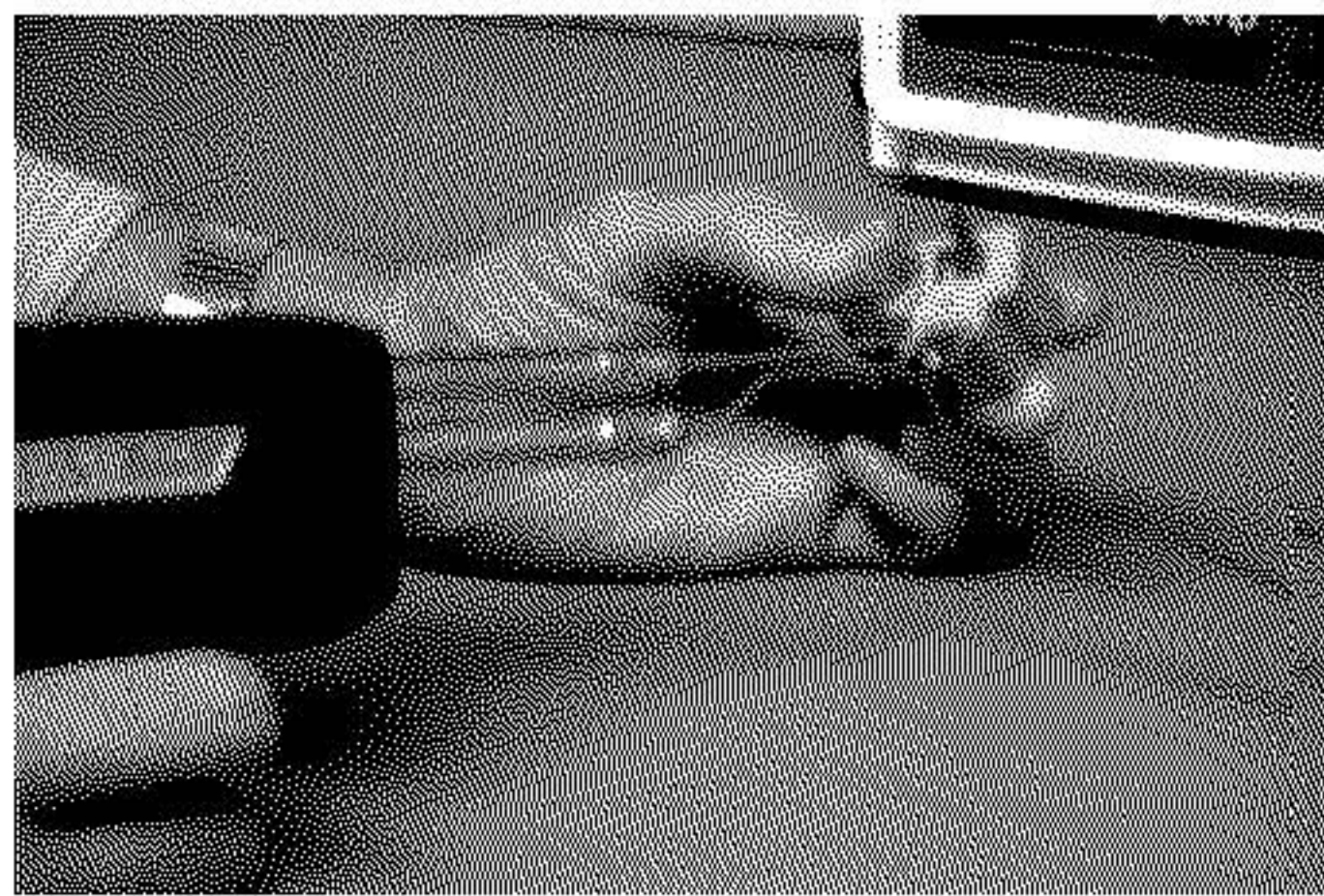


Figure 8. Soldering a one-foot length of hookup wire to the ground tab of the three-prong adapter.

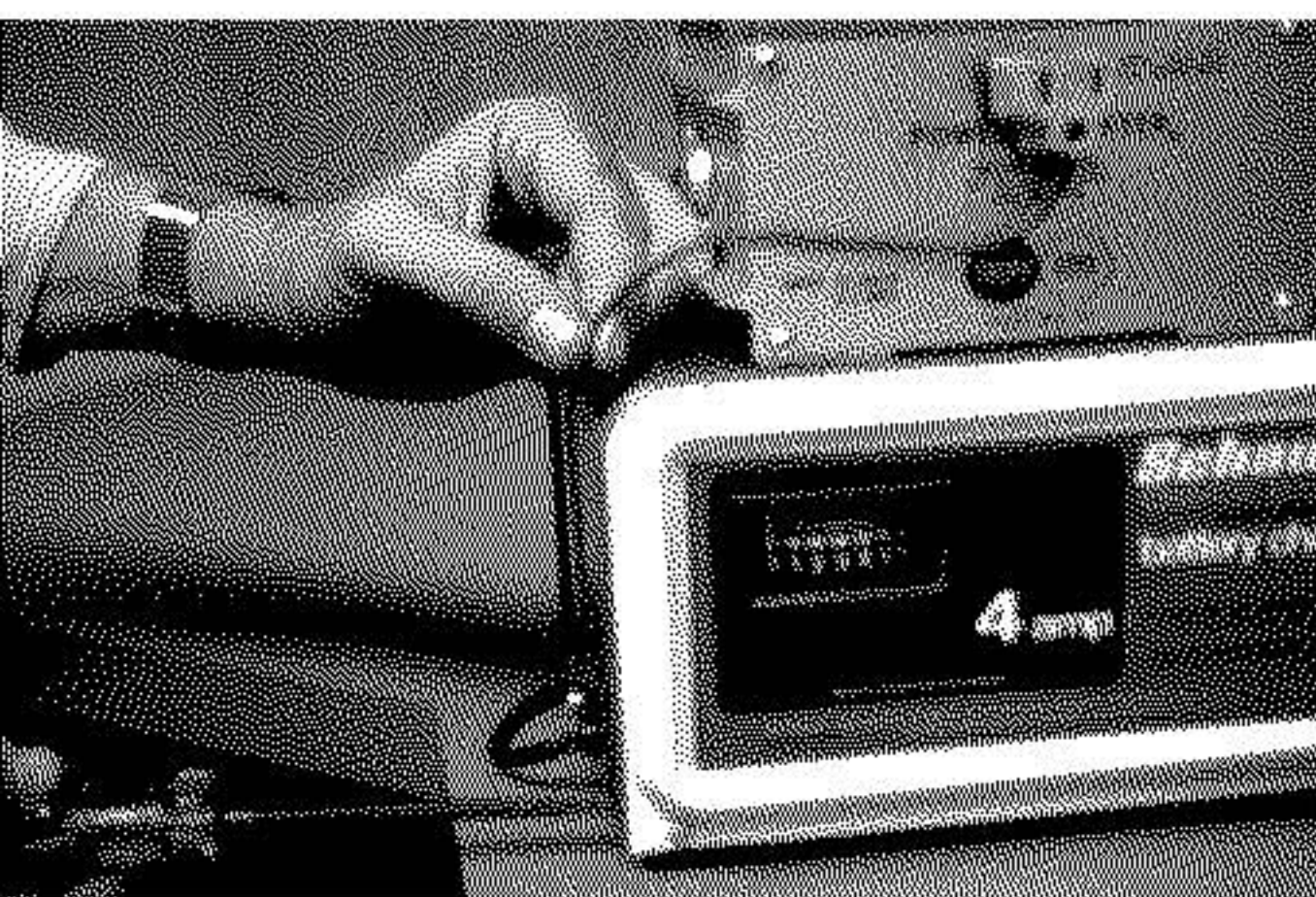


Figure 9. Attaching the three-prong adapter's ground wire to the grounded case of the charger.

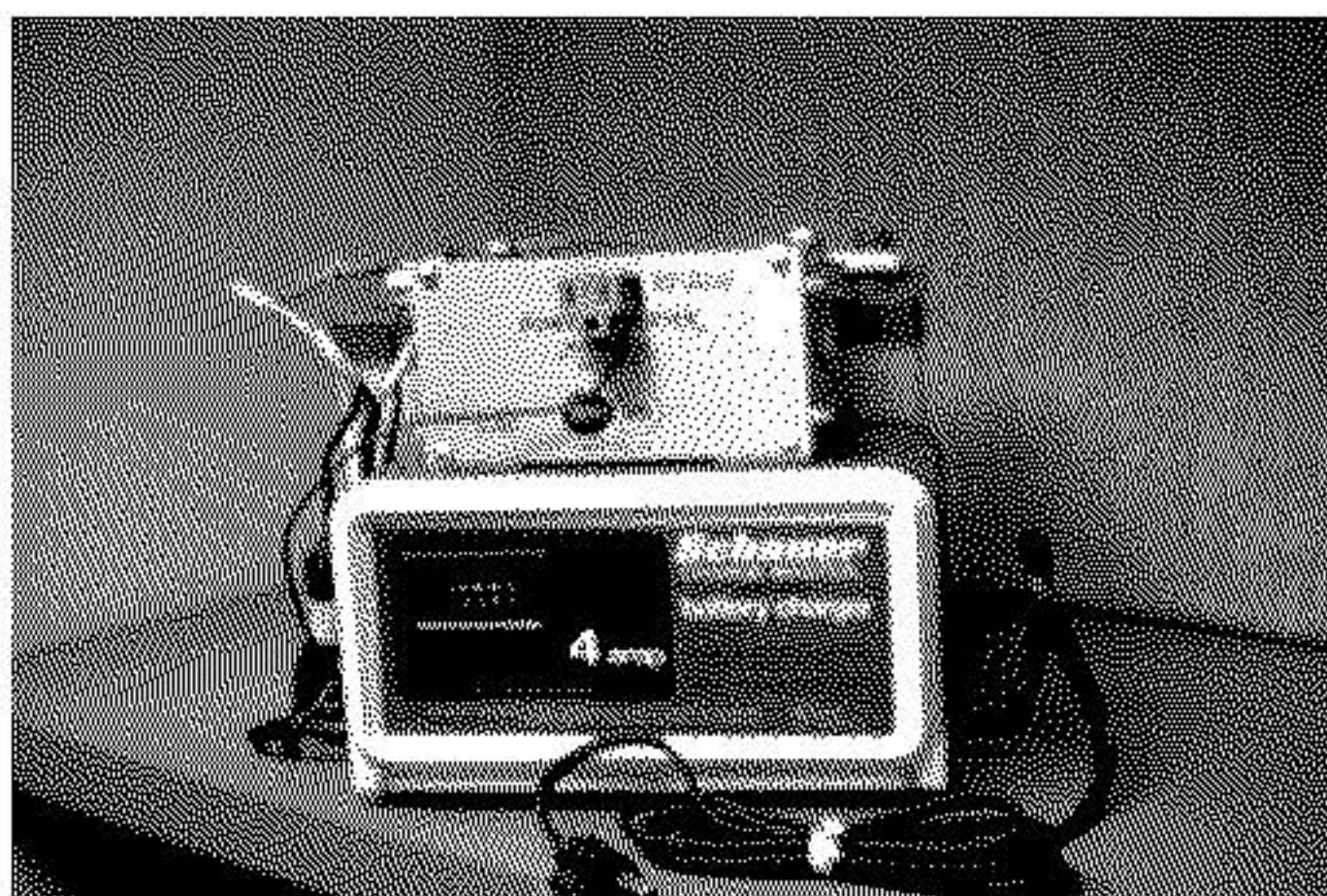
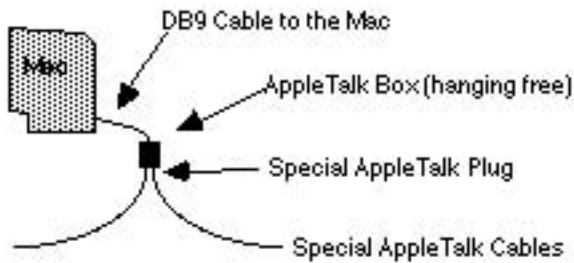
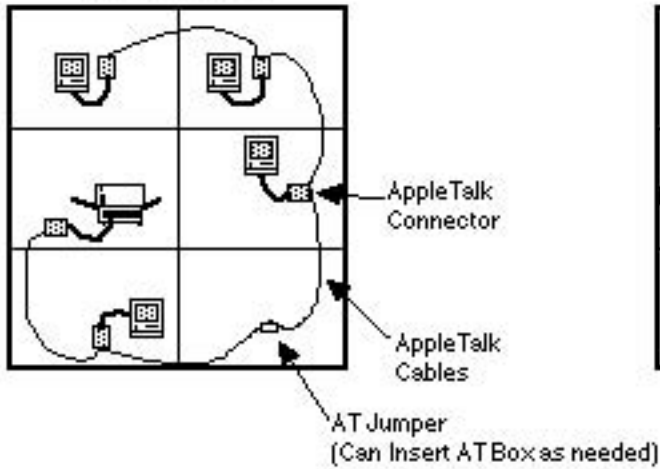


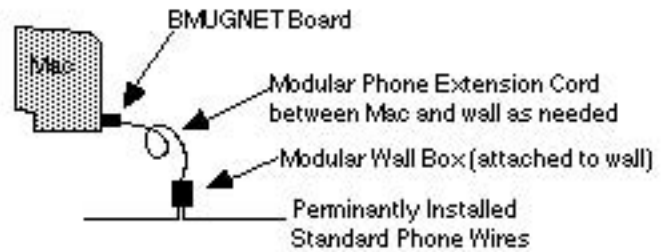
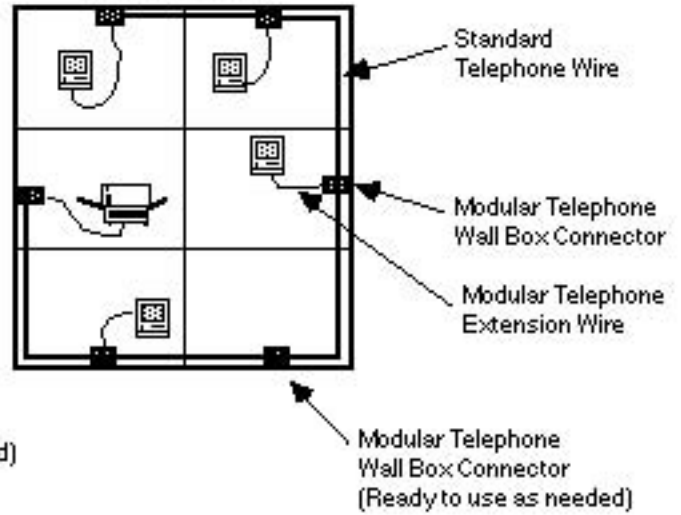
Figure 10. The finished project, ready to power the Macintosh! The battery is in back.

Laying Out AppleTalk and BMUGNET in a Building

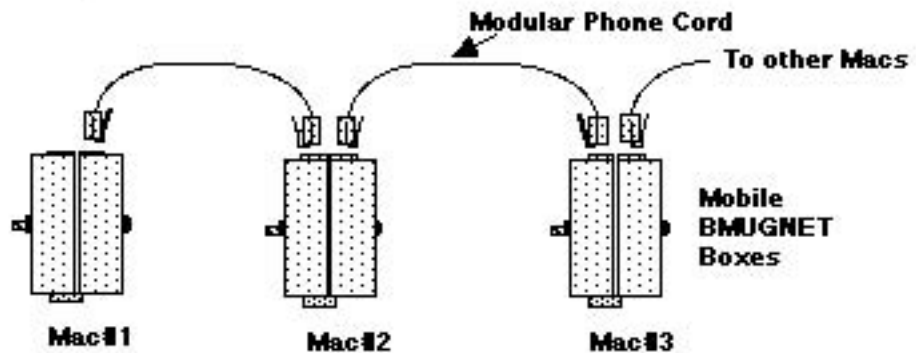
Standard AppleTalk Installation



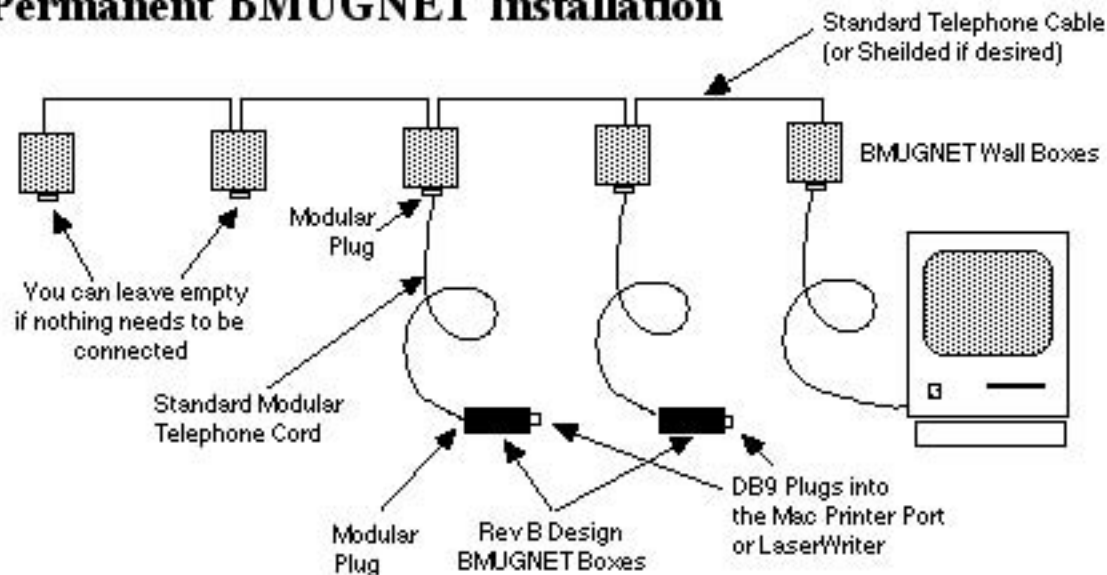
Permanent BMUGNET Installation



Temporary BMUGNET Setup

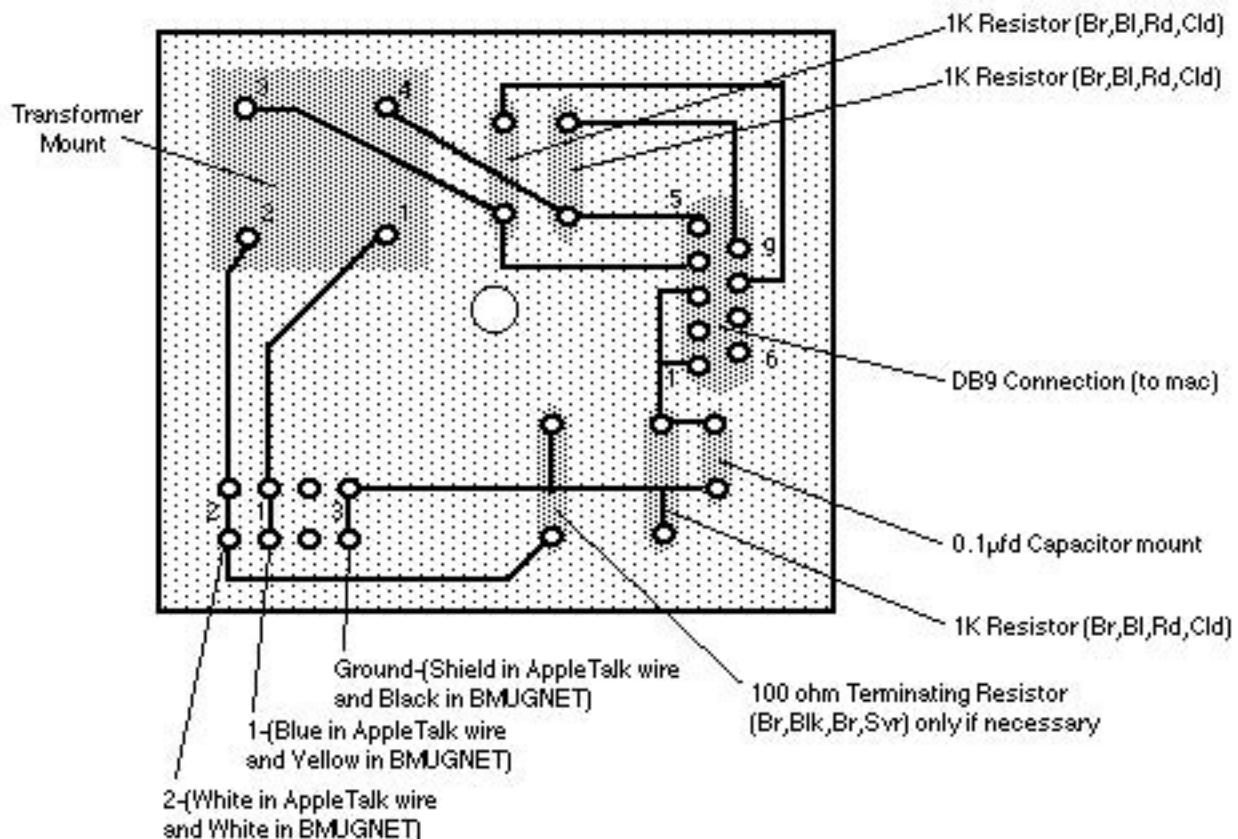


Permanent BMUGNET Installation

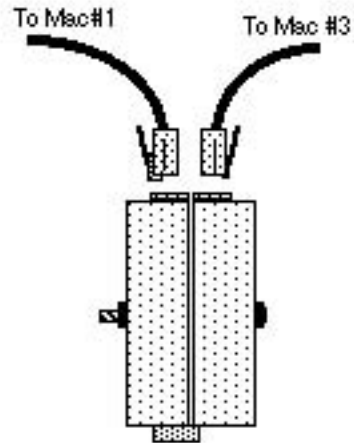


Rev A Design

(for Temporary Installations, although it can be used in permanent installations)

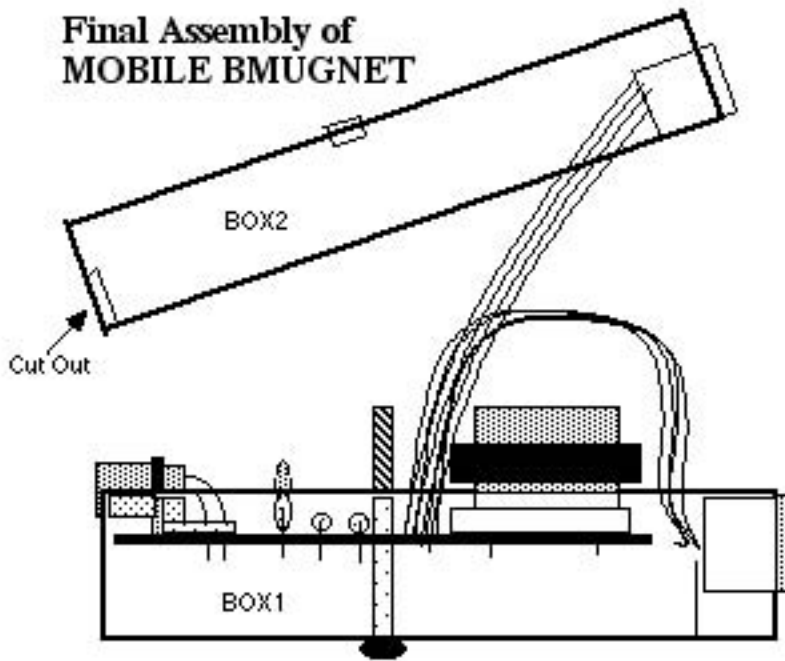


Temporary Modular Phone Cable BMUGNET Configuration

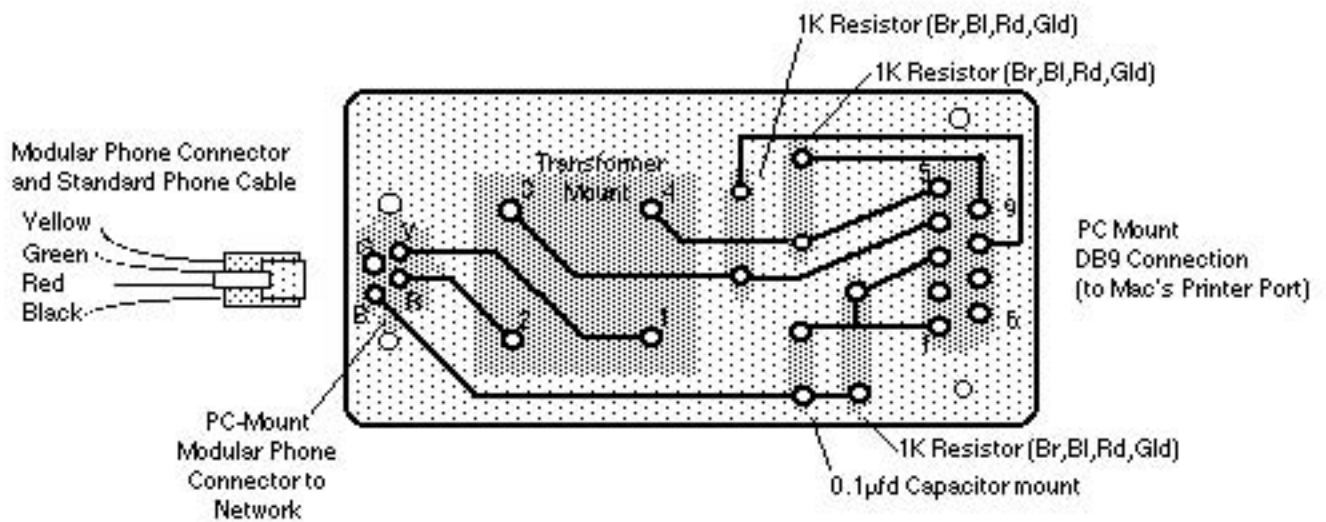


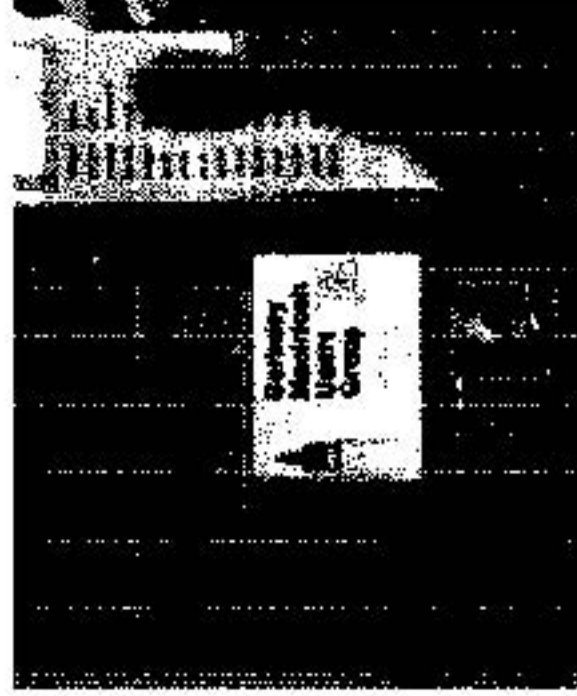
DB9 Connector
Plugs directly into the
Printer Port on the Mac
or connect to with
a RadioShack JS Cable

Final Assembly of MOBILE BMUGNET

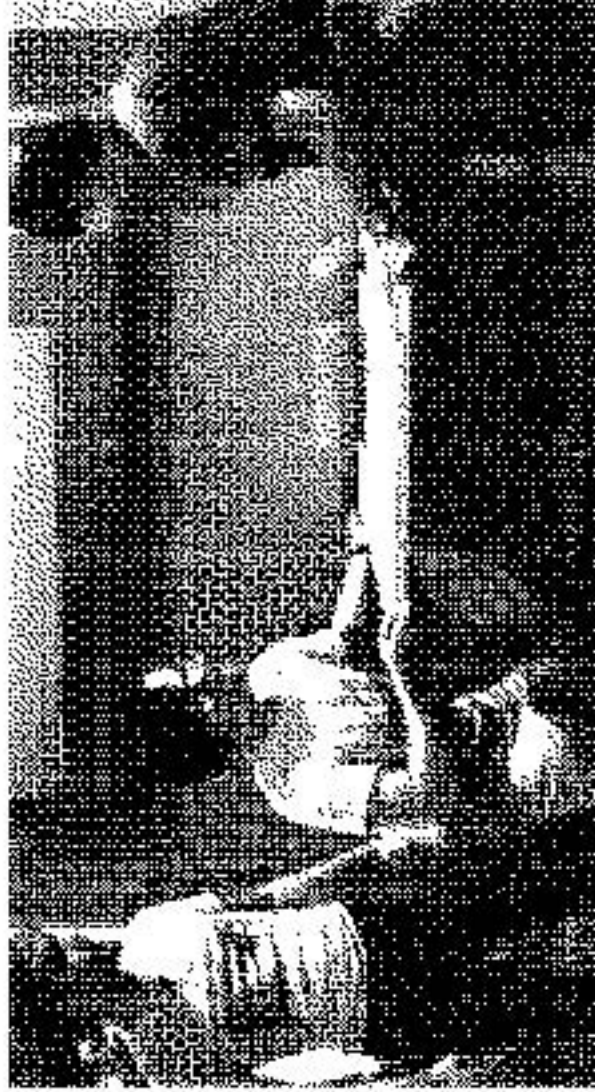


Rev B Design



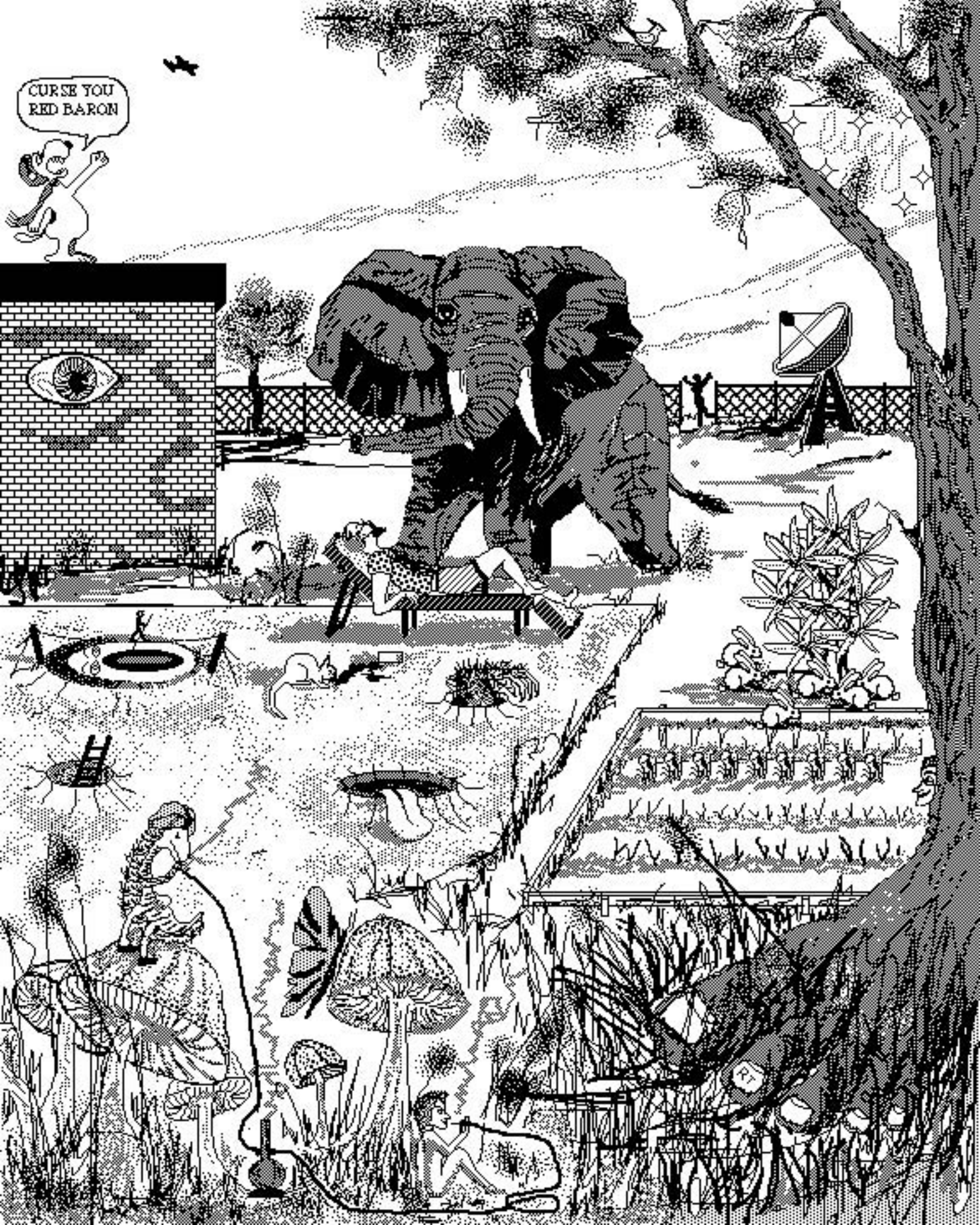


*The world's biggest Macintosh Disk, created by Don
Wood, Carol Cohen, and Elaine Cohen, sets against
the world's biggest Macintosh (the 13-inch monitor)
that Apple sees at computer shows) at the 1985
Boston Mac World Expo.*



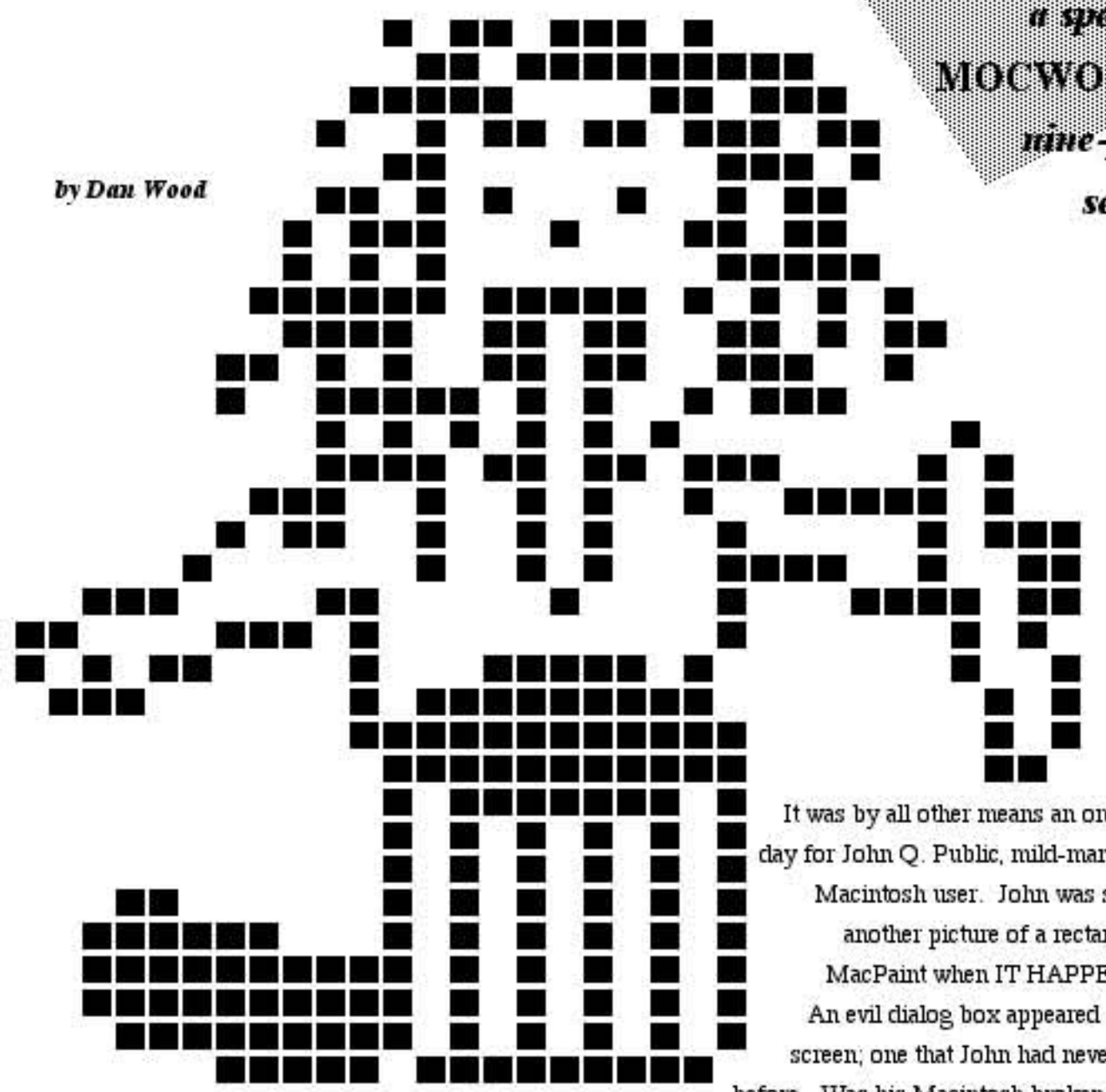
(L-R) Don Wood, James Cohen, and Kelley Weeks attempt to boot a 13-foot Macintosh with the great EMMUC disk at the 1985 New World Expo in Boston

CURSE YOU
RED BARON



First in a special MOCWORLD nine-part series

by Dan Wood



It was by all other means an ordinary day for John Q. Public, mild-mannered Macintosh user. John was saving another picture of a rectangle in MacPaint when IT HAPPENED. An evil dialog box appeared on the screen; one that John had never seen before. Was his Macintosh broken? Was

it time to take it in to the ASUC store for repairs? Or was something else in order? Suddenly, a BMUG newsletter flew through the closed window next to his desk, landing with this page facing upright. Follow Mr. Public as he learns the facts about...

Getting the most out of your trash can

Copyright 1985 by Dan Wood, Berkeley Macintosh Users Group

© 1985 Dan Wood/BMUG

The Problem

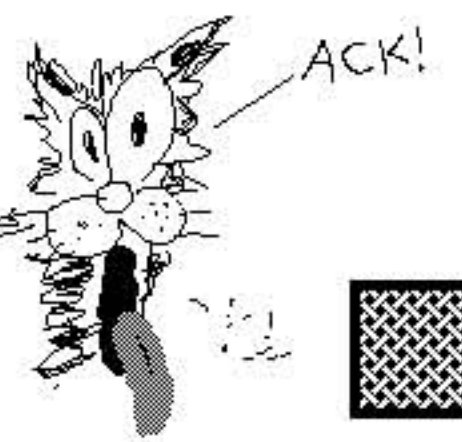
What Mr. Public failed to realize was that you can only fit so many pictures of rectangles and Madonna on a floppy disk before it starts to get full. The reason is simple: any picture, as simple as a single dot or as complex and beautiful as a Madonna masterpiece, when saved to the Apple 3.5" Floppy disk drives, takes up space on the disk. When a few pictures are saved onto the disk, no problem seems to occur. This is a rather misleading feature of the Macintosh, since it creates the illusion that there is nearly an infinite amount of storage space on the disk. And then, suddenly and without warning, the disk will be too full! This is the problem that our friend Johnny experienced. You may have run across this problem yourself. You may have felt the anguish that John felt, then again, you may have felt worse. But cheer up! Help is on the way.

The Solution

Actually there are a number of solutions to this dilemma, but we will be focusing on just one of these today. The only requirements are that you, as a Macintosh user, are willing to part with some of the files that you have

You may have felt the anguish that John felt; then again, you may have felt worse.

created. Of course, this may not always be difficult. For instance, some of your pictures may not be very pleasant to look at (figure 1-1) or are so



Figures 1-1 and 1-2 Some pictures you might draw with MacPaint are either so ugly or so ordinary that you really ought to get rid of them.



Different Looks For The Trash

Most of the time the picture you see will look like a garbage can. (See figure 2-1, with an enlargement in figure 3.) Sometimes, however, you may be greeted by a not-so-friendly face. If you are running a prerelease version of the Macintosh finder, you should be ashamed of



Figure 1-3 and 1-4 An example of a picture you would never, ever want to throw away.



Figures 2-1 through 2-3 Three of the many possible pictures for the trash: Ordinary (left), Grim Reaper (center), Gene Simmons (right).

Your Actions are important

Before you can get rid of these files, you must know how to do so. In order to know how to do it, you must know what to do. By looking at your screen and by reading this you will know what to do, so that you will know how to do it, so that you can do it. Notice in the lower right corner of the screen there is a little picture. This picture — usually a garbage can — is the key to your success.

There are billions and billions of possibilities for the appearance of the trash.

with anticipation (figure 2-2). Or, you may see something else entirely. Since the picture can be customized by advanced Mac Users, there are billions and billions of possibilities for the appearance of the trash.



Figure 3 A Fatbits image of the ordinary trash.

The First Step

Now that you are ready to face the possibility of losing some of the files you have created, you are ready to quit MacPaint and use the Macintosh Finder. If this sentence did not make any sense, you should:
• Stop reading this article, or
• Run the "Guided Tour" disk and cassette tape supplied with your Mac, or
• Read all the past issues of Moc World.
By the time you have returned to your desk with a cup of coffee, the finder should be ready for your use. Your screen will look something like figure 4, with a few notable changes:
• The actual location of the arrow pointer may be different on your screen.
• The name of your disk may be different



Figure 4 A typical screen in the Finder showing several files.

- The name of your files may be different
- The arrangement of your files on the desktop may be different
- The background pattern on your desktop may be different if you have been fooling around with the "Control Panel."
- Everything else looks different. This is, after all, a professional magazine: how can you possibly expect what you do to look as slick as what we do? At this time, take a deep breath. You are about to eliminate some of your files.

Throwing it away

Using the standard Macintosh methods, just drag the files you do not want into the trash. Your arrow pointer must be positioned over the trash icon; otherwise you will just be depositing the files on the desktop near the trash can (figure 5). The trash icon will inverse itself ("open up") when you have touched it. Let go of the mouse button, and voila! It's in the can!

Dan Wood is a no-good bum who likes to be mean to slick magazines.

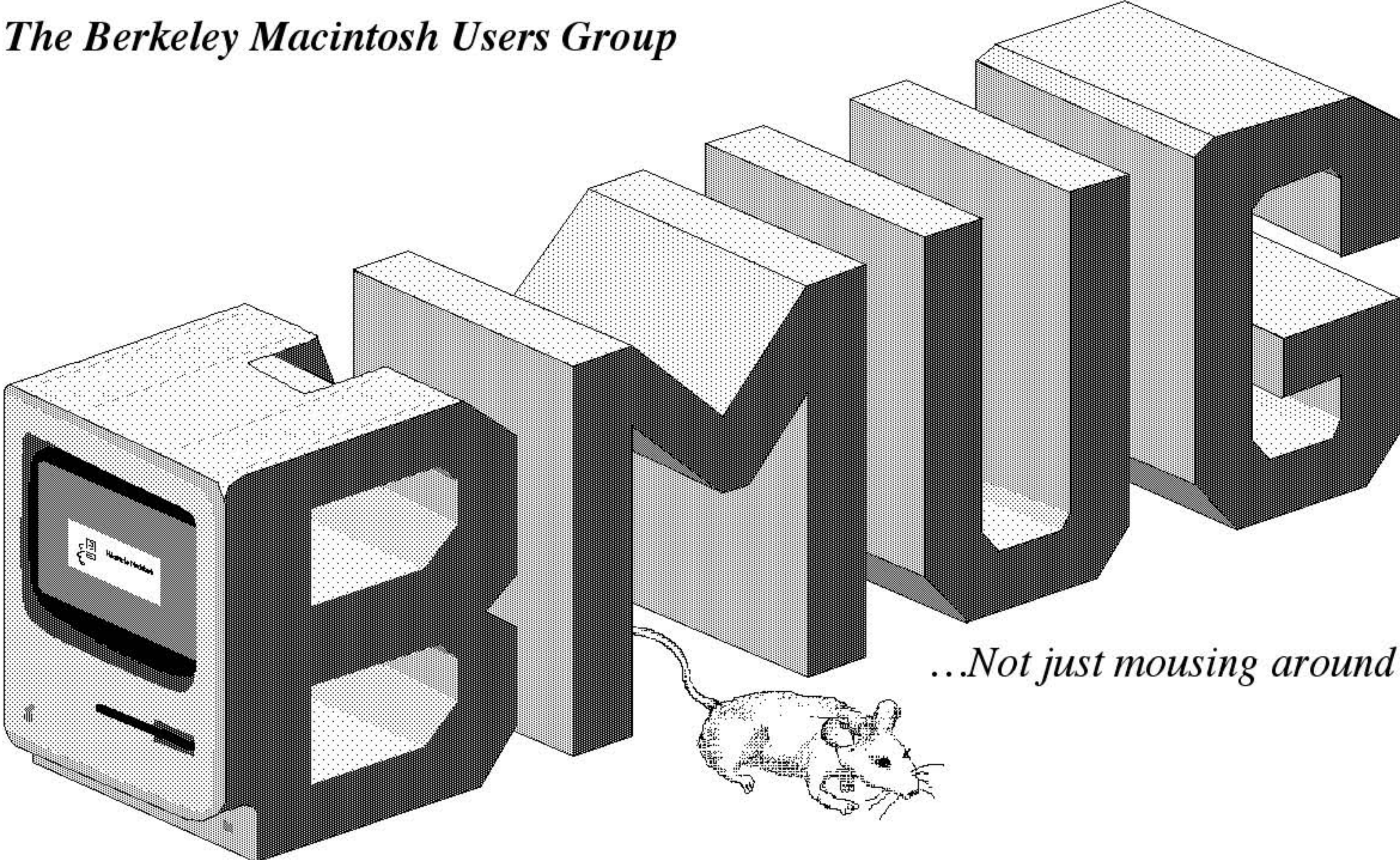


Figure 5 The wrong way to throw things away

In part 2: Emptying the trash can with a minimum of odor

© 1985 Dan Wood/BMUG

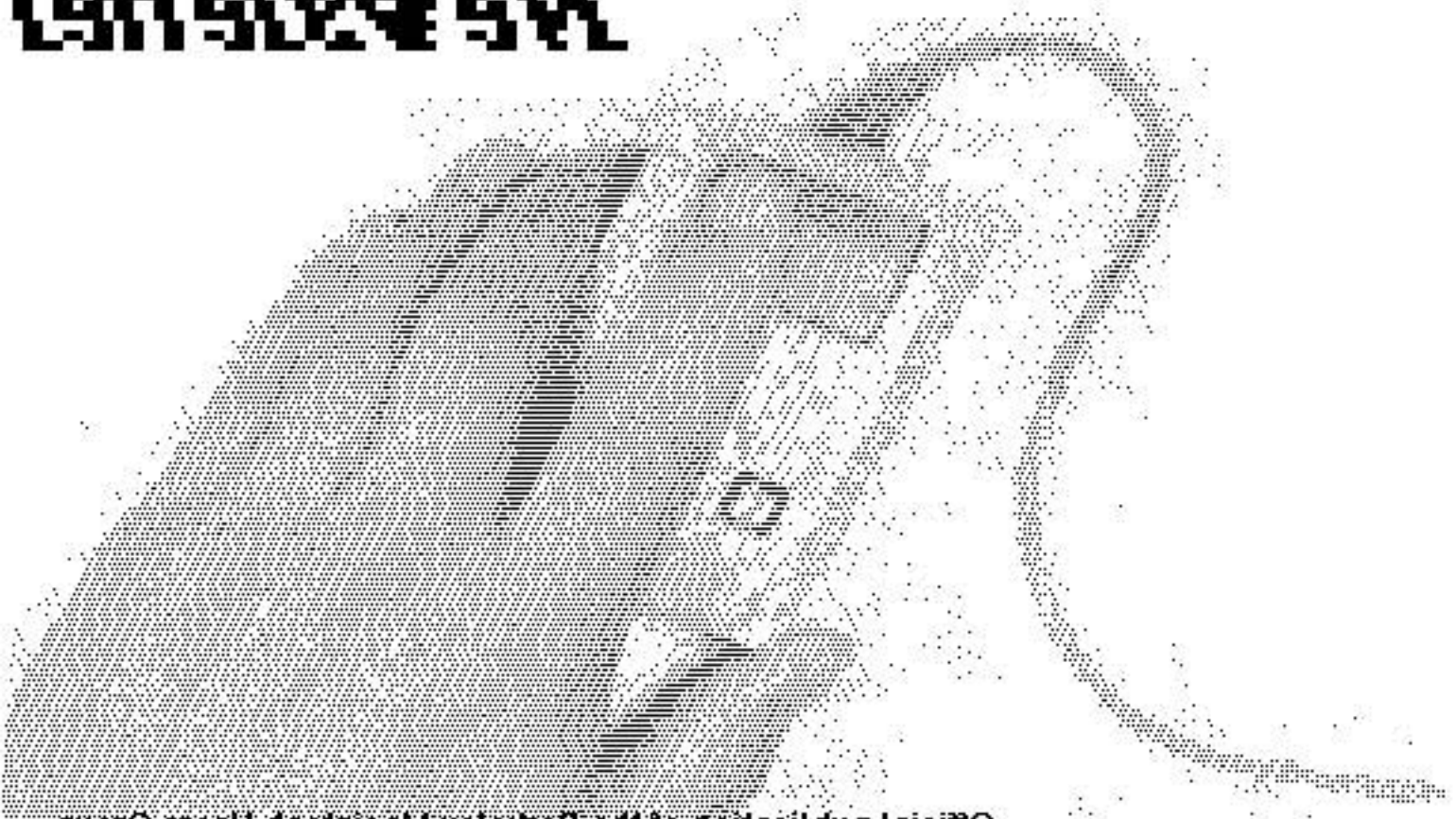
The Berkeley Macintosh Users Group



...Not just mousing around

Copyright 1985 Dan Wood

THE MACHINE





televised
technical
event
quora