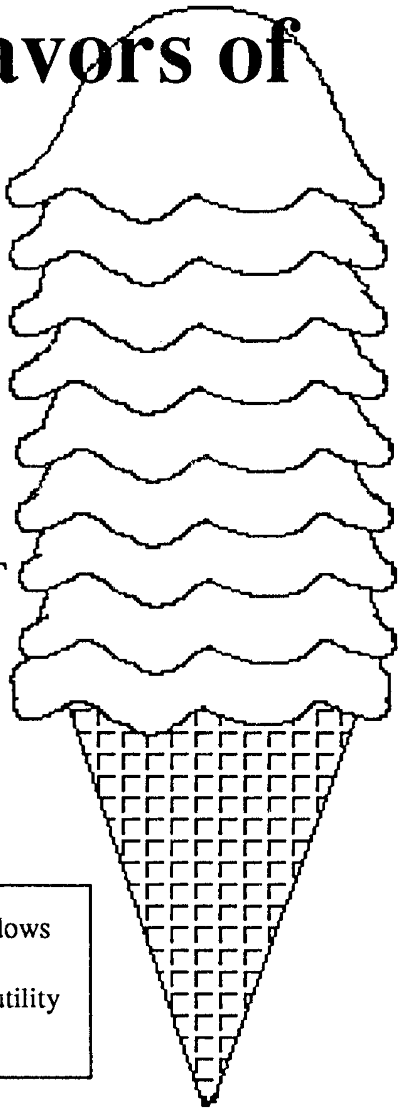
the world of 68' micros

Support for Motorola Processors

November 1994 *Vol. 2 Number 3* *$4.50 Canada, $4.00 US*

Nine flavors of OS-9...

6809
PT68K
Hazelwood
MM/1
AT306
Amiga 500/1000
Atari ST & MegaST
Macintosh
OS-9000



⌘ Ed Gresick on G-Windows

⌘ Great new DECB file utility supports wildcards!

CONTENTS

SDSK512K is 7 pages! Due to the size of the article and program listing, a new Tandy Color Basic support column will not debut until the next issue. As long as the material comes in, there will be at least one Basic listing in each future issue. Have a great Thanksgiving (and don't eat to much)! [illegible]

POSTMASTER:
If undeliverable return to:
FARNA *Systems* PB
Box 321
Warner Robins, GA 31099

Address Correction Requested

the world of 68' micros

Published by:
FARNA Systems
P.O. Box 321
Warner Robins, GA 31099-0321

Editor: F. G. Swygert

Subscriptions:
$23/year (8 issues) US; $30/year for Canada/Mexico ($12 US, $16 C/M for six months- four issues). Overseas $35/year ($18 for four issues) surface. Add $8/year, $4 six months for air mail.
microdisk: $40 per year, $21 six months, or $6 per issue. Overseas add $10/year, $5/six months, $1/single issue for air mail delivery. microdisk contains programs and source listings from each magazine; not stand-alone.

Advertising Rates:
$15 1/6 page, $20 1/4 page, $35 1/2 page, $60 full page, copy ready. Add $10 for special placement, $10 for typesetting ($5 1/4 or less). Dot matrix will be typeset if deemed unacceptable and submitter billed. 10% discount for four or more appearances.

All trademarks/names property of their respective owners.

The publisher welcomes any and all contributions. Submission constitutes warranty on part of the author that the work is original and not copywritten by another party. All opinions expressed herein are those of the individual writers, not necessarily the publisher or editor. FARNA Systems reserves the right to edit or reject any submitted material without explanation. Renumeration discussed on an individual basis.

Back issues are $4 per copy. Overseas add $1 each for airmail delivery.

Newsstand/bulk orders available. Dealers should contact the publisher for details.

Problems with delivery, change of address, subscriptions, or advertisers should be sent to the publisher with a short description.

The publisher is available for comment via **Internet** (dsrtfox@Delphi.com) or **Delphi** E-mail (DSRTFOX). The CoCo and OS-9 SIGs on Delphi are also frequented (The Delphi SIGs are still sponsored by Falsoft).

The editor speaks...

F.G. Swygert

I'm finally getting caught up on rest and "home work" after this year's Atlanta CoCoFest. It's hard to believe we've been doing this for five years already! And there will be continuing fests for the next two or three years. After that, who knows?

Attendance at both CoCoFests is steadily declining. Worse, from a vendors point of view, is that pretty much the same people are coming every year. This means vendors have to have something new to sell or upgrade to every show. I would guess that between 200 and 250 attended this years Atlanta fest, which is a decent amount of people.

But there are many good things about the CoCoFest! One is the fact that the same staunch supporters keep returning, even though I did mention that as a negative point in the previous paragraph. If it weren't for the fests, I wouldn't get to see some of these people! It is nice to see those who I've been chatting away with on Delphi, FIDO, and Internet. Some have become real friends that I'd miss seeing. And we ALWAYS have a great time! Getting to see new items demonstrated, the ability to ask questions, pick up some spare or replacement equipment, and to meet all the vendors are other pluses of attending a CoCo Fest.

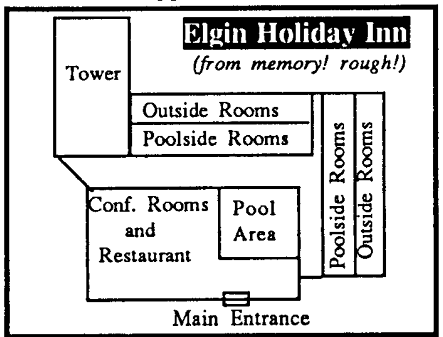
You will notice an advertisement for the next Chicago CoCoFest in this issue. Start making plans to attend NOW! Schedule your time off, save your money, get those items you need to sell together, and whatever else you may need to do to attend! Chicago is the oldest fest, and is still THE fest to attend. If you can only get away once every year, make it Chicago!

One of the best things about the Chicago fest is that the Holiday Inn where it is held is a Holidome recreation center. There is an indoor swimming pool and public jacuzzi along with an excercise room equipped with stationary bicycles and weight lifting machines.

But the best thing is that rooms have poolside entrances on two sides of the pool/lobby area, which is huge! This is very conducive to socializing after the fest. Several groups gather in this large area for impromptu discussions. Need to find someone? Go up on the balcony and look around! If they're in the lobby area, you can easily see them!

If you want a poolside room, make reservations early! If you can't get poolside, you'll most likely get a tower room. The tower is five stories of rooms located on one end of the hotel. It isn't bad, and has easy inside access to the fest area. If you make reservations to late, you'll end up in an outside room, meaning you'll have to walk around outside the building to a main entrance. Chris Hawks (of HawkSoft) says you definately would rather have a tower room... he had an outside room last year!

So make plans to attend! As long as attendance is over 150 or so each year, there will continue to be CoCo/OS-9 fests (I've been getting onto the fest planners about this... around 25% is OS-9/68000, and this needs publicizing!). The only way the fests will continue, and the CoCo and OS-9, is with your continued support! *< 268'm >*



~~PROGRAMMING CONTEST!!!~~

FARNA Systems has set aside some cash and prizes ($150 total!) for a programming contest!

ALL COMPUTER TYPES SUPPORTED BY 68'micros ARE ELIGIBLE!

Send a disk with a running copy of the program as well as an ASCII listing of the source code or BASIC listing, running/installing instructions, and a descriptionto FARNA Systems PC, Box 321, WR, GA 31099 by March 1st, 1994. Programs may be of any type.

First place gets $50 cash, Second place gets $25!

One of the greates services you provide is the advertising contained in your magazine. You provide an important link between the vendors of software and hardware and owners of CoCos and later generation 68's.

There have been good articles like those on the CoCo Fests, CoCo Hard Drives, and on learning how to use different operating systems and programming languages. Occasionally writers tend to let their thoughts wander off their topic and fail to summarize. [illegible] interest to me are articles on other machines except as they relate to the CoCo such as the CoCo II emulator and proposed CoCo III emulator for MS-DOS machines.

My son has an Apple IIc and articles would be appreciated on making null modem cables to connect the CoCo to other computers and on cables to permit sharing of peripherals such as mice, joysticks, printers, and monitors.

You need to receive and publish more letters from readers. I miss the programs published in Rainbow and wish you well on your program contest.

Sincerely,
Charles A. Radatz,
General Manager KTNC Radio

Charles, I chose your letter to be first for a particular reason... I DO need more letters from readers! I wish I'd get them. I publish all that have some general interest or comments pertaining to the majority of subscribers. In fact, I did get a good many this time, and have devoted more than a page to letters in this issue.

When speaking of not summarizing, you couldn't mean me, now could you? I do tend to leave one hanging sometimes. The reason is that the thought process isn't over! I supply information... it is up to you to decide what to do with it. I don't fully close all the time because I don't want to tell you what you should think about what I wrote, I leave that up to each individual. And the series may not be over yet...

Thanks for all the good comments. I'll try to do better on closing articles in the future, and I'm sure I can get the null modem issue handled in a future issue.

In the letters section of vol. 2 no. 3 you made mention of an article on adding 512K to a CoCo2, and a modified OS-9 Level II. I am using a CoCo2 with 64K and OS-9 Level I Ver. 2 for a data logger. I have not finished all of the hardware and am all readyy running out of room for the software. I also have other problems inherent to OS-9 Level I. That could be eliminated by using Level II

Any help on where to find info on putting 512K and Level II on the CoCo2 would save me from putting the CoCo2 back in the closet just to keep the CoCo1 company. I would just use a CoCo3, but since I have the CoCo2 just setting around I thought I would put it to good use.

In your article on the Speech/Sound Pak converstion for OS-9, you mentioned that the "o" connects to pin 17 IC8 and pin 20 IC5. My schematic says this should be connecting to pin 17 IC8 and pin 20 IC11, not IC5. Another way to get the -6V for the op amps is to use an SK3672 (-6V regulator) to cut down the -12V from an MPI. Of course, this only works with an MPI.

I also modified an X-Pad model GT-116 (cat. #26-1196) so that it works with a CoCo3 (hardware only). If you feel there is a need for this information just let me know and I can give an explanation of what is required.

Mike Jenkins
12A South McKinley
Kennewick, WA 99336

Mike, I was under the impression I already had the Level II conversion info, but don't. Let me explain. A couple years ago, the European OS-9 Users Group (EFFO) came to the Chicago CoCo Fest. They were selling complete sets of their previouos newsletters on disk to help finance their trip. I had heard about the 512K/Level II conversion for a Dragon and inquired about it of the European group. I was told there was an article on the newsletter disks, so bought a set, with the understanding that I would also have reprinting permission of all contents. The disks were all archived, so I only looked through a few at a time. Recently, I finished going un-archiving all of the disks. There was only a mention of the conversion, no article! I have contacted a person in Germany for assistance in finding the info, and will soon have another contact in England who may be able to help. As soon as I get the info, I'll be printing it!!

In the meantime, you might consider writing an "in progress" article describing what you have managed to accomplish and what you need further assistance on. I'm sure readers would be interested and willing to help. A lot was done with Level I, so I'm sure your program obstacles can be overcome without simply throing more memory at it.

I have been told that there were two versions of the Speech/Sound cartridge. I'll check my sources and see if I made a mistake or your info is for another revision.

I would really like to see the info on converting the X-Pad, and also what kind of software you use it with, and/or programming examples. I did have a couple of those things that I had a hard time getting rid of. Now maybe I shouldn't have...

I was particularly interested in Chris Perrault's comments in *Letters*, Sept. 94. For quite a few years now I have been using the IBM compatible, but the CoCo is still at my beck and call at the click of a switch. And in some ways, it still surpasses the high clock speed of the clone. Like Chris, I'll not likely abandon the Coco, while continually expanding the compatible.

Arthur S. Hallock
Route 1 Box 198HHH
Deming, NM 88030

Arthur, you echo my sentiments exactly! Text based applications run nearly as fast on the CoCo as a clone. I too use a clone for some tasks, such as producing this magazine, but the CoCo for others, such as basic text processing, communications (except when at work or I need a compatible program.. I do a lot of my telecom at work these days because I backup a mini-computer at night, and a clone and modem are handy while waiting for backup tapes to run. When at home I use the CoCo), and my business records. No clone spreadsheet is any more efficient than DynaCalc, no address label program better than Bob van der Poel's DML9. Besides, I can run both of those at once with no problem!

Your celebratory August 94 issue has provided good reading. So much so that I'm compelled to pen a few comments about your survey results.

It seems that this data helps to profile your "typical reader". I think the most obvious observation we can draw is that most readers own two or more systems, perhaps 8 out of 10. While CoCo 3 users may be "typical" at 57%, those also using Intel clones aren't far behind (50%). Delmar's System IV/V has apparently won the "CoCo 4" honors with a 10% share, the rest of the 68K machines combined accounting for an additional 15%.

continued on page 17

Letters are printed on a space available and popular subject matter basis. If you don't want your letter printed, or wish to withhold your address and/or name, please state so when writing. In some cases, letters are edited for space and/or clarity. If a personal reply is desired, please enclose an SASE.

Nine Flavors of OS-9

F.G. Swygert

The most widely known versions of OS-9 useful to the hobbyist.

There are many more than nine flavors of OS-9. To many to comment on here! This article will cover the versions of most interest to the OS-9 hobbyist. These are:

6809 (mostly Tandy Color Computer)
68000:
- AT306
- PT68K/CDS680x0
- MM/1
- Hazelwood
- Atari ST
- Amiga
- Macintosh

OS-9000 (68020+, 80386+, PowerPC)

6809 and the CoCo

I'll start with the first version of OS-9 and a little history. In the mid 1970s, Ken Kaplan was a student at Drake University in Des Moines, Iowa. He and a friend received a research grant to work with an experimental version of Motorola's first generation 6800 microprocessor. This eventually lead to a ROMable real-time operating system called RT/68. RT/68 became the first product offered by Microware, and established them as a real-time operating system provider early in the computer age.

Since Motorola had worked with Microware earlier, they commisioned them to develop an advanced programming language for the then new (1978) 6809 processor. At the same time, Microware developed the OS-9 operating system to go along with the new language, Basic09. Versions of OS-9 were written for several SS-50 bus computers (Gimix, Smoke Signal, etc.) as well as Motorola's ExorBus systems.

An interesting side note appears when discussing OS-9 for the Tandy Color Computer, the only mass-market system to ever run OS-9. Ken Kaplan contacted Frank Hogg Labs about porting OS-9 to the CoCo before an agreement was reached with Tandy. Ken and Frank purportedly had a "gentleman's agreement" that FHL would port OS-9 to the CoCo. After many months of work, Frank received a phone call from Ken with the news that an agreement had been reached with Tandy instead. FHL used their advance knowledge of OS-9 to come out with som of the first CoCo OS-9 software.

OS-9/68000

In 1983 OS-9/68000 was released. This was developed with assistance from Motorola for the new Motorola VMEbus systems. Basic09 was also ported, but under the name Microware Basic.

OS-9/68000 is currently available as either "Industrial" or "Professional". Industrial OS-9 consists of the main kernel and serial I/O only, and is intended for embedded systems (microcontrollers). Professional is intended as a multi-user development system including everything in industrial plus a file system and tape support, a full range of utilities, and the Microware C compiler. An earlier "Personal" OS-9 package intended for hobbyists has been discontinued.

32 bit processors require OS-9/68020. This version comes with a different kernel and enhanced C compiler to allow access to the more powerful 32 bit instruction set. The correct kernel must be used for the host CPU, but there is no difference for application software. The OS-9/68020 versions cost about 50% more than the 68000 version.

AT306

The AT306 is a new motherboard designed especially for the hobby market by Kevin Pease and Carl Kreider. In fact, this board isn't readily available at this time. It should become available by December 1994.

With the demise of "personal" OS-9 from Microware, the cost of the operating system became a major obstacle. Low cost boards (such as the PT68K) are available, but Professional OS-9 for these boards is the major cost of a system, usually being at least 1/3 of the total cost of the entire system.

Personal OS-9 V3.0 for the AT306 is not a direct product of Microware. It is Industrial OS-9 with the individually licencsed SCF, RBF, PipeMan, and RamDisk file managers with several utilities from Microware. Other necessary utilities are either public domain or written by Carl himself. A version of MGR (a graphical user interface popular in Europe) and BASIC (similar to Microware BASIC) will also be included with the package. Device drivers were also written by Carl Kreider.

The board is based on an MC68306 microcontroller. This chip is a 68000 core with all necessary support and I/O to make a compact computer using a minimum amount of parts and board space. The chip runs at 16.67MHz, has two serial ports, a 16 bit timer/counter, 16 programmable parallel I/O lines, and a DRAM controller onboard. A PC type integrated I/O chip is used to provide all necessary I/O support .

The board sports 4 SIMM sockets (1-16MB), an IDE hard disk interface, up to 1.44MB floppy disk interface, two serial ports (up to 115K baud), a bi-directional parallel port, real-time clock, and six 16 bit PC/AT type expansion slots. Thses slots currently support a Tseng VGA card and two SCSI I/O interface cards (Future Domain 1680 and Adaptec AAH 15xx). Other cards will be supported in the future.

The target retail price for the board and "Personal OS-9 V3.0" is $400.00. At this point several vendors are planning on selling the boards. We'll let you know when more concrete info is available.

PT68K & CDS680x0

Peripheral Technologies' PT68K series is perhaps the most well known of all 68000 based "hobby" computers. In the mid eighties, Popular Electronics Magazine ran a series of articles on how to build your own 68000 based computer. This was Fred Brown's PT68K These boards had an 8MHz 68000, low density (up to 720K) floppy controller integrated into the motherboard, and PC/XT eight bit expansion slots. This allowed the use of readily available components such as display adapters and hard drive controllers. Unknown to many, this board had already been in use in the industrial market.

Currently, the PT68K2 (10MHz) and PT68K4 (16MHz) boards are available. A brand new K4 is around $250, while a recertified K2 is about $125. Professional OS-9 V2.3 is $300 for either board. Supported cards include internal modems, an IDE interface, mono/ CGA/VGA display cards, and MFM/RLL hard drive interfaces. Specific brands of cards are required except in the case of MFM/RLL and mono/CGA cards (combination cards aren't supported).

In the early 90s, Fred Brown released a 68020 based computer using the PC/AT 16 bit expansion bus. A first for OS-9 computers was the use of a daughterboard for the processor. This makes the motherboard upgradable to more powerful 680x0 series processors in the future. The 68020 board was released under a separate company name, Computer Design Services, rather than Peripheral Technologies.

If one doesn't want to build their own system, complete, ready to run systems are available from Delmar Co. Ed Gresick has been building complete PT68K based systems for the industrial and hobby markets for some time now. Delmar currently offers the System IV (PT68K4) and System V (CDS680x0). Delmar currently ships OS-9 V3.0 with their systems.

Peripheral Technologies
1250 E. Piedmont Rd. Tel: 404-973-2156
Marietta, GA 30062 Fax: 404-973-2170

Delmar Co. Tel: 302-378-2555
Box 78 Fax: 302-378-2556
Middletown, DE 19709

MM/1

The MM/1 started out as a dream. It was envisioned to replace the CoCo while maintianing some compatibility with existing software, and to have the hardware capabilities to capitalize on the infantile multi-media market (hence the name "MM/1"). A committee was set up to develop the machine. This led to a lot of compromises and late deliveries, which ultimately gave the machine a bad reputation.

Unique to the MM/1 is its windowing system. K-Windows was designed by OS-9 programmer Kevin Darling. It is a multi-window graphical user interface. It was specifically designed to be similar to the CoCo 3 OS-9 Level II windowing system. This was to make the learning curve for the system short for CoCo programmers moving up to the MM/1.

Due to the lack of promised CoCo compatibility, late deliveries, and lack of good manufacturer support, the MM/1 "died" in 1991. Even though there were some design compromises, the MM/1 did find a great deal of support from CoCo OS-9 users. Many of these people continued to use their MM/1s even though they had little support.

In 1993 an enterprising MM/1 enthusiast managed to acquire production rights to the MM/1 along with access to some old stock. David Graham then formed BlackHawk Enterprises, Inc., to produce and market the machine. The MM/1 is currently available from bare boards to completely assembled, ready to run systems. Professional OS-9 V2.3 comes with all boards and systems.

BlackHawk Enterprises, Inc.
Box 10552 Tel: 405-234-2347
Enid, OK 73706-0552

Hazelwood

Mike Hazelwood has been designing 6809 (SS-50 bus) and 680x0 based motherboards for some time. His current boards are based on the 68020 and 68030 processors. These are available from Frank Hogg Labs as the KiX/20 and KiX/30, respectively. Previous FHL computers were also based on Hazelwood designs (such as the TC70).

The main importance of the KiX systems is that they are full 32 bit machines. All the other designs mentioned are 16 bit (except for the CDS680x0, which uses a 32 bit processor and 16 bit expansion bus). This makes them a good bit faster (and expensive) than most of the others in this article.

The KiX boards sport onboard SCSI controllers, floppy controllers (up to 1.44MB on 20, 2.88MB on 30), four serial ports, parallel port, and SIMM memory sockets (up to 16MB on 20, 64MB on 30), and Professional OS-9 V2.4. The 20 has a single 32 bit expansion port, the 30 has six.

In experiments using standard VGA boards and chip sets, Mike and Frank discovered that they were just not fast enough for the KiX/30. In fact, when the machine was set to display video as fast as possible, a pure white screen resulted. After looking into this, it was discovered items on the screen were being displayed so fast that the phosphor didn't have time to fade, thus the white screen! To take advantage of the speed of the Kix/30, the MGA (Multi Graphics Adapter) video board was designed. This board has the high speed video circuitry as well as a keyboard connector. It is needed to turn any KiX system into a high performance graphics workstation.

Frank Hogg Labs, Inc. Tel: 315-469-7364
204 Windemere Road Fax: 315-469-8537
Syracuse, NY 13205

Atari ST and Mega ST

Cumana Ltd. of England owns the license for the Atari version of Professional OS-9 (V2.3). The standard package is supplied on double sided 3.5" diskettes. A single sided boot disk can be supplied as an optional extra (one double sided drive will be necessary on your system) (*editor:* Atari 520ST systems came with one single sided 3.5", 720K drive)

The screen driver supports monochrome high resolution 25 and 50 lines, medium resolution 25 lines, and color medium or low resolution 50/60Hz. All resolutions have an option for smooth screen scroll.

The keyboard is completely software driven. Any key that produces ASCII code can be programmed with a desired long string. All Atari ports including the mouse and midi ports are supported. No Atari ROM routines are used.

Atari hard drives are supported. Totally Atari compatible drives should be OS-9 compatible but cannot be individually supported.

Caching is incorporated in the floppy disk driver to increase data throughput, and multi-sector "read and write" is implemented in the hard disk driver (as an alternative to caching) to ensure data integrity.

The following programs for software development and immediate use are included:
Sculptor - Fourth generation database and programming language.
Stylograph - Powerful word processor with mail-merge and spell-check.
DynaCalc - Powerful electronic spreadsheet.
C Compiler - K&R standard, UNIX and OS-9 compatible libraries. Generates position independant, re-entrant, ROMable code.
BASIC - Enhanced structured BASIC with interactive compiler.
uMACS - screen oriented text editor.
Assembler, Debugger, and Linker - full featured relocating macro assembler.
Price - 99.95 pounds sterling, includes UK taxes and shipping to US (approx. $150 US).

If one has an Atari ST series computer, or can find a used one at a reasonable price, then consider Cumana's version. This is truly one of the most affordable systems available, especially when all the software is considered.

Cumana Ltd.
Pines Trading Estate, Broad Street
Guildford, Surrey, GU3 3BH
Tel: (0483) 503121
Fax: (0483) 451371
Ask for Yvonne Kavanaugh (sales)
NOTE: This is an older Cumana product. Technical support is therefore limited.

There is also another port of OS-9 for the Atari. This port is owned by Dr. R. Keil of Germany. No details were available by press time except an address:

Dr. R. Keil GmbH
Gerhart-Hauptmann-Str. 30
D-6915 Dossenheim
Tel. +49 6221 86 20 91
Fax. +49 6221 86 19 54

Amiga 500/1000

The Amiga version is based on Professional OS-9/68000 V2.4. The price is US$600.00. Tesseract uses the system in house on A1000, A2000 and A3000 (experimental 68020 version-- not available). It has also been used at Microware on A500s.

The system takes over the Amiga completely so it retains its realtime qualities and performs like a very standard OS-9 environment. It requires no ROM replacement or other hardware modification, so switching between OS-9 and AmigaDOS (or any other operating system) requires only a re-boot.

The Amiga display is supported as a color text terminal with customizable emulation and font module support. It also supports multiple virtual screens much like that provided by Amiga Unix. A graphical environment is not currently available.

Tesseract is updating their licence to include the new Microware 'Ultra C' compiler. This will add about US$150.00 to the price. If purchased separately from Microware, Ultra C will be priced at US$1,300. This would seem like a good buy for professional users, but does increase the entry cost for hobyists.The older version with K&R C is still available for now.

Booting:
Can boot from floppy or hard disk. OS-9 Kickstart may be used on A1000 to gain an extra 256K of write-protectable ram. Supported by AmigaDOS

Terminals:
VT100 emulation on Amiga display. Full support for Amiga serial port. Driver for ASGD serial card available. Commodore A2232 driver under development
Disks:
Internal floppy drive supported. Primary format is Amiga standard encoding. OS-9 Universal format supported for compatibility. Driver for A2090 HD available. Driver for A2091/A3000/A590 HD available soon (new SCSI driver technology will support SCSI tape, disk, etc.)
Utilities:
AmigaDOS file system access utility. Convert Amiga font files to OS-9 font modules
SSM:
May be used when hardware supported.
Price:
US$600.00 (US$750.00 w/ UltraC)

Digby Tarvin Fax: +61 2 698 8881
Tesseract Pty Ltd
53 George St.
REDFERN, NSW 2016
Australia
Internet: digbyt@extro.ucc.su.oz.au

Macintosh
The Macintosh port of OS-9 is owned by Ultrascience (A division of Gibbs Laboratories). According to their literature, it even allows the Macintosh operating system to run as a process under OS9. The 68000 version is approximately $1000.00, the 68020 and higher version is approximately $1200.00.

Ultrascience Tel: 708-808-9060
Box 847 Fax: 708-808-9061
Wheeling, IL 60090

OS-9000
Microware has released an OS-9 kernel which has been re-coded in C, called OS-9000. Coding the operating system in C does slow it down some, but has the added effect of being made easily portable to almost any processor. It does, however, require a minimum of 68020 Motorolla or 386 Intel.

PowerPC
Just recently, Microware officially announced that PowerPC OS-9 was ready. This is currently packaged as a Developer's Pak for use with Microware's resident and FasTrak (integrated C cross development package for Unix and Windows) development tools. Full optimization is supported for the entire PowerPC line. See "Micro News".

Microware Systems Corp.
1900 NW 114th Street
Des Moines, IA 50325-7077
Tel: 1-800-475-9000
Fax: 515-224-1352
Internet: info@microware.com

< 268'm >

OS-9/OSK Answers! *Joel Mathew Hegberg*

Plumbing with OS-9 (pipes); mono play command for the MM/1.

Last month, you may have noticed pages 15 and 16 reversed (at least it was in my issue). Hope that did not confuse many people. Like Rick Ulland mentioned in his last article, I can be a little sneaky! Rick always has something interesting to talk about. Let me check... yep, Rick's written yet another great column over on page [xx], so be sure to check it out — AFTER you've read my column, of course!

This month, we start out with a letter regarding OS-9/68000 security. I encourage all OS-9/68000 users to set up their systems as multiuser systems, since the software is included. There are many times this benefits you in the long-run.

Hello, 68'micros!

Ulland's 68micro OS-9 security discussion came just as I was thrashing about. Noting references to cc3 I think he was not looking os9/68k. I'm on an MM1 and had began thinking I should protect all the stuff in there.

My original startup opened several windows with preset DIRs. Thinking these would present sneak paths around the login I moved them to a script file to be used after logon. After changing startup file to one incorporating "login" I notice that three CRs after "Username?" gets me back to shell prompt without needing to enter name or password. Of course if I do enter name & password then the sequence shows me the OS-9 welcome field from SYS/motd (where I put a note about the scriptfile to set up my preferred windows). But when login fails and falls thru to original startup shell prompt I can still use my scriptfile (The original shell prompt is $, whereas my password file gives $:). Indeed, both password AND name can be nothing. The system is wide open.

So what have I missed? Is there some way to unboot or unlink a useful file and require rebooting in case the login falls thru? Is there a better login module or patch? I tried the format in OS-9 manual— login [name] [,] [password] . They are indeed optional. ex also seems optional.

My startup file:

```
iniz d0 d1;load runb cls unlink syscall color
more display
loadbuffs
chd /dd;chx /dd/cmds
merge sys/stdfont_01.fnt sys/stdfont_02.fnt
xmode /t0 baud=2400;legi;datn
ex login<>>>/term
```

—Fran Walters

Dear Fran,

Thanks for writing. As it turns out, you are half right. In setting up security on your system, you need to run the "login" program to let the user enter their name and password. As you point out, there needs to be something to re-run the "login" program when it exits after 3 invalid attempts.

The program that does this is "tsmon", which is actually the program you should be running from your startup script file instead of "login". tsmon (time-sharing monitor) automatically runs "login" when the [RETURN] key is pressed in a window or on a terminal. When "login" dies, it waits for another [RETURN] key and runs it again.

Even better, tsmon can monitor more than one window/terminal at a time, making multiple login windows/terminals a breeze! In my startup script file, I open two windows from which I can log in from. The very last line of the file is:

```
ex tsmon -p /term /w1
```

This runs tsmon (without an underlying shell), allowing logins from /term and /w1 devices. tsmon takes care of both devices automatically! And there is no way for someone to drop out of tsmon so the system is rather secure. The option '-p' prints out an "online" message to all specified devices to let users know they can log in.

Hope that helps, Fran! Let me know if you have any other problems.

Ted Jaeger has sent in an example on how to use BASIC's TYPE command to create a new datatype, which makes the creation of graphical objects (in this case, K-Windows) very easy. Instead of a series of PUT commands, using this technique you can merely use a single PUT#1,button command. Ted also notes that this method is much faster as well, requiring fewer i/o calls to the operating system.

The program could easily be ported to C, but with the availability of the cgfx.l graphics library for free, there really is no need. The BASIC "TYPE" command is very similar to the C "typedef struct" concept. My thanks to Ted!

Listing #1: makebutton.bas
```
PROCEDURE makebutton
(* how handy is TYPE !!
(* Ted Jaeger — June 29, 1994

(* here we use TYPE to establish a
(* graphics object—a button— by collecting
(* all its parts into one variable
TYPE buttonparts= startpoint(6), clr1(3),
frame(6), locate(6), clr2(3), shade1(6),
shade2(6), clr3(3), shade3(6), shade4(6),
sync(1):BYTE
(* The "sync" byte is for K-Windows...
(* sometimes it is needed to get K-Windows
(* back in sync. It has a value of $00.
DIM button:buttonparts
DIM downbutton:buttonparts

(* we also use it in a more conventional
(* way—to define a new variable to function
(* as a registrar mask and let us
(* do system calls in BASIC
TYPE registers=d(8),a(8),pc:INTEGER
DIM regs:registers

(* an array to quickly change
(* foreground color
DIM fclr(3):BYTE
fclr(1)=$1b
fclr(2)=$32

DIM char:STRING[1]
DIM callcode:INTEGER

(* now lets load the button variable
(* with the escape sequences to draw it
button.startpoint(1)=$1b
button.startpoint(2)=$40
button.startpoint(3)=$01
button.startpoint(4)=$09
button.startpoint(5)=$00
button.startpoint(6)=$63
button.clr1(1)=$1b
button.clr1(2)=$32
button.clr1(3)=$00
button.frame(1)=$1b
button.frame(2)=$48
button.frame(3)=$01
button.frame(4)=$48
button.frame(5)=$00
button.frame(6)=$73
button.locate(1)=$1b
button.locate(2)=$40
button.locate(3)=$01
button.locate(4)=$47
button.locate(5)=$00
button.locate(6)=$64
button.clr2(1)=$1b
button.clr2(2)=$32
button.clr2(3)=$0f
button.shade1(1)=$1b
button.shade1(2)=$46
button.shade1(3)=$01
button.shade1(4)=$0b
button.shade1(5)=$00
button.shade1(6)=$64
button.shade2(1)=$1b
button.shade2(2)=$46
button.shade2(3)=$01
button.shade2(4)=$0b
button.shade2(5)=$00
button.shade2(6)=$72
button.clr3(1)=$1b
button.clr3(2)=$32
button.clr3(3)=$01
button.shade3(1)=$1b
button.shade3(2)=$46
button.shade3(3)=$01
button.shade3(4)=$47
button.shade3(5)=$00
button.shade3(6)=$72
button.shade4(1)=$1b
button.shade4(2)=$46
button.shade4(3)=$01
button.shade4(4)=$47
button.shade4(5)=$00
button.shade4(6)=$64
button.sync(1)=$00

(* now we have the button constructed
(* lets make downbutton similar
(* [downbutton=button]
(* but different shading
downbutton.clr2(3)=$01
downbutton.clr3(3)=$0f

(* lets display button on a lite gray background
(* with no text cursor
SHELL "display 1b 33 0e 05 20 "
(* no echoing
SHELL "tmode noecho"
(* clear screen
PRINT CHR$(12)

PRINT CHR$(2); CHR$(54); CHR$(42);
"Press a key to see button move"

(* all we have to do to produce the button is
PUT #1,button
PRINT CHR$(2); CHR$(67); CHR$(45);
"Quit"

(* now lets make it move when
(* user strikes a key
GET #0,char

fclr(3)=$0f
PUT #1,fclr
PUT #1,downbutton
PRINT CHR$(2); CHR$(67); CHR$(45);
"Quit"

(* sleep for a bit
(* so user can see button in down position
callcode=$0a
regs.d(1)=$64
RUN syscall(callcode,regs)

(* now replace the up button
fclr(3)=$00
PUT #1,fclr
PUT #1,button
PRINT CHR$(2); CHR$(67); CHR$(45);
"Quit"
(* put cursor back
SHELL "display 05 21"
(* turn echo back on
SHELL "tmode echo"
```

Well, that's all for this month. Next month I hope to start exploring the world of termcap! Stay tuned for some exciting OS-9 Answers, and as always, if you have something of interest you think others would benefit from, by all means send it in! Stay happy and healthy.

< 268'm >

Any comments, questions, or source code to be included in Joel's column may be sent in care of 68'Micros or directly to Joel at:
Joel Mathew Hegberg
936 N. 12th Street
Dekalb, IL 60115
E-mail : joelhegberg@delphi.com

MM/1 and OSK support from BlackHawk Enterprises

Hardware:

MM/1 Serial Cards	$35
MM/1 Midi Cards	$45
68340 acelerators	$325
SCSI Tape drives	call
SCSI Hard drives	call
BGFX in stock!	$45
RAM prices	call
Floppy Drives	call
Coming Soon - Modems, CD-ROM	

Software:

PixUtils	$25
DeskTop for MM/1	$79
Fontasee	$35
Paint for MM/1	$79
New Software on the way!	

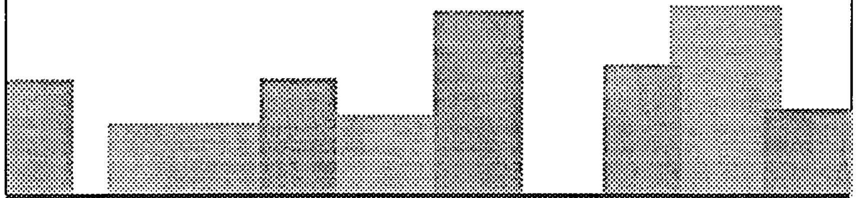
Now available -
~~COMPLETE~~
MM/1 Systems!
(call for pricing)

BlackHawk Enterprises, Inc.
P.O. Box 10552
Enid, OK 73706-0552
Phone 405-234-2347
Internet: nimitz@delphi.com

Announcing the

4th Annual "Last" Chicago CoCoFest

April 29 & 30, 1995

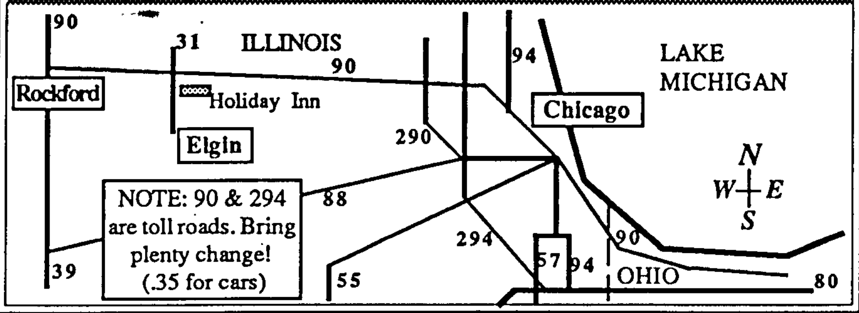


at the

Elgin Holiday Inn

a Holidome Indoor Recreation Center

345 West River Road, Elgin, IL



NOTE: 90 & 294 are toll roads. Bring plenty change! (.35 for cars)

Show Hours:

Sat., April 29 10:00AM-6:00PM
Sun., April 30 10:00AM-4:00PM

Admission:

$5.00 in advance, $8.00 at door
(2 day pass only, order before 16 April)

Address for advance tickets:

Tony Podraza, Fest Coordinator
119 Adobe Circle
Carpentersville, IL 60110-1101

Reservations:

1-708-695-5000
1-800-465-4329

Sponsored by:

Glenside CoCo Club

Vendor Information:

Booth Price: $35 ($30 for 2nd)
Member Price: $30
(Membership can be purchased at registration -- $15 per year)
Reservations must be received no later than 3/25/95. Deposit of $20 per booth required with balance due due no later than 4/26/95. Balance received after 4/16/95 subject to a 20% late fee. Vendor setup Sat, April 29 5:30AM-9:45AM

PRINTER RIBBONS RE-INKED

Don't throw away worn ribbons!

● Tandy DMP130

I have aquired equipment to re-ink black ribbons for the following printers:
Tandy DMP105/107/130 thru 134
Panasonic KXP110/115/145/1080/1090/1124/ 1180/119/1524/1624/1592/1595/1695
Roland DG PR 1010/1011/2417/2450

I will re-ink black cartridge ribbons for **$3.00 each including return shipping.** ***Money back if not satisfied.***

The ink used is a black abrasive free lubricant ink that is darker than most new ribbons.

L.Winterfeldt,
4045 E.Crocus Drive
Phoenix, AZ 85032-5417

SMALL GRAFX ETC.

"Y" & "TRI" cables. Special 40 pin male/female end connectors, priced EACH CONNECTOR	$6.50
Rainbow 40 wire ribbon cable, per foot	$1.00
Hitachi 63C09E CPU and Socket	$13.00
512K Upgrades, with RAM chips	$72.00
MPI Upgrades	
For all large MPIs (PAL chip)	$10.00
For small #26-3124 MPI (satellite board)	$10.00
Serial to Parallel Convertor with 64K buffer, cables, and external power supply	$50.00
2400 baud Hayes compatible external modems	$40.00

ADD $2.00 S&H TO EACH ORDER

SERVICE, PARTS, & HARD TO FIND SOFTWARE WITH COMPLETE DOCUMENTATION AVAILABLE. INKS & REFILL KITS FOR CGP-220, CANON, & HP INK-JET PRINTERS, RIBBONS & Ver. 6 EPROM FOR CGP-220 PRINTER (BOLD MODE), CUSTOM COLOR PRINTING.

TERRY LARAWAY, 41 N.W. DONCEE DRIVE
BREMERTON, WA 98310 206-692-5374

The Industrial OS-9 User... by *Ed Gresick*

G-*WINDOWS*

This is the first of a series of articles about G-WINDOWS. This installment will cover the history, objectives, organization and a few features of G-WINDOWS. Future articles will describe how to use G-WINDOWS and if there is sufficient interest, this series can continue with programming under G-WINDOWS with the Developer's Pack.

Much of the information to be presented will come from the section titled 'GETTING STARTED' from the G-WINDOWS USER'S MANUAL distributed with G-WINDOWS by DELMAR CO.

HISTORY

About 6 years ago, Steve Adams felt OS-9 needed a windowing system. At the time, he was working with graphics for a Company providing VME equipment. He investigated X-Windows being developed at MIT. Reviewing this code, he decided it would be better to follow his own knowledge of graphics and OS-9 and his instincts.

Steve purchased an Atari 1040ST and went to work. Many of his concepts are original but many of the features have roots in the work done by the Xerox Corporation at their Pala Alto facility in the early 80's as have most of the popular windowing systems including X-Windows. Steve's work came to the attention of GESPAC, an International Company based in Switzerland, one of the leading providers of OS-9 hardware. GESPAC was looking for a windowing system and a marriage was made. G-WINDOWS has been available commercially since 1990.

OBJECTIVES

Some of the objectives in designing G-WINDOWS were:

- multiple, active, text/graphics windows to take full advantage of OS-9's multitasking capabilities.
- support for pointing devices such as mice and touch screens.
- ease the use of the more common OS-9 commands.
- insure applications written under one platform will work on all platforms supporting G-WINDOWS.
- support for screen resolutions from 640 x 480 pixels and higher
- support 2 colors (monochrome), 16 colors and 256 colors.
- small code size to minimize resource requirements.
- permit easy porting to almost any OS-9/OS-9000 platform.

ORGANIZATION

As distributed, G-WINDOWS comprises 2 main components; WFM, the window file manager and DESKTOP, the Graphical User Interface.

WFM is central to G-WINDOWS and permits the creation and control of multiple, varying sized and types of windows on a graphics screen. Keyboard and mouse (or other pointing device) inputs are routed to the selected process by WFM. WFM supports pop-up menus, alert messages and request boxes when required by an application. A process may have more than one window and access them at will. WFM expands the command line editing capabilities of OS-9 and it will store the last 50 commands which may be recalled with the arrow keys. Two classes of fonts are supported by WFM. Quick fonts (qfonts)provides three user selectable font sizes. The second class of fonts, the general fonts, provide several different font types, Courier, Helvetica and Times in sizes ranging from 9 pts to 72 pts.

The DESKTOP Manager is an application which runs under WFM. DESKTOP replaces many of the file handling and program starting functions of the OS-9 shell. DESKTOP is not intended to replace the shell as an execution or development tool, but may be used in conjunction with shell. It is visually oriented rather than text oriented. Most DESKTOP commands are entered by pointing and clicking a pointing device - usually a mouse. Several other features are included to ease usage. A custom, user configurable menu is supported. The custom menu permits selection of processes by pointing and clicking the mouse. Additionally, many file types are recognized permitting the user to point to the file name or icon and click the mouse to start the appropriate process.

It is not necessary to run DESKTOP when using G-WINDOWS. Indeed, there are many instances, mostly in the industrial/commercial markets, where a G-WINDOWS application will be used and the user will never leave it.

UTILITIES & DEMOS

There are many utilities and demo programs provided with most G-WINDOWS distribution diskettes. Some of these are:

calc - a simple calculator - does decimal to hex to binary conversions
clock - default is small and unobtrusive
compress_image - saves disk space
cpu_usage - shows actual cpu usage graphically
edit_8bit - a simple graphics editor
edit_64 - similar to edit_8bit but restricted to a small image.
export_gif - export a G-W image file to a .gif file
eyes - tells you where the mouse is
fns - fish and sharks demo program
font_demo - displays the different font styles and point sizes available.
image_to_deskjet - print an image to a deskjet printer
image_to_paintjet - print an image to a paintjet printer
import_gif - import a .gif file to G-W image format
maze - nice demo
mem_usage - shows memory usage - especially useful when memory is tight
menu_demo - demonstration of menuing capabilities of G-W
savecrt - save all or selected portions of a G-W screen/window.
viewimage - view a G-W image file

Next month we'll get into using G-WINDOWS and the benefits it may offer the user.

For comments or questions, Ed can be reached via this magazine or:
E-mail:
EDELMAR@delphi.com
76576,3312@CompuServe.com

U.S. Mail:
PO Box 78
Middletown, DE 19709

Telephone:
302-378-2555 Voice
302-378-2556 FAX

The Hardware Hacker

F.G. Swygert

Using a null-modem cable to pass info between computers

Where does time go? It seems that it wasn't but days ago that I just put out a n issue of this magazine, and here I am up against the deadline for another issue! So son't worry about Marty... he will continue this column in the next issue. It's just that I didn't give him enough notice to prepare a column this time, so I'm filling in just for this issue.

At first I intended to fill in with an excerpt from "Tandy's Little Wonder" (see that FARNA Systems ad in this issue) on repairs or upgrades. Then I remembered a letter I had just typed in hours earlier from Charles Radatz asking about a null-modem conection between the CoCo and an Aple IIc. I realize that many readers have morethan one computer, so a quick method to transfer data between computers is more than appropriate. Besides, a null-modem connection is relatively simple, isn't it?

The Basics

Although there are 25 lines defined in the RS-232c standard, only three are required for minimal communications equipment. These are transmit, receive, and signal ground. If the system is to support full duplex operation, a carrier detect must be added to offer some sort of handshaking signal. This was how the CoCo serial port is designed.

So why are there so many lines if only three or four or actually necessary (most systems use no more than seven signals)? The RS-232c standard was defined by the Electronics Industries Association (EIA) in the early days of computing (mid 70s). They had no idea what types of equipment would be developed in the future, so they provided many different signals to make the standard as flexible as possible.

Handshaking

In serial communications, the computer is considered the data terminal equipment (DTE... the computer or terminal) and another the data communications equipment (DCE... the modem) or data set. The two lines on an RS-232 connector that controls communication between these pieces of equipment are the DTR (data terminal ready) and DSR (data set ready). When the computer is ready to send or receive data, it holds the DTR line high. When the modem is ready, it holds the DTS line high. No data will flow betwen the computer and modem until both are high.

The "Null" Connection

Look up "null" in the dictionary. Webster says it means "literally none". So it's easy to see how the term "null modem" was derived.

Since computers are set up as DTE, they expect to send data on one line and receive on another. If connections between computers are made line to line, as with a real modem, this wouldn't work. The transmit and receive lines must be crossed, so that one computer is sending to the other's receive line.

But more than that is involved due to handshaking requirements. The CD line of the CoCo must be connected to the DSR, CD, and DTR lines of the other computer, just as it is with a modem. This takes care of the necessary handshaking.

Data Transfer

Once a proper cable is made, all that is needed is a communications program for each computer. The sending computer will be set to upload, or send, a file. The receiving computer will be set to download, or receive, a file. Start the send first, then the receive. Both computers must, of course, be set at the same baud rate. Set the computers at the highest baud rate that the communications programs will support. The CoCo should easily handle 9600 baud through a null modem connection. Set both communications packages for half-duplex operation if possible for the best results.

The Commodore VIC-20, 64, or 128 aren't mentioned for good reason. These computers have a non-standard, TTL level serial port. Unless you have a true RS-232c adapter for them, you will have to make the connection between actual modems. But you can't just link a piece of telephone wire between most modems and expect them to work. They expect voltage on one of the telephone lines. I have seen such a "phone line eliminator" before, but can't find the information at this moment. If someone really needs this info, or has it, write and let me know. I'll see what I can dig up.

Conclusion

That's all there is to it! I have supplied pin-outs for several computers in the sidebar. I would not suggest trying on of the ready made null modem adapters available in computer stores. They may not work properly due to the handshaking requirements of the CoCo.

Standard EIA RS-232C Connections: All computers with 25 pin DB-25
Note: Signals are usually abbreviated with initials, i.e. Request to send = RTS, etc.

1- chassis ground	14- secondary transmit
2- transmit data	15- xmit clock (DCE)
3- receive data	16- secondary receive
4- request to send	17- receive clock (DCE)
5- clear to send	18- unassigned
6- data set ready	19- secondary RTS
7- signal ground	20- data terminal ready
8- carrier detect	21- signal quality detect
9- data set test	22- ring indicator
10- data set test	23- data signal rate
11- unassigned	24- xmit clock (DTE)
12- secondary CD	25- unassigned
13- secondary CTS	

Color Computer 4 pin DIN

1- carrier detect	3- signal ground
2- receive data	4- transmit data

IBM AT (clone) 9 pin DB-9

1- carrier detect	6- data set ready
2- receive data	7- unassigned
3- transmit data	8- clear to send
4- DTR	9- ring indicator
5- ground	

Macintosh 9 pin DB-9

1- chassis ground	6- DTR
2- unassigned	7- carrier detect
3- signal ground	8- unassigned
4- unassigned	9- recieve data
5- transmit data	

Apple IIc 5 pin DIN

1- DTR	4- transmit data
2- DSR	5- receive data
3- signal ground	

Macintosh Plus 8 pin Mini DIN

1- DTR	5- receive data
2- clear to send	6- unassigned
3- transmit data	7- unassigned
4- signal ground	8- unassinged

Note: These connections were taken form a Maxon MAX2400 modem manual. The pins labled "unassigned" may have a function on the listed computers that are not necessary for modem/null modem communications.

Standard Null Modem Connection:

CoCo 4 pin	*IBM AT (clone) 9 pin*
1 (CD)	1,4,6 (CD, DTR, DSR)
2 (RD)	3 (TD)
3 (GND)	5 (GND)
4 (TD)	2 (RD)

SDSK512K

Disk Organizer for DECB
Requires a CoCo 3 with two drives (one may be a RAM disk)

by Charles R. Connolly

I will start out by describing just how I came to decide to write this application. I am a bit of a late comer to the CoCo. I got my 128K CoCo 3 in 1990 because my IBM XT compatible was monochrome only and not very suitable for games and because the CoCo 3 had a had a serial port that could easily be connected to my XT so I could write letters on the CoCo while my XT was busy printing other letters on my slow 9 pin printer.

For the first nine months I used a tape recorder & cartridges, learning some assembler with an EDTASM+ cartridge, writing with SCRIPSIT, and playing some games. Then in 1991 purchased a DISTO no-halt SuperControler II, one single sided drive, OS-9, and some other disk based software. With this setup OS-9 was limited so I worked more with Disk Basic.

In '92 I added 512K of memory, a modem, and two DSDD disk drives. I also learned the basics of OS-9 and Video Titling with CoCo Max 3 and the switch box setup described in the December '90 Rainbow Magazine.

In '93 I got back to Disk Basic more because two of my three drives went bad, leaving me with just one single sided disk drive and 512 K. Most of my OS-9 efforts were stranded on DSDD disks so more Disk Basic.

With just one disk drive I was annoyed to find that all the "Point & Enter" (arrow and enter key driven) disk organizers required one swap or more for every file. If only there were some way to just pick all the files you wanted to copy from the SOURCE disk and then have them copied to one of the two RAM Disks my DISTO 512K upgrade supports. Then you would only need to swap disks once to transfer all files to the final destination. The only problem was I knew nothing of how the Tandy disk system worked. If I had a Tandy disk controller the included Disk Basic manual would have the required information. The DISTO controller, however, is primarily targeted at those whose main interest is OS-9 and so did not include a Disk BASIC manual.

After a while my disks became very disorganized with the result that I found myself wasting more and more time looking for programs rather than using them. I borrowed the 5 1/4" drive from my XT for a few days to clear up some of the problems using the standard organizers and to recover some OS-9 stuff stranded on double sided disk drives. I then decided to write my own file organizer, so I borrowed the Tandy Disk Basic manual from a friend for a few weeks and wrote the core of this program.

After the core was written I found myself thinking of and adding new enhancements every now and then, until I decided that perhaps other people would be interested in this organizer. Even after I got two more DSDD drives, I continued to add enhancements.

Eventually I ran out of memory so I decided to break the organizer down into three files. The Main Organizer performs the file listing, marking, copying, killing, and renaming functions and allows changing drives and loading or running a BASIC program.

The Final program has most (but not all) of the standard features found in other file organizers, is very easy to use, and has the following advantages over other file organizers I have seen:

1. Can Organize Files on RAM Disks as well as physical disks (I have tested it with the RAM Disk that comes with the DISTO 512K Memory Upgrade)
2. Can Select Files Using Wildcard Matching (this is with regular DECB)
3. Can do a disk backup while running a file editor in a window
4. Has context sensitive on line help windows for most functions
5. Single letter commands for most functions
6. 40 column screen for the main file organizer screen. Is easy to read and the three column format displays up to 66 files at a time.
7. Can handle up to 68 files.
8. Written completely In BASIC, so the program could be adpated for a CoCo 2 or a HYPER-IO system.

What Could be Added and Improved.

As I have suggested the program, being written in BASIC, could easily be adapted for other circumstances. Also although the program does everything I want and is very user friendly, it is not perfect. In particular there is very little error trapping done.Four possible projects:

1. Bullet Proofing - Try changing the DSKI$ commands to calls to the DSKCON which will not abort on I/O errors, then use the I/O errors to go to appropriate error trapping routines.
2. Backup routine is very slow. Perhaps It could be redone in assembly or 6309 code.
3. Add a disk format command. I have no Idea how to do this and still be able to return to the disk organizer, but I am sure It's possible.
4. Do a LIST of the contents of a file from the organizer. I will outline how you could go about this later.

How the program works.

Most of the programming is quite strait forward and comments are liberally used. However after running out of memory I started removing comments so It is not as well commented as I would have liked. Also all the INKEY$ I/O is done by a single GOSUB which makes debugging difficult, you might want to use regular INKEY$ statements until you find yourself feeling the memory crunch.

The key to understanding this program is to understand how Disk BASIC organizes it's directories, as this program uses the BASIC DSKI$ command to do a direct read of the disk directory. To make use of this information we have to know where to find it and how to interpret it.

The directory information on a Disk BASIC disk is located on track 17, sectors 3 to 11. Each directory entry is 32 bytes long and contains the following information used by this program:

Bytes 0 to 7 contain the file name.

Bytes 8 to 10 contain the file extension.

These names we save in the arrays F$(N) and E$(N) and list to the screen in three column format.

Byte 11 contains the file type which is saved in so it can be displayed as the extended file information on the highlighted file.

Byte 12 is 0 if the file is in BINARY format and FF if the file is ASCII.

Byte 13 is the granule number of the first granule in the file (the minimum allocation unit under DECB, 2304 bytes long).

In this program we duplicate all the information shown in by the standard DIR command, most of which we can get directly from the directory entries. But the file size is not part of the directory entry. To determine file size one must get the start granule and then use the file allocation table (Track 17 Sector 2) to determine if the start granule is the last or if there are more granules in the file.

For example, suppose you looked at the directory entry for an ASCII, word processor text file named FILENAME/EXT. The contents might be something like this.

Bytes 0-7	8-10	11	12	13
FILE NAME	EXT	3	FF	10

You would then do a DSKI$ D,17,2,SL$,SH$. Now D is just the current disk drive, SL$ contains the 128 low bytes of the sector and SH$ contains the 128 high bytes of the sector. Lets assume that FILENAME/EXT is the only file on this disk, then if we look at the string SL$ we might see:

```
N=  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14
  255 255 255  56 255 255 255 255 255 255  11  12  21 255 255
N=15 to N=74
  255 255 255 255 255 255   3 255 255 255 255 255 255 255 255
  255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
  255 255 255 255 255 255 255 255 255 255 255  57  58  59 196
  255 255 255 255 255 255 255 255   0   0   0   0   0   0   0
  N=75 to N=128 all zero's.
```

where the number ASC(MID$(SL$,N+1,1)) is shown in each box above.

How do we use this information?

Remember that from the directory information we found that the start granule is number 10 so we know that the next granule in the file is ASC(MID$(SL$,N+1,1)) provided that value is between 0 and 67 in the example shown above the value is 11. Now we know that we have at least 2 granules 10 and 11 in the file we repeat the operation and find the next granule in the file until we find an entry whose contents are not between 0 and 67. In this case we get the following chain:

10-11-12-21-3-56-57-58-59-196

Counting the granules in the chain we see that the file is 10 granules in size. So that is how I find out how large a file is.

If you wanted to add a file listing function to this program you could use this method to determine where the file's information is on disk and thus list it no matter what type of file it is. To do this you would need two more pieces of information. First you would need to know what track and sectors each granule number refers to, and secondly you would need to know how many characters of the last granule are actually in the file.

To convert granules to track and sectors remember that each granule is made up of 9 consecutive sectors and that the formula for finding the track is:

TRACK = INT(GRANULE # / 2)

and the start sector is 1 if the granule number is even and 10 if the granule number is odd. The number of characters of the last granule that are actually in the file is given by bytes 14 and 15 of the directory information. The rest of this article is short manual on the use of the program.

Adding Apps to SDSK512K.

This organizer has been designed in such a way as to be a desktop for applications which follow a few rules:

1. If the program is written in BASIC it should either not use the memory from address 30000 to 32500 or SAVEM these addresses before using them and restore them when done.
2. A ML program should be located above address 8000 and also either not use the memory between 30000 & 32500 or save and then restore this area if single tasking or not use this area if multi-tasking.

The memory map of the reserved region above address 30000 is a follows:

30001- FLAG Indicate if a RAM disk already exists.
30010- The scratch disk number or FF if none.
30100- FLAG Skip open- ing screen ?
30101- The program disk drive number.
30198- Current Drive
30199- "Y" if next bytes contain the OC(N).
30200-30267 A copy of the OC(N) - file marking array.
30301-30308 The name of the ML program to be executed by X.BAS.
30309- The start granule of the ML program to be executed by X.BAS.

The intended purpose of the rest of the area between 30310 and 32500 is for allowing programs written for the SDSK512K environment to share data (i.e.: to let you cut from one application and paste into another). I have not implemented any examples of this other then the way the OC(N) array is passed from SDSK512K to WILD and back in memory. But It makes a good possibility.

If you decide to use any of this memory this way you should code the addresses near the front of the file so they can be changed easily in the BASIC program or provide for a file from which the addresses are loaded. That way it will be easier for other people to resolve memory conflicts between applications written by different authors.

How To Use The Disk Organizer.

First you must either edit this file to replace all references to RAM.BIN with the name of your RAM disk program or rename your ram disk program to RAM.BIN and copy your RAM disk program file(s) onto the same disk as the file organizer. Then run your RAM disk program to format a blank RAM disk in memory. Use the MAKBLANK.BAS program to create the file BLANK.BAS. This file has a copy of the sectors 2 to 11 of track 17 of the blank ram disk. The SDSK512K program uses BLANK.DAT to delete all the files on the scratch disk quickly when required. This summary of how to use SDSK512K is based on the assumption that you are starting with the PROGRAM disk in drive 0.

From Basic Type:

PCLEAR1: CLEAR 100,30000: RUN"SDSK512". You might want to make a little loader program to do this (see "GO")

The 40 Column title screen will appear with the title at the top and the prompt:

INSTALL RAM DISK(s) (N/y)?

If you wish to Install your ram disk type Y as long as you have a RAM disk setup program that does not use addresses below 30K, the RAM disk setup program will run and you will return to SDSK512K automatically, now with a RAM disk available.

After your RAM disk setup program has finished running the screen clears and the following prompt appears near the bottom of the screen:

No Scratch Disk NEW #>_

If you want one of your RAM disks to be available for use by the BACKUP program then you type the number of that disk now. If you don't want a scratch disk type E. The screen clears again and the following message appears at the top of the screen:

Insert Data Disk in Drive #0
Type 'R' when Ready

At this point you can remove the program disk from drive 0 and replace it with the disk containing the files you want to organize. Then type R. The screen clears and the message Reading Directory is displayed while the the disk drive is accessed. Then the directory starts to appear and may be as many as three columns wide if you have more than 44 files.

The Second line from the bottom contains a menu which looks like this:

Menu: B C D F G K L Q R U V W X ?=Help

This menu is a summary of the single letter commands available. Pressing ? or / pops up a help window that gives a more detailed explanation of the options available on the menu. The bottom line contains more detailed information on the highlighted file which would be the file at the top left of the screen initially. If you were to highlight the SDSK512K.BAS program itself the bottom line would be:

DRIVE # 0 FILE:=SDSK512K.BAS 0 B 7

Menu Options

The help screens will help guide you through the program however to add some detail to your understanding of how the program works we will describe the function of the main keys here. Essentially the way this application works is that you mark all files you want to deal with first by highlighting them and then pressing C K or R, and when you are happy that you have marked all the files you want to deal with you press G for GO. The files are then dealt with and you are returned to the OK prompt where you can use DRIVE n, DIR n, and ? FREE(n) commands if you wish and still be able to type RUN to return to SDSK512K.

A" K", "R", or drive number follows the name of each marked file. If you accidently mark the wrong file then just highlight it and type U to UNDO your mistake.

Q lets you decide wether you wish confirm copies and kills before they are executed.

B runs the single disk backup program.

W runs the wildcard builder program that lets you mark all programs with a particular filename substring extension or type. This program as written only works when the data disk is in drive 0. The current drive must be zero before running the program.

D lets you determine which disk drives will be assigned each function. If you press D the following prompt will appear:

Data Program or Scratch CD>_

If you type D you will be asked "are you sure ?" If you type Y you will be asked for the drive #. The current directory will change to that drive and the program will be rerun from that drive.

If you pressed P you will be prompted for the new program disk. Due to the memory crunch I have only implemented drive 2 as an alternative disk. You will be asked if you want to copy the program files to drive 2. Say yes if you are not sure if they are there.

If you press S you will be asked to assign a new scratch disk. Keep in mind that some operations of the disk organizer will destroy all data on the scratch disk and be careful not to forget any files on the scratch disk.

F is used to figure out how much disk space your marked files will require and how much space is available on the destination disk(s). For details just type F and then ?.

L loads a basic program and the X executes a BASIC or a ML programs. To start a ML program from the organizer you must have a copy of X.BAS on the same disk as the ML program you wish to execute.

Finally, the V option turns verify on or off.

External Programs

Lets talk about the two external programs WILD and BACKUP. Wild is a great time saver and a very friendly program. Backup is an interesting demo If someone could increase the speed of this program it might be more useful, it's still kind of neat because it gives the appearance of two programs multitasking.

To use the WILD program the data disk must be drive 0. The program could be easily generalized for more drives, a good programing exercise for a beginner.

The WILD program can be run by typing W. You will be prompted to insert your program disk in the appropriate drive. As soon as the program is loaded you will be asked to insert the data disk in drive 0. The screen will change to 32 columns with the message READING DIRECTORY.

The rest of WILD is pretty self explanatory. Note that the default extensions I have provided can be changed by editing the MENU-EXT.DAT file.

To run the backup program type B. If you have not designated a scratch disk you will be prompted to do so. If you are not sure you have a safe scratch disk available press BREAK to exit at this point. If you do not BREAK The disk backup utility runs in an 80 column screen after you select a scratch disk. Insert the disk you want backed up and type Y at the PROCEED ? prompt. The backup utility will display each track and sector number to indicate progress.

If you press CLEAR an editing window will open so you can write a short letter while you are waiting for backup to finish. You will find that this editor is not suitable for touch typing as a fairly firm press is needed for each key. The carriage return and all arrow keys work and the processor is always in overwrite mode. F1 clears the current line and F2 saves the file to Drive 2. Normally I have drive 2 and 3 as RAM Disks; drive 2 as the program disk and drive 3 as the scratch disk. This setup is just fine if you want to use the editor. Remember, you should not use the editor if you are using drive 2 as a scratch disk. Doing a backup with BACKUP is very very slow but it's kind of neat to see this example of BASIC pretending to multi-task makes you appreciate OS-9.

Conclusion

When using the file organizer be sure that the file highlighted and the file detailed on the bottom line are the same. I think I have eliminated such "sync" bugs. But they have occasionally cropped up and if you do not pay attention such an error could result in one selecting the wrong file. As long as you have copied the main help subroutine correctly you can always re-sync the files by selecting the HELP menu and then rechecking the files just before you do a GO. If you have any questions just send an SASE to:

Charles R. Connolly
6 Silver Maple Court, #915
Brampton, Ontario, Canada
L6T 4N5

(editor: Please send any changes and/or improvements to this magazine, along with an explanation of what you did and how you did it, for future publication. It is not often that I get such good DECB programs and explanations!)

Program listing for GO.BAS:

```
5 PCLEAR 1:CLEAR 100,30000
10 LOAD "SDSK512K.BAS",R
```

Program listing for SDSK512K.BAS:

```
1 POKE &HFFD9,0:' Set COCO to 2MHz.
4 GOTO16
7 'INPUT
10 K$=INKEY$:IF K$="" THEN GOTO10
13 RETURN
16 ' SDSK512K.BAS
19 ' (c) Charles R. Connolly 1994
22 ' First Serial Rights - FARNA Systems
25 ' All Other Rights Reserved
28 PCLEAR 1
31 PALETTE CMP:ATTR 0,0
34 CLEAR 3000,30000
37 FILES 3,768
40 DIM F$(68),E$(68),T$(68),A$(68)
46 DIM SG(68),NG(68),FSL(68),EC(68),OC (68),GC(7)
52 DIM HH$(42),HO(5),WS(5),AT(5)
58 QM=2:SD=-1:PD=0:CO=0:'
61 POKE 30101,PD
64 CLS
67 NF=0:'# of Files Found.
70 SE=3:SA=442368:'HSCREEN @
73 D=PEEK(&H95A):'Current DRIVE #.
76 GOSUB1279:' Initalize HELP screen data
79 D0=PEEK(30100):'Opening Screen
82 WIDTH 40
85 IF D0<>78 THEN GOSUB619:'Opening Screen &
Init.
88 CLS:PRINT"  Insert Data Disk in Drive #";D
91 PRINT:PRINT"  Type 'R' when Ready"
94 K$=INKEY$:IF K$<>"R" THEN IF K$<>"r" THEN
GOTO94
97 CLS 1
100 PRINT:PRINT:PRINT"Reading Directory" :
PRINT: PRINT: PRINT
103 'REPEAT
106 :DSKI$ D,17,SE,SL$,SH$
109 :SEC$=SL$+LEFT$(SH$,127)
112:FORX=0TO7
115 ::Z=X*32:' Each Directory Entry is 32 Chars
118 ::IF MID$(SEC$,Z+1,1) = CHR$(255) GOTO133
121 ::IF MID$(SEC$,Z+1,1)<>CHR$(0) THEN
GOSUB328
124:NEXTX
127:SE=SE+1
130IFSE<12GOTO103:'UNTIL
133 NF=NF-1:F$(NF+1)=CHR$(255)
136 FOR I= 0 TO 67
139 :P0=PEEK(30200+I)
142 :OC(I)=P0
145NEXT
148 FOR I=0 TO 68:EC(I)=0:NEXT
151 'FIND SIZE OF FILE IN GRANULES
154 DSKI$D,17,2,SL$,SH$
157 FOR X=0 TO NF
160:G=1:GL=SG(X)+1
163:IFMID$(SL$,GL,1)>HEX$(191)THENNG(X)=G:
GOTO178
166::G=G+1:'While NOT last Granule
169::GL=ASC(MID$(SL$,GL,1))+1
172::IFMID$(SL$,GL,1)>HEX$(191)THENNG(X)=G:
GOTO178
175:GOTO166
178NEXT
181 PL=21:'File Names on Lines 0-21
184 FOR I=0 TO PL
187:FSL(I)=I+1+I*79
190:FSL(I+22)=I+27+I*79
193:FSL(I+44)=I+53+I*79
196NEXT
199 GOSUB1426:'List Directory
202 NN=0
205 'End of the INIT section.
208 N=NN
211 'MAIN Program Loop START
214:IFCD=1THENDRIVEDN:GOTO16:'READ NEW
DRIVE
```

```
217 :C=EC(N)
220 :S=FSL(N):'start addr of file highlite area.
223 :E=FSL(N)+22:'End Address
226 :'Show FILE cursor.
229 :FOR I=S TO E STEP 2
232 ::LPOKE SA+I,1
235 :NEXT
238 :GOSUB7
241 :SN=N
244 :'Menu:
247 :IF K$="0"OR K$="1"OR K$="2"OR K$="3"
THEN GOSUB385:GOTO211
250 :IF K$="W" OR K$="w" THEN GOSUB1462:
GOTO211:'Mark using wildcards.
253 :IF K$="B" OR K$="b" THEN GOSUB646:
GOTO211:'Backup a Disk.
256 :IF K$="C" OR K$="c" THEN GOSUB370:
GOTO211:'Mark Files To Copy
259 :IF K$="D" OR K$="d" THEN GOSUB529:
GOTO211:'Change Drives
262 :IF K$="F" OR K$="f" THEN GOSUB994
:GOTO211:'Report Disk Space Free
265 :IF K$="G" OR K$="g" THEN GOSUB409:
GOTO211:'Execute Commands
268 :IF K$="K" OR K$="k" THEN K$="K":
GOSUB391:GOTO211
271 :IF K$="L" OR K$="l" THEN K$="L":
GOTO1663:GOTO211
274 :IF K$="Q" OR K$="q" THEN GOSUB739:
GOTO211:'Set Query Mode.
277 :IF K$="R" OR K$="r" THEN K$="R":
GOSUB391:GOTO211:'Mark For Renameing
280 :IF K$="U" OR K$="u" THEN K$="U":
GOSUB457:GOTO211:'UNSELECT
283 :IF K$="V" THEN VERIFY ON: GOSUB970:
GOTO211
286 :IF K$="v" THEN VERIFY OFF: GOSUB970:
GOTO211
289 :IF K$="X" OR K$="x" THEN GOSUB 1510:
GOTO211:'eXecute highlited program.
292 :IF K$="?" OR K$="/" THEN GOSUB1213:
GOTO211:'Help Screen.
295 :K=ASC(K$):'CURSOR MOVEMENT
298 :IF K=10 THEN N=N+1
301 :IF K=94 THEN N=N-1
304 :IF K=9 THEN N=N+22
307 :IF K=8 THEN N=N-22
310 :IF N<0 THEN N=65
313 :IF N>65 THEN GOSUB1525
316 :'-
319 :IF N<>SN THEN GOSUB355
322 GOTO211
325 END
328 'SUB Get Disk Directory Entry Contents.
331 F$(NF)=MID$(SEC$,Z+1,8):'The FileName is
the first 8 Bytes of the Entry.
334 E$(NF)=MID$(SEC$,Z+9,3):'The File
Extension is the next 3 Bytes.
337 IF MID$(SEC$,Z+12,1)=CHR$(0) THEN T$
(NF)="0" ELSE IF MID$(SEC$,Z+12,1) = CHR$(1)
THEN T$(NF)="1" ELSE IF MID$ (SEC$,Z+12,1) =
CHR$(2) THEN T$(NF)="2" ELSE T$(NF)="3"
340 IFMID$(SEC$,Z+13,1)=CHR$(0)THEN A$(NF)
="B"ELSE A$(NF)="A"
343 SG(NF)=ASC(MID$(SEC$,Z+14,1))
346 NF=NF+1
349 RETURN
352 'SUB Undo the CURSOR
355 FOR I=S TO E STEP 2
358 :LPOKE SA+I,C
361 NEXT
364 GOSUB439:'DISPLAY THE FILE INFO.
367 RETURN
370 'HIGHLITE a CELL
373 '-FOR COPYING -
376 LOCATE 1,23:PRINT"DRIVE # ? ";
379 GOSUB7
382 IF K$<"0" OR K$>"6" THEN GOTO439
385 '- 0 1 2 3 ect.
388 IF VAL(K$)=D THEN GOTO439
391 '- or for otherreasons - kill, rename, etc.
394 LPOKE SA+E+1,ASC(K$):' Mark with opp. letter
397 LPOKE SA+E+2,24:' Highlited.
400 EC(N)=64:' Underline
403 OC(N)=ASC(K$)
406 GOTO439
409 '- GO & DO IT -
412 PRINT
415 FOR DI=0 TO NF
418 :IF OC(DI)=32 GOTO430
421 :IF OC(DI)>=48 AND OC(DI)<=54 THEN
GOSUB472: GOTO430
424 :IF OC(DI)=ASC("R") THEN GOSUB502:
GOTO430
427 IF OC(DI)=ASC("K") THEN GOSUB517:
GOTO430
430 :'skip
433 NEXT
436 END
439 'SUB CLEAN UP LINE 23
442 GOSUB577
445 LOCATE 1,23:PRINT"DRIVE # ";D ;" FILE:=";
F$(N);"."+E$(N);" ";T$(N);" ";A$(N);" ";
448 PRINTUSING "##";NG(N);
451 'Done.
454 RETURN
457 '—SUB UNDO
460 LPOKE SA+E+1,32
463 LPOKE SA+E+2,0
466 EC(N)=0:OC(N)=32
469 RETURN
472 ' COPY TO ANOTHER DISK
475 IF QM<>2 THEN GOSUB493:GOTO487
478 PRINT"COPY ";F$(DI)+"."+E$(DI)+":"; D;"
TO "+F$(DI)+"."+E$(DI)+":"+ CHR$ (OC(DI))
481 INPUT "(Y/N) ";CP$
484 IF CP$="Y" OR CP$="y" THEN GOSUB493:
'Do The Copy
487 'Done.
490 RETURN
493 'SUB to DO THE COPPYING.
496 COPY F$(DI)+"."+E$(DI)+":"+CHR$(48+D)
TO F$(DI)+"."+E$(DI)+":"+CHR$(OC(DI))
499 RETURN
502 'RENAME
505 PRINT"RENAME ";F$(DI)+"." +E$(DI)+" TO ";
508 INPUT RE$
511 RENAME F$(DI)+"."+E$(DI) TO RE$
514 RETURN
517 'KILL
520 IF QM<>0 THEN PRINT" KILL ";F$(DI) +"."
+E$(DI)+":"+CHR$(D+48):INPUT KL$
523 IF KL$="Y" OR QM=0 THEN KILL F$(DI) +"."
+E$(DI)+":"+CHR$(D+48)
526 RETURN
529 ' CHANGE DRIVE?
532 GOSUB577
535 GOSUB607:PRINT"Data Program or Scratch
CD>";
538 GOSUB7
541 IF K$="S" OR K$="s" THEN GOSUB811:
GOSUB586:RETURN:'Set Scratch Disk.
544 IF K$="P" OR K$="p" THEN GOSUB 793:
GOSUB586:RETURN:'Set Program Disk.
547 GOSUB607:PRINT"ARE YOU SURE ?";
550 K$="N"
553 GOSUB7
556 IF K$="Y" OR K$="y" THEN GOSUB562
ELSE GOSUB586
559 RETURN
562 ' CHANGE DRIVE.
565 CD=1:' Yes We Will Change Drives.
568 LOCATE 1,23:PRINT"DRIVE # ";
571 INPUT DN
574 RETURN
577 ' CLEAR LINE 23
580 LOCATE 1,23:PRINT" (40 spaces) ";
583 RETURN
586 ' Menu
589 LOCATE 1,22
592 PRINT"Menu: B C D F G K L Q R U V W X
?=Help"
595 FOR IM= 444127 TO 444207 STEP 2
598 :LPOKE IM,24
601 NEXT
604 RETURN
607 'SUB to Ready 22 for Write
610 LOCATE 1,22:PRINT" (40 spaces) ";
613 LOCATE 1,22
616 RETURN
619 ' SUB Opening Screen.
622 FOR I=0 TO 68:POKE 30200+I,32:NEXT
625 LOCATE11,1:PRINT"SDSK512K V1.185";
628 LOCATE 8,2:PRINT"Disk Orgnizer For:";
631 LOCATE9,3:PRINT"512K RamDisk Users";
634 LOCATE 5,22:PRINT" INSTALL RAM DISK(s)
(N/y)? ";
637 GOSUB7
640 IF K$="Y" OR K$="y" THEN GOSUB1495
643 RETURN
646 'SUB to BACKUP DRIVE 0 Using The Scratch
Disk (presumably RAM).
649 CO=4
652 GOSUB607:' Ready Line 22
655 PRINT"BACKUP UTILITY DRIVE #";
658 PRINTUSING "#";D
661 IFSD>0 THEN PRINT"BACKUP WILL USE
DRIVE";
664 IF SD>0 THEN PRINTUSING "#";SD;
667 IFSD>0 THEN PRINT"AS SCRATCH DISK";
670 IF SD<1 THEN GOTO694
673 PRINT" B[?LGQ]>";
676 GOSUB7
679 :IF K$="?" OR K$="/" THEN GOSUB1213:
GOSUB607:PRINT"B>";
682 :IF K$="L" THEN IF SD>0 THEN CLS5: WIDTH
80:DIR SD:GOSUB1213:GOTO649 ELSE GOTO703
685 :IF K$="G" THEN GOSUB706:RUN
688 :IF K$="Q" OR K$="q" THEN GOTO721
691 GOTO676
694 PRINT"You MUST make a SCRATCH DISK 1st.";
697 GOSUB577:LOCATE 0,23:PRINT" Type: S and
then ? at the S> Prompt";
700 K$=INKEY$:IF K$="" THEN GOTO700 ELSE IF
K$="S" OR K$="s" THEN GOSUB811
703 'Done.
706 'SUB to BACKUP
709 ATTR 6,1:CLS:LOCATE 2,3
712 PRINT"Place PROGRAM Disk in Drive";PD
715 PRINT" Type R when READY"
718 GOSUB7:IF K$="R" THEN BF=D:BT=SD: RUN
"BACKUP:"+CHR$(48+PD)
721 GOSUB1246
724 RETURN
727 GOSUB607:PRINT"QM = ";QM;
730 CO=0
733 GOSUB586
736 RETURN
739 'SUB to set the Query Mode.
742 CO=2
745 GOSUB607:' Ready Line 22
748 PRINT"Current Query Mode = ";
751 PRINTUSING "#";QM;
754 PRINT" Q>";
757 GOSUB7
760 :IF K$="?" OR K$="/" THEN GOSUB1213:
GOSUB607:PRINT"Q>";
763 :IF K$="0" THEN QM=0:GOTO778
766 :IF K$="1" THEN QM=1:GOTO778
769 :IF K$="2" THEN QM=2:GOTO778
772 :IF K$="Q" OR K$="q" THEN GOTO778
775 GOTO757
778 'Done.
781 GOSUB607:PRINT"QM = ";QM;
784 CO=0
787 GOSUB586
790 RETURN
793 'SUB to change the PROGRAM disk.
796 GOSUB1585
799 GOSUB607:PRINT"Copy ?"
802 GOSUB7
805 IF K$="Y" OR K$="y" THEN GOSUB1609:'Copy
Program to New Disk.
808 RETURN
811 'SUB to setup a SCRATCH disk.
814 CO=3:SZ=0
817 ' Redisplay Scratch Disk Message.
820 GOSUB607
823 IF SD<1 THEN POKE 30010,255: PRINT "No
Scratch Disk";
826 IF SD=>1 THEN POKE 30010,SD:PRINT "Current
Scratch Disk #";
```

```
829 IF SD=>1 THEN PRINTUSING "#";SD;
832 IF SZ=1 THEN FOR DE=1 TO 1000:
NEXT:GOTO865
835 SZ=1
838 PRINT" NEW #>";
841 GOSUB7
844 :IF K$="?" OR K$="/" THEN GOSUB1213:
GOSUB607:PRINT"#>";
847 :IF K$="1" THEN SD=1:GOTO817
850 :IF K$="2" THEN SD=2:GOTO817
853 :IF K$="3" THEN SD=3:GOTO817
856 :IF K$="K" THEN GOSUB880:GOTO817
859 :IF K$="E" OR K$="e" THEN SD=-1: GOTO817
862 GOTO841
865 'Done.
868 GOSUB607:PRINT"SD = ";SD;
871 CO=0
874 GOSUB586
[illegible] GOSUB1246
877 RETURN
880 ' SUB to kill all files on the Scratch Disk.
883 GOSUB607
886 IF SD<1 THEN PRINT:SOUND 2,7:ATTR 3,3:
PRINT"SCRATCH DISK NOT DEFINED":RETURN
889 PRINT"ARE YOU SURE ? ";
892 GOSUB7
895 IF K$<>"Y" THEN GOTO949
898 ATTR 2,4:CLS:LOCATE 5,3:PRINT" All Files
on Disk # ";
901 PRINTUSING "##";SD;
904 PRINT:PRINT
907 ATTR 3,3:PRINT"        WILL BE KILLED        ";
910 ATTR 2,4:LOCATE 20,10:PRINT"A to ABORT";
913 GOSUB7
916 IF K$="A" OR K$="a" THEN GOTO961
919 GOSUB607:CLS:PRINT"INSERT RD-TOOLS
DISK IN :";PD
922 GOSUB577:LOCATE 7,21:PRINT" PRESS
ANY KEY TO START";
925 GOSUB7
928 IF PD=0 THEN OPEN "D"#3,"BLANK.DAT:0"
931 IF PD=2 THEN OPEN "D",#3,"BLANK.DAT:2"
934 FOR KS=2 TO 11
937 :GET #3,KS*2-3
940 :INPUT #3,SL$
943 :GET #3,KS*2-2
946 :INPUT #3,SH$
949 :DSKO$ SD,17,KS,SL$,SH$:' ZAP scratch disk.
952 NEXT
955 IF PD<1 THEN UNLOAD 0 ELSE IF PD=1 THEN
UNLOAD 1 ELSE UNLOAD 2
958 IF D=SD THEN GOTO16
961 'Done
964 GOSUB1246
967 RETURN
970 'SUB VERIFY on/off message
973 GOSUB607:' Ready Line 22
976 PRINT"VERIFY ";
979 IF K$="V" THEN PRINT"ON";
982 IF K$="v" THEN PRINT"OFF";
985 FOR DE=1 TO 1000:NEXT
988 GOSUB586
991 RETURN
994 ' SUB to report Disk Space FREE
997 GOSUB607:' Ready Line 22
1000 CO=1
1003 DN=D:GOSUB1150
1006 ' Repeat
1009 :GC(7)=0
1012:GOSUB7
1015 :IF K$="?" OR K$="/" THEN
GOSUB 1213: GOSUB607:PRINT"F>";
1018 :IF K$="C" OR K$="c" THEN GOSUB1042
1021 :IF K$="D" OR K$="d" THEN GOSUB1135
1024 :IF K$="K" OR K$="k" THEN GOSUB1111
1027 :IF K$="q" OR K$="Q" THEN GOTO1033
1030 GOTO1006:' Until "Q" or "q"
1033 CO=0:'Done.
1036 GOSUB586
1039 RETURN
1042 ' SUB Grans. Marked for COPY ?
1045 FOR ZC=0 TO 6
1048 :IF ZC=D THEN GOTO1060
1051 :FOR ZO=0 TO NF
1054 ::IF OC(ZO)=(48+ZC) THEN
GC(ZC)=GC(ZC)+NG(ZO)
1057 :NEXT
1060 :'Skip
1063 NEXT
1066 ATTR 6,1:GOSUB607:PRINT "COPY to DRIVE
NUM.>";
1069 ZC=-1
1072 'While
1075 ZC=ZC+1
1078 :IF GC(ZC)<>0 THEN PRINT"#";
1081 :IF GC(ZC)<>0 THEN PRINTUSING "# ";ZC;
1084 IF ZC<6 THEN GOTO1072
1087 GOSUB577:LOCATE 1,23:PRINT "NUMBER
of GRANULES >";
1090 ZC=-1
1093 'WHILE
1096 ZC=ZC+1
1099 :IF GC(ZC)<>0 THEN PRINT ;GC(ZC);
1102 IF ZC<6 THEN GOTO1093
1105 PRINT" F>";
1108 RETURN
1111 'SUB KILLFREE
1114 FOR ZO=0 TO NF
1117 :IF OC(ZO)=ASC("K") THEN GC(7)=GC(7)
+NG(ZO)
1120 NEXT
1123 GOSUB607:PRINT"There will be ";
1126 PRINTUSING "##";FREE(D)+GC(7);
1129 PRINT" Gran's FREE";
1132 RETURN
1135 ' SUB Grans. FREE on drive #n.
1138 PRINT"D";
1141 GOSUB1177:'Request Drive #.
1144 GOSUB607:GOSUB1150
1147 RETURN
1150 'SUB to print Granules Free.
1153 GF=FREE(DN)
1156 PRINT"Drive #";
1159 PRINTUSING "#";DN;
1162 PRINT" - ";
1165 PRINTUSING "##";GF;
1168 PRINT" gran's free";
1171 PRINT" F>";
1174 RETURN
1177 'SUB to SELECT a Drive.
1180 GOSUB7
1183 :IF K$="0" THEN DN=0:GOTO1207
1186 :IF K$="1" THEN DN=1:GOTO1207
1189 :IF K$="2" THEN DN=2:GOTO1207
1192 :IF K$="3" THEN DN=3:GOTO1207
1195 :IF K$="4" THEN DN=4:GOTO1207
1198 :IF K$="5" THEN DN=5:GOTO1207
1201 :GOSUB607:PRINT"Drive Number ?";
1204 GOTO1177
1207 'Done.
1210 RETURN
1213 'SUB Help Window
1216 POKE &HFFD9,0:' Set COCO to 2MHz.
1219 LOCATE 37,23:PRINT"^";
1222 LOCATE 37,23
1225 A1=INT(AT(CO)/10):A2=AT(CO)-(A1*10)
1228 ATTR A1,A2
1231 FOR I=0 TO WS(CO)
1234 :LOCATE 2,4+I
1237 :PRINTHH$(I+HO(CO));
1240 NEXT
1243 AK$=INKEY$:IF AK$="" THEN GOTO1243
1246 'SubSub Redo-Screen
1249 POKE &HFFD9,0:' Set COCO to 2MHz.
1252 CLS1:WIDTH40
1255 GOSUB1426
1258 FOR I=2 TO PL
1261 :LPOKE SA+FSL(I)+22+2,24:' Return highliting.
1264 :IF OC(I)<>32 THEN FOR J=FSL(I) TO
FSL(I)+22 STEP 2:LPOKE SA+J,1:NEXT
1267 NEXT
1270 ' Done.
1273 POKE &HFFD8,0:' Reset COCO to 1MHz.
1276 RETURN
1279 'SUB Initilize Help
1282 POKE &HFFD9,0:' Set COCO to 2MHz.
1285 CO=0:WS(0)=15:HO(0)=0:AT(0)=24
1288 WS(1)=7:WS(2)=5:WS(3)=10:WS(4)=5
1291 HO(1)=18:HO(2)=26:HO(3)=32:HO(4)=32
1294 AT(1)=24:AT(2)=25:AT(3)=33:AT(4)=33
1297 HH$(0) ="PRESS KEY      FOR          "
1300 HH$(1) =" 'arrow' Move arround screen  "
1303 HH$(2) ="    B    BACKUP use Scratch Dsk "
1306 HH$(3) =" 0123 C  Mark for COPY to Dsk # "
1309 HH$(4) ="   D    Set DEFAULT drives.   "
1312 HH$(5) ="   F    FREE disk space avail. "
1315 HH$(6) ="   G    GO do marked opps.    "
1318 HH$(7) ="  K   Mark file for KILL    "
1321 HH$(8) ="  L   LOAD current file.    "
1324 HH$(9) ="   Q    QUERRY mode setting "
1327 HH$(10)="   R    Mark file for RENAME "
1330 HH$(11)="  U   UNMARK the file      "
1333 HH$(12)="  V v  VERIFY on / verify off"
1336 HH$(13)="  W   run Wild card builder "
1339 HH$(14)="   X    eXecute the program "
1342 HH$(15)=" "
1345 HH$(16)="                       "
1348 HH$(18)=" At the F>  Prompt you can type "
1351 HH$(19)=" any of the fillowing commands. "
1354 HH$(20)=" D# - Disk Number # has ? free. "
1357 HH$(21)=" C# - Granules marked to copy "
1360 HH$(22)="     to disk number #.       "
1363 HH$(23)=" K -GranulesThat will be free"
1366 HH$(24)="     if marked files killed    "
1369 HH$(25)=" Q - Exit FREE utility to MAIN"
1372 HH$(26)=" At the Q>  Prompt you can type "
1375 HH$(27)="any of the fillowing commands. "
1378 HH$(28)=" 0  - To set for No Query.     "
1381 HH$(29)=" 1 -To set copy for No Query"
1384 HH$(30)=" 2 - To set all opps for Query"
1387 HH$(31)=" Q  - To Quit Query Set.     "
1390 HH$(32)="         WARNING          "
1393 HH$(33)="   THE DISK THAT YOU USE AS   "
1396 HH$(34)="    A SCRATCH DISK WILL BE    "
1399 HH$(35)="   OVERWRITEN BY THE BACKUP "
1402 HH$(36)="         OPERATION          "
1405 HH$(37)="                       "
1408 HH$(38)=" To Make a SCRATCH DISK type    "
1411 HH$(39)=" The Drive Number 1 2 ... 6  "
1414 HH$(40)=" To Kill All on Scratch type K"
1417 HH$(41)=" To End a disks service type E  "
1420 POKE &HFFD8,0:'COCO->1MHz.
1423 RETURN
1426 'LISTDIR
1429 PL=21
1432 FOR JJ=0 TO PL
1435 :C1$=CHR$(OC(JJ)):C2$=CHR$(OC(JJ+PL)):
C3$= CHR$(OC(JJ+2*PL))
1438 :PRINTF$(JJ)+"."+E$(JJ)+C1$+F$(JJ+ PL +1)
+"."+E$ (JJ+PL+1)+C2$+F$(JJ+2*PL +2)+"."+ E$
(JJ+2*PL+2)+C3$
1441 NEXT
1444 PRINT
1447 GOSUB586
1450 PRINT;"DRIVE#";D; "FILE:=";F$(N);"."+E$(N);
1453 PRINT;" ";T$(N);" ";A$(N);" ";
1456 PRINTUSING "##";NG(N);
1459 RETURN
1462 'SUB to Mark ALL Entrys
1465 FOR IX=0 TO 68:POKE 30200+IX,OC (IX): NEXT
1468 POKE 30199,89:'OC(N)FLAG.
1471 POKE 30198,D:'Note Drive
1474 CLS
1477 PRINT" INSERT RD-TOOLS DISK IN :";PD
1480 LOCATE 7,21:PRINT" PRESS ANY KEY TO
START";
1483 GOSUB7
1486 IF PD=0 THEN RUN "WILD:0",R
1489 IF PD=2 THEN RUN "WILD:2",R
1492 RETURN
1495 'INSTALL RAM DISK
1498 CLS:LOADM"RAM:"+CHR$(48+PD): EXEC:
CLS:GOSUB811
1501 POKE 30001,ASC("R")
1504 CLS
1507 RETURN
1510 ' eXecute a program.
1513 WIDTH32
1516 PRINTF$(N):IF E$(N)="BAS" THEN RUN
""+F$(N)+"",R
1519 IF E$(N)="BIN" THEN GOSUB 2000
1522 END
```

```
1525 ' Files 67 & 68 If they Exist.
1528 IF NF<66 THEN N=0:RETURN
1531 CLS:PRINT"DRIVE # ";D
1534 PRINT"  FILE #67:=";F$(66);"."+E$(66) ;" ";
T$(66);" ";A$(66);
1537 PRINT"  "+CHR$(OC(66))
1540 IF NF=67 THEN PRINT" FILE #68:=";
F$(67);"."+E$(67);" ";T$(67);" ";A$(67);"  "
+CHR$(OC(67))
1543 PRINT:PRINT" FILE # ";
1546 K$=INKEY$:IF K$="6" THEN PRINT" 6" ;
ELSE GOTO1546
1549 GOSUB7
1552 IF K$="7" THEN PRINT"7":F9=66:
GOSUB1561 ELSE IF K$="8" THEN PRINT"8":
F9=67:GOSUB1561
1555 N=0:GOSUB1246
1558 RETURN
1561 'DEAL WITH FILE
1564 PRINT" C K R or U >"
1567 GOSUB7
1570 IF K$="R" OR K$="K" OR K$="0" OR
K$="1" OR K$="2" OR K$="3" THEN OC(F9)
=ASC(K$)
1573 IF K$="r" THEN OC(F9)=ASC("R") ELSE IF
K$="k" THEN OC(F9)=ASC("K")
1576 IF K$="C" THEN PRINT" 0 1 2 3 >":
GOTO1567
1579 IF K$="U" OR K$="u" THEN OC(F9)=32
1582 RETURN
1585 'SUB New Program Disk
1588 GOSUB607
1591 PRINT" New Program Disk #>";
1594 K$=INKEY$:IF K$<>"2" THEN GOTO1594
1597 P0=PD
1600 PD=VAL(K$)
1603 POKE 30101,PD
1606 RETURN
1609 'SUB Copy Program Files From Current
Program Drive To New Program Drive.
1612 GOSUB607:PRINT"INSERT RD-TOOLS DISK
IN DRIVE 0"
1615 PRINT" AND PRESS ANY KEY";
1618 GOSUB7
1621 COPY "BLANK.DAT:0" TO "BLANK.DAT:2"
1624 COPY "MENU-EXT.DAT:0" TO "MENU-
EXT.DAT:2"
1627 COPY "RAM.BIN:0" TO "RAM.BIN:2"
1630 COPY "WILD.BAS:0" TO "WILD.BAS:2"
1633 COPY "BACKUP.BAS:0" TO
"BACKUP.BAS:2"
1636 COPY "SDSK512K.BAS:0" TO
"SDSK512K.BAS:2"
1642 CLS:PRINT"  PROGRAM FILES COPIED"
1645 PRINT"  TO PROGRAM DRIVE #2"
1648 PRINT:PRINT:PRINT"RETURN DATA DISK
TO DRIVE ";D
1651 LOCATE 1,22:PRINT"PRESS ANY KEY";
1654 GOSUB7
1657 GOSUB1246
1660 RETURN
1663 'LOAD A FILE.
1666 FE$=F$(N)+"."+E$(N)
1669 IF T$(N)<>"2" THEN LOAD FE$
1700 END
2000 REM exec a ML program
2010 FOR I=1 TO
8:C$=F$(N):A=ASC(MID$(C$,I,1)):POKE
30300+I,A:NEXT
2020 POKE 30300+9,SG(N)
2030 RUN "X",R
2040 RETURN
```

Program Listing for MAKBLANK.BAS:

```
10 'MAKBLANK.BAS - MAKE THE BLANK.DAT
FILE
20 ' (c) Charles R. Connolly 1994
30 ' First Serial Rights - FARNA Systems
40 ' All Other Rights Reserved
50 CLEAR 1000
60 FILES 3,768
70 INPUT " OUTPUT BLANK.DAT TO DRIVE # ";OD
80 IF OD=0 THEN OPEN "O",#1,"BLANK.DAT:0"
90 IF OD=1 THEN OPEN "O",#1,"BLANK.DAT:1"
100 INPUT " READ BLANK DISK IN DRIVE # ";BD
110 FOR SE=2 TO 11
120 :DSKI$ BD,17,SE,SL$,SH$
130 :PRINT "SECTOR ";SE:PRINT SL$;SH$
140 :WRITE #1,SL$,SH$
150 NEXT
160 CLOSE #1
170 PRINT
175 IF OD=0 THEN OPEN "D",#2,"BLANK.DAT:0"
180 IF OD=1 THEN OPEN "D",#2,"BLANK.DAT:1"
190 FOR SE=2 TO 11
200 :DSKI$ BD,17,SE,SL$,SH$
210 :PRINT #2,SL$
220 :PUT #2,SE*2-3
230 :PRINT SE
240 :PRINT #2,LEFT$(SH$,127)
250 :PUT #2,SE*2-2
260 NEXT
270 CLOSE #1:CLOSE #2
280 END
```

Program Listing for X.BAS:

```
10 'X.BAS
20 '
30 ' EXECUTE A ML PROGRAM OR TELL USER
HOW TO.
40 '
50 ' (c) Charles R. Connolly 1994
60 ' First Serial Rights - FARNA Systems
70 ' All Other Rights Reserved
80 '
90 PCLEAR 1
100 CLEAR 1000,10000
110 FILES 3,768
120 D=PEEK(&H95A)
130 F$="":Q$=CHR$(34)
140 FOR I=30301 TO 30308
150 :F$=F$+CHR$(PEEK(I))
160 NEXT
170 SG=PEEK(30309)
180 GOSUB 310
190 DSKI$ D,T,S,SL$,SH$
200 AH$=MID$(SL$,4,1):AL$=MID$(SL$,5,1)
210 AH=ASC(AH$):AL=ASC(AL$):A=AH*256+AL
220 EA=PEEK(27)*256+PEEK(28)-1:'END ADDRESS
OF THIS PROGRAM
250 IF A>EA THEN LOADM F$:EXEC ELSE
GOTO 275
255 PRINT
260 PRINT " PUT SDSK512K IN DRIVE 0:"
265 PRINT " PRESS ANY KEY"
267 K$=INKEY$:IF K$="" THEN GOTO 267
270 RUN"GO",R
275 ' SKIP
280 CLS:PRINT " type ":PRINT " LOADM"
+Q$+F$+Q$+":EXEC"
290 PRINT:PRINT " to run the program"
300 END
310 ' Translate SG to Track and Sector
320 T=INT(SG/2)
330 S=SG-2*T
340 IF S=1 THEN S=10
350 IF S=0 THEN S=1
360 RETURN
```

Program Listing for BACKUP.BAS:

```
10 'BACKUP.BAS - PROGRAM TO BACKUP
USING DSK I/O
20 ' (c) Charles R. Connolly 1994
30 ' First Serial Rights - FARNA Systems
40 ' All Other Rights Reserved
50 '
60 CLS 5:WIDTH 80:PCLEAR 1
70 CLEAR 8000,30000
80 DIM
C(10),SL$(19),SH$(19),TP(5),LL(75),EL$(12)
90 D=PEEK(&H95A):'Get Current DRIVE#.
100 PRINT "—= Disk Backup Utility =—":LOCATE
4,4:PRINT"Checking Memory ..";
110 SD=PEEK(30010)
120 FS=0
130 GOSUB 720
140 IF SD=0 THEN PRINT:PRINT "%E% ONLY
DRIVE 0 BACKUPS ARE SUPPORTED":
GOTO 640
150 IF SD=255 THEN PRINT:PRINT "%E%
SCRATCH DISK NOT SPECIFIED":GOTO 640
160 PRINT "  Place SOURCE disk in drive #";D
170 PRINT
180 PRINT" Backing Up drive ";D
190 PRINT"        to drive ";SD
200 PRINT:PRINT " DISK #";SD;" WILL BE OVER
WRITTEN"
210 PRINT "PROCEED ";
220 INPUT P$
230 IF P$<>"Y" THEN GOTO 640
240 FOR TR=0 TO 34
250 :LOCATE 5,19:PRINT "READING TRACK
#";TR
260 :FOR S=1 TO 18
270 ::LOCATE 5,20:PRINT "       SECTOR #";S
280 ::DSKI$ D,TR,S,SL$(S),SH$(S)
290 ::GOSUB 760:'Multitasking Editor.
300 :NEXT
310 :LOCATE 5,19:PRINT "WRITING TRACK #";TR
320 :FOR S=1 TO 18
330 ::LOCATE 5,20:PRINT "       SECTOR #";S
340 ::DSKO$ SD,TR,S,SL$(S),SH$(S)
350 ::GOSUB 760:'Multitasking Editor.
360 :NEXT
370 NEXT
380 CLS 5
390 PRINT "—= Disk Backup Utility =—
":PRINT:PRINT
400 PRINT "  Place DESTINATION disk in drive
#";D
410 PRINT
420 PRINT" Backing Up drive ";SD
430 PRINT"        to drive ";D
440 PRINT:PRINT " DISK #";D;" WILL BE OVER
WRITTEN"
450 PRINT "PROCEED ";
460 INPUT P$
470 IF P$="y" THEN P$="Y"
480 IF P$<>"Y" THEN GOTO 640
490 GOSUB 1830
500 FOR TR=0 TO 34
510 :LOCATE 5,19:PRINT "READING TRACK
#";TR
520 :FOR S=1 TO 18
530 ::LOCATE 5,20:PRINT "       SECTOR #";S
540 ::DSKI$ SD,TR,S,SL$(S),SH$(S)
550 ::GOSUB 760:'Multitasking Editor.
560 :NEXT
570 :LOCATE 5,19:PRINT "WRITING TRACK
#";TR
580 :FOR S=1 TO 18
590 ::LOCATE 5,20:PRINT "       SECTOR #";S
600 ::DSKO$ D,TR,S,SL$(S),SH$(S)
610 ::GOSUB 760:'Multitasking Editor.
620 :NEXT
630 NEXT
640 PRINT "EXITING"
650 GOSUB 1290
660 IF FS<>0 THEN GOSUB 1750
670 PRINT "  Place PROGRAM disk in drive #";D
680 PRINT "PROCEED ";
690 INPUT P$
700 IF P$<>"Y" THEN CLS: GOTO 670
710 RUN "SDSK512K:"+CHR$(48+PD),R
720 ' Initilize the Multi-tasking Editor.
730 T=0:
740 TP(0)=0:TP(1)=0:TP(2)=0:TP(3)=0: 'TASK
POINTERS
750 RETURN
760 ' Multi-tasking Editor.
770 IF TP(0)=0 THEN LOCATE 30,2:PRINT"   THE
MEMO EDITOR":TP(0)=1:RETURN
780 IF TP(0)=1 THEN LOCATE 30,3:PRINT"PRESS
[CLEAR] TO ACTIVATE":TP(0)=2:RETURN
790 IF TP(0)=2 THEN GOSUB 1220:IF
TK$=CHR$(12) THEN TP(0)=3 ELSE RETURN
800 IF TP(0)=3 THEN GOSUB 830:RETURN
810 IF TP(0)=4 THEN GOSUB 1720: TK$="":
TK=0:RETURN
820 RETURN
830 'OPEN MEMO PAD
840 IF TP(1)=0 THEN LOCATE 30,3:PRINT"
```

```
":TP(1)=1:RETURN
850 IF TP(1)=1 THEN LOCATE 30,2:PRINT"          ":
TP(1)=2:RETURN
860 IF TP(1)=2 THEN LOCATE 3,2:PRINT
"++========- MEMO PAD -===- F1-Clear F2-
Save -==R/C==- / -=========++":TP(1)=3:RETURN
870 LOCATE 3,3:PRINT "!!  '  1  '  2  '  3  '  4
'  5  '  6  '!!"
880 IF TP(1)=3 THEN GOSUB 920:RETURN
890 IF TP(1)=4 THEN LOCATE 3,16:PRINT"++ ==
(66 total "=" signs) ==++":TP(1)=5:RETURN
900 IF TP(1)=5 THEN TP(0)=4:TP(1)=0:RETURN
910 RETURN
920 'INITALIZE THE NOTEPAD
930CX=1:CY=1
940 IF TP(2)=0 THEN GOSUB
1130:TP(2)=1:RETURN
950 IF TP(2)>0 THEN IF TP(2)<13 GOSUB 1180:
[illegible] TP(2) [illegible] RETURN
[illegible] THEN TP(1)=4:TP(2)=0:RETURN
970 RETURN
980 'EDITING FUNCTIONS
990 GOSUB 1220:' GET TK$
1000 IF TK$="" THEN RETURN
1010TK=ASC(TK$)
1020 'AUTO ADVANCE
1030 IF TK=9 THEN GOSUB 1490:RETURN
1040 IF TK=8 THEN GOSUB 1550 :RETURN
1050 IF TK=10 THEN GOSUB 1650:RETURN
1060 IF TK=94 THEN GOSUB 1610:RETURN
1070 IF TK=4 THEN GOSUB 1290: RETURN:
'F2=SAVE
1080 IF TK=13 THEN LOCATE 5,3+CY:PRINT EL$(
CY)+"!!";:CX=1:TK=10:GOTO 1020
1090 IF TK=103 THEN TK=PEEK(343)
1100IFTK=191THENGOSUB1400ELSETK=103
1110IFTK>32ANDTK<128THENGOSUB
1440:GOTO1020
1120RETURN
1130 'CLEAN SLATE
1140FORI=1TO12
1150 EL$(I)="                              "
1160NEXT
1170RETURN
1180 'DISPLAY THE NOTEPAD ON SCREEN
1190IX=TP(2)
1200LOCATE3,3+IX:PRINT"!!"+EL$(IX)+"!!";
1210RETURN
1220'INKEY WITH TIME OUT
1230 T=0:PR=16:'PRIORITY
1240TK$=INKEY$:T=T+1:IF T>PR THEN GOTO
1260
1250 IF TK$="" THEN GOTO 1240
1260 'DONE
1270T=0
1280 RETURN
1290 'SAVE
1300FS=FS+1
1310LOCATE 18,18:PRINT "FILE NAME/
EXTENSION";
1320 INPUT N$
1330 OPEN "O",#1,N$+":2"
1340FORI=1TO11
1350 :PRINT #1,EL$(I)
1360NEXT
1370 CLOSE #1
1380 LOCATE 18,18:PRINT "                    ";
1390 RETURN
1400 'CLEAR LINE
1410 EL$(CY)="                              "
1420LOCATE5,3+CY:PRINT EL$(CY)+"!!";:CX=1
1430 RETURN
1440 'SUB TO PUT A CHARACTER IN THE
EDITOR.
1450 LOCATE 4+CX,3+CY:PRINT TK$;
1460EL$(CY)=MID$(EL$(CY),1,CX-
1)+TK$+MID$(EL$(CY),CX+1,67-CX-1)
1470TK=9
1480 RETURN
1490 'MOVE AHEAD
1500CX=CX+1
1510IFCX>65THENCX=1:CY=CY+1
1520IFCY>12THENCY=1
1530 TK=0:TK$=""
1540 RETURN
1550 'MOVE BACK
1560CX=CX-1
1570IFCX<1THENCX=64:CY=CY-1
1580IFCY<1THENCY=12
1590 TK=0:TK$=""
1600 RETURN
1610 'MOVE AHEAD A LINE
1620CY=CY-1:IFCY<1THENCY=12
1630 TK=0:TK$=""
1640 RETURN
1650 'MOVE BACK A LINE
1660CY=CY+1:IFCY>12THENCY=1
1670 TK=0:TK$=""
1680 RETURN
1690 'UPDATE CURSOR LOCATION
1700 LOCATE 58,2:PRINT USING "##/##-========
++";CY,CX;
1710RETURN
1720 'GET KEYS AND REPORT POSITION.
1730 GOSUB 980:GOSUB 1690
1740 RETURN
1750 'Disk Change WARNING.
1760 WIDTH 32:PRINT:PRINT "YOU HAVE
SAVED":PRINT " ";FS;" FILES"
1770 PRINT "YOU WILL BE REROUTED"
1780 PRINT "    TO DRIVE 2"
1790 PRINT "TO COPY THESE FILES"
1800 PRINT "TO A PHYSICAL DISK."
1810DRIVE2
1820 RETURN
1830 'Refresh Editor Screen.
1840 LOCATE 3,2:PRINT "++========- MEMO
PAD -===- F1-Clear F2-Save -==R/C==- / -=====
===++"
1850 LOCATE 3,3:PRINT "!!  '  1  '  2  '  3  '  4
'  5  '  6  '!!"
1860FORIX=1TO11
1870 :LOCATE3,3+IX:PRINT"!!"+EL$(IX)+"!!";
1880NEXT
1890 LOCATE 3,16:PRINT "++=(66 total "=" signs)
===++";
1900 RETURN
```

Listing of file MENU-EXT.DAT.
Create this file with a word processor or text editor and save as MENU-EXT.DAT.

```
* This is the Menu
* File for The
* Extensions Menu
.BAS
.BIN
.CM3
.DAT
.DOC
.GIF
.MGE
.TXT
*End.
```

The program listing for WILD. BAS will be in the December issue. I apologize for the inconvenience, but these were LONG listings!

There will be some problems where blank spaces are shown. The user will have to run the programs then go back and adjust the number of spaces needed. This problem was unavaiodable. A font had to be found that was small yet easy enough to read.

< 268'm >

Letters to the Editor

continued from page 3

Since half the readers claim use of an MS-DOS machine, it's unfortunate to be missing the stats for DOS and Windows useage. It might be interesting to see how effectively RAM isutilized under OS-9/OSK versus MS-DOS and Windows. I have heard some real horror stories on this.

Now that old XT hard drive controllers are becoming so hard to find, what about using a SCSI drive with my Disto HD Interface? How hard will XT controllers be to find a couple years?

I think you would be doing us a real favor with some in-depth discussion of the PowerMac line. This could be a follow-up on you MPC601 article of about a year ago.

Henry Harwell
2110 West Roma Ave.
Phoenix, AZ 85015-4445

Henry, I didn't think the difference between DOS only use and Windows use would be of particular importance. I have discovered that many CoCo owners (probably about 1/3) use DOS only, on XT and some AT class machines, but not so many 386+ platforms.

The XT controllers for the B&B units are not that hard to find. Pick up a copy of Computer Shopper or Nuts&Volts. Several surplus vendors always have them.

The Disto HD will support only one SCSI drive. All you need do is make a cable for it. The Disto drivers only support 256 byte sectors. You will need Matt Thompson's SCSI-SYS drivers to use a standard PC type drive (512 byte sectors.. available from Northern Xposure). To use the Disto drivers, you need a Seagate N series SCSI drive with ROM revision 105 or greater. Rodime 65x drives also support 256 byte sectors.

I'm not sure a review of the PowerMac would be in the best interest of this magazine. Microware has announced a PowerPC version of OSK, but only in a developers pack. When a version becomes available for end users I'll review the platform and OS together.

Congratulations on the one year anniversary! One quick question: Is a serial to parallel convertor a box with electronics in it or just a cable?

D. Concepcion
Korea

Dan, the convertor has electronics. I'm planning a project on making one in the future. Until then, write CoNect and see if Rick has a used on for sale.

< 268'm >

Operating System -Nine

Russell E. Hoffman, II

Permanently change drive step rate, frequently asked questions.

The Trees!

I find myself writing an article like this every so often. With a complex operating system like OS-9, it's so easy to get caught up in details..... but to really appreciate a tree one must occasionally step back to look at the forest.

Typically the basic problem of running the same application on a wide variety of machines with different capabilities isn't handled very well. Apple doesn't even try (in fact, strongly discourages any attempt). MS-DOS allows a platform to vary only as far as can be described in a relatively small piece of firmware, one reason these boxes are referred to as 'clones'.

More robust operating systems typically provide a 'virtual machine' of known capabilities, which each platform then emulates as best it can. Obviously easy to port to, and very secure since applications normally have no direct access to the hardware. But...

If familiar with computing, you already know anytime the words virtual and emulate appear in a sentence, the process being described is stone slow. The result is similar to interpreted basic- lots of code slinging that doesn't really have anything to do with the program itself. As a result, systems like OS-9 and Unix have a reputation for requiring more hardware power for a given result.

Which brings us to the CoCo. Many cannot believe a machine like this can possibly do half we claim- and even if it could, surely not today. The opposite is also true- after banging around level 2 for a few years a small OSK machine can be something of a let-down. The CoCo just isn't as much slower as it should be, relative to 680x0 systems. In fact, it preforms better than an 808x or 80186- which is puzzling, since this is a direct comparison and MS-DOS should be faster!

Don't dis our DAT

The greatest problem with 6809 systems is address range, and it comes up again and again. A simple DAT (dynamic address translation -- the memory management system used by Level II) can increase the user space, but the switching routines can be incredibly complex, and the graphics screen has to be visable at all times for screen refresh, which is pretty slow over the byte sized data buss. In fact, the CoCo2 expended an immense amount of ram- up to a quarter of the total, and an identical amout of buss cycles, just to maintain pmode4. (Actually drawing something is an extra cost option).

The minimum acceptable screen is now 640x200, which is a whopping 32K of data and obviously wouldn't fit in space or time, except for the GIME. You may have noted the normal mode of operation has the GIME discarding one byte from each 16 bit RAM read, to feed the desired 8 bits on to the cpu. When it's doing a screen refresh, this is not the case, and it paints twice as fast as it ought to. Changing the screen still uses plenty of slow 8 bit transfers, but there is more time for this with the rapid repaint.

Of course, we can't ignore the logical problems. The cpu only addresses 64K, and fitting opsys, app, and graphics in this space is difficult — no wonder Level 1 was such a pain! — and here, the GIMEs second memory map comes into play. The functional definition of 'Level 2' OS-9 is that it exceeds the cpu's RAM limit (OSK is 'level 1'again, but with megs not k). The CoCo 3s mapping system is almost perfect for a system like OS-9, which already has a definate division between application and system code. Add the CoCo's divison of code and graphics, and you can triple the amount of ram effectively used, to 192K.

Communication between the three maps will be a problem, and the greatest amount of byte slinging involves the graphics system. Ever wonder why GrfDrv is loaded from disk and not in os9boot? That's so it can copy over to map 2 with the screen it's massaging! With the driver and the screen in the same map, there is no problem communicating, and the driver can use the entire cpu range that's not data- the best graphics that can possibly be drawn with a 6x09.

We still need to get commands to the driver, and there has to be a lot of talk between the application and the system. There are a few ways to accomplish this.

The first possible way is to share. Since a Motorola always maps it's hardware as RAM, some address space is going to have to be shared anyway, and there must always be enough code present to switch back to the system map if something comes up in the hardware. On a CoCo, the large 8K data blocks mean only one block is needed for all three functions.

The second way to swap data is by using the cpu's internal registers. Load up the cpu just like the code you want to run existed, then swap maps. The new code acts on the cpu registers as if some previous code (that doesn't exist now) had set them up, then swaps back. Syscall sometimes uses additional data areas but you get the idea. These two base methods are used to impliment the pipes and signals we usually talk about.

You'll note we ran out of GIME register sets awhile back, but it's not that hard to move new data into them- OS-9 keeps tables of DAT images, using a new app map 10 times per second. The actual screen displayed is independent of this 10 per second switching (in fact, pretty much independent of the cpu at all).

Multiple application maps allow multitasking, but they could be wasteful. For example, when writing code for a single user, the fastest, easiest way to store variables is to define an area of RAM inside the program for temporary storage. The problem, of course, is a second user must load another copy of the code for separate storage space.

Re-entrant code doesn't define any specific spot for storage, instead it uses offsets from an undefined index point. Since nothing is ever changed inside the code block, it can be shared by giving each user their own index, and mapping the same physical block into each users space. Which means one can logically use more ram than he physically has (That's a quote, folks :-)).

To pull this off, OS9 always loads files at the start of a DAT block. A file can have more than one module in it, but they are treated as a single block for memory mapping/sharing. The block size used by CoCo is relatively coarse, which both runs faster and is cheaper to build, but can waste space (at 8K minimum per file) unless the user organizes smaller modules into 'merged files'. So now and know you know!

The other shoe drops- After subtracting the common block, there are 7 blocks left in a 64K process map. Usually, there is a shell underling the application, so subtract another block for program and data,leaving five. Use that disk util file and there are 4 blocks left, and we haven't gotten to the app yet.

A real common example- runb, gfx2, inkey, and syscall use a block each. Merged together, they only fill two, so usually these modules are merged, since many programs can't afford 4 blocks. Some might not be able to afford two, which explains MWares policy of distributing the smallest possible parts- if you just need runb, you can just load runb.

Every benefit of Level 2 brings it's own cost- for example, Level 2 programs don't need as much stack space, since much of the stacking is over in the system map. This can be a heavy load, like when format sets up it's 6.5K buffer (in the system map). Every user has to pay all the time so some can format a floppy occasionally. Anytime a device driver wants a buffer, that comes from the system map also, so a big OS-9 with a big serial port buffer doing a floppy format just might release some of the vital smoke.

Note that while there is a good line of communication between app, sys, and grf, there is no easy way to connect two applications. One way is with pipes- these tie two disconnected maps together by a buffer in system map and let the output of one feed the input of another. Great- another buffer in the system map.

Another idea is to share data blocks, like we already share code. Obviously more difficult since data can change, but data modules are a staple in OSK, and they can be used on the CoCo as well!

Which brings us to the disk of next month- VRN is the key to performing many tasks not possible under the system just described. Tandy products like Kings Quest and Flight Simulator 2 use a similar scheme, and running them under your normal boot is impossible....On a serious side, future programs are going to be OSK sized, and any CoCo port will likely need the room (or be impossible)!

All you really need to play with the concept is a copy of VRN from the online services, my disk of the month, or microdisk. Installation requires a new boot, and all that implies, so I'll sweeten the $5 disk of the month to include vrn, kq3/lsl/fs2 mvue stuff, and kq3 or fs2 (your choice, while supplies last) so you get some use from the new boot. Next month, we'll talk about it.

The Future of OS9?

One almost hates to admit it, but it looks like CD-I players are going to catch on despite being an OS-9 product. I base this decision on the newer players, which are made a little cheaper, but can realistically be sold at a price folks will pay. This is what happens to successfully emerging products... much like when Radio Shack released the CoCo 2, which dropped price to the magic $300 (now $500) buy-in limit for most consumer electronics, ensuring the CoCo would live a while, even if it was built a little on the cheap side.

This applies to us since CD-I is based on a 68070 running an extended OS-9 called RTOS (Real-Time Operating System... editor's note: I'm not so sure of this, Rick will have to do some more homework on RTOS). Base RAM is usually a megabyte, with another Meg in the FMV (full motion video) cartridge. Rumors of a console adapter (to add keys and drives) continue, but some didn't wait. Boisy Pitre copied his MM/1 drive to a cd, then performed the time honored OS-9 ritual of connecting a terminal to /t2 and selecting a termcap. The more things change, the more they stay the same. And this is what he found:

```
Module Directory at 21:40:35

kernel      cio        FONT8X8   pipeman
nrf         ucm        cdfm      scf
math        copyright  init      sysgo
t2          u68070     ds1216    tim070
tim070driv  nil        null      pipe
nvdrv       nvr        video     vid
vd2         vdk        v12       v96
msuart      pt2        gtuart    gt12
gt96        ckeydriv   ckey      pck2driv
cdivolset   pck2       kb1driv   kb1
drvdsp      ap         cd        csd_220f3
csdinit     config     kbdrvr    kb
pointer     ptr        sldriv    slave
sv          launcher   dspdriv   play
ps_bck      ps_data2   ps        ps_data
font1.ft    font3.ft   cdgr      dummy
spawnshell  shell      mdir      csl
```

There are parts of this even a CoCo owner could love (remember Tandy's penchant for renaming everything cc3 something- i.e. cc3io instead of cio). Those already running OS9/68K have already figured out what they need to upgrade 2.4.....programmers are heartened to see pieces of both Microware C compilers. For emphasis, lets ident a CD-I game:

```
Header for:       cdi_hotel
Module size:      $1B958    #112984
Owner:            0.0
Module CRC:       $F004C    Good CRC
Header parity:    $8841     Good parity
Edition:          $7        #7
Ty/La At/Rev      $101      $8001
Permission:       $555      ------e-r-e-r-e-r
Exec off:         $52       #82
Data size:        $2600     #9728
Stack size:       $C00      #3072
Init. data off:   $1B940    #112960
Data ref. off:    $1B948    #112968
Prog Mod, 68000 obj, Sharable
```

Obviuosly, the version of OS-9 running on these players is close enough to what we are familair with to be used. There are even more similarities. With the same cpu and video chip as the mm/1 (2 video chips in the CD-I players) and very little RAM, 1 or 2 megabytes, we have exactly the machine K-Wwindows was written for. Which means we may yet see another release of the old CoCo window system.

The return of the AIF........

Just to prove you can teach an old dog new tricks, I went and learned one. It ties in neatly with some problems users are still having with MultiVue, so a quick review......

If you have a hard drive or RAM disk and you want it to appear on the device bar at left, edit /dd/sys/env.file. This file sets up quite a bit of the system, and can be adjusted either by using MVue utilities like Control and Printer, or just grab a trusty text editor and start whacking. In this case you'd just extend the line RBFDEV to perhaps RBFDEV=/d0,/d1,/h0,/r0. These devices can be placed in any order, with the first displayed on top.

Page 9-28 and 9-29 explains what each line does, and if you are a get it done kinda person you might want to adjust everything while editing. Any line that srtarts with a * is considered a comment, which explains the extra RAM lines you'll find in the stock file. Tandy felt c/128/512/ to difficult a procedure for the average user.

Even if you don't use MultiVue, control makes a handy standalone utility. Add control -e to your startup file to reconfigure the system during boot. Should you want to change anything, run control and change it! Done. Incidentally, there is no need for the backflipping described in the manual to run a 16 color control- just get the patched gshell, then select a 16 color main screen.

Setting up individual programs to run with a click is relatively simple after obtaining some sort of icon editor. This task can be complicated by the fact most icon editors are really demo programs for alternate or extended gfx systems. I've got versions that require guib, gfx3, modified gfx2, or a modified windint. Read file descriptions carefully before selecting one! There are a few that run on anything, mainly the older ones. With an icon, the rest is simply a text file.

The easiest way to explain this is to look at how a few command lines split up. Here are some variations:

```
lha x file.zip
runb myprogram #32k
myprogram
```

	AIF.lzh	AIF.run	AIF.unb
program	lha	runb	myprogram
parameter list	x	myprogram	...
icon	...	...	...
ram	...	32	...
screen type	...	...	...
width	...	...	...
length	...	...	...
background	...	...	...
foreground	...	...	...

You'll note the first example (AIF.lzh) has some information missing, namely the file to dearc. This AIF is not designed to be clicked on itself- instead, it will be tied to the .lzh file extension and it's icon will appear over any .lzh file. A double click and it's done!

continued on page 25

Programming in "C"
The big SWITCH

P.J. Ponzo

You may recall, from an earlier lesson, that we checked various cases by using the if-else if construction:

```
1     if ( such-and-such)  {
2     ——do this——
3     }
4     else if (this-or-that) {
5     ——do this——
6     }
7     else if ( whatever )  {
8     ——do this——
9     }
10     else {
11     ——do this——
12     }
13    ——continuation of program
```

Realize that only one of the statements 2, 5, 8, 11 will be executed, depending upon which of the conditions 1, 4, 7, 10 is satisfied first. (If none of 1,4 or 7 are satisfied, then 10 IS satisfied and 11 will be executed). Even if the conditions 1 and 4 and 7 are all satisfied, only statement(s) 2 will be executed, then the program will continue with statement 13, etc. (There's a MORAL here. To speed up execution, put the most probable condition first ... then the program won't have to do so much checking). But there's another (more natural) way of checking a number of cases in C.

```
 switch (x)  {
 /* Begin the SWITCH on the integer x.*/
 case 1:     /* If x is the integer 1, then*/
    do this;   /*execute this statement */
      and this;    /*  and this too. */
 case 2:    /* If x is the integer 2, then */
    do this;  /*  execute this statement*/
      and this;     /*   and this too. */
 case 3:    /* If x is the integer 3, then  */
    do this;  /*  execute this statement*/
      and this;     /*   and this too. */
      and this;     /*   and this too. */
      and this;     /*   and this too.  */
 case 4:     /* If x is the integer 4, then  */
    do this;     /*  execute this statement*/
 default:  /* If x is none of the above, then*/
    do this;    /* execute this statement */
 }
```

Notice the opening and closing brackets for the SWITCH!

```
 switch (x)  {
 }
```

... and here's a variation:

```
1  switch (x)  {  /* Begin the SWITCH*/
                  /*  on the integer x.  */
2  case 1:        /* If x is the integer 1, then*/
3  case 2:        /* If x is the integer 2, then*/
4  case 3:        /* If x is the integer 3, then*/
5      do this;   /*execute this statement */
6      and this;      /*  and this too.  */
7      and this;      /*  and this too.  */
8      and this;      /*  and this too.  */
9  case 4:       /* If x is the integer 4, then*/
10    do this;   /*  execute this statement*/
11 default:  /* If x is none of the above, then*/
12    do this;   /*  execute this statement*/
13  }
```

If the integer x is equal to 1 or 2 or 3 then statements 5 to 8 will be executed! (...so the SWITCH will NOT STOP with the first case that is satisfied, but will check ALL SUBSEQUENT CASES!) If you don't want that to happen, then you may terminate a case with a break.

```
1   switch (x)  {
 /* Begin the SWITCH on the integer x.  */
2   case 1:       /* If x is the integer 1, then*/
3   case 2:       /* If x is the integer 2, then*/
4   case 3:       /* If x is the integer 3, then*/
5      do this;   /*  execute this statement*/
6        and this;     /*   and this too. */
7        and this;     /*   and this too. */
8        and this;     /*   and this too.  */
9        break;   /*and now BREAK OUT/*
                  /* OF THE SWITCH!  */
10  case 4:       /* If x is the integer 4, then*/
11     do this;   /*  execute this statement*/
12  default:  /* If x is none of the above, then*/
13     do this;   /*  execute this statement*/
14   }
```

Now, if x is a 1 or 2 or 3, the statements 5 to 8 will be executed and (because of the break; in line 9) we leave the SWITCH and continue beyond line 14. If, however x is a 4, then only the statement(s) for this case are executed (line 11, in this example). Only if all cases fail will the default statement(s) be executed (for example, if x is a 6 then statement 13 is executed).

More SWITCHIng

You may switch on any type of variable (not just integers). For example you may have declared x to be a char, so ...

```
1    switch (x)  {
       /* Begin the SWITCH on the char x. */
2   case 'A': /* If x is the character 'A', then*/
3   case 'u': /* If x is the character 'u', then*/
4   case '#': /* If x is the character '#', then*/
5      do this;    /*  execute this statement*/
6        and this;     /*  and this too. */
7        and this;     /*  and this too.*/
8        and this;     /*  and this too.  */
9        break;
        /*and now break out of the switch!*/
```

Note that the case comparison must be consistent with the variable type.
If x is an int then you may use case 7:
If x is a char then you may use case '+':
If x is a float then you may use case -1.234:

Think of the case comparisons as being equivalent to:

if(x==7) or if(x=='+') or if(x==-1.234) etc. ... and you may leave out the default if you wish!

CALL BY VALUE and REFERENCE

We mentioned in an earlier lesson that a function call, in which you pass certain parameters (like average(a,b)), gives to the function copies of the parameters. The function may change these copies but the "originals" won't be changed. This is CALL BY VALUE.

You may, however, WANT to have a function change the originals. In this case you must tell the function where, in memory, the "originals" live. To do this you may pass the addresses of the parameters (or ~rpointers to the parameters). This is CALL BY REFERENCE. Knowing where the "original" parameters are, in memory, a function may now modify them.

Suppose you want to exchange() the values of two floating point numbers, say x and y, by calling upon a function exchange() :

```
exchange(&x,&y);
 /* call the function, give it addresses of x,y*/
```

The exchange funtion may look like:

```
exchange(u,v)
  /* this function exchanges two "floats".*/
float *u, *v;
  /* declare u and v as pointers to "floats".*/
{     /* the opening bracket for exchange().*/
  float temp;   /* declare a temporary float*/
    temp=*u;
    /* make it equal to "what u points to". */
    *u=*v;
          /* place the contents of  v into u. */
    *v=temp;
     /* place the "temp"orary float into v.*/
    return;
    /* return ... no need to return anything! */
}      /* the floats have been exchanged !!*/
```

You can try it out with:

```
main()  {
   float x=1.23, y=4.56;
       /* declare and define two floats    */
     printf("\nx=%f, y=%f",x,y);
                     /* printf their values.  */
   exchange(&x,&y);
            /* call the exchange program.  */
     printf("\nx=%f, y=%f",x,y);
                /* printf their values again! */
}               /* that's the end of main().  */
exchange(u,v)
  /* this function exchanges two "floats"*/
```

```
float *u, *v;
  /* declare u and v as pointers to "floats"*/
{    /* the opening bracket for exchange()*/
   float temp;
             /* declare a temporary float */
   temp=*u;
    /* make it equal to "what u points to" */
   *u=*v;
        /* place the contents of v into u. */
   *v=temp;
    /* place the "temp"orary float into v.*/
   return;
  /* return ... no need to return anything! */
     /* the floats have been exchanged !!*/
```

Now exit the text editor, saving the above with the name sam.c, then compile using cc sam, then link using link sam, then execute via: sam, and get:

```
   x=1.230000, y=4.560000
   x=4.560000, y=1.230000
```

Here's the exchange() function again:

```
exchange(u,v)
  /* this function exchanges two "floats"*/
float *u, *v;
  /* declare u and v as pointers to "floats"*/
{    /* the opening bracket for exchange()*/
  float temp;  /* declare a temporary float*/
   temp=*u;
    /* make it equal to "what u points to"*/
   *u=*v;
          /* place the contents of v into u.*/
   *v=temp;
    /* place the "temp"orary float into v. */
   return;
   /* return ... no need to return anything!*/
}    /* the floats have been exchanged !!*/
```

Here's another variation:

```
exchange(u,v)
  /* this function exchanges two "floats"?*/
float *u, *v;
  /* declare u and v as pointers to "floats"*/
{       /* the opening bracket for exchange()*/
   float *temp;
          /* declare a temporary pointer.*/
   temp=u;
       /* make it equal to the pointer "u"*/
   u=v;
/*make "u" point to what "v" points to */
  v=temp;          /*make "v" point to what*/
                     /*  "temp" points to */
   return;
   /* return ... no need to return anything!*/
}   /* the floats have been exchanged ??*/
```

Why won't the latter function work???

```
exchange(u,v)    /* this function does NOT*/
                 /* exchange floats! */
float *u, *v;
 /*declare u and v as pointers to "floats" */
{    /* the opening bracket for exchange()*/
   float *temp;
           /* declare a temporary pointer*/
   temp=u;
       /* make it equal to the pointer "u"*/
   u=v;
/* make "u" point to what "v" points to*/
   v=temp;        /* make "v" point to what*/
                  /* "temp" points to.   */
   return;        /* return.    */
}      /* the floats have not been exchanged!*/
```

In this variation, the pointers u and v are copies and, although this function does change these copies of the pointers, their contents do NOT change! (so the floats never do get exchanged!).

Passing FUNCTIONS to FUNCTIONS

In an earlier lesson we computed the roots of some equation x=f(x), with f(x)=2*sin(x).

```
1  double x=1.0, y, e; /* double precision */
2  do {              /* start of the do-loop*/
3     y=2.0*sin(x);      /* calculate y */
4     e=fabs(y-x);       /* calculate error */
5     x=y;               /* change x to y*/
6  } while (e>.0000005); /*end condition*/
7  printf("x-2sin(x)=%f when x=%f",e,x);
```

Now suppose we turn this piece of code into a function, solve() which we call via: root=solve(f,x,e);
where we pass to solve() the function f(x), and some initial guess of the root, namely x, and an error specification e. We expect solve(f,x,e) to return a root (which, naturally, we call root!). We may write solve() like so:

```
1 float solve(fcn,x,error)
                      /* returns a FLOAT!*/
2 float (*fcn)();     /*  !!!!!!!!!!!!!!!!*/
3 float x, error; /* x-value & error are floats*/
4 {
5   float y, e;          /* declares 2 floats.*/
6   do {              /* start of the do-loop*/
7     y=(*fcn)(x);        /* calculates y. */
8     e=fabs(y-x);
       /* calculate absolute value of y-x.*/
9     x=y;               /* change x to y.  */
10  } while (e>error);
         /* check error if e is too large. */
11 return(x);  /* return x=root if e<=error*/
12}
```

Line 2 has the curious declaration of fcn() as a ~rfunction pointer. The (*fcn) says it's a pointer, and the () says it points to a function and the float says this fcn returns a float! Note too, in line 7, that whereas fcn is a pointer, *fcn IS the function! (The parentheses are necessary).

```
main()  {
   float f1(), f2(), f3(), solve();
                   /* declare functions used.*/
   printf("\nA root of x=f1(x) is %f",
                    /* printf the root ...   */
   solve(f1,1,.00005)); /* solve x=f1(x).*/
   printf("\nA root of x=f2(x) is %f",
                    /* printf the root ...   */
   solve(f2,-1,.00005)); /* solve x=f2(x)*/
   printf("\nA root of x=f3(x) is %f",
                    /* printf the root ...   */
   solve(f3,2,.00005)); /* solve x=f3(x)*/
}
float f1(x)
float x;
{ return(2.*sin(x)); }  /* f1(x) = 2 sin(x)*/
float f2(x)
float x;
{ return(2.-x/2.); }   /* f2(x) = 2-x/2 */
float f3(x)
float x;
{ return(1.+1./x); }  /* f3(x) = 1+1/x*/
float f1(), f2(), f3(), solve();
                /* declare functions used.*/
```

Here we declare all the functions we use (they all return a float).

```
   float f1(), f2(), f3(), solve();
                 /* declare functions used.*/
   printf("\nA root of x=f1(x) is %f",
                   /* printf the root ...   */
   solve(f1,1,.00005)); /* solve x=f1(x)*/
```

Here we print (after a \newline). A root of x=f1(x) is followed by the %float returned by solve(f1,1,.00005). Note that we pass the pointer f1, a starting value 1 and an error specification of .00005

```
   printf("\nA root of x=f1(x) is %f",
                   /* printf the root ...   */
   solve(f1,1,.00005)); /* solve x=f1(x).*/
```

... then we continue with two more functions f2(x) and f3(x), each time specifying not only the pointer to the function but also a starting value and error specification.

REMEMBER: To pass the function sam(a,b,c) as an argument to another function george(sam,x,y), then include the declaration float (*sam)() (make this declaration within george()) and use it (within george()) as (*sam)(a,b,c). If sam() returns an int or char then (of course) it should be declared as int (*sam)() or char (*sam)()!

```
A root of x=f1(x) is 1.895475      Here's
A root of x=f2(x) is 1.333324      our
A root of x=f3(x) is 1.618026      output.
```

And (because we use only float and not double, and we gave an error specification of .00005) we get (roughly) 4 decimal place accuracy.

Well, the programming ain't too sexy (how useful are these 3 built-in functions, f1(x), f2(x) and f3(x) ?) and the mathematics are even worse (you can't guarantee that the program won't get stuck in the solve() function ... forever trying to reduce a growing error!), BUT ... we get the idea ... right?

< 268'm >

P.J.Ponzo
Dept. of Applied Math
Univ. of Waterloo
Ontario N2L 3G1

Basic09 In Easy Steps

Chris Dekker

DECB, Basic09, and MS-DOS Qbasic conversions!

As promised in part 6 this time we will look at the commands embedded in the "Disk" ROM of DECB and how to deal with them under Basic09. The "disk" ROM honors it's name in more ways than one. Not only is it (physically) situated inside your floppy disk controller so you won't be able to use it until you have (upgraded to) a disk based system; it also is almost entirely devoted to executing commands dealing with access to disk drives.

Under OS-9 this whole concept becomes obsolete because OS-9 doesn't use any of the ROMs inside your system. All code dealing with disk access and functions is spread out over a variety of managers, device drivers, device descriptors and utilities. This isn't done to make the system hard to understand for people, but to make it easier to adapt to a variety of hardware setups.

For Basic09 this means that all commands you give it that need access to a disk (or other input or output device) are not handled by Basic09 itself but passed on to OS-9. There are basically three ways in which you can tell your application to do a certain job, although sometimes one of them is certainly preferred over the others.

The first way is to use commands embedded in Basic09. For example **OPEN, CREATE, DELETE,** etc. In this case Basic09 will gather and check the information necessary to execute an OS-9 system call and pass the information on. In this way OS-9 becomes completely transparent since the user has no way of knowing whether Basic09 executed it's own code or something else.

The second way is to use the **SYSCALL** utility. As described in part three of the series you can use this utility by issuing a **"RUN syscall"** statement. In this case your program must supply and check the information passed to OS-9 for executing a certain system call. Although this is a little more work it also allows you access to functions not accessible through corresponding Basic09 statements, thus complementing and enhancing Basic09.

The third way is through the use of Basic09's **SHELL** command. In this case OS-9 starts a new shell to execute whatever command Basic09 passes on to it. This way is usually preferred for launching other processes, calling and/or loading programs, utilities, etc.

Now the conversions. Following is a list of commands that, under OS-9, are included in the system as separate programs called utilities: **BACKUP, COPY, DIR, KILL, LOAD, MERGE, RENAME** and **SAVE.**

Generally speaking you will use these commands most often from the OS-9: prompt. However there may be cases in which you will want to run them from within a program. For instance when your application uses temporary files during execution and has to clean up afterwards. Under DECB you would rename a file as follows: **RENAME oldfile TO newfile.**

In Basic09 this becomes: **SHELL "rename oldfile newfile".** In this example "oldfile" and "newfile" are presumed to be literal names. If they represent string variables the command would look like this: **SHELL "rename" "+oldfile+" "+newfile.** Of course your program must take care of two things here: A) the variables contain valid names and B) your current working directory is the directory where these files are located.

This second problem doesn't exist with DECB because it keeps all it's files in one directory, which is always both your working and execution directory. Under OS-9, generally speaking, your working directory holds your data files, while your execution directory holds your programs, system utilities, etc. When you boot up OS-9 will set the execution directory to CMDS and your working directory to the root directory of your boot disk. On a floppy system this is typically called /d0.

Although DECB has no equivalents I do want to mention the **CHX** and **CHD** commands here. One uses these commands under OS-9 (as well as from within Basic09) to reset the system's directory pointers. Suppose you have a collection of programs in a separate directory called "programs". For OS-9 to be able to run them it must be able to find them. You can tell OS-9 where it can find the programs by typing (in this case): CHX /d0/programs. CHD works in the same way, but resets the pointer to the working directory. For instance: to access a directory called "textdata" on drive /d1 you would type **CHD /d1/textdata.** Once you have set the directory pointer, you can access all files within that specific directory by just typing their name rather than having to type an entire pathlist to (one of) those files.

In a way one could say that CHX/CHD are the OS-9 equivalents of DECB's **DRIVE** command. They are just a little more complex to use because OS-9's directory system is more complicated.

With regard to the commands mentioned above I want to point out two more things. OS-9 and Basic09 do not support the **LOADM** and **SAVEM** commands. The reason for this is that OS-9 can load programs from any language as long as the programs' module header is recognized by OS-9. This module header contains, among other things, a code for the language in which the program is written. This code is checked by OS-9 before it tries to run the program, but not when it loads a program. So a simple **LOAD** command will do for all applications.

The second point is that Basic09 does recognize the **KILL** command but not for deleting files. **KILL** is used by Basic09 to unlink modules that it no longer needs from it's workspace. This is very useful when you write programs too large to fit into the CoCo's 64K address space. If you want to delete a file from within Basic09 you must use the **DELETE** command.

The following commands have been dealt with in previous articles so I won't do a repeat: **CLOSE, EOF, GET, INPUT, OPEN, PRINT, PUT, RUN** and **WRITE.**

CVN and **MKN$** are not supported. OS-9 uses a technique often called streaming to access disks. In simple terms this means data transfer without format checking, so there is no real need for conversions.

On the point of conversions I would like to point out that if you write a program that needs to share it's (numerical) datafiles with programs written in languages other than Basic09 you may want to store numbers as strings. The reason for this is simple: If you store a real number on disk it has the same 5-byte format as in memory. Most other languages use either 4-byte and/or 8-byte formats to represent real numbers. So, even if you get a program to read the correct number of bytes it may not have a clue about their meaning. If you store a number in string format it looks the same as you would enter it from the keyboard; a format that is universally recognized.

DSKINI is not recognized by OS-9/Basic09. Use **FORMAT** instead.

DSKI$** and **DSKO$ are not supported by OS-9 either. OS-9 does have a low level disk access command but it works differently. You can get low level access to a disk under OS-9 by adding "@" to the drive name when you open a path to it. E.g. **OPEN #path, "/d0@":READ.**

The main difference with the DECB

commands is that you still can not access a disk by referencing track/sector numbers. Instead you must specify what's called a Logical Sector Number or LSN. OS-9 converts this number into track/sector numbers using data contained in the device driver.

Using this command, the entire disk is regarded as a single file. On a DS 40 track disk this file is 1440 sectors long (numbers: 0-1439). This approach makes for easy programming when you want to write a backup utility and it also works fine for [illegible] DECB format disks. A drawback is that you can not access, for instance, MS-DOS format disks this way on a standard setup. The reason for this is that addresses get incorrectly decoded by the device driver. To correct the problem you will have to install an extra driver capable of handling the 512 byte sector format used by MSDOS.

All **FIELD** statements in your program can be replaced by **TYPE** statements. This essentially defines your record as a complex data structure which can be transferred to and from files with the same **GET/PUT** commands as used in DECB. Example:

FIELD #1, 5 AS A1$, 10 AS A2$, 7 AS B$

would translate into:

TYPE record=A1$:STRING[5]; A2$: STRING[10]; B$:STRING[7] DIM buffer:record

Note that the #1 (the buffer number) is not specified since Basic09 returns a path number when you **OPEN** a path to the file. "Buffer" as defined in the **DIM** statement is simply a label to identify the memory space set aside for "record". The buffer that is alluded to by #1 in the **FIELD** statement does exist under OS-9 but is handled internally by OS-9 and completely transparent to the user.

The same is true for the **FILES** statement. Every time you **OPEN** a path to a file OS-9 creates the necessary buffer(s) but this process is transparent to the user.

LOC and **LOF** are not supported by Basic09. This leaves basically two ways to deal with the problem of where you are. Your application can take care of it either by using an internal counter or by defining a byte (or two) as part of the record to hold that record's number.

Another way of dealing with the problem is using OS-9's filepointer. For every open file of every program running OS-9 maintains a filepointer. This pointer always points to the next byte to be read from or written to in that file. Upon **OPEN**ing or **CREAT(E)**ing a file the pointer is automatically set to 0. This means that your first access to a file always starts at the first byte in the file.

If you want to access a different part of the file you must use the command **SEEK #path, bytenumber** with bytenumber being the exact location of the start of the next disk access.

This doesn't help much, however, if you want to know where you are. Basic09 doesn't have any direct commands built-in to tell you, but we can use a system call to read the current value of the filepointer.

Assuming you have a data structure set up to mimic the 6809's registers the following code will do:

regs.a=path (path associated with file)
regs.b=5 \ RUN syscall($8D,regs)
filepointer=65536*regs.x+regs.u

The number of the record last read or written to is calculated as:

record#=filepointer/SIZE(buffer)-1

Note that this line only works if the above mentioned **TYPE** and **DIM** statements are also included in the program.

LSET and **RSET** are not recognized by Basic09. Strings are always left justified in the space allocated to it. If you don't define a variable as a string of length x, but use a variable name ending with $; Basic09 sets the length to 32 characters. If a string is longer than it's allocated space Basic09 truncates it to make it fit and discards the rest. If a string is shorter than it's allocated space Basic09 terminates it with a CHR$(255).

If you have to justify a string for output formatting use the **PRINT USING** statement with the following codes:

Sxx> right justifies a string in a field xx chars wide

Sxx< left justifies the same setup

Sxx^ will center the text in the field

The **UNLOAD** command is not recognized by Basic09 either. The best way to prevent problems with paths to files being left open, which can happen, is to exit your program with a **BYE** statement instead of **END**. As I mentioned earlier in this series, **BYE** forces OS-9 to execute a F$exit system call. Part of the function of this system call is to close the paths left open by a program, so it implies an **UNLOAD** command. **BYE** also exits Basic09 so the best way to implement it, is AFTER you have debugged your code.

Before you pack your code replace all **END** statements in the PRIMARY module with **BYE**. In this context "primary" refers to the module that runs all the other modules that are part of a program.

If you have problems with a program not closing paths it usually shows up as follows: After you have run and exited a program you want to delete a file the program accessed. After typing the command "del filename", OS-9 doesn't delete the file but prints an ERROR #253.

VERIFY ON/OFF has no direct counterpart in OS-9. Although it is possible to turn write verification ON/OFF under OS-9 you will have to get a utility from a third party source to do it. Rainbow (2/90) contained an article by Stephen Goldberg on the subject, which includes C source code for a similar program.

There is a more radical way of dealing with the problem: altering the drive's device descriptor and saving the changes in a new bootfile. This will boot your system with write verification disabled which results in speedier programs if they have to write to the disk fairly often. This has no impact on the speed with which data is read from a disk.

Most hardware seems to be reliable enough that this is a safe way of speeding up programs without creating unpleasant surprises when it's time to read the data back from the disk. If you want to go this route change the byte at offset $1A (#26) in the floppy drive descriptors from 0 to 1. If you know how, fine, but it is somewhat beyond the scope of this article to explain the process in detail.

If your program was written under ADOS, JDOS or similar implementations, you may run into commands like RATE, which OS-9 handles in the same way as VERIFY (reading a code byte from the device descriptor) and the FREE command whose duties are taken over by a utility with the same name. However since I'm not familiar with these ROM's I won't deal with them in this article.

The one last thing I want to mention is the **BAUD** command which can be replaced by OS-9's TMODE (or XMODE) utility. There is one way to alter device parameters from within Basic09 that I haven't mentioned yet. However, that will be part of another article.

< 268'm>

Please refer comments on this series to this magazine or directly to the author:

Chris Dekker
RR #4
Centreville, NB E0J 1H0
CANADA

Don't forget to look over the C. Dekker ad on page 28 for the Christmas special!

MM/1 Update

David Graham

BlackHawk and the MM/1 in Chicago!

Well, here we are, back from Atlanta, and boy - is it nice to be home! Not that Atlanta was a bad place to be, far from it! But the trip was long and tiring, and I again saw a lot of people!

I started by driving down to Dave Wordell's place near Fort Worth. No fires this time, but we did run into some road construction work that slowed us down quite a bit. I spent the night after a very interesting evening at Dave's house. We discussed many things, and Dave showed off his System IV for me. We called it a night late, planning to get an early start the next day.

Early Friday morning, we headed out to Lee Veal's home, on the other side of Ft. Worth, picking up another fest-goer (Charlie something), on the way. Jim Noah, retired fire Chief, took us down over to Atlanta in his mobile home - you guessed it - Noah's ARK! So the ride was great, very comfortable all the way there and back again.

The fest itself was great fun, I had a boot right next to Sub-Etha Software, and Al Huffman and I had fun with the sound capabilities of the MM/1. Though for some reason, when I played a WAV file that ooowwwww, he kept saying 'Stop it!' in reply!

Sales were - adequate. Not surprising , as I really had no new software. I'd planned to release the long awaited DeskTamer2.0, but Bill Wittman showed up with a newer version somehow billed as 2.0. This rather unpleasant situation did not keep me from having an otherwise great weekend.

Many folks got a look at the new 306 board from Kreider Electronics, on display both at my booth and Wittman Computer Products. This board will be the heart of the new MM/1B and Bill Wittman's WCP306. Unfortunately, the boards did not reach Kevin Pease in time for him to have working boards at the show, though they are impressive! In a baby AT form factor, Kevin has managed to place an IDE hard drive controller, a driver for 2 1.44 meg floppies, AT keyboard driver, serial ports with onboard buffers for 115K baud support, a parallel port, and much more. That doesn't even begin to count all the goodies you can pack into the 6 AT class ISA buss slots!

A great happening on the KWindows front was the outcome of my meeting with Kevin Darling. A new team has taken over on maintenance and upkeep on KWindows, with full authority from Kevin to act on problems in his absence. This will allow for much more rapid fixing of bugs, and introduction of features that will help applications programmers do their thing. Already we've seen the addition of key sensing getstat calls that should result in the release of the long awaited MM/1 version of KB-Com as well as support for CTRL-ALT-DEL reboot support. This version is now in the hands of developers, so look for it to be released soon after our beta test results meet standards. Look for new bitmap standards and other programming help in the very near future.

Another mile stone was the formation of a new marketing association for OS-9 vendors working on products for those using OS-9 as a PC or for a host developement system. We use the term "Desktop OS-9" to describe this market, and plan to work with the OS-9 Users Group to improve the OS-9 business environment. I'm working now with Colin McKay, Frank Swygert, and Tim Johns to write the bylaws for the new organization and get in incorporated.

Perhaps THE most exciting point of the fest, for me, was the arrival of the first of the new 8 meg backplanes for the MM/1. This marks an upward point in the process towards a complete rerelease of the MM/1. As I write, we are assembling the first new MM/1 I/O boards. Working with a new assembly company is nice, though we are experienceing those difficulties you might expect with a project that is new territory for both of us. Still, I expect to be shipping new MM/1's by the end of October or mid-November.

The month since the fest has not been without it's events either. We've received more orders for the 8 meg backplane, and have shipped the first 5 ordered. We'll ship more within the next week. Also, I've talked to Hazelwood, and BHE will begin selling Hazelwoods 68020 and 68030 systems effective immediately. The EK-20 and EK-30 boards are widely known as the main component of FHL's KiX/20 and KiX/30 machines, and are available now with support from BlackHawk Enterprises, Inc. These boards will be available at prices competetive with Frank Hogg Laboratory, Inc.'s - but take a look at our prices on peripherals and you will see that you can save quite a bit. I'm excited about this, as BlackHawk Enterprises, Inc. is now the only vendor in the personal OS-9 market to offer a full range of machines from the 68000 to the 68030. One more step on the way to becoming your one-call-does-it all shop for OS-9 excellence!

On a personal note, the last month has been marked by some distressing news for me. It seems I'll be needing to have surgery on my gallbladder soon. I'll be making every effort to keep this from slowing down the ongoing work on MM/1 production, as well as increased marketing work that involved working a contact in Germany to open up sales in that country. The opening of this market and the Australian market is creating some exciting opportunities in the OS-9 world. Until next issue, good bye.

< 268'm>

BlackHawk
Enterprises, Inc.
Supporting
Desktop OS-9
now and in the future!

Micro News

The biggest news this time is the introduction of a **new entry-level OS-9/68000 computer** designed specifically with the hobbyist in mind. This is Carl Kreider and Kevin Pease's **AT306**. It uses a 68306 integrated processor and a PC/AT 16 bit expansion bus. The board is a compact 9" square (approx.) and designed to fit in any standard PC case, even the smallest! See "Nine Flavors of OS-9" for more details.

Do you use a 6309 processor? Do you wish there was more software that would take advantage of the extra speed and capabilities of the 6309? Look no further than Chris Dekker's ad in this issue. He has updated several of his popular OS-9 Level II packages to use some of the 6309 functions and to add a little speed to the programs.

ISO 9001 Certification

Microware has achieved international certification that it meets ISO 9001 standards from Underwriters Labrotories.

ISO 9001 sets the quality standards for design, development, production, final inspection, testing, installation, and service. This certification will become more important in the future as more companies initiate TQM (Total Quality Management). To maintain TQM status, companies are encouraged to use products which meet ISO 9001 standards for quality and reliability. Microware states that existing customers should notice no difference, as Microware has always practiced a total quality philosophy. This certification makes Microware the first systems software company to be certified. *(from Microware's SIGNALS, Fall 1994)*

OS-9 for PowerPC

"The PowerPC hardware design strategy closely matches Microware's by providing 'plug-n-play' solutions for the complete spectrum of high-performance industrial systems to small embedded electronic products. OS-9 and PowerPC are especially suited for low-cost consumer electronics, such as interactive television set-top decoders, where real-time functionality are essential."

Ken Kaplan, President, Microware Systems Corp.

OS-9 is now available for the entire range of PowerPC processors, including Motorola's MPC505 and IBM's 403GA embedded controler versions that will eventually appear in set-top boxes.

Development tools include FasTrak, an integrated C language cross development environment for UNIX and Windows that combines Microware's Ultra C optimizing compiler with automated edit, debug, build, and source code control tools. All development tools as well as the OS-9 operating system have been optimized to take full advantage of the PowerPC's RISC architecture. Users can take advantage of the PowerPCS's Visual Caching mechanism to lock critical routines in the processors' internal cache, thus provideing rapid real-time response.

PowerPC OS-9 is currently packaged as a Developers Pak License for use with Microware's resident development tools and with FasTrak for UNIX and Windows cross development. *(from Microware's SIGNALS, Fall 1994)*

Positions Available!

Microware has the following positions available. All require a minimum of a BS in Computer Science or related field, experience with OS-9 or other real-time operating systems, and experience with the C programming lnguage. Entry and Senior level positions are available.

Multimedia Engineer
UNIX Application Engineers
C++ Compiler Engineers
Operating System Engineers
Product Integration Engineers
Netwrok Engineers
Debugging Engineers
Customer Support Engineers
Technical Training Engineers

For more information, contact the Human Resources Administrator, Microware Systems Corp., 1900 NW 114th St., Des moines, IA 50325-7077; Phone 515-224-1929, Fax 515-224-1352 *(from Microware's SIGNALS, Fall 1994)*

FARNA Systems is currently working on a re-release of Paul Ward's "Start OS-9". The first version, expected to be released early in 1995, will be for OS-9/6809. A version tailored for OS-9/68000 will soon follow. Price will be $25 and include the disk.

FARNA was able to obtain rights to publish from Paul earlier this year. Content will be edited and information updated. **Anyone with a copy of the original "Start OS-9" can earn a $1 off coupon for the new version** simply by writing FARNA and specifying any additions, corrections, or problems they encountered with the original volume.

If you have new soft or hardware products, let us know! We will gladly print a free blurb for you here in MicroNews whether you advertise or not (though we will be happy to have your ad also).

Operating System Nine

continued from page 18

The second AIF illustrates the how to launch a packed Basic09 program. It always works, but the third version often does the job, taking advantage of a little smarts built into OS-9 (calls runb when asked to run a b09 module). The advantage is it displays the correct title under the icon.

The remaining lines are pretty self-explanitory, but have some quirks. First, though Tandy told users to put their icons in ICONS, they didn't tell programmers. The icon line has to extend the path from CMDS ('ICON/icon.name' rather than 'icon.name').

Last but not least, Allen Huffman reports a feature I didn't know about. On graphics type windows, it you specify a size smaller than the whole window, you get the size box used by calc, clock, etc. It's minimum size is the size you gave, and it can be expanded to the entire screen. Graphics are scaled to fit, but text scaling is limited to adjusting the starting cursor position, which means a cursor followed by a short text label will appear in the right place, but little else.

< 268'm >

Comments and questions may be sent in care of 68'micros or directly to Rick at:
Rick Ulland
449 South 90th
West Allis, WI 53214
E-mail is rickuland@delphi.com

CoCo-C

Complete K&R C compiler, editor, linker, and standard libraries for the CoCo 1, CoCo 2, and CoCo 3 (separate versions). All CoCo 3 memory and hi-res screens supported.

Send $25.00 ~~(plus $2.50 S&H)~~

Christmas Special.... FREE shipping!

for your copy today, and start learning to program in C on your CoCo <u>*without OS-9*</u> !

FARNA Systems
Box 321
Warner Robins, GA 31099

Christmas Special: All FARNA orders until 05 Dec 94 get free shipping!

Christmas Special: All FARNA orders until 05 Dec 94 get free shipping!

Christmas Special: All FARNA orders until 05 Dec 94 get free shipping!

NEW PRODUCTS *from* FARNA Systems!

<u>CoCo Family Recorder/OS-9 1.0</u>
If you are into genealogy, then the CoCo Family Recorder is the *absolute best* program for the CoCo 3. But it runs under DECB, and many OS-9 users simply don't want to leave the multi-tasking environment. Well, we now have a solution for you! CCFR has been ported to OS-9! The program is very easy to use and menu driven. The OS-9 version is nearly identical to the DECB version in appearance, but takes advantage of many OS-9 features such as pop-up windows for entering data. DECB users can send their *original* CCFR disk (it will be returned) to get the OS-9 version for only $20.00. Others must pay the regular price of $28.50. Requires at least one 40 track double sided drive (FD-502) or larger. Can be shipped on 3.5" 720K disk if requested.

FARNA Systems
Box 321
Warner Robins, GA 31099
$2.50 shipping and handling per order.
Canada S&H $4.00; Overseas $7.00

<u>OS-9 Point Of Sale 1.0</u>
If you have a small retail store business, this may be just what you've been waiting for! Designed specifically for the small business that needs more than one check-out station but can't afford the $7,500.00 or more for an MS-DOS based system. This easy to use, menu driven software uses OS-9's inherent multi-user/tasking features, eliminating the high cost of DOS based networks. Has all necessary features to replace your cash register and keep track of your sales and inventory automatically. Supports multiple serial ASCII terminals.Current price is only $62.50.

FARNA Systems

Box 321
Warner Robins, GA 31099
Phone 912-328-7859
Internet: dsrtfox@ delphi.com

Software, Books, and Hardware for all OS-9/OSK Systems!

<u>CoCo DECB Software:</u>

CoCo Family Recorder - $17.50
Genealogy program for CoCo 3. Requires 2 drives, 80 col. monitor.
NEW! <u>OS-9 Version</u> - $32.50

DigiTech Pro - $12.50
Sound recorder for CoCo3. Record any sound for easy play-back in your BASIC or M/L programs.

ADOS: Support for double sided drives, 40/80 tracks, faster formatting, much more!
Original (CoCo 1/2) - $15.00
ADOS 3 (CoCo 3) - $25.00
Extended ADOS 3 - $30.00 (ADOS 3 req., RAM drives, support for 512K-2MB)
ADOS 3/Ext. Combo - $50.00

Mind Games - $7.50
Collection of 9 classic games. Run from included RAM disk w/512K.

Cross Road II - $7.50
Simple Tic-Tac-Toe, but with amazing sound and graphics! Sound recorded with Digi-Tech.

Space Intruders - $14.50
Looks just like Atari's classic "Sapce Invaders"! CoCo 1/2 and 3.
Donut Dilemma - $14.50
Climb, jump, and ride elevators to top of Donut factory to shut it down! 10 levels. CC 1,2,3.
Rupert Rythym - $14.50
Collect Rupert's stolen notes, then work our correct sequence. Great action adventure!Get **Space Intruders, Donut Dilemma, and Rupert Rythym for only $33.50! Save $10!**

<u>CoCo OS-9 Software:</u>

Patch OS-9 - $7.50
Automated program installs most popular/ needed patches for OS-9 Level II. 512K and two 40T/DS (or larger) drives required. (*128K /35T users can install manually- state 35T.*)

OS-9 Point of Sale - $62.50
Maintain inventory, print invoices, customer catalog, etc. Multi-user capable under Level II. Supports ASCII terminals. Basic09 required. Simple menu driven interface.

<u>Books:</u>

Tandy's Little Wonder - $22.50
140 page softbound book with history and technical info for all CoCo models. Schematics, peripherals, upgrades, modifications, repairs, much more- all described in detail! Vendors, clubs, BBSs also listed.

OS-9 Quick Reference Guides Level II (Revision 2) - $7.50
68K (based on 2.3) - $10.50
Get that bulky manual offyour desk! These handy QRGs have all the most needed information in a 5.5"x 8.5" desk-top size. Includes command syntax, error codes, special keys functions, etc.

<u>CoCo Hardware:</u>

DigiScan Video Digitizer - $150
Capture images from VCR, camcorder, or TV camera. No MPI required- uses joystick ports. CoCo Max3, Max 10, Color Max 3 compatible. Special order- allow 90 days for delivery. Send $75 deposit.

Ken-Ton SCSI Hard Drive System and Components

Complete, ready to run, "plug and play" 85MB system. Top quality drive, case, and ps. $500.00
No-Drive Kit: controller, OS-9 drivers, RGB-DOS in ROM, 2 pos. "Y" cable, and drive cable (specify type). Seagate N series drive with ROM rev. 104 or greater needed. ---- $225.00
(all systems sent w/ROM for 8 [2MB], 110 [30MB], or 256 [70MB] DECB drives; specify)

Controller w/cable (specify SCSI case or direct to drive) --------------------------- $140.00
OS-9 Drivers ------------------------ $25.00
RGB-DOS (for DECB access)--------$35.00
*$50 for both when purchased w/controller.
"Y" cable, $25 for two position, $35 for three.
Drive cables - specify direct to drive or SCSI case type connector ---------------- $20.00

Add $2.50 S&H per order. Canada/Mexico add $4.00; Overseas $7.00

FARNA Systems Publishing Services

Type Setting and Printing: We can prepare professional typeset manuals, books, booklets, catalogs, and sales flyers for you - we can print or you reproduce as needed from a master set! Very reasonable prices - inquire!

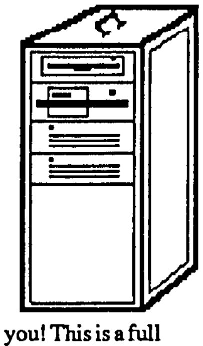
Mailing Service: If you send 200 or more catalogs, letters, disks, or newsletters at once we can do all work for you at pennies over the cost of your materials alone! How much is your time worth? Perfect for overseas and Canadian catalogs sent to the US!

Contact Frank Swygert at above address/phone for quotes

Christmas Special: All FARNA orders until 05 Dec 94 get free shipping!

for all your CoCo hardware needs, connect with

CoNect
449 South 90th Street
Milwaukee, WI 53214
E-mail: rickuland@delphi.com



NEW PRODUCT: The Cube
This tower enclosure was designed specifically for the CoCo and peripherals, even an MPI if desired! Four drive bays, two will hold a pair of 3.5" drives sideways. Easy access, carry handle mounted on top!

Mini RS-232 Port: Don't let the name fool you! This is a full featured serial port, supporting the signals needed for flow control as well as the basic 4. Jumper blocks allow readdressing or swapping DSR/DCD. No custom cables or hardware widgets needed here! Y cable users will need to add $9.95 for a power supply. **$49.95**

Hitachification: CoNect will install a Hitachi 63B09E CPU and a socket into your CoCo. Machine MUST be in working condition! The 68B09E will be returned unharmed. 90 day limited warranty. Chip and installation only **$29.95**

REPAIRS: We can repair most damaged CoCos, even those with bad traces where a 68B09 was removed. Costs vary with damage. Bad 68B09 sockets repaired for only $40! **Inquire BEFORE sending your computer.**

DISTO Products

Quality hardware for your Color Computer!

2MB Upgrade (no RAM) - $99.95
Mini Disk Controller - $70
Super Controller I - $100
Super Controller II - $130
MPROM Burner - $50
3-N-1 (parallel, RS-232, RTC) - $75
SASI/SCSI Hard Disk Adapter - $75
Full Turn of the Screw - book with all Tony DiStefano's Rainbow articles, January 1983 to July 1989 - $20
Complete Schematic Set - all DISTO product schematics except 2MB upgrade - $20

Include $2 S&H for book or schematics. Hardware S&H is $4.50 for one item, $6.50 two or more. Certified check or International Money Order only!
Running low on some items- please call for availability!
514-747-4851

DISTO
1710 DePatie,
St. Laurent, QC H4L 4A8
CANADA

SUB-ETHA SOFTWARE

By arrangement with StrongWare, Sub-Etha Software is proud to bring you...

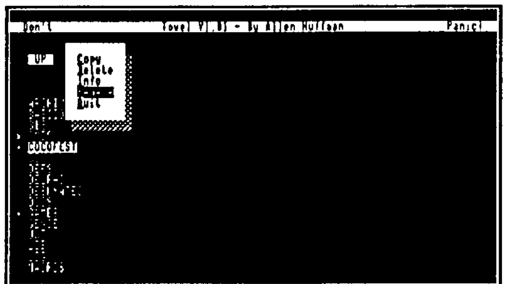
Soviet Bloc - Called the best Tetris(tm)-like game by many, this is a bigger version of the now-classic falling shapes puzzle game. RS-DOS Req: CoCo 3, Joystick/Orchestra-90 Pak optional ________ $19.95

GEMS - "Columns" of colors fall as you change the order of the colors. Match three in a row, column, or diagonal at the bottom and those colors disappear. Sounds simple, doesn't it? RS-DOS Req: CoCo 3, Joystick/ Orchestra-90 Pak optional ______ $24.95

Copy Cat - Simon says "match the sequence of tones as the colored diamonds flash". Great for building memory skills. RS-DOS Req: CoCo 3 ________ $ 9.95
OSK Req: MM/1 or 100% K-Windows Systems ________ $14.95

HFE (Hprint Font Editor) - A fantastic editor for those HPRINT fonts with lots of options. Create your own character set which you can LOADM and us in your own programs. Also creates and edits fonts compatible with MiniBanners! RS-DOS Req: CoCo 3, Joystick optional ________ $19.95

Font Collection - A collection of 18 useable HPRINT or MiniBanners fonts. (Importable to OS-9 for use with MiniBanners09 as well!) RS-DOS Req: ________ $ 9.95



Towel! V1.01 by Allen Huffman - NEW VERSION! A program no intergalactic hitchhiker should be without! Use mouse or keyboard hot-keys to perform common disk and file commands from pull-down menus. Tag multiple files for Copy, Delete, Rename, etc., and even point 'n click a Backup, Cobbler, Dcheck, or whatever. User definable menu for custom options. Runs under the EthaWin interface (included) on a high-speed text screen. All commands/colors configurable. OS9 Req: CoCo 3, OS-9 Level 2 ________ $19.95

Sub-Etha Software
P.O. Box 152442
Lufkin, TX 75915

Add $2.50 S&H. Texas residents add 8.25% tax. Write us for more info!

Sub-Etha Originals:

MultiBoot V1.03 by Terry Todd & Allen Huffman - Type "DOS" and bring up a list of up to sixteen OS9BOOT files! No more floppy swapping. A serious must-have for intense OS-9 users. OS9 Req: CoCo 3, OS-9 Level 2 ________ $19.95

1992 CoCoFest SIMULATOR V1.02 by Allen Huffman - NEW VERSION now uses compressed graphics and has scoring. Take a walking tour of the '92 Atlanta CoCoFest with this graphics adventure. 16-level digitized photos of the event and a text parser (ie, "get the box of disks") to let you interact. Runs on a 640x192 graphics screen. OS9 Req: 512K CoCo 3, OS-9 Level 2, 500K+ of Disk Space ________ $ 9.95
OSK Req: MM/1 or 100% K-Windows System ________ $14.95

Write-Right by Joel Mathew Hegberg - Featureful word processor for the MM/1 with what you would expect from a "real" word processor. What you see is what you get! ________ $54.95

Etha-GUI by Joel Mathew Hegberg - A neat program launcher for the MM/1 which includes handy desktop utilities like a phone dialer and nifty screen savers...even a trash can. Point and click icons to run programs. ________ $34.95

Quality OS-9 Software from

ColorSystems

NEW! K-Windows Chess for MM/1
Play chess on your MM/1..........................$24.95

NEW! X-10 Master Control for MM/1
Use MM/1 to control you home!................$29.95

Variations of Solitaire
Pyramid, Klondike,Spider,Poker and Canfield
MM/1...............$29.95   CoCo3..............$19.95

OS-9 Game Pack
Othello, Yahtzee, KnightsBridge, Minefield, and Battleship
MM/1..............$29.95   CoCo3.............$19.95

WPShell
An OS-9 Word Processing Point and Click Interface
CoCo3..$14.95

Using AWK with OS-9
Includes V2.1.14 of GNU AWK for OS-9/68000
MM/1..$14.95

To order send check or money order to:

Color Systems
P.O. Box 540
Castle Hayne, NC 28429
(916) 675-1706

Call or write for a free catalog! Demo disks also available.
NC Residents please add 6% sales tax
Owned and operated by ZacK C. Sessions

EDTASM6309 Version 2.02 ---------------------------- **$35.00**
This is a major patch to Tandy's Disk EDTASM to support Hitachi 6309 codes. Supports all CoCo models. CoCo 3 version uses 80 column screen, 2MHz. YOU MUST ALREADY OWN TANDY'S DISK EDTASM TO MAKE USE OF THIS PRODUCT. It WILL NOT work with a disk patched cartridge EDTASM.

CC3FAX -- **$35.00**
Extensive modification to WEFAX (Rainbow, 1985) for 512K CoCo 3. Uses hi-res graphics, holds full 15 min. weather fax image in memory. Large selection of printer drivers. Requires shortwave receiver and cassette cable (described in documentation)

HRSDOS -- **$25.00**
Move programs and data between DECB and OS-9 disks. Supports RGB-DOS for split DECB/OS-9 hard drives. No modifications to system modules (CC3Disk or HDisk) required.

DECB SmartWatch Drivers --------------------------- **$20.00**
Access your SmartWatch from DECB! New function added to access date/time from BASIC (DATE$). Only $15.00 with any other purchase!

RGBOOST -- **$15.00**
Make the most of your HD6309 under DECB! Uses new 6309 functions for a small gain in speed. Compatible with all programs tested to date! Only $10.00 with any other purchase!

Robert Gault
832 N. Renaud
Grosse Pointe Woods, MI 48236
313-881-0335
Add $4 shipping & handling per order

Christmas Special: All software ordered before January 1, 1995 comes with a money-back guarantee. If you are not satisfied with it, just send it back along with the manual for a complete refund.

CoCoTop 1.0	$24.95	CoCoTop 1.1	$19.95
CoCoTop 1.1 + Tools 3	$34.95	CoCoTop 2.0	$29.95
CoCoTop 2.1 + Tools 3	$39.95	CoCoTop 2.1	$24.95
OScopy/RScopy	$10.00	Tools 3 1.1	$29.95
Quickletter 2.0	$19.95	Homepac-09	$29.95
Accounting Level II	$34.95	Investing Level II	$24.95
Level II Graphics 1.2	$34.95		

NOTES:
* Upgrades from CoCoTop 1.x to 2.x are $10.00 (plus S&H)
* Upgrades for all other titles are only $5.00 (plus S&H).
* CoCoTop: order version x.1 if you already have Tools 3, x.0 if you don't. 2.x now includs a calendar, hot keys, and one key repeat.
* CoCoTop 2.x, Quickletter, and Accounting Level II detects and uses 6309 if present.
* OScopy/RScopy now supports multiple file transfers. Easily transfer files between DECB, MS-DOS, and OS-9!
* HomePac-09 keeps track of all personal finances (checking, savings, & CC accounts, addresses, phone numbers, programs, and even makes blank forms!

C. Dekker   *User-friendly Level II Programs!*
RR #4 Centreville, NB
E0J 1H0, CANADA
Phone 506-276-4841

Prices in US$. Call for Canadian $ prices. US/Canada add $3, all others $5 for S&H

Peripheral Technology Specials

486SLC/33MHz Motherboard w/CPU	$139.00
486SLC/50MHz IBM, ISA,CPU, 0K	$199.00
486SLC/66MHz IBM, VESA, CPU, Math	$299.00

IBM boards - Made in USA - 3YR warranty

1MB SIMM 70ns DRAM	$47.00
356MB Samsung IDE Drive	$229.00
420MB Connor IDE Drive	$275.00
546MB Maxtor IDE Drive	$379.00
IDE/Floppy/Serial/Parallel	$24.95
1.44MB TEAC Floppy	$49.95
Mini Tower, 200W, LED readout	$79.00
Panasonic Dual Speed CD ROM	$169.00
VGA Card ET4000-1MB, 1280x1024	$99.00
VGA Monitor WEN .28mm 1024x768	$249.00

UPS Ground $7.00 on most items except Tower & monitor ($12.00 UPS Ground).

1250 E. Piedmont Rd.   404/973-2156
Marietta, GA 30062   FAX: 404/973-2170

The OS-9 User's Group, Inc.

Working to support OS-9 Users

Membership includes the Users Group newsletter, MOTD, with regular columns from the President, News and Rumors, and "Straight from the Horse's Mouth", about the use of OS-9 in Industrial, Scientific and Educational institutions.

Annual Membership Dues:

United States and Canada	Other Countries
25.00 US	30.00 US

The OS-9 Users Group, Inc.
6158 W. 63d St. Suite 109
Chicago, IL 60638
USA

Northern Xposure *'Quality Products from North of the Border'*

OS-9 Level II Color Computer 3 Software

NitrOS-9 v1.20 Call or write for upgrade info or new purchase procedure. Requires Hitachi 6309 CPU	$29.95
Shanghai:OS-9 Introductory price Send manual or RomPak to prove ownership	$25.00
Thexder:OS-9 Send manual or RomPak to prove ownership	$29.95
Smash! Breakout-style arcade game	$29.95
Rusty Launch DECB/ECB programs from OS-9!	$20.00
Matt Thompsons SCSI System v2.2 'It flies!' 256/512 byte sectors, multipak support	$25.00

Disk Basic Software

Color Schematic Designer v3.0 New lower price	$30.00
Oblique Triad Software	*Write for catalogue*

Color Computer 3 Hardware

Hitachi 6309 CPU (normally 'C' model, may be 'B')	$15.00
SIMM Memory Upgrade Runs Cooler! 512k $44.95 0K	$39.95
Sound Digitizing cable	$15.00

OS-9/68000 Software

OSTerm 68K v2.2 External transfer protocol support TTY/ANSI/VT100/K-Windows/Binary Emulation	$50.00
Upgrade from TasCOM (Send TasCOM manual please)	$30.00

7 Greenboro Cres
Ottawa, ON K1T 1W6
CANADA
(613)736-0329

All prices in U.S. funds. Check or MO only. Prices include S&H

Internet mail: cmckay@northx.isis.org

Last Issue's "microdisk"

Diretcory of Volume 2 Number 2 "microdisk". Single issues are $6 each (US). Please specify volume and number when ordering. See inside front cover for subscription rates and other country prices.

Disk BASIC Side:
- DEMO.BAS
- DEMO1.BIN
- DEMO2.BIN
- ZDEMO.BAS
- ZENIX.BIN

OS-9/OSK Side:
- cgfx7.ar
- cgfx7doc.ar
- cgfx7index.ar
- Stream.ar
- playsndm.c
- make_ifile.c
- install_ifile.c
- mainloop.bas
- keydata.bas

The DECB side of the vol.2 no.2 disk contains some interesting demos, including Alan Dekok's infamous "bouncing ball" (demo2).

I apologize for the lack of DECB content in the last issue, vol. 2 no. 3 should make up for it with SDSK512K though.

If I am to continue to support DECB, I must receive articles and programs to print! Nothing is to simple, there is always someone who doesn't know something you may take for granted!

ADVERTISER'S INDEX:

BlackHawk Enterprises	*7*
C. Dekker	*28*
Color Systems	*28*
CoNect	*27*
Delmar Company	*BC*
DISTO	*27*
FARNA Systems	*25, 26*
L. Winterfeldt	*8*
Northern Xposure	*29*
OS-9 User's Group	*29*
Peripheral Technologies	*28*
Robert Gault	*28*
Small Grafx Etc.	*8*

The best way to keep your issue coming is to get more friends to subscribe! Let them see yours, then encourage them to get their own copy! If they mention your name when subscribing, I'll send you a $1.00 off coupon for any FARNA product!

For superior OS-9 performance, the

SYSTEM V

Provides a 68020 running at 25 MHz, up to 128 MBytes of 0 wait-state memory, SCSI and IDE interfaces, 4 serial and 2 parallel ports, 5 16-bit and 2 8-bit ISA slots and much more. The **SYSTEM V** builds on the design concepts proven in the **SYSTEM IV** providing maximum flexibility and inexpensive expandability. Optionally available at 33 MHz.

AN OS-9 FIRST - the MICROPROCESSOR is mounted on a daughter board which plugs onto the motherboard. This will permit inexpensive upgrades in the future when even greater performance is required.

G-WINDOWS benchmark performance index of a **SYSTEM V** running at 25 MHZ with a standard VGA board is 0.15 seconds faster than a 68030 running at 30 MHz with an ACRTC video board (85.90 seconds vs 86.05 seconds).

For less demanding requirements, the

SYSTEM IV

The perfect, low cost, high-quality and high performance OS-9 computer serving customers world-wide. Designed for and accepted by industry. Ideal low-cost work-station, development platform or just plain fun machine. Powerful, flexible and expandable inexpensively. Uses a 68000 microprocessor running at 16 MHz.

G-WINDOWS - a PROVEN WINNER

FOR OS-9

Available for the **SYSTEM IV** and **SYSTEM V** computers, the PT68K4 board from Peripheral Technology and the CD68X20 board from Computer Design Services. Resolutions from 640 x 480 x 256 to 1024 x 768 x 256. Support for multiple VGA cards running different processes, different portions of the same process or both.

G-WINDOWS is now AVAILABLE for OS-9000 from *delmar co*

Support for generic VGA boards and SUPER VGA boards including ET4000 (Tseng Labs), OTI067 (Oak), CT452 and CT453 (Chips and Technology), GENOA, WD90C11 (Paradise) and S3 (S3 Inc.) for modes from 640 x 480 to 1280 x 1024 depending on board.

Distributor of MICROWARE SYSTEMS CORPORATION Software

This ad was prepared and printed using QuickEd under OS-9.

delmar co

PO Box 78 - 5238 Summit Bridge Road - Middletown, DE 19709
302-378-2555 FAX 302-378-2556