

# the world of 68' micros

Supporting Tandy Color Computer Disk BASIC, CoCo OS-9, and OS-9/68000

## Special Notice: Another CoCoFest!!!


Right when people were beginning to wonder if CoCo support were really on its "last leg", news of ANOTHER CoCoFest comes! This is actually a one day "mini-fest" sponsored by Ricks' Computer Enterprise, publisher of CoCo Friends Disk Magazine. Support will be mainly for DECB, but I'm sure some OS-9 literate people will be there!

From 8am to 6pm, Saturday, July 15, the fest will be held (actual site to be determined yet). At 7pm everyone will move to Rick's house for a catered picnic. Tickets are \$15 each, but include the picnic dinner. Vendors can purchase tables for \$5 each (includes two chairs). As of this time, there is no admission without the picnic. I'll get more details later!

The fest will be held in Liberty, Kentucky. For additional info write: Rick's Picnic, Box 276, Liberty, KY 42539 (phone 606-787-5783; e-mail cfdmrick@delphi.com)

Due to a possible commitment with my printing business, I am unable to make plans to attend myself at this time. Baring this possible conflict, however, I DO intend to ride my motorcycle up there, pulling a trailer load of goodies!

**MONSTER  
HARD DRIVES**  
*for your CoCo (OS-9)*  
page 4

No more Flippies  
for OS-9!  
  
(page 14)

## CONTENTS

<i>The Editor Speaks</i>	2
<i>Letters to the Editor</i>	3
<i>Monster CoCo HDs</i>	4
(article) Robert Gault	
<i>Delmar System V</i>	5
(article) David Breeding	
<i>BASIC in Color</i>	10
(column) Fred Remin	
<i>Industrial OS-9 User...</i>	11
(G-Windows series) F.G. Swygert	
<i>The Hardware Hacker</i>	12
(column) Dr. Marty Goodman	
<i>Nintendo Controller for CoCo</i>	13
(article) Jason Reighard	
<i>Converting OS-9 "Flippies"</i>	14
(article) Robert Newhart	
<i>Using Syscalls in Basic09</i>	14
(article) James Jones	
<i>Operating System-Nine</i>	15
(column) Rick Ulland	
<i>OS-9/OSK Answers!</i>	17
(column) Joel Hegberg	
<i>Programming in "C"</i>	18
(column) P.J. Ponzo	
<i>Basic09 In Easy Steps</i>	20
(series) C.hris Dekker	
<i>MM/1 Update</i>	22
(column) David Graham	
<i>Micro News</i>	23
<i>Advertiser's Index</i>	27

**POSTMASTER:**

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

**the world of 68' micros**

Published by:  
FARNA Systems

P.O. Box 321  
Warner Robins, GA 31099-0321

Editor: F. G. Swygert

**Subscriptions:**

\$25/year (8 issues) US; \$32/year for Canada/Mexico (\$13 US, \$17 C/M for six months- four issues). Overseas \$45/year (\$23 for four issues) AIR; \$37/year, \$18 six months for surface 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 surface, \$2.50 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 e-mail at dsrtfox@Delphi.com. The Delphi CoCo and OS-9 SIGs on Delphi are also frequented (The Delphi SIGs are still sponsored by Falsoft).

## The editor speaks...

F.G. Swygert

Many of you apparently like the new look of the cover. Allen Huffman of Sub-Etha Software was a bit dismayed that I mentioned possibly moving the table of contents inside the magazine. He stated that with the contents on the cover, it is easier to find articles in back issues. Good point Allen... very good reason to *keep* it on the cover!

Which reminds me, I need to make an index of all issues to date! It is getting hard to keep up with what I've printed already! Someone had volunteered to write an index program, but I seem to have lost the name and sample he sent. Hopefully he'll read this and get back in touch with me, I can sure use that program now! (yes, it will be made available to readers on an issue of "microdisk", if it becomes a reality!)

Those who have expressed an interest in the CoCo emulators should read my reply in the letters column. You may find it quite interesting.

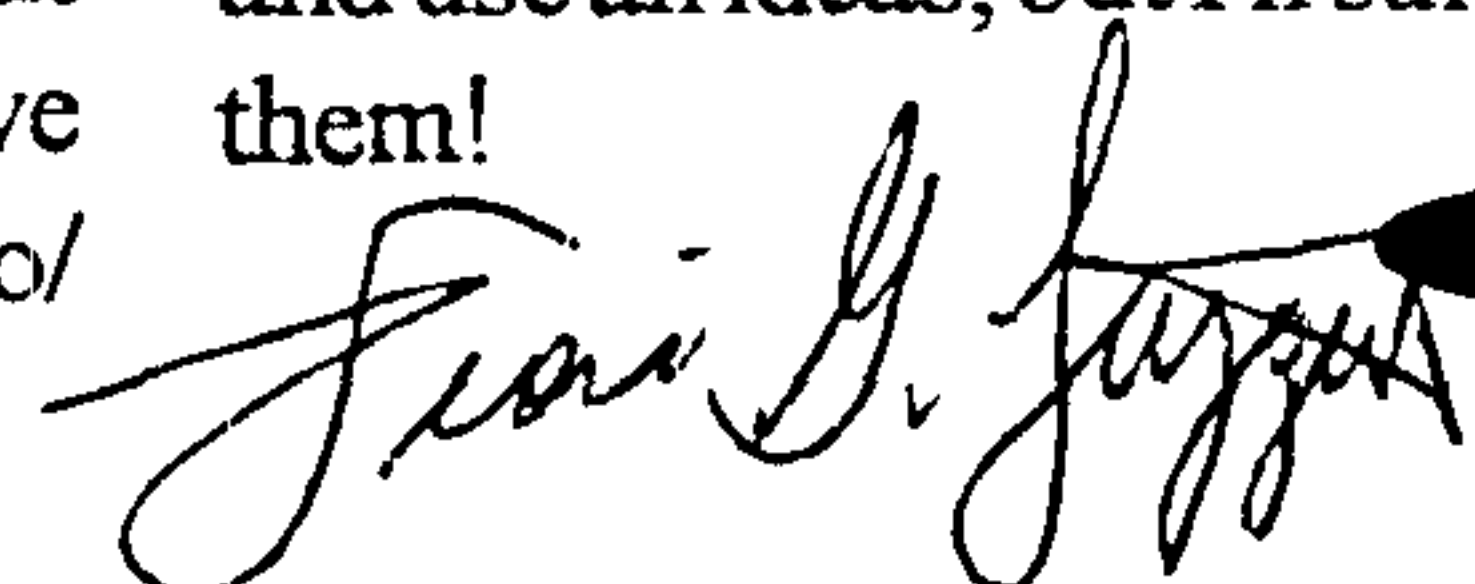
Another item of note: you'll notice that my signature at the end of this column looks MUCH better! The one I was using was scanned in and placed as a graphic. After looking at it in the last couple issues, I decided it looked to much like what it is -- a bit-map graphic image. The column should look better without the tacky image! Then it dawned on me that I never needed such an image at all! All I have to do is physically sign the master copy, not all 250 or so! Oh well, I'm not the first person to use a computer to make a simple task more complicated! And everyone once thought computers would make life so much simpler...

How many of you intend to be at the Chicago CoCoFest? I should have some good deals on 80MB CoCo/

OS-9 hard drive systems. But I'll only have two or three there! RGB-DOS can be used with these 512 byte sector drives, but would only support half capacity (40MB). Still if you only want a few DECB partitions, it won't waste much drive space. And even 40MB is a lot of space for OS-9 or DECB (40MB holds the equivalent of 146 35 track single sided floppies!). I may have one or two 100+MB drives at a fair price (under \$1 per MB). Even if you aren't buying, please do stop by the FARNA booth and say hello!

I'm also working on a CoCo, OS-9, and OSK desktop catalog. This will feature all vendors I can find in a single catalog. In order to get the best response, ad space will be cheap.. \$5 per page. The first catalog will have two pages FREE for each vendor. But this won't cover printing and shipping cost. The catalogs will carry a minimal price, most likely under \$5.00, to cover printing and shipping. Price really depends on finished size. The catalog will be printed on legal size paper, folded (each page 7"x8.5", image area 6"x7.5", 1/2" margins) and stapled. This maximizes image area for a minimal increase in paper cost. Do urge anyone still in the CoCo market to participate in this! It may be to late for the first catalog, which I hope to have ready for the fest, but another will be printed in late summer or early fall.

Well, I'll close with saying that if there is anything you can think of that would make this magazine look better (forget color! Costs to much!), read easier, etc., please feel free to let me know! Can't promise I'll like and use all ideas, but I'll sure consider them!

 < 268'm >

# Letters to the Editor

## CoCo Emulator: Great! (but could be dangerous!)

Enclosed is my check for a year's renewal. Although my interest is waning somewhat now that I finally made the plunge for a PC, I still want to protect all the software I wrote for my own uses in OS-9 over the years. Without the active support of Microware, it's a miracle that OS-9 is still around in any form. Your tip in December's mag about the CoCo emulators available from Jeff Vavasour is worth the entire year's subscription to me. Keep up the good work.

Joel Sherman  
343 Concord Drive  
Watertown, CT 06795

*Joel, you won't find anything in the PC world as versatile, powerful, yet simple as OS-9. Yet I understand why you want a PC. I use OS-9 and my CoCo for some of the business aspects for this magazine, but have to produce it on a PC. There is simply more good software for the prolific PC, and not that much being produced for OS-9, not even for the 68000 based OS-9 systems.*

*The emulator is indeed a god-send for many people. Now they can clear some desk space by only having a single computer up. Many hardly use the CoCo much after purchasing a new PC anyway. But I'll keep my CoCo up and running for a while longer all the same. Multi-tasking just isn't that great on the PC, and I usually have the PC and CoCo up at the same time anyway!*

*There is a hidden danger in the CoCo emulators though. People who write software for the CoCo do so because they USE the computer. People who tend to use an emulator do so just to make use of some old software, play a few old favorite games, and transfer some old data files. Note the word "old" in front of software, games, and data files. People who want the emulator most aren't concerned with keeping the CoCo alive, only the memory. For those who use only a CoCo and not other computers at home, the emulator does nothing. There may be a few who will use the emulator as a handy software development platform, but the average CoCo user won't benefit from the presence of an emulator as has been suggested.*

## WANTED: Clearer "C"

If there were an article I wish for, it would be a "walk through" of the CoCo OS-9 C compiler from Microware (you need THESE modules, they need to be in THESE

directories, your source code must have THAT extension, etc.). I've got the manuals, but nothing works like the manuals make me expect it to. Thanks for supporting the CoCo!

Craig Wheeler  
Box 136  
Mears, MI 49436

*Craig, I think I have someone working on just such an article. Henry Harwell mentioned something about getting ready to write some C articles, but if he doesn't cover setting the compiler up I'll get someone else to. Maybe Rick would like to tackle that in one of his columns. I'll check with these two and see what we can come up with!*

## RE: Vol 2 #3 page 8, "Basic in Color"

A phone not near you at work? Here's my solution: I bought from Radio Shack a 50 foot extension reel telephone cord with two jacks. I had Bell install another telephone jack near my computer. When needed I plug the extension cord in the new outlet and unreel it to where I need it. I plugged my modem in the extension reel also. Make sure the cord is not a tripping hazard.

I hope this solution helps someone else.

Mrs. L. Boulton  
330 Metsalfe #403  
Ottawa, ON K2P 1S4 - Canada

*That extension reel is a very handy unit! No cord to get tangled up when not needed, and two phone jacks! A very good idea!*

## ALERT! CoCo XT Problem:

Among the many outstanding contributions made by Chris Burke of Burke & Burke to the CoCo community in both software and hardware has been, notably, the "CoCo XT" (XT-RTC) adapter. This makes it possible to connect a PC/XT hard drive controller to a CoCo at far less than previous systems. Yet I endured more than a year of continual frustration and disappointment as I tried several XT controller cards, cables and connectors, and checking the OS-9 drivers, descriptors, and formatting process trying to get my system up and running. Considering that I did not order a CoCo XT-RTC until late September '93, I felt fortunate to have obtained one at all, especially after I began reading of other's difficulties in even contacting B&B about their orders.

I finally turned the problem over to Al Dages of the *Atlanta Computer Society*. While checking out my controller (with known good chips provided by fellow ACS member R.C. Smith), Al discovered a connector that had not been soldered before shipping. The tip-off that the material quality was not up to the usual "Burkean" standards should have been the aluminum housing, which is cut in a manner suggestive to a high school shop assignment. Definitely not the housing supplied with earlier B&B units, I was assured.

It may well be that others who bought one of the last adapters have had similar problems. Certainly any reader with soldering skills will want to check for bad or missing solder joints! Otherwise, I hope they can find someone like Al Dages to help out! Mega-thanks to Al, who seems truly to enjoy helping people, and to ACS!

J. David Baker  
260 Jockey Club Drive  
Athens, GA 30605-4008

*David, I'm glad you found your problem! It is most likely that Chris Burke was rushed getting those last few CoCo XT adapters out, and was hand soldering them himself. Things like missing/bad connections do happen occasionally, especially on short run hand assembled items. Add the rush into the equation, and it gets worse! But you are indeed fortunate to have received your unit in a relatively timely manner. Many never did! One person had to file in the post office for a "lost" money order, but I guess the \$5 fee was well worth getting most of the money returned.*

*I didn't like the way B&B exited from the market. Chris still puts out a message on Internet occasionally, the last stating that B&B was just on hold and not out of the CoCo market. I really wish he wouldn't keep people hoping like that... simply letting go and then coming back later as a surprise would probably be better! But I must defend him on the aluminum housing. Getting 50 or so made professionally may be cost effective, but getting them made by the dozen sure isn't! The last ones were probably made by hand at a sheet metal shop rather than stamped by a large outfit as the previous ones were.*

# Monster Hard Drives for the CoCo... Part 1

Robert Gault

## Installing a large hard drive under your CoCo/OS-9 system.

There have been a number of questions on the InterNet and in my local area about the maximum size hard drive that is usable with OS-9 Level II on a CoCo. There are two answers to the question which depend on whether you are using a stock system or have patched system modules. Let's start by looking at a stock system.

Logical sector number zero (LNS0) on an OS-9 disk contains information on the size, format, and other parameters of the disk. The number of sectors on a disk are indicated by the 3 byte value DD.TOT. The maximum value for a three byte number is \$FFFFFF or 16,777,215. Stock CoCo disks use 256 byte sectors so \$100 x \$FFFFFF = \$FFFFFF00 or 4,294,967,040 bytes. In perhaps more familiar terms, that is a 4.3 gigabyte hard drive.

There is one slight problem with this. OS-9 disks have an allocation map which keeps track of used and unused sectors on the disk. This map is allowed DD.MAP bytes of disk space where DD.MAP (two byte variable) can have a maximum value of \$FFFF or 65535. Each bit in the map represents one sector on the normal CoCo disk (DD.BIT=1). So, eight (bits/byte) times \$FFFF = \$7FFF8 or 524,280 as the maximum number of sectors that a standard Tandy OS-9 disk map can handle. This then is \$100 x \$7FFF8 = \$7FFF800 or 134,215,680; a 134 megabyte hard drive.

As you can see, the two results above (4.3G vs 134M) are different and we must accept the smaller value for an unmodified CoCo system. However, drive technology now makes available much larger drives at reasonable prices. In fact, it is getting darn hard to find drives under 340Megs. How can an OS-9 user on a CoCo make use of these larger drives?

The answer lies with the term DD.BIT which indicates the number of sectors represented by each bit in the allocation map. DD.BIT is a two byte variable which should be a power of two; ie. 1,2,4,8,16,32,64, etc. The largest power of two which can fit into two bytes is  $2^{15}=32768$  or \$8000. Now since the

maximum number of bytes in our map can be \$FFFF at \$8000 sectors per bit that gives:  $8 \times 32768 \times 65535 = 17$  gigabytes.

This is an enormous drive; larger than DD.TOT could handle. So if we can find a simple way to increase the value of DD.BIT we can use drives as large as permitted by DD.TOT.

How large must DD.BIT get? The answer is: DD.TOT/(DD.MAP\*8) or 64 as the smallest value of DD.BIT which allows the allocation map to track the maximum value of DD.TOT - a 4.3 gigabyte hard drive.

The above analysis is for a "stock" system that uses sectors of 256 bytes. But disk controllers are not constrained to use 256 byte sectors and 512 or 1024 bytes per sector are routine with other systems. Would anyone like a 17 gigabyte drive at 1024 bytes per sector?

To take advantage of larger sectors, you will need to change RBF, HDisk (or whatever your driver is called), format, and perhaps other modules. This could be more effort than it is worth. As you will see below, only format needs to be changed if you can be happy with 256 byte sectors.

So far so good, but how can you modify OS-9 to get the desired value of DD.BIT? The only OS-9 module we need to consider is format. This module creates our disk structure based on the information in the drive device descriptors. In an unmodified system, format has DD.BIT hard coded to the value one. We have two main alternatives, make format get DD.BIT from the descriptor or from an option in the format command syntax.

While it is desirable to have this information in the descriptors, there is no defined place to put it. At first glance, IT.SAS would seem to be where DD.BIT info should be stored, but OS-9 uses IT.SAS (minimum segment allocation size) in a slightly different manner from DD.BIT. A new byte would need to be added to the descriptor format and agreed to by the OS-9 community. Until this happens adding DD.BIT to the format command line seems best. After all, how

often do you format a hard drive? Strangely there is already built into format just such a routine. However, it has been deactivated because it is bad code which DOES NOT WORK.

The modpatch file below will patch the OS-9 Level II format command to accept the following syntax; () are required:

```
format /device (n) [additional options]
where (n) is the value of DD.BIT
```

The patch includes error checking which forces DD.BIT to be the largest power of two which can fit into the value of n. So if you enter an illegal value, you get the next smaller legal one. A value of zero will be made 1. The maximum value installed by this routine is 128.

What value should you use for DD.BIT? I recommend the smallest value that will map your disk. What are the trade offs as DD.BIT increases in size? With large drives, small values of DD.BIT will waste drive space in large allocation map and directory structures. Large DD.BIT values will waste drive space as unused sectors tacked to the end of files. A good example of this is how a one byte file uses one gran (9 sectors) on a standard DECB floppy.

If you understand the above discussion, you can pick a DD.BIT value for optimal disk usage. Perhaps Frank can hold a contest where the winner is the user of the largest hard drive on a CoCo.

*Editor:* Well, just who DOES use the largest hard drive on a CoCo? I also want to bring your attention to a product from Northern Exposure: Matt Thompson's SCSISYS drivers. This is a complete set of SCSI drive modules (including a new format command) to allow using common 512 byte sector SCSI drives with the CoCo. I send a shareware version with every Ken-Ton interface I sell unless the system is being used for DECB. The registered version supports buffering and has some other enhancements to make it even faster, I use this on my personal system. It is well worth looking into, and allows the maximum drive capacity to be 268 megs instead of 134.

Modpatch file for the stock format module; ed.22.1 CRC \$744268.

- \* modpatch file to permit format
- \* to alter DD.BIT
- \* (!)ink to format module; must be loaded
- \* in memory
- l format
- \* reinstate DD.BIT option
- c \$1d5 \$1c \$74
- \* replace incorrect option with
- \* good code
- c \$249 \$dc \$d6
- c \$24a \$1f \$20
- c \$24b \$4d \$26
- c \$24c \$27 \$02
- c \$24d \$02 \$c6
- c \$24e \$c6 \$01
- c \$24f \$01 \$4f
- c \$250 \$d7 \$58
- c \$251 \$27 \$4c
- c \$252 \$50 \$24
- c \$253 \$5a \$fc
- c \$254 \$d4 \$c6
- c \$255 \$27 \$00
- c \$256 \$27 \$56
- c \$257 \$04 \$4a
- c \$258 \$c6 \$26
- c \$259 \$01 \$fc
- \* verify module for optional save to
- \* the command directory

There are several other issues that should be addressed before I close this part of our discussion of large hard drives. Available to the CoCo community are replacement modules for RBF. Some can't handle values of DD.BIT larger than 1. This will be addressed in a following article.

Also available are PowerBoost and NitrOS9. These great software packages don't patch format, and if you use RBF ed. 28, don't otherwise affect the above discussion. There will be a problem with RBF ed.s 30 or greater though.

With the above change to format and using RBF ed.28, I have no problems of any kind with FREE, DCHECK, or any other disk operation.

free ed 6.1 crc \$E71F9D  
dcheck ed 4.1 crc \$3ED665

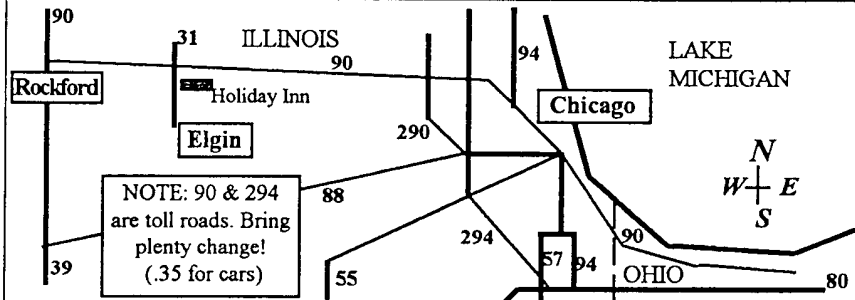
< 268 m >

Robert Gault  
832 N. Renaud  
Grosse Pointe Woods, MI 48236  
313-881-0335  
<ab282@detroit.freenet.org>

# 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



### 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

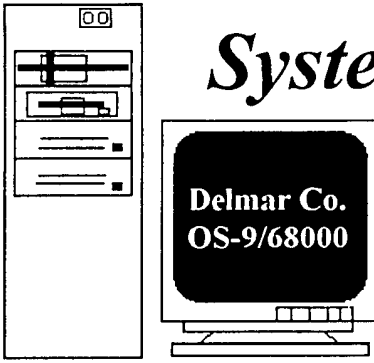
### Vendor Information:

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 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

# Delmar Co.'s System V



## Observations of an OSK Layman

*D. Breeding*

I would like to share a few of my observations after finally getting my feet wet in going to OSK. Of course others have written about this subject, but perhaps I can add another angle to the topic. It might be well to keep in mind that I am by no means any kind of an expert, and just another user.

I finally took the plunge back in May '94. I had been debating the subject for several years, actually since Tandy had discontinued the CoCo. To be honest, the CoCo was still serving me very well. I am not, however, a very good hardware hacker, and I knew that sooner or later there would come a time that it would become very difficult to maintain my CoCo, and I would be forced to go to something else. In that light, I found it ever more difficult to invest more in that machine. I have not totally abandoned the CoCo yet. I will probably keep it hooked up for quite a while. There are several files that I can use - or adapt rather easily - on the OSK machine.

In trying to make my decision as to the direction I wished to take in my upgrade path, I gave a tremendous amount of thought to all options. One serious question I had to deal with was - do I really want to invest in another OS-9 system? Let's face it, it's quite doubtful that you will ever be able to walk into Wal-Mart and see a shelf full of OS-9 software. There is not an overabundance of OS-9 software, but if you will look around, you can definitely find something to get your job done. It is a fact that at this time, we are not very well established in the CD-ROM field, and we don't yet

have scanner capability, to my knowledge, but I think these and other capabilities are just around the corner, and it's just a matter of getting it done. As for CD-ROM's, it is my understanding that they are working on this on the MM/1's, and this should drift over to the other systems soon.

In my decision-making process, I looked at the Macintosh; I really hated to go to MS-DOS, even with the wide variety of software and hardware available. I have to admit, I just have a prejudice against MS-DOS and Windows. After looking at everything, however, and finding out that the Mac's multitasking capability was not all that great, I finally decided to go with OSK.

Now, if I'm going with OSK, which system is best? From the time the MM/1 first came out, I've gone through a phase in which I thought each of the three systems is *\*THE\** system to have. At this time I would like to go through all my ideas of the pros and cons of each system. Keep in mind though, that I am not in a position to recommend any system, and in fact, I would not attempt to endorse one, because to be honest, I feel that any system one would go with would be a good choice.

Originally, when the MM/1 was first announced, I thought that this was it. Of course, I never did order, as IMS always seemed to be in trouble getting orders out. Then came the TC70, still, I never ordered and it soon was discontinued and replaced with the KiX series. I finally got around to the Peripheral Technologies/Delmar system, and, in fact, this was the route I took, finally ordering a System V from Delmar.

I finally took the plunge in May '94. At that time, Dave Graham was just in the process of resurrecting the MM/1. Again, I gave this machine serious thought. Some of the attractions of this machine was the fact that it appeared that the most programming activity was with this machine, although I now see more software coming out for my Delmar, especially in G-Windows. In fact, in a recent survey by "68' micros", to my amazement, an overwhelming majority of the respondents using OSK were Delmar customers. This was quite surprising to me, and it could be possible that this is not truly representative of the entire personal OSK user base. It could be possible that for some reason

more Delmar customers might have been inclined to subscribe to "68' micros", or they (we) might be more inclined to respond to the survey. I do seem to come across more people almost daily who are Delmar or PT customers, but just didn't seem to be very vocal.

Another plus for the MM/1 was the fact that one could use the old CoCo monitor, but on the minus side, for replacements, it seems to be getting more and more difficult to find compatible monitors, but they can still be had. Of course, with SVGA output, you have some really nice -- and \*BIG\* -- monitors if you are willing and able to get them. In my case, I opted for a 15" (although an "economy" version).

Although I had spent years in making up my mind, when I did get it made up, I wanted it NOW! At that time, there would have been maybe a two-month delay in getting one of the MM/1's, so that was one factor in my decision. Another con was that the main board on the MM/1 allowed a max of only three megs of memory. Of course now a board is pretty much available that will allow up to eight megs. One other thing was that it was not for sure that G-Windows would be supported on the MM/1, and it appeared that K-Windows might not get any further support either. In my analysis, (again not necessarily very reliable), I felt that G-Windows was the way to go. I have been rather well impressed by the MM/1, but I cannot really judge the performance of K-Windows, as I have not really seen it in action close-up.

Then there was the FHL Kix series. It implements a proprietary video card that will connect to any SVGA monitor. But as of last spring this video card was still under development, or just barely in full production, so I was a little timid about going this route. This is not meant as a detraction to FHL, but after the short life of the TC70 I was a little worried that this system might not last. Another thing that affected my decision was that the FHL machines were still being shipped with version 2.3 of OS-9. I don't know if this would be all that bad, but I did want to be a little more current.

In looking at the Delmar and

Peripheral Technologies line, the idea that it accepted standard PC cards looked rather appealing. In my final decision, this was the route I took. The System IV can be had for less, but it supports a maximum of four megs of memory, and the System V allows up to 128 megs. My system came with four megs, and so far I've never come close to running out of memory, although I do expect to need more some day, especially if I install Microware's UCC C compiler.

There is one little gotcha in regard to the PC cards that these computers use. It's not a matter of just picking up any card and stuffing it in and going with it. You are restricted to either using cards that have drivers for them, or you will have to write your own. If you get a card on your own, and don't write the driver software yourself, you will have to purchase the driver from Delmar if it is available. At this time, they support video cards and an internal modem. The modem requires a minor hardware modification. If you purchase the modem on your own and do it yourself, the warranty is voided, but if you purchase it from Delmar, they will do the modification and provide any warranty work in case of problems. So the PC card thing is not a total cure-all, but it would seem to me that at some time in the future, this might encourage more wide-spread adaptation of various cards.

There is one other system that is just now being put into circulation. Known as the "AT306", it might be a real winner. I am not familiar with it, but I'm sure the editor will update readers as soon as it is available.

Now for my general analysis of the system. I'm still quite impressed with it. The transition from OS-9/6809 to OSK is rather smooth. Most of the commands are identical, or nearly so. The hardest thing to get used to is the fact that most of the OSK parameters are preceded by a "-". For example, on the CoCo, you type "dir e"; on OSK, you type "dir -e". In going between the systems, I catch myself getting it backward, but don't worry the system will tell you. Anybody ever see a message that starts out "Error..."? But, in the final analysis, the commands are very

similar. One of my biggest problems is going back and forth between the keyboards (standard PC type). The backspace key, which I find myself using rather frequently, is located on the OSK system where the BREAK key is on the CoCo. I find myself hitting the BREAK key on the CoCo often when I want to backspace.

The speed of the system really spoils you. The most impressive thing you can do is do an "mdir". On a standard 6809 CoCo, it comes up rather slowly. There is a big speed increase if you use a 6309, but on the OSK system, in a text screen, as soon as you hit the ENTER key, the whole display just pops up in your face. Of course you see a big difference when you unzip files and the like.

My system uses an IDE hard drive. Disk access is extremely fast. A file seems to load and execute almost instantaneously. One thing that is a little anecdotal. The other night I did a "dcheck" on my System V. It has a 213 Meg HD. It takes less than a minute to do the process. For some reason, I went over to the CoCo and did a dcheck in \*IT's\* HD - an 80 meg with Disto controller. Gosh! It seemed like it took 15 Minutes or more. I didn't time it but it seemed like forever. Of course, I would think some of this is due to the speed of the HD interface.

I can't say that I have any real gripes, and I don't think I've ever regretted going with OSK. I guess the only thing I would like to see improve with my system is the graphic speed. The text in the graphic screens is a little slower than I would like, but it still does quite well. I'm not sure if it's in the processing overhead or the bus that makes it this way. This might be one area where the KiX might do better with its local bus video. Of course, from the ads, the max resolution for the KiX is 640 x 480, whereas mine can do 1024x768. I've never seen the display of a Kix, so can't compare them. In my judgement, my graphics screen scrolls at a rate something close to that of a CoCo in the 2-color graphics mode. Initially, I was not able to get anything close decent reception of data at a high port speed in the graphic windows. Indeed, I would lose characters at 9600 baud when

echoing characters to a 80x24 screen. However, in the text screen, I have no trouble with the computer set at 38.4 KBaud. One thing that might help here would be a more efficient terminal program. At this time, we don't have a very fancy terminal program for this particular system. I use STerm, using external file transfer programs. I have been looking into my serial driver and believe I have fixed the graphics screen performance. I found that the buffer that holds incoming data was quite small. After increasing the size of this buffer, I am now getting quite reliable data reception at 38.4 Baud in the graphics windows.

As a comparison for speeds, I ran a series of tests. I copied a file into /r0 on the System 5 (to eliminate the disk read variable). I then tried the various outputs. The file size was 124,172 byte in length. I listed it first to a G-Windows (graphic screen) of size 80 x 24. The command line I used was 'date>-time;list file;date >+time;list time''.

For comparison, I then did a similar test on the CoCo. The results are shown below. Times to read file in /r0, size 124,172 bytes

System 5 (G-Windows 80x24)(1) -	4:31 4:31	458 cps
System 5 (G-Windows 80x24)(2) -	3:09 3:10	653 cps
System 5 (Text Screen) -	16 sec. 16 sec.	7,760 cps
System 5 (>/nil) -	4 sec. 3 sec. 4 sec	31,043 cps
CoCo (Text Screen) -	2:23 2:24	862 cps
CoCo (Graphics 2-color) -	4:24	
CoCo (Graphics 4-color) -	11:26	181 cps
CoCo (>/nil) -	33 sec. 32 sec. 32 sec.	3,762 cps

Of course, when looking at the comparisons of the graphics screens, it should be kept in mind the tremendous amount of data that has to be transmitted. The G-Windows screen is in 256-color mode, and it takes about 2 bytes to make up a single pixel. The text for the G-Windows screens above were (1) was normal size letters, 6x13 pixels, and (2), small size letters, 5x10 pixels. Note that there was something like a 25%

increase in speed from the normal font when the small font was used, due to the decreased number of pixels per letter.

For those of you who might be needing to access the modem in the windows but are experiencing trouble, I have found one way around the scrolling problem. First, as I said above, the small letters can be displayed a little faster. I have discovered that I can display about 80 x 11 characters of text with no loss of characters, so I create a full 80 x 24 screen and then shrink it down to display 80 x 10. With a BBS, you really need to see when to input something. So I just keep the screen shrunk till I want to see the whole display. For most uses, as with the routine logon screens, and the like, you can just keep it small. The display will show the prompt and you can see what you're typing, so you can proceed rather well like this. This takes a bit of mouse clicking, but it does work.

You might want to use this method if you are accessing a BBS that has only one line. Here you might need to set an autodialer in motion and then go about something else till you get an answer.

The only other little nuisance I find with my system is that it does not support DMA for disk access, and if you are formatting a floppy or copying a large file to/from the floppy, you are, to a certain extent, back to the days of the CoCo with its masked-interrupt floppy system. However, in most cases, you don't have to fool with the floppies, and the Hard Drive is so fast, you don't notice with it.

Getting back to the software question, we do have a nice assortment of programs to use. We have many programs in common with the CoCo - STerm, lha, zip/unzip, rz/sz, ar, InfoXpress, ved/vprint. There are spreadsheets, both pd and commercial, databases, disassemblers.

For those involved with BBS's, there is a good QWK mailreader. For those not familiar with this, many BBS's now have the capability of encoding the messages (you can select which topics you wish to read) into a series of files and indexes, and usually compressing them using zip, lha, etc. After you go offline, read the messages, reply to any you wish, and enter new messages, you

can then go back online and upload these entries. It's really nice. There is also one available for the CoCo, it has a feature or two missing from the OSK version, and one or two missing that the OSK version has.

For fun, my system has a couple of gif viewers, a flic viewer (does animations), and there is a graphic display viewer for the G-Windows system. The gif viewers and the flic viewer only work from the text screen and don't allow viewing under G-Windows. However, there is a pair of programs (export\_gif and import\_gif) whereby you can translate back and forth between the two graphic formats. If you wish you can do a screen capture of your G-Windows screen and use export\_gif, and upload them for PC users to view. There is also another viewer for general vga pictures, again working only from the text screens. Also there are a few games coming out.. Stephen Carville has done several quite nice games.. one is a Blackjack game which can provide diversion for a tired user. He has also just released another game, I believe it is a space invaders game, I have not yet gotten it to see how it works.

As far as support is concerned, I have to say that in my case, Mr. Gresick has been wonderful. I'm sure he is quite busy, but he has always taken time to explain things to me and to help me with any problems I have had. In one case, I had trouble getting my system to do hi-speed zmodem transfers. He discovered that the driver for the comm ports had inadvertently had some of the critical portions commented out during some of their debugging work and they had overlooked reinserting them. In no time at all, he e-mailed me a corrected copy and all was well. I know I have asked some really silly questions, but he has always been quite patient with me. I must note that the Delmar systems were developed primarily for industrial, or business use, but Ed will take the time to work with the personal users. Of course, I'm confident that Frank Hogg or Dave Graham (MM/1) show an equal concern for their customers. I'm only guessing on that, but I have no doubt that they will do the same. I can say that in my case, I know that Mr.



Gresick displays a good general knowledge of his system and he can give very satisfactory answers to your questions rather quickly.

I don't intend for this to be an advertisement for OSK systems or anything of the like, but I do want to express my total satisfaction with what I have. If there is anyone who feels that the OS-9 system on their CoCo is adequate for their usage, but would like more speed or is concerned about keeping the CoCo going, then I would like to say that you should have no fear. These things work! I do feel that now, after experiencing OSK, it would be a little hard going back to the CoCo. After getting used to the screen display, the CoCo screen does not look as good as it did. As for memory, all you have to worry about is total system memory - well, not exactly true, you are now back to the Level 1 possibility of getting fragmented memory, but it's not happened to me yet. To backup a 1.44 meg floppy, just ask for about 150 meg of memory and grab the whole thing in one bite.

With a text editor, you will probably be able to hold the entire file in memory at once. One problem with the CoCo version of scred was that you were restricted to about 32K. If you were editing a larger file, you had to do one buffer full, then store this on disk and get the next section. On this edit, you couldn't do anything with a previous section that had been written. Ved would not accept a file over about 53K. You had to break a larger file down to smaller parts. I have yet to obtain ved for OSK, but I have been told that ved/OSK is much superior to the CoCo version.

InfoXpress for OSK also has several more useful features than the CoCo version. This is by no means a put-down for the CoCo or any of these programs. I have used the CoCoversion of them all and they are all quite fantastic. It's just that you are not as constrained in what you do when dealing with OSK.

As for getting started, you can, if you wish, start any of these systems as a terminal system... That is, for starters you can simply purchase the motherboard/operating system and if your budget is really tight, only a floppy

drive, although I would suspect that it would be somewhat uncomfortable trying to operate without a hard drive (although most of us did this for quite a while with the CoCo). At any rate, you can connect a terminal, or even your CoCo running a terminal program, to the system through one of the serial ports on the OSK system. Naturally the windowing systems would not be usable, but with this setup, you could begin to collect some of your software and begin learning the system. Of course, you would be better off if you can get the complete system from the start, but it is possible to ease into the system if you simply must.

## Operating System Nine

*(continued from page 16)*

### Launching Programs

Unlike MS-DOS, OS-9 has almost no built in commands and almost every line will need something loaded from somewhere. Common utilities can often be found already loaded (in mdir) but don't count on it. Assume every command and file will have to be loaded and the proc file will probably be moved some odd ball place-use complete pathlists to everything except perhaps /dd/cmds.

MS-DOS (and others) include a path command. Thanks to shell+, we do to. Path temporarily adds extra cmds directories to the exe search path- a very handy feature. For instance, a hard drive user may want to test out a floppy based program and still keep the main CMDS dir online. Just use:

```
Path /dd/cmds /d0/cmds
```

Easy enough from the command line, but from a proc file, the path statement can easily 'hang' the computer if an empty drive is included in the path and the file in question isn't found before it hits that drive. Always prompt the user before including floppy directories.

When the name of a program is given, OS9 normally 'forks' an extra shell to run it. This already happened once, when the original shell started another 'child' to run the batch file. When a program is 'launched' from a batch file by name only, a third shell joins the stack. This action sometimes causes changes the proc file made to 'disappear' as the shells are unstacked back to the original parent.

So, the command line option was born. Stock:

**progame**

the mess described above. Note these 'mortal' process can be killed with ESC (ctrl-brk) key.

**ex progame**

leaves off the new shell. When progame dies, it's really dead. More important, your batch file won't come back- do any ex-ing on the last line!

**progame i=/dev**

'immortal' process is kind of like the Energizer bunny- you can't kill it. Another end of the line command.

Shell+ adds:

**progame z=/dev**

Think about the mess described above. Now, kill the parent shell, and keep the child shell instead. In other words, the chd, path, and variables remain.

**progame ^#**

Where # is 0-255, and represents the priority for the program. This was (I think) broken in the original shell+ 2.1.

### Going further

Procedure files are very useful, but they can't do everything. Sometimes a packed Basic09 program is needed to handle the job. MultiVue reveals another problem. When a proc file is placed in cmds (or worse, an alternate location set by path), calling it from an AIF requires the magic word shell as 'program', with the proc file name as 'parameter'- all the icons just say 'shell'! The solution is to have a packed basic09 procedure do the job (or just call the proc file you wanted in the first place). Now the AIF can be rewritten with the B09 procedure name as program, parameters blank, and the icons display a meaningful name.

Other languages aren't as easily interfaced. One possibility is using the 10 shellsub variables to pass data back to the proc file, but we are getting a little bit beyond 'operating'.....

< 268'm >

*Comments and questions may be sent in care of 68'micros or directly to the author at:*

Rick Ulland  
449 South 90th  
West Allis, WI 53214  
E-mail is rickuland@delphi.com

Before you start feverishly typing in what you believe to be the correct code for your program, there are a few things that you should be aware of. Firstly, do not think that the computer is a super smart machine that can understand exactly what you would like to do. For example, it would not understand a command like "draw a box". To draw a box, you would have to use a number of commands in order for the computer to do the job.

By way of explanation, let's look at a simple action like closing a door. If you were to say to someone "close the door", they would simply go close the door. If that person were a computer, they would need commands like this:  
TURN 1/2 RIGHT (to face the door)  
MOVE FORWARD 3 FEET  
RAISE RIGHT ARM 4 FEET  
PLACE RIGHT HAND ON DOOR KNOB.

CLOSE RIGHT HAND OVER DOOR KNOB  
PUSH DOOR 2 FEET  
LOWER RIGHT ARM

I think you get the idea: you must break the action down into small steps.

This does not mean that the computer is totally dumb. The manufactureres have placed a number of simple commands into the memory area of the computer which will allow you to group a number of actions together to obtain a desired result.

With the above information in our memory banks, we must decide what we want the computer to do. This decision is as important for a game as it is for any other thing that we are trying to get the computer to do. By way of explanation, let's have a close look at a very simple program. We will then dissect the program into each decision that the computer will be required to make.

Let's say that we want the computer to ask for our names and then write that name on the screen. To achieve the

desired result we would break down the computer's actions into a logical sequence and then write the code to follow this "computer path of decision".

The following steps are required by our example:

1. Clear the screen
2. Ask for a name
3. Remember the name
4. Clear the screen again
5. Print the remembered name

This is a very simple decision process. You should be aware that in a more complicated program the break-down of the computer's actions could run into quite a few pages.

By looking through the "Getting Started with Color BASIC" manual, you will see that for the computer to clear the screen we use the CLS command. Therefore, the first line of our program would read:

```
10 CLS
```

This would clear the screen to the default background color-- green. At this time, read the information on the CLS command and what it can do for you. You can change screen colors for one thing.

The next thing we want the computer to do is ask for a name. Again looking through the manual, we see that we can use the command INPUT or LINE INPUT. So our next bit of code would be:

```
20 INPUT "what is your name"; A$
```

```
or
```

```
20 LINE INPUT "what is your name"; A$
```

At this stage you should read the information on each command and determine their differences.

By reading the manual, you should be able to determine that the previous line takes care of the next step also: remembering the name. Line 20 first asks for your name and then stores the name you type in as variable A\$. This information will remain in the computer

as variable A\$ until you either turn it off or change A\$ with another command. Your code should now look like this:

```
10 CLS
```

```
20 INPUT "what is your name"; A$
```

The next thing that we decided needs to be done is to clear the screen again. We already know the command to do this, so we repeat that here:

```
30 CLS
```

Now we are going to get the computer to print on the screen the remembered name:

```
40 PRINT A$
```

Now our complete program looks like this:

```
10 CLS
```

```
20 INPUT "what is your name"; A$
```

```
30 CLS
```

```
40 PRINT A$
```

You can see now that we first broke down into logical steps what we want the computer to do. Then we went to the manual to determine which commands were needed to carry out each step.

At this stage let me point out that this is not the only process that can be used to determine the code required for a program. Flow charts are often used. The "linear chart" that we used is, however, the simplest method. You should use it unless you are already familiar with flow charts or until you become more familiar with the CoCo's BASIC command set.

Well, I think I have given you enough to think about for this little segment. Now what you need to do is go through your BASIC manual and become familiar with all the commands. Try a few out! You can't hurt anything, and should become more familiar with how they work. I'll cover some more info on programming in the next issue.

# G The Industrial OS-9 User... by F. G. Swygart

# G-WINDOWS

Recently, someone on the OS-9 Usenet group asked what hardware they should pick to get the most speed of GWINDOWS. Mr. Alain Butzberger, an employee of GESPAC (an industrial controller/computer manufacturer) supplied the following information. He also offered some additional information about G-Windows that should prove helpful.

## G-Windows Hardware Speed Benchmark (in seconds)

	(OS-9) MVME-162 IP VGA	(OS-9000) 486DX50 VGA	(OS-9) 68030 GPD64400	(OS-9) 68030 ACRTC	(OS-9000) 386SX VGA VGA
Big Solid Boxes*	1.01	1.20	2.08	9.09	8.21
Small Solid Boxes*	1.62	1.86	2.94	4.16	11.26
Bit Block Transfer*	1.76	3.56	3.91	9.40	13.09
Raw Block Read*	0.56	2.27	0.84	3.33	16.97
Quick Font*	1.62	2.29	3.44	4.64	16.71
Normal Font*	1.64	1.63	3.66	5.08	11.68
Thin Outline Boxes	0.41	1.66	0.64	0.91	40.51
Thick Outline Boxes	0.85	2.23	0.79	5.03	18.39
Thin Lines	4.63	3.36	0.70	1.37	21.89
Thick Lines	1.32	3.61	1.38	7.62	29.84
Circles	3.13	5.13	0.81	0.33	31.43
Ellipses	2.30	2.98	0.44	25.60	18.32
Solid Rounded Boxes	1.52	1.75	1.59	9.49	12.53
Total* (seconds)	8.21	12.81	16.87	35.70	77.92
Total (seconds)	22.37	33.53	23.22	86.05	250.83

\*=denote the most used functions.

GWINDOWS supports LCD screens with no problem, a descriptor is used to drive any resolution and timings. GWINDOWS is extremely memory efficient. All parts are totally modular and reentrant. Minimum GWINDOWS configuration uses 195 Kbytes, average GVIEW gadgets: 14K, memory used for every open window: 7K, GVIEW window with 50 button gadgets: 21K. GWINDOWS is fast, efficient, multitasking and 100% ROMable. GWINDOWS run on OS-9 and OS-9000, Motorola or Intel platform. A self booting demo is available at no charge from GESPAC. It will show you the real thing on any PC with 4 Meg minimum RAM, 3.5" floppy as drive A: (editor: You must be able to boot from a 3.5" drive. Some setups allow configuring to boot from drive B:), and VGA. The demo will work even if you don't have a mouse.

Editor: I recently received the disk and info package from Gespac. The demo comes on three 3.5" high density disks. Unfortunately, I've been way to busy to try running the demo... hopefully I'll have time to run it on my 50MHz 486DX before long and give my impression of this interface.

Alain Butzberger  
Dept. of Support Engineering  
Gespac Inc.  
50 West Hoover Avenue  
Mesa, Arizona 85210 / U.S.A.  
The Grand-Canyon State

| Since 1979, answers to the needs of industry  
| through innovation, technology, quality and  
| services.  
| Phone: (602) 962-5559 Fax: (602) 545-5796  
| Technical:alain@gespac.com Phone: ext 208  
| Information: info@gespac.com 1 800-4-GESPAC

# 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

## The Hardware Hacker

Dr. Marty Goodman

Converting IBM joysticks to work on the CoCo and MM/1.

Someone recently asked me, "Do you have a CoCo anymore?" Heh Heh. Now that I just sold off one of my CoCo 3's, that leaves me with only FOUR CoCo 3's here, 2 Multipaks, 6 disk controllers, 3 RS232 paks, and numerous disk drives with case and power supply, numerous RGB monitors, numerous monochrome monitors. More important, perhaps... I still have one full CoCo 3 system set up, which I USE for development of some dedicated applications that I run on CoCo 2's.

I've got about 20 CoCo 1 and 2's lying around... when I need another dedicated controller, I pull one off the shelf and program it for the task.

### Now to the task at hand...

Tandy no longer sells the excellent Deluxe Joystick made for the CoCo and Tandy 1000 computers. These were very rugged units made by Kraft. Similar units are, however, still available for IBM compatible computers.

IBM joysticks are wired as variable resistors. The CoCo joystick is wired as potentiometers. That is, for the CoCo, one side of the pot is ground, and the other is 5 volts. The wiper, then, is hooked to the joystick circuitry voltage sensor for x or y axis. Thus, you have a potentiometer that is a voltage divider, producing a voltage of 0 to 5 volts. The pot is 100,000 ohms, give or take (50,000 is OK, but 1,000,000 is too big, and 10,000 is too small)

With the IBM, one is using just one side of the pot and the wiper. The other side of each pot is not used. Thus, the pot is not being used AS a pot, but rather as a variable resistor.

Do note that there are some IBM compatible joysticks that use digital switches or other schemes instead of potentiometers. These CANNOT be easily converted to work with a CoCo, Tandy 1000, or MM/1 (the MM/1 was designed to use CoCo joysticks). If the term "digital" is used on the box, don't buy it. Note also that the thin Nintendo/Sega style "control pads" are digital types and can't be easily converted.

Some things to remember:

(1) Make SURE that the IBM joystick potentiometers are 50K to 100K. 1 meg pots WILL NOT WORK. However, all

IBM joysticks I've seen ARE 100K or thereabouts. Apple joysticks sometimes, tho, are 500K to 1 meg.

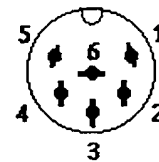
(2) The CoCo has one side of each pot hooked to ground. The other side of each pot is hooked to a source of +5 volts (a current limited source which has the +5 volts first sent thru a 100 ohm resistor inside the CoCo, to protect against a dead short in the joystick connector or cable). The wipers of the two pots go to the X axis and Y axis inputs on the joystick connector. The two buttons have one side ground. The other side of each button goes to the button pin on the joystick connector.

(3) The pin out for the joystick is as follows:

- 1 X axis signal (wiper)
- 2 Y axis signal (wiper)
- 3 ground
- 4, 6 joystick buttons
- 5 +5 volts (current limited to 50mA)

Pin 6 is in the middle of the connector, and pins around the circumference of the DIN connector are numbered in sequence clockwise from the upper right.

With this information, you should be able to convert an inexpensive IBM compatible joystick to work with your CoCo or MM/1. Do use a VOM to check the pots in the stick before removing any wires from a new unit! If they are not the correct values, you may be able to reassemble the case and return it.



CoCo Joystick Port

Numbered looking at the port  
from the back of the Computer

- 1 - Right/Left comparator input
- 2 - Up/Down comparator input
- 3 - Ground
- 4 - Fire button 1      *Fire buttons*
- 5 - +5V DC, 50mA      *are high open,*
- 6 - Fire button 2      *low closed*

# Nintendo Controllers on the Coco!

Jason Reighard

This all started when I saw someone selling Nintendo Contollers (for the 8 bit system) at a flea market for three dollars. I had had an idea to convert one for use on the Coco so I went ahead and bought one. I rembered that seeing ads for Atari type joystick converters for the Coco, so I thought if they did it for Atari type joysticks why not Nintendo controllers? Although this means making the controllers not work on your Nintendo, extra controllers are very inexpensive, usually about \$5-\$7 in video game stores.

The first step of course is opening the controller up. Be careful of those little screws they are misplaced very easily! Then use a razor blade or exacto knife to remove the chip by cutting each leg of the chip as close to the body as possible. Then get your trusty soldering iron and heat each leg and remove them by using pair of needle nose pliers while the solder is still hot. Then clean the holes that were ocupied by the chip by using de-soldering braid or solder sucker. Also desolder the wires from the board the lead to the Nintendo controller plug and clean the holes out as decribed eariler.

The next thing to do is to cut the 3 traces in figure 1, marked with an "X". Then drill a small hole in the spots indicated in figure 1, with the numbers one through four. I suggest a very small drill bit or even better yet as I did used a small jewelers screw driver and turned it by hand to make the holes. Keep in mind you can very easily break the board by using a drill. The screw driver method may take longer but will help prevent you from breaking the pc board. Next solder a wire to the circuit board trace through each hole and connect to the appropiate pin as indicated by the table in figure 1.

Looking at figure 2 connect pin 1 to the common, pin 5 to the right, pin 3 to the left, pin 5 to down, pin 3 to up, pin 6 to B, and pin 5 to A. Next connect the potimers as shown in Figure 3. It's a good idea to put those pots in a small project box near the end of the cable that plugs into your Coco.

Once that is all done the next thing is to calibrate your center. This is done by adjusting the pots. I have included a joystick calibration program (JOYTEST.BAS) that does this very easily. I suggest you set your X and Y to 31 or 32 since that is close to half of 63 you can get. Figure 4 shows the the correct values on the directional controller. JOYTEST.BAS will also test your fire buttons by indicating a red rectangle for the primary button (pin4) and a black rectangle for the secondary fire (pin6).

Once that's done your ready to go and happy gaming!

Questions and comments can be sent to me via Email at:

DELPHI: KB8SFC

Internet: kb8sfc@delphi.com

Fidonet: Jason Reighard@1:129/201

Or via U.S. Mail

Jason Reighard  
441 Ridgeland Drive  
Toronto, Ohio 43964-2022

Phone (614)-537-4875

Also Note: Kits are available from the author for \$15 and complete units for \$25 (Shipping included).

< 268'm >

- 1 to pin 2
- 2 to pin 2
- 3 to pin 1
- 4 to pin 3

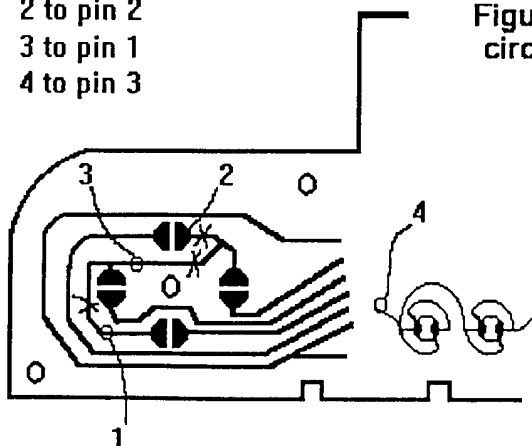


FIGURE 1  
Nintendo Circuit Board

## Parts List

- 1 6 pin din connector (HEI # DP6M)
- 2 100K potentiomers (HEI # TP19)
- 1 Small project box (HEI # JAL-1)
- 6 conductor wire

I obtained some of the parts from Hosfelt Electronics (1-800-524-6464), the part numbers are listed above.

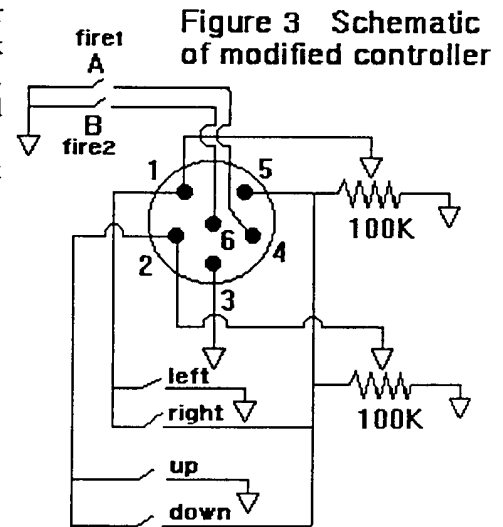


Figure 3 Schematic of modified controller

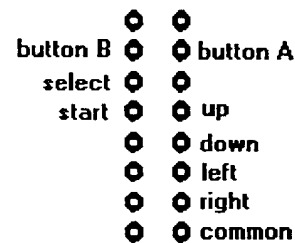


Figure 2 Pin-out of chip on circuit board (chip side)

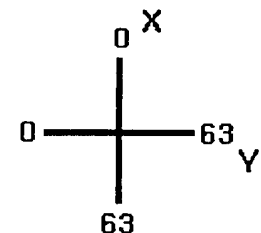


Figure 4 Correct X/Y Readings

## Converting an OS-9 ‘flippy disk’

Robert Newhart

Several months ago, I had obtained a copy of the game ‘‘Where in the World is Carmen Sandiego?’’, written for the CoCo. I enjoyed it very much, but didn’t particularly like the idea of flipping the diskette so often. I knew there must be a way of getting around this, but several other CoCo users told me it could not be done.

The techniques described in this article are also usable for any other diskette needing to be converted, or to convert a program on 2 single sided diskettes such as Desk Mate.

You will need an OS-9 diskette configured for double side, 40 tracks, and 2 disk drives capable of reading both sides of the diskette.

The first step is to remove the copy protection (if there is any) from the diskette to be converted. Where in the World is Carmen Sandiego is copy protected, but others I used these procedures on are not. Boot up OS-9, and put the diskette to be converted in drive 1. Then make drive 1 the working directory with the command `chd /d1`. By using the `dir e` command, you can see which files are protected. For a file to be copyable, the fifth character in the attributes column must be an `r`, not a hyphen. To remove the protection from

the protected files, use the command `attr <filename> pr`, putting the actual file name in place of `<filename>`. Be sure to do this for all the protected files in all directories on both sides of the diskette.

After the copy protection is removed, remove the diskette from drive 1, and insert a blank diskette. Format it, with the command `format /d1 2`, and if it is properly formatted you will be able to count all 80 tracks being formatted (10 lines of 8 each, in groups of 2 lines separated by a blank line). If not, then you need to re-do your OS9 boot disk then start over. See the OS9 manual for details on this. After the diskette has been properly formatted, then enter the command `cobbler /d1`, which will copy the 80 track double sided OS9 boot file from the boot diskette in drive /d0 to your diskette in drive /d1. The diskette in drive 1, which will be referred to now as a blank boot disk, is now ready to have the files copied from the flippy diskette. At this point, the directory for /d1 should only show OS9boot.

Now you are ready to begin. Type in the following 2 commands: `load makdir` and `load dsave`. Then remove your OS9 boot diskette from drive 0, and put the diskette you will be copying in drive 0.

Type in the command `dsave -s48 /d0 /d1 ! shell`. (no period) When all of the files have been copied, flip the diskette in drive 0 (or insert the other single sided diskette if converting two single sided diskettes to one), then again type in `dsave -s48 /d0 /d1 ! shell`. After you have finished, remove both diskettes from the disk drives, turn off the computer, put the newly copied diskette in drive 0, and type DOS. If you followed these instructions correctly, you now have a working copy of the game, with no need to flip the diskette. I have used this procedure with several diskettes, including of course ‘‘Where in the World is Carmen Sandiego?’’, and three programs consisting of two single sided diskettes each, ‘‘Home Publisher’’, ‘‘Desk Mate 3’’, and the Pascal compiler, with no problems running any of them, and with no need for changing or flipping diskettes.

Keep in mind that these diskettes are copyrighted, and that it is ILLEGAL to pass copies to your friends, although you may make a backup for your own personal use. Continue to support the manufacturers of CoCo software.

< 268/m >

---

## How to use System Calls from Basic09

James Jones

from Delphi....

89561 26-AUG 07:29 Programmers Den

RE: System Calls(?????) (Re: Msg 89511)

From: JEJONES To: CPERRAULT

Chris Perrault: Can someone point me to some reference material related to System calls and how to use them from Basic09(syscall)?

James Jones: Well...here’s as good a short description as I can manage: If you look in the OS-9 Technical Manual, you’ll see for each system call a description of what the system call expects to see in the CPU’s registers coming in, and a description of what the system call puts in the registers before you get control back. The BASIC09 syscall procedure takes as its parameters the following:

1. an integer number that corresponds to the system call you want to run
2. a structure that contains an image of the CPU registers

What syscall does is create code that does the system call you want, copy the contents of the structure into the CPU registers, executes the system call, copies the contents of the registers after the system call completes back into the structure, and returns.

So...if you want to use the syscall procedure, you have to look up the system call, find the number that corresponds to it, set up the input parameters, run syscall, and then retrieve the output parameters from the structure you passed syscall to get the results. (You’ll want to check the condition code flag image to tell whether it succeeded or not.)

The basic concept behind a procedure file is simple. Write a list of commands, save them in a file someplace, and anytime you need that list of commands performed, type the name of the file. Everybody has them, although 3/4 the world refers to them as 'batch files'. You may also see 'shellscript' - the basic idea is still the same. There are variations, since the command interpreter influences how the file will act, and Shell isn't that similar to command.com! In this article, 'proc file' means OS-9, while 'batch file' is reserved for MSDOS .BAT files.

The old simple list of commands format has been expanded, and modern systems include many features of higher level languages, like variables and logic structures. As usual, the only way to modernize a CoCo is to patch something, and here you'll need 'shell+' v2.1 or newer.

### What are they used for?

Procedure files are the smallest, fastest 'program' you can write. They require little support beyond the DOS itself and are user transparent. Perfect for small jobs like taming an unwieldy command line... to back up my hard drive to /d1, one types 'bak' or clicks the pretty bak icon, instead of the always confusing 'cd /h0; files -e ! stream -bv /d1'

Under the stock shell, proc files are limited to executing a 'canned' set of commands. For example, changing directories before running a program. Under MultiVue, they can be used to extend an AIF's functions- if you inspect any of the Tandy vdg game to MultiVue patches, the AIF always calls a procedure file to set up the vdg screen before launching the program.

Shell+ adds user input and some logic to the proc file. With these tools it's a simple matter to write a menu 'front end' for any difficult to use utility.

### Creating Proc Files

Any text editor, word processor, even 'build' can be used to create a proc file. An extension isn't used by OS-9 to identify proc files, so each should have a simple, one word name, similar to executable programs. Making a true ascii file on most word-whackers is simply a matter of leaving the printer control codes out, but if yours offers a separate ascii save option, use that. Also, some OS9 utilities (notably dsave) automatically create proc files which can then be edited before executing.

Normally, a proc file is placed in a data directory. This is useful on text screens, a dir lists the available files. But you have to be in that directory, or type a complete pathlist, to get to the file. Shell+ avoids this limitation by

allowing proc files in the cmds directory. Just set the execute attribute (remember to reset it if you edit the file later).

Now you have to remember which proc files you have, but there are two instances where this is a good thing. First, system wide procs files (like the one that switches my printers) are usually pretty memorable. But the best case is a proc file launched by a MultiVue AIF. You don't need to see it with the AIF, don't want to see it's ugly uniconed box, and hate to see how all these odd little files slows down MVue's movement. So put them in CMDS and don't ever go there!

One other location is allowed by shell+, RAM! There is a pd utility called datamod that merges multiple text files into a 'data module' which can be loaded like merged executable files. Floppy users can load many proc files (as a single 8K datamodule) from the boot disk before swapping to runtime.

### Display & Output

Echo is the command used to display text within a proc file. It's very similar to a batch files ECHO- command line symbols <>>>!;:&^ are taken literally and can't be echoed. Leading spaces are also useless for formatting text, since they are taken to be a single command line separator.

In MS-DOS batch files, the commands ECHO ON and ECHO OFF determine if the text of the batch file is displayed as it executes. OS-9 uses the (almost unknown) t and -t command. For example

```
t
merge /dd/sys/stdfonts /dd/sys/
stdpats_4.....
-t
```

to explain the long disk load in startup. T is also a good debugging aid. Just add to the start of a problem file to see each line before it executes. If t is in effect, shell+ also prints the results of a comparison (true or false) after the source line itself.

A very helpful adjunct to echo is OS9's display command. If you haven't tried display yet, turn to the 'Windows' section of your manual. All of the general commands listed there can be issued from a proc file with 'display code parameters'. Some of the basics

## The Great Hoffman Caper

I let this guy write ONE article and he tries to take over the whole column...

Well, this is actually all MY fault, and I owe Rick Ulland (and readers!) an apology! You see, I picked up the FAQ (Frequently Asked Questions) list by Russell Hoffman from the CoCo List on Internet with the intention of using it in a future issue. Rick ran into some hard drive problems on his system and lost the file for volume 2 number 3. So I used Russell's FAQ in place of Rick's article. Then Rick came through and delivered a day or two before that issue was sent to the printer. A couple frantic hours of re-editing removed the FAQ and placed Rick's article instead, only I forgot to change the name! So Russell was actually Rick...

For volume 2 number 4, I correctly used the FAQ, since Rick was still busy trying to recover from several hard drive crashes. He was having a hard time finding the problem and getting his system reliable again! Now most hard drive users won't have this problem, but Rick's is, to put it mildly, severely hacked... hardware AND software wise. One went bad and....

To continue this story, I had a printer driver problem when getting volume 2 number 5 ready. The problem was PageMaker. I had changed the printer driver for a job, and didn't change it back before trying to print 268'm. PageMaker saves your file with the chosen printer's codes, so this messed things up big time! I had about 3/4 of the magazine done, only a couple articles to place (Rick's being one of them!), and only four days before printing. Then I looked at the file and everything was out of place!! Had to go through the entire magazine file and adjust EVERY article and little item so that it looked right again. That's why a few items looked strange...

Well that's it! Not an excuse, mind you, but an explanation of what happened. And a much deserved apology...

include:

```
display c    clear screen, home cursor
  1    home cursor, no clear
  2 x y    cursor to any x y
  7    bell
```

Display includes everything a CoCo can do, so if you really need blinking underlined reverse video in your proc file try `display lf20 lf22 lf24 ;-`. Even graphics- there are a nice selection of graphics balls, boxes, and fills, with unusually good ellipsoidal tendencies, even a set of relative 'draw' commands.

One of my favorite displays is the old overlay window trick. Sometimes you find yourself on a screen you don't want to mess up. `display lb 22 1 xstart ystart xsize ysize forecolor backcolor` opens a nice overlay to run your proc file in. `display lb 23` when you leave.

### Shell+ adds more options

**Pause text** echos text, then waits for key or mouseclick.

**Prompt text** echos text, cursor stays at end of text (no cr). Used as an input prompt for `var.x (qv)`.

When writing to a file, one feature OS-9 has always lacked was an append/overwrite switch. Now we have them-

```
>+ >>+ >>>+ filename
append to end of filename
>- >>- >>>- filename
overwrite filename
```

### Input

Under shell+, there are 10 shell variables %0 to %9. These look alot like MSDOS shell variables, but are completely different. (msdos vars pass arguments from the command line typed to start the file). Shell+ variables are input from the keyboard with 'var.#'. A useless example:

```
prompt Enter disk name ?
var.l
format /dl r %l
```

These variables live as long as the shell does, so if you come out of the proc file using the parent/child swap (covered later) they will still be around the next time a proc file is executed. This is a handy way to pass data between proc files. There is one more user input hidden in the logic statement, which tests for a one key (y/n) response.

Also added is a new type of module, the 'shellsub'. These asm routines return data to your proc file in 10 read only variables %%0 to %%9. Sdate is included in the shell+ archive. To see what it does:

```
var.?
sdate bin (assuming it's in cmds)
var.?
The double percent variables can be used
```

directly in comparisons, but can't be changed. They can be assigned to a regular variable for further manipulation (see variables).

Although not technically 'user' input, there are many utilities intended to supply data to (or from) procedure files. Many are clones (or subsets) of unix utilities- for example ls. Depending on which ls you have, this generates unformatted lists of all the files in a directory- some sort the output in various ways. A similar utility is included in the shell+ archive- `param` performs wildcard expansion, then feeds the resulting filelist out one per line- for example `param she*` might list shell, shellmate, shell.doc, etc. With tools like `param`, utilities can be written simply and the selecting done from shell.

### Program Flow and Logic

Procedure files aren't limited to one command per line- in fact you can cram 280 characters in a line. There are three 'separators' used- semicolon for 'normal' sequential execution, an ampersand for concurrent (multitasking) execution and the exclamation point, which 'pipes' output from the first process to the second.

Especially in the case of printed output, a little multitasking can speed things up, but be careful with utilities that use the disk drives. Your data will be fine, but the drive keeps shuffling back and forth between the two- it can take awhile if the files are many tracks apart...

The stock shell flows one way. (I'll let you name the direction). It does allow comments by starting a line with \*. Both shells share a few commands that affect how the file executes. We've already discussed `t` and `-t` (the echo modifier). To this add `x` and `-x`, the error abort switch. And the last single letter switch is `p` and `-p`. This switch turns off 'prompting'. When you are force feeding a procedure, `-p` prevents it from sending input prompts like OS9:

### On to GOTO

The high-level guys may smirk, but if you've only got two commands, one had better be GOTO. Without line numbers, the first 40 characters after a comment \* are used as labels. Goto label starts looking at the start of file, while goto +label starts at the current position. The label \*\ stops any goto search right there. Between these 3 options, it's possible to do some fancy stepping.

Almost as important, `onerr goto label` (and the related `onerr goto +label`) provide enough error trapping ability to keep the average user running. A special variable

%\* contains the last error code generated.

Our ten variables. %0 - %9 aren't typed at all, just raw storage. It's up to you to keep the numerics and the alphas separated when you start comparing things later. Where # is 0 to 9:

```
var.#    up to 80 chars from keyboard.
var.#='string' assign up to 80chars
constant
dec.#    decrease # one
inc.#    increase # one
Everybody knows this construct:
if test then statements else statements
endif
```

spread over several lines. The 'test' itself is unique- first a special case. `-y` (used alone) tests one character from the keyboard. It's either true (y) or false (n). For normal boolean tests, the compare defaults to ascii. Leading the test expression with a + forces numeric comparison. More example:

```
var.l= 2.00
if%l=2 then..... is false. "2" <> "2.00"
if +%l= 2 then..... is true. 2=2.00
any semi-normal operator is accepted. <
<= =< = >= => >
```

Since after all, the main reason for a DOS is managing disk files, there are many file tests.

```
-f path    true if file exists
-r path    true if file exists and readable
p
-w path    true if file exists and writable
-e filename true if file exists in
           execution dir
-d path    true if file is a directory
```

A false test jumps to else or endif (or it's alternate spelling, fi) and errors out on overrange ( >80 characters ) or absence. Unless abort on error is turned off- then an error is treated as true. (Unlike system errors, proc file errors can't be trapped).

That's it. If you are used to a language like basic09, the above looks a little sparse. But if you think of what the high level commands do, and you'll see most anything can be accomplished. Inc and dec make handy loop counters (both wrap around at 65535). C style 'case' structures are emulated with goto %#. (I used to do that on my Sinclair!) The only difference between while/do and do/until is which end of the loop you put the test at....

*(continued on page 9)*



Over the past couple issues, we've been working on our sample termcap program. Last issue, we discussed the basic required variables and functions for termcap programs, and their initialization sequence. In this issue, we learn how to use termcap functions to control the terminal's cursor and to read keyboard input from the terminal.

### Sending Terminal Control Strings

During the initialization sequence, we obtained all the terminal control strings that we needed to control the terminal's cursor. For instance, "CM" (cursor movement) points to the string of characters that will move the terminal's cursor to a new location. We need to refer to this variable whenever we wish to move the cursor.

Moving the cursor (or at least preparing the set of codes to allow the terminal to perform the task) actually requires a bit of processing, but luckily there is a library function to take care of this for us called `tgoto()`. By giving `tgoto()` our CM pointer and our desired cursor coordinates, it will return a pointer to a control string that can be sent to the terminal to reposition the cursor.

As it turns out, we always use the `tputs()` library function to send out control strings to our terminal. `tputs()` needs a pointer to a control string... Hey, that sounds familiar! That's exactly what `tgoto()` returns for our move-cursor operation, which means we can combine the `tgoto()` and `tputs()` calls together in one function! So, to reposition the terminal's cursor to coordinates (x,y) use: `tputs(tgoto (CM,x,y), 1, user_tputc)`;

You should notice there are two other parameters for the `tputs()` function ("1" and "user\_tputc"). The "1" is what's referred to as "lines\_affected", and relates to our discussion of the "ospeed" variable in the last issue. On certain (older) terminals, a delay is needed to allow the terminal time to process special requests. The "lines\_affected" parameter is used in calculating how long the delay should be. As you may recall, we set our `ospeed` variable to -1, which means no padding delay is necessary so the "lines\_affected" we specify here really is of no consequence. The "user\_tputc" parameter is just a pointer to our user-defined function, whose task it is to output a single character to the terminal. Review the last issue for more information on this.

Looking at the `tputs()` call above, it really would be annoying to have to remember to type the "1" and "user\_tputc" parameters each and every time we move our cursor or do any special terminal operation in our program. Plus, what if we decide to rename our "user\_tputc" function? If we specified "user\_tputc" several times within our program, we'd have quite a few changes to make. So, you will notice in our termcap program we wrote another function named `putpad()` which only requires a string pointer be passed to it. This means in our program, we move the cursor to the coordinates (x,y) like this: `putpad(tgoto(CM,x,y))`;

To clear the screen: `putpad(CL)`;

If we had initialized a 'US' (underline start

variable, we could turn on underlining like this: `putpad(US)`;

Now that's more like it! Easier to understand. For more examples on this, you can look in your *Microware C manual* under "Using the Termcap Library". You will note that "CM" is the only variable that requires the use of another function (namely `tgoto`).

### Reading Input from a Terminal

This may seem like a trivial task to the unsuspecting programmer. Why can't we just use the `scanf()` or `read()` functions to get data or characters typed in by the user? The reason is our program must appropriately respond to any arrow keys the user may press, including the backspace key. These keys may send more than one character back, depending on the terminal. On one terminal, the up-arrow key may send back only a one-byte code, whereas on a VT-100 terminal a 3-byte string may be sent. It is this variable-length keypad string that is the cause for all the complexity.

Unfortunately, there are no library functions to help us out. So, let's think about this for a moment. Whenever the user presses a "normal" key (the 'J' key, for example), we want to immediately process that keypress by displaying it on the user's screen. If the user presses a special key (arrow-key, home, or backspace), we want to immediately process the keypress by moving the cursor to an appropriate location. The problem is the terminal data will be coming in one byte at a time, our input routine must be smart enough to realize if it is in the middle of processing a special key and know when to decide if a sequence of keys does not match any special key.

There is the interesting question of what to do if our program started to notice a special-key match, but turns out to be wrong. For instance, let's say the terminal we're using has it's up-arrow key defined to send the four characters "+SUP" (which would be very weird, but this helps get across the concept easier). Now, pretend you're the user and you are working on some numbers and you start to type "+\$100,000"... what is our program thinking after each character is pressed, and how should it respond?

When our program receives the "+" character, it notices that the user may have pressed the up-arrow key so no processing can yet take place. The "+" character must simply be placed into a buffer and the program waits for another character. Then our program receives the "\$" character and remembers that it's in the middle of checking a special-key string. It still cannot determine whether or not the up-arrow key was pressed, so the "\$" character is also buffered and the program waits again. Now, when the "1" is received, it is obvious to our program that the up-arrow key was not pressed. That means we can store the "1" character in our buffer, and then empty our buffer one character at a time to the user's screen. That is the important point here... when our program thinks it might be receiving a

special-key sequence and it turns out to be wrong, we must then remember to send any buffered characters to the user's terminal (perhaps with the exception of unprintable characters).

I realize that sounds really tough (and maybe that's why there aren't a whole lot of termcap input programs out there), but the logic for "process\_key()" isn't too bad:

```

if (EMPTYING BUFFER)
    remove next key from buffer
    display key
else
if (NO KEYS IN BUFFER)
    read key from terminal
    if key matches a special-key string
        process special key
    else
    if key partially matches special-key string
        buffer key
    else
    display keypress
else
if (KEYS IN BUFFER)
    buffer key
    if keys match a special-key string
        process special key
        destroy buffer
    else
    if buffer does not partially match special-
key string
        turn on "EMPTYING BUFFER" flag

```

Looking at the "process\_key" function in our sample termcap program, you should be able to match up the logic to the source code. As always, if you are having trouble please feel free to write to me.

You will notice when an arrow-key has been detected by the program, I have written it so the 'cx' and 'cy' variables (which I use to keep track of the current cursor position) are updated, and then later the cursor-move (CM) control string is sent out to the terminal. For the backspace key, I blot out the character to the left of the cursor in addition to moving the cursor to the left.

Well, that's all for this issue. I sincerely hope the past few issues has provided a valuable tutorial on how to utilize the termcap C library and how to write termcap programs. Writing termcap programs which only do output isn't too tough. The hard part comes when you want to process input from the user, but now you have the routines written to take care of this for you!

< 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

# Programming in "C"

## Using FIELDS.

P.J. Ponzo

Suppose we wish to manipulate the results of a questionnaire containing 20 questions with answers either YES or NO. If we store the answers in an integer variable *x*, with *x*=1 corresponding to a YES answer and *x*=0 to a NO, then (since integers occupy 2 bytes of memory, usually) this would require 40 bytes per questionnaire and, if 1000 people answered the questionnaire the results would occupy 40,000 bytes ... to much!! SO, we store the answers in a 1-byte character variable *x* with either *x*='Y' or *x*='N'. That would take 20,000 bytes of memory ... also to much! SO (since 1=YES and 0=NO only needs 1 bit of memory) we store the results in 20,000 bits ... let's see ... at 8 bits per byte that would take ... 20000/8=2500 bytes ... just right. Is there a DATA TYPE called bit (so we could declare bit *x*;) ? Alas, there is NOT ... but we can create a structure with members which occupy bits of memory!

```
struct byte { /* define a structure called byte */
    unsigned member1 : 1; /* member occupies 1 bit */
    unsigned member2 : 4; /* member occupies 4 bits */
    unsigned member3 : 3; /* member occupies 3 bits */
} x; /* declare x to be such a structure */
```

Now *x* is a structure of type *byte* which has 3 members (in this example). Each member is an unsigned integer. These members are made to occupy a number of bits. The colon : followed by an integer (like 1, 4 and 3) arranges this for us. Now we can (as usual) refer to *x.member1*, *x.member2* and *x.member3* and (for example) say: *x.member1*=0; or *x.member2*=13; , etc. Since *x.member1* is 1 bit wide, it will hold numbers 0 and 1 only. Since *x.member2* is 4 bits wide, it will hold numbers 0,1,2,..., 15. Since *x.member3* is 3 bits wide, it will hold numbers 0,1,2,...,7... and the whole structure, with all 3 members, only fills one byte of memory (since 1+4+3 bits =1 byte).

The members, occupying a number of adjacent bits in memory, are called FIELDS. If we define another struct called *bytes*, and declare \**x* ...

```
struct bytes { /* define a structure called bytes */
    unsigned member1 : 1; /* member occupies 1 bit */
    unsigned member2 : 4; /* member occupies 4 bits */
    unsigned member3 : 4; /* member occupies 3 bits */
} *x; /* declare x to be a pointer */
```

so that *x* is now a pointer to a structure, then we would access a member with: *x->member1* (remember?) If you're really squeezed for memory space, then cram as much into a 2-byte integer as possible by using bits.

Note that, in the above structure, we now have 1+4+4 bits which is more than a byte will hold, so the compiler will put *member3* into a second byte (and "waste" the remaining 3 bits of the first byte and probably the remaining 4 bits of the second byte too!)

If you're still in need of memory space you can arrange to

have several variables occupy the same memory space ... but not at the same time This union of variables is called a ... UNION (what else?).

```
union sam { /* define a union called sam */
    int x; /* the first member is an int */
    float y; /* the second member is a float */
    char z; /* the third member is a char */
} jeckyl, *hyde; /* declare some unions */
```

Since these 3 variables *x*, *y* and *z* occupy different amounts of memory the variable *jeckyl* will occupy sufficient memory to hold the largest of *x*, *y* and *z* (in this case, it's *y*). You may point to a union (*hyde* is a pointer) and/or access one of its members (*jeckyl.x*=123 or *hyde->z*='A').

```
union sam { /* define a union called sam */
    int x; /* the first member is an int */
    float y; /* the second member is a float */
    char z; /* the third member is a char */
} jeckyl, *hyde; /* declare some unions */
```

If you say: *jeckyl.x*=123 ( an integer ) then the space set aside for *jeckyl* will be occupied (in part) by the integer 123. If you then say: `printf("%f", jeckyl.y);` the `printf` function will go to the memory location where *jeckyl* lives, interpret the 2-byte int 123 as a 7-byte float ('cause we said %f) and print garbage! MORAL: *jeckyl* and \**hyde* will contain an int, float or char depending upon whether the last thing you put there was an int, float or char! So, keep track with some variable called *WhoThere*.

```
#define CHAR 1
#define INTEGER 2
#define FLOAT 3
.....
int WhoThere;
.....
```

Each time you say:  
*hyde->y*=12.34; then also say: *WhoThere*=FLOAT;  
so you can check if *WhoThere*==CHAR or *WhoThere*==INTEGER etc. to see who's currently a member of this union! Easy eh?

< 268'm >

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



244 S. Randall Road • Suite #172 Elgin, Il. 60123  
 (708)742-3084 eves & ends • MO, Check, COD; US funds  
 Shipping included for US, Canada, & Mexico

**MM/1 Products (OS9-68000)**

**COMING SOON!!! CDF - CD-Rom file Manager!** Unlock a wealth of files on CD with the MM/1!!

**VCDP \$50.00** - New Virtual CD Player allows you to play audio CDs on your MM/1!! Graphical interface emulates a physical CD player. Requires SCSI interface and NEC CD-Rom reader.

**KLOCK \$20.00** - Optional CUCKOO on the hour and half hour!! Continuously displays the digital time and date on the /term screen or on all open screens. Requires I/O board, audio cable, and speakers.

**WAVES vr 1.5 \$30.00** - Now supports 8SVX and .WAV files !! Allows you to save and play all or any part of a sound file. Merge files together or split into pieces. Record, edit and save files with ease. Change playback/record speed. Convert Mono to Stereo and vice-versa!! Record and Play requires I/O board, cable, and audio equipment.

**SOUND CABLE \$10.00** - Connects MM/1 sound port to stereo equipment for recording and playback.

**GNOP \$5.00** - GNOP is the AWARD-WINNING version of PONG(tm) exclusively for the MM/1 !! You'll go crazy trying to beat the clock and keep that @\$%& little ball in line! Professional Pong-ists everywhere swear by (at) it !!! Requires MM/1, mouse, and lots of patience.

**Coco Products (DECB)**

**HOME CONTROL \$20.00** - Put your old TRS-80 Color Computer Plug 'n' Power controller back on the job with your Coco 3!! Control up to 256 modules, 99 events!!

**HI & LO-RES JOYSTICK ADAPTER \$27.00** - Tandy Hi-Res adapter or NO adapter at the flip of a switch!!

**KEYBOARD CABLE \$25.00** - Five foot extender cable for Coco 2 and 3. Custom lengths available.

**MYDOS \$15.00** - CUSTOMIZABLE! EPROM-ABLE! The commands Tandy left out. Optional 6 ms. disk drive speed. Supports double-sided and forty track drives. Set CMP or RGB palettes on power-up. Power-up in any screen. Speech and Sound Cartridge supported. Point and click mouse directory and MORE. For all Coco 3 with Disk Basic 2.1. More options than you can shake a joystick at!!

**DOMINATION \$18.00** - MULTI-PLAYER STRATEGY GAME! Battle other players armies to take control of the planet. Play on a hi-res map. Become a Planet-Lord today! Requires CoCo 3, one disk, and joystick or mouse.

**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 Converter 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**

*Want to move your CoCo keyboard? Get a*

**Puppo PC/XT Adapter**

Allows use of any PC/XT compatible keyboard with your CoCo. Easy access to OS-9, DECB, and other options. Hold space bar on power-up to skip menu. Switchable and most auto-sensing keyboards supported.

**Only \$72.50 post paid!**

**\$100.00 post paid with keyboard!**

**FARNA Systems**

**Box 321**

**Warner Robins, GA 31099-0321**

NOTE: Keyboards will be 101 key. FARNA reserves the right to substitute 84 key models if necessary without notice.

**ROMY-16**

**EPROM EMULATOR**

- Emulates ROMS (2716-27010) or RAMs in 8- and 16- bit systems.
- Window/menu driven interface.
- Provides 8 hardware breakpoints for 8-bit systems.
- **\$195** (2716-27256) or **\$245** (2716-27010), 90 day warranty.
- 15 day money-back guarantee.
- Optional assembler, disassembler, and ROM debugger add \$100.

**Universal Microprocessor Simulator/Debugger V2.1**

- Simulates Z8, Z80, 64180, 8048, 8051, 8085, 6800, 6801, 6805, 6809, 68HC11, 6303, 6502 & 65C02.
- Assembler, Disassembler, & Windowed Symbolic Simulator. Supports on-board debug through RS232.
- **\$100** each CPU (S&H \$8)

**6809 Single Board Computer**

- Support for 8K RAM, 8K ROM.
- Two 6821 PIAs connect 32 bits of I/O to outside world.
- No jumpers for 2732, 2764, & 6116.
- Two interrupt signals on CPU bus.
- Size is 2.75"x5". **\$60** each board.
- For an integrated development system with assembler, disassembler, and on-board debugger please add \$70.

**68HC11 Microcontroller Development System**

- Eight channel 8-bit A/D converter. 32K ROM and 32K RAM.
- \$120 each SBC. Add \$70 for assembler, disassembler, BASIC interpreter and on-board debugger.

**J&M Microtek, Inc.**

83 Saman Road, W Orange, NJ 07052  
 Tel: 201-325-1892 • Fax: 201-736-4567

# Basic09 In Easy Steps

Chris Dekker

## Advanced Programming: file management

It is number 10 in this series already. It also could be the last, although the subject of this article, more advanced programming, will stretch over several installments. Before we get into that, however, I want to spend a few lines on explaining why the last; why the question marks.

Actually I have a feeling that I have said all I wanted to say about Basic09, which is not the same as having said everything there is to say about it. There is much more to say and explain about it. I suppose I could write a book bigger than the OS-9 manual on it by simply explaining all the possibilities I have found so far. But that wasn't the intention here. I just wanted to provide a framework and show in some broad strokes how (to make) things work.

Why the question marks?? As I write this it is still March (1994) with lots of snow on the ground. Three parts of the series have been published and number four is about to be shipped. According to my editors, the initial response to the series is quite positive. If enough people start asking how to do this or that, I might write some articles on specific items. We'll just wait and see.

Now, advanced programming. What is it?? Actually I don't know. It just sounded too good to leave out. I suppose the closest thing to an explanation is that it has taken me quite a few years to figure some of these things out. That doesn't mean I have been working on those things that long. Sometimes the need just wasn't there. Like routines dealing with the 6309 processor, which became feasible a few months ago after I installed my own 6309. On the other hand, I suppose writing a routine that can figure out what processor is in the system wouldn't be the first program one ever writes.

Basically I want to divide this article into three pieces: file management, keyboard/screen I/O and memory management. These are three major areas where a little ingenuity goes a long way to pushing Basic09 beyond what seems possible at first glance.

We'll start out with the most comprehensible of the three: file management. Although Basic09 has the basic "file commands" built in, it is not very sophisticated in that area. That wasn't really necessary either. Remember that it was developed along with OS-9, so everything OS-9 could do didn't have to go into Basic09.

That leaves us with something I stressed the importance of before: system calls. Since I will use them a lot over here, here is the initialization code:

```
TYPE registers=cc,a,b,dp:BYTE; x,y,u: INTEGER
DIM regs:registers
```

For our purposes we don't use the dp register, but it's got to be there to satisfy "syscall"; the utility that does the dirty work. I will start out with something simple that is sometimes hard to get by without: filelength. One may need to know this to figure out how many records there are in a file, whether the file fits into a buffer etc. The following code will get you that information:

```
DIM path:BYTE
OPEN #path,file:READ (WRITE or UPDATE
are fine too)
regs.a=path regs.b=2
run syscall($8D,regs)
filelength=65536*regs.x+regs.u
```

To wit: \$8D means we are using a system call labeled as I\$Getstt. This along with it's "sister call" (I couldn't find a better term) I\$Setstt, is probably the most used system call of all. These two have another unique feature: they do not perform one function, but can perform literally dozens of functions based on the value in register b.

Since we set register b to 2, I\$Getstt knows we want it to get the current filelength for us. It reads four bytes from the file's descriptor and returns those in registers x and u. Note that to get an accurate reading on large files, "filelength" must be defined as a REAL number. By the way, descriptor is the name of the first sector of a file that is used to hold a standard file header and information on where the various parts of the file actually are on the disk.

Now suppose your program has accessed the file a number of times and lost track of where it is (this shouldn't happen, but...). There is little to worry about because OS-9 did keep track of it's filepointer and knows exactly where it is. So all you have to do is get to that information. Again I\$Getstt helps out. This time through function 5:

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

Now you, too, know where you are in the file. Your next read or write operation will start at the byte pointed to by "filepointer".

Another problem that pops up from time to time is setting the filelength. You may have deleted some records from a file and a

sorting routine has squeezed out the empty spaces. Unless your program uses some kind of marker to positively identify the last record, next time you run it you will be loading some junk. The reason here is that OS-9 will not voluntarily shrink a file. The reasoning behind this is that sooner or later your file will expand again to cover that space. Leaving the empty space with the file thus reduces file fragmentation and gives you better disk access times when loading or saving data.

If this causes your program to load junk, here is how you can adjust the filelength:

```
regs.a=path regs.b=5
RUN syscall($8D,regs)
regs.b=2
RUN syscall($8E,regs)
```

As you can see, we first pick up the filepointer and then execute a I\$Setstt call (\$8E) to set the new filelength. This method assumes that the last byte written to the file is actually the last byte in that file. This is no problem in sequential access files (text files, etc.) where this is a safe assumption. However you must be careful with direct access files (keeping records) that only a module that write all records to the file in sequence can execute this code.

If, for instance, you just edited record 2 in a file of a 100 records and then execute this code you will loose 98 records in a hurry. Worst of all, the disk space occupied by those 98 records is now marked as FREE and can be overwritten at any time, making recovery impossible.

A little earlier I mentioned the file descriptor. There is a system call that specifically accesses that file descriptor, although this is not mentioned in the OS-9 manual. I read about it in the Rainbow and will pass on this information here.

The following code allows you to read the entire file descriptor:

```
OPEN #path,file:READ
regs.a=path regs.b=15
regs.x=ADDR(buffer) regs.y=256
RUN syscall($8D,regs)
```

Make sure you have DIMensioned a buffer large enough to hold 256 bytes. Page 5-4 of the technical reference in the OS-9 manual does a real good job explaining the contents of your buffer, so there is no need to repeat that here.

This call also has an accompanying Setstt call, but this works quite differently. If you

replace \$8D in the code above with \$8E, you will find that only three fields in the descriptor actually get updated: owner, date created and date modified. It would be real nice if we could update the file attributes also using this system call, but unfortunately that doesn't work.

So how does one alter a file's attributes? By far the easiest approach is an indirect one through the use of the attr utility. Your program can assemble a string with the attributes in the same way as you would type it in from the command line, and then use the SHELL command to execute it:

```
DIM attributes:STRING[10]
attributes="e w r"
SHELL "attr "+file+" "+attributes
```

In this example "file" is a string variable holding the name of your target. There is another way of resetting the attributes which goes roughly as follows: you open a path to the directory that holds the file, read that directory entry by entry until you hit the file's name and pick up the LSN of the file's descriptor there. Once you know the LSN, your code would look something like this:

```
DIM attr,path:BYTE
OPEN #path,"/d0@":WRITE (assumes file on /d0)
SEEK #path,LSN
attr=7 (set attributes e w r)
PUT #path,attr
CLOSE #path
```

This approach works very well (and fast) in certain programs. CoCoTop, for instance, uses code similar to this. But keep in mind that CoCoTop always holds filenames and LSNs (of the directory you are looking at) in its internal buffers. It also employs a special ML subroutine to find a file's name very quickly.

For most other programs the overhead associated with picking up a file descriptor's LSN makes this route less than appealing. Nevertheless, if you want to try: OS-9's directories are files containing sets of pre-defined records. That record's definition is:

```
TYPE record=filename:STRING[29]; sector
(3):BYTE
DIM dirrec:record
OPEN #path,directory:READ+DIR
SEEK #path,64 (skip first 2 entries)
WHILE NOT(EOF(#path)) DO
GET #path,dirrec
REM check filename here
REM exit loop when found
ENDWHILE
```

Use the following to get the LSN::

```
LSN=dirrec.sector(1)*65536
```

```
LSN=LSN+dirrec.sector(2)*256
LSN=LSN+dirrec.sector(3)
```

If we go back for a moment to the code for updating file attributes, you will see the term "/d0@" in the OPEN statement. This is way a of accessing a disk that is both very powerful and very dangerous. In effect you tell OS-9 to treat the entire disk as one large file, accessible on a byte for byte basis.

You already saw the "powerful" in getting to bytes that otherwise can not be reached. The "dangerous" aspect is that you are working outside the protection OS-9 normally offers. Especially if you open a path in WRITE or UPDATE mode, you can wipe out the disk's allocation map and root directory in a hurry, leaving your disk all but useless.

The same is true for unwanted alterations of the disk's identification sector (LSN 0). If you just want to inspect some of the values in those sectors, however, you can open the path in READ mode, which is quite harmless since OS-9 will now refuse to write anything to the disk. After opening the path you can access the identification sector with SEEK #path,0 and the allocation map with SEEK #path,256.

The definitions for those sectors can be found on pages 5-2 and 5-3 of the technical reference section of the OS-9 manual. If you want to read a disk's root directory, you should first read and process bytes 9-11 (=offsets 8-10) of the identification sector. Although this is a little extra work, it ensures that your program runs on any kind of hard or floppy disk.

If you take a look at your OS-9 manual's technical reference you will see (on pages 8-44 through 8-65) a complete list of system calls labeled I/O system calls. To separate those from other calls their name starts with an I. A lot of these calls (like I\$OPEN, I\$CREATE, I\$CLOSE) have equivalents in Basic09 commands like OPEN, CREATE and CLOSE.

Actually, when Basic09 receives a command like that it simply executes a system call. As a result it is usually easier to use the Basic09 command than to set up the entire system call. There are a few exceptions though. Like I\$attach and I\$detach. The only time I explicitly use those are with windows, which is not what we are discussing here. For the curious: OS-9 always executes these calls when opening and closing paths to devices but this is transparent to the user.

Another exception is I\$Dup. Just about everyone uses this call from time to time, although probably unknowingly. For instance: OS-9 executes I\$Dup every time you redirect output like in: list file >/p.

There is, however, very little use for it from within most Basic09 programs, so I will skip it here.

That leaves one call I do find useful, though mostly in installation routines for packages I sell: I\$Mkdir. This call creates directories. OS-9's mkdir utility, for instance, uses this call. It is also very simple to use from within Basic09 and serves as an easy to understand example.

Any half-decent Basic09 program that does some file access should have string variables that hold a pathlist to a directory and a filename. Up front this may look like overkill: you can, after all, code a pathlist right into the OPEN statement. However the advantages in added flexibility and easier error recovery more than make up for this. So all you really need is a pointer to the string; set register b and execute the system call. It looks like this:

```
DIM directory:STRING[30]
directory="/d0/TEXTFILES"
regs.x=ADDR(directory) vregs.b=63
RUN syscall($85,regs)
IF LAND(regs.cc,1)=1 THEN
ERROR regs.b \ ENDIF
```

This code attempts to create a directory called TEXTFILES in the root directory on drive /d0. If it is successful it creates the directory. If not: no harm gets done to your disk. For instance an existing directory with the same name simply gets left alone and no duplicates are created. Note that by setting register b to 63 we set the following attributes for the directory: r w e pr pw pe; thereby allowing full access to it. If you want to restrict access to the directory: use another value.

The IF statement isn't really necessary but comes in handy for tracking typos in a pathlist. It also serves as an example. There are plenty of calls that shouldn't go without it. Like I\$read to avoid crashing a program by trying to read and/or process non-existing data. Or I\$write trying to save data on a full disk and you coming back the next day to discover your work disappeared when you turned off your computer, etc..

Well,, my allotted space is full once again. Next time we'll take a look at manipulating screen, keyboard and the likes. Till then, keep on hacking. < 268 m >

Chris can be reached in care of  
this magazine or directly at:  
**Chris Dekker**  
RR #4  
Centreville, NB E05 1H0  
CANADA

# MM/1 Update

David Graham

## Desktop OS-9 Development and Marketing Association formation.

Hello all! Once more, we bring you mixed news from the home of the MM/1. Our SCSI port trouble shooting continues, no success in obtaining a completely functional system yet. As before, our troubleshooting team continues to work. Kreider Electronics continues to work on their new 68306 based board. Scheduled for release last month, we hope to release this machine in February or March, 1995.

Sub-Etha Software is expanding it's OSK based efforts. We welcome Sub-Etha owner Al Huffman to the ranks of MM/1 owners. Look for more games from this fine company, to be released in April at the Chicago CoCo Fest. The future looks bright for applications as well. We expect the long awaited FoxBase DBM library to be released in April by BudgetWare, and we have programmers even now working on scheduling software for small to medium sized businesses. Also on the agenda are OS-9 based POS systems, fax/modem support software, and improved and expanded checkbook programs. We're excited about 1995. The OS-9 applications movement is picking up momentum, and more and more people are jumping on the bandwagon!

At the Atlanta CoCoFest, vendors banded together to form a marketing and development association. Today, Colin McKay, Frank Swygert, and myself are working on the articles of incorporation for this organization. Bill Wittman and Tim Johns were also planning to be involved, but interruptions in E-Mail service have slowed things down here. The aims of this organization will include gathering marketing data and information useful in increasing sales of OS-9 based computer and software services centered around the desktop. Currently, the general applications market for OS-9 is almost non-existent. But there is something each and every one of you can do. Answer the included survey today. Tell the vendors supporting your market what programs you want most, and what you are willing to pay for them. Then, buy the programs we generate in response. The Desktop OS-9 Development and Marketing Association (DODMA) will distribute the results of your mail to all member vendors. We'll also work to assure that information is shared between our members in such a way as to improve the overall quality of available programs. DODMA can be reached at :

Desktop OS-9 D&M Association.  
P.O. Box 10552, Dept DDMA  
Enid, OK 73706-0552 USA

I plan to enlarge upon this survey in the following months, based on information from the first returns. We look forward to your responses.

DODMA SURVEY #1 March 1995  
Name : \_\_\_\_\_  
Address: \_\_\_\_\_  
City : \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_  
Occupation: \_\_\_\_\_  
E-Mail Address: \_\_\_\_\_

Annual Income: (Optional)  
10000 or less \_\_\_\_\_ 10001 to 15000 \_\_\_\_\_  
15001 to 25000 \_\_\_\_\_ 25001 to 35000 \_\_\_\_\_  
35001 to 45000 \_\_\_\_\_ 45001 to 65000 \_\_\_\_\_  
65001 or more \_\_\_\_\_

Family Size : Number in household:  
Adults (21+ years) \_\_\_\_\_  
Children 0 to 4 years \_\_\_\_\_  
Children 5 to 10 years \_\_\_\_\_  
Children 11 to 14 years \_\_\_\_\_  
Young Adults 15 to 20 years \_\_\_\_\_

OS-9 Variants : (Check each type you use in your household)  
\_\_\_\_ OS-9/6809 Level 1 \_\_\_\_\_ OS-9/6809 Level 2  
\_\_\_\_ CoCo 3 Graphics? \_\_\_\_\_ OSK Version 2.3  
\_\_\_\_ OSK Version 2.4 \_\_\_\_\_ K-Windows  
\_\_\_\_ G-Windows \_\_\_\_\_ OS9000

Machine Type(s) : (Enter number of each)  
\_\_\_\_ MM/1 \_\_\_\_\_ MM/1A \_\_\_\_\_ KiX-20 \_\_\_\_\_ KiX-30  
\_\_\_\_ System IV \_\_\_\_\_ System V \_\_\_\_\_ TC-70 \_\_\_\_\_ TC-9  
\_\_\_\_ CoCo (1 or 2) \_\_\_\_\_ CoCo 3 \_\_\_\_\_ PT-68K  
Other (Specify) \_\_\_\_\_

Program Types Used : (Please enter number of each type of program owned)  
\_\_\_\_ Word Processing \_\_\_\_\_ Checkbook Register  
\_\_\_\_ Arcade Games \_\_\_\_\_ Spreadsheet  
\_\_\_\_ Data Base Manager \_\_\_\_\_ Strategy Games  
\_\_\_\_ Terminal \_\_\_\_\_ BBS software  
\_\_\_\_ Programming Language (specify) \_\_\_\_\_  
Other (Specify) \_\_\_\_\_ ( )

Suggestions for New Programs : Please fill in your suggestions for programs you would like to buy during the next one to two years. Specify a price range you would be willing and able to pay for such programs.

*Please photo-copy this page or write the requested information on a separate sheet. If you are a programmer/software developer, please inquire about becoming a member of DODMA. We are still working on the group charter. Membership information will be forwarded as soon as it is complete.*

## micro notes

Since CD-i technology relies heavily on OS-9/68K, some of you may be interested in this: A new **CD-i newsgroup on Internet** has passed the call for votes. The new group, **rec.games.video.cd-i**, should be available by the time you read this article. *(Passed along by Colin McKay)*

**The next PowerPC chip: MPC620.** This chip is designed for file servers on networks. It is more like the DEC and MIPS high end RISC chips. The chip has 64 bit architecture, a 128 bit bus to main memory, 32K instruction and data caches (64K total cache), operates at 133MHz, and has on-board hooks for external cache memory of up to 128MB. That's right, up to 128 *megabytes* of off cache memory, not 128K! The bus will be coded to run up to four MPC620's at once with no additional circuitry, making multi-processor systems easier to implement. Should be available by the summer of 1995.

If you want more PowerPC info, try reading "Inside the PowerPC Revolution" (Jeff Duteman & Ron Pronk, \$24.95, Coriolis Group Books, 800-227-8365, ISBN 1-883577-04-7).

**RiBBS to Go** is a package of 5 dsdd 40 track disks with a reworked OS-9 boot disk from what you supply. To run it requires a CoCo III with 512k memory, OS-9 Level II, upgraded Multi-pak, RS232 pak or equivalent, and a minimum of 2 dsdd 40 track floppy drives. A minimum 1200 baud autoanswer modem is needed, but Hi-Speed modems can be accommodated for 9600 baud use. Connects at up to 16,800 will work as my ZyXEL modem will connect at 16,800 to another ZyXEL.

RiBBS to Go is the latest V2.10 and can easily be expanded to more menus, larger msg base and moved to a Hard Drive. Readme info to do this is included along with 2 disks with all docs, and a set of utilities and games. The other 3 disks are the boot disk and a /dd CMDS and /d1/FILES disk (the running part). RiBBS to Go will run 'out of the box' with 15 min. of editing a few files; readme files describe this procedure. It is also recommended to install an RS232 breakout box between your RS232pak and modem and rewire the pins 6-8 reversed between the pak and modem.

Many of you have asked about the availability of **Puppo keyboard adapters** and/or circuit boards. Well, I have some good news! I have been in contact with the person who built these things for Bob Puppo, and he has **received permission to make more!** But I need at least six confirmed orders to get them into production! The cost will be **\$70 each plus \$3 S&H. For \$100 total, you will get the adapter AND an XT keyboard** (so you don't have to worry about finding one)! The keyboard has 10 function keys to the left of the main keys and the arrow keys are not separate but are within the numeric keypad. I'll get 101 keyboards if possible, but 84 key standards may be substituted without notice.

**ANY XT compatible keyboard will work.** The keyboard of choice is the 101 key models with a physical switch between XT and AT modes on the bottom, but MOST (I can't guarantee all!) "auto-switching" keyboards will work as well. If you are seriously interested, send at least a \$20 deposit by the end of February. You will receive your adapter (and keyboard) no later than the end of April 1995.

*"micro notes" is for news on anything that may be of interest to readers, personal classified ads, and new product releases. If you see anything that may be of interest to other readers, have something for sale, or need an item, please write in and let us know! Vendors: be sure and let us know when you have a new product even if you don't already advertise!!*

PC Magazine gives it high marks and says that it is very detailed. While browsing B. Dalton's at the local mall, I also spied a couple other PowerPC books. Maybe this chip will give Intel some serious competition one day!

NEC has developed a **1 Gigabit memory chip** (1GB x 1 bit) which is slated for mass production around the turn of the century. About four hours of music could be stored on such a chip, which measures 1" x 1.5" and transfers the data at 400 MB/second. Hitachi has developed a similar chip. It would take eight of these (nine for PCs!) to make one Giga BYTE, which would require only 12 square inches of board space. Sound like a lot? That's about the same space taken up by the eight SIMM sockets on current motherboards (3"x4")... which holds a maximum of "only" 128 Mega Bytes.

RiBBS will work without this but not really the best way.

### \*\*\* How to get RiBBS to Go \*\*\*

Format 5 DSDD 40 track 5-1/4" disks with OS-9 Lev II. Copy or cobbler one disk with a copy of your boot. Put in a floppy disk mailer or padded mail packet with your return address and a check or M.O. for \$5.00 for S/H. Send to:

Warren Hrach  
4369 Newport Ave.  
San Diego, CA 92107

Voice (619) 221-8246  
BBS (619) 224-4878  
Fido net/node 1:202/744  
Internet warren@ocnbeach.jd.com

I am also an MM/1 Sales Representative. A version of RiBBS is being worked on for the MM/1.

### WANTED

1. OS-9 version of Checkbook Plus with docs.
2. May thru November 1989 issues of Nine-Times disk magazine.
3. 2400 baud modem with docs and cable.
4. July 89 and newer Rainbow on Disk

L. T. Day  
Box 32332  
Columbus, OH 43232

## NEW PRODUCTS

from  
**FARNA Systems!**

### Puppo PC/XT Keyboard Adapters Now Available!!

We now have a source for NEW Puppo PC/XT keyboard adapters. Use the 84-101 key keyboards with your CoCo! Has a menu burned into the on-board EPROM for easy access to OS-9 or DECB and other options. Requires a PC/XT compatible keyboard; original, switchable, or auto-sensing types. Cost is \$70+ \$2.50 S&H for interface alone, \$100.00 post paid with keyboard (101 keyboards are sent if available, but 84 key models may be substituted if 101's not available at this price). Will ship by mid March.

### FARNA Systems

Box 321

Warner Robins, GA 31099

\$2.50 shipping and handling per order.  
Canada S&H \$4.00; Overseas \$7.00

**NOTICE: WE CAN NOW ACCEPT CREDIT CARDS!!!** Visa and MasterCard (only!) can now be accepted for payment. There is a 6% service charge and a minimum charge of \$25. Cards are accepted through a third party and will be billed through "FS Printing".

### CoCo Family Recorder/OS-9 1.0

If you are into genealogy, then the CoCo Family Recorder is the *absolute best* program for the CoCo 3. 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

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

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

### CoCo DECB Software:

CoCo Family Recorder - \$17.50

Genealogy program for CoCo 3. Requires 2 drives, 80 col. monitor.

**NEW! OS-9 Version - \$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 - \$5.00

Collection of 9 classic games. Run from included RAM disk w/ 512K.

Cross Road II - \$5.00

Simple Tic-Tac-Toe, but with amazing sound and graphics! Sound recorded with Digi-Tech.

Space Intruders - \$10.00

Looks just like Atari's classic "Space Invaders"! CoCo 1/2 and 3.

Donut Dilemma - \$10.00

Climb, jump, and ride elevators to top of Donut factory to shut it down! 10 levels. CC 1,2,3.

Rupert Rythym - \$10.00

Collect Rupert's stolen notes, then work our correct sequence. Great action adventure! Get **Space Intruders, Donut Dilemma, and Rupert Rythym** for only \$25.00! Save \$5!

### CoCo OS-9 Software:

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.)

Pixel Blaster - \$12.50

High speed graphics tools for OS-9 Level II. Easily speed up your game programming with this tool kit and subroutines!

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.

### Books:

Tandy's Little Wonder - \$22.50

History, technical info, schematics, peripherals, upgrades, modifications, repairs, much more- all described in detail for all CoCo models! Vendors, clubs, BBSs also listed.

### Quick Reference Guides

OS-9 Level II - \$7.50 OS-9/68K - \$10.50

These handy QRGs have the most needed info in a 5.5"x 8.5" desk-top size. Command syntax, error codes, special keys functions, etc.

### CoCo Hardware:

DigiScan Video Digitizer - \$150

Capture images from VCR, camcorder, or TV camera. No MPI required- uses joystick ports. CoCo Max3, Max 10, Color Max3 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. Send how much space for DECB, OS-9 — \$550.00

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. — \$250.00 (state how much of drive to be used for OS-9)

Controller w/drive cable ————— \$135.00  
OS-9 Drivers ————— \$25.00  
RGB-DOS (for DECB access) ————— \$35.00  
\*\* \$50 for RGB-DOS and OS-9 drivers when purchased together with a controller \*\*  
"Y" cable, \$25 for two position, \$35 for three.  
Drive cables - specify direct to drive or SCSI case type connector ————— \$25.00

**Add \$2.50 S&H per order. Can/Mex \$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 catalogs or letter correspondence to 200 or more persons at once, we can do all work for you for about the same cost of your materials alone! How much is your time worth???

Contact Frank Swygert at above address/phone for quotes



for all your CoCo hardware needs, connect with

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

The main problem with OS9 under a CoCo is the serial port. With a one character buffer, it's hard to do much before the serial port needs service. Windows and OS2 have the same problem. Or did, until National released the 16550 uart-16 bytes of internal fifo buffering gives multitasking systems time to do some.

### Announcing Fast232

Tandy with Sacia			Fast232		
bps	load	thruput	load	thruput	
2400	28.3 sec	237 cps	25.3 sec	235 cps	
9600	73.6 sec	938 cps	31.4 sec	950 cps	
57600	not available		32.6 sec	5373cps	

Local machines with faster modems can now be connected to properly (or improperly at 115K!). More down to earth, Fast232 is a ROMPak sized case, which will accept a daughterboard (once I get the case to close) to give two ports in a very 'concise' package. OS9 drivers by Randy Wilson. Free bonus software! The pd release of 'SuperComm' (Dave Phillipson and Randy Wilson). All software includes 6809 and 6309 versions.

<b>Fast232</b>	<b>\$79.95</b>
<b>Second port</b>	<b>\$45.00</b>

## NEW FOR 1995 FROM DISTO!

1. "Inside 2-Meg": A technical booklet that fully describes how the DISTO 2-Meg Upgrade kit works. Includes schematic, PAL listing, theory and chip by chip circuit explanations. \$20 + \$2.50 S/H.

2. "Blank Board Kit": Includes blank virgin boards (no components) of the SCII, SCI, 4IN1, MEBII, MPROM and Mini Controller. Collect all the components and make your own! \$29.90 + \$4.50 S/H.

3. Call for other DISTO products in stock  
(limited quantities available)

### DISTO

1710 DePatie  
St. Laurent, QC H4L 4A8  
CANADA

Phone 517-747-4851

### Color Computer OS-9 Software

- Invasers09** by Allen Huffman  
High speed machine language arcade game for OS-9. Multi shots and increasing levels.  
Req: CoCo 3, Joystick Optional .....\$14.95
- Chicago '94: The Game** by Allen Huffman  
Experience the 3rd Annual "Last" Chicago CoCoFest with this graphics adventure. Over 30 digitized locations from the 'Fest to explore. Req: CoCo 3.....\$ 9.95
- MultiBoot** by Terry Todd & Allen Huffman  
Have up to 16 OS9BOOT files on one disk. Select with scrolling menu. Booster/Nitros compatible.  
Req: CoCo 3 .....\$19.95
- Towel!** by Allen Huffman  
Point 'n click disk utility. Use mouse or hot keys for all common commands. Configure all colors/commands. Includes EthaWin interface.  
Req: CoCo 3, Mouse Optional.....\$24.95
- 1992 CoCoFest Simulator** by Allen Huffman  
Dozens of digitized images from the '92 Atlanta 'Fest. Explore the vendor area and hotel locations.  
Req: CoCo 3, 500K+ Disk Space.....\$ 9.95
- MAC to DMP Print Utility** by Carl England  
Print classic Macintosh images to Tandy printers.  
Req: Level 1 or 2, DMP Printer.....\$19.95
- CheckBook+09** by Joel Mathew Hegberg  
Point 'n click account manager with graphing. Save pic, bar, or line graphs out in VEF format!  
Req: CoCo 3, Mouse Optional.....\$24.95
- MiniBanners09** by Allen Huffman  
Single or multi-line banners on ANY printer. Dozens of fonts included. A classic!  
Req: CoCo 3, ANY Printer.....\$19.95



### Now Offering StrongWare Products!

- Soviet Bloc** - The best CoCo 3 Tetris(tm)-like game ever. Supports stereo! RS-DOS  
Req: CoCo 3, Orch-90/Joystick Optional.....\$19.95
- GEMS** - A falling "columns" puzzler. Supports stereo! RS-DOS Req: CoCo 3, Orch-90/Joystick Optional....\$24.95
- Copy Cat** - Simon says match the flashing colors and sounds (Also available for MM/1, \$14.95)  
RS-DOS Req: CoCo 3.....\$9.95
- HFE Hprint Font Editor** - Powerful machine language hprint/MiniBanners font editor. (Includes fonts!)  
RS-DOS Req: CoCo 3, Joystick Optional.....\$19.95

...with more to come!

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

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

### Color Computer Disk Basic Software

- RS-DOS OS-9 Terminal Emulator** by Terry Todd  
Use a CoCo 3 as a "smart" terminal. Emulates screen/color codes, overlay windows, attributes and more via the bitbanger port at 2400 baud. Run many full-screen apps over the modem.  
Req: CoCo 3, Serial Cable.....\$24.95
- InfoPatch** by Terry Todd  
Patch classic Infocom(tm) text games to run on 80 columns, upper/lowercase, 6ms disk on CoCo 3.  
Req: CoCo 3, Infocom Disk.....\$ 9.95
- Super Boot** by Carl England  
The BEST! Type DOS to set tracks, sides, step rate, printer baud, colors, speed and more. Auto runs program or gives menu, too.  
Req: 64K+ CoCo, DOS 1.1/2.1....\$14.95
- 512K Copy Utilities** by Carl England  
Copy, rename, kill and format using full memory.  
Req: 512K CoCo 3.....\$14.95
- Super Backup** by Carl England  
Use from 64K to 512K for easy backups.  
Req: 64K+ CoCo.....\$14.95
- 4-D Checkers** by Nick Johnson  
Classic twist. Even talks with speech sound pak!  
Req: CoCo 3, Speech Pak Opt.....\$14.95
- CheckBook+** by Joel Mathew Hegberg  
The "CoCoMax" or checking programs! Fully graphics and mouse driven.  
Req: CoCo 3, Hi-Res Int/Mouse.....\$24.95
- MiniBanners** by Allen Huffman  
The original multi-line banner printer. A classic!  
Req: CoCo 3, ANY Printer.....\$19.95

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

### WPSshell

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

Summertime is "off" season for a lot of CoCoists. If you are one of those, look forward to new releases and upgrades this fall. If you use your CoCo all year 'round, the following titles are currently available:

CoCoTop version 1.0	\$24.95
CoCoTop version 1.1	\$19.95
CoCoTop 1.1 + Tools 3	\$34.95
OScopy/RScopy	\$10.00
TOOLS 3 version 1.1	\$29.95
Quickletter version 2.0	\$19.95
Accounting level 2	\$34.95
Investing level 2	\$24.95
Level II graphics 1.2	\$34.95

upgrades only \$5.00 (return original disk)

Shipping+handling: US/Canada \$3.00 all others \$5. Prices in US dollars Send cheque or money order NO COD'S. Call or write for Canadian dollar prices. Mention the name of this magazine in your order and you will receive a free bonus disk!

**C. Dekker** ...  
RR #4 Centreville, NB  
E0J 1H0, CANADA  
Phone 506-276-4841

User-friendly Level II  
Programs!



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 (DATES). 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

## Mid Iowa and Country CoCo Club

(non-profit)

If you want support, we're here for you! MI&CCC publishes the UPGRADE disk magazine, now in its tenth year, five as a national publication. We've grown to be one of the largest CoCo outreaches in the world! We have subscribers in over 40 states and five provinces of Canada, as well as in Australia and England.

Your MI&CCC membership brings you:

1. A year's subscription to the UPGRADE disk magazine (requires CoCo 3 and one disk drive), 8-10 issues per year. This is a news magazine, not a software disk.
2. Access to our shareware/public domain/orphanware library - we keep only the best!
3. Optional Christian sub-chapter that gathers Christian oriented software for those interested.
4. ROM burning and other support.

Say you saw this ad in "68 micros" and receive a bonus disk along with your new membership!

## Mid Iowa & Country CoCo Club

Terry Simons, Treas./Editor  
1328 48th, DesMoines, IA 50311

Please include your phone number and system information

# 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	\$25.00
Send manual or RomPak to prove ownership	
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!'	\$25.00
256/512 byte sectors, multipak support	

**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
OK	\$39.95
Sound Digitizing cable	\$15.00

**OS-9/68000 Software**

OSTerm 68K v2.2 External transfer protocol support	\$50.00
TTY/ANSI/VT100/K-Windows/Binary Emulation	
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](mailto:cmckay@northx.isis.org)

Don't have a subscription to "microdisk"? Don't want to pay the high price for back issues? You can now get the complete Volume 1 of microdisk for \$30 (plus \$2.50 S&H)! That's an \$18 savings over back issues and a \$10 savings over the subscription price! Just write and tell us you want the entire volume 1.

All files will be on as few disks as possible, not separate disks for each issue. "microdisk" is not a stand-alone product, but a companion to this magazine. Subscriptions are \$40 per year. Single issues are \$6 each in US, \$45/\$6.50 in Canada. Overseas add \$10 per year, \$1 each for airmail.

## ADVERTISER'S INDEX:

<i>BlackHawk Enterprises</i>	12
<i>C. Dekker</i>	26
<i>Chicago CoCoFest</i>	5
<i>Color Systems</i>	26
<i>CoNect</i>	25
<i>Delmar Company</i>	BC
<i>DISTO</i>	25
<i>FARNA Systems</i>	19,24, 27
<i>HawkSoft</i>	19
<i>J&amp;M Microtek</i>	19
<i>Mid Iowa CoCo Club</i>	26
<i>Northern Xposure</i>	27
<i>OS-9 User's Group</i>	27
<i>Peripheral Technology</i>	26
<i>Robert Gault</i>	26
<i>Small GrafX Etc.</i>	19
<i>Sub-Etha Software</i>	25

**Don't have a subscription yet?  
WHAT ARE YOU WAITING FOR?!  
Subscribe today!**

(Details inside front cover)

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 15 MHz.

## G-WINDOWS - a PROVEN WINNER

FOR OS-9

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

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.

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