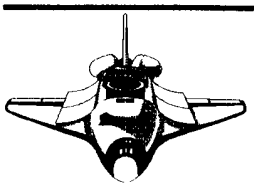
SPECIAL PNW COCOFEST EDITION

# OS-9 Newsletter©

Volume IV Issue 5 | Bellingham OS-9 Users Group | May 31. 1993



## The Rocket

Color Computer Replacement CPU

* Blazing fast 14 MHz 68EC000 Processor
* Plugs into Color Computer CPU socket
* Boot ROM with OS-9 68000 Kernel and 6309 Simulator
* 16-bit Wide Memory (512K/2M/8M)
* RBF (Disk), SCF(Serial), View(Window), and Interprocess (Pipe) Managers
* Utility programs (shell, assembler/linker, editor, file/task/ resource maintenance
* Uses existing Color Computer Peripherals
* Uses Color Computer memory for 6x09 emulation, RAM disk and graphics

**PRICE:**

(0K kit with software) $195 (2Meg kit with software) $295

**DESCRIPTION:**

***"The Rocket"*** combines the latest in microprocessor technology with a unique bus interface circuit developed by Burke & Burke, to actually replace the Color Computer's 8/16 bit 6809E CPU with a 16/32 bit 68EC000 CPU. In addition to being faster, the 68EC000 can directly run larger programs and manipulate more data than the 6809E. ***"The Rocket"*** gives you the benefits of OS-9/68000, as well as compatibility with Color Computer Software.

**AVAILABILITY:**

Recent advances by Motorola and software developers have given Burke & Burke the technology to build ***"The Rocket"*** and offer it at an amazingly low price. But there's a catch - There's no sense in building the ***"The Rocket"*** if nobody wants one. However, Burke & Burke is prepared to build and deliver ***"The Rocket"*** to it's customers within 90 days of receiving commitments totaling 100 or more units.

Burke & Burke is currently accepting advance orders for ***"The Rocket"***. And as always, they will not cash your check or bill your credit card until your order is shipped to you. But remember, if Burke & Burke doesn't receive orders totaling 100 or more units by June 30, 1993, they will cancel your order and return your uncashed check to you.

**SEEING IS BELIEVING:**

Chris Burke will demonstrate his prototype of ***"The Rocket"*** at the PNW CoCoFEST III in Port Orchard, Washington, June 25-26,

## Aghhh !!!

## My disk crashed, now what do I do?

Fixing crashed disks or at least recovering data from crashed disks can be a time consuming ordeal which is why the best recommendation is to ALWAYS keep current backups or DON'T use the disk at all if it is your only copy. However, crashed disks do not have to mean lost data if you know how to go about recovering the data.

That's what this article is all about and comes as a result of the three crashed disks I've had this past week as well as Chris Spry's current predicament. I have co-written one other article on this topic, but that pertained more specifically to boot disk problems. (See the *OS9 Newsletter, Volume III Issue 11B*, November 30, 1992)

You'll be needing a few tools handy to follow along with this article and if your planning to just practice for the eventuality of a crashed disk then for God sakes use a backup of something.

A screen dump utility of some sort would also come in handy

## <<- In This Issue ->>

**The Rocket:** 14MHz 68K plug in for the CoCo-3 **Pg 1**
*Burke & Burke has a super deal for you*

**TUTORIAL: Repair "Crashed" disk sectors** **Pg 1**
Step by step, easy to follow instructions, by Dave Gantz

**C MODULES:** Subroutine pointers for C **Pg 9**
How to call subroutine modules from within C, by Jason Bucata

**Upgrade to 512K for about $16** **Pg 10**
Piggy back 16 - 41256 chips, by Marty Goodman

**Chicago Fest Report: The 2nd Annual Last CoCo FEST** **Pg 12**
Huge in depth report by *Sub Etha's* Huffman

**PNW CoCo FEST III:** Who, When, Where and How much **Pg 17**
Itinerary, Accomodations and *PNW CoCo FEST Photo Gallery*

**PDS Database Returns: Basic09 print-out procedures** **Pg 19**
A more sophisticated approach with better results, by Wes Payne

**Questions & Answers:** **Pg 22**
CGA to CM-8, BINEX/EXBIN, Development Pak, LZH Compression

**Club Activity Reports** **Pg 23**
*Bellingham - Seattle - Port Orchard*

for this exercise in avoiding futility. See the end of this article for suggested files and source(s).

----------**Tool List**----------

OS9 Level II Manual, the big thick one that comes with the OS9 Level II disks.

*dEd*-- Copyright 1987 by Doug DeMartinis -- preferably the edition below

Note: *This edition has a patch made to it so that it will recognize and identify the Bit Allocation Map or BAM for short. The BAM is also sometimes called DAM or Disk Allocation Map. Besides, its the one I use. You could also use* ***Qtip*** *or* ***Zapper,*** *but the display will differ.*

Header for: *dEd*

Module size: $17A2 #6050
Module CRC: $299A3F (Good)
Hdr parity: $8F
Exec. off: $0665 #1637
Data Size: $0316 #790
Edition: $05 #5
Ty/La At/Rv: $11 $82
Prog mod, 6809 obj, re-en, R/O

**Now to dive in there and rescue Data** <Grin>.....

The first thing we will cover is the breakdown of Logical Sector Number zero (LSN0)on any OS9 disk, as well as the invocation of *dEd*. All numbers with a preceding $ are in hexadecimal (base 16) and others will be in decimal (base 10).

To invoke *dEd* for this exercise type the following from any OS9 prompt on any 80x24 or 80x25 text screen or graphics screen with stdfonts merged:

OS9: **dEd /Dx@** (where x = the drive number with the disk to be worked on in it.)

Note: *The @ in the above command allows us to open any disk just as if it were a file by itself, thus allowing us to work with any and all data the disk contains with the exception of high density disks in most cases.*

Here is an example of my LSN $00. Offsets (Relative Addresses) are read as LSN+the row index number+the column index number. See the ** in the last row? This would be read as LSN or $00+the row index or $70+the column index or $04 or a total offset of $0074. If we were on sector 1 then it would be $0174. The definitions of the import bytes follow the excerpt. Also see page 5-2 in the technical reference section of the OS9 Level II manual.

```
LSN=$00  00
     0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    0 2 4 6 8 A C E
00:  00 0B D0 12 01 7A 00 01 00 00 03 00 00 FF D4 E3    ..P..z........ Tc
10:  07 00 12 00 00 00 00 00 00 00 5D 03 08 10 2D 52    ..........]...-R
20:  69 42 42 53 20 43 6F 6D 6D 61 6E 64 73 20 44 69    iBBS Commands Di
30:  73 EB 00 00 00 00 00 00 00 00 00 00 00 00 00 01    sk..............
40:  01 03 21 03 00 54 02 00 00 12 00 12 03 09 00 61    ..!..T.........a
50:  40 00 00 00 00 00 00 00 00 00 00 00 00 74 2D 00    @............t-.
60:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
70:  00 00 00 00 ** 00 00 00 00 00 00 00 00 00 00 00    ................
```

| Name | Relative Address | Size (Bytes) | Use or Function | Mine |
|---|---|---|---|---|
| DD.TOT | $00 | 3 | Number of sectors on disk. | $000BD0 |
| DD.TKS | $03 | 1 | Track size in sectors. | $12 |
| DD.MAP | $04 | 2 | Number of bytes in allocation map. | $017A |
| DD.BIT | $06 | 2 | Number of sectors per cluster | $0001 |
| DD.DIR | $08 | 3 | Starting Sector of Root Dir | $000003 |
| DD.OWN | $0B | 2 | Owners ID number (usually 0) | $0000 |
| DD.ATT | $0D | 1 | Disk attributes | $FF |
| DD.DSK | $0E | 2 | Disk identification (internal use) | $D4E3 |
| DD.FMT | $10 | 1 | Disk Format, bit mapped | $07 |
| DD.SPT | $11 | 2 | Number of sectors per track | $0012 |
| DD.RES | $13 | 2 | Reserved for future use | $00 |
| DD.BT | $15 | 3 | Starting sector of bootstrap file | $000000 |
| DD.BSZ | $18 | 2 | Size of bootstrap file | $0000 |
| DD.DAT | $1A | 5 | Date of creation (Y:M:D:H:M) | $5D0308102D |
| DD.NAM | $1F | 32 | Disk name, last char has MSB set | see above |
| DD.OPT | $3F | | Path descriptor options | |

Probably the most important byte to us here are the bytes at offsets $08, $09, and $0A which tell us where the root directory begins. Speaking of which, that is our next stop in the CoCo Zone....

```
LSN=$03  03
      0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    0 2 4 6 8 A C E
00:  BF 00 00 5D 04 1A 0D 0A 02 00 00 01 20 00 00 00    ?..]........ ...
10:  00 00 04 00 07 00 00 00 00 00 00 00 00 00 00 00    ................
20:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
30:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
40:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
50:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
60:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
70:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
80:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
90:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
A0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
B0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
C0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
D0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
E0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
```

Well here we are exactly where LSN $00 said the root directory starts, at LSN $000003. But where are the filenames, you ask? Well they start on the next sector.

This sector is called a File Descriptor sector or FD for short. Every file or directory on an OS9 disk has one of these. This is why you can't store a true 360K worth of files and user data on a DSDD 40 track drive for example.

Starting with offset $10 ($0310) is what is called a segment list. This segment list tells OS9 where a file or directory on disk is located and how many sectors that file or directory occupies. There are 48 of these segments available each being 5 bytes wide. For you programmers, think of it as a two dimensional array such as: DIM segment(48,5). What this means is that your file or directory can occupy space in 48 different locations on disk if it is badly fragmented. In this case mine only occupies one segment starting at LSN $0400 and is 7 sectors in size.

So guess where our trip through the OS9 disk takes us next? If you said sector 4 or offset $0400 your right!

Note: *For our purpose I skipped the explanation of the first 16 bytes and will continue on with what we need from this sector to start finding data.*

```
LSN=$04  04
      0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    0 2 4 6 8 A C E
00:  2E AE 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
10:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03    ................
20:  AE 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
30:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03    ................
40:  43 4D 44 D3 00 00 00 00 00 00 00 00 00 00 00 00    CMDS............
50:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0B    ................
60:  53 59 D3 00 00 00 00 00 00 00 00 00 00 00 00 00    SYS.............
70:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 07 0A    ................
80:  72 69 62 62 73 2E 63 66 E7 00 00 00 00 00 00 00    ribbs.cfg.......
90:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 07 2F    ...............0
A0:  72 69 62 62 73 67 EF 00 00 00 00 00 00 00 00 00    ribbsgo.........
B0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 AC    ................,
C0:  4D 45 4E 55 D3 00 00 00 00 00 00 00 00 00 00 00    MENUS...........
D0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 07 42    ...............B
E0:  4C 4F 47 D3 00 00 00 00 00 00 00 00 00 00 00 00    LOGS............
F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 09 A3    ...............#
```

Well here we are, finally. The root directory. (Again for our purposes I'm going to skip the first 32 bytes of this sector.) Each entry for each file or directory is composed of 32 bytes. 29 of them represent the file or directory name while the last 3 tell where to find those individual files or directories, where FD is located on the disk. Looking at this perhaps you can see the importance of having your directory names in ALL UPPERCASE and your file names in all lowercase.

In this example I have 4 directories (CMDS, SYS, MENUS, and LOGS) and two files (ribbs.cfg and ribbsgo). Lets start with a file, *ribbsgo* in this case:

***ribbsgo*** entry starts at offset $A0 ($04A0) and ends at $BF ($04BF). The first 29 bytes as I said are for the file name, the last character of which has its Most Significant Bit set to mark the end of the file name. The last 3 bytes tell us where to find the FD for *ribbsgo* which is $0002AC or $02AC since the Most Significant Byte is 0. So this is where we are off to next, sector $02AC.

```
LSN=$2AC  684
      0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    0 2 4 6 8 A C E
00:  0B 00 00 5D 04 19 0C 11 01 00 00 04 A5 5D 04 19    ...].........%]..
10:  00 02 AD 00 05 00 00 00 00 00 00 00 00 00 00 00    ..-.............
20:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
30:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
40:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
50:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
60:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
70:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
80:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
90:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
A0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
B0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
C0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
D0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
E0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
```

OK, here we are. This FD is very similar to the one we examined on our way to the root directory. It contains all the same information and takes on exactly the same format as the FD for the root directory except that this time we are talking about a file and not a directory.

It tells us that our file, *ribbsgo*, begins at sector $0002AD or $02AD and occupies 5 sectors. So that is where we will go next. For our purposes I will only include the first and last sectors in this text as examples. I forgot to mention that we have proceeded through pages 5-3, 5-4, and most of 5-5 of the technical reference section at this point.

```
LSN=$2AD  685
      0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    0 2 4 6 8 A C E
00:  6F 6E 65 72 72 20 67 6F 74 6F 20 66 61 74 61 6C    onerr goto fatal
10:  65 72 72 6F 72 0D 63 64 20 2F 64 64 0D 63 78 20    error.cd /dd.cx
20:  2F 64 64 2F 63 6D 64 73 0D 76 61 72 2E 30 3D 22    /dd/cmds.var.0="
30:  22 0D 64 69 73 70 6C 61 79 20 30 43 20 30 32 20    ".display 0C 02
40:  33 34 20 32 32 20 31 42 20 33 32 20 30 33 20 30    34 22 1B 32 03 0
50:  35 20 32 30 0D 65 63 68 6F 20 50 6C 65 61 73 65    5 20.echo Please
60:  20 49 6E 73 65 72 74 20 79 6F 75 72 20 4F 53 39     Insert your OS9
70:  20 42 6F 6F 74 20 44 69 73 6B 20 69 6E 20 2F 44     Boot Disk in /D
80:  30 2E 0D 64 69 73 70 6C 61 79 20 30 32 20 34 45    0..display 02 4E
90:  20 32 45 20 31 42 20 32 32 20 30 31 20 31 41 20     2E 1B 22 01 1A
A0:  31 30 20 31 39 20 30 33 20 30 30 20 30 31 20 30    10 19 03 00 01 0
B0:  31 20 30 32 20 32 36 20 32 32 0D 65 63 68 6F 20    1 02 26 22.echo
C0:  50 72 65 73 73 20 41 6E 79 20 4B 65 79 0D 76 61    Press Any Key.va
D0:  72 2E 30 0D 2A 6E 6F 6B 65 79 70 72 65 73 73 0D    r.0.*nokeypress.
E0:  69 66 20 25 30 3D 22 22 0D 67 6F 74 6F 20 6E 6F    if %0="".goto no
F0:  6B 65 79 70 72 65 73 73 0D 65 6E 64 69 66 0D 64    keypress.endif.d
```

Well we made it! The actual file data. There are no special codes or anything of that nature here to explain. Just the ASCII codes for the contents of the *ribbsgo* script file. With program modules it would be the hexadecimal representations of the commands and variables and such within the program. As I said there are 5 consecutive sectors (or 1 segment) that this file occupies but I will only include this and the last sector, because everything in between is technically the same.

```
LSN=$2B1  689
      0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    0 2 4 6 8 A C E
00:  6F 63 6D 64 73 0D 67 6F 74 6F 20 2B 65 78 69 74    ocmds.goto +exit
10:  0D 2A 66 61 74 61 6C 65 72 72 6F 72 0D 65 63 68    .*fatalerror.ech
20:  6F 20 45 72 72 6F 72 20 25 2A 20 69 6E 20 52 69    o Error %* in Ri
30:  42 42 53 47 6F 2E 20 20 46 69 78 20 61 6E 64 20    BBSGo.  Fix and
40:  74 72 79 20 61 67 61 69 6E 0D 67 6F 74 6F 20 2B    try again.goto +
50:  65 78 69 74 0D 2A 66 69 6E 69 73 68 75 70 0D 64    exit.*finishup.d
60:  69 73 70 6C 61 79 20 31 62 20 32 33 0D 72 69 62    isplay 1b 23.rib
70:  62 73 6D 61 69 6E 20 23 31 36 4B 20 3C 3E 3E 3E    bsmain #16K <>>>
80:  2F 77 37 26 0D 2A 65 78 69 74 0D 64 69 73 70 6C    /w7&.*exit.displ
90:  61 79 20 31 62 20 33 32 20 30 30 20 30 35 20 32    ay 1b 32 00 05 2
A0:  31 20 30 43 0D 00 00 00 00 00 00 00 00 00 00 00    1 0C............
B0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
C0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
D0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
E0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
```

OK, this is the last sector of the *ribbsgo* file. The important thing to mention here is that this file does not contain essential information throughout the entire sector. The file ends with a carriage return ($0D) at offset $A4 or $355 taking into account

for the sector we are on. See all the $00's following that carriage return? We don't need them. I'll explain how to get rid of them later, for now its enough for you to know that they're not needed. In some cases those extra bytes may contain $E5's which is the value that OS9 writes to each sector when you format a disk.

Now that you have a basic understanding of how an OS9 disk is put together to become an effective storage medium we can go on and discuss how we are going to go about recovering data.

```
   Up/Down Arrows  Read & display Next/Previous sector  <CR> Clean up the screen display
   *  Restart
   $  Fork a SHELL (Ctrl-BREAK to return)
 *  A  Append displayed sector to output file
 *  C  Close output file
   D  Diddle (adjust) file length
   E  Edit the displayed sector
   F  Find a byte or text string (BREAK aborts)
   H  Help screen (also use '?')
   L  Link to a module - List all modules
 *  N  Next occurrence of byte(s) or string (Find)
 *  O  Open a file for output (use with Append)
   P  Push current sector onto stack
   Q  Quit dEd - Exit to OS9
   R  Remove and display a sector from stack
 *  S  Skip to given sector (sector # in hex)
   U  Unlink from module
   V  Verify all modules in file
   W  Write the sector back to the disk
   X  eXpert mode toggle on/off
   Z  Zap (fill in) the sector displayed
```

What you see above is the built in help menu from *dEd*. The starred options are the ones most often used:

* **S Skip to given sector (sector # in hex)**
  This option will let us skip to the sector(s) that we have identified from the file descriptors (FD's) and will speed things up considerably.
* **O Open a file for output (use with Append)**
  Once we have found the first sector of the data we wish to recover we can use this option to open a path to another disk (or RAM disk) on which we will store the recovered data. Since we will have to do some editing on the recovered file a RAM disk is recommended.
* **A Append displayed sector to output file**
  Once we have opened the destination file for the data we are trying to recover this option will let us add the current sector to that new file. You use this until you either reach the end of that particular segment (Another FD will most likely be displayed at the end of a segment or file) or the end of the file.
* **C Close output file**
  Now that we have recovered the data or file we must close the file before doing anything else with it.
* **F Find a byte or text string (BREAK aborts)**
* **N Next occurrence of byte(s) or string (Find)**
  If you know specific words or byte sequences to look for within the data or file your trying to recover then these two are handy for locating those words or sequences.

Well, we've recovered a file or data. There is, if you recall, quite likely some extra unwanted bytes. What do we do to get rid of them? That's easy, again using *dEd* (and *ident* for program modules) we diddle with the file length. Now you won't be dealing with real sector numbers, just the relative sector offset from the beginning of the file. In this case it will read LSN $00 through $04 although we may not actually be on sectors 0-4.

At any rate you need to find the last relative sector of the file probably using the arrow keys to scroll through it. When you reach the last sector look at the LSN, the row index number, and the column index number and determine the offset for the last byte (the carriage return) and add 1. In this example that last byte will be at $04A4 then add 1 giving us $04A5.

**Hit D for Diddle with file length.** It will tell you the old length and ask for the new length. Type it in ($04A5 for this example) and press enter. You will see the extra bytes disappear in front of you.

**Now hit Q to quit and answer 'Y'** and you have just recovered your first file. Give yourself a pat on the back, get a cup of coffee and dig in cause your gonna be dancin on the keyboard for several hours to completely recover one DSDD disk.

**SUMMARY:**

It took me roughly 24 hours to recover all data from 3-3 1/2" 756K floppies (I have mine formatted for 84 tracks double side rather than the usual 80 tracks double side <Evil Grin>). For some disks your directories will get trashed and there is little one can do to recover the directories (that I know of) in which case you will have to sit there with the arrow keys in *dEd* identifying FD's and locating the 'lost' files. This is what took me so long, my directories got trashed.

This is, as I said, a time consuming method but I know of no program that will do it for you. If I ever get some of my other programming projects finished I intend to write something, but for now this method will have to do.

Good luck recovering that lost data!
==Dave Gantz==

# A Hardware Analog to Digital Converter

## Part II

***NOTE: In last month's issue, Richard Kottke explained the need for an external Analog to Digital Converter for the CoCo. The article contained theory of operation, circuit descriptions and schematic diagrams. In this second part, Richard provides the software drivers and descriptor for OS-9.***

My project was constructed on a Radio Shack perfboard, the one with the 68-pin card edge connector. The connector can be trimmed to 40 pins using a saw or (preferrably) a Dremel tool with a cutoff wheel. I used wire wrap for all of the chips, except for the decoupling capacitors which were soldered directly to the socket pins. I used a 26-pin "header" style connector to get the joystick signals onto the board and made up joystick Y cables to extract the signals. Since my whole system goes into a full-sized clone case, this all works rather nicely. Another option would be to fit the board inside the coco somewhere and tap the expansion connector by soldering a ribbon cable to the underside of the motherboard; this makes getting the joystick signals easier also since you can tap off the connectors directly. If desired, you could design, mask and etch a printed circuit board for this project; this will make it somewhat smaller and make it easier to mount in a case if desired.

Some general construction tips: bypass all of the chips with a .1 uF capacitor. This is best done by inserting the cap into the perfboard inside of the socket's footprint, then inserting the socket on top of the cap and soldering the cap leads to the power and ground leads of the socket. DO NOT use wire wrap wire for power distribution. The power and ground leads should be hooked up using #24 or heavier sold wire to ensure clean power to the chips. U4 can either be a 7405 or 7406 open collector inverter. If you are going to power this project directly from the coco's internal supply, I recommend replacing all of the LS family chips with their HCT family varients (ie 74HCT138, 74HCT374 ...) This will drastically cut the total power required by the project at only a modest cost increase. The CA3310 and 74HCT4052 are CMOS chips, so observe static precautions when handling them. What I usually do prior to inserting CMOS chips into a project is go to the kitchen sink, dry it carefully, touch both hands to the sink and insert the chip with the project resting in the bottom of the sink. This may be somewhat excessive but it's easy to do and can prevent hard-to track bugs in the project from electrostatic discharge. It is usually good practice to tie all unused inputs either high or low and leave unused outputs unconnected.

**PART SOURCES**

NOTE: *do not endorse these companies! I just offer them as possible sources of parts for your projects!*

**Digikey**
**701 Brooks Ave S.**
**P.O. Box 677**
**Thief River Falls, MN 56701**
**1-800-DIGI-KEY**
**no minumum**

**Jameco**
**1355 Shoreway Rd.**
**Belmont, CA 94002**
**1-800-831-4242**
**$30 minimum**

**JDR micro**

JDR micro
2233 Samaritan Dr.
San Jose, CA 95124
1-800-538-5000
no minimum

Only Digikey has the CA3310 (it costs $11.77 as of 4/15/93). The remainder of parts and supplies can be obtained from the above companies or others listed in the backs of many electronics magazines. Several parts (caps, wirewrap, sockets & wire) can be found at your local Radio Shack, for slightly higher prices.

PROGRAMMING NOTES:
The project's memory map looks like this:

| | Write | Read |
|---|---|---|
| $FF38 bit 0: | analog mux sel 0 | 3310 D2 |
| bit 1: | analog mux sel 1 | 3310 D3 |
| bit 2: | nc | 3310 D4 |
| bit 3: | nc | 3310 D5 |
| bit 4: | nc | 3310 D6 |
| bit 5: | nc | 3310 D7 |
| bit 6: | nc | 3310 D8 |
| bit 7: | 3310 START* | 3310 D9 |
| $FF39 bits 0-5: | nc | nc |
| bit 6: | nc | 3310 D0 |
| bit 7: | nc | 3310 D1 |

To get a value, use this algorithm:

1. Place the analog mux value (0-3), ORed with $80 in $FF38.
2. Wait 13 cycles. This allows the analog mux to settle and allows the converter to complete any pending conversions.
3. Strobe START by pulling bit 7 of $FF38 low & then high.
4. Wait 13 cycles.
5. Read the result in $FF38 and $FF39.

NEW *CC3IO* PATCH LISTING
Use the following assembled code with ***Ipatch*** by Robert Santy to modify the original CC3IO (CRC = $F737C2, edition #16). Following the *ipatch* modification, the new CRC value should be $3B4DED.

```
*******************************
*  J O Y S T I C K
*
* Entry: x -> calling stack
*        R$X = 0 for right
*              1 for left
*
* Exit:  R$X = x value (0-63)
*        R$Y = y value (0-63)
*        R$A = 0 no buttons
*              1 button #1
*              2 button #2
*              3 both buttons
*******************************

gJoy    clrb            B=0, CC=0
        pshs  y,x,cc    save stuff
        ldy   <D.CCMem  WTF?
        cmpu  <$20,y    ???
        beq   joy10     if CCMEM=DRVMEM then ok?
        clra            else clear out x, y a
        std   R$X,x
        std   R$Y,x
        sta   R$A,x
        bra   joy60     and return

joy10   pshs  cc
        ldb   #$80
        stb   $ff38     ensure a/d start=off
        orcc  #$10      mask IRQ
        lda   #$FF      2 pull keybd col hi
        sta   >$FF02    4
        lda   >$FF00    4 get rows values
        puls  cc        6 unmask irq
        coma            hi bits=joy buttons
        ldy   R$X,x     get right/left
        bne   joy20     bra if left
        anda  #$05      mask for right buttons
        bra   joy30
joy20   anda  #$0A      mask for left buttons
        lsra            pre shift
joy30   lsra            shift button 1 to CC
        bcc   joy40     bra if no button 1
        ora   #$01      else set bit 0
joy40   sta   R$A,x     save button status

        ldd   R$X,x     get left/right
        lslb            put in bit 1
        andb  #$02      clear other bits
        pshs  b         save
        bsr   cnv64     do conversion
        std   R$X,x     save x value
        puls  b
        orb   #1        setup for y conversion
        bsr   cnv64     do conversion
        pshs  d         save value
        ldd   #63       val = 63 - val
        subd  ,s++
        std   R$Y,x     save y value
joy60   puls  pc,y,x,cc done

cnv64   orb   #$80      set mux
        pshs  cc        save cc
        orcc  #$50      disable irq
        stb   $ff38     5 send to mux
        andb  #3        2 start
        lbrn  cnv64     5 wait for mux settle
        lbrn  cnv64     5
        stb   $ff38     5 A/D converter
        orb   #$80      2
        stb   $ff38     5 un start
        nop             2 wait for conversion
        nop             2
        ldb   $ff38     5 get msb of result
        puls  cc        6 restore irq
        lsrb            div by 4
        lsrb
        clra            clear a
        rts

*****************************************
*          M   O   U   S   E
*
*

gMouse  pshs  u,y,x
        ldx   <D.CCMem
        cmpu  <$20,x
        beq   mous12
        ldy   ,s
        ldb   #$20
mous10  clr   ,-s
        decb
        bne   mous10
        leax  ,s
        bsr   mpkmov
```

```
        leas  <$20,s
        puls  pc,u,y,x

mous12  lda   <$63,x
        bne   mous14
        lda   <$60,x
        bne   mous16
        pshs  u,y,x
        bsr   mous20
        puls  u,y,x
        lda   <$66,x
        anda  <$67,x
        beq   mous16
        lda   #$03
        lbsr  L072D
        clr   <$67,x
        bra   mous16
mous14  lda   <$3D,x
        ora   #$40
        sta   <$3D,x
mous16  lda   #$01
        lbsr  L072D
        leax  <$3C,x
        ldy   ,s
        bsr   mpkmov
        pshs  cc
        lda   $01,x
        anda  #$7F
        sta   $01,x
        puls  cc
        puls  pc,u,y,x

mpkmov  ldu   $04,y
        ldy   <D.Proc
        ldb   $06,y
        clra
        ldy   #$0020
        os9   F$Move
        rts


mous18  ldx   <D.CCMem      external call
mous20  leax  <$3C,x        x -> mouse pak
        ldb   #$80
        stb   $ff38             ensure a/d start=off
        clra
        ldb   <$17,x        pt.res (0=lo 1=hi)
        tfr   d,y
        ldb   #$03          1= do vert 2= do horiz 3=both
         lda   $01,x        0= auto, 1= right joy, 2=left
        pshs  y,x,b,a       save parameters on stack
        ldd   #$FFFF    space for x & y values on stack
        pshs  b,a
        pshs  b,a
        clrb                default mux mask (right)
        lda   $04,s         pt.actv (L, R, Auto)
        anda  #$02          left joy?
        beq   mous22        bra if not
        ldb   #2            mux mask for left

mous22  leay  >cnv640,pcr   y <- adr of hires routine
        pshs  b             save left/right status
        lda   $0A,s         a= pt.res
        bne   mous24        bra if hires
        leay  >cnv64,pcr    else y <- adr of lores routine

mous24  lda   $06,s         get enable byte
        anda  #$02          check for do horiz
        beq   mous26        bra if not
        jsr   ,y            else do horiz
        std   $01,s         save X

mous26  lda   $06,s         get enable byte
        puls  b             restore left/right
        anda  #$01          check for do vert
        beq   mous28        bra if not
        orb   #1            do vert
        jsr   ,y
        std   $02,s         save Y

mous28  ldx   $06,s         x -> mouse pak
        ldd   ,s            d = x val
        bmi   mous30
        tst   $09,s         pt.res
        bne   mous30
        lda   #$0A          mul by 10 if lores
        mul
        bra   mous31
mous30   bsr   mous42        use convert routine if hires
mous31  cmpd  <$18,x        same as last value?
        beq   mous32        bra if yes
        std   <$18,x        else update and
        lda   #$01
        sta   <$2B,x        set magic flag

mous32  ldd   $02,s         d = y val
        bmi   mous34
        bsr   mous38        convert to 0-191
mous34  cmpd  <$1A,x        same as last value?
        beq   mous36        bra if yes
        std   <$1A,x        else update and
        lda   #$01
        sta   <$2B,x        set magic flag
mous36  leas  $04,s         clean x val & y val off stack
        puls  pc,y,x,b,a    done

mous38  tst   $0B,s         test pt.res
        bne   mous40        bra if hires
        lda   #$03          else mul  Y val by 3
        mul
        rts

mous40  lslb                mult Y val by 48
        rola
        lslb
        rola
        lslb
        rola
        lslb
        rola
        pshs  d
        lslb
        rola
        addd  ,s++
        tfr   a,b           div by 256
        clra
        rts

mous42  pshs  d             conv 1024 -> 640
        lslb
        rola
        lslb
        rola
        addd  ,s
        lslb
        rola
        lslb
        rola
        lslb
        rola
        addd  ,s++
        lslb
        rola
        rolb
        rola
        rolb
        exg   a,b
        anda  #3
        subd  #4
        bpl   mous44
        ldd   #0
        rts
mous44  cmpd  #639
```

```
        bls   mous46
        ldd   #639
mous46  rts

cnv640  pshs  cc
        orb   #$80      set up mux
        orcc  #$50      mask IRQ
        stb   $ff38     4 but don't start a/d
        lbrn  cnv640    5
        lbrn  cnv640    5 wait for mux to settle
        andb  #3        2
        stb   $ff38     5 start conversion
        orb   #$80      2
        stb   $ff38     5 un start
        nop             2 wait for conversion
        nop             2
        lda   $ff39     5 get data
        ldb   $ff38     5
        puls  cc        6 unmask interrupts
        anda  #$c0      mask garbage
        lsla
        rolb            convert to 0-1023
        rola
        rolb
        rola
        rts             done
```

I wish you luck with this project. If you want to contact me, my address is:

Richard J. Kottke
331 South Kools St Apt #4
Appleton, WI 54914

==Delphi: RICHKOTTKE==

# C MODULES

C Modules C Modules C Modules C Modules C Modules C Modules

**by Jason Bucata**

I've come up with an idea for how we can implement subroutine modules for C programs. We need to add some special handler code to the main program to do the job of automatically swapping in and out modules, and we need to introduce the concept of an extended function pointer. Each of the subroutine modules that a program will be using will have a descriptor that will be in a linked list, in order of the last time the module was used. Each descriptor will contain: a pointer to the name of the subroutine module; the address of the start of the module ($0000 if the module isn't linked into the map); the link count of that module; and the pointer to the next descriptor in line.

Each of the modules that need to use a subroutine module will use a call to the handler code (thru SWI3) that maintains the linked list. Every time a part of code needs to get a subroutine module, it makes a call to the routine (call it SUB$AddModule) that gets the module ready to use. It creates a descriptor for it, and adds it to the linked list, and it F$NMLink's it (or, if that doesn't work, F$NMLoad's it). It doesn't link it into the map yet. It also returns the pointer to the descriptor it created.

Whenever a function wants to call a function in a subroutine module, it needs to use the pointer to the subroutine module descriptor and a word for the number of the subroutine function in the module to be called. The calling function will use a call to SUB$CallRoutine, and it will take the module pointer and JSR to the entry point with the subroutine number in a register, or on the stack.

If the module is already linked in when the SUB$CallRoutine call is made, then the module address will be valid, and it will use that address. But if it hasn't, the address will be $0000, and it needs to link it in with F$Link (it knows it's already in external memory because it F$NMLinked or F$NMLoaded it beforehand). Then the address will be stored in the descriptor, and will then be used. Each time a module is called, the descriptor for it will be moved to the head of the linked list, because it was the most recently used.

If the program needs a subroutine module that isn't linked in, and there isn't enough space in the map to link it in, the handler code needs to unlink one or more of the modules already in the map to make room for it. It uses the module at the end of the linked list, because since each time a module is used, its descriptor is moved to the front of the list, the least recently used one will wind up at the end of the list. It will zero the address field of the descriptor, and unlink the module from the map.

In the main module in the program, we will have a function that will initialize the linked list. After that's called, the program will call SUB$AddModule for each of the modules that it will need throughout its execution. Then, each time it needs to call a routine, it uses SUB$CallRoutine, and it looks up the descriptor and calls the module. In order to call any function this way, instead of using just a regular function pointer, the program would use what I call an extended function pointer--the pointer to the module's descriptor, and the subroutine number.

The method we use to automatically link and unlink subroutine modules has to be always available, to both the main module and other subroutine modules. That means that it can't use any static variables. The best (and almost only) way to do that is with a software interrupt. Probably most of the subroutine modules will be linked in and used by the main program. But some of the modules *could* be used by other subroutine modules. This works fine; the SWI3 will work from anywhere in any module, because it doesn't require each module to use a predetermined address in the data section as a pointer to the code (the SWI3 vector acts as this pointer).

We need to be careful with subroutine modules, to make sure that they're 100% position-independent. All

OS-9 modules are position-independent in that they can be linked in anywhere. But they always stay in one place--so they can use absolute addressing for pointers to parts inside the module. For example, it's perfectly legal to have a LEAX <label>,PCR inside a routine, and then STX <address>,Y. Since that module won't move during execution, the address stored in memory will be valid.

*But*, with this setup the subroutine module *can* move! It's possible for a subroutine to call a subroutine in another module, and to have it be linked in, and have the original module be linked out to make room for it. When the called subroutine returns to the calling routine, that module will be linked back in--possibly at a %totally different address% from where it was before! Thus any pointers like the one above would become invalid.

To handle this, whenever a subroutine module calls SUB$CallRoutine, the stack frame from the SWI3 will be moved up in memory two bytes. The old PC address (one of those pesky pointers that point into the subroutine module) will be converted into the *offset from the start of the module*, and the extra word will hold the pointer to the subroutine module descriptor. (Another kind of "extended function pointer", except it has an offset instead of a routine number.)

When the subroutine module returns, it will RTS back to the handler code (which JSR'd into it in the first place). The handler will then link in the old module (if necessary), calculate the return address, restore the stack, and RTI back.

(For some routines, the paramaters will be passed in registers. For those that need the paramaters on the stack, we'll need to use a pointer that points to beyond the stack frame and to the paramaters. Programs now need to account for the return address on the PC anyway; we just need to extend that by several more bytes.)

This way, any code can keep the extended function pointers of all the subroutines it uses, and can SUB$CallRoutine to them at any time with impunity.

The link count field of the descriptor isn't for the link count of the module itself. Rather it's for the number of times that subroutine module has been SUB$AddModule'd. When a module is to be initialized with SUB$AddModule, the handler first checks the linked list to see if it's already initialized. If it is, and a descriptor is found, its link count is incremented by one, and the pointer to it is returned.

When a program or function is all done with a subroutine module, it calls SUB$RemoveModule to get rid of it. The link count in the descriptor is decremented by one, and if it's zero, the handler gets rid of it. It's unlinked twice--from the map, and from external memory to undo the F$NMLink--and the descriptor is removed from the list, and its memory is returned to the free pool.

C programs that use these subroutine modules will need to have dummy functions to represent the actual ones it will use in the subroutine modules. Those functions will leave the paramaters given to it on the stack for the actual routine, load the number of the subroutine it wants to use, grab the pointer to the descriptor from static memory (this pointer will be stored in a global variable at the start of the program when SUB$AddModule is called, and returns the pointer), and use these two combined as the extended function pointer to pass to SUB$CallRoutine, which takes over from there. It will then need to take the return value from the routine and return with it.

Subroutine modules written in C will need a special cstart.r module to handle translating the subroutine number into the address to branch to, the special format for the sent paramaters, and the special module header.

Please give me your comments and ideas. What do you think?
=Jason Bucata;DELPHI:(JBUCATA)=

# Upgrade to 512K for about $16

As of this writing (February, 1993), the CoCo 3 has long been orphaned, and support for it is now nearly non-existant. As part of this situation, it has become impossible to buy 512K memory upgrade boards for the CoCo 3. Well, if you have a 128K CoCo 3 and want to upgrade it to 512K, it still is possible to do this without a 512K memory board. The good news about this approach is that all that is needed are 16 41256 DRAM chips and four .1 mfd to .47 mfd miniature capacitors, and four 18 pin DIP headers. The BAD news is that this approach is very tedious, and requires considerable skill and experience with fine soldering, and a lot of patience.

**Materials:**

16 - 41256 (256K x 1) DRAM chips These should preferrably be 150 or 120 ns (nanoseconds) access speed, although faster chips (100 ns or faster) will likely work just fine, too.

4 - 18 pin DIP headers, of a size that will plug into the four 18 pin memory sockets on the CoCo 3. A Machine Pin style 18 pin socket may work fine for this purpose. Machine Pin style 20 pin sockets can, of course, be hack-sawed into 18 pin sockets.

4 - .1 to .47 mfd (I recommend .33 or .47 mfd) capacitors, physically very small in size (the size of a match-head is best).

1 - short piece of wire.

**Approach:**

What you will do is make four QUADRUPLY STACKED sets of memory chips. That is, you will be making four, four-chip-high piggybacks on top of the four 18 pin headers. Most of the pins will go straight thru and be soldered to the pins of the

chip or header below them. However, a few pins will be rerouted.

Bend straight out (horizontally) pins 1, 2, and 14 of a 41256 DRAM chip. Bend pin 16 of that DRAM chip "forward" so it reaches just a bit outward in the direction of the length of the 41256 chip.

Line up the chip over the 18 pin header, so that pins 1 and 16 of the DRAM chip lie over pins 2 and 17 of the header. See to it that pin 16 of the 41256 is bent in such a way that it can make contact with pin 18 of the header.

Solder pins 3,4,5,6,7, and 9 of the DRAM chip to pins 4,5,6,7,8, and 9 of the header. These pins will be going STRAIGHT DOWN, unbent.

Solder pins 9,10,11,12,13, and pin 15 of the DRAM chip to pins 10,11,12,13,14, and pin 16 of the 18 pin header. These pins will be going straight down.

Solder pin 16 of the DRAM chip to pin 18 of the header. This requires that pin 16 be bent a bit "forward", as noted above. You might want to put a "dog leg" type bend in it to keep it from contacting pin 17 of the header.

Prepare three more 41256 chips in a similar fashion, EXCEPT don't bend forward or out pin 1 or 16. Leave pin 16 alone. Now stack those three chips on top of the chip and header you prepared, one on top of the other. Attach pin 1 of the chips to pin 1 of the lowest chip, and attach pin 16 of the chips to pin 16 of the chip below it. Note that all the pin 16's are routed to pin 18 of the header, and that all the pin 1's are routed to the pin 1 of the lowest chip, which in turn is NOT hooked to the header, but instead is bent out.

Now, using wire wrap or other small gauge wire, join on each chip pins 2 and 14. Run wires from the joined pins 2 and 14 of each individual chip to pins 2,3,15, and 17 of the 18 pin header. It does not matter what pair of joined pins goes to what pin of the header... just be sure that each of those four header pins is connected to a pair of joined pins 2 and 14 from one of the four DRAMs in the stack.

Now solder on top of the stack the little capacitor, hooking one side of the capacitor to pin 8 of the DRAM chip and the other side to pin 16 of the DRAM chip at the top of the stack.

Make up a total of four such stacks as above. Note that you will likely find it easiest to clip off the narrow part of the horizontally bent out pins, leaving just the fat part of them to solder to. This may in fact be necessary to keep the pins from hitting each other when the stacks are installed.

Now plug each of the four stacks into the four 18 pin DRAM sockets in the CoCo 3. Solder short wire jumpers between the joined pin 1's of each of the four stacks.

Solder a wire from the joined pin 1's of all 16 DRAMs to a plate thru solder pad just in FRONT of pin 11 of CN4 on the CoCo 3 mother board.

*Note: The plate thru pad in question is located very near CN4, between CN4 and CN2. CN 4 is a 12 pin socket used for part of the 512K memory board, and CN 2 is the socket for the keyboard of the CoCo 3. Be SURE to remove the keyboard of the CoCo 3 before doing this, else you risk destroying the ribbon cable of the keyboard when you try to solder to the CoCo 3 mother board.*

Your CoCo will look a bit weird, as if there are four "high rise towers" in the memory chip sockets. However, if you carefully follow these instructions, you SHOULD get out of this a 512K CoCo 3.

**Reference Information:**

Pin out of 41256 chip compared to pin out of 4464 chip socket in CoCo 3:

```
             41256   4464
             ------------
                       1  *OE   (not used by CoCo 3 Ignore
extra adr    A8  1     2  I/Oa  bi directional I/O
 Data In     Din 2     3  I/Ob  bi directional I/O
             *WE 3---4    *WE   write enable
            *RAS 4---5    *RAS  Row Address Strobe
             adr 5---6    adr   address line
             adr 6---7    adr   address line
             adr 7---8    adr   address line
             Vcc 8---9    Vcc   +5 volts

            adr  9--10    adr
            adr 10--11    adr
            adr 11--12    adr
            adr 12--13    adr
            adr 13--14    adr
data in      Din 14  15   I/Oc  bidirectional I/O
            *CAS 15--16   *CAS  Column Adress Strobe
 ground     gnd 16  17   I/Od  bidirectional I/O
                  \-18    gnd   ground
```

What is going on here is that the Din and Dout of each 41256 is connected to each other, and those two joined pins are then connected to one of the bidirectional I/O lines of the header. The one extra address line (pin 1 of all 16 chips) is hooked to the mother board so that it can connect to that address line of the GIME chip's memory management section. Finally, pin 1 of the 4464 chips is not actually used, for it is an enable line that is grounded all the time. Therefore, we can ignore the fact that the 41256 lacks such a line.

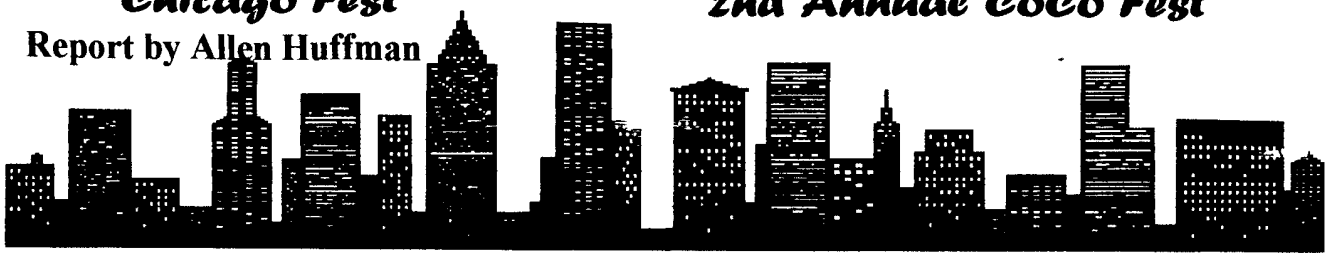
It is a great pity we could NOT use four 44256 chips to upgrade the CoCo 3 to 512K memory. This would have eliminated the hassel of staking chips four high. However, the refresh cycle for 44256 chips is 512 cycles, and the durn GIME chip in the CoCo supports only 256 cycle refresh. One would need special extra circuitry (like that on the Disto 2 meg upgrade) to use such chips.

I've never done this kind of an upgrade, but I once DID put 128K of memory in a CoCO 3 using four quadruply stacked 4164 chips, so I am quite confident that this approach will work. Please let me know if you try it and have success with it.

MARTYGOODMAN on Delphi
MARTYGOODMAN@DELPHI.COM on Internet

# Chicago Fest "2nd Annual CoCo Fest"

## Report by Allen Huffman



**VENDOR DISPLAYS:**

There were no half-booths at this show! **S-BUG, Disks 'n Dat, The International OS9 Consortium, and Sub-Etha Software** all held multiple-booth spaces. This was one big show, folks! The vendors were as follows:

**Al Dages** - There to show support for this event, Al (Atlanta Computer Society) was present with "previously owned" hard/software, rom-paks, power supplies, etc. Al furthers attempts to keep hardware and software from passing out of existence. Thanks Al! (Note: Al was also there to see if there was enough interest to justify ACS sponsoring another Atlanta CoCoFest this year.)

**Adventure Survivors** - These shows just aren't as warm without this couple. For those who don't know, a monthly adventure newsletter is available with hints, tips, walkthroughs, solutions, maps, etc, for many CoCo adventure games. They also sell several classic adventures and sold graphic adventures based on the past (and current - how'd they manage that?) CoCo Fests. Neat stuff, and fun folks!

**AniMajik Productions** - Not present, but a letter from Alan Sheltra explaining what has been happening with The OS9 Underground Magazine was there. The letter basically sighted some financial problems, then stated that OS9UG WILL be publishing again VERY soon and that it IS NOT DEAD. Three cheers for Alan. Please keep things runnin', man. We need the support. [*Note: I spoke with Alan the other day and OS9UG should be back on schedule anytime now!*]

**BARSoft** - Ya know...this guy could probably sell even more if he threw his weight around <grin>. (Er, uh...just kiddin' man!) Dave Barnes was there with various software titles and bits of hardware. Yet another vendor helping take up the slack. (Plus someone had to buy the drinks!)

**Burke & Burke** - Okay...here we go. Chris Burke was there. Apparently Mrs. Burke was at home watching a pet (?). Her presence was missed. Also, the presence of Chris' CoCo w/direct hookup hard drive was missed but he more than made up for it with goodies like: PowerBoost 2.0 which patches OS9 to run in NATIVE MODE on a 6309 chip. I picked up my copy and let me say that from initial checking it's a VERY NOTICEABLE speed improvement! Well done, Chris! Also, the Thexder-OS9 converter was there (moves Thexder to OS9 and runs just like the rompak! Amazing. Conversion done by Alan Dekok <aka, Mr. Ball Demo> of Canada). Enhanced 512-byte SCSI drivers were available (I use the B&B interface so I didn't pay too much attention to these) and lotsa classic Burke goodies like Repack, XT-ROM, RSB, etc, etc. [*Note: Spell checking with TSPellW under Booster is about DOUBLE what it used to be!*]

Of SPECIAL INTEREST was an amazing "new" (though designed long ago) product prototype called ***The Rocket***. The Rocket is a small board containing a 68000-type CPU and memory which plugs into the CoCo where the 6809 goes and turns the CoCo into a true OSK machine using the CoCo's hardware and I/O. The unit is set to sell for just under $200 and will be made if enough interest is shown. Folks, this 14mhz "update" might be just the way for us to get into OSK without spending $1000. Needless to say, the Burke booth was one of the most talked about at the entire show. (Heh...he sold out of PowerBoost software again and had to buy disks and photocopy manuals at the show for the second day...!)

**Chicago Area OS-9 Users Group** - Promising "big plans" in the future, this club was there to show their support. It's this type of attitue that helps us survive!

**ColorSystems** - Solitaire for the MM/1, a word-processor shell for the CoCo and games. And, best of all, BUMPER STICKERS reading "***I <Heart> my CoCo***", "***I <Heart> me MM/1***", "***OS-9 Users Group***", etc. NEAT STUFF! "***CoCo Classic***" and "***Friends Don't Let Friends Use MS-DOS***" buttons were also there at great prices. I hope they return to future shows so I can pick one of each up!

**CoNect** - Hmmm...Rick Uland's RS-232 pak replacement was nice. Dual port version? Sure...but what was this other thing they were talking about??? Do my eyes deceive me when I see a 232 pak capable of over 5000 CPS under OS9? Say hello to the next generation of 232 paks coming "soon" from CoNect which will make 14.4 v.42bis modems an "easy reality" for the CoCo. Set to sell for about $130, I believe. This received much talk and attention as well! According to the spec sheet 5300cps on this new pak is only slightly more CPU intensive than 900cps using SACIA and a normal 232 pak. Amazing!

**Cook County CoCo Club** - Fest T-Shirts on blue (again) designed by Nancy Myers. Nice and only $8 ($5 near close of show). CCCC took care of ticket sales, etc, and worked closely with Glenside during this event. Thanks guys!

**Dave & Nancy Myers** - Ahem. While "CoCoPro!" is no more, Dave and Nancy were there to show support, have fun, and sell down (and in some cases give away!) old bits of software and hardware they still had. I was touched to see them there. Dave (in my opinion) is very much responsible for non-Rainbow

supported Fests being possible. Thanks, man. It means alot to us. (Side note: I ended up with one of the LAST pieces of CoCoPro software...snif...) Dave and Nancy are going to be moving, but it is rumored we shall see them again at future Fests. I sure hope so.

**Delmar** - Remember Ed? He had a PC running OS-9000, and his System IV VGA OSK machines running G-Windows. His red foam rubber hammers were ALL over the 'Fest show. (At one point, the Kala Soft guys performed a nice "drum solo" by beating on the Yamaha keyboard at their booth with Ed's mighty hammers.) Delmar, being one of the strong OSK hardware vendors, seemed to get alot of attention. Ed mentioned that the user-base for G-Windows was around 1000 or so, which might make it a good platform to start developing on... (If we can get a system to Joel, he has expressed interest in porting some of our K-Windows products to G-Windows.)

**Disk 'n Dat** - A local vendor was there with EVERYTHING. Labels, disks, cases, ribbons, cables, etc, etc, etc. This was the NEATEST collection of goodies. They even had transfer paper for ironing on printouts to shirts. It was nice to see non-CoCo specific vendors interested in participating.

**Disto** - Tony Distefano was there with various working and non-working left overs from the days of CRC/Disto products. Let's see...what else? Hmmm. Oh yeah, there were 2 meg upgrade boards there. No big deal. Two small circuit boards. One plugs in on top or in place of the 6809, then the 6809 goes on it. Connect two wires and then plug the second board with the 2 meg SIMMS on it into where the 512K board goes, jumper across a resistor, install some software patches and 2 MEGS is yours. How can this man act so casual about something this NEAT???

Selling for less than $100 for the 0K board, Tony sold out the first day. He only brought 15 (he had at least double that back home) and he seemed to be kicking himself the entire show, amazed at response to his "no big deal" device. While "officially" Tony is out of the CoCo market, he apparently knew how to do this and decided to give our market another chance to make it worth his while. I hope it has been!

Also, his "Full Turn of the Screw" books were there including many of the hardware projects he designed over the years for the CoCo. It was very nice to FINALLY meet this legend in our community. (Hint: ORDER YOUR 2 MEG BOARD TODAY!!!)

**Farna Systems** - Officially there showing the Patch09 disk and OS-9 Quick Reference Guide, Frank Swygert sold copies of his newly complete "Tandy's Little Wonder" book which was PACKED with more complete information about the CoCo's history than I had ever seen compiled. It listed common hacks and upgrades, and Frank even got permission to include several schematics in the book! (He claims that feat was done simply by sending in a certified letter of request to the head-guy at Tandy...I was impressed!) Also, what is probably the world's only Tic-Tac-Toe game that requires 512K was shown.

From Australia, it was an AMAZING looking game with FULL DIGITAL SOUND and SPEECH with great graphics. "Crossroads II" was, I think, $12. The computer didn't play that well, but it was nice to look at!

Of special importance is Frank's efforts to publish "The World of 68 Micros", a new publication designed to take up where Rainbow left off. Columnists like Marty Goodman, Joel Hegberg, and various others will be on-staff. Let's support this. It looks good, and Frank (being in the military) is pretty much locked into doing it...heh heh...). THREE CHEERS for Frank... even if he does have a silly Southern accent! <grin...not like I'm from Texas or anything...> (Frank...it's just a joke, okay? ;) Look for my series of articles on "CoCo Supporters and their Accents" in an upcoming issue...

**Frank Hogg Laboratories** - Frank didn't make it, but he sent a rep down with his Kix/20 (or was it a 30?) OSK "tower". It looked neat, but wasn't doing much other than running a terminal. The specs of this device sound amazing. Imagine a graphics card capable of displaying an 8-bit color image in 1/40ths of a second. FULL SPEED. Faster than what current CPUs are capable so it's graphics wo't be a bottleneck like VGA cards are on the PCs. Of special interest was the Kix's capability to hook multiple graphics cards into it. Each one could have it's own monitor, mouse, and keyboard so you could run four dedicated graphics terminals off of one box. Hmmm...now THAT sounds interesting! Good luck, Frank. I hope you make it to Atlanta so we can see this amazing graphics card up close!

**Glenside CoCo Club of Illinois** - Mugs, buttons, and newsletters. This group made it all happen. They had lots of neat stuff...and their still-evolving OS9 based point of sale system (CoCo with terminals, of course) seemed to keep track of all their activity. Thanks, gang, for inviting us all back! (By the way...how many more of those CoCoFest mugs do you have?)

Of special interest on Sunday (?) was Tony Podraza's PA announcement about Bob Rosen's original old CoCo 1. They had it. It was an old grey case CoCo 1 with a Model 3 keyboard (with numeric keypad) and a series of switches that controlled power, reset, and the cartridge interrupt (for inserting rom-paks with the power on to make them not auto-start...). Inside was a 300 baud modem (Modem I it looked like) with the on/off and ans/org switches coming through. Also, a series of color LEDs indicated volume level of the cassette port. It was really neat! Built-in lowercase (with a switch) too. Tony talked about a possible auction and I suggested that Lee Veal outta buy it to go with his other prize CoCo. Seeing this machine "in person" made me realize just how big and great our community once was, and made me marvel at how well we have hung together over the years.

**HawkSoft** - Chris Hawks, designer of the SlotPak and various other gadgets, was there with his robe full o' name badges. (I didn't even try to compete with him this year!) A new version of IconBasic for OSK was there (program with pictures, basically - neat concept!) and a new SOUND editor for the MM/1.

Keyboard extension cables, dual hi/low-res joystick adapters for the CoCo, and so much more. He's a fun guy, and I think there was pizza at his house after the Fest on Sunday evening... Rumors of the "next generation" Icon Basic program with a context-sensitive point-n-click interface were circulated.

**Interactive Media Systems** - Paul Ward was nowhere to be found, but a rep was there running MM/1 demos and handing out updates on IMS. IMS is not dead, they claim, and several new configurations of the MM/1 will be available soon. Good news to those still on the waiting list, and those like myself who want an MM/1 more badly now than ever.

One interesting demo was when Brian White from Canada showed off a networked game called Assassin. It put the player in a 3-D "real world" view of a dungeon as you walked around and battled other creatures. FULL STEREO SOUND! You would hear footsteps off in the distance, etc. The best thing was you could like multiple MM/1s (and even Amigas, and soon other computer platforms as well) together to play! 16 MM/1s could be linked all battling each other, or over modems or whatever! TRULY amazing and one of the most interesting things I have seen. I hope we hear more about this soon. (It is a pity we didn't get around to linking up all the MM/1s at the show, as was discussed...grin!)

**Kala Software** - MIDI MIDI MIDI! An upgrade to UltiMuse for CoCo OS9 was there, and LOTSA music files for both the CoCo and MM/1 version of this notation sequencer. They played tons of tunes through their keyboard setup and kept us entertained the entire show with Led Zeppelin, Beatles, and
various other classics. BUY THIS if you love Midi! (Unfortunately, they STILL don't have Bohemian Rhapsody or any Monkees tunes done...blah!)

**Dirt Cheap Computer Stuff Company** - (?) Hmmm...some guy named Mark was there casually showing off various MM/1 goodies. If I owned an MM/1 I would have paid more attention. (Hmmm...was that a totally automated offline reader for Delphi or CompuServe that ran on the MM/1 or CoCo??? HEY! If you are on Delphi or CIS, GET THIS NOW and save time and money. It is GREAT and will pay for itself and reduce connect hours! If Delphi is long distance, this program will help save your phone bill by allowing you to read and reply to mail and forum messages OFFLINE then upload replies automatically later... gotta have it! Now if I can just talk him into doing a version for GEnie...) That was neat.

**OS-9 Users Group, USA** - Jim Destefano and others were on-hand with membership applications, sample MOTD newsletters, and library disks. I strongly urge any OS9 user to sign up. Some major things are in the works and we all need to help out.

**Overseas OS-9 User Group Consortium** - This was one of the most major events of ANY CoCoFest. Representatives from around the world were on-hand to talk about how they use OS9 and discuss how we may all benefit by getting together. Represented were:

- o Australian National OS-9 User Group with Gordon Bentzen
- o EFFO with Stephen Pashedag from Switzerland
- o EUROS-9 with Peter Tutelaers (he got the ball rolling here.
- o OS-9 User Group of Japan with Chikara Ymaguchi. He had a neat dual-6309 based machine running OS9.

VERY interesting and the dual processors gave this "old" machine much speed. Upgraded to 6309s much like we upgrade our CoCos.

The book, "*The OS-9 Guru*", was available. This is one thing you don't find sitting on bookshelves in the US. This was the highlight of attention.

**Microware** - Officially there selling OS-9000 not for $995 but for the amazing show price of $350 to OS9 User Group members. OS-9000 allows a 386/486 to run OSK, basially. The MASSIVE package was very complete and they sold perhaps 20-30 units. Also on display were CD-I machines running full motion 24-bit video (VEY impressive) all running OSK <grin>. Of special interest was this tiny portable CD-I machine with small LCD color screen and controller. I have never seen anything like it. It was great. Seeing this really makes me believe we need to help support CD-I more and get OSK into the mainstream. CD-I players have potential to go where even PCs aren't... any comments?

**Radical Electronics** - From "nowhere", this first-time vendor had something truly amazing. A CoCo-3 Schematic Designer. It fully supported laser printers and postscript and the output was 100% PROFESSIONAL. Finding something like this at the Fest was amazing. Many other add-ons and upgrades are planned. If you build circuits, you gotta check this out! I would never have guessed our "lowly CoCos" could do something on this scale. (Well, okay, maybe I DID think they could. . .)

**S-BUG** - "CoCoPro West"? Perhaps. Andre Lavelle was here again driving those miles to sell lots of new/used items from disk drives to printers to CoCo goodies. The Bob Van der Poel software products were also on sale, and he had a nice printer with serial and parallel (CoCo & PC hookup) that oddly didn't get sold even at it's great show price.

**StG Net** - Scott Griepentrog sat around working on the StG V4 BBS system (which is nearing completion and has some stunning options) and also sold some neat micro-PC keyboards for $50 or so. Very slick. Scott also has a program called DFIX that will "seek out and restore" clobbered disk files, directories, etc. This product, currently available for OSK, will soon be out for CoCo OS9 as well. Price will be around $30 for the CoCo version, and a bit higher for the OSK version. Watch for more details. This is a serious MUST HAVE for any hard drive owner! (Price subject to change.)

**StrongWare** - John Strong and his Team OS9 crew was there will Soviet Bloc, Gems, Font Editor, and lots of OSK products. Of interest were some (John, can I tell people about this?) upcoming MM/1 items including some games (neat titles screens) and various routines which will end up being used in

some PROFESSIONAL OSK stuff. Wow! By the way...John had the only Stereo CoCo games at the Fest, and his MM/1 "file requester" was great.

**Sub-Etha Software** - 1000 miles from home, Sub-Etha was back with "the same old stuff"...with one exception. Joel Hegberg's praised MM/1 word processor, Write-Right, turned heads with it's what-you-see-is-what-you-get screen supporting all printer options and colors. A true REAL word processor for the MM/1 with on-screen ruler and lotsa goodies to click on. [Demo now available!]

Also at the Sub-Etha booth, Brian White showed off his OSK "Speed Disk" program (under construction) which looked JUST LIKE Norton's utility for the PC. I can hardly wait! The software is just getting better and better all the time! (Brian? Care to tell us what else you have in the works? And can we please help market it when finished? ;)

Of odd interest was the announcement of the Sub-Etha sponsored programming contest in which programmers are challenged to create PONG games...awards to be given in Atlanta this October based on Memory Efficiency, Speed, Originality, Special Effects, and Playability for RS-DOS, OS9, or OSK updates to this classic arcade game that started it all. Any takers? Watch for "press release" information on this contest, which claims to offer the winners a $1 prize! <g> (By the way, it was noted that Chris Burke might be able to write the most efficient pong game for the 6309 thanks to his knowledge of some undocumented 6309 commands...such as the PNG <pong> command which Chris described as "draw paddle and beep"...!)

Oh, just to get a fair plug in..."the same old stuff" includes MiniBanners, CheckBook+, Etha-GUI, and the recent InfoPatch which converts old CoCo 1/2 Infocom text adventures to run in 80 columns on a CoCo 3. We also had Carl England's disk utlities and some old N*Johnson software. We promise MORE NEW ITEMS for the next Fest!

Briefly shown was the demo RS-DOS player for the SES (Spasm Eyeball Software, Nick Johnson's new venture) EthaSampler. Some stereo samples were played via an Orchestra-90 pack. Hopefully the full package with hardware will be out (under $50???) "real soon now".

JWT's Optimize Utility Set was supposed to be available, but the package did not arrive in time - something that has been hitting CoCo vendors alot lately. (This kept Sundog from having Contras for sale at the Iowa Fest...sigh...) I'd write more, but you could probably just read old Fest reports to hear more about what we offer. . . (Oh, gotta plug it...as the CoCo SysOp on GEnie, I had stacks of GEnie flyers to hand out as well! To prove a point, we unrolled a 27-foot listing of GEnie access numbers. It's great for those who don't have Delphi access numbers locally. . .)

**Sundog Systems** - Unable to attend, they were represented well. Games like Contras (2 player action!) and Photon attracted quite a bit of attention. When it comes to CoCo games, it doesn't get much better than Sundog...even if Photon is a STUPID, FRUSTRATING, and IMPOSSIBLE game...to me, anyway. Everyone else seemed to get the hang of this truly unique puzzle game with great graphics and digital music. (Hint: Buy it if you like addiction.)

**T&D Subscription Software** - While in the process (apparently) of shutting things down, they made an apperance and sold disks and tapes of old CoCo software. It was great to see them there and I hope they did well enough to consider future Fests.

**Lee Veal** - Another displaced Texan. Lee, proud owner of CoCo #1, sold Dave Wordel's video tapes (Learning OS9 and Installing the 6309) in addition to the STUNNING "Planet Engine" product. Kinda like a planaterium for the CoCo. Rumors of an OSK port of this product were circulating.

---

This place was crowded. The unusual setup of the show made things quite exciting. The overal layout was my favorite of any Fest I have attended. Enthusiasm was VERY intense and overall people seemed stunned at all the amazing things that were being shown and sold at this show. It was detrimental to many vendors, though, since there were so many great pieces of hardware to be found that some of the non-earth-shattering items got overlooked. Still, I think we will all agree this was one of the BEST Fests in many many years.

## THE SEMINARS

***"Future Support of the CoCo"***

Frank Swygert and Allen Huffman spoke on the future of the CoCo and some of the ways it can be supported in the future. [*Frank and I took turns talking about things we believe in. I discussed the importance of staying in touch, getting online, and supporting vendors, and Frank was able to talk about his efforts to link us together with his new CoCo publication, "The World of 68 Micros". Thanks to all those who attended and seem to care. You guys make it worthwhile!*]

***"Telecommunications and Networks"***

Eddie Kuns and Paul Jerkatis spoke on the telecommunications and networks, their uses and various means of support that are available. Eddie talked about Delphi and terminal programs while Paul explained StG-Net and BBSing.

***"Hardware Hacking"***

Mark Griffith and Rick Uland spoke on hardware hacking and answered questions.

***"Programming & Graphics"***

John Strong and Joel Hegberg spoke on programming with a strong emphasis on Graphics and User Interfaces (GUI).

***"OS-9000"***

Boisy Pitre and Mike Burgess spoke on and answered questions about OS-9000. This is an operating system that is designed much like OS-9 but runs on higher level platforms such as the Intel '386 or higher machines and the Motorola 68020 or higher machines.

***"International OS9 Consortium Discussion"***

A corner of the Fest area was turned into an enclosed meeting room and tables and chairs were brought in. At 8pm or so things got started as a table with reps from all around the world sat to introduce themselves and discuss the

future of OS9.

This meeting was of staggering proportions and the amount accomplished was incredible. Basically, we got to learn about how OS9 is used around the world. Australian users are CoCo'ist more or less, while Japan users seem to use odd OS9/OSK machines we have never seen The Japan users group seems to be around 400 members. In Europe OS9 and OSK are in use in homes and businesses, but in Switzerland the 6809-OS9 is nowhere to be found.

We learned so much in the hours that this meeting encompassed and here is what I would call the summary. They are working on a way to let us all share information, such as distributing articles from our various newsletters to each group. (The Australian newsletter will be made available through the USA UG on disk or by article reprints. The Japan newsletter would require translation.) Software/ hardware information will be exchanged to allow us to write programs that will run on systems around the world. Any chance of graphics standards? Not yet, but MGR offers some chance of that. (MGR is a graphics system for OSK that was ported to the MM/1 in about 1.5 hours at the show!) Libraries will be shared and programming tips exchanged. The OS9 Users Group of USA will do it's best to fill us in on further developments amongst the groups.

***"Glenside Meeting"***

Tony Podraza and cast of ???? held a meeting (albeit a condensed version) of the Glenside CoCo Club complete with the official gavel. [it takes batteries and makes space noises...go figure]

***"Network Communications"***

Scott Griepentrog and Paul Jerkatis spoke on network communications and BBSes. If you have never heard of LAN or WAN then you must attend this seminar. [Scott talked about a revolutionary change in phone systems coming soon that allows channels of digital data to be send with voice calls all real-time. It could literally allow BBS packages like StG net to become more like internet offering real-time links between all systems...it could happen within a few years... Scott used a variety of overhead transparencies, too, which he designed on Friday night.]

***"Programming"***

Okay, okay...this one was a mistake. This seminar was actually Joel Hegberg and Myself making an attempt to show off a new user-interface for the CoCo that works on a text screen offering pull-down menus, point and click via a text cursor, pop-up boxes, etc, etc, all high-speed and EASY TO USE and IMPLEMENT with hot keys, etc. An MM/1 port will also be done and ANYONE who wishes to develope with it can get the library once complete FREE without royalty fees. We are trying to get some more professional software and make programmers jobs easier. If interested, get in touch with Sub-Etha.

***"User Groups and the Future"***

Peter Tutelears and Carl Boll spoke about future support for both OS-9 and the CoCo. Learn about user groups and the role they will play in the future by keeping users in contact with each other. [This is a *VERY* important issue. I hope someone will put together a report of this seminar for mass distribution.]

---

**SUMMARY:**

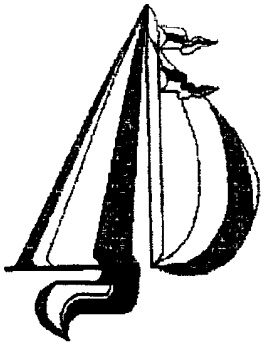
PHENOMENAL! Without a doubt the *BEST*, most-exciting CoCoFest I have ever gone to. There was so much to see that it was almost detrimental to vendors such as us with nothing totally new to offer. Sub-Etha only did good thanks to Joel Hegberg's new word processor for the MM/1. The rest of the money seemed to be spent on 2-meg upgrades, OS-9000, and other high-dollar items. I am sure that the overall dollar-per-ticket amount at this show was higher than many previous shows.

The show area itself was packed, though I am not sure how many actual tickets were sold. It sure seemed full and busy the entire time, but someone else would need to qualify this statement. ("Rumored" attendance was under 300.) I loved the layout of the booths. This was a very refreshing change, and probably necessary to fit everyone in.

Things seemed VERY stable in the CoCo Community. It really feels as if we have control back of our declining group. Surprisingly, with the Rainbow "gone" it seems like support is now actually greater than it was a year ago. New publications such as "'68 Micros", "UpTime", "No Name Magazine" (*don't forget the OS-9 NEWSLETTER)* along with the resurgence (cross fingers) of "OS-9 Underground" let us know there WILL be places to find information about upcoming CoCoFests. There WILL be places to find out about new products such as *The Rocket* and CoNect's *super 232 pak*. There WILL be support...as long as we support their efforts.

There really was more neat stuff here than you could "shake a stick at" (Terry made me say this). Bumper stickers and buttons and lotsa great things to take home. I don't think I've ever had quite this much fun at a Fest (well, maybe I have...this Fest was soooo busy that I'm sure I missed alot of great things). Did I mention the "breakout" type game for OS9 under works by an un-named Canadian that used a series of HIGH SPEED graphics routines? Did I talk about the point-and-shoot interface to play sound files? So much was being shown...I'd have to double this Fest report to put it all in. I hope someone else might write a "What Allen Left Out" report. (If you do, send me a copy!) I feel like I have left out more in this Fest report than I INCLUDED in many of my past reports...!

Allen C. Huffman, Owner - Sub-Etha Software
COCO-SYSOP on GEnie
COCO-SYSOP@GEnie.GEIS.COM on Internet
P.O. Box 152442, Lufkin, Texas 75915
(409) 560-9696
(815) 748-6638 (OSK Midwest Division)



Hosted by Port O'CoCo Computer Club of Port Orchard

# PNW COCO FEST III

## June 25-26, 1993
## Port Orchard, Wa.

**FRIDAY, June 25**

**6-7 p.m.** **REGISTRATION ($25) and Social Hour**

**7-7:45** **WELCOME**
**"WHERE ARE WE IN THE REAL WORLD?"**
Brief History of the CoCo, RS-Dos, OS9/OSK - Bob van der Poel

**8:45-9:15** **CONTROLLING 250 MOTORS WITH ONE COCO-II**
Creative Art application - Paul Surey
**MULTIMEDIA APPLICATIONS**
Controlling a Laser Disk Player - Rodger Alexander

**9:15-10** **MAKING CIRCUIT BOARDS FOR YOU COCO**
Surface mount process - Scott Honaker
**RECOVERING CRASHED OS-9 DISK WITH A DISK EDITOR**
Disk File Structure and Repair - Dave Gantz

**SATURDAY, June 26**

**7:30-8 a.m.** **SET UP FOR COMPUTER SALES AND FLEA MARKET**

**8-10** **REGISTRATION ($25)**
**BREAKFAST**
**OPENING OF COMPUTER SALES AND FLEA MARKET**

**10-10:45** **DISK EXTENDED COLOR BASIC ANIMATION - Chet Simpson**
**FHL's KIX\30 68K COMPUTER DEMONSTRATION - Allen Morgan**

**10:45-12** **BURKE & BURKE'S "THE ROCKET" - Chris Burke**
14MHz 68000 processor for the CoCo
**RECOVERING CRASHED DISK IN DISK EXTENDED BASIC**

**12-2 p.m.** **LUNCHEON - with a Presentation by Phillips CDI**
**Compact Disk - Interactive (Controlled by OS9/68K)**

**2-2:45** **COCO SUPPORT - Terry Laraway, Chris Johnson, Dennis Mott**
(Magazines, BBS's, Vendors, OCN)
**IMS's MM/1 68K COMPUTER DEMONSTRATION - Bernie Besherse**

**2:45-3:30** **TOWER CONSTRUCTION - Rodger Alexander & Gene Elliot**
**TELECOMMUNICATIONS - Sysops: Dennis Mott and Allen Morgan**
"How to get on a BBS"

3:30-4:15 **OS-9000 ON A 386/486 PC DEMONSTRATION - Mark Johnson**

4:15-5:00 **C PROGRAMMING CONCEPTS - Randy Kirschenmann**
Syntax, Constructs and Practices
**PACKETT RADIO (Ham Radio BBS'ing) - Scott Honaker**

PRE-REGISTRATION $20 or PRE-REGISTRATION + FEST MUG + T-SHIRT $30
REGISTRATION AT THE DOOR $25 FEST MUGS $6 T-SHIRTS $7

COMMERCIAL VENDOR DISPLAY SPACE $20 NON-COMMERCIAL DISPLAY SPACE $3
DISPLAY ATTENDANTS $10 (includes breakfast & lunch otherwise **no charge**)

**CONTACT:**
**DONALD ZIMMERMAN**
**3046 Banner Rd. SE**
**Port Orchard, WA 98366-8810**

**FEST LOCATIONS:**
**MARCUS WHITMAN JR. HIGH**
**1887 Madrona Dr. SE**
**Port Orchard, WA**

**ACCOMODATIONS:**

VISTA MOTEL
1090 Bethel Rd. SE
Port Orchard, WA 98366
1-206-876-8046

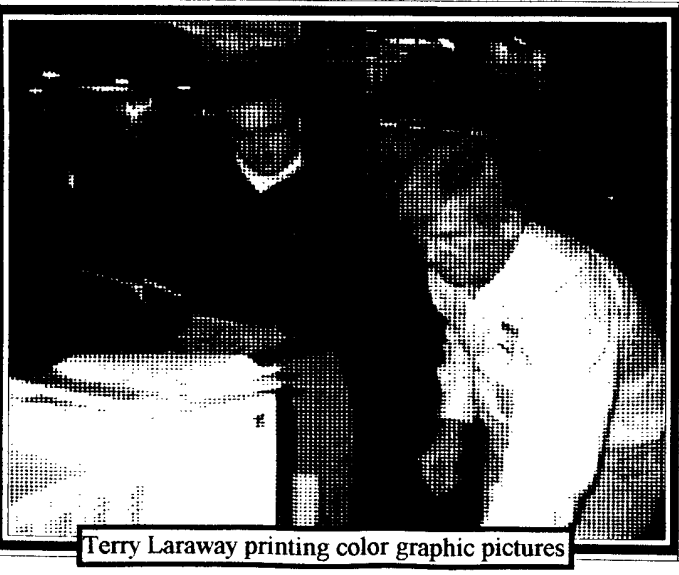
MANCHESTER STATE PARK
Located 3 miles from FEST
1-206-871-4065

NENDELS in Bremerton
Located 9 miles from FEST
1-800-547-0106

BEST WESTERN in Bremerton
Located 9 miles from FEST
1-800-528-1234

SUPER 8 MOTEL in Bremerton
Located 9 miles from FEST
1-800-800-8000

# PNW CoCo FEST Photo Gallery


Terry Laraway printing color graphic pictures


Scott Honaker posting a message via Packett Radio

# *PDS Database (revisited)* Part 10 ?

## Basic09 Tutorial: *Using Boolean variables and not Using "Inkeys"*

Remember way back when, like last Winter (September '91-June '92) when the Seattle 68xxxMicro Users Group was putting together a Basic09 Database in a 9 part series in this Newsletter? Well, **It's Back.** I guess we could call this "*Basic09 Database Tutorial, Part 10*".

The ***PDS Database*** as it was presented in the 9 part series was/is complete. What is presented here is a more sophisticated print-out procedure that you may choose to use to replace the ***Print_LB*** Procedure in the database. The original procedure works fine and is short and sweet, however this updated procedure provides options so that the user can have more control of the output. Such as: Do you want to print only the currently displayed record or all of the records. Instead of only single wide printouts, now you are prompted as to whether you want singe or double width printouts. Instead of accepting the default order of the name fields (First_Name, Last_Name), you are now prompted for "First Name First or Last Name First". Not really necessary but a nice little extra.

The listing below is well commented and I would suggest that all of the "**REM**" comment lines be deleted (except for the author and creation date). Wes Payne, who wrote this procedure originally learned BASIC on a Texas Instrument (TI-99) when he was 12 years old. He graduated to Basic09 while in High School working on a GIMIX OS-9 Level II GMX-3 multi-user system. Believe it or not, they used Dale Puckett's ***Baisc09 Tour Guide*** (in hard bound) as their text book. Wes' approach to programming in Basic09 is so beyond most Basic09 users, that I wanted to share his technique with you in this article.

```
PROCEDURE Print_LB
(* New Version by W. Payne  26May93
REM Defined data type
TYPE address=FName:STRING[10]; LName:STRING[15]; address1:STRING
[20]; address2:STRING[20]; city:STRING[15]; state:STRING
[2]; zip:STRING[10]; phone:STRING[14]
DIM rec(2):address; tmp:STRING; a,b,c,d,x:INTEGER; CurOnly,
Double,NameOrd:BOOLEAN
DIM PrtBlnk(2):BOOLEAN
DIM Menu:STRING[1]; NextLine(2):STRING[35]
REM Refer to the PDS Procedure file for parameter passing (Sept '91 OS-9 Newsletter)
PARAM DB_Path:BYTE; Top:INTEGER; Current:INTEGER; Out_Path:
BYTE
PRINT
PRINT "Print <C>urrent record or <A>ll records?  ";
REM Use the GET statement rather then using INKEYS
GET #0,Menu
REM Test for proper reply
IF NOT(Menu="c" OR Menu="=C" OR Menu="a" OR Menu="A") THEN
  END
ENDIF
REM Assumes entry is "c" or "C"
CurOnly=TRUE
IF Menu="a" OR Menu="A" THEN
  CurOnly=FALSE
ENDIF
PRINT
IF CurOnly=FALSE THEN
  PRINT "Print lables <S>ingle width or <D>ouble?  ";
  GET #0,Menu
  IF NOT(Menu="s" OR Menu="S" OR Menu="d" OR Menu="D") THEN
    END
  ENDIF
REM Assumes Menu="d" or "D"
  Double=TRUE
  IF Menu="s" OR Menu="S" THEN
    Double=FALSE
  ENDIF
  PRINT
ELSE
  Double=FALSE
ENDIF
PRINT "Press <1> to list names with first name first (ex:  John Doe) or"
PRINT "      <2> to list names with last name first (ex:  Doe, John)";
GET #0,Menu
REM Test for proper key entry
IF NOT(Menu="1" OR Menu="2") THEN
  END
```

```
ENDIF
REM Assumes Menu="1"
NameOrd=TRUE
IF Menu="2" THEN
  NameOrd=FALSE
ENDIF
REM Ring Bell on Printer
PRINT CHR$(7)
PRINT "Make sure that printer is properly aligned and then press any key...";
GET #0,Menu
PRINT
IF CurOnly=TRUE THEN
REM if Current record only is to be printed
  x=Current
  d=1
  GOSUB 10
ELSE
  IF Double=FALSE THEN
    d=1
  ELSE
    d=2
  ENDIF
  FOR a=1 TO Top STEP d
    x=a
    GOSUB 10
  NEXT a
ENDIF
END

10    (* Get and print records here *)
      REM record is already open from PDS Procedure
      SEEK #DB_Path,(x-1)*SIZE(rec(1)) \ REM Find beginning of current record
      GET #DB_Path,rec(1) \ REM Get current record
      IF Double=TRUE THEN \ REM If double width has been selected (else goto print routine)
        IF x+1<=Top THEN \ REM and the current record +1 is less or equal to the end
          GET #DB_Path,rec(2) \ REM Get the second record
        ELSE  \ REM IF x+1 is greater then the Top, field variables are assigned ""
          rec(2).FName=""
          rec(2).LName=""
          rec(2).address1=""
          rec(2).address2=""
          rec(2).city=""
          rec(2).state=""
          rec(2).zip=""
        ENDIF
      ENDIF
      PRINT #Out_Path \ REM Print blank line to printer (defined as /p)
      PrtBlnk(1)=FALSE \ REM Reset PrtBlnk(1) boolean variable
      PrtBlnk(2)=FALSE \ REM Reset PrtBlnk(2) boolean variable

      (* Print Name Line *)
      GOSUB 16 \ REM Construct Name Line
      PRINT #Out_Path USING "S35,S5",NextLine(1),"";
      REM PrintUsing statement "String35, (Name1), String5 ("");
      IF Double=TRUE AND x+1<=Top THEN
      REM If double width has been selected and the current record +1 does not exceed
      REM the last record, then printout next line
        PRINT #Out_Path,NextLine(2) \ REM Print Name2 in column 40 (35 + 5)
      ELSE \ REM if current +1 exceeds end of record (Top)
        PRINT #Out_Path \ REM Print blank line
      ENDIF

      (* Print Address Lines *)
      PRINT #Out_Path USING "S35,S5",rec(1).address1,""; \ REM Print 1st address line
      IF Double=TRUE AND x+1<=Top THEN \ REM If double width and not exceeding last record
        PRINT #Out_Path,rec(2).address1 \ REM Print 1st address line or 2nd record
      ELSE
        PRINT #Out_Path \ REM Print Blank line
      ENDIF

      IF rec(1).address2="" THEN \ REM If 1st record's 2nd address line is blank
        PrtBlnk(1)=TRUE
        c=1 \ REM column one
        GOSUB 18 \ REM Goto Construct City, State, Zip line routine
      ELSE \ REM If address 2 is not blank
        PrtBlnk(1)=FALSE
        NextLine(1)=rec(1).address2 \ REM NextLine(1) variable = record1's 2nd address
```

```
    ENDIF
    IF Double=TRUE THEN
      IF rec(2).address2="" THEN \ REM If record2's 2nd address line is blank
        PrtBlnk(2)=TRUE
        c=2 \ REM column two
        GOSUB 18 \ REM Goto Construct City, State, Zip line routine
      ELSE
        PrtBlnk(2)=FALSE
        NextLine(2)=rec(2).address2 \ REM NextLine(2) equals record2's 2nd address
      ENDIF
    ENDIF
    PRINT #Out_Path USING "S35,S5",NextLine(1),""; \ REM print 1st column, 2nd address
    IF Double=TRUE AND x+1<=Top THEN
      PRINT #Out_Path,NextLine(2) \ REM Print 2nd column, 2nd address line
    ELSE
      PRINT #Out_Path \ Print blank line
    ENDIF

    (* Print City, State, Zip line or blank if done already *)
    IF PrtBlnk(1)=FALSE THEN \ REM If 1st column (city,state,zip) exist
      c=1 \ REM column one
      GOSUB 18 \ REM Goto Construct City, State, Zip line routine
      PRINT #Out_Path USING "S35,S5",NextLine(1),""; \ REM Print Column 1, City,St,Zip
    ELSE
      PRINT #Out_Path USING "S40",""; \ REM Print blank space in column 2,
    ENDIF
    IF Double=TRUE AND x+1<=Top AND PrtBlnk(2)=FALSE THEN
    REM If double column and not at end of file and record 2 (city,state,zip) exist
      c=2 \ REM column two
      GOSUB 18 \ REM Goto Construct City, State, Zip line routine
      PRINT #Out_Path,NextLine(2); \ REM Print 2nd column City, State, Zip
    ELSE
      PRINT #Out_Path \ REM Print blank line
    ENDIF

    IF Double=TRUE AND (PrtBlnk(1)=FALSE OR PrtBlnk(2)=FALSE) THEN
    REM If double column and 1st or 2nd column (city,state,zip) exist
      PRINT #Out_Path \ REM Print blank line
    ENDIF
    PRINT #Out_Path \ REM Print blank line
    RETURN

16  (* Construct Name Line *)
    FOR b=1 TO d \ REM d = 1 or 2
      IF NameOrd=TRUE THEN \ REM First Name First, Last Name Last
        NextLine(b)=rec(b).FName+" "+rec(b).LName
      ELSE \ REM Last Name First, First Name Last
        NextLine(b)=rec(b).LName+", "+rec(b).FName
      ENDIF
    NEXT b \ REM do the second entry if double (d = 2)
    RETURN \ REM Return to Print Name Line routine

18  (* Construct City, State, Zip line *)
    NextLine(c)=rec(c).city+", "+rec(c).state+"  "+rec(c).zip
    REM NextLine(1 or 2) equals record(1 or 2) city, state, zip
    RETURN
```

The entire ***PDS Database*** listings are available in the back issues of the *OS-9 Newsletter* starting with Volume II, Issue No. 9 and concluding with Volume III, Issue No. 6. The database is complete with the original *Print_DB* Procedure. This installment of the database serves only to provide an alternative *Print_DB* Procedure with more features. Even if you don't use this procedure in your version of the *PDS Database*, you will find the above listing very educational and maybe even provide you with some new insights into programming with Basic09.

==Rodger Alexander==

*If you do not have all of the issues you may order back issues at 75 cents or receive the entire source code and packed RunB version of **PDS Database** for $1 or send a blank 5-1/4 or 3-1/2 inch disk with sufficient postage for return mail to:*

PDS Database
c/o OS-9 Newsletter
3404 Illinois Lane
Bellingham, WA 98226

# Q&A

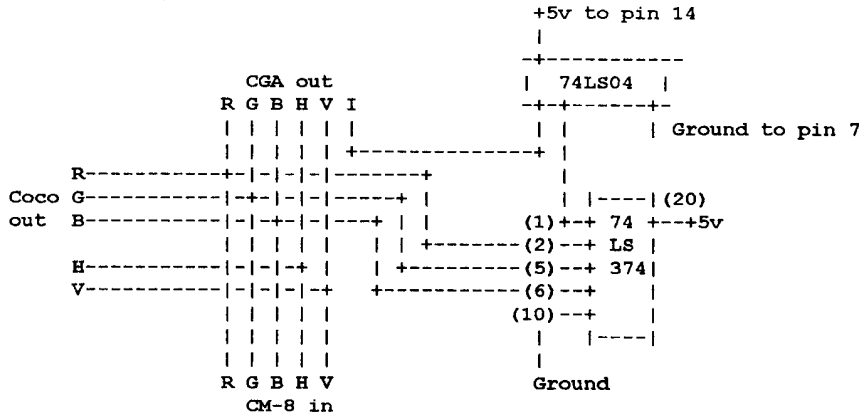
**Q:** I would like to connect my old CM-8 to an XT. I am having trouble doing this. It seems like the hsync and vsync, red, blue, & green are easy enough to wire up, but I notice there is no pin 6 (intensity) on the CM-8 monitor to connect to the XT. Is this my problem? Can it be done?

==Tom Colston==

**A:** The signals on these pins are not the same. The CGA uses ttl levels while the analog signals for the CM8 vary between 0 and 1 volt for color and 3 volts for the sync pulses.

All one needs to do is invert the Intensity signal using a 74LS04 then put it into a 74LS374 via pin 1 (OE). Tie the RGB sigs. to pins 2,5, and 6 and the CGA out to the CM-8. Supply both TTL chips with +5vdc. and Ground. This will let you run CGA on a CM-8. I have set it up to use both the CoCo and CGA output, but NOT AT THE SAME TIME. Use a small project box from Radio Shack and you are all set.

Here goes a rough draft of the schematics.




==Paul Voreland==

**Q:** I don't for sure how to go about getting *gcc/g++*. The file is bigger than the capacity of the floppy and I have no idea how one would split a big file across two or more floppies.

==Robert Hiller==

**A:** If you can get *gcc/g++* onto your OS-9 system, you have two ways to deal with your splitting problem. *Binex & Exbin* or *Uuencode & Uudecode.*

I'll use Binex and Exbin in these sample: ARChive it first before traansfering to your OS-9 system. Use "BINEX" to create an ascii file that can be split by any text editor into as many files as you would like, ie.. disk 1 of 9, disk 2 of 9, disk 3 of 9 etc.

The cmd line would be: **BINEX filename new_filename <ENTER>**

at the next promt: Enter starting address for file: **(type) 100 <ENTER>**

next: Enter name for header record: **(type) (real_filename) <ENTER>**

The receiver would use *Exbin* to uncode the file: First, Join all files together (NO SPARE LINES/SPACES). Next: type **EXBIN filename1 filename2 <ENTER>**

filename1 = real_filename that YOU gave the "header record" above.

filename2 = what ever!

The CRC will have to be checked after *Exbin* has been used. *Uuencode & Uudecode* are used much the same way.

==Gerry McCleary==

**Q:** I received a message referring to Page 6-71 of the OS9 manual for information on *MODPATCH,* however, some of us did not receive a *modpatch* section in our OS-9 manual. There must have been at least 2 editions of the manual.

==Farrell Kenimer==

**A:** The docs for *MODPATCH* came in the manual for **OS-9 Development System**. Remember, our OS-9 Level came in three packages, the main package, the development pak, and the multivue package... if you don't have all three then you do not have ALL of OS-9.

==Jim Vestal==

**Q:** How does OS9 Unzip4.5 compare to the new LZH compression?

==Rodger Alexander==

**A:** First, my only information on the 'new LZH' are the few messages that I've seen on the Fido OS9 and Ribbs echoes about LZH v1.0. So I gather that you are referring to this LZH v1.0.

Unzip45a is a decompressor only. Lzh10 is both a compressor and decompessor based on the -lh1- compression method from what I have read. Somebody else may be able to be more accurate on the details of the compression method, but my minimal understanding is that -lh1- and the PKZIP deflate are Huffman compression schemes. The basic data structure for -lh1- is a 4K (or 8K, I forget) window on the uncompressed data while the PKZIP delfate is based on a 32K window.

==Gene Krenciglowa==

## OS-9 Newsletter

**Editor: Rodger Alexander**

*OS-9 Newsletter* is published monthly by the **Bellingham OS-9 Users Group** and is protected under United States Copyright Laws. No material may be reproduced or copied in whole or in part without the expressed written permission of the Bellingham OS-9 Users Group, 3404 Illinois Lane, Bellingham WA 98226

*Submissions* are welcomed in any format and can be mailed to the above address or sent via electronic mail to the editor: Rodger Alexander, on Delphi (UserID: SALZARD) or FidoNET (1:301/3401@fidonet.org) or Internet (9040180@nessie.com.wwu.ed). Unfortunately, we do not have funds to reimburse authors of selected articles, however a complimentary copy of the *OS-9 Newsletter* containing your article will be mailed to you, PLUS the satisfaction that you have the admiration and appreciation of all of our readers.

The ***Bellingham OS-9 Users Group*** is a hobbyist club, organized for the purpose of providing information, services, products and events that support the OS-9 operating system for 6809/68xxx based computers. Our efforts are not intended to earn or generate any profit for the club or any of


## TO SUBSCRIBE

**For 12 monthly issues of the *OS-9 Newsletter*, please send a US check or money order for $10 or $6 for a 6 month subscription. Mail your subscription order to:**

**OS-9 Newsletter**
**3404 Illinois Lane**
**Bellingham, WA 98226**

**Include your name, address and telephone number. You will receive your *OS-9 Newsletter* no later than the 10th of each month. Canadian orders, $11.50 for 1 yr. or $6.60 for 6 mo. Foreign orders $16 for 1 yr. or $9 for 6 mo.**

PNWCoCoFest

# Club Activities Report

***Bellingham OS9 Users Group - Longview/Kelso CoCo Club***
***Mt. Rainier CoCo Club - Port O'CoCo Club - Seattle 68xxx Mug***

## Bellingham OS-9 Users Group

The May 26th meeting of the Bellingham OS-9 Users Group met in Room 153 at Fairhaven Middle School. Only three people, Rodger and Barbara Alexander and Wes Payne showed up, but a great deal of work got done and will be shared with a large number of people so no regrets in the small turnout.

**Wes Payne** immediately got to work comparing the Basic09 outlines he put together at the last meeting and comparing the print-out procedures of the *PDS Database* that was written by the Seattle 68xxxMUG. Wes almost thinks in Basic09 and just flew through the program. Within a half hour he had a working version done and then began adding options. He debugged the program continuously as he was writing. The final version is published elsewhere in this issue of the Newsletter.

**Rodger Alexander** finally solved the problem with the Laser player demonstration that went awry at the Seattle meeting earlier in the month. Turns out the send and receive wires were solder in backwards, but then, it was Rodger who soldered them in backwards in the first place!

We didn't have an RS-232 Pak available, so we were forced to use a PC (80286) and Microsoft Works Terminal Program to operate the laser player. It worked great! An example of the advantage of the computer interface was demonstrated by the manual procedure required to operate the remote control to search the disk for a specific chapter and get it to play. The operator would have to push the CHAPTER button, then the number desired, then the SEARCH button, and finally the PLAY button. From the computer keyboard the following keys, were typed in: CH8SEPL <enter>. By the time the manual operator had pushed the chapter number on the remote control, the computer had the disk up and running.

Next step is to build/write some simple script files that student can use to operate the laser player for classroom presentations using a combination of computer commands and the remote control unit. Hopefully a successful demonstration will viewed by many at the PNW CoCoFEST in Port Orchard.

==Rodger Alexander==

## Port O-CoCo

The may meeting of the Port O' CoCo Club got off to an excited and prompt beginning. First there was lots of wheeling and dealing. One CoCo user was moving away and needed to liquidate his system. I suggested that he try to make the meeting and bring his equipment. Fortunately, Stock Market Foods has a cash machine just outside our meeting area. He was able to sell his expansive system and software for $200 CASH on the spot. Another older and smaller system sold for $75.

Enough of the capitalism and entrepreneurialism going on among the ranks. We had a meeting to conduct. On the administrative side we have a set of bylaws now. They are only two pages and covers the basics. Taking care of what was needed was due to the efforts of **Bud Helch**. The bylaws were passed as written. Thanks for all the work Bud, hope you are feeling well enough to make it to the festivities in June!

One of our big decisions to be made was the number of mugs to be ordered for PNW FEST III. We have to commit to that about 5 weeks in advance. We have ordered 72 mugs in the past. They have sold very well. This year's design is of universal appeal. It's entertaining and desirable for any of the platforms coming to the event. We will have an IBM area going as well as a Commodore, Atari, Apple IIGS and hopefully others be the 25th comes around. Their numbers may not be as strong as ours. We have been doing this for 3 years but for us to issue a mug just for us is missing a great opportunity. So the mug addresses the chronology and genealogy of chips used in personal computers from the beginning of the amazing period. If you haven't ordered your set of mugs, do so now before the price goes up at and after the event.

But back to the question. How many do we order. The will of the group was to risk and order up to the next price break and get 144 mugs. Just by taking a survey of those at the meeting we had just about 30 mugs sold already. So we shouldn't be in trouble.

**Mark & Barbara Griffith** are testing the waters for interest in a new magazine for the CoCo. They are willing to send out samples to anyone requesting it. They will do this for four months and then decide whether to continue and offer the unnamed publication on a subscription basis. Write for a sample to Mark & Babrar Griffith, Dirt Cheap Computer Systems, 1368 Old Highway 50-E, Union MO 63084.

The big event of our meeting was **Jim Coleman**, sysop of DARKSTAR NASA MLP. I found out about him from a big page article in our regional paper, The Sun. As the title leads you to believe, Jim's board focuses on NASA information, which he gets directly from

NASA daily. He also has information on earthquakes (like every earthquake in the Americas since 1933!). What is not as apparent is his commitment to education. He has enough information in the 180 Megs of memory for someone to home school a student from first grade through high school. And all of this is in Port Orchard at (206) 871-3965. He has just added a CD ROM so the wealth of data begins to blow the mind.

Coleman got his bulletin board from a kid who like games. This board looked pretty much like very other board out there. But Jim is a writer and he wanted to have a place where writers could share ideas. It wasn't too long before he found out that more than 1,200 users a month are also interested in his collection of topics and education. This has all evolved in just nine months (and $6,0000).

In a quiet and unassuming way, Jim beguiled us with how his passing interest took on a life of its own and now consumes much of his time. But just what did all of this LOOK like? Well, Jim's board (his den) was less than a mile away and so those who wanted to journey into NASA MLP BBS were invited to come to his house.

I knew this was an interesting person when looked up at the ceiling of his den. Consuming most of the ceiling space (about 6x8 ft.) was a handmade, professional model of a Klingon "Bird of Prey". Jim is a professional graphic artist. And to have a Bird of Prey means he is into Star Trek. Into Star Trek? Yes, he used to be a dealer for memorabilia and when he grew tired of it he sold his inventory. That transaction made it possible for him to leap into computers with booth feet and the funds to make his fantasies a reality.

The outcome of his fine presentation was his willingness to create a conference area for the CoCo community (area #28) so we can post messages. He is also interested in some of the FidoNet echoes. He is also putting in a second node (phone line) so those calling from around the nation can call in and pull off what they want at high speed without a hassle.

Jim's board is a member of a national association of BBSs dedicated to education and vows to exclude adult (and therefore any pornographic material) from the board. He has met in the last couple weeks with the several school districts in Kitsap County to encourage teachers and students to use his board. He has been VERY WARMLY received.

As a result of his presentation several of our group has already signed on to his board. If any out of towners sign on, leave a message for Jim and for me so we know you are there!

We, here in Port Orchard, are looking forward to seeing so many of you at the FEST. Remember, if you have a friend who is into another computer, bring them along also. We will have something for everyone!

=Donald Zimmerman=

*The Port O'CoCo Club meets the 3rd Monday of each month in the Stock Market Foods Store located on Mile Hill Rd. in Port Orchard at 7:30 p.m.*

## Seattle 68xxxMUG

The May meeting featured a demonstration of a Pioneer (model 6010) 12 inch laser disk player and how it could be controlled by any computer with an RS-232 port (even a CoCo).

**Scott Honaker** began with the basic operation of a laser disk player using a familiar remote control hand held unit similar to a TV remote control. The unit operates similar to a VCR with the exception that the back of the unit has an RS-232 port that can be attached to a computer.

The model demonstrated was connected to a Radio Shack Deluxe RS-232 plugged into a CoCo-3 running *OSTerm v1.8*. BUT, it didn't work. This was very disappointing since it was working the night before. Trouble shooting technique was employed to discover the problem.... Maybe it's the Deluxe RS-232 Pak or the software or the CoCo-3 with it's tendancy to lock up due to IRQ interrupts from the RS-232 Pak.

Another computer was used, a model 200 lap top. It didn't work either. Could it be the laser disk player? Maybe it's the cable, after all it was home made and was not the cable used the night before when it did work properly??? Oh well, another computer meeting and another 80 mile drive (160 miles round trip) all for naught. Grrrrrrrr )~:

Copies of the *OS-9 Newsletter* were passed out to those subscribers present, and attention was focused on the agenda list for the upcoming PNW CoCo Fest III, June 25-26 in Port Orchard. **Donald Zimmerman** was present to answer questions and fill everyone in on other events happening in the Port Orchard area that same weekend. June 25-26 does not have to be reserved for the computer nut in the family, there will be lots of things happening for everyone to enjoy; from a five ring circus to a parade and antique shops galore.

A more detailed and finalized itinerary, along with a map showing how to get there was promised for the next meeting.

=Rodger Alexander

*The Seattle 68xxx Micro Users Group meets the first Tuesday of each month at Gugenheim Hall on the University of Washington Campus at 7:30 p.m. For more information call Scott Honaker at 635-4471 during working hours or 836-9236 in the evenings.*

## Washington State BBS List

**COLUMBIA HTS. BBS**
-- Lonview/Kelso --
RiBBS (FidoNET)
(206) 425-5804

**DATA WAREHOUSE BBS**
-- Spokane --
RiBBS (FidoNET)
(509) 325-6787

**BARBEQUED RIBBS**
-- Bellingham --
PC-Board (PC-Net) - CoCo Conference #5
(206) 676-5787

**OS-9 TACOMA BBS**
-- Tacoma --
RiBBS (FidoNET)
(206) 566-8857

**ULTIMATE EXPERIENCE BBS**
-- Anacortes --
RiBBS (MaxNET)
(206) 299-0491

## *Bellingham OS-9 Users Group*

**OS-9 and the Color Computer** **$5**
*Tutorial* and *Hardware Hacker's Manual.*
Includes 5-1/4 Disk of (360K) of upgrade software

**Color Computer Video Library** **$10**
Fixing the MultiPak IRQ * Installing Floppy Drives
Installing 512K Memory * Installing B&B Hard Drive

**OS-9 Newsletter** **$10/yr.**
12 monthly issues packed with OS9 Update, Tutorials, Listings, Classifieds and PNW "Club Activity Reports"

**Mail your order to:** ***Bellingham OS-9 Users Group***
***3404 Illinois Lane, Bellingham WA 98226***



***OS-9 Newsletter***
***3404 Illinois Lane***
***Bellingham, WA 98226-4238***