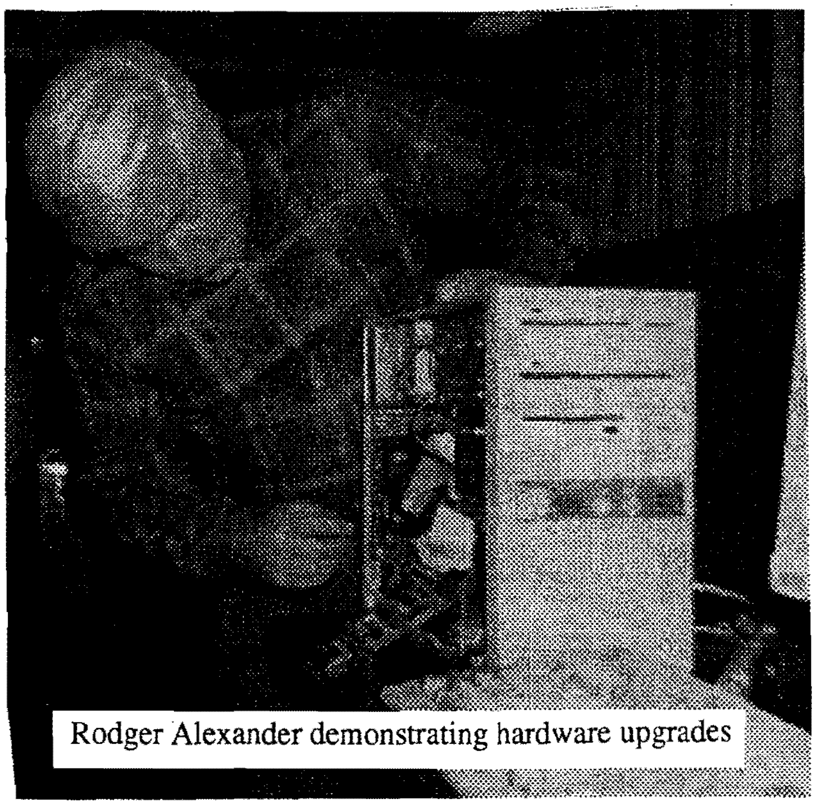
OS-9 Newsletter

Volume II No.7 <<< BELLINGHAM OS9 USERS GROUP >>> July 30, 1991

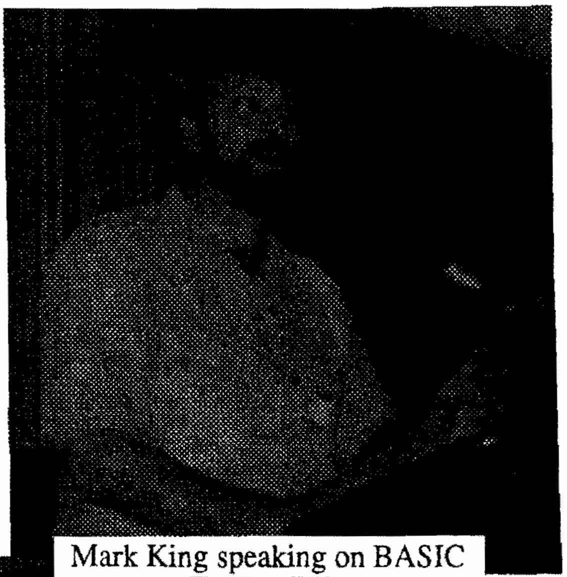

Rodger Alexander demonstrating hardware upgrades

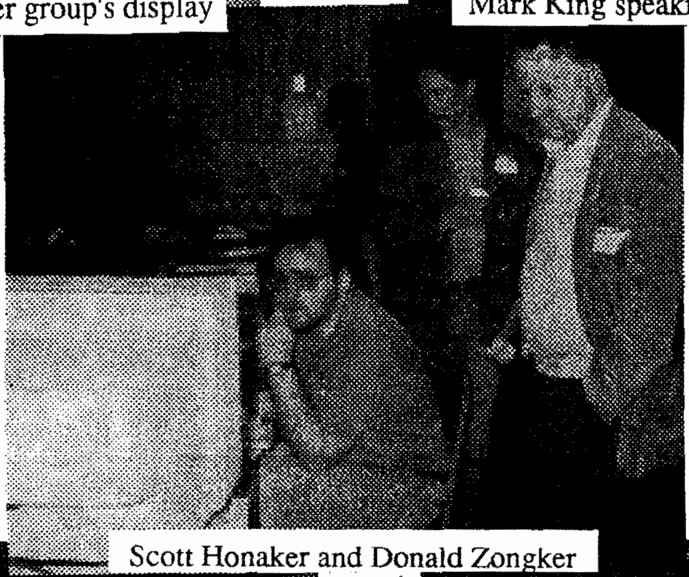
<<-- IN THIS ISSUE -->>

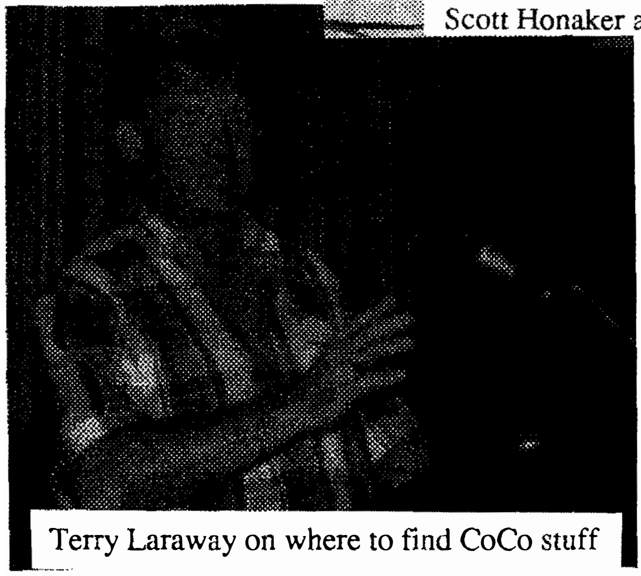
NW CoCoFest

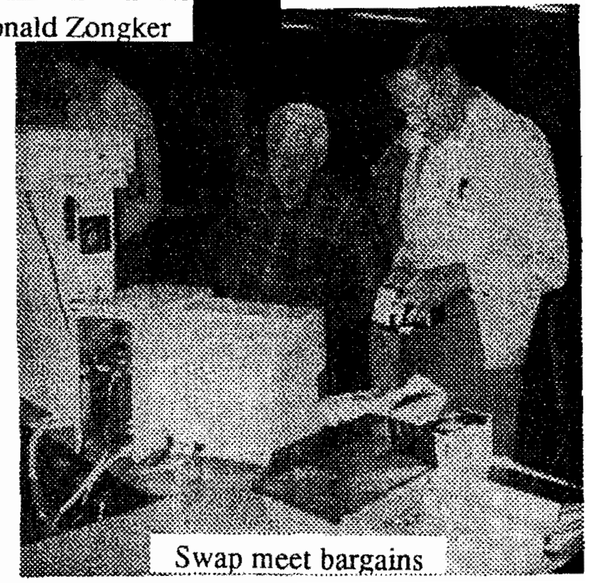
A LAST LOOK....

Bellingham OS-9 user group's display


Mark King speaking on BASIC


Scott Honaker and Donald Zongker


Terry Laraway on where to find CoCo stuff


Swap meet bargains

28 and 32 pin 27 series EPROM Pinouts

						1 Mbit	1 Mbit	Intel
Pin #	2764/			# Pins		571000	271001	270n0 series
28 pin	27128	27256	27512	28	32	27010	571001	2,4,8Mbit
					1	Vpp	-	A19 (8MB)
					2	A16	*OE	A16
1	Vpp	Vpp	A15	1	3	A15	-	-
2	A12	A12	A12	2	4	A12	-	-
3	A7	A7	A7	3	5	A7	-	-
4	A6	A6	A6	3	6	A6	-	-
5	A5	A5	A5	5	7	A5	-	-
6	A4	A4	A4	6	8	A4	-	-
7	A3	A3	A3	7	9	A3	-	-
8	A2	A2	A2	8	10	A2	-	-
9	A1	A1	A1	9	11	A1	-	-
10	A0	A0	A0	10	12	A0	-	-
11	D0	D0	D0	11	13	D0	-	-
12	D1	D1	D1	12	14	D1	-	-
13	D2	D2	D2	13	15	D2	-	-
14	GND	GND	GND	14	16	GND	-	-
15	D3	D3	D3	15	17	D3	-	-
16	D4	D4	D4	16	18	D4	-	-
17	D5	D5	D5	17	19	D5	-	-
18	D6	D6	D6	18	20	D6	-	-
19	D7	D7	D7	19	21	D7	-	-
20	*CE	*CE/PGM	*CE/PGM	20	22	*CE	-	-
21	A10	A10	A10	21	23	A10	-	-
22	*OE	*OE	*OE/Vpp	22	24	*OE	A16	Vpp/*OE (8 MB)
23	A11	A11	A11	23	25	A11	-	-
24	A9	A9	A9	24	26	A9	-	-
25	A8	A8	A8	25	27	A8	-	-
26	NC/A13	A13	A13	26	28	A13	-	-
27	*PGM	A14	A14	27	29	A14	-	-
28	Vcc	Vcc	Vcc	28	30	NC	-	A17 (2,4,8 MB)
					31	*PGM	-	A18 (4,8 MB)
					32	Vcc	-	-

* = function is active when line is LOW (preceeds relevant function)
Vpp = Programing Voltage   NC = Not Conected   OE = Output Enable
Vcc = 5 Volts   PGM = Program   CE = Chip Enable

Pins l labelled as *CE/PGM are often given as merely *CE in spec sheets. I used the *CE/PGM notation to emphasize that on those chips the *CE line is used as a *PGM line during programing.

The 27512 makes the same pin SHARE *OE and Vpp functions. Programing it requires that Vpp be turned on and off It is in this respect unique among the 28 pin 27 series EPROMs, and was made that way to squeeze the most available address lines. The proposed 8 MB 32 pin chip will work similarly. The 27256 and the proposed 4MB 32 pin EPROM also are similar in how they are programmed.

571001 EPROMs are DIFFERENT from 27010 EPROMs in two pins: Pins 2 and 24 (A16 and *OE respectively on a 27010) are SWAPPED on a 571001. BEWARE of this difference between these two otherwise similar 8 by 128K 1 megabit EPROMs!!! 571000 EPROMs are the SAME as 27010 EPROMs.

28 pin 231000, 231024, and 231026 Masked ROMs are like a 27512, except they use pin 22 as an A16 line, and have no *OE line. The CS line might be CS, not *CS, on some of these ROMs.

Make your own 512K

A while ago I got hold of a sack full of about 40 NEC brand 4164-15 chips (64K by 1 150 ns) that had been desoldered from some printed circuit boards. The pins had been clipped quite short.

Well... 4164's brand new retail for about a buck or two apiece, and in quantity and used often can be had for 50 cents each or less. In part because demand is low for these parts, dealers these days seldom will offer more than 20 cents per chip even for prime quality used 4164's that have full length pins. Clearly it was not worth my trouble to salvage these chips by soldering them to a header: The header alone costs more than what the chips were worth!

I put these chips aside, figuring I'd probably throw them out in a few months once I confirmed I had no use for them.

The other day, somewhat bored, I was looking over the pin outs for various DRAM chips and looking at the circuit for the memory chip wiring for 128K and 512K CoCo 3's. It occurred to me that four 4164 chips could be wired up so as to act like one 4464 chip, at least from the point of view of a CoCo. All the address and the CAS and RAS lines would line up perfectly. The ground pins would have to be shifted over one, and the data in and data out of each chip would have to be wired to each other, then each of the four pairs of data lines routed to the appropriate four pins on an 18 pin header for the four data lines for a 4464. The enable line (pin 1) of a 4464 would not be supported by this "pseudo 4464" chip, but the CoCo 3 does not use the enable line of its 4464's anyway.

So... just for grins... I piggy backed four 4164 chips, bending out the data lines and re-routing the ground line. Using an 18 pin machine pin socket as a header and a few pieces of wire wrap wire, I created a very weird looking stack of four chips attached to the machine pin socket and associated with a small rat's nest of wire used to re-route the ground and data lines.

I stuck the thing into one of the 4464 sockets of my main office CoCo 3, and... the machine died! Even when I put back the original 4464, the machine remained dead. Cursing loudly, I checked things out. Although there was no VISIBLE damage to the machine, testing indicated that the ultra cheap single wipe socket used for the DRAM had been damaged by the insertion of the round machine pin of my header. I took the machine apart, desoldered all four 18 pin RAM sockets, and replaced them with high quality Augat gold fill machine pin sockets. When the machine was again fired up, it worked perfectly.

That little bit of upgrading out of the way, I again plugged in my 4 chip piggy-back kludge in place of one of the 4464 chis. BINGO! The machine worked fine. I then made up four such kludges, and was able to get the machine to work fine (even with a disk controller) using these weird pseudo 4464 things. The CoCo sure LOOKED weird, with four little towers four chips high in place of the simple little 4464 chips, but it worked just fine.

Of what significance is all this? Probably not much... this was more a mad hacking little folly, with relatively little practical significance. However, if you are in a place with lots of 4164 chips (which are in fact quite plentiful) and no 4464 chips, and need a 4464 chip, and have acess to an 18 pin header, and the devide in question has LOTS of clearance above the DRAM chips, and the circuit that uses the 4464 chips does not make use of pin 1 (ties it high, as does the CoCo 3) on the 4464 chips, THEN you, too, can kludge up a pseudo 4464 chip. Similarly, if you have a gizmo that needs a 4 by 256K chip that does not use the OE line on those chips, you probably can kludge up an acceptable substitute in a pinch by using four 41256 chips and a 20 pin header and appropriate re-routing of data and ground lines. If you actually are crazy enough to try this, tho, I suggest you solder a .33 mfd cap between +5 and ground pins on top of the stack of chips, to quiet switching noise.

Happy hacking!

-Marty Goodman;delphi-

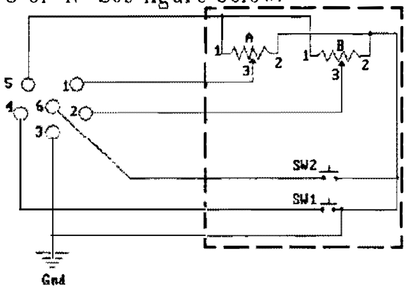
***** FOR YOUR CONSIDERATION *****

ANNOUNCEMENT: Are you interested in the latest and future developments of ALL the new 68XXX machines; application/utility software reviews, advantages of the available operating system(s) and hardware action? We are pleased to announce "The 68XXX Machines". "The 68XXX Machines" is an inexpensive newsletter communication between Users, Software/Firmware/Operating System Writers, Hardware Hackers and all Machine Manufactures. Come and be a part of the newest, most in-depth information vehicle for the new 68XXX machines, their operating systems and software.

Call or write today:
The 68XXX Machines
RD1, Box 375
Wyoming, DE 19934
302/492-8511

Install a Keyboard Switch and a PC Power Supply

I wanted to install a "keyboard lock" switch on my CoCo and saw the OS9 Tips article in the March '91 OS9 Newsletter, in which Brian White suggested switching +5VDC to pin 4 to lock up the keyboard. But before I went ahead and tried this, I checked the CoCo-3 Technical manual and discovered that the ground potential, not +5VDC, accomplished the lock-up of the keyboard, and that the ground potential can be applied to either pin 3 or 4. See figure below.



One of the advantages of installing a Color Computer into a PC case is the large single power supply available to supply B+ to all of your floppy and hard drives plus the Multi-Pak and the Color Computer. You eliminate at least three heat producing transformers and lots of power to spare.

Connecting Disk Drives:

Connecting power to the drives is made simple since the PC power supply (150-250 watts) already has several power cables coming out of the power supply case, each with a standard 4 pin keyed connector. It doesn't matter which connector you use, all of them have +12VDC, +5VDC and GND properly located in the connector. Just plug one of them into your drive(s)!

Connecting Multi-Pak:

Connecting power to the Multi-Pak is a little more complicated. Since you no longer need any of the Power Supply components on the Multi-pak circuit board, you will need to remove the large heat sink and the voltage regulator that was attached to it. BUT before you do that, locate convenient connecting points on the circuit board in front of slot #1 that have +12VDC -12VDC +5VDC and GND.

After removing the heat sink and the voltage regulator, solder 20 gauge or large diameter wires to the 4 voltage connecting points that you located in front of slot #1. Wire length should be about 10 inches or longer in order to make a convenient connection to the six pin power connector jacks coming from the PC power supply.

Use a voltage meter and/or check the wiring diagram posted on the power supply to determine which potentials are available from the two six pin jacks. Connect the 4 wires coming from your Multi-Pak to the appropriate pins on the 6 pin power connector jacks. This can be done either by soldering your wires to a 6 pin header or a 6 pin male connector designed to match the 6 pin female connector coming from the power supply. AGAIN make sure you match the wires with the proper potentials.

Check your work! With your four wires connected to the PC power supply, turn on the power supply and use a volt meter to check the voltage test points labeled on the Multi-Pak circuit board. The voltages should be within a couple of tenths of a volt of the desired ratings. Usually the voltages will measure a little low due to a lack of sufficient load on the PC power supply. When everything is connected (Computer and Drives) the voltages will increase to their optimum potential.

Connecting the CoCo:

Connecting the PC power supply to the computer's circuit board is similar to what was done above to connect the Multi-Pak, with a couple of exceptions.

Locate the large heatsink extending out from the left bottom corner of the computer's circuit board (keyboard facing you). Locate the 16 pin integrated circuit (IC8) mounted just in front of (below) the heat sink. This is the "salt" chip which contains the 5VDC voltage regulator and the plus and minus RS232 voltages for the "bit-banger" port. Turn the computer power on and carefully locate the +12VDC and -12VDC on the salt chip. The +12VDC should be found on pin 16 and the -12VDC should be found on pin 15. Locate the 5VDC on pin 2. The ground potential is located on pin 8 or any ground reference on the circuit board.

Remove the heat sink and the attached power transistor "Q1" (looks the same as the voltage regulator on the Multi-Pak) from the computer's circuit board. Solder approximately 10 inch length wires to the +12 -12 +5 and GND potentials located in the above paragraph. Connect the wires to the correct pin-outs on the PC power supply 6 pin connectors in the method described above when connecting the Multi-Pak wires.

Check your work: Turn on the power and observe any smoke, sparks, etc. The computer should operate normally with one exception....No audio or TV signal output.

Audio Amp Power:

The Audio amplifier and the TV "box" have their own half wave power supply that produces a regulated 8VDC. Since the PC Power Supply does not supply 8VDC, the solution is to supply +12VDC to the input of the 8 volt regulator (IC36). This voltage regulator looks like a small transistor located next to the two large diodes on the left center side of the circuit board (keyboard facing you). Unsolder or cut the 8 volt regulator (IC36) input lead (labeled "IN" on the circuit board). Solder a short length of wire to the input lead on the voltage regulator and solder the other end of the wire to pin 16 of the salt chip where you have brought in the 12VDC from the PC Power Supply.

Comments:

The voltage regulator on the Multi-Pak and the Power Transistor on the computer (Q1) are removed not only because they are not necessary, but because they also consume power and generate a great deal of unwanted heat. If you have the older Multi-Pak, the board itself can be cut down eliminating all of the Power Supply components. See Craig DuBois modifications in the April '91 Newsletter.

--Rodger Alexander--

DOUBLE DOS

Many hardware mods have been implemented lately for the color computer with various controllers. I have taken this into consideration here and made this article as brief as possible to eliminate possible confusion. Some thought has to go into this to make it work correctly.

This modification assumes that your DOS fits into an 8k EPROM now and by combining 2 of them, will then fit into a 27128 16k EPROM.

I use the disto controllers exclusively and have very good results from their use. This article can be used as a guideline for adding the modifications to either the Super Controller-I or II.

First of all, let's get the software combined into one file. If you are familiar with offset loading of machine language programs this will be a snap for you but it's really intended as a beginners project and the results should be very rewarding.

If you have in mind the 2 DOS's you would like to combine into 1 file we are all set. If not, here are is an example of what you can combine. I've done this so I know it works. Let's say for example you want Radio Shack DOS and RGB-DOS together in your Super Controller II. First determine which you want in the lower position, either will work in that position. Make a M/L save of both of your dos's to a disk. This is done by typing the following line and substituting your DOS name for the filename.

```
SAVEM "DOS/BIN",&HC000,&HDFFF,&HC000
```

This will make a copy of your DOS on the disk for use in our offset loading process. Now, run your other DOS from either disk or eprom and save in the same manner to disk. when you have them both saved, we will have our 2 DOS's to combine.

Offset load the first or lower position DOS to 2000 by typing this next command substituting your filename for "DOS1".

```
LOADM"DOS1",&H6000
```

After it's loaded into memory, we will need to save it back to disk at the new addresses that it's been loaded to in RAM. Do this by typing the following line.

```
SAVEM"DOS1",8192,16384,40999
```

We now have the first DOS saved to the disk ready for combining with the second DOS. At this time, repeat the procedure with the second DOS by offset loading and saving it by the method described above.

Now we will load the newly saved DOS's into ram for combining into one file by typing the following lines.

```
LOADM"DOS1"
LOADM"DOS2",&H2000
```

This will load DOS1 in the lower position and then offset load DOS2 to the upper position in ram. When you have both DOS's loaded in this manner, proceed with the next step by saving the combined 2 DOS's into one file with the following line.

```
SAVEM"DOS/BIN",8192,24576,40999
```

You now have them both combined into one file on your disk, ready for burning into a 16k 27128 EPROM. This can be done in various ways so I'll leave that part up to you. Contact a friend or company that does this service. I used the DISTO EPROM PROGRAMMER with my Super Controller-I to achieve this.

<u>HARDWARE</u>:

We now arrived at the second part of the project. I will describe the hardware wiring for both the SC-I and SC-II controllers which this project was designed for.

First the SC-I:

Remove the screws from the case and open it up. Now examine it where the rows of jumpers are next to the DOS chip socket. Jumper 4 is the one closest to the notched end of the DOS chip or looking at it in it's plugged in position in the coco's ROM pak port, it's the rear most jumper. Refer to your SC-II manual for locations if this is not totally clear!!

Remove jumper 4, and find 3 lengths of wire that when soldered to each of these now exposed 3 pins, will extend to the floppy plug end of the controller. You'll need a SPDT toggle switch such as the Radio Shack micromini switch part # 275-625. Locate a place to drill a mounting hole at that end of the controller lower case frame near the floppy connector opening. Make sure that it will not interfere with any cards you may have installed in the controller such as a 4-in-1 board or other add-on board. With that in mind, mount your switch and solder one wire to each of the 3 terminals on that switch. Do not cross any wires while soldering to the pins or switch !! Keep them in position as they appear at the jumper's pins and solder to the switch in a similar manner, center jumper pin must be connected to center on the switch or it will not work !!

Now insert your new 2 DOS chip if you haven't already done so and install the controller into the coco'S ROM pack port. Power up the coco and watch for a normal power up screen. If all's well, you'll see one of your DOS's power up messages as you normally would. Now, power down the coco again and flip the switch to the opposite position, and power up again. You should now see your other DOS in action. If all seems normal, power down, remove the controller and replace the top and screws. Reinstall the controller when you are through and enjoy your efforts!

For the SC-II **controller**: we will be looking for the jumpers closest to the notched end of the chip. locate it and remove the jumper top and solder three wires to these pins, keeping them in order, also solder the other ends to the # 275-625 toggle switch from radio shack. Install your 2 DOS chip in slot one of the controller and test it as above.

I hope this has been a rewarding project for you with the DISTO controllers. You can now have 5 dos'S in the SC-I and 2 in the SC-II.

--SKYLINE;Delphi--

A New 'C' Routine that doesn't exist

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 (JBUCATA);Delphi

Preview . . .
Public Domain Files

FREE:
This is my replacement for the Microware FREE command. It has several features that I've been wanting for some time:

- Speed. This is up to 4 times faster than the Microware FREE command, which is especially nice on large hard disks

- User-friendly output. This FREE command outputs the relevant information in kilobytes, rather than in sectors. It also gives the total disk usage as a percent of the disk capacity. I find this output format much easier to read and understand. Options are available to force the output to be in bytes, sectors, clusters, or megabytes

- User-friendly input. Will accept device names with or without a leading slash, or directory names such as '.'. Such names are converted into a device name for which the information will be displayed. Specifying '-x .' will display information about the device containing the current execu tion directory. This FREE will also accept multiple device names on the command line.

Full source is included. Tell me if you manage to improve it!

- Tim Kientzle -

● ● ● ● ● ● ● ● ● ● ● ● ● ● ●

VU:

Vu is a simple text file viewer made to allow you to quickly look through files. It can be used for viewing documentation, searching files for specific items, and printing or saving certain sections of a document.

Vu requires a Tandy Color Computer 3 running OS9 Level II. The program must be run in a screen capable of 24 lines of 80 columns of text. The program requires 8K of memory for the program and 8K of memory for data.

Two versions of vu are available. One version uses direct screen access for increased speed, but only supports text screens and may not work with future versions of OS9. The other version uses standard system routines to display text. It is much slower, but will work with any screen that has at least 80 columns and 24 lines. An install program supplies directions on how to install either version of vu.

To start the program simply type vu <filename>. This will clear the 80x24 display in the window's current colors. The beginning of the file will be displayed on the screen as well as an inverted bar at the bottom. On the right side of this bar is a number that indicatesthe percentage of the file that the bottom of the screen is at. This bar is also where any input will be typed in. To get a list of commands hit ?. The commands are fairly straight forward. You can use the up and down arrows to move backward and forward though the file or SHIFT with the arrow keys to scroll through it. The text file can be of ANY size. Vu will read the file as it needs it. Because it doesn't need to read the whole file in at the beginning, it requires very little time to get the first screen of text up. Vu will also display a page of text more quickly than would the standard list command. Vu is not very fancy and is not intended to be. It is meant to be small, fast and convenient for viewing and searching text files. A more elaborate version might be made in the future that will add other features that are needed.

- Vaughn Cato;Delphi -

● ● ● ● ● ● ● ● ● ● ● ● ● ●

INTERLEAVE:

The whole point of this utility is to see what disk interleave factor (used when formatting a disk) is the most efficient. The BASIC09 program formats the disk with the various factors and then runs a series of read/write tests and times them. The results of each test is then written to a data file on your default directory.

It seems from the results on my system that an interleave of 2 would be slightly more efficient than the default interleave of 3.

- Bob van der Poel;
Compuserv[76510,2203] -

● ● ● ● ● ● ● ● ● ● ● ● ● ● ●

PHONE:

Phone is a fairly simple program which does so many things. It can be used as a dialer for voice or data calls, plus with it's scripting options it can be used to totally automate BBS logins, etc. As a matter of fact, you could even use phone to totally automate an entire BBS session without a terminal program.

Phone can access file in two basic ways. First the data can be specific to a particular login (eg. calling up CIS). In this case the data for phone should be placed in a file in /dd/sys/phone. On the other hand (mainly to save the effort on maintaining numerous small files) you can place a series of numbers in the file /dd/sys/phone/numbers. In this case phone will scan the file and search for the specified name. For testing purposes you may want to create files in your current data directory, etc. Phone will ignore the /dd/sys/phone directory if you start the filespec with either a "." or a "/". In this case the specified filename will be assumed.

At this point I'll include a couple of examples. After these we'll discuss the various options available. First off, a short portion of my "numbers" file:

```
\: Rick
\s ATDT 1234567 $0d \i \q

\: Bob
\s ATDT 444 4152 $0d \i \q

\: Peter Smith
\s ATDT 473 1111 $0d \i \q

\: Rick Jones
\s ATDT 1 444 123 3540 $0d \i \q
```

Note that each "name" line is terminated by a carriage return.

Next a sample logon file. This is the one I use to access CIS (of course the password is changed to protect my bank account!):

```
\e +
\d 10
\1
\s ATZ $0d
\w OK 2 1
\p 1
\s AT DT 466 4501 $0d
\w CONNECT 20 9
\p 1
\2
\s $03
\w : 5 2
\s 76510,2203$0d
\w : 5 2
\s my*password
\q

\9
\s +++
\p 2
\s ATH0 $0d
\q
```

If you haven't figured it out by now, all phone commands are prefixed by a "\". Commands are not case specific, you may use either \s or \S. Following is a list of all the commands:

\s Send a string to modem path.
\p Pause.
\w Wait for string.
\a Wait for any character.
\x Shell command.
\i Get key from stdin and echo to modem
\q Quit.
* comment
\o Open modem path.
\c Close modem path.
\e Toggle echoing of received characters.
\d Set output delay.
\j Jump to a label.
\1 .. \9 Label.

- Bob van der Poel;
Compuserv[76510,2203]-

MM/1 Update
by Paul Ward

Before we get started, let's evaluate where the MM/1 is now:

- First, the two board kit is still being sold. The price has been raised to $975, so the $875 special price is now over. Also, our case price has been raised from $100 to $125. With the kit comes some additional free software, enhanced windowing software, new documentation and some technical literature.

- Second, more technical literature is being prepared that will also provide specific ideas on projects with the MM/1.

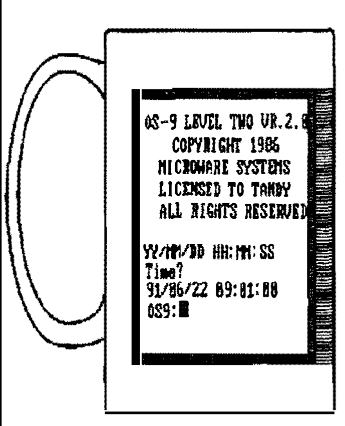
- Third, improved SCSI drivers are in beta test that bring the MM/1's SCSI subsystem to workstation level speeds -- 1.8 Megabytes per second transfer rates on a Quantum Pro drive.

- Fourth, updated hardware and software is beginning to be distributed to developers.

- Fifth, the MM/1 kits are shipping. We had some delays after the Fest with a mask error Western Digital introduced into their SCSI host adapter chip. Kits WITHOUT the I/O board are shipping now, and have been shipping for weeks. We should have well over one hundred MM/1s in the field by the end of July, and shipping will continue through August until we are caught up.

The mask change of the SCSI host adapter chip was introduced only in later models by Western Digital. Three hundred chips from an older mask have been reserved for IMS. These are being tested this week. An update will be available soon on this information service, or at 202-232-4246 between the hours of 2 pm and 5 pm, Monday through Friday.

Paul K. Ward
Interactive Media Systems, Inc.



NY CoCoFest Cups $6

CONTACT DONALD ZIMMERMAN at 871-6535

"*HURRY*"...LESS THAN 15 CUPS LEFT....

Club Activities

Seattle 68xxxMug

The July meeting of the 68xxxMug almost started on time. Don Zimmerman displayed and sold some NWCoCoFest souvenir Mugs at $6 each. "THANK YOU" and "JOB WELL DONE" were expressed to Don and the Port O'CoCo Club for hosting and organizing a very successful NWCoCoFest.

Rodger Alexander began the first part of the meeting with a hardware modification project involving everyone in the meeting helping out. Following the instruction presented in the July Rainbow Magazine on converting the Modem Pak to an RS232 Pak, Rodger demonstrated the actual process of cutting off part of the circuit board and soldering wires to the integrated circuit while individuals were also busy cutting away part of the case to accommodate the RS232 jack and drilling holes in the circuit board. At the end of the half hour demonstration, approximately 2/3 of the conversion was completed with only 2 capacitors and 4 wires still left to solder in. Rodger promised to bring the completed project to the next meeting or better yet, a videotape of the entire project.

The second presentation involved Donald Zongker and Jeff Brittan installing a second hard drive to Donald's CoCo system. There were some minor bugs at the beginning that had to be corrected and a proper hard drive descriptor had to be installed. but after that things begin to go together properly.

During a "technical break" in the hard drive installation, several OS9 software programs were demonstrated:

- 1. **PT File Management Systems** manufactured by r3 systems is a complete utility program that features file options such as ALIAS file naming, an automated FILE INFO/ATTR utility, a LIST VIEWER, a MOVE utility, and a FIND utility. PT File Management also includes directory options such as COPY DIRECTORY, SEARCH DIRECTORIES and a TREE DIRECTORY display. You can also indicate what default command you want in effect. Highlighting a file/directory and hitting the enter key cause the default command to process the selected file.
- 2. **MMENU** is another product by r3 systems that display attractive overlay windows with menu options read from a script file. Written in assembly code results in a fast reliable menuing environment for those who don't want to have to deal with DOS commands.
- 3. **EASY EDIT** is a Public Domain program from the Europe OS9 User Group. It reads your OS9 boot files and displays all of the modules in your boot file that can be modified. The user highlights the desired module and is then prompted for all possible modifications permitted to that module. When all modifications are completed. EASYEDIT re-writes your OS9Boot file. No COBBLER or OS9GEN or EZGEN or DEBUG or DED is required.
- 4. **HOME PUBLISHER** was displayed. Although not a new program it's features are greatly enhanced via a hard drive and the new high speed GRFDRV utility from Kevin Darling.

At the end of the meeting, members were asked to bring their problem software and hardware to the meeting for help and solutions.

The next meeting wll be Tuesday, August 6 at 7:30 p.m. at Gugenheim Hall on the U of W Campus. Part of the presentations will include "Configuring" your OS9 Level II System disk to match your hardware.

● ●

Bellingham OS-9 Users Group

The Bellingham OS9 Users Group had no formal meeting in July. However, Craig DuBois completed the finishing touches on his tower case installation and traced down the lost audio problem (See the PC Power Supply article). We discussed the scheduling of remaining video tape session and the need to update the Floppy Drive sessions to include the different configuration of the Tandy FD-502 Drives.

Our newest member, Ray Flick, is a professional video technician and offered some suggestions to improve the appearance, reproduction and some marketing ideas. Ray also just purchased a second FD-501 drive and a second FD-502 drive and wanted help to install the second drives into the first drive cases. The new drives required that they be reconfigured as DRIVE 1's. The FD-501 drive was simple since the drive select jumper are well marked and easily accessible. But the FD-502 drive did

"How To"

Video

Installing "IRQ" Hack
Installing 612K Ram Kit
Installing B&B Hard Drive
Upgrade Speech Sound Pak
Installing a 2nd Floppy Drive
Modify ModemPak to RS-232

$10.00

Plus $2.00 Shipping & Handling

1-734-5806

not have the drive select jumpers labeled and took some educated guess work to find the jumper pins. We lucked out and just happened to move the jumper block to the correct pins to re-configure the drive to DS1 (Drive Select 1). We also noticed that the 34 pin card edge connectors did not have missing pins for drive select. This is the only model that Radio Shack did not pull the pins out of the jacks to accomplish drive selection.

●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●

Longview/Kelso CoCo Club

No meeting is scheduled for August due to vacation conflicts. Contact Steve Hammond at 577-7316 for information about the Longview/Kelso CoCo Club.

●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●

Mt. Rainier CoCo Club

No meeting is scheduled for August due to vacation conflicts. Contact John Schliep at 472-0228 for additional information about the Mt. Rainier CoCo Club.

●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●●

Port O'CoCo Club

The group is now meeting at Stock Market Foods store in the K-Mart Plaza half way up Mile Hill in Port Orchard. The Kitsap Bank Community room is not available because of company training. We don't know when that commitment will be complete.

The meeting began with a thank you to Phyllis Young. She has assumedthe role of publicity. She calls everyone on our list and reminds them of the meeting. She will be expaining her efforts by regularly notifying the local papers of our meeting time, place,and program. Thanks, Phyllis!

We were honored with guests this month from afar. Cliff Remmem journeyed by bus and boat all the way from Edmonds. He had also attended the Fest in June. Thanks, Cliff, for taking the time to become part of our group!

The Lake's were from a little further. They came from Downey, California! They had read about the group in the NW Computer User on the ferry and made a point of attending. Since we were not in our usual location it took a little sluthing on their part, but they found us. Both Bill & Dee are members of the South Bay Users' Group, SBUG. The group meets twice a month at alternate locations. About 40 people show up at a meeting. They have a monthly newsletter. It was their group that had put on the Rainbow Fest in California some years back. Bill is exclusively into BASIC. He has written several programs for the club's use. Since they are traveling around the country he has his entire system mounted in his 30ft. RV. So every time he looks out the window, he has a different view! Dee writes long letters to their daughter who in turn mails copies to the rest of the family. The Lakes store the letters on disk thus having a disk journal of their travels.

The meeting started out with a critique of the NWCoCoFest. A few of the comments were a) more time for the the swap meet, b) have the entire event at one location, c) start the publicity far sooner, maybe even December or January, d) spread the word over a much larger area to draw people from further away. If you have any comments/suggestions, please let us know so we can make the next event that much more successful.

Just having the event this year makes NWCoCoFest-I a success. But to underscore that Tom Brooks gave us a report on the money needed to put on the event and the money in the cookie jar now. It cost over $300 to put it on; the income was just over $600. So the event was a financial success. There were a lot of volunteer man/hours and a lot of commitment by a lot of people. That's what made it work. And thanks to Tom we have a record of everything.

The next question was what to do with the funds. One suggestion was to buy a CoCo 3 for the club so we would always have one at the meeting. That idea was not supported by the group. The discussion ended with agreement that we would put the funds in a savings account in SeaFirst Bank and then work from there. Tom, you can have your cookie jar back!

Another topic was whether to have two meeting places for the club. One in Silverdale the other in Port Orchard alternating between the two. That idea was not supported because of the possible confusion that would result from switching back and forth.

Mark King was the highlight of the evening. He recapped the beginnings of his presentation on C and continued with a full lesson. His series on C will continue for the next few months.

The meeting ended with requests to purchase several of the unique NWCoCoFest mugs. We are down to our last few! Order yours now or go without! The next meeting is on August 19th at Stock Market Foods.

OS-9 NEWSLETTER Subscription Information

The OS-9 Newsletter is compiled and printed monthly by the Bellingham OS-9 Users Group. Subscription rates are $4 for 6 months or $7 for 12 months. Mail Subscripton Check to:

Rodger Alexander
3404 Illinois Lane
Bellingham, WA 98226

Washington State BBS List

FAR POINT BBS - Seattle
RiBBS (Fido NET)
(206) 285-8335

COLUMBIA HTS. BBS - Longvie/Kelso
RiBBS (Fido NET)
(206) 425-5804

DATA WAREHOUSE BBS - Spokane
RiBBS (Fido NET)
(509) 325-6787

BARBEQUED RIBBS - Bellingham
PCBOARD (PC-NET)
(206) 676-5787 Conference #5

Bellingham OS-9 Users Group
presents . . .

Written by
Scott Honaker & Rodger Alexander
Mail $2 + $1 shipping fee to:
3404 Illinois Lane, Bellingham, WA 98226

OS-9 Newsletter
3404 Illinois Lane
Bellingham, WA 98226