OS-9 Newsletter ©

Volume III Issue 9 | Bellingham OS-9 Users Group | September 30, 1992

Another OSK Machine?

The [redacted]30 (working name only) is a very impressive computer as I'm sure you'll agree once you've read the details below.

The [redacted]30 is our new computer system that is compatible with existing graphics computers like the **TC70.** However it is much faster, more expandable, more flexible, and lower cost than anything currently available. Now that's a strong statement!

Here are some facts:

- Motorola MC68030 (Full version, not the limited EC version)
- 17 times faster than the TC70. (16 Mhz version, 25 Mhz and 33 Mhz proportionally faster)
- 1 to 128 Meg of RAM ON the motherboard using SIMM memory. Uses either 256K, 1Meg, 4Meg or 16Meg SIMMs.
- 8K Battery backed parameter RAM
- Full SCSI port with fast DMA.
- All density floppy controller with DMA. Supports up to 2.8Meg floppies including 180K, 360K, 720K, 1.2Meg, 1.4Meg and 2.8Meg floppies. All formats supported.
- 2 Full serial ports. With full modem control hardware.
- 1 full parallel port for printer or other use.
- Real Time Battery backed clock.
- Autoboot from floppy or hard disk included.
- 8 Built in IO expansion buss.
- 4 Built in 32 bit data and 32 bit address expansion bus.
- Available as 16Mhz, 25Mhz or 33Mhz
- Math co-processor included (25Mhz and 33Mhz version. Option on 16Mhz)
- Nominal 9 x 12 " size drops in any PC case without modification.
- Can run as a terminal system and add a video board(s) later.

EXPANSION CARDS

Video Graphics Card - Uses Signetics VSC (Like TC70) - On board video RAM (Does not use system RAM nor does video contention slow the system like on the TC70) - Palette Controller (16+ million colors) - Dual Stereo port, in and out. - Mouse port (can also be used as a standard serial port) - AT Keyboard port - G-Windows is also available.

All external connections are on the rear

Continued Next Page

OSK on ATARI ?

GREAT NEWS! for AtariST and MegaST owners or future owners: OS-9/68000 for the private user is now available at a much reduced price.

If you have access to Compuserve, see file ATARIOSK.ZIP in the Atari 8-bit help section #1. I will also try to up load the file to Delphi AtariST section.

The new package comes with Sculptor, Stylograph, Dynacalc, C Compiler, Basic, Screen Editor, 68000 Assembler, debugger and linker and a complete set of manuals (8kg weight).

Address of vendor & copyright holder: CUMANA, Ltd.
Pines Trading Estate
Broad Street
Guildford, Surrey,GU3 3BH
England

Their press release is dated Sept 7th, 1992 so this is new.

Hoping to see new Atari OS9ers in the good ole USA soon.

-- Farrell Kenimer;FidoNET --

<<-- Inside This Issue -->>

bracket. Installation is plug in and install 1 screw. This board is a complete self contained graphics workstation board. Up to 4 Video boards can be installed in the system making this the only Multi-Graphics, Multi-User system in the world! As the cost of the video board is so low this means that a multi-graphics system is much lower cost than multiple workstation systems.

Hi-Res Video Graphics Card

Details not complete. Goal is for a 1024 by 768 resolution. Can be intermixed with VSC video boards. Available December 92.

4 port serial card. Up to 4 can be installed for a total of 16 additional ports. Uses 4 DB9 connectors (All mounting hardware and connectors included.)

Dual SCSI with Dual DMA.

This is in addition to the SCSI port on the motherboard. It is designed to support the SCSI-COM. (SCSI-COM is a 16 port Intelligent IO processor. Powered by a 68020, the SCSI-COM connects to one of the Dual-SCSI ports and adds 16 ports. Up to 7 SCSI-COMs can be added to each of the two SCSI ports. More than 1 Dual SCSI card can be added. Physical limit is therefore 14 SCSI-COM boards for each slot. With 8 slots that works out to 112 ports.) The bus structure is elegantly designed and has no limitations for future possibilities. Complete bus specifications will be uploaded later. It is possible to have a memory card that can hold up to 512 Megabytes with this bus design. Four such memory cards could be installed for over 2 Gigabytes of RAM! However with 128 Meg on the motherboard this may not be needed.

The bus is a full 32 bits, both for data AND address. The bus runs at full CPU speed. The bus is not a limiting factor like the AT bus used on most PCs available today. (The AT bus is limited to 8Mhz and is only 16 bits.)

The performance gain of this system over current systems like the TC70 is dramatic. In one benchmark this system ran 17 times faster than the TC70, and that was the 16Mhz version! The 25Mhz and 33Mhz versions are proportionally faster.

In todays economy cost is a major factor. This system is designed to start at only 1500. However for those who order early the price is only 1199! Please call for full details on this amazing offer. This price includes all software, including Professional OS9 with C and BASIC!

We feel that this system is the definitive computer for OS9/68000. There is nothing like it in the world.

All system include full schematics and other hardware details not usually supplied with computers today. Nothing is held back. No hidden 'extra' costs.

{ Price: $1199 }

The $1199 special price includes the 16Mhz KX30 Motherboard with zero K, Professional OS9/68000 and all utilities, manuals, schematics etc..

It requires at least 4 SIMMs (80 ns or faster). Because 1 Meg SIMMs are selling in the $25-35 range on the open market I would suggest that. You need to add SIMMs 4 at a time (It's 32 bit, 4 x 8 = 32 bits) Some people have 256K SIMMs on hand and could certainly use them. The KX30 holds a total of 8 SIMMs and they can't be mixed. ie all 256K or all 1Meg etc.

A 'kit' that includes Mini-Tower case w/200 watt PS, 1.4/720K 3.5" floppy and all cables sells for $185

Hard drive prices are erratic. A Quantum LPS105 with software loaded currently sells for $540. Call for latest prices. You can install any brand SCSI hard drive. I think Quantum is the best.

The VSC video board will sell for $400. If it is ordered with the KX30 it is only $300.

The 4 port serial is $200 as is the Dual SCSI board.

The SCSI-COM board only is $799.95. It needs a case/Power Supply and 16 serial cables.

Assembly is very easy. The KX30 mounts to the standard PC motherboard mounts. The 2 serial DB9 connectors and the DB25 parallel fasten to the back of the case and plug into the KX30. The floppy and hard drive mount in the case and connect to the KX30. The video board just plugs into one of the slots and is held firm with the top bracket screw just like PC cards. All connections on the video board are on the bracket, no internal cables needed. Makes for a very neat installation.

The whole thing should take less than 20 minutes.

We can maintain these low prices if we have the volume so order now and save $300. Shipments will begin in 6 weeks!

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

PS. The above information could also be called. "What we did on our summer vacation." We're very pleased with our ability to keep this a secret until we were ready to take orders. In the past we usually let the cat out of the bag months before the product was ready. This time we're ready to take your order the day we announce the system.

-- Frank Hogg -- ---

TUTORIAL: Changing file names (NOT Renaming)

A DETAILED general procedure for renaming an executable module by lengthening it and appending the name at the end. A "general procedure" because it's more important to know HOW and WHY to patch, so that you can later patch your own modules rather than patch by rote when lucky enough to find a patch posted by someone else. This exercise "renames" the module rma to the new name of c.asm. Applying the METHOD to be shown, you may be surprised to find you can patch the name of a module of your choice, not just my example, to a new one.

Why not just use RENAME? Because "rename" works on files, and you can have many modules merged together within the same file. Changing the name on the outside of a jar of blue marbles, to rename that jar "green marbles", doesn't change the marbles inside. Similarly "renaming" a file does not change the modules inside. You have to get inside the modules themselves to change their names. Cosmetic changes to the name of the file folder is not going deep enough. Files are different from modules.... are they? YES. A misconception is that when you give a "dir x" command, you see a listing of the modules in your execution directory. Hmmm... What you see is a list of file names, with each file name USUALLY containing a module with the same name as the file name you see displayed. Got the Rainbow Guide to Windows? If you do, check out page 126. You can put a LOT in a file, and what you call the file, or later RENAME the file, doesn't touch the module names inside it. You have to go deeper. With the proper tools, we will. This procedure assumes that you have...

A) 512K CoCo3 (if you have 128K, place all the commands I ask you to "load" instead in /d0/CMDS so OS-9 can find them without changing diskettes. You haven't room to load them all with 128K).

B) OS-9 Level 2, vr. 2.00.01

C) OS-9 Level 2, Development System (Sure, some, like me buy it when they buy Level 2, and later, maybe NOW, find a use for it).

D) 2 floppy disk drives /d0, /d1

E) Some basic understanding of OS-9, such as the difference between a root directory and a directory of executable commands.

F) Some need to patch an executable module's name. My need comes from page 83 of the October 1989 Rainbow, changing the name of the "rma" module to "c.asm". Do not patch frivolously. The system you may screw up is your own.

ADVISORY... All OS-9 diskettes you use in this procedure will be backups or copies of your original software. NEVER use the originals from Tandy except to make backups. Since you are using copies to work from, if you do make a mistake, you can start over.

1) Boot your system with the Operating System Master disk (copy) in /d0, and the OS-9 Level 2 Development System Side -A- disk (copy) in /d1.

2) Whenever you see an OS9: prompt, YOU type what follows, beginning now...

OS9: window.t80s

If you have an rgb monitor then also type:

OS9: montype r

OS9: (press **CLEAR** key, you are now on our 80 column work screen.)

3) Load the tools you will need into memory for easy access. (With 128K, just make sure you have all those loaded in this step in /d0/CMDS)

OS9: load attr huild edit ident modpatch rename
OS9: chx /d1/cmds
OS9: load dump save verify
OS9: chx /d0/cmds

Check to make sure all the modules you thought you loaded are in memory.

OS9: mdir (if not, load each again separately).

4) Load the module to be renamed, in my example this is...

OS9: load /d1/cmds/rma

5) Remove the (source) disk holding the module to be renamed, (in my example, the Development System disk) from /d1, replace it with a newly formatted OS9 diskette.

6) **OS9: makdir /d1/CMDS**

an unaltered copy of the module to be patched will remain in /d1/CMDS.

OS9: save /d1/cmds/rma rma

7) Determine preliminary info about the module to be patched...

OS9: ident -m rma

my "ident -m rma" told me this:

Header for: rma
Module size: $4EAF #20143
Module CRC: $F83DD9 (Good)
Hdr parity: $C9
Exec. off: $001A #26
Data Size: $190D #6413
Edition: $0B #11
Ty/La At/Rv: $11 $81
Prog mod, 6809 obj, re-en, R/O

The module name is rma, it is 4EAF hexadecimal (that's what the '$' means), or 20,143 decimal bytes long.

The numbers for module CRC and header parity are determined by an arcane formula known to ancient wizards and dedicated hackers. I am interested in the word (Good), and after I have patched, will look to see whether the numbers and status of the header parity and module CRC change. An executable module with bad CRC, or header parity error will not load into OS-9. A nice safety feature against loading corrupt modules.

This is the $0B (11th) edition of this particular module. If you have an earlier or later edition, or are patching a module other than "rma", what you see throughout this tutorial from my example will not match what yonr own attempts will show. But that shouldn't matter. 1 am showing a method, and the method should work on any executable module that you can load and save.

8) **OS9: ident /d1/cmds/rma**

The info you see for the module to be patched in memory (step 7), and the reference original in /d1/cmds is the same. It won't always be after patching, but is now.

OS9: attr /d1/cmds/rma

--e-rewr

The module "rma" is public execute, public read, owner read, owner execute, owner write. Eventually you will have to ensure the new module, after patching, has the same attributes as the old module had before patching. Make note of what you started with.

9) Examine chapter 3 of the Technical Reference Manual you received with OS-9 Level 2. Notice Figure 3.5 in the manual. There is a lot of information on that page; we'll need some of it.

We are going to dump the module to be patched to determine what the byte values referred to in Figure 3.5 contain. In the case of rma, or any long module, pressing the BREAK key will stop the dump when desired.

OS9: dump /d1/cmds/rma

(press BREAK when you have about a half screen full of information). ERROR #002 as you can see from Appendix A of your OS-9 Manual is KEYBOARD ABORT. That's exactly what you did, so that's what you can expect OS-9 to say. OS-9 also probably feels "you tap dance like a polar bear" for exiting so crudely. Yeah, well..., you've probably had a few words for OS-9 from time to time too.

We're interested in the first "block" of hexadecimal values produced by the dump. Try it again, this time redirect the output to your printer. Press the BREAK key when about 6 or so lines have been printed by this...

OS9: dump /d1/cmds/rma>/p

Here's what I get...

```
Addr   0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0 2 4 6 8 A C E
----   ----  ----  ----  ----  ----  ----  ----  ----  ---------------
0000  87CD 4EAF 000D 1181 C900 1A19 0D72 6DE1 .MN/....I....rma
0010  0BA6 A0A7 C030 1F26 F839 3420 3440 4F5F  .& '@0.&X94 @O-
0020  A7C0 5A26 F8AE E433 8430 8909 8D34 1031 '@Z&(.D3.0...4.1
0030  8D48 31AE A127 048B D8EE (etc.. etc...)
```

Any dump shows the hexadecimal values contained in a module with each value shown relative to the beginning of the module. At offset 0000+0 from the beginning of the module, i.e. at the beginning, the first byte contains the hex value 87. Offset 0000+1 contains the hex value CD. Offset 0000+8 contains C9. Offset 0000+E (where E is hexadecimal equivalent to 14 decimal), the value contained is 6D.

From the dump of MY rma example, can you tell where I would look to find where the module name is stored within the module? Each executable module stores it's own name within itself, then leaves a clue in the header telling where within the module it stored it's name. The module's name could be stored anywhere, but the clue is always stored in the same 2 bytes inside the module header.

Figure 3.5 tells us that at offsets 04 and 05 from the module beginning is the location where the module stores its clue (the pointer) to where the module name can be found. Bytes 04 and 05 contain the value $000D. Now 000D is not the module name, it is the pointer to where to go look to find the module name, the offset from the module beginning where the module name is stored.

If in MY dump, you find the value located at offset 000D from the module beginning to be the value 72, then you are correct. But what is "72"? It is not "seventy-two" a decimal value, because all values dumped are hexadecimal. So the 72 found is hexadecimal "72", shorthand as $72. The farthest right columns of the dump show what alphabetical equivalents are represented by each hexadecimal value dumped. A "." in the far right indicates "no alphabetical equivalent". From the far right of the dump we can see that $72 represents an ASCII coded letter "r".

Can we prove this? Sure. In appendix C (OS-9 Keyboard Codes) of your Manual, note that the hexadecimal value 72 in the NORM column represents the letter "r". You will now be able to see that the hex value found at offset 000E of the dump, the value $6D is ASCII code for "m".

Simple, right... so the value $E1 found at offset 000F in the dump represents an "a". NO, not in appendix C it doesn't. $E1 is not ASCII code for "a". $E1 is somewhere in the twilight zone, unmentioned in appendix C. *OS-9 chuckling away in the background mumbles...GOTTCHA AGAIN!*

A short course in hexadecimal tells me that ASCII code for "a" is $61, which in binary is 0110 0001. $E1 is 1110 0001 in binary. The difference between $61 and $E1 when expressed in binary is the furthest left bit, which is set (1) in $E1, but unset (0) in $61. Checking with Kevin Darling, (hopefully long to remain OS9UGPRES), he confirms that OS-9 uses the setting to "1" of the left most bit (called the most significant bit) of the ASCII code for the last letter in the module's name. It does this so that OS-9 knows the final letter of the name when it sees it, and need not read further.

When you find such a high order/left most/most significant (all mean the same) bit set in the original module name, do the same to the last letter of the new name you are patching.

Getting back on track, OS-9 knew my module's name is "rma". When it came to the ASCII code letter with the high order bit set, it knew this was the last letter of the name, disregarded the high bit when decoding, and recognized the "a".

10) Figure 3.5 also tells us that bytes at offset $02 and $03 contain the module size information. Since we are going to give our module a new name, and store it at the end, after lengthening the module to make space for it, we'd like to know where that end is. Offsets $02 and $03 of my dump example show the module size as $4EAF. Can you see this? Notice also this is the same information obtained by ident in steps (7) and (8) above.

OS9: dump /d1/cmds/rma

In this dump I am interested in the very last line, or perhaps the last two lines of the module, so won't interrupt with a BREAK. I find the last line of MY module dump to be....

```
Addr   0 1  2 3  4 5  6 7  8 9  A B  C D  E F  0 2 4 6 8 A C E
----   ---- ---- ---- ---- ---- ---- ---- ---- ----------------
  (leaving out all but the last line of the last block...)
4EA0   000D 000B 0009 0007 726D 6100 F83D D9    ........rma.x=Y
```

This confirms that the module is $4EAF bytes long. No way you say, it only goes to byte $4EAE, $4EAF is blank. True, but the module offset addresses are shown by dump, and offsets begin at address $0000, not $0001. So from $0000 to $4EAE is a total of $4EAF bytes including the first byte, the one at address $0000. It's like saying a count in decimal from 0 to 9 has the same number of values counted as a count from 1 to 10 decimal.

Notice something interesting, the module name seems to be repeated at the end of my module. I heard, but am not sure, that "C" compiled modules have that characteristic. Ignore it for now. I'm showing a general method, and your module may not have a duplicated name at the end. Besides our new name is LONGER than the old one so won't fit in the same space.

How much should I lengthen the module by? The new name is c.asm, that's 5 characters counting the "." dot. Playing it perfectly safe, so as not to mess up anything existing, I must lengthen the module by the 5 bytes for the new name, plus 3 MORE BYTES to hold the module's CRC check value. Figure 3.5 will tell you more graphically than literally, that the very last 3 bytes of a module are the CRC. Space for them MUST be reserved.. I need to lengthen my module by 5+3=8 bytes. UNNECESSARILY, I'll chuck in another byte of length just for good measure, and lengthen by 9 bytes.

```
11)$4EAF ----- original length
   +  9 ----- additional length
   -----
 =$4EB8 ----- new length.
```

Module size information is stored at offset 02 and 03 in the module header, so these values must be changed.

address	current value	new value	
$0002	$4E	$4E	(unchanged)
$0003	$AF	$B8	(requires change)

12) Since address $0002 need not change, we will build a patchfile that the modpatch utility command can use to change the value at $0003. We'll name this patch1. In this instance I will show the pressing of the (ENTER) key when needed. Prior to this, and hereafter, unless specifically shown, it was assumed.

OS:9 chd /d1(ENTER)
OS9: build patch1(ENTER)
? l rma(ENTER)
? c 03 AF B8(ENTER)
? v(ENTER)
? (ENTER)

OS9:list patch1

You will see...

l rma ---------- link to the module in memory named rma
c 03 AF B8 ----- change the contents of relative address $03 from $AF to $B8
v ---------- update and verify the module CRC.

CRC changes each time you alter anything within the module, so must be updated.

To implement the patchfile, call modpatch and tell it to find it's instructions in the patch1 file in drive /d1...

OS9: modpatch</d1/patch1

Modpatch shows each patch instruction it executes, then returns you to the OS9 prompt when complete. If you got an error from this command, such as "modpatch illegal command", did you use SCRED or some other text processor to build the patchfile? Go back and try it again with BUILD. Make sure the modpatch didn't occur successfully, and then modpatch just didn't like your end character. This happens to me with SCRED.

13) Having patched the module, you must now check your work.

OS9: ident -m rma>/p

Everything seems OK. ident likes the CRC, and the module is the new length. Now we'll save the lengthened module to the /d1 root directory, (not to CMDS), then unlink it from memory.

OS9: save /d1/rma rma
OS9: unlink rma
OS9: mdir

Note that rma is no longer in memory.

So far so good. right!

OS9: dir /d1 (Yes, rma is there.)
OS9: load /d1/rma

ERROR #236 (Check Appendix A page A-5 of your manual. BAD MODULE HEADER). You hear *OS-9 chuckling in the background...GOTTCHA AGAIN!* A self protective measure. OS-9 won't load a bad module. Your "v" command to modpatch verified overall module CRC, but did it update the header parity check?

OS9: ident /d1/cmds/rma

Notice the header parity check of the unchanged rma residing in /d1/cmds is the same as the header parity of the patched rma. Should it be? No it shouldn't, because when you changed the module size bytes at offsets $02 and $03, you changed values in the header, (see figure 3.5), and though modpatch with "v" updated the overall module CRC, it didn't change header parity. Now the header parity is wrong.

modpatch "v" updates and verifies overall module CRC, but does not change header parity. Likely it's assumed most people won't fool with the header when patching, so modpatch doesn't change header parity. ident is a little weak too. Try this...

OS9: ident /d1/rma

Looks perfect to ident, doesn't it? That's because ident doesn't check header parity, just like modpatch doesn't update it. AND IT DOESN'T WARN YOU.

OS-9 chuckling away in the background again....GOTTCHA!

"Verify" fills the need. Careful using the verify command though. It ALWAYS requires at least a redirected input.

OS9: verify</d1/cmds/rma

Header parity is correct and CRC is correct.

OS9: verify</d1/rma

Header parity is INCORRECT, but CRC is correct.

14) Fix the header parity.

OS9: chd /d1

OS9: verify u <rma >rmau

OS9: dir /d1

We have updated and verified the rma module found in the /d1 root directory, and told verify to save the updated module in the file rmau, also in the /d1 root directory.

OS9: ident /d1/cmds/rma>/p

OS9: ident /d1/rma>/p

OS9: ident /d1/rmau>/p

Compare the 3 idents above. Note the changes. Top one is the original rma. Second one has Header parity trouble (not updated). Last one has a new header parity value (now updated), AND perhaps unexpectedly, a new CRC as well. Whenever anything changes within the module, including a value in the header, the CRC has to be updated again. "Verify u" took care of this.

But why did an ident of "rmau" still show a module name of "rma"? It's just an example of the jar being relabelled, but marbles still the same color. Gets confusing when the file name and the name of the module it contains are different. Lets fix that and get rid of the copy of rma with bad header.

OS9: del /d1/rma

OS9: rename rmau rma

OS9: dir

15) Build a name change patch...

OS9: dump /d1/rma

We're interested in the very end line or last 2 lines of the module. This dump will show whatever random values occupied memory in the lengthened portion at the time you "saved" it in step 13. Since the values can be random, yours may not be the same as mine, so the method is important. Dump showed the end of my lengthened rma module to be...

```
4EA0  000D 000B 0009 0007 726D 6100 F83D D9FF
      (alpha equivalents omitted)
4EB0  FFFF FFFF FF12 B741
```

This dump was necessary because modpatch wants to know what you want to change a particular address' contents FROM as well as TO. In my case, though not necessarily in yours, the values to be changed FROM all have the value $FF.

Back in Appendix C we see the name we want to give the module has these ASCII codes...

c = $63

. = $2E

a = $61

s = $73

m = $6D (but we'll use $ED because high order bit of the last letter in the new name is set high if we found that in the old name).

Our modpatch file will be built so that we do not write on the last 3 bytes of the module, the CRC bytes. I will now count backward from the end of the extended module by (3 CRC bytes) + (number of bytes -5- in new name) = 8. Therefore I'll patch in the new name beginning at $4EB0. Yes, it's possible I could have at least overwritten the 3 old CRC bytes from before the module was lengthened, but let's keep it simpler.

My modpatch file called patch2 will be built like this...

OS9: chd /d1(ENTER)

OS9: build patch2(ENTER)

? l rma(ENTER)

? c 4EB0 FF 63(ENTER)

? c 4EB1 FF 2E(ENTER)

? c 4EB2 FF 61(ENTER)

? c 4EB3 FF 73(ENTER)

? c 4EB4 FF ED(ENTER)

? v(ENTER)

? (ENTER)

OS9: list patch2

16) Execute the modpatch... To use a modpatch command, the module to be patched must reside in memory.

OS9: mdir

Your right, we need to load our lengthened rma from /d1 root directory.

OS9: load /d1/rma

ERROR #214 NO PERMISSION - Read the fine print in the Commands Section of the Development System manual. The VERIFY command changed rma's attributes.

OS9: attr /d1/rma

----r-wr (missing the executable attributes in the patched module)

OS9: attr /d1/cmds/rma

--e-rewr (original unchanged module's attributes)

OS9: attr /d1/rma e pe

--e-rewr (now the patched rma can be loaded into memory)

OS9: load /d1/rma

OS9: modpatch</d1/patch2

17) The modpatch has occurred. Now check it out.

OS9: ident -m rma

You've just been misled by ident again... or have you? Best way to find out is to see if you can reload the module after saving it.

OS9: del /d1/rma

OS9: save /d1/rma rma

OS9: unlink rma

OS9: mdir (prove it's gone from memory)

OS9: load /d1/rma

What! It loaded this time without error #236. Why? Check figure 3.5 in the manual. The bytes you fiddled with this time weren't in the modules header, just in the general module. For this modpatch with a "v" is adequate. Since the header didn't change, the header parity didn't change, and didn't need correcting.

Still suspicious, verify with "VERIFY"

OS9: verify </d1/rma

Header parity is correct. CRC is correct.

If there was an error #236 on loading, possibly modpatch did the patch, but didn't update the CRC. Check the module with a dump, examine the last 2 lines of the dump. Did the ASCII codes you wanted get inserted where you wanted them? If so, just do a verify with "u" as shown earlier.

When you use verify just to check on the CRC and header parity, i.e. not update, then verify does not alter the attributes of the module.

With a verified patched rma loaded into memory with the new name at the end, it's time to tell OS-9 where to look for the new name.

18) Telling OS-9 where to find the name...

OS9: dump /d1/rma

See the changes you made at the end of the module. Now direct OS-9 to look here. Figure 3.5 tells me the module name offset pointer (the contents of bytes $04 and $05 in the module) must be changed to point to the new name.

address	old value	new value
$0004	$00	$4E
$0005	$0D	$B0

My completed modpatch file, patch3, after 2 previous examples, you can BUILD it unaided. When built mine lists like this...

OS9: list /d1/patch3

```
1 rma
c 04 00 4E
c 05 0D B0
v
```

OS9: mdir (you see "rma" currently in memory.)

OS9: modpatch</d1/patch3

modpatch executes showing each command executed

OS9: mdir

What happened to "rma"? It's still there but OS-9 sees it under its new name as "c.asm".

19) Think your done? Nope... check your work!

OS9: ident -m c.asm

OS9: save /d1/c.asm c.asm

OS9: unlink c.asm

OS9: dir /d1

OS9: load /d1/c.asm

ERROR #236 - remember you were back working in the header with that last patch. You know what to do, right...

OS9: chd /d1

OS9: verify u <c.asm >c.asmu

OS9: dir

OS9: del c.asm

OS9: rename c.asmu c.asm

OS9: attr c.asm

----r-wr (attributes need to be made executable again after verify)

OS9: attr c.asm e pe

---e-rewr

OS9: load /d1/c.asm

OS9: del /d1/rma

OS9: save /d1/cmds/c.asm c.asm

OS9: del /d1/c.asm

YOU'RE DONE.

Residing on /d1 root directory you have patch1 patch2 patch3 modpatch files to remind you how you did it. In the /d1/cmds directory you have an unpatched original "rma" and it's renamed twin, "c.asm".

One final step...

OS9: build /d1/log

? *NOTE: the c.asm module in CMDS is a patched "rma"

? *not the original "c.asm"

? (ENTER)

Hope you got something, besides eyestrain from this exercise. There are easier ways of doing the job, if you have the utility($) provided by others. BUT as well as achieving a name patch, this is also trying to be a tutorial. With a user friendly utility customized to do the job..., less thought/less learning.

It's taught me why more people haven't posted DETAILED explanations in the past. Takes a LONG time to write them. Hope more people will though. One things for sure, as detailed as you get, likely as not, you've left something out. Hope it wasn't anything important.

-- Al Semeluk;Delphi: OS9 Sig --

SPOKANE OS-9 CLUB?

What started on the PNW OS9 Echo (FidoNET) as comments about being left out of the incorporation plans of the Port O'CoCo Club, has resulted into the re-formation of the Spokane CoCo Club.

Suggestions from Chris Johnson and Randy Kirschenmann about suggested meeting topics and activities has inspired Steve and Judy Holts, Dave Gantz, Dennis Mott, Todd Stites and Richard Baysinger to give it a try! Spokane already boast as being the Hub for the PNW Echo, thanks to Dennis Mott, so they have a lot going for them. GOOD LUCK GUYS!

"SETBIT" Update

In last month's article ***File Allocation BITS Explained and 'set'***, I named several disk scanning utilities that would check the integrity of your disk media and report bad sectors. Then I explained that you would need to go into the File Allocation table and set the bit associated with that sector so that OS-9 would not write to that bad sector. I included a listing of Rick Adam's ***setbit*** utility that was designed for just that purpose. All of the programs/utilities given in last month's article were public domain. However, there is another way.....

Burke & Burke's ***File System Repack*** package includes a utility called ***CCHECK*** which combines the disk scanning utilities and the *setbit* utility. *CCHECK* performs a disk physical integrity check, identifies bad sectors found and if the "-a" option is used, it will set the allocation bitmap bits for all of the defective sectors found. All in one simple operation.

As a personal note: I was aware that Chris Burke's *Repack* package might contain an all inclusive utility, but I did not own a copy of the program and the printing deadline came before I could borrow a copy of the documentation to find out. Actually I was surprised no one called me on it! Hey you guys, WAKE UP!!! Anyway, thanks to Scott Honaker for lending me the documentation for the *File System Repack* by Burke & Burke.

POST SCRIPT:

It should be noted that the *File System Repack* package was designed for reformatting your Hard Drive to eliminate file fragmentation. Fragmentation of files on your drive slows disk access and over works your hard drive resulting in excessive ware and early retirement (**CRASH**)for your hard drive. Unfortunately, I have heard reports that it takes more than 24 hours to "repack" a 20 Meg Hard Drive. Public Domain to the rescue. Currently available on most OS-9 Bulletin Boards (and our PD Library) is a utility named ***STREAM*** which reformats your hard drive much faster. Example is a 20 Meg Hard Drive in 2 hours.

--Rodger Alexander --

Carl Kreider's comments on development of a new *AR*

I decided to try to clean up the *ar* mess before I start on my CD-ROM driver/manager. I still haven't received all the mods from the hackers, but those who didn't send me their modifications will just lose out. I've spent a day so far and it will be a bit more. I have one hacked version for a VMS port, one DOS port, two Unix ports, and a couple unreleased hacks that I will try to merge into one. None of these nice people had the foresight to contact me for the latest version of my source c es before starting, so of course I can't just compare them each against my source. Even worse, there is inbreeding among some of these other versions. So the task is not as easy as I thought it would be. To those waiting with bated breath (and those who are swearing under it) blame your friendly neighborhood hacker for the delay. The final upgrade is under way and there will be a release. I ain't saying when, but it is now a sure thing.

--Carl Kreider;Compuserv --

Let's Hear for OCN
(OS-9 Community Network)

by John Wight

LET YOUR IMAGINATION RUN WILD

Imagine a database filled with freeware, shareware and public domain files, utilities and programs for OS-9/6809 and 68XXX, and OS-9000. Imagine being able to download this software via modem or order it via mail.

Imagine a database filled with BBS systems, clubs and users groups from Canada, the United States and points across the sea.

Imagine a database filled with OS-9 vendors of both hardware and software for the OS-9 Operating System, offering support when you need it.

Imagine being able to search a database for hardware and software experts to help with your problems. Imagine this same database holding names of companies needing programming and hardware experts.

Imagine receiving a catalog of all the above for the price of a stamped, self addressed envelope . Or, if you have a modem, being able to go to your Regional OS-9 Library and download to your heart's content.

Along with all this, imagine receiving the OS-9 Conununity NetNews monthly, packed with articles and information on the OS-9 Operating Systems for the 6809, 68XXX and OS-9000.

Imagine no more! As a member of the OS-9 Community Network, all this and more is slowly becoming a reality for you.

The dynamic new OS-9 Community Network is now

(OCN Cont'd from page 8)

forming and would like to offer you a charter membership. To become a member, simply send your name, address, phone number and the type of OS-9 system you use, along with any comments or suggestions you might have to help us better serve you.

VOLUNTEERS WELCOME!

Greg Morgan
OS-9 Community Network Membership Coordinator
7859 Villa Park Drive
Richmond, VA 23228

Previews . . . .

CCStacks and Windows???

MICROSOFT WINDOWS FOR OS-9 ???

What's this? Variable window sizing, sliding bars, drag and drop functions, clipboard application? Maybe! The following messages by Daniel Hauck previews the features he has incorporated into his Graphics User Interface.

As far as the slide-bars/scroll-bars/bump-bars (they have many names) are concerned, there will be more improvements added to them such as a more effective "drag" function that will look like the MS windows function.

Many other the objects here are designed to resemble Microsoft windows. In fact, if you use the new library, it requires at least a 100Meg HD. !!KIDDING!! Just kidding. /grin. (I wonder if anyone caught the underlying knock on Microsoft hidden in there?)

At any rate, the product is almost ready for release. It has been slowed a little due to the fact that I got a job and am finally gainfully employed now and can afford to buy food and stuff. Anyway, I have less time to devote to the GUI library. But it will still happen!

-- Dan Hauck;FidoNET --

Another GUI !

CCStacks is a general interface for presenting data in a graphical (point and click) environment. From a main menu, you select a stack of information using a mouse. For example, from the main menu, using your mouse, you select a stack called "The Periodic Table". The main menu screen is replaced by a periodic table. To get information on a specific element, you use the mouse to select that element. An overlay screen pops up with information on that element such as its valences, properties etc.

Another stack is called "A Tour of the Planets". You select this stack, and the main screen is replaced by a picture of the solar system. Click on the planet that interests you, and the solar system screen is replaced by a picture of that planet.

Basically speaking, CCStacks is similar to HyperCard on the MacIntosh. It is a general interface which allows you to view "stacks" of related information, and jump from one screen to another. The general interface allows you to present information, be it graphical, text or sound, put buttons on the screen, popup windows and menu bars and select them with the mouse. From any one screen, you can jump to another screen depending on what buttons are enabled.

It takes advantage of the GFX2+ partial release package that Kevin Darling put out a few years ago. In addition, it also uses the Play command to play digitized sound files, and a program called VEFIt by Alan DeKok for loading and saving vef files. The only real restrictions that it has in terms of system requirements is that it requires a lot of storage space, depending on the size of the stack. Vef files are used for storing the graphics principally because of the speed with which they load. Squashed vef format is used to save some space. VEFIt will also allow you to overlay a screen on top of another one. The purpose of this is to allow you to do things like load in a picture of a country, superimpose (overlay) the borders, and superimpose the cities on top of that, for example.

Its actually a pretty neat program, but it does suck up the hard drive space. It can be run from floppy, but this limits the number of stacks that you have immediately available.

-- Daniel Hauck;FidoNET --

* Origin: Micro80 Computer Club of Ottawa BBS (1:163/306)

Club Activities Report

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

Incorporating:
Port O'CoCo vs. CoCoFEST?

Last month the Port Orchard CoCo Club voted to incorporate as the Port O'CoCo Color Computer Club.
I won't discuss the obvious benefits of incorporating right now, but if there are questions, the answers are easy to provide (e.g., limited legal liability, easier access to accommodations for the CoCoFest, possible industry contributions, etc.)

While the original plan was to incorporate a group reflecting all of those who participated in the PNW CoCoFest, a large majority of those present at the meeting in August wanted to make the incorporated organization more specifically a Port O CoCo Club.

At the time, I tended to think the name was not a major issue, and that the purpose of the group would be a more important consideration. After reflecting, I'm not so sure ... Donald Zimmerman (as always, the "spark plug" in this effort), has asked for input from "all" of the OS9 Community in the Pacific Northwest.

Several options exist for us ...

(1) Continue on the present course, with the Port O'Coco Club operating the CoCoFest for the benefit of the Color Computer/OS9 community ....

(2) Incorporate our other clubs/groups separately, work with the Port O'CoCo Club on a peer-to-peer basis.

(3) Persuade the members of the Port O'CoCo Club to change course and set up the corporation as a geographically inclusive organization, probably with a name that reflects that change ...

I'm not positive that any one of the above options is correct; however, certainly those need to be discussed. As it stands now, the name choice may already be a "done deal" -- but other choices are possible.

-- Chris Johnson;Tacoma BBS --

Bellingham OS-9 Users Group

The August meeting was spent in repairing one of the GIMIX computers. The September meeting was devoted to exploring the software on the GIMIX.

Before the GIMIX fun started, **Rodger Alexander** presented a "products update" regarding the need to copyright our Newsletter and Booklet. Letters from subscribers and messages read on the FidoNET OS9 echo indicate that *OS-9 Newsletter* articles and club produced booklets, such as the *Installing a CoCo into a PC* and the *OS-9 Level Two and the Color Computer-3* booklet are being photocopied and distributed without notification or permission. Even though the club doesn't earn a profit from these products, control of the distribution needs to be maintained or guarded, depending on how you look at it. **Mike Pleas** and **Rodger** have been checking into copyright procedures and have found that under the new copyright laws, proof of original or first publish evidence is all that is required. That a notice of copyright is not required by highly recommended so that those so inclined will know up front that they are breaking the law by copying and distributing our material without permission and are liable.

NOW ON TO THE FUN PART:

A simple *exam* program written in Basic09 would not run due to a proper statement that Basic09 was not accepting: **WHILE NOT(EOF) THEN**. According to the manual, it should have worked find, so we took ***BASIC09*** from the CoCo's version of OS-9 Level-II by Microware and installed it into the CMDS directory of the GIMIX, renaming the original program to *BASIC09.OLD*. We loaded the program and.....**Tah Dah!** It worked!

Wes Payne was the student SYSOP responsible for setting up the computer (way back in 1987) and had found it necessary to add extra UNIX like security to the standard OS-9 login system. We were hoping to add all of the members of the club to the password file. The idea was to have each of us set up our own directory and have near super user level of access on the system. BUT, due to Wes's "improved" security system, we couldn't log on. Wes had added a utility called ***ADDUSER*** which would automatically check the validity of a new user against current users status and then permit access to the shell. Unfortunately, there was no documentation on the program and we couldn't figure out the correct responses to the questions, so we were in affect KICKED OFF THE SYSTEM!

Our major objective was to review programs on the GIMIX and clean up the mess that had accumulated over the years. Each member logged onto the system as the super user and started to jointly view, dump and run files found on the execution directory. On one terminal we ran *Stylograph* word processor and typed a "helpfile" record of the files we found. A lot of useless junk was contaminating the system. Some programs written in Basic09 just designed to "bug" other users on other terminals. Even programs designed to simply crash

one terminal or the entire system. Teenagers! We also found a few real winners. Simple graphic games like "pong", "packman" and "tank", to a program written by **Wes Payne** called ***purge*** that prompts the user for a date from which to begin deleting all files from a directory. Great program but almost costly.....fortunately we were able to ESCape out of it.

We noticed that the practice of merging files to conserve memory was not practiced and a review of the manual indicated that the GIMIX version of OS-9 loads modules according to their actual size. And that there didn't seem to be a 64K memory block limitation. It was suggested by **Mike Pleas** that the club devote itself to studying the method by which these two benefits are accomplished and try to modify the CoCo version of OS-9 towards obtaining the same features.

-- Rodger Alexander --

Mt. Rainier CoCo Club

Our August meeting was held in the air conditioned comfort of the Fern Hill Library. We opened with a discussion of ways to attract and keep new members. It was agreed that getting back to basics was important to put new members at ease. A newcomer hearing about rebuilding OS9Boot files, the ins and outs of RIBBS, and executing I-code programs is not likely to return, let alone stay for the entire meeting. We also talked about ways to advertise our group including having someone write an article for "On Target Computing" which now has a format "for computer users of all computers". We are still looking for a volunteer for this item.

Donald Zimmerman showed us pictures of CoCo Fest II and some of the club meetings. He also had a copy of the Australian OS9 Newsletter which has been in publication for 6 years. Donald then gave us a demo of the Mid Iowa & Country CoCo disk magazine. It not only has interesting articles but some great art work.

Alan Johnson presented a demo of several spelling checkers. Two were for RSDOS users and 3 for OS9 folks. The group discussed the pros and cons of their favorite spelling checkers. The meeting was then adjourned until September 10th.

-- Alan Johnson --

NOTE: September meeting minutes were not available by press time.

Port O-CoCo

The September meeting was conducted by **Tom Brooks** due to the illness of **Donald Zimmerman**. The longest reaching moment of the meeting was the decision to file papers of incorporation as a non-profit organization. The matter was turned over to **Bud Melch**, a retired bank VP. We hope this will allow the group to grow and to protect itself from some of the pitfalls. It will be this non-profit organization that will design and sponsor the 3rd NW CoCo Fest June 25-26, 1993 in Port Orchard.

The highlight of the evening was the demonstration and presentation by **Gene Elliott**. Gene has been looking for the RIGHT TOWER. It has been like a search for the Holy Grail! Too narrow. Too cluttered. Power supply in the wrong place. Too much money. But through all of that Gene found the ONE tower. He had the CoCo3 out of the case and into the tower and attached to the MPI and ready to install the drives. One of the men commented that there was a rattling sound when he carried int the tower. Well, come to find out, the whole system was just laying in place. Nothing was attached! But it all fit and was a snap to put into place. The group approved the funds to finish up the tower for a complete floppy disk system. It should be ready to show off in October!

We will see if there is enough interest to hold another Tower Party as we did last year. There are about 4 people very interested in getting into a tower for their system.

Chris Johnson talked about the fact that the ***NW OS-9*** is going to be on the backbone of FidoNet. He has closed down his feed of that echo in anticipation of the new arrangement.

We also planned for the upcoming Computer Swap Meet. As it turned out only **Terry Laraway** and **Chris Johnson** were able to represent the CoCo community at Kent. Everyone else was either tied up or ill. Thanks, fellows, for filling in!

Another part of the meeting was looking at Gene's artwork and proposed calendar reminding us of our meeting dates. This will be a continued project for next month. Our meetings are always open to all. Our next meeting will be on October 19th.

-- Donald Zimmerman --

Seattle 68xxxMug

The September meeting was a special treat for those in attendance. **Rodger Alexander** brought a 6809 GIMIX OS-9 Level III Cr[illegible] with 4 Freedom 110 terminals and **Scott Honaker** [illegible]he club's own CoCo-3 in a brand new Tower!

The computer is approximately 19 x 30 x 9 inches and weighs approximately 60 pounds. The power supply consumes 1 square foot of chassis space, mostly transformer. The terminals are very sleek and only weigh about 16 pounds each.

Each club member present was given terminal time just to mess around with OS-9 just as it looks appears on the CoCo (except for no Graphics). With four terminals going hot and heavy, there was no noticeable slow down. That is, of course, until one of the members figured out a routine that consumed the cpu's time, bringing OS-9 to a screeching halt! There's one in every crowd, right?

When everyone finally got bored with the GIMIX, Scott and Rodger showed off the club's computer housed in it's new tower case.

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

Bellingham OS-9 Users Group

OS-9 Level Two and the Color Computer-3
Tutorial and *Hardware Hacker's Manual*. includes 5-1/4 Disk of (360K) of upgrade software **$5**

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

OS-9 Newsletter **$10/yr.**
12 monthly issues packed with OS9 "stuff" 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

[illegible]
19560 BURBANK BLVD. #D
[illegible]
03/06/93