ATTR Syntax: Attr filename [permissions] Usage : Examine or change the security permissions of a file Opts: -perm = turn off specified permission perm= turn on specified permission -a =

AUSTRALIAN OS9 NEWSLETTER

EDITOR :
Gordon Bentzen
8 Odin Street
SUNNYBANK Qld 4109

(07) 345-5141

Display s converted characters to standard output DSAVE Syntax : Dsave [-opts] [dev] [pathname] Usage : Generates procedure file to copy all files in a directory system Opts : -b make a system disk by using OS9boot if present -b=<path> = make system disk

APRIL 1989

error messages for given error numbers EX Syntax: ex <modname> Usage: Chain to the given module FORMAT Syntax : Format <devname> Usage : Initializes an OS-9 diskette Opts ; R - Ready L - Logical format only "disk name" 1/2 number of sides 'No of

AUSTRALIAN OS9 NEWSLETTER
Newsletter of the National OS9 User Group

EDITOR : Gordon Bentzen

HELPERS : Bob Devries and Don Berrie

SUPPORT : Brisbane OS9 Level 2 User Group.

Hi there ! Welcome to the ninth edition of the newsletter.

You know I'm surprised that some of you guys are not sick of seeing the same names at the bottom of all the articles all the time. Personally I am. I would really like to see some input from other members of the OS9 community. I mean, I heard from a fellow OS9'er the other day how he had trouble with his Tandy FD500 drive setup when he added another drive, and how it kept corrupting his disks. You know that might just be bugging some other people who have bought Tandy systems. How about it fellows ? Do you think you could put in some effort too? Just look for examples in the question and answer page of this issue for some of the types of things we can probably cope with.

I have now the pleasure of telling you of another newsletter that has sprung up with the demise of the Australian CoCo magazine. This publication is sponsored by Robbie Dalzell, and Garry of Port Noarlunga Sth. , South Australia. The newsletter is aimed at all aspects of all versions of the Colour Computer and will also I believe, include some OS9 stuff. Robbie tells me the newsletter is being produced bi-monthly, and it will cost you $8.00 to join up, and $12.00 a year for subscription. I'm very happy to see this effort being put into the CoCo community by these two gentlemen, and I wish them every success, and exhort all of you strongly to give them your support.

The address to write to is:
CoCo-Link
31 Nedland Cres.
Pt. Noarlunga Sth.
South Australia. 5167

Phone: (08) 386 1647 (Robbie)
(08) 386 1139 (Garry)

In this issue you will find part 2 of the database in C by me, and also a discussion of the trials and tribulations of attaching a hard disk drive to the Colour Computer. Gordon will prattle on some more about beginners OS9, and you will find the first of our (I hope) series of question and answer pages. I hope others of you will take the initiative and ask us some more questions, or if you feel you could have answered the question better yourself, why not drop us a line, and we'll include it in the next issue. The material doesn't have to be sent as a disk file from a word-processor (although we'd prefer it), but a few lines on paper will also suffice.

There is another request I'd like to make. All of you by now have heard or read about the large quantity of OS9 public domain software we have on disk. Well, most of that has either originated from the American OS9 User Group, or from local (Brisbane) users. If there are any users out there in the real world who have access to public domain OS9 software from another source, please let us include it in the Australian user group library, so it can benefit all the users. If you're not sure whether we've got it, or whether it is really public domain, give us a call or drop us a line, and we may be able to find out for you.

Well, that's all from me for now, enjoy this newsletter, and you'll be hearing from us again next month. Until then, happy (OS9) computing to all of you.

Regards,
Bob Devries.

A Hard Disk for your CoCo OS9 System?

One of the most sought after acquisitions for your system this year will be a hard drive, unless you are one of the lucky few who already have one. I decided that it would be timely to write an article about the hardware requirements, and the setting up of, a hard drive system, particularly in relation to the CoCo. (That is the system which more than 90% of our readers use.)

For a good general discussion about the available systems for the CoCo, I recommend that you read the article in the March 1989 edition of US Rainbow, pages 44 - 56 entitled 'Adding a hard drive to your system', by Marty Goodman.

Obviously my experience is limited. However, as I have just (in the last two weeks) finished setting up my hard drive system, I will try to answer some of the 'unanswered' questions.

Firstly, what hardware do I need?

After reading everything I could about the various systems, including ads in the US Rainbow, and taking price into consideration, I decided on the Burke and Burke XTC. The primary reason for choosing this system was because I already had a 20 meg. NEC drive and MFM controller available after upgrading my XT clone to a 30 meg RLL drive. I wrote to Frank Hogg Labs in New York, and ordered the Burke & Burke XT Controller with real-time clock option, and the Burke & Burke XT-ROM, to allow direct booting from the hard disk.

After what seemed an interminable delay within the postal system, my package from the US finally arrived. The system was all there, plenty of documentation, and even an Allen-Key to open the controller case. I had almost everything I needed to run my hard drive. However I still needed ......

Yes, you need more than just a drive, XT controller, and the B & B interface etc. For a start, you need a power supply. Great, I will strike a deal with Bob Devries and get him to build me a power supply. And while he's at it , he probably has a case just lying around to package the whole thing in!

Bob does a great job (in a big hurry, after all my incessant prompting), and we try to get the system up and running in time for our local OS9 Users Group meeting. We finally get the drive formatted and the new system configured. Suddenly, FIZZZ!!! .. no power supply. That drive takes more power than we anticipated. Oh well, no hard drive for the Users Group Meeting.

Next day ... Let's try a higher rated transformer. It seems to work fine. Hang on, is something burning? Boy does that thing run hot! You could fry an egg on top of the drive case. I know, ... lets fit a fan to the case... Here we go again. Off to the local electronics shop and purchase a slimline fan.

And so on.........

Finally it all goes together, works reliably (so far), and is looking good.

Seriously though, I wanted to try to convey to you some of the things which may happen which people don't seem to write about, particularly when they are trying to sell things.

Back to the hardware requirements. NOT ALL hard drives and controllers will work with the Burke & Burke system, but the main limitation seems to be the controller. Not all PC-compatible hard disk controllers are alike; some will not work with the CoCo XT at all, and some will not fit into the case. The controller case and the interface software are optimised for use with particular controllers. The manual states that the following controllers work well with the interface:

Western Digital WD1002-WX1 (MFM)
Western Digital WD1002A-WX1 (MFM)
DTC 5150CRH (MFM)
Western Digital WD1002-27X (RLL)
Western Digital WD1002A-27X (RLL)

Adaptech 2072 (RLL)
DTC 5160CRH (RLL)

Most popular drives will work, provided that they can be used with the above controllers. The following is a list of some drives which are mentioned in the XTC manual:

ATASI AT3046, AT3051, AT3085
CDC 9415-21&36, 94205-30&51, 94155-30,51,25,38,48,67&86
CMI CM5410, CM5616, CM6426, CM6426S, CM6640, CM6853
Evotek 5820
Microscience HH725
Miniscribe 1006, 1012, 3012, 3053, 3085, 3438, 3650, 3675, 8425, 8438
Rodime 101, 102, 103, 104, 201, 202, 203, 204
Seagate ST-506, ST-412, ST-419, ST-213, ST-225, ST-138R, ST-238R
Shugart SA604, SA606, SA612
Tandon TM252, TM262, TAN501, TAN502, TAN503, TM602S, TM603S

As well as these drives, I know that an NEC D5126 will work, because that is the type of drive which I am using at present.

When you have got the hardware together the next step is to put it all together. Because hard drives are an order of magnitude larger and faster than floppy drives, so too are the timing considerations of disk access. You will need to know things like number of tracks, number of surfaces and heads, stepping rate, park track, starting track for write precompensation, storage format (RLL or MFM) and such like in order to configure your drive. This means you will almost certainly need to have a data sheet for your drive.

The Burke and Burke system comes with software that will allow you to build a device descriptor that is particular for your drive. Their programme "Ddmaker" poses a number of questions about the above drive data in order to accomplish this. You will then need to select the device driver from a number (13) which are supplied when you order the system. Why so many? Well there are different drivers for different versions and levels of CoCo OS9, as well as versions for formattability and for single and multiple physical drives.

The formattability question is an interesting one! The theory is that when you initially configure your drive, you select a device driver that will allow physical and logical formatting of the drive. After you format your drive, you go through the process over again, this time selecting a driver that does not allow formatting. This then is the driver that you would normally use. You cannot (accidently or otherwise) format the drive using that driver.

Once you have constructed the device descriptor and selected the device driver you will need to generate a new boot (floppy) disk. You can use OS9gen to do this, simply by adding the new modules to your existing boot. Now supplied with the system is a utility called EZgen. This programme is a system for manipulating all types of merged files (of which os9boot is one) with options for ensuring that the output files are written in one block, reside in a particular place on the disk, provision for rewriting DD.BT on LSN0 of the disk for boot disks, and many other options. I feel that this programme is worth more than the hardware. Things like manipulating the order of modules within the os9boot file become a breeze.

There is a small section in the documentation about the not so famous "boot order" bug, which causes the OS9 BOOT - FAILED message, and how to rectify it. This becomes relatively simple using EZgen.

So after generating a new boot, by whatever method, you are ready to go. Well almost! If you are running Level 2, you will need to make a CMDS directory on the floppy, and copy to it Shell and Grfdrv. Level 1 users do not need to do this. Now we put the disk in floppy drive 0, and press the button. Hopefully the system boots up. We are now ready to format the drive. In my own system, I use the D.P.Johnston SDisk3 drivers and descriptors, and the sformat utility which is supplied with that package. Whoops, the hard disk system doesn't like sformat. So I have to dig out my original system disk, and use the standard format from there.

When you type format /h0, you will find that the format utility recognises< that you are attempting to format a hard drive, and asks you a number of questions which are not asked when formatting a floppy disk. You will need to answer yes to all these questions in the initial instance. Now is the time to get a cup of coffee, do the washing or take the dog for a walk ... it takes at least 25 minutes for a 20 meg hard drive. On a 20 meg drive, you will be formatting $99C0 ($0 to $99BF) sectors. That is decimal 39360 (512 byte) physical sectors, or 78720 logical sectors. The standard MS-Dos sector size is 512 bytes, but OS9 uses only 256 bytes. The logic in the B & B device driver looks after the translation.

After you have the drive formatted, there are still a few things to be done. When you boot your system, it would be nice to be able to have the initial directories set for /H0 and /H0/CMDS. There is a patch to the Init module described in the manual that will allow you to do this. Incidentally, when this patch is made, it has the added advantage of reading the Shell file from the hard drive, and you will find that this will make your boots just that much quicker.

If, like me, you have also purchased the XT-ROM for the controller, you will also need to patch the boot module of the kernel. Burke and Burke have assigned track 128 of the hard drive as the location for the kernel. The standard kernel resides on track 34 (that is where the DECB DOS command looks). I presume that this change was made because of the larger (32 instead of 8) minimum file segment size, which is also used by OS9 to determine the size (in sectors) allocated to directories when they are created.

Burke and Burke have also, when using their XT-ROM system, given us the ability to use an "alternate boot", using an alternate kernel. This alternate kernel is stored on track 129, and looks for a bootfile called altboot. This is quite handy when you need to be able to have module relating to a particular piece of software, or perhaps need to use a VDG /TERM device etc. For example, the popular game Kingsquest3 needs a specialized driver/descriptor combination which uses VIRQ's, and a specialized clock module. These modules cannot be successfully loaded into memory. I would not suggest that you waste space on your hard drive for such things.....

So there you have it, a hard drive for your system. When you have used this type of system, you can really appreciate the comments of the people who write about OS9 really coming into it's own with a hard drive. It's really true. Multiview really shines, and what's more it runs at a speed that IS useable. Profile, a database programme will amaze you. And of course, the hard drive does not use interrupts<, has a 2K buffer (within the controller) and therefore will really support multi user capacity.

If you have any further questions, or require further discussion, please don't hesitate to call me.

Don Berrie (07) 375-3236

oooooooooo0000000000ooooooooooo

The OS9 Operating System

Last month we published an article on the OS9 Kernel and Boot file and other "user" & "system" modules which was prompted by the submission by Ian Clarke. This month I will continue with some comments on the OS9 modules.

There are several types of modules each of which have a different use or function. To maintain the modular concept of OS9 a module must be position-independent so that OS9 can load or relocate it wherever memory space is available. Each module has three basic parts: a module 'header' a module 'body' and a 'cyclic-redundancy-check' value (CRC value). The header contains information about the module and its use including size, type, attributes, data storage requirements and execution starting address. The body contains the programme or constants which are usually pure machine code or BASIC09 compiled code, etc. The last three bytes of the module are reserved for the CRC value to verify the integrity of the module. The above is a very simple description of a module format. For a complete description, the OS9 technical reference manual would be a good place to start.

The reference manual will provide details of the modules we touched on last month, e.g. the file managers RBF, SCF, PIPEMAN, and the device drivers & descriptors. These are 'System' modules.

The executable 'user' modules are many and varied. The standard user modules supplied with Microware's OS9 are all those included in the CMDS directory, Format, OS9Gen, dir, list, etc. Many of these you would use on a regular basis, and no doubt you have added other executable modules to your CMDS directory to perform a specific function, perhaps some of your own creation or those from public domain sources. You may even merge the commonly used modules with SHELL so that they are loaded into memory when you 'boot' OS9. The level 2 OS9 for the CoCo 3 has a number of these modules merged with SHELL when you get it. Do a MDIR just after you boot OS9 and you will see a number of CMDS modules following 'shell'. These are the ones Microware has merged to maintain a block of less than 8k. If some of these modules are not used very often they can be replaced with something else of your choice.

The merged SHELL file can be split with 'tools' such as modbuster, modsplit, or if you do not have any of these the selected modules can be saved from memory and then merged together with any other executable module selected. The 'save' command syntax is: save pathname module-name.

```
e.g. save shell shell              (will save shell in the current data directory )
or   save /d0/shell shell          (will save shell in directory /d0)
     save /d1/copy_of_shell  shell (will save the module 'shell' from memory to the root directory of /d1 with the
                                    name of 'copy_of_shell')
```

Note: the 'save' module comes with the Level 2 developers pak. The 'save' from level 1 is in fact identical.
You will need to have all the modules that you want to merge with 'shell' in the current directory. The original merged shell in your CMDS directory could be renamed and then save 'shell' on its own to your CMDS directory as, say, shell.tmp, then merge selections e.g.

```
chd /d0/cmds
merge shell.tmp this that other etc etc >shell
```

The Execute attributes will need to be set.

```
attr shell e pe    <<Don't Forget This>>
```

My merged 'shell' takes up 3x8k blocks - 24k. This is probably now too big, but with 512k I seem to get away with it, so far anyway. With 128k it is probably a good idea to keep it down to one block.
Anyway, you decide on your needs.

Gordon Bentzen

oooooooooOOOOOOOOOOoooooooooo

QUESTIONS & ANSWERS

The following text may help those of you who have raised a question or two.

Question:

Wayne Patrick of Gympie Queensland writes .. "Now, regarding the manual that comes with OS9 Level 2, and the book by Dale Puckett and Peter Dibble, the first thing that the manual tells us is how to create a file. E.g. names, addresses, phone numbers etc. It tells you how to list the file by the use of the LIST utility. As a beginner, my question is, how do you add to this or other files?"

Answer:

This enquiry is not strictly a question related to OS9, but it does demonstrate one of the multitude of differences between OS9 and RS-DOS.

Under RS-DOS, the only editing function available from ROM is the normal BASIC line editing command, which among other things can only handle lines which start with a line number. RS-DOS, as it comes, is not designed to handle text at all. Rather, it is strictly a BASIC programming environment. In order to do any text handling, another programme needs to be run, either from the BASIC interpreter (slow), or in place of the interpreter.

OS9 on the other hand, is designed as an operating system. If text editing is your requirement, then you simply use the system to run an editor. If you just want to list a textfile, then there is a utility called LIST, which is designed purely and simply to list a file of text.

The answer to Waynes' question is of course, the use of a text editing program. The one supplied with the OS9 package is the macro text editing programme, EDIT.

EDIT is a line oriented editor, and its use is well documented in the manuals. Available with the OS9 Level 2 Development Pack, is SCRED, a screen oriented text editor. Other OS9 editors are available, both in the public domain (such as SLED), and copyrighted commercial programs (such as STYLOGRAPH).

Question:
I keep finding references to toolkits in the OS9 literature. Can you please tell me what these toolkits are and what they are used for?

Answer:
Toolkits are collections of utility programmes which have a number of common features. One would normally expect each utility within a toolkit to have a related function, say file manipulation, text manipulation or system modifications. It would also be expected that they would have a similar syntax and method of operation. Other names for groups of related programmes are Filter kits and Hackers kits.

Question:
I have a Tandy FD502 disk drive. How can I format, read and write to side B of the drive?
Answer:
OS9 does not treat the B side of a double sided drive as a separate drive the way Disk Basic does, but writes to each side alternately, track by track. To use the drive properly you should run the CONFIG programme that comes with the system disk, and choose the drive descriptor called D0_40D which stands for Drive 0, 40 tracks, Double sided. The other is to change the device descriptor in memory to double sided, 40 tracks, format a disk after patching the descriptor, and COBBLER the Os9Boot file to the new disk. The necessary modpatch file follows:

```
l d0
c 14 00 03
c 18 23 28
c 19 01 02
v
```

P.S. when you add the extra drive, don't forget to change the JP7 jumper on the FD502 drive to connect C-A so that the motor on is selected by the motor on signal of the controller, not by drive select. If this is not done, it will cause trouble with OS9 disk writing.

Question:
I have been unable to get PhantomGraph or Home Publisher to print because of lack of printer drivers.
Answer:
I have been able to patch the printer drivers for PhantomGraph to work with both the Epson printer (I have a BMC BX1000) and the Tandy DMP110 (I have to admit under duress to owning one of these). Let me warn you however.... the way in which the printer drivers were written, means that they are S L O W !!! They only print one pixel line at a time. This must also cause more wear on the one print head wire. Here are the patches, as usual using MOodPatch:

For the Epson printer we modify DmpIbm.drv. So first LOAD it into memory.

```
l dmpibm.drv
c 02cf 86 20
c 02d0 1b 0d
v
```

Now I used MRENAME to change the module's name in memory to dmpeps.drv, but you could just leave it at dmpibm.drv. Now after deleting the old driver (except if you used MRENAME) just use SAVE to save the changed module to the PG disk.

```
save /d1/cmds/dmpibm.drv dmpibm.drv
```

For the DMP110 printer we modify DmpTandy.drv. So LOAD it into memory, and use ModPatch as before.

```
l dmptandy.drv
c 019f 40 31
c 01a3 86 20
c 01a4 0d 0e
v
```

Here I used MRENAME and called it Dmp110.drv. Now save as for the Epson driver.

I have looked into the matter of the Home Publisher printer drivers, and I have found that the printer driver prn.Dmp105n works OK with the DMP110 printer. It does not, however use the full page width. Without re-writing the whole driver, (not me chaps) I think you'll have to live with that. I modified the prn.EpsonRX driver to work with my BMC printer.

P.S. The MRENAME command is available on one of the local Public Domain disks.

RdV

ooooooooooo000000000ooooooooooo

A Database in C.
by Bob Devries.

Here's part two of my database programme in C. No doubt there are some of you who would be able to pick holes in my code, and find better ways of doing them, but I feel it is good enough to present here as a sample of what can be done in C without a great deal of trouble or experience. Remember I wrote this as an assignment for my semester in C programming.

This is the first part of the real code of the database, and presents the command entry screen, and via calls to the other parts of the programme, will display the data entry array. Again, I must remind you that this will not compile successfully by itself, because the linker part of the compiler will not be able to find all the functions mentioned in this section of the code. It should, however, go through all the compiler sections up to the c.asm (RMA) pass correctly. If you do compile it you will get the message 'LINKER FATAL Unresolved references' and it will print a list of the functions it could not find in the standard library.

It probably is a good idea to compile at least to the assembler code, to see that you did not have any syntax errors in the code. To do this, type: CC1 database.c -r . This will now compile, but not link this part of the code, and you will be left with a file called 'database.r' in your data directory.

I mentioned last month the possibility of using this programme on OS9 Level 1 systems using Opak or Word-Pak screen drivers. I am happy to report that no changes are required to use the programme with the Word-Pak RS 80 column card, but some small modifications are required to 'ansi.h' to make it work on the Opak system. Here are the necessary changes for Opak:

```
eraselin()
{
        printf("%c",3);
}

revon()
{
        printf("%c%c%c",27,82,3);
}

revoff()
{
        printf("%c%c%c",27,82,0);
}
```

Remember these changes are for use with FHL Opak module ONLY! You will also need to split the 'options' line (printf("a=amend......) into two separate lines if you are using a screen of less than 64 characters with Opak.

Now here is the code for the database programme section one. I hope you people get some use and some educational value out of this tutorial. Please let me know of any problems with it, especially with the patches for Opak. As usual you can reach me via the national user group address.

Regards,
Bob Devries.

```
/* DATABASE A simple database in C             */
/* This programme has no sorting routine       */
/* but does re-use deleted records for new ones */
/*                                             */
/* Copyright (c) 1988 by Bob Devries           */
/* This programme and its source code files    */
/* may only be used for private use,           */
/* not to sold or used in commercial products  */


#include <stdio.h>/* get stdio header file */
#include "ansi.h"/* get local header file */

#define TRUE 1/* set some local defines */
#define FALSE 0
#define BS 8

/* this is the structure for each record in the database */

struct address {
        char surname[21];
        char firstname[21];
        char street[21];
        char city[21];
        char state[4];
        int postcode;
        } mail;

long lastrec;/* global var to hold last record number */

FILE *fp;/* file structure pointer */

main(argc,argv)
int argc;
char *argv[];
{
        char *strcpy();/* declare some functions */
        char *strncpy();
        int recno;/* and some variables */
        char db[29];
        char ch;

        pflinit();/* tell linker we are using longs */

        if (argc > 2)/* can't handle more than two arguments so */
                {       /* tell him how its done */
                        usage();
                        exit(0);
        }
        cls();
        if (argc == 1)  /* if no file given, ask for it */
        {
                cursor(10,12);
                printf("Input the pathname for the database : ");
                scanf("%s",db);
        }

        if (argc == 2)
                strcpy(db,argv[1]);
        if ((fp=fopen(db,"r+")) == NULL)/* try to open file */
                {
                        cursor(10,14);
                        printf("File does not exist. Create ? Y or N ");
                        ch = '\0';
                        while (ch != 'Y' && ch != 'N')
                        ch = toupper(getch());
                        if (ch == 'N')
                                exit(0);
                        if ((fp=fopen(db,"w+")) == NULL) /* didn't exist, create it */
                                {
                                cursor(10,15);
                                printf("Can't create %s.",db);/* oops can't do it */
                                exit(0);
                                }
                        }       /* ok the file is open for update here */

                        cls();
                        fseek(fp,0l,0);
                        recno = 1;
                        if (getc(fp) == EOF)/* if there's data display it */
                                {
                                cursor(10,23);
                                printf("Database Empty");
                                }

                        ch='\0';
                        while (ch != 'e')
                                {
                                fseek(fp,0l,2);
                                lastrec = (ftell(fp)/sizeof(mail));
                                scrnmask();
                                scrndata(recno);
                                cursor(5,14);/* print options in reverse video */
                                revon();
                                printf("a=amend p=prev n=next i=insert e=exit d=delete f=find \
m=match h=help");

                                revoff();
                                cursor(5,16);
                                printf("Your choice ?  %c",BS);
                                ch = tolower(getch());
                                cursor(10,23);
                                eraselin();
                                switch (ch)
                                        {
                                        case 'a':
```

```
                                        amend(recno);/* goto amend (edit) */
                                        break;
                                case 'p':
                                        recno--;/* back up one record */
                                        if(recno < 1)
                                                {
                                                cursor(10,23);
                                                printf("First record.");
                                                recno = 1;
                                                }
                                        break;
                                case 'n':/* forward one record */
                                        recno++;
                                        if (recno > (int)lastrec)
                                                {
                                                cursor(10,23);
                                                printf("Last record.");
                                                recno--;
                                                }
                                        break;
                                case 'i':/* insert one record in the file */
                                        recno = insert();
                                        break;
                                case 'd':/* delete currently displayed record */
                                        delete(recno);
                                        break;
                                case 'f':/* goto find record */
                                        recno = find();
                                        break;
                                case 'm':/* find another record from data */
                                        recno = match(recno);
                                        break;
                                case 'h':/* get help */
                                        help();
                                        break;
                                case 'e':/* quit back to OS9 */
                                        fclose(fp);
                                        break;
                                }
                        }
                        cls();
}
```

oooooooooo0000000000oooooooooo

Well folks, that is it for this month and we do hope that you find something useful in these pages.
We could have had something on the rest of this page if somebody (anybody) had sent us an article. It's over to you now.

Until next month, happy OS9ing.

These directories are on one of our 'local' public domain disks. These are available from us under the normal conditions.

```
 Directory of /d2   20:25:27
RAMDISK         WPROC           COMMS           DISASM
MISC            UTILS

 Directory of /d2/ramdisk   20:34:55
ramdisk         rammer2.ar      ramdisk.doc

 Directory of /d2/wproc     20:35:53
sled.ar         Sled.Bin        Sled1.c         Sled2.c
Sled3.c         Sled4.c         Sled5.c         Sledef.h
Sled.Docs       Sled.Hlp        Read.Me         Sled.c
format.b09

 Directory of /d2/comms     20:36:09
bigt            xcom9.newdoc    xcom9.bin       s1
sioacia         bigt.doc        bigtman.1       bigtman.2

 Directory of /d2/disasm    20:36:19
disasm.doc      disasm.data     disasm

 Directory of /d2/misc      20:36:34
clock.ar        clk             hclk            clk.doc
wclk            clk.windows     modpatch.tutorial

 Directory of /d2/utils     20:36:43
util            util.doc        dmode           mverify
files           ar09            files.doc       docs
mrename         ar09.doc
```

The following is a short description of some of the programmes on this disk.

included in these utilities is dmode
dmode works like tmode/xmode use for example dmode /d0 use dmode -? to get a list of options or use type type dmode to get syntax

----

mverify verifies a modules in memory
to use just type for example (mverify d0)

----

mrename renames a module in memory. Just type for example (mrename d0 s0)
That's all there is to it

----

ar09 is a file compresser and decompressor
you must use it to break all the .ar files on os9 bulletin boards
syntax: type ar09 <return> for command syntax
for example ar09 -x clock.ar will break away all the files in clock.ar

----