AUSTRALIAN OS9 NEWSLETTER

EDITOR :
Gordon Bentzen
8 Odin Street
SUNNYBANK Qld 4109

(07) 345-5141

DECEMBER 1989

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 = inhibit rms : d - directory file s - nc to owner w - write permit to own r - read permit to public pw - te permit to public BACKUP Syntax ge : Copies all data from one de ead error occurs s = single rites BASIC09 Syntax : Basic0 ge BUILD Syntax: Build filenar s from standard input CHD S nge working directory to specifi > Usage: Change execution directory to specified path CMP Syntax: Cmp filename1 filename2 Usage : File comparison utility COBBLER Syntax: Cobbler devname Usage : Creates OS-9 bootstrap file from current boot CONFIG Syntax: Config Usage: Create system boots and system disks COPY Syntax data from one fil E Syntax : Date [t Opts: t = specify ame> Usage : Check directory for wor isters -m = save of unused cluster nly -o = print <devname> {<devn x: Del [-x] filenam s : -x = delete x: Deldir directo yntax: Dir [e] [x] the file names y x=print executi] Usage : 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 using path as source i = indent for directory levels l = do not process b makdir command o num K ECHO Syn tandard output ED oriented text edito s text 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.

Well how did we go with that wish list called for in October's editorial? Not very well, is the short answer. You may recall that we offered some suggestions for content of this newsletter and sought your "wish list" of just what you wanted to see. I must say that the number of replies did not prompt me to visit the local hardware store for a larger mail box. There was one response though, by way of a note added to a subscription renewal, which I will repeat for you. "More C, less Basic09". Do you agree?

On a more encouraging note however, we are happy to report that membership of this User Group has grown to forty-seven (47) which includes a number of renewals, and some new members. To those members who have renewed we offer our sincere thanks for your continued support, and to those members who have joined this year we say WELCOME and hope that you find something of interest in each coming edition.

Just as a of matter interest, the membership last year reached a total of seventy-four, which we were very pleased with. Some of these members who have not renewed for this current subscription year were good enough to write and advise that they have made a change to other systems, for a variety of reasons. It seems that in spite of the lack of support by Intertan of both CoCo and OS-9, and their decision to drop the CoCo completely, there is still a good deal of interest in OS-9 by owners of this powerful little computer. The OS-9 operating system is widely used on the world scene for a variety of applications, and of course runs on a variety of hardware including computers supported by OS-9 68K. It may well be that we will change our computer system at some time in the future and toss out OS-9 for MS DOS. Perish the thought!

Although this newsletter reaches only the converted, or dedicated OS9ers, we feel that our efforts can only help to maintain interest in this great operating system, and we hope to present articles and news about all implementations of OS-9. Therefore we do need material from all members, at all levels of experience, in order that each of us can learn something from these pages.

Now, I must confess to not following all our newsletter articles with the enthusiasm each author might hope for. At a local user group meeting during October, Don Berrie asked "Who has tried the Macro facilities of EDIT covered in the October newsletter?" Much to Don's disappointment, I had to say that I had not tried to use what he had covered in his article. How about you? I did, however, compile the Database in C by Bob Devries, and learnt a few things along the way. The instructions that Bob presented looked foolproof, but then he was probably not counting on somebody like me who jumped in with C compiler in hand and little reference to the instructions. This sounds very familiar doesn't it? Anyway, after a couple of "unconditional abort" errors, and a little reading of the instructions, success was mine. If you have any problems with compiling the code presented in the series I am sure that Bob would be happy to help, but take it from me that all is in the instructions.

You will all know that the process of learning requires some help from others, but most of all it requires some effort by you, so please make the effort to help yourself and to help others. The fundamental aim of this newsletter, after all, is just to help each other.

This is not news, I hear you say, we already know all that. May I suggest then, that to stop me from continually pleading for articles and feedback, YOU send something to us. We want to publish your article, hint, tip, question or comment. So here is a challenge to every member to spend 41 cents and mail something for our next edition, which will be February 1990. We don't need a masterpiece (they are few and far between) we do need at least a comment from everybody.

We wish a Merry Christmas to all members, and trust that the new year brings to all, health, happiness and prosperity. Bob, Don & Gordon.

LINEATER

An OSK Basic programme by David Eaton.
OS9_6809 version by Bob Devries.

Here's a little programme from one of our 68000 OS9 users. The programme simply shows how screen accessing is done on the ATARI version of OS9, and could be made the basis of a learn-to-spell type programme. I have made the necessary changes and provided the CoCo version. It would be relatively easy to add to this piece of code to make it into a spelling programme. The speed of printing and erasing the characters on the screen can be changed by altering the FOR-NEXT loop variable y, which is currently set at 3000. An interesting point about the two different versions, is that they both run at the same speed, even though the ATARI computer runs at a much higher speed.
I hope one of our readers will take up the challenge to write a programme around this small piece, for the benefit of others in our group. Below are the two listings, first the OSK (ATARI) version, then the OS9 (CoCo) version.

Regards...Bob Devries.

```
PROCEDURE lineater
0000      (* When using, replace 'DIM ab' with 'PARAM ab' *)
0034      (* A word or sentence is shown for a short time and then eaten *)
0078      (* Version TDE 1.2 *)
0090      (* Adapted from programme in Nibble Vol. 2. No. 3 *)
00C6      DIM ab:STRING[60]
00DC      DIM x,y,length:INTEGER
00F4      PRINT CHR$(27); "E"
0106      PRINT "Enter the message you wish to show for a short time "
0142      PRINT \ PRINT
014A      INPUT ab
0152      PRINT CHR$(27); "E"
0164      PRINT CHR$(27); "Y"; CHR$(44); CHR$(50);
018C      PRINT ab
0194      length=LEN(ab)
01A2      FOR x=1 TO 20000\ NEXT x
01D4      FOR x=length TO 0 STEP -1
01FE        PRINT CHR$(27); "Y"; CHR$(44); CHR$(50+x); " ";
0232        FOR y=1 TO 3000\ NEXT y
0264      NEXT x
0276      PRINT
027A      END
```

```
PROCEDURE lineater
0000      (* When using, replace 'DIM AB' with 'PARAM AB'
002F      (* A word or sentence is shown for a short time and then 'eaten'
006F      (* Version TDE 1.2 (adapted for CoCo Basic09 by Bob Devries)
00AB      (* Adapted from a programme in 'Nibble' Vol 2 No 3
00DD      DIM AB:STRING[60]
00E9      DIM x,y,length:INTEGER
00F8      PRINT CHR$(12)
00FD      PRINT "Enter message you wish to show for a short time "
0131      PRINT  \ PRINT
0135      INPUT AB
013A      PRINT CHR$(12)
013F      PRINT CHR$(2); CHR$(50); CHR$(44);
014D      PRINT AB
0152      length=LEN(AB)
015B      FOR x=1 TO 20000 \NEXT x
0177      FOR x=length TO 0 STEP -1
018E        PRINT CHR$(2); CHR$(50+x); CHR$(44); " ";
01A4        FOR y=1 TO 3000 \NEXT y
01C0      NEXT x
01CB      PRINT
01CD      END
```

Resistor Calculator.
presented by Bob Devries.

Here's a handy little programme to work out what parallel or series combination of resistors are needed to produce an odd value that is not in the normally available range of resistors. Very handy for when you accidently connect your multimeter to the mains with it still switched to the ohms scale. The resistors in that part of a multimeter are usually very odd values.

This programme came from 'Silicon Chip' magazine, and was written by Steve Payor, who wrote it in GWBasic for an IBM PC. I converted it to Basic09 and made some minor cosmetic changes to it. So here is the code:

```
PROCEDURE Resistor
0000      BASE 1
0002      DIM E12(85):REAL
000E      DIM i,j,nearest:INTEGER
001D      DIM res,R1,Rmin,Rmax,tolerance:REAL
0034
0035      PRINT CHR$(12)
003A      PRINT  \ PRINT
003E      FOR i=1 TO 85
004E        READ E12(i)
0057      NEXT i
0062
0063      LOOP
0065        PRINT "Input desired resistance (1 to 999.9) ";
0090        INPUT res
0095        IF res=0 THEN
00A2          END
00A4        ENDIF
00A6        PRINT "Input desired tolerance (%) ";
00C7        INPUT tolerance
00CC        Rmin=res*(1-tolerance/100)
00E0        Rmax=res*(1+tolerance/100)
00F4        PRINT CHR$(12)
00F9        PRINT  \ PRINT
00FD        PRINT "Looking for a value between "; Rmin; " and "; Rmax             ; "."
0131        PRINT
0133        PRINT "Calculating..."
0145        PRINT
0147        nearest=25
014E        LOOP
0150        EXITIF E12(nearest+1)>=res THEN
0163        ENDEXIT
0167          nearest=nearest+1
0172        ENDLOOP
0176        FOR i=nearest-24 TO nearest
018B          FOR j=nearest-24 TO i
01A0            R1=E12(i)+E12(j)
01B2            IF R1>Rmin AND R1<Rmax THEN
01C7              PRINT USING "R7.2",E12(i);  \ PRINT " and ";
01E2              PRINT USING "R7.2",E12(j);  \ PRINT " in series,   "                ;
0206              PRINT USING "R4.1",(R1/res-1)*100;
0220              PRINT "%"
0225            ENDIF
0227          NEXT j
0232        NEXT i
023D
023E        FOR i=nearest+1 TO nearest+25
```

```
0256          FOR j=nearest+1 TO i
026B            R1=1/(1/E12(i)+1/E12(j))
0289            IF R1>Rmin AND R1<Rmax THEN
029E              PRINT USING "R7.2",E12(i);  \ PRINT " and ";
02B9              PRINT USING "R7.2",E12(j);  \ PRINT " in parallel, "                ;
02DD              PRINT USING "R4.1",(R1/res-1)*100;
02F7              PRINT "%"
02FC            ENDIF
02FE          NEXT j
0309        NEXT i
0314        PRINT  \ PRINT "Search complete."
032A        PRINT
032C      ENDLOOP
0330      DATA .01,.012,.015,.018,.022,.027,.033,.039,.047,.056,.068,.082
0388      DATA .1,.12,.15,.18,.22,.27,.33,.39,.47,.56,.68,.82
03E0      DATA 1,1.2,1.5,1.8,2.2,2.7,3.3,3.9,4.7,5.6,6.8,8.2
0434      DATA 10,12,15,18,22,27,33,39,47,56,68,82
045C      DATA 100,120,150,180,220,270,330,390,470,560,680,820
048B      DATA 1000,1200,1500,1800,2200,2700,3300,3900,4700,5600,6800,8200
04BF      DATA 10000,12000,15000,18000,22000,27000,33000.,39000.,47000.,56000.,68000.,82000.
0505      DATA 1000000.
```

As usual, the listing is from Basic09's LIST command so that you can check for errors in the lines by comparing the psuedo line numbers (actually number of bytes from the start). Have fun with it, and I hope it will get you out of trouble sometime.

Regards,
Rob Devries.

oooooooooo0000000000oooooooooo

CoCo Horizon

A new colour computer magazine is about to be released here in sunny Queensland!

Its name will be CoCo Horizon. It will probably be published on a monthly basis (depending on subscriptions and authors) and will cost approximately $3.00 per issue. It is hoped that the first issue will be available sometime in January 1990.

Readers who are interested in supporting this venture should contact:-

Andrew McQuirk
10 Deborah Court,
Slack's Creek.
Queensland. 4127.
Phone: 07 2082966

oooooooooo0000000000oooooooooo

THE FUTURE OF THE COCO3

A "Philosophical" viewpoint by Nickolas Marentes

THE FOLLOWING ARTICLE IS BASED ON MY OWN OBSERVATIONS OF THE COCO3 MARKET AND COMPUTER INDUSTRY WORLDWIDE. I HAVE NOT BEEN IN CONTACT WITH TANDY CORPORATION FOR ANY OF THE INFORMATION OUTLINED IN THIS ARTICLE. FEEDBACK TO THIS ARTICLE IS QUITE WELCOME AND SHOULD BE DIRECTED TO COCO-LINK MAGAZINE.

Anyone who owns a CoCo3 would surely be deeply disappointed by the recent "chop" of the CoCo3 by InterTan. I, as one who has spent many hours developing software and hardware for it am more so. But I do believe that there are two sides to this. The side we are all familiar with, that Tandy has pulled the rug on its loyal CoCo3 customers. The other side is that this is the way the market is heading. Before going into more details, lets look at the CoCo3 market prior to the "tragedy". Let's look further than just our little confined CoCo3 world and see what's happening in the rest of the computer world.

Commodore are on the verge of chopping the C-64 and C-128. A C-64 with a disk drive retails for around the $600 mark. Anyone wanting to spend this much money on a computer which they will primarily be playing games on may as well fork out the extra $300 and get an Amiga 500 pack giving them 512K of Ram, 880K high speed drives, 4096 color graphics, stereo sound and access to fantastic software. Anyone buying a C-64 solely for games may as well spend only $200 for a Nintendo games console and be done with slow loading procedures.

Someone who is not interested in just playing games but wants to do some serious computing will generally spend a bit more to get a more powerful computer.

What I am trying to highlight here is that the low end computer market is dividing into two distinct areas. The first is the very low end game playing market. The other is the upper-low end computer enthusiast (play-the-occasional-game) market. The very low end market is heading towards the games console area. If games is all they want, then games is all they need. No need to know about keyboard commands, loading procedures, programming languages etc., just slap a cartridge in and play. Games consoles are cheaper to produce and therefore cheaper to buy and to many, a low price is the right price.

The upper-low end user wants big power for lower bucks. Half a meg, one meg and eight meg of Ram. A fast 16 bit processor and fast hardrives if the money can reach that far. They want a large and varied software base. IBM compatability is a big thing because it brings many users together. Software developers love it because it means more customers, computer developers love it for the same reason.

The CoCo3 as many other computers of the era, fall mid way between these to areas. It plays games for a reasonably low cost but can be expanded to run powerful operating systems. Customer demands nowadays are more specific. For one who just wants a games unit, the CoCo is expensive and the variety of games is small. For one who wants a power system, the CoCo needs to be expanded. By the time a CoCo is equipped with a disk drive, 512K Ram, operating system and RGB monitor, it isn't too much cheaper than a PC, Amiga or Atari ST system. This is exactly the same problem that the C-64 and C-128 face (although the C-64 market has more momentum than the CoCo market and will stay around a little longer).

Well, we see that the CoCo3 is without a solid market area and in this light, we can understand why InterTan have given it the chop. Enough of the grim facts and let's look at what the future holds.

If Tandy were to do a CoCo4, what shape or form must it take to recapture a market position. From the last few paragraphs, we can conclude that it must either go upmarket by adding extra memory, built in drives, be faster and more professional and make OS-9 the standard operating system OR go down market and make it into a games console.

The upmarket approach would fail before it even started. Many companies have tried to go their own way, avoiding the IBM PC standard. Most of these companies are not heard of anymore. The IBM standard is a big standard. Commodore, Atari, Amstrad even Tandy have seen this. All these companies have gone IBM compatible. Commodore and Atari have continued their quest with their own standard and have achieved reasonable success but the IBM standard still dominates. The CoCo would not have a chance and even if it did, it would be too big a risk for a company like Tandy to take on.

The future of the CoCo3 is in the games console area. Now before you start swearing and make statements like "I don't want a games computer!", let me explain how I (if I was in charge of Tandy) would go about a CoCo4.

Imagine a box which has a cartridge slot in the top for game packs, connectors in the side to take two (or four!) joysticks, several buttons for selecting and starting games and video connectors at the back for a standard TV and RGB monitor. This would be the Tandy CoCo game console unit. It would basically be a CoCo3 with no keyboard, no BASIC ROM, 128K RAM plus a 3 channel sound chip (as used in many games consoles). This would be cheap to produce and low cost to buy. Now what if this box also had a duplicate cartridge connector as used in the CoCo3 but placed at the back of the unit. What if it also had a connector at the front of the unit marked "keyboard". Tandy could sell an add-on kit consisting of an Extended Color Basic Rom cartridge and separate detachable keyboard which when plugged into the "box" transforms it into a CoCo3! Bring out a new (or re-introduce the old) disk drive unit with Disk Extended Basic and plug it into the back connector and there you have a CoCo3 disk system!!

Now let's get really creative! How about also offering an OS-9 Level 2 ROM cartridge !!! Maybe even with Multi-Vue!!!!

I feel that this design would keep existing CoCo owners very happy and will bring together a lot more CoCo users. Users who first bought the system as a games console and later decided to upgrade it to a computer and enter the world of CoCo computing! The more I think about it the more great ideas I come up with!

Well, time to get off my soap box and return to the real world of gloom where the only thing that keeps us going is the hope that the CoCo4 become reality.

THIS ARTICLE IS BEING SENT TO TANDY CORPORATION IN FORT WORTH TEXAS IN THE HOPE THAT THEIR ENGINEERS AND MARKETING DEPARTMENTS ARE "STIMULATED".

oooooooooOOOOOOOOOoooooooooo

Let's Clear the Slate.
3 Different Ways.

by Bob Devries and Don Berrie.

I'll bet many of you have cursed OS9 for not having a CLS in its repetoire of available commands. Well, here are now three different versions for you to look at and use. They all do the same thing, that is, send to the screen the code(s) necessary to clear it.

While Don and I were discussing another programme used in this newsletter, Don showed me a CLS command which he had written for his ATARI 520 with OS9/68000. He had written it in 'C', and it was an astounding 4698 bytes long ! I said to him that I thought we should write one in 68000 assembler. There was only one snag. Neither of us is on speaking terms with 68K assembly language, even though both of have a computer with that CPU in it.

We did some investigation and experimentation, and after some in depth study of the manuals, we came up with the programme below. Also shown are two different versions for OS9/6809, one for the RMA and RLINK programmes, and one for ASM. Of course, the 68K version is compiled with R68 and linked with L68.

Here's the source code for 68000/OS9

```
        nam cls

Edition equ 1                   first edition
typelan set (Prgrm<<8)+Objct    set program type and language
attrev  set (ReEnt<<8)+1        set attributes and revision

* next comes the programme section

        psect cls,typelan,attrev,Edition,0,start

        use /h0/defs/systype.d  definitions files to use
        use /h0/defs/oskdefs.d
```

```
* next is the initialised data and stack area

        vsect

        ds.b 255

        ends

start   moveq #1,d0             select stdout
        moveq #2,d1             printing two chars
        lea   clscmd(pc),a0     point to chars to print
        os9 I$Write             do write command
        moveq #0,d1             clear the error number
        os9 F$Exit              and exit
clscmd  dc.b 27,69              $1B,$45 clears the screen on ATARI OS9/68K
        ends                    end of programme
```

To assemble this, we used the following command lines:-

```
R68 cls.a -o=/R0/cls.o
L68 cls.o -l=/H0/LIB/syslib.l -o=/R0/cls
```

This resulted in the binary file 'cls' to be stored in the ramdisk, a mere 116 bytes. Because of its length (or lack of it), it loads much more quickly than the C version, and also works more quickly.

Here is the OS9/6809 version for the RMA assembler and RLINK linker.

```
        nam Cls
Edition equ 1
Prgrm   equ 16
Objct   equ 1
ReEnt   equ 128

typelan set Prgrm+Objct
attrev  set ReEnt+1

        psect Cls,typelan,attrev,edition,0,start

        use /D1/LIB/os9defs.a

        vsect
        rmb 255
        endsect

start   lda #$01                select stdout
        ldy #$0001              print one char
        leax clscmd,pcr         point to char to print
        os9 I$Write             go print it
        clrb                    clear error if any
        os9 F$Exit              and exit
clscmd  fcb 12                  $0C clears the screen on CoCo
        endsect                 end of programme
```

Assembly is done like this:-

```
RMA cls.a -l -o=/R0/cls.o
RLINK cls.o -l=/D1/LIB/sys.l -o=/R0/cls
```

Now here's the code for those of you with OS9 Level One, or those who don't have the RMA assembler. You'll notice that the code is written differently to the previous two.

```
       nam Cls
       ifp1
       use /dd/defs/os9defs
       endc

Edition equ 1
Typelan set Prgrm+Objct
Attrev  set ReEnt+1

       mod ClsEnd,ClsNam,typelan,attrev,start,ClsDat

ClsNam  fcs /Cls/
       fcb 1

       rmb 255

ClsDat  equ .                  the period in this line must not be left out

start   lda #$01
       ldy #$0001
       leax ClsCmd,pcr
       os9 I$Write
       clrb
       os9 F$Exit
ClsCmd  fcb 12
       emod
ClsEnd  equ *                  asterisk here important!
```

To assemble this version the command line is like this:-

```
ASM cls.a l,o=/R0/cls #12k
```

Note the #12K here is necessary else the assembler quits with symbol table full error.

Well, there you have it. You'll notice that in all the examples I've used the R0 ramdisk as my temporary storage. You may of course use any drive and directory. This comment also applies to the pathlist for the libraries and other definitions files. You'll need to change them to suit your system. For example, in the last version, I used '/DD' as the drive specifier, which is the correct way for OS9 level two, and denotes the 'default drive' (read the article about that elsewhere in this issue), you may change that to /D0 or /D1 or whatever to suit the situation.

For those of you who are using the Cls programme which appeared in 'The Rainbow' of November 1986 page 204, there are two errors in the source code. The first one in line 18, is not fatal, and is rectified by replacing the '*' with a '.'. The second error causes the programme to modify itself, which is, of course, quite illegal in OS9. To change this change line 22 to read ... leax char,u ... not ,pcr as it is now. This also results in the code being two bytes shorter.

Well, have fun with that, OS9 level two users may want to include this command with the 'Shell' so that it will always be in memory.

Regards,
Don Berrie
Rob Devries

An explanation on windows for your system disk
by Rob Unsworth

What I am going to do is explain how to configure windows for your system disk. Windows that are loaded with your boot file and do not require wcreate or display sequences in your startup file.

The listing below is a copy of the startup file which was being used by one of user group members, who gave a copy of his system disk to a new member, and I have since noticed a third member with the same problem. Now you know why I am writing this article.

```
iniz r0
* start system time from keyboard
setime </1
format /r0 </1
iniz w1
iniz w2
iniz w7
wcreate /w1 -s=02 0 0 80 24 00 01 01
wcreate /w2 -s=02 0 0 80 24 02 07 07
wcreate /w7 -s=02 0 0 80 24 05 02 02
shell i=/w1&
shell i=/w2&
shell i=/w7&
dates
```

As you can see w1, w2, and w7 are going to be available to the user each time the system is booted. These three windows are being loaded in with the boot file, probably with the default descriptors that came with the system master disk. As the windows are what we are discussing here I will only refer to that part of the startup file which relates to windows.

The three iniz commands are not really necessary, unless you want to reserve memory for each window, as the system will allocate memory dynamically (as needed) to each window.

The wcreate command will be loaded, then /w1 will have it's paramaters changed to an 80,24 text screen with a white foreground a blue background and blue border. Then wcreate will be unlinked and then loaded again for /w2. The changes to /w2 will be made then it will be unlinked and loaded again for /w3.

Get the picture.....A lot of wasted time.

There is a simple solution, change the default paramaters of the windows in the boot file. To do this requires a little understanding of what paramaters need changing and where they are located. I will now endeavour to give you the means to obtain that understanding.

Colour codes		Screen type code
00	White	1 = 40 * 24 text screen
01	Blue	2 = 80 * 24 text screen
02	Black	3 = not used
03	Green	4 = not used
04	Red	5 = 640 * 192, 2-colour graphics
05	Yellow	6 = 320 * 192, 4-colour graphics
06	Magenta	7 = 640 * 192, 4-colour graphics
07	Cyan	8 = 320 * 192, 16-colour graphics

The codes above are the ones we require along with the start location of the window, this being the column and row of the first character space. You will also have to decide on the size of the window, do not get this confused with screen type, as you can have an 80 column text screen (screen type code 2) that is only 40 columns wide.

Now choose.... Screen type, Start location, Window size, Foreground, Background and Border colours.

The next step is to see where these changes need to be made. Then we can look into how to make the changes. The list of numbers below is a dump of the W1 module supplied with your system disk. What we are looking for is to locate the bytes that need to be changed to produce the window we want when we boot OS9. The byte for the number of columns is at $2c and you will find that the number of columns is $1b (27). The rows are at $2d which shows the number of rows at $0b (11). The screen type is located at $30 and is $01. The start location of the window can be found at $31 (column) and $32 (row) and in the listing for W1 they are both $00 indicating that the first character space is the top left hand corner of the screen. The only thing left is the colours, these are located at $33 (foreground) which is $02 (black), $34 (background) which is $00 (white) and $35 (border) which is $04 (red).

```
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 0043 0036 F181 B000 3800 3B03 07FF  .M.C.6q.0.8.;...
0010  A11A 0000 0100 0101 0000 1808 180D 1B04  !...............
0020  0117 0305 0807 8000 0036 0000 1B0B 0101  .........6......
0030  0100 0002 0004 57B1 5343 C643 4333 49CF  ......W1SCFCC3IO
0040  7CD1 80
```

Now to change this window descriptor to be the same as the one in the wcreate w1 command in the listing of the startup file. A 80 column by 24 line text screen with white lettering on a blue background with a blue border.

```
OS9:load modpatch
OS9:modpatch
l l w1          *links to the w1 module in memory
c c 2c 1b 50    *changes $2c from $1b (27) to $50 (80) column
002c 1b 50
c c 2d 0b 18    *changes $2d from $0b (11) to $18 (24) rows
002d 0b 18;
c c 30 01 02    *changes $30 from $01 (40) to $02 (80) screen type
0030 01 02
c c 33 02 00    *changes $33 from $02 (black) to $00 (white) foreground
0033 02 00
c c 34 00 01    *changes $34 from $00 (white) to $01 (blue) background
0034 00 01
c c 35 04 01    *changes $35 from $04 (red) to $01 (blue ) border
v               *to verify and update the modules CRC
<control><break>
```

The W1 module in memory has now been changed. Any of the other window modules can be changed in the same way. You can now create a new system disk using cobbler or OS9gen, and eliminate those wcreate commands from your startup file.

I hope this has made the configuring of windows a little clearer. A helpful hint that will save you a lot of hassles, if you have booted OS9 with wcreate in your startup file and they are the windows you want each time you boot up. A cobbler to a newly formatted disk will make the changes for you. Your startup file should then look like this....

```
Iniz r0
setime </1
format /r0 </1
shell i=/w1&
shell i=/w2&
shell i=/w7&
dates
```

Isn't this a lot neater than what we started with ?....Happy computing.

Rob Unsworth

HOW TO BECOME A MILLIONAIRE

If I knew that I would be one myself, don't you think?

But I can tell you this: The more money you are able to save over as long a period as possible at the highest possible interest rate, the greater is your chance of becoming a millionaire! But you knew that already, right? So what you (and I) really need to know is how to calculate just how much and for how long, etc to save.

Or another situation: The Government is talking about us all providing our own old age pension in the form of Superannuation, but would you be better off just simply saving up your hard earned cash and control your own investment and save paying some Fund Manager (a company such as AMP Society) large fees. Of course, in that case you need a lot of self control so that you don't spend the money prematurely.

The factors to consider are the following: How much money do you want to (and are you able to) save? Since we live in times with constant inflation you should consider increasing your savings each year by at least the rate of inflation. Since you don't know what the inflation rate will be in the future, you simply have to make a guess. Another factor is how high an interest you can get on your savings. In an ordinary savings account the interest rate is fairly low, so you might look for better investment accounts, but remember the general rule that the higher the potential return on your savings the higher the risk on the investment; in other words the less safe your money, so you need to balance one against the other. Yet another factor is your marginal tax rate, ie the rate of tax you pay on the top Dollar you earn. For instance, right now the highest tax rate in Australia is 47%, but you may pay a lower rate, it depends on your income. Do remember that as your savings grow and your interest earnings grow with it, that may cause you to come into a higher tax bracket, even the top tax bracket. The final factor to consider is the length of the period of saving and for how long the money is left to earn interest before it is withdrawn.

So I wrote a BASIC09 (excellent programming language that) program which I called SAVINGS CALCULATOR. Below is the program listing and a sample calculation. Type in the program and try out a number of different savings plans that suit your circumstances. I think you may be staggered by the potential of your savings.

Disclaimer: I am NOT an accounting professional. This program may be helpful as a guide, but for professional advice please speak to an accounting professional.

Happy computing!

Your fellow CoCoNut,

Ole Eskildsen

*** S A V I N G S C A L C U L A T O R ***

 The developer does not accept any liability as the result of using this program. If in doubt seek professional advice. This calculator helps you determine the result of different savings plans. First you enter the initial annual savings contributions, e.g. $100 per month = $1200 per annum. Then you enter the percentage you intend increasing your savings by each following year. As a guideline you should at least increase your savings by the expected inflation percentage. You are then asked to enter the expected annual interest rate you expect to receive on your savings. Next, the marginal tax rate, which is the rate of tax at the top end of your income bracked at which the interest earned on your savings will be taxed. And finally, for how many years do you want this savings plan to run? Try several different variations! If you have a printer you can also print the result.

```
INITIAL ANNUAL CONTRIBUTION: 3600
% INCR IN CONTRIB/YR: 10
EST. INTEREST RATE: 13.5
MARGINAL TAX RATE: 41
TERM IN YEARS: 20
```

```
PRINT? (Y/N): N

INIT ANN CONTRIB: 3600.  % INC 10.  INT RATE: 13.5  TAX % 41.  TERM: 20.

YEAR       CONTRIB       INTEREST       TAX PAID   ACCUMMULATED
 1.        3600.00         486.00         199.26       3886.74
 2.        3960.00        1059.31         434.32       8471.73
 3.        4356.00        1731.74         710.02      13849.46
 4.        4791.60        2516.54        1031.78      20125.82
 5.        5270.76        3428.54        1405.70      27419.42
 6.        5797.84        4484.33        1838.58      35863.01
 7.        6377.62        5702.49        2338.02      45605.10
 8.        7015.38        7103.76        2912.54      56811.70
 9.        7716.92        8711.36        3571.66      69668.32
10.        8488.61       10551.19        4325.99      84382.14
11.        9337.47       12652.15        5187.38     101184.38
12.       10271.22       15046.51        6169.07     120333.03
13.       11298.34       17770.24        7285.80     142115.81
14.       12428.18       20863.44        8554.01     166853.42
15.       13670.99       24370.80        9992.03     194903.18
16.       15038.09       28342.07       11620.25     226663.10
17.       16541.90       32832.68       13461.40     262576.28
18.       18196.09       37904.27       15540.75     303135.89
19.       20015.70       43625.47       17886.44     348890.62
20.       22017.27       50072.57       20529.75     400450.71
   ------------------------------------------------------------
         206190.00      329255.44      134994.73     400450.71



PROCEDURE savings
 0000
 0001      REM PROGRAM: SAVINGS
 0014      REM Date Written: 2 Dec 89
 002D      REM Programmer:   Ole Eskildsen
 004B
 004C      REM Copyright (C) 1989 by Ole Eskildsen, All Rights Reserved
 0087
 0088      REM This program has been donated to the public domain for personal
 00CA      REM use only.  It may be freely used and shared with others for non-
 010D      REM commercial, personal use provided no fee other than the cost of
 014F      REM the media is charged and on the condition that this copyright
 018F      REM notice is left intact.  The developer does not accept any
 01CB      REM liability as a result of using this program.
 01FA      REM If in doubt, seek professional assistance.
 0227      REM
 022A      REM Ole Eskildsen, 11 Monarch St, Kingston QLD 4114, Australia
 0267      REM Telephone: (07) 209 4322
 0282      REM ------------------------------------------------------------------
 02C5
 02C6
 02C7
 02C8 10
 02CC      DIM PRINTER:BYTE
 02D3      DIM AP,IR,TR,YR,IC,AC,TP,B,TI,TA,TT,ANCNT:REAL
 0306      DIM LINE:STRING[60]
 0312
 0313      AC=0 \TP=0 \B=0 \TI=0 \TA=0 \TT=0 \ANCNT=0
 034B      LINE="------------------------------------------------------------"
```

```
038E
038F 70   PRINT CHR$(12)
0397
0398      PRINT TAB(15); "*** S A V I N G S  C A L C U L A T O R ***"
03CD      PRINT ""
03D1      PRINT "Copyright (C) 1989, all rights reserved, by"
0400      PRINT "Ole Eskildsen, 11 Monarch St, Kingston QLD 4114."
0434      PRINT "Donated to the public domain for personal use only."
046B      PRINT "The developer does not accept any liability as the result of"
04AB      PRINT "using this program.  If in doubt seek professional advice."
04E9      PRINT "This calculator helps you determine the result of different"
0528      PRINT "savings plans."
053A      PRINT "First you enter the initial annual savings contributions,"
0577      PRINT "e.g. $100 per month = $1200 per annum."
05A1      PRINT "Then you enter the percentage you intend increasing your "
05DE      PRINT "savings by each following year.  As a guideline you should  "
061E      PRINT "at least increase your savings by the expected inflation "
065B      PRINT "percentage.  You are then asked to enter the expected annual"
069B      PRINT "interest rate you expect to receive on your savings."
06D3      PRINT "Next, the marginal tax rate, which is the rate of tax at the"
0713      PRINT "top end of your income bracked at which the interest earned"
0752      PRINT "on your savings will be taxed."
0774      PRINT "And finally, for how many years do you want this savings plan"
07B5      PRINT "to run?  Try several different variations!"
07E3      PRINT "If you have a printer you can also print the result."
081B
081C 100  PRINT ""
0823 200  INPUT "INITIAL ANNUAL CONTRIBUTION: ",AP
084B      INPUT "% INCR IN CONTRIB/YR: ",IC
0869      INPUT "EST. INTEREST RATE: ",IR
0885      INPUT "MARGINAL TAX RATE: ",TR
08A0      INPUT "TERM IN YEARS: ",YR
08B7
08B8      INPUT "PRINT? (Y/N): ",P$
08CE      IF P$="y" THEN
08DB        P$="Y"
08E3      ENDIF
08E5      IF P$="Y" THEN
08F2        INPUT "MAKE PRINTER READY  [ENTER]",R$
0915        OPEN #PRINTER,"/P":WRITE
0922      ENDIF
0924
0925 300  PRINT CHR$(12);
092E      PRINT "INIT ANN CONTRIB: "; AP; "  % INC "; IC;
0958      PRINT "  INT RATE: "; IR; "  TAX % "; TR; "  TERM: "; YR
098A      PRINT
098C      PRINT "YEAR       CONTRIB       INTEREST       TAX PAID ";
09C2      PRINT "  ACCUMMULATED"
09D4
09D5 320  IF P$<>"Y" THEN 500
09E8
09E9      PRINT #PRINTER,"INIT ANN CONTRIB: "; AP; "  % INC: "; IC;
0A19      PRINT #PRINTER,"  INT RATE: "; IR; "  TAX % "; TR;
0A42      PRINT #PRINTER,"  TERM: "; YR
0A57      PRINT #PRINTER," "
0A61      PRINT #PRINTER,"YEAR       CONTRIB       INTEREST       ";
0A93      PRINT #PRINTER,"TAX PAID   ACCUMMULATED"
0AB3
0AB4 500  FOR I=1 TO YR
```

```
0AC9        AC=AC+AP
0AD5        ANCNT=AP
0ADD        TP=TP+AP
0AE9        B=AC*IR/100
0AF9        TI=TI+B
0B05        TA=B*TR/100
0B15        TT=TT+TA
0B21        AC=AC+B-TA
0B31        AP=AP+AP*IC/100
0B45        PRINT USING "R4.0^"; I;
0B54        PRINT USING "R15.2^",ANCNT; B; TA; AC
0B6F
0B70 720    IF P$="Y" THEN
0B80          PRINT #PRINTER USING "R4.0^"; I;
0B93          PRINT #PRINTER USING "R15.2^",TP; B; TA; AC
0BB2        ENDIF
0BB4      NEXT I
0BBF
0BC0 1010 PRINT "   "; LINE
0BCE      PRINT "    ";
0BD7      PRINT USING "R15.2^",TP; TI; TT; AC
0BF2      IF P$="Y" THEN
0BFF        PRINT #PRINTER,"   "; LINE
0C0F        PRINT #PRINTER,"    ";
0C1D        PRINT #PRINTER USING "R15.2^",TP; TI; TT; AC
0C3C      ENDIF
0C3E
0C3F 2000 INPUT "PRESS [ENTER]",AA$
0C57      IF P$="Y" THEN
0C64        CLOSE #PRINTER
0C6A      ENDIF
0C6C 2200 END
```

ooooooooooOOOOOOOOOOoooooooooo

Device Descriptor Query

Geoff Donges of Holt ACT, has asked a question about the descriptor "**dd**" he came across where he expected that the correct descriptor should have been "**d0**".

The following is from the Microware OS9 reference manual for the Color Computer:- "Device Descriptors - Term, T1, D0, and so on, are device descriptors. These files describe the devices connected to the system. They contain device initialization data as well as code that directs OS9 to the physical addresses of the ports to which devices are connected." (End quote.)

The descriptor "dd" has a full name of "default descriptor" and is a floppy disk drive descriptor, as is d0, d1, d2. The Microware Level 2 OS9 version for the Color Computer 3 is supplied with a descriptor "dd" which is identical to "d0" apart from the name. Therefore, a pathlist containing "dd" will access the drive "d0". Why two descriptors for the same drive? The logic behind this is that you may wish to have some other drive as the "default drive", for example "d1", in which case the "dd" descriptor would need to be the same as "d1".

Some programmers use the descriptor "dd" in the program code. One example of this is Multi-Vue, which includes the command "chd /dd" in the startup file. Multi-Vue also looks for its environment file (env.file) by the path /dd/sys/env.file. Now, if the descriptor "dd" is for drive 1 (drv=01), Multi-Vue should run from Drive 1 - well at least that is the idea of the "dd" descriptor. A user should be able to choose which drive to set up as the default drive and then run application programs without the need to make changes to the program. This, however, is not always the case. Let's go back to Multi-Vue for a moment. The Autoex & Multistart modules of Multi-Vue are

hard coded to d0, d0/CMDS which of course will look at Drive 0 for files without regard for the default drive which is now set at d1 (drive 1). As not all programmers follow the convention as intended, the only options are to patch all modules which are hard coded for /d0, or to take the easy way out and have "dd" set for Drive 0. This does mean that the Boot disk must be removed from /d0 and the application program run from Drive 0. Now it does not matter much if the program looks for files by a pathname beginning with "dd" or "d0".

The most common approach is to have exactly that configuration, ie both "dd" and "d0" are set as Drive 0. It is a good idea to have this "dd" in the OS9Boot file so that fewer ERROR #221's or ERROR #216's are encountered.

It seems to me that the advantages to be gained from a default drive descriptor "dd" have largely been diminished by use of hard coded pathlists to a particular drive, usually "d0".

Now whichever way you choose to go, it must be remembered that OS9 on the CoCo can ONLY boot from drive 0 or hard drive. In the case of a hard drive, a "dd" descriptor identical to the "H0" device descriptor could be used and of course any hard coding of "d0" in programs running from the hard disk, would need to be patched.

Good programming should not include hard coding to "d0", so do your bit to avoid it and your programs will be more flexible in the OS9 environment.

A better approach to "dd" default drive would be the use of dot or dot dot ie, "." ".."

Each directory has two entries with names "." and ".." (dot and dot dot). These entries contain the logical sector numbers of the directory and its parent directory respectively. This opens up a lot of possibilities.

eg "dir ." will display the files of the current working (data) directory.
"dir .." will display the files of the parent of the current working directory.
"dir ..." will display the files one level higher ie the files of the parent of the parent.

So, instead of a pathname of "/d0/cmds" or "/dd/cmds" we could use "../cmds". The ".." will be the parent of CMDS in this example, and will be the current working directory in most cases. Let's run Multi-Vue from drive /d1. Place the Multi-Vue disk in drive 1 and "chd /d1;chx /d1/cmds". We would now expect Multi-Vue to be run from drive 1, and it would if all hard code in the modules for "d0" were changed to ".."

OS9 does not seem to mind if the number of parent levels called for is too many. If you have not used "." or ".." then go ahead and give it a try with a few directory levels. Regards, Gordon Bentzen.

oooooooooo0000000000oooooooooo

FURTHER THOUGHTS ON THE CAPABILITIES OF SHELLPLUS2

One of those things that I had put off until sometime in the future finally came back to mind the other day. I had always intended to explore the shellscript capacity of shellplus, but somehow I never got around to it, until recently.

What I wanted was to be able to have a simple command to format my ramdisk. Simply put, instead of typing ... sformat /r0 r "RamDisk", I wanted to be able to type ... rf, or something similar and the job would be accomplished for me by a shellscript. The following is what finally developed from that.

I use sdisk3, a copyrighted disk driver from D.P. Johnston, as my standard disk driver, and my formatter is called sformat. I normally use this driver, as it allows me to read, write and format disks in a number of other formats apart from Color Computer OS9. The following shellscript is what finally developed:

```
*ramdisk formatter
prompt Format Ramdisk? (y/n) :
var.0
if %0=y
   display 1b 22 01 13 02 28 14 05 04
```

```
        display 1b 22 01 15 03 24 12 05 02
        display 1b 22 01 17 04 20 10 05 02
        sformat /r0 r "RamDisk"
        display 1b 23
        display 1b 23
        display 1b 23
    Echo Ramdisk Formatted
    endif
    *end
```

I guess I should try to explain what it does.

First of all, you will need to have the following commands in your execution path:

prompt	(simply a modified version of echo, but without a <CR> at the end of the string)
display	(a standard OS9 command)
sformat	(special sdisk3 formatter)
echo	(a standard OS9 command)

You can of course, use the standard format command instead of sformat, but you will then have to alter the window dimensions to allow for the differing outputs.

I decided that I would incorporate it in my startup file. Consequently, I had to have the ability to choose whether I wanted it to do the format or not. Then I thought, well why not have the formatter running in a window? Then, if I decide to reformat the ramdisk from a shell running under another process (edit for example), then at least the original contents of the screen are not destroyed.

So, running through the scriptfile, firstly, we ask whether or not to continue with the format. The answer to this question is placed in the shellscript variable var.0. The variable is then tested. If the result is anything but yes, the script drops through the loop and exits without doing anything.

If, however, the first character of var.0 is "y", then the contents of the loop are executed. The loop creates a number of non-destructive overlay windows, and runs the formatter in the "ready" mode, and sets the disk name to RamDisk. It then closes the windows sequentially, until they are all removed, and the finally writes the message "Ramdisk Formatted" onto the original screen. All from a simple shellscript!

The great thing about shellscripts is of course, that they are, comparatively, quite small. They are very effective because they have the capacity to call, and string together a number of other commands. Because of the way in which Shellplus2 handles memory data modules, it is possible, if we provide a shellscript with the appropriate header and CRC, to actually load them into memory, and execute them from there, just like an executable file. What this really means, is that you could write a large number of shellscripts, and merge them together as one single file. Because of their small size, you could cram quite a lot of them into one 8K block of memory ... The datamod utility which is supplied with the shellplus2 programmes is the best way to accomplish this.

Another example of creative hacking came to light the other day. I was examining the prompt setting facilities of Shellplus2, and I couldn't see any reason why I couldn't have my prompt a different colour from the rest of the output!! I thought that, if I could include an escape string (such as is used with the display command) to change the screen attributes, within the prompt, then it should be possible to have all sorts of modified prompts. How about a flashing prompt? Or underlined? It brings to mind all sorts of possibilities.

Unfortunately, I have not as yet found a simple way to accomplish this. I am working on it. Let me describe my initial attempts.

If you read the documentation that comes with shellplus2, some details are provided of just where in the shell module, the prompt string is located. What I did was to save a copy of the shell to my ramdisk, and then use "zap", our public domain disk zapper to actually modify the area of shell where the prompt resides, using zap's disk modifying command. After writing the modified sector to the disk, and exiting from the zapper, I reverifyed the shell module, and then incorporated it in my merged shell file that is loaded from the commands directory on bootup.

Hey presto, it worked!! Now I have a different coloured prompt from the remainder of the text. There are a few problems associated with this approach, but I will leave them for you to explore. In the meantime, I am attempting to find a way to pass escape strings direct to the shell via the P= prompt setting facility of shellplus2. I will keep you informed.

Until then, keep on hacking.

If you have any questions or comments, please feel free to contact me on
(07) 375-3236. Cheers, Don Berrie.

oooooooooo0000000000oooooooooo

Operating System Notes.

Ever tried to use a Level I startup file under Level II? If, like some of us you suddenly found that your screen pause was not enabled, then read on.

The normal arrangement for setting the page pause flag under Level I, was to use the line:

```
Tmode.1 pause
```

This then set the global (under Level I) flag to on, and consequently, even when the startup file terminates, the page pause feature is still enabled.

Under Level II, however, the system is "purer" OS9. The first thing that the system does is load grfdrv, and then loads the merged shell file from the commands directory. It then looks for a startup file, and if found tries to run it. However, it does this as a shell process. What this means is that if you include the above line in your startup file, the tmode command processes as per normal, but when the startup file terminates, all of the attributes that were set dissappear with the shell. If, however, you have started shells in any other windows from the startup file, then the pause attribute for those windows is enabled!! (Because they will inherit all the attributes of their parent shell).

To successfully set the screen pause attribute from the startup file under Level II, you should use:

```
Xmode /term pause
```

Try it and see.

oooooooooo0000000000oooooooooo

COCO-LINK Magazine: Just a reminder for those interested in a wider section of CoCo systems. CoCo-Link is produced in South Australia and is well worth your support. Contact the Editor, Robbie Dalzell, on (08) 386 1647 or write;

CoCo-Link
31 Nedland Crescent,
Pt. Noarlunga South,
South Australia 5167