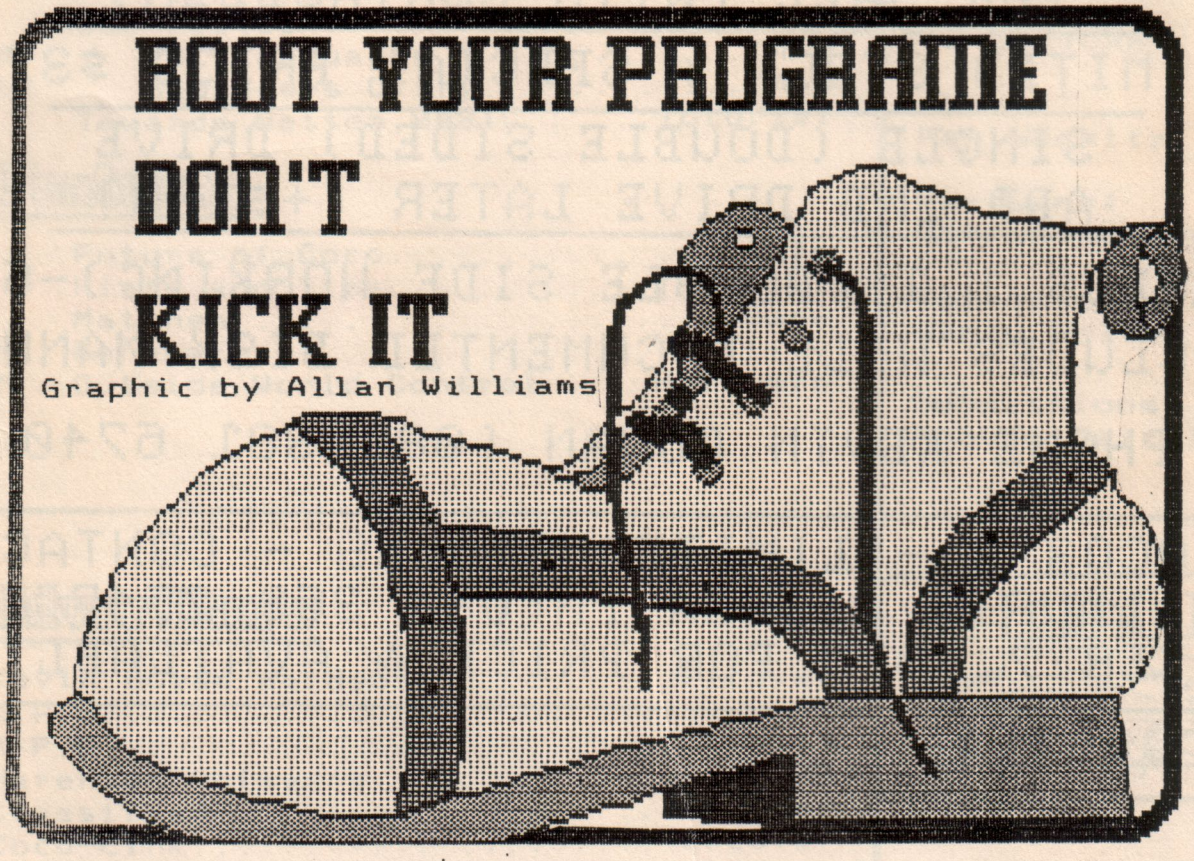
August 1990 Vol 3. No.4

COCO LINK

The Color Computer Magazine



Featuring:

COCO DISK DRIVES

FOR SALE (WITH CONTROLLER)

LIMITED STOCK : SPECIAL PRICE $375

SINGLE (DOUBLE SIDED) DRIVE
ADD 2ND DRIVE LATER (+$100)

TRSDOS 1.1 (DOUBLE SIDE WORKING)-6MS
INCLUDES WELL DOCUMENTED DISK MANUAL

PHONE KEVIN GOWAN (08) 381 6740

COCO3'S - LIMITED STOCKS - CONTACT
LAURIE O'SHEA PH(08) 363 2647
AFTER 7.30PM FOR PRICE & AVAILABLITY

Contents

Departments

Columns

Features

Advertisers

COCO LINK

EDITOR:

Robbie Dalzell

ASST. EDITOR:

Garry Holder

SUB-EDITORS:

OS9:
Ken Wagnitz

Hardware:
Darren Ramsey

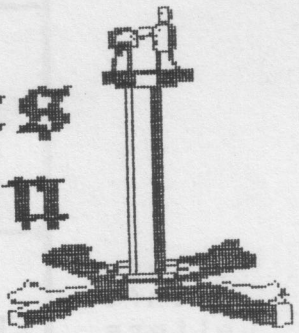
Correspondence:
Garry Holder

Submissions:
Robbie Dalzell

COCO LINK

Copyright Notice

Robbies Column



COCO AND COCO-LINK'S FUTURE

Nick Marentes has written a follow up on his article on "The Future Of the COCO". This gives the details of the two new machines which will be marketed under that banner. Whether either will ever reach our shores is in doubt but, should we ever get the oportunity to purchase either, we will still be stuck with the dilemma of what direction to take....the Hogg or the Leigh version. Nick has stated his views and as a commercial programmer they must be taken seriously. Programmers are the people who will ultimately decide which machine will have the backing of sufficient software.

Whether the new Coco's are ever available in Australia does not matter to the majority of us at this time. The Coco2 and 3 will still be used for many purposes for a number of years to come.

Those of you who will be looking to carrying on computing into the years ahead will have the decisions to make as to what will be your next step forward. It is a decision which deserves the closest attention.

In the meantime, let us make the most of our Coco2's and 3's

To assist COCO-LINK to do this we need material from you, the readers. We have a small nucleous of enthusiasts who write programmes, articles and send us hints and questions. All these things help us to keep the magazine interesting and entertaining.

The trap lies in material only being submitted by the enthusiastic few. It means that the areas of interest will be spread over a narrow spectrum.

Conversely, material received from a greater cross-section of readers gives us a far broader band of interest subjects. This is what lifts the magazine to greater heights. So get your printers and brains into gear and send in your programmes or any other material you think would be of interest to our readers. We can never have too much material.

There's still a good number of years in the Coco yet! Let's make them good ones.

****

ON SUBMISSIONS

We have a new system for when we receive material from readers. On receipt of the material an acknowledgement card is sent to the submitter of the material. The material is then subject to scrutiny to assess its potential. A decision is then made as to when and if the material will be used in the magazine.

If there is a delay before the material is used we try, when possible, to inform the submitter of article/programme's impending publication.

The editor holds the right to edit material in line with magazine policy. This will not alter the concept or intended views of the writer.

All material submitted to COCO-LINK remains the property of the writer.

****

OOPS!!!

I have to make an apology to Stephen Bell for heading his two educational programmes which appeared in the April magazine, as being written by Stephen Vagg. Johanna assures me that no Vagg of that name exists. (At least not yet).

This error was pointed out by Johanna and I must apologise to her also for adding to her family prematurely.

Stephen Bell is 10 years of age and anyone who transcribed his two programmes must agree with me on the good job he did on them.

Again, I apologise to both Stephen and Johanna for any inconvenience this may have caused. I will not promise not to make mistakes in the future. As they say,"To err is human" and if you go by the number of errors made, then I am certainly that.

As proof I have another couple of little errors to off-load. Both are Phone numbers.

Glenys Ferres number has been misquoted in one section of the Noticeboard page. The correct number is 332 4264.

The other one pertains to the article on "THE TALKING BEAR" where Jim Eadsforth's number was given incorrectly. The number should read 298 2843.

****

PROBLEMS, PROBLEMS

As part of the service provided by COCO-LINK, we attempt to answer queries on the various problems subscribers come up against. Quite often these problems are related to trouble with programmes listed in magazines other than COCO-LINK.

While we are happy to be of assistance in all cases, I would like to highlight the need for submitters to thoroughly check their listings before sending an enquiry.

It is daunting to have spent considerable time in tracing

a programme bug only to find that it is a simple case of bad typing which should have been picked up in the first place by the submitter.
So please, please double or treble check listings sent to COCO-LINK for problem purposes.

****

SUCCESS AT A PRICE
The world of high finance never fails to amaze me.
S.A. personal computer manufacturer, MicroByte Systems, who export computers to Britain and Europe has been placed in receivership by the ANZ Bank.
This, when the company has orders for 450 computers per month but can only manufacture 350 per month thereby creating a waiting list. They need money to expand their manufacturing base but borrowing more money puts them in deeper debt to the banks. Therefore no money is forthcoming.
It just goes to show that success isn't all that it is cracked up to be.

****

COCO-LINK SURVEY
I have not received as many survey returns as I expected. This may be partly my fault.
I have been contacted by subscribers who do not wish to dismember the magazine and who do not have access to a photocopier.
This has prompted me to continue the survey for another issue. This time I have included the survey form separate from the magazine.
Those of you who have not already done so would you please fill it in and return it as soon as convenient. It will be a great help in formatting future magazines to your wishes.

****

THE FRENCH-CANADIAN CONNECTION
I am now in correspondence with Armand Belanger in Quebec, Canada. I have recieved a copy of his magazine which is printed in French. Hopefully we will be able to glean some further knowledge from it's pages.
Clubs in Canada are having the same problems as we have in Australia. With the demise of the Coco as a saleable item by Tandy, we find that memberships are on the wane.
Armands correspondence in the letters page gives us a bit of insight into the scene in Canada.

****

WHAT'S NEW AROUND THE WORLD
They are developing a computer in the United Kingdom which will be able to tell the difference in the aroma of different types of beer.

I know a couple of Aussie fellows that could do just as well for half the price. In addition they would be able to judge the taste as well as the aroma.

Sony Corp. has for release in July the Data Discman. This is a palm-sized portable compact disk player with a screen that displays text recorded on special CD's called electronic books.
Each 8cm compact disk can store 200Mb of information - about 1000 pages of text, or more than 30 paperback books. A small typewriter-like keyboard allows users to select particular items.

****

CLUBS
I have had a request to list all the clubs in Australia which deal with the Coco.
Our CLUB NOTICEBOARD does that to the best of our knowledge but we can only print details of clubs we know about.
If you are a member of a club please send us its name and the name of the club contact plus his phone number. Clubs have always been the best places to learn about computing.
So, again, it is up to you!

****

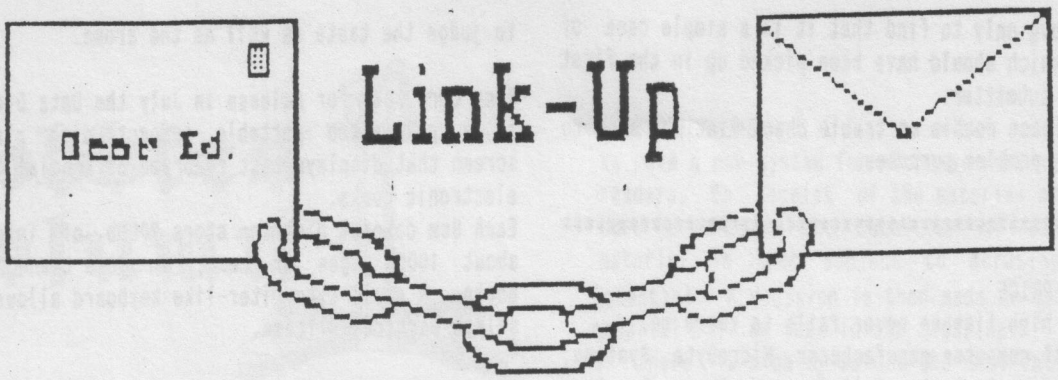
Keep Hacking
Robbie

☐ Gobbledegook Corner

"OBJECT: Every object is an instance of Object and every class is a subclass of Object. The metaclass of object is Object Class. Object Class is a subclass of Class. However, Object class does not have a metaclass.

CLASS: Every metaclass is a subclass of Class. The metaclass Class class (of Class) is a subclass of Object class which, as we said earlier, is itself a subclass of Class! Thus, Class class is transitively a subclass of itself.
METACLASS: All metaclasses are instances of Metaclass. The metaclass of Metaclass if Metaclass class which is a subclass of Class."

Definitions of terms in computing paper "Object Orientation" by Messrs Goldberg & Robson, submitted by Russell Talbot.



Dear Ed,
Yesterday I received a letter from one of your readers asking how to copy an ML file from one disk to another; this is my reply.
"Thank you for your letter, I have not done much computing since the beginning of this year, but I am able to answer straight away because I know the answer and don't need to 'fiddle'. The ML LOCATOR program only works for programs CLOADMed from tape. I wrote about this in one of the articles in COCO-LINK. With that article there should have been a program called FIND ADD.

You do not really need this program to be able to copy an ML file from one disk to another. It really is quite simple once you know.
To copy a file (eg JO/BIN), with a one drive system, have your disk with the file in the drive and type:

COPY"JO/BIN <Enter>

The computer will prompt you to insert your destination disk. I think that for a long file, you could have to swap disks a few times; just follow the prompts on the screen. As I have two double drives, I don't have to swap disks. I type:

COPY"JO/BIN:0"TO"JO/BIN:1 <Enter>

With the original in drive 0 and the destination disk in drive 1"

By the way, the programs you think you have been saving, and re-loading and executing successfully, are executing because they are still in the computer until you turn it off. That is why you have not had any success with these same files, "after turning the computer off".

Johanna Vagg. NSW

Sorry for the ommission of your programme in our earlier edition. I have included it below in the hope of making amends (and help a few people into the bargain).

```
10 CLS:PRINT@130,"":INPUT"FILENA
ME/EXT";L$
20 OPEN "D",#1,L$,1:IF LOF(1)=0
THEN CLOSE:KILL L$:RUN
30 FIELD #1,1 AS F$:LF=LOF(1)
40 FOR Q=1TO5:GET#1,Q:BY(Q)=ASC(
F$):NEXTQ
50 B=0:FORQ=LF-4 TO LF:B=B+1
60 GET #1,Q:BE(B)=ASC(F$):NEXT
70 ST=(BY(4)*256+BY(5)):EX=(BE(4
)*256+BE(5)):LN=(BY(2)*256+(BY(3
)-1)):EN=ST+LN
80 CLOSE
90 PRINT:PRINT:PRINT"ST="ST;"EN=
"EN;"EX="EX
100 PRINT:PRINT
110 PRINT"ST=&H"HEX$(ST),"EN=&H"
HEX$(EN),"EX=&H"HEX$(EX)
```

****

Dear Ed,
I read in LINK-UP April 89, where Bernard Fletcher had difficulty in understanding his DMP 105 printer. I have a DMP 107. I don't know how much they differ but I don't think they would vary much. The Scripsit booklet is a bit vague on the subject of fonts, but it was made to cover many Tandy printers. Page 60 line 2 gives the only clue to the hex characters, while the last paragraph refers you to the printer's manual. Simply use the HEX column in the appendix of the printer manual instead of the clumsy Basic CHR$(27);CHR$(nn) etc. The fonts can also be overlapped to suit your needs.
For instance type in:

.hx 1B,42,01 <enter>

and your in italic font. If you then type in any other code and it will be in italics. Elongation code can be followed on the next line by bold face code and the result is more impressive for a heading. A list of common fonts is set out below. If a line starts with a stop (.) the printer sees it as a command and ignores it if it doesn't make sense, hence the lines below start with an asterisk.

	SCRIPSIT	BASIC CHR$(nn9 etc	
*	.hx 1B,13	27,19	Standard-10
*	.hx 1B,12	27;18	Correspondence-10
*	.hx 1B,17	27;23	Standard-10
*	.hx 1B,14	27;20	Condensed
*	.hx 1B,1D	27;29	Correspondence
*	.hx 1B,0E	27;14	Start Elongation
*	.hx 1B,0F	27;15	End Elongation
*	.hx 1B,1F	27;31	Start boldface
*	.hx 1B,20	27;32	End boldface

*	.hx 1B,42,01	27;66;10	Start italic
*	.hx 1B,42,00	27;66;00	End italics
*	.hx 1B,53,01	27;83;01	Start subscript
*	.hx 1B,53,00	27;83;00	Start superscript
*	.hx 1B,58	27;88	End Sub/sup/scrpt
*	.hx 1B,4D	27;77	Microfont
*	.hx 1B,51	27;81	Set left margin
*	.hx 1B,52	27;81	Set right margin
*	.hx 0F	15	Start underline
*	.hx 0E	14	End underline

Hoping this will help Bernard, I can't help him with the tape problem yet, but I will work on it.

Sam Thompson. Gold Coast. QLD.

****

Dear Ed,

I was very pleased to receive your letter along with the June 1990 COCO-LINK, I find your magazine very interesting and well informed. The magazine I edit called "Le COCOTIER" relates as much to the container (coquetier) which holds the meat of the egg as much as the fruit of the palm tree. Your magazine will be very useful to me although the one I send you may not be much use to you since it is all in French. Please show it to Kieran Kenny who reads French, he may be interested. Our Club season is now over till the fall in September when its activities will resume.

Up till last year our Club had many members who were very active, but with the disappearance of the CoCo and the lack of new programs for it, it has started to show up greatly. All our expert members in OS9 and Machine language are switching to IBM. Since new members are impossible to find now, our Club may not have more than 20 COCOists in the fall, hence we will form an IBM section to at least help cover our expenses although both sections will operate separately. From having had meetings every week we will only have 2 a month next year plus one for the IBM section. I do not know how long the Club will persist.

I use the TW128 to write the magazine because it has all the Accents. (To print in French). I do not use Window Writer as it is too slow and not quite bug free. All other Desktops are not good enough or too cumbersome such as VIP Writer etc, MAX10 only has Accents in one FONT and is not too easy to use for long texts.

I will be interested in the articles written on Sculptor which is a fairly complex although very powerful DATABASE. As for OS9 I prefer DATA MASTER which is a very user friendly DATABASE and much more useful and faster than Sculptor. I have, for instance, with it listed with explanations for our members all the articles published in Le COCOTIER since its beginning. If I had time I would do the same for Rainbow.

Armand Belanger. Quebec, Canada.

****

Dear Ed,

I cannot tell you how delighted I was to hear that you are having problems getting started in OS9! Around 18 months ago I bought the two disks and played around with OS9. I found the book with disks almost almost undecipherable. I then bought the Rainbow book on OS9 level 2, to my astonishment it too was almost undecipherable or else written in a foreign language.

I am a reasonably intelligent professional, a retired doctor, and yet it seems that everyone I talk to has the same problem!!!

They all assume you you to be fully OS9 computer literate. I speak a reasonably fluent basic yet I am lost in the sea of OS9!

I have struck the same problem with BIN, ML and Assembly language. I have the tape version of EDTASM+ including its booklet. The recommended Assembly language book from Radio Shack seems to be written in the same foreign language. I am starting to feel like a moron with an IQ of around 10%..... H E L P !!!!!!

Seriously, how do I get the first grasp?

Sure I can load the disks and get the OS9 OK! I can also print some of the "read me first" items in some of the Rainbow OS9 disk sections and yet I get lost when I try and load some of the programs included in UPPER CASE the 'Read me' first is in lower case.

To revert to ML, do you or do you know anyone who has a disk version of EDTASM+ or a copied ROMPACK on disk that I could get a copy of?

Also another HELP.....

How do you list, as in BASIC (LIST ENTER)a BIN program?

An odd program such as the Australian Geography one supplied with my original COCO2, had a LOADM one which listed; most ofcourse do NOT. Once again HELP!

I have a program FLIGHT SIMULATOR from Tandy. On DIR you get 2 programs, 1 FLYSIM/BAS and 2 FLTSIM/BIN. Number 2 LOADM's OK and executes..... Number 1 FLYSIM/ BAS loads and on RUN only gives an OK. Listing also gives an OK only. Does this mean that there is nothing there? How do you tell if there is a program there? I have tried all the list disabler POKES that I know or can find, NO GO!!! HELP!!!

Sorry for all this whinge, but I am lost. Maybe I ask a lot, but to hear you are having trouble is to say the least, encouraging. Maybe just MAYBE I am not such a moron after all!!

Congrats on a fine magazine, and particular congrats on the Beginners Diary.

Dr. Val Stephen. Camberwell. VIC.

A BIN programme cannot be listed in the same manner as a BAS programme. If the programme was written in Assembly Language then it is possible to list the source code. (This is the mnonics file before it is compiled to machine language). You would normally list this with a

word processor or line editor.
Thank you for your kind comments on the magazine and the DIARY column. As stated before this is not meant as a tutorial on OS9 but merely a diary of how I personally am progressing with it.
It might interest you to know that the OS9 Users Newsletter is running a series of articles on OS9 for beginners. This is by Gordon Bentzen who has assisted me on a couple of occasions. (The group is advertised in our Noticeboard page). Also I have been told that the "START OS9" package of disk and book are the best OS9 tutorial on the market
I can't answer your Flight Simulator query at the moment but I am sure someone out there will come up with the answer and let us all know.

****

Dear Ed,
I'm writing to you to inform you of some excellent service I received off one of your advertisers.... BLAXLAND COMPUTERS.
It took one letter from me, and one week later to receive information about a 512k upgrade for my CoCo 3. And it took one letter and $186 (including postage & handling), and one week later to receive it.
This is a lot QUICKER than having to wait FOUR MONTHS for just ONE LETTER off one of your other advertisers in the magazine, here in Queensland.
All the best from another happy CoCo'ist.
Desmond Rae. Mt Isa. QLD

****

HOW TO SUBMIT MATERIAL TO COCO-LINK
****

PROGRAMMES: On tape or disk.
At least two copies should be on the tape/disk one of which should be saved in ASCII format.
Where possible include a description of your prodgramme saved as below for articles.

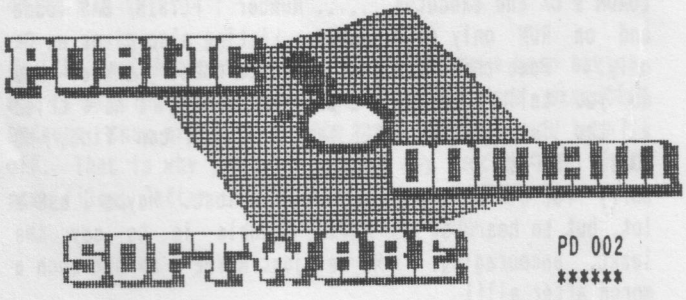
ML PROGRAMMES:
These require Source code saved on a suitable word processor. Two copies should be made.
A working copy of the programme should be included for checking by COCO-LINK.

ARTICLES:
At least one copy saved in ASCII format plus one copy on a commercial word processor where possible. (VIP Writer etc.)

HINTS AND TIPS:
Hand written or typed is acceptable.

LETTERS TO THE EDITOR:
Hand written letters will be accepted subject to the length. Long letters should be submitted on disk in the manner above for articles.

All disks and cassettes will be returned in due course.



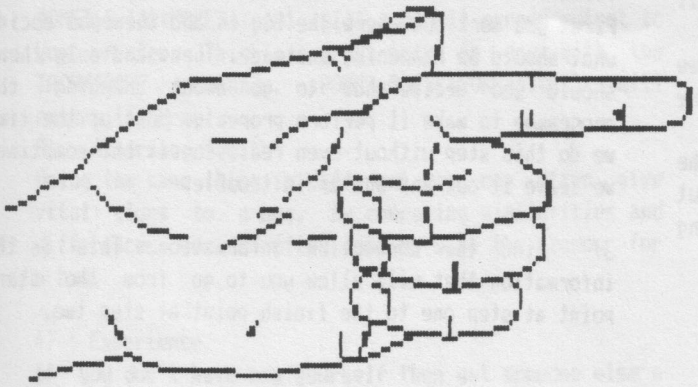
PD 002
******

This months Public Domain Disk is another educational one. We have tried to make it cover as much ground as possible, from the littlies at pre-school to the older students at high school.

The programmes on the disk are:

BINARY.....Given a decimal number, this programme will show you the equivalent Hexadecimal number and the Binary number.
COCOHOME...This programme is for the pre-schoolers.
COINDEMO...Uses Coco 2 graphics to show the Australian currency.
FORMULA....This programme covers just about every mathemetical formula that you can think of
MATCHEM....Another for the young and not so young.
MATH.......Tests your mathematical skills by giving you set questions on the subject of your choice.
MATHSMT....A fun way to learn maths.
MEMORY.....Tests your short term memory.
NUMFUN.....Another fun way for the youger set to get into maths.
PUZZLE.....Makes up those word puzzles you love to do in magazines. Use it to entertain your friends.
TRIGSHOW...For the high school students. This programme uses graphics to explain the trigonometrical formulae.
WORD.......This is a multi-player scrabble type word game.

Who says education can't be fun!
This disk is available from COCO-LINK for $5.00 (Postage included).



Better BASIC Part 11

Debugging

By Robbie Dalzell

This article has been put together using ideas etc. from various sources. It is a primary look at how to go about debugging a programme which, for all your skill and typing expertise, refuses to work properly.

First of all, what is a bug?
A BUG.....is an errer in a programme which is buried in the code and is difficult to see at first glance.

What is debugging?

DEBUGGING.....is the elimination of probable causes of error until the actual cause of the error is identified and corrected.

Many times you will think you have found the possible cause of a bug in your programme and then be completely stumped when that turns out not to be the problem after all. The question must be asked time and time again,
"What else could give these same symptoms?"

Basically there are two types of bugs. Syntax errors and Logic errors.

SYNTAX ERRORS.
A typing error means that you have in fact had the right idea, but didn't use exactly the words that the computer could understand. Computers are absolutely stupid.
Remember that you are dealing with a dumb machine. It cannot anticipate what you meant to say. If it sees "FIR", it will treat it as another variable, not "FOR" as you intended, but because you are a 60 mistakes a minute typist you hit "I" instead of "O".

In some cases the mistake occurs so as to make sense to the computer in a way you never intended. The computer then follows the wrong path until it finds something that doesn't fit. By the time comes to your attention, the error shows in a place where there is no mistake at all.

xxxx B=AFORI=1 TO 10
In this case, there is a colon missing between "A" and "FOR". However, the computer read that as a variable "AFOR" and continued on looking for an operation or a colon,etc. to come after this "variable". When none appeared an error was flagged on the "I". The line should read:
xxxx B=A:FOR I=1 TO 10

xxxxPRINT SCORES"
In this case the quotation mark was left out at the beginning of the string "SCORES". Again it was interpreted as a variable and the quotation mark after it was flaffed because the quotation marks shouldn't follow variables. The line should read:
xxxxPRINT"SCORES"

Here is a brief list of typical Typing Time errors:
1) Variable names containing reserved words: for example the variable "FORM" contains the reserved word "FOR". (Reserved words are those set aside as commands, functions, etc.)
2) Wrong abreviations: Particularly when the abreviations are similar for commans such as "GOTO" and "GOSUB".
3) Quotation marks left out.
4) Mistaken characters. Particularly the number 0 and the letter O. (This is especially prevalent on Cocol

and 2 where the zero is not slashed). The number 1 and the letter I are another couple which look alike especially if you have been typing into the 'wee small hours.'

5) Insufficient spaces between words etc. All three Coco's are very tolerant of spaceless code but there are a few traps waiting for the unwary.

To remove the chance of syntax errors caused by the lack of a space it is good procedure to always put spaces between the following words when using alphabetical variables:

FOR-TO-STEP-NEXT
IF-THEN-ELSE
ON-GOTO
ON-GOSUB

6) Truncated lines. This occurs mainly when you don't look at the screen and you type past the end of the buffer. (This means putting too many characters on a line: Coco allows you to put 128 characters in one line.) This happens most often with inexperienced or tired programmers.

LOGIC ERRORS.

Here are four steps that can be used in finding programme bugs:

1) Firstly identify exactly what is known. This is the most crucial step as it is the basis for all the others. A common mistake at this time is to jump ahead and try to guess the solution.

In identifying what is known, three things stand out as the most important.

a) The actual symptoms. Get these clear in your mind first before continuing.

b) Identify the exact location of the bug and/or the conditions that produced it.

c) Have your manual handy.Being thoroughly acquainted with your manual is very important. quite often the answer to your problem is right there in the manual supplied with your machine. As the sayig goes, 'If all else fails, read the instructions.'

Here are a couple of examples:

Your printer may work perfectly when printing a heading but start to produce garbage when printing actual data. You now know that something has happened in between the two PRINT statements.

In another case it may be that most values will produce valid results, but particular values produce results that are obviously wrong. Those particular values need to be noted.

At other times it is a combination of factors that produce the problem. Isolating these factors helps greatly in locating the bugs.

2) Work out what is happening when everything is working correctly. Once you know exactly what you require the computer to do, instead of what it is currently doing, you can proceed to investigate the bug further.

First you work out where the bug is and then you decide what should be happening instead. Then, and only then, should you decide how to go about changing the programme to make it perform properly. Most of the time we do this step without even realising it but sometimes we leave it out and end up in trouble.

3) Find the connecting information. This is the information that will allow you to go from the start point at step one to the finish point at step two.

We can now move on to some techniques that can help:

a) Isolating the section causing the problem.

The first place to look is at the line that has been flagged as the error. When this line is not the culprit, start and stop the programme in various places (Using STOP and GOTO etc.). Start from a place that you know is working and gradually work your way through the programme until you reach the place where the bug begins. On some occasions it will be easier to work backwards to find this point. Obviously starting the programme in the middle can create problems so you might have to print some variables with suitable values beforehand.

b) Check variables and parameters.

Make sure the variable names do not include reserve words, are functionally different from each other and follow the correct format for your Coco. Remember that only the first two letters of the variable count on the Coco. (i.e. The variable TR would be the exact same as the variable TRIP.)

Get the programme to print out working variables as it is running. Sometimes the screen can be used for this; other times the printer is used.

You can also do the reverse by substituting known values for the variables and observing the results. This will show you if the algorithm is faulty or if the programme is picking up bad values for the variables. (An algorithm is the way a variable is processed.)

c) Trace the programme flow.

This is most important when you have a lot of looping and branching.There are three main methods of doing this:

1) Use pen and paper to go through the programme yourself step by step in the same way the computer does.

2) Leave markers generated by the programme. These markers will vary with the particular situation and can include the following:

Printing out messages such as 'THIS IS LINE 100'
Changing screen colours at strategic points.
Altering a sound register.
Using the TRON/TROFF commands. This magnificent

debugging tool can be used in the programme or separate from the programme as you require.

POKE360,162:POKE361,191 will copy all screen output to your printer. This is a great help in use with the TRON/TROFF commands. POKE360,130:POKE361,115 will return to normal.

3) Altering code.

Doing the same thing in different ways can often give vital clues to a bug. By comparing similarities and differences, you can often come up with the reason for a bug.

4) Experience.

If you don't have any yourself then get someone else's experience to work for you. The Computer Club is a good place to start. Many computer hobbyists like nothing better than a good problem.

5) Don't let them happen in the first place.

With experience you will find that you will make fewer and fewer mistakes. Here are a few hints to help stop the headaches before they begin:

Before you even put finger to keyboard, put pen to paper. Decide firstly the aim or goal of the programme and write an outline of what you want it to do.

When writing the programme, include enough comments to explain to another programmer what you are doing. This also helps you to understand your own programme six or more months later. It also focuses your attention on what you are doing and many times you will find a bug as you are writing.

While you are writing it is also a good idea to keep track of the variables you are using. This will help to stop them accumulating to vast proportions as well as helping to stop them being used for two jobs.

Use a pre-programmed set of commonly used subroutines. This saves on typing and cuts down on the chances of making errors. (GOSUBBER is such a programme and can be found in our club Tape Library.)

Finally, develop and test your programme in small sections. This means some form of structured programming which is another subject all together.

Here are a few extra hints to help you in the debugging process:

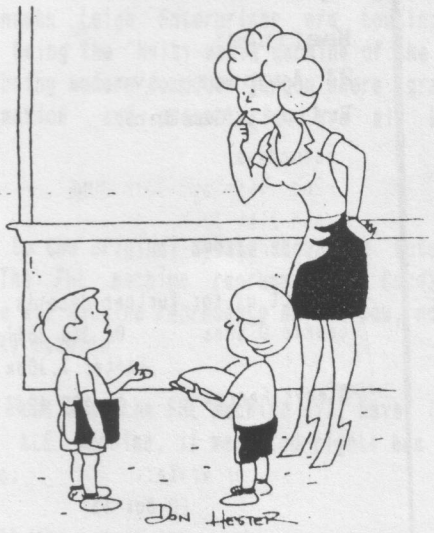
If you think that deleting a certain line might help your programme then place a REM before it. The computer will skip that line. To replace the line if that is found to be neccessary all you have to do is delete the REM.

One bug encountered for the first time gives lots of trouble to programmers who like to deal with numbers. This is the bug in the TANDY ROM which gives a problem with some decimal numbers. For instance if you type in 30.2 - 30 you will see what I mean. The answer given will be .199999 . Although this may seem a minute discrepency, when multiplying by large numbers the total discrepancy can grow at an elarming rate. The use of a formula such as.....C=INT(((A-B)*1000)+.5)/1000 will give you the answer to the above problem correct to five decimal places.

Another little preventative action I would recommend is to leave out all Hi-speed Pokes until the programme is up and running. Put them in while writing the programme by all means but put a REM in front of them until full debugging is completed.

Another trap to watch out for is returning to the start of a programme with the variable values still intact when what is needed at that point is all variables zeroed. Make sure when returning to areas of programme that the variables to be used are zeroed if that is what is required.

There are various programmes available both commercially and through magazines such as RAINBOW, COCO and SOFTGOLD which can help in the debugging process. One of the most used is a Variable Mapping programme. This type of programme lists all the variables in the programme and shows where all GOTO's, GOSUB's etc. start and end.



"Miss Smith, we can't work out why we have to learn all this stuff when there are thousands of computers which already know the answers."

Club Noticeboard

NATIONAL OS9 USER GROUP

The fullest OS9 information service in Australia.

Monthly magazine included in annual subscription of $18.00

Write now to:

Gordon Bentzen
8 Odin St.
Sunnybank
Qld. 4109

GENERAL NOTICE

This page is provided free for the use of clubs to let people know who, what, and where you are and to let them know what you are doing.
Please send your notices for the following two months no later than the 1st of the month previous to publication.

WANTED URGENTLY

Articles, programmes, hints and tips for inclusion in the pages of COCO-LINK magazine. See details of how to submit material to the magazine elsewhere in these pages.
Support COCO-LINK and help us to spread as much knowledge and information on the Coco as we possibly can.

MAROONDAH.....RINGWOOD

Every third Wednesday at 8.00pm.
(Subject to school holidays).

Contact...Andrew Rawlings 03 894 1443

ADELAIDE COMMUNITY COMPUTER CLUB

Meetings:
24 Avonmore Ave,
Trinity Gardens

Saturdays:
2nd, 16th and 30th JUNE
and 21st JULY
Small Fee (40 cents)

Contact us for further details:
Laurie O'Shea 08 363 2647
(After 7.30pm)
Glenys Ferres 08 332 4246

or write:
PO Box 157
Kensington Park
S.A. 5068

CLUB CONTACTS

Adelaide.....	Laurie O'Shea
.	08 363 2647
.	(after 7.30pm)
.	Glenys Ferres
.	08 332 4246
Basic.....	Johanna Vagg
.	068 522 943
Brisbane North....	M.Webster
.	07 285 6551
Brisbane S/W.....	Bob Devries
.	07 372 7816
Geelong.....	David Collen
.	052 432 128
Moe User Group....	Joseph Hester
.	051 277 817
.	Ian Taffs
.	051 275 751
OS9 User Group....	Gordon Bentzen
.	07 354 5141
Ringwood.....	Andrew Rawlings
.	03 894 1443

Clubs or persons wishing to be added to this list please inform the editor.

The Future of the Coco

The last word?

By Nickolas Marentes

It has been said that I was being overly pessimistic in my views and opinions towards the future of the Coco in my last article but I must say that I would disagree; prefering the description, practical and realistic. The fact is (and I wish I could be wrong) that Tandy will never release an upmarket and more powerful Coco in the future due to the fact that they have chosen to follow the IBM market path. The only chance of a Coco revival from Tandy is in the form of an expandible video game console.

I said in my original article that an upmarket approach to a new Coco design would fail. As it turns out, not one but two new Coco 4's are about to be released. In the time that follows we will see which approach succeeds.

Of the two new "Coco 4's" about to be unleashed in the US, one is from a company called Kenneth Leigh Enterprises, the other by Frank Hogg Laboratories (NO Tandy!). A brief specifications list of each follows:-

FRANK HOGG MACHINE

Based on Frank Hogg's (FHL) K-Bus system which he has been selling for several years now. This is a system whereby a computer is built up from various cards. He has now produced a card which makes it into an enhanced Coco3.

It uses a 6309 processor running at 3Mhz (25% speed increase), uses a PC compatible keyboard, 2 serial ports, parallel printer port, 512K RAM upgradable to 1 Meg, 8 bit D to A and A to D, has a standard Coco bus, will run Coco3 OS9 and RS-DOS (with seperately available Extended Color Basic ROM chip). Expansion via extra K-Bus cards to make system into a 68000 based system later.

Estimated price is round US$500 for base model.

KENNETH LEIGH MACHINE

A complete machine which, I must stress now, is NOT a Coco at all although it can be made to run Coco software via some connection to an existing Coco3.

It runs a Signetics 68070 at 15Mhz! (68000 compatible), has a custom Philips VSC video chip which has modes such as 320X200/420 at 256 colors per pixel, 384X480 at 256 colors per pixel, 640X210/420 at 16 colors per pixel and 768X480 at 16 colors per pixel. These modes include interlaced and overscan modes. Video modes can be changed on a line-per-line basis. System includes 1 Meg of video RAM, 3 serial ports (one midi configurable, the other for an IBM mouse), two bi-directional parallel ports, an IBM PC keyboard connector, SCSI interface for hard disks, 1X1.44 megabyte 3.5" floppy drive, 150 Watt power supply, low profile case housing everything including drives, expandable to at least 8 Megabytes of RAM, stereo DMA 8-bit sound, runs OS-9 68K in ROM.
(OS-9 68K is the OS-9 package based on the motorola 68000 processor).

Estimated price is around US$899 for base model and around US$1200 for expanded model.

This is a very impressive machine containing many of the features of the Amiga and more. Kenneth Leigh Enterprises seem to be going about promoting this machine in a professional manner and if all goes well, this machine may become big. Amiga developers have been approached and many have shown keen interest. This machine seems to be getting the most interest from existing OS-9 gurus in the states. Kenneth Leigh Enterprises are touting this machine as being the "Multi-media machine of the 90's". Multi-media being modern computer jargon where graphics, sound, animation and presentation are all brought together.

Getting back to the original debate about the future of the Coco. The FHL machine represents a Coco3 moved upmarket. The KLE machine represents an all new, non-Coco (but OS-9 68K) machine.

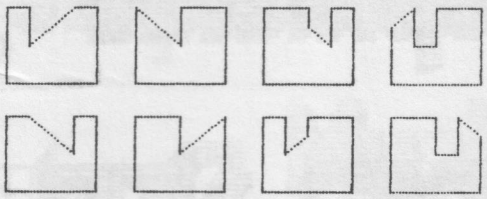
I personally feel that the FHL machine will have limited future. The KLE machine, if marketed right, has a very bright future.

Time will tell.

MATCHEM

BY KEIRAN KENNY

SYDNEY - 1990



MATCHEM will put four shapes at the top of the screen and a fifth shape at screen center. The object is to compare the center shape with the top four and press 1 - 4 if one matches or N (for NONE) if no match is found. Just to make your choice a little more difficult, the center shape will appear lying on its left or right side or up-side down, but never right-way up.

The data lines 750-820 contain strings for eight shapes, read as A$(1) to A$(8). There are four pairs of shapes, each shape being the mirror image of its pair. All eight are displayed on the title and credit screen.

Lines 890 - 950 sort the eight shapes in random order and lines 960 - 990 put the first four of the sort on the screen with the choice prompts and the center shape. Press the key of your choice. Lines 1030 - 1120 evaluate your choice and display the result.

As the shapes are, they should not pose any difficulty for a person with normal powers of observation, but I would see this program as a possible aid in exercising disadvantaged children or adults in distinguishing subtle differences in form. In this case, some initial guidance may be necessary. I think eight shapes is a good number. With more you would get too many No Match situations.

The screen will show an ongoing, cumulative score, RIGHT/WRONG and percentage after each try. After the first twenty tries, an open-ended trending score will be displayed every ten tries. Your cumulative percentage at that point is compared with your cumulative percentage at the end of the previous ten tries and displayed with an appropriate message.

The character set in this program, lines 200 - 740, includes the most used punctuation and math signs on your keyboard. The string for each character is labelled Z$ plus its CHR$ number in brackets.

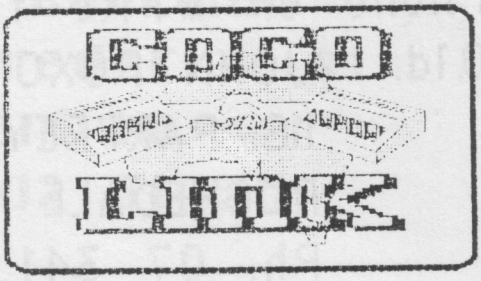
One of the characters has been given a string other than its makers intended. Z$(93) is a backspace which removes the sometimes unwelcome space preceding a numerical string, as in line 1110. I find this much simpler than a statement like RIGHT$(STR$(W),LEN(STR$(W))-1).

Not all of the characters in the letterset are used in this program but they are there for your convenience if you want to use it for your own programs. The letterset will give you a safe 27 characters per screen line. The value LL=27 is established in line 850. It can be varied, as in line 1120 where the backspace counts as a character but it makes room for two more.

You can enter as much text as a program line will carry by specifying the horizontal coordinate, B, and the vertical coordinate, C. Then enclose the text in a string labelled ZL$ and GOSUB130. If a text string is longer than twenty-seven characters, subroutine 130 will give you word wrap so that no words are split at the end of a line.

Line 20 sets the hi/lo speed poke values at SP=65497 and SL=65496 if you are using a CoCo 3, otherwise at SP=65495 and SL=65494.

The character set and subroutines are at the beginning of the listing, so you have all the space from line 840 on at your disposal if you want to use any of them for your own programs.



```
0 'MATCHEM'
1 'COPYRIGHT 1990 KEIRAN KENNY
10 CLEAR500
20 IFPEEK(&HFFFE)*256+PEEK(&HFFF
F)=&H8C1B THENSP=65497:SL=65496E
LSESP=65495:SL=65494
30 POKESP,0
40 X=RND(-TIMER)
50 DIM Z$(93),A$(8),T(8),D(8),PC
$(20)
60 GOTO200
70 B=135:C=96:ZL$="NO.]"+STR$(Z)
+" MATCHES!":GOSUB130:RETURN
80 B=150:C=96:ZL$="NO MATCH!":GO
SUB130:RETURN
90 FORD=1TO2000:NEXT:RETURN
100 C=C+15:ZL$="PRESS ANY KEY.":
B=INT(128-LEN(ZL$)*5):GOSUB130
110 K$=INKEY$:IFK$=""THEN110ELSE
RETURN
120 DRAW"BM"+STR$(B)+","+STR$(C)
:RETURN
130 IFLEN(ZL$)<=LL THEN170
140 FORT=LL TO1STEP-1:IFMID$(ZL$
,T,1)=" "THEN160
150 NEXTT:GOTO170
160 P$=LEFT$(ZL$,T):W$=P$:GOSUB1
80:ZL$=RIGHT$(ZL$,(LEN(ZL$))-T):
C=C+15:GOTO130
170 W$=ZL$
180 DRAW"BM"+STR$(B)+","+STR$(C)

190 COLOR0:FORZB=1TOLEN(W$):DRAW
Z$(ASC(MID$(W$,ZB,1)))+"BR4":NEX
T:RETURN
200 Z$(65)="U4E3F3D2NL6D2" 'A
210 Z$(66)="U7R5F1D1G1NL5F1D2G1N
L5BR2" 'B
220 Z$(67)="BRHU5ER4FDBD3DGNL4BR
" 'C
230 Z$(68)="U7R5FD5GNL5BR" 'D
240 Z$(69)="NR6U4NR4U3R6BD7" 'E
250 Z$(70)="U4NR4U3R6BD7" 'F
260 Z$(71)="BRHU5ER4FDBD2NL2D2GN
L4BR" 'G
270 Z$(72)="U7BR6D3NL6D4" 'H
280 Z$(73)="BR1R4L2U7NL2R2BR1BD7
" 'I
290 Z$(74)="BRHUDFR4EU6BD7" 'J
300 Z$(75)="U7D3R2NE3LF4" 'K
310 Z$(76)="NU7R6" 'L
320 Z$(77)="U7F3E3D7" 'M
330 Z$(78)="U7F6NU6D" 'N
340 Z$(79)="BRNR4HU5ER4FD5GBR"'O

350 Z$(80)="U7R5FD2GNL5BRBD3" 'P

360 Z$(81)="BRHU5ER4FD4G2L3RUERF
2" 'Q
370 Z$(82)="U7R5FD2GL5R2F3R1" 'R

380 Z$(83)="BRNHR4EU2HL4HUER4FBD
6" 'S
390 Z$(84)="BR4U7NL4R4BD7" 'T
400 Z$(85)="BRHU6BR6D6GNL4BR" 'U

410 Z$(86)="BU7D4F3E3U4BD7" 'V
420 Z$(87)="NU7E3F3NU7" 'W
430 Z$(88)="UE5UBL5DF5D" 'X
440 Z$(89)="BR3U2H3U2BR6D2NG3BD5
" 'Y
450 Z$(90)="BU7R5DG5DR5" 'Z
460 Z$(48)="BRHNE3U5ER4FNG3D5GNL
4BR" '0
470 Z$(49)="BR2R4L2U7NG2BR2BD7"
'1
480 Z$(50)="BU6ER4FD2GL4GD2R6" '
2
490 Z$(51)="BU6ER4FDGNL3FD2GL4NH
BR5" '3
500 Z$(52)="BR6U7L2G4R6D3" '4
510 Z$(53)="BRNHR4EU2HL5U3R6BD7"
'5
520 Z$(54)="BRHU3NE3D2ER4FDGNL4B
R" '6
530 Z$(55)="BU6UR6G6DBR6" '7
540 Z$(56)="BRHU2EHUER4FDGNL4FD2
GNL4BR" '8
550 Z$(57)="BRNHR4EU5HL4GD2FR4EB
D4" '9
560 Z$(32)="BR" 'SPACEBAR
570 Z$(33)="BR2UBU2U4BR4BD7" '!
580 Z$(36)="BUR3DNU6UR2EHL4HER5B
D5" '$
590 Z$(37)="BU5UERFDGLHBU2BR6DG5
DBR4REUHLGDFBR2" '%
600 Z$(39)="BRBU5NU2BD5BR" '
610 Z$(42)="BR3BU7D3NE2NR3NF2ND3
NG2NL3NH2BD4BR3" '*
620 Z$(43)="BR3BUU4D2NL2R2BRBD3"
'+
630 Z$(44)="NUNG" ',
640 Z$(45)="BU3BR2R2BR2BD3" '-
650 Z$(46)="BR2NU1BR4" '.
660 Z$(47)="UE5UBD7" '/
670 Z$(58)="BR3BUUBU3UBR3BD6" ':

680 Z$(59)="NGUBU4NUBD5" ';
690 Z$(60)="BR3H3E3BD6" '<
700 Z$(61)="BU2BR2R2BU2NL2BR2BD4
" '=
710 Z$(62)="BU6F3G3BR3" '>
720 Z$(63)="BR2UBU2UREUHL2GDBR6B
D5" '?
730 Z$(92)="BU7DF5D" '\
740 Z$(93)="BL10" ']=BACKSPACE
750 DATA U28R7D14E14R7D28L28
760 DATA U28R7F14U14R7D28L28
770 DATA U28F14U14R14D28L28
780 DATA U28R14D14E14D28L28
790 DATA U28R14D7F7U14R7D28L28
800 DATA U28R7D14E7U7R14D28L28
810 DATA U21E7D14R7U14R14D28L28
820 DATA U28R14D14R7U14F7D21L28
830 FORX=1TO8:READA$(X):NEXT
840 PMODE4,1:COLOR0,1:PCLS:SCREE
N1,1
850 LL=27:B=7:C=35:DRAW"S12":FOR
X=1TO4:ZL$="MATCHEM":GOSUB130:B=
B+2:C=C+2:NEXT:DRAW"S4":B=4:C=C+
15:ZL$=STRING$(22,42):GOSUB130
860 B=40:C=C+20:ZL$="BY KEIRAN K
ENNY":GOSUB130:B=50:C=C+20:ZL$="
SYDNEY - 1990":GOSUB130
870 B=35:C=C+40:FORX=1TO7STEP2:G
OSUB120:DRAWA$(X):B=B+40:NEXT:B=
35:C=C+40:FORX=2TO8STEP2:GOSUB12
0:DRAWA$(X):B=B+40:NEXT
880 K$=INKEY$:GOSUB100:PCLS
890 RD=RD+1:FORX=1TO8:T(X)=X:NEX
T
900 FORX=1TO8
910 E=RND(8)
920 IF T(E)=0 THEN 910
930 D(X)=E
940 T(E)=0
950 NEXT
960 B=20:C=45:FORX=1TO4:GOSUB120
:DRAWA$(D(X)):B=B+40:NEXT
970 B=26:C=10:FORX=1TO4:ZL$=STR$
(X):GOSUB130:B=B+40:NEXT
980 B=180:C=45:ZL$="NONE":GOSUB1
30:B=194:C=10:ZL$="N":GOSUB130
990 Z=RND(8):X=RND(3):B=95:C=96:
DRAW"A=X;":GOSUB120:DRAWA$(Z):DR
AW"A0"
1000 K$=INKEY$:B=0:C=145:ZL$="WH
ICH SHAPE MATCHES CENTER SHAPE?
CHOICE":GOSUB130
1010 GOSUB110:K=VAL(K$)
1020 IFVAL(K$)<1ANDK$<>"N"ORVAL(
K$)>4THENB=130:C=96:ZL$="1-4 OR
N ONLY."+STRING$(16,32)+"TRY AGA
IN.":GOSUB130:GOSUB90:COLOR1,1:L
INE(130,89)-(250,111),PSET,BF:GO
TO1010
1030 IFD(1)=Z ANDK=1GOTO1100ELSE
IFD(1)=Z ANDK<>1THENZ=1:GOSUB70:
GOTO1110
1040 IFD(2)=Z ANDK=2GOTO1100ELSE
IFD(2)=Z ANDK<>2THENZ=2:GOSUB70:
GOTO1110
1050 IFD(3)=Z ANDK=3GOTO1100ELSE
IFD(3)=Z ANDK<>3THENZ=3:GOSUB70:
```

Hints and Tips

==========

NOTE: THESE HINTS AND TIPS WERE WRITTEN WITH THE COCO 2 IN MIND. SOME OF THE POKES USED MAY NOT BE COMPATIBLE WITH THE COCO 3.

==========

1) Running some games? here are a few pokes which will let you cheat a bit and help you build up a better looking score:-

x = Number of men

Astro Blast	&H190F,x
Double Back	&H10E9,x
Electron	&H36C3,x
Frogger	&H22E2,x
Ghost Gobbler	&H2373,x
Lunar Rover	&H5761,x
Katerpillar	&H29F3,x
Mr. Dig	&H5439,x
Mudpies	&H441D,x
Whirlibird Run	&H2078,x
Trapfall	&H2CBC,x
Space Invaders	&H1D16,x
Planet Invasion	&H1D16,x
Zaxxon	&H6418,x

These pokes were supplied by Stephen Quinn.

****

2) When you use an ON...GOTO or ON...GOSUB statement you don'tneed to check for out of range values.
If a value is out of range, BASIC will skip it and jump to the next statement. For example:-

```
10 INPUT"YOUR CHOICE";D
20 ON D GOTO 50,70,80,90:GOTO10
```

If D is <1 or >4 the statement will be skipped and the next command (GOTO 10) will loop back to the input statement.

****

3) In a graphics programme which has a bug, it can help to indent SOUND commands at strategic points. If used in a scale it will be easy to find the area of your listing where the bug occurs.
This can be used instead of the TRON command which leaves the video display in text mode.

****

4) The Extended BASIC book tells us to always use the B option directly before the M motion command when moving the draw position. Their reasoning is that you may get unwanted lines. If you purposely omit the B option, you may get WANTED lines easier than with any other method. It is useful anytime you need to draw a line from a point to another point that does not fall on one of the standard angles (U E R F D G L H). The N (no update) option also seems to work well with this method.

****

5) Here is a Line Printer available test. PEEK(65314) If you get a 4 or 0 as a result, then the printer is ON line and available. If you get a 5 or 1, then it is OFF line.

This test should be incorporated into programmes requiring a printer to be on line. If the printer is not available, the computer locks up when it attempts to do a LLIST or PRINT#-2 function.

If no keys are pressed at the time of PEEKing,
Printer on-line = 1 Printer off-line = 0
If keyboard is in use at time of peeking,
Printer on-line = 5 Printer off-line = 4

NOTE: Decimal 65314 = HEX FF22 (&HFF22)

****

6) The use of the TIMER function can give a result in seconds, making it suitable to time operator response. Australia uses 50Hz Mains frequency, whereas in the USA 60Hz is used.
BASIC programmes written for the American market use the TIMER/60 function to approximate seconds. If this formula is used in Australian computers, slower times than normal will result.
You must convert the formula to TIMER/50 to suit.

****

7) In programmes where you need to draw shapes which are the same or similar you should put the shape or the repeated part of the shape in a string and then concatinate it. ie Join it on to the rest.

```
eg  A$="L12;D12;R12"
    F$="BM40,60;"+A$:DRAW F$
```

****

8) Clear memory to a COLD START without switching your computer off with:

```
POKE 113,0:EXEC 40999
```

****

9) TANDY does not mention in their instruction books that "?" can be used instead of the PRINT command.

****

10) Subtraction of decimals makes errors to the 6th decimal place. ie Try 30.20 - 30.0 and you get the answer 0.199999
To correct to the 6th decimal place:

```
C=INT(((A-B)*100)+.5)/100
```

****

11) The Hi-speed poke used in conjunction with the SOUND function will raise the pitch of that sound by precisely 130 tones. This increases the range of the SOUND function to be effectively 1 - 385 instead of 1 - 255.

****

12) The SHIFT and @ keys stop a listing scrolling. Any key will allow it to continue.

****

13) Here is a complete list of the editing subcommands in Extended Color Basic. For some reason, a few of these were left out of the Extended Basic Manual.

ENTER	Records all changes and returns to command mode.
n SPACEBAR	Moves 'n' spaces forward.
n left-arrow	Moves 'n' spaces back.
SHIFT up-arrow	Escapes from 'X', 'I' and 'H' insert modes.
L	Lists rest of line, places cursor at beginning.
X	Moves cursor to end of line and enters insert mode.
I	Enters insert mode at current cursor position.
A	Cancels changes already made, places cursor at beginning of line.
E	Exits edit mode with changes saved.
Q	Quits edit mode without changes.
H	Hacks off all characters after cursor, enters edit mode.
nD	Deletes specified number of characters at cursor. ('D' by itself deletes one character).
nC	Changes specified number of characters at cursor. ('C' by itself changes one character). 'C' is followed by the new characters you want to put in.
nSc	Searches for the nth occurrence of character 'c'.
nKc	Deletes all characters up to the nth occurrence of character 'c'.

****

15) When copying programmes from magazines use a highlight pen to mark the lines which have speed up pokes. Leave them out of your programme untill it has been fully debugged and is working properly. This will prevent you from saving parts of the programme in Hi-speed where it might be impossible to get them back.

****

16) The function STR$(number) which converts numeric expressions to strings also adds a space (ASCII 32) in front of the string. To eliminate the space:

```
N$=STR$(X):N$=RIGHT$(N$,1)
```

If X is to be of varying lengths, the second part of the line could read:

```
N$=RIGHT$(LEN$(N$),1))
```

****

17) 32 X 16 Screens only. Here is a useful little screen dump subroutine:

```
10000 ZZ=0
10010 FORXX=1024 TO 1535
10020 YY=PEEK(XX):ZZ=ZZ+1
10030 PP=YY AND 127
10040 IF PP>95 THEN PP=PP-64
10050 PRINT#-2,CHR$(PP);
10060 IF ZZ=32 THEN PRINT#-2:ZZ=0
10070 NEXTXX
10080 RETURN
```

****

18) The PCLEAR statement can inherit a number of pages from a previous programme if no new PCLEAR statement has been written. (This only applies if a cold start has not been executed). It is preferable to place a PCLEAR statement after CLEAR and PMODE.

****

19) To write scrolling games in BASIC.....Use Lo-res graphics and use a PRINT@511," "; to scroll the screen up one line.

****

20) To enable the second side of disks to be accessed in the all-RAM mode:

```
RS-DOS 1.0   POKE &HD7AC,65
             POKE &HD7AD,66
```

RS-DOS 1.1 POKE &HD895,65
            POKE &HD8A0,66
This is how the drives will then read:
1st DRIVE   Side 1 Drive #0
            Side 2 Drive #2
2nd DRIVE   Side 1 Drive #1
            Side 2 Drive #3
If your drives can step at 6ms track - to - track:
RS-DOS 1.1 POKE &HD7C0,0
            POKE &HD816,&H14
On some computers the DOS may start at &HE000 instead of &HC000. If so add &H2000 to the above figures.

****

21) The best method of finding bugs in a programme is to hand it over to your kids and let them play with it!!!

****

22) If your keyboard goes dead remove your joysticks. If this rectifies the problem check the fire buttons. You will probably find that one of them has stuck in.

****

23) When renumbering programmes your subroutines sometimes get lost. To remedy this situation place dummy lines at the end of your programme to call these subroutines. A REM after each line can describe which subroutine is being called.
On renumber these lines will always tell you where your subroutines reside.

****

24) 63999 is the highest line number you can use on your Coco.

****

25) Line too long? Sometimes, when you enter lines from a published listing, you can't get the last few items of a long line into your computer. The reason may be that the original line may have been "packed".
The BASIC monitor won't let you exceed 240 characters in a single line entry. If you need to get more into the line, just press ENTER to keep what you've already entered, and then EDIT that same line. Press "X" to move the cursor to the end of the line being edited, and you can now add those few more characters.
This trick does stretch your lines, but it often causes confusion when someone doesn't know about it as they type in a programme from a printed listing.

****

26) EXEC 35337 will renumber any basic programme in memory with an increment of 10. ECB only.

****

27) To slow down the screen printing speed,
    POKE 359,60
To restore to normal POKE 359,126
This feature is useful when LISTing a programme.
NOTE: This feature will not work with a DISK DRIVE.

****

28) If "G" and "O" keys do not work when in the Hi-speed poke then shift them and all should be well.

****

29) To find Machine Language addresses:
    PEEK(487)*256+PEEK(488) ⟨enter⟩
      This gives the start address.
    PEEK(126)*256+PEEK(127)-1 ⟨enter⟩
      This gives the end address.
    PEEK(157)*256+PEEK(158) ⟨enter⟩
      This gives the execute address.

This information is required to copy Machine Language programmes (i.e. Backup copies). For example:
SAVEM"DEMO",3584,16744,4466
START---------
END-----------------
EXEC----------------------

****

30) The following will centre a title or a string on a line:
xxx PRINT TAB((32-LEN(A$))/2)A$
Where A$ is the string to be centred.

By substituting 40 or 80 for the 32 this line can be used in the Coco 3 Hi-res text screens.

****

31) EXEC44539 waits for a keystroke and can be used as a substitute for:

```
xxx I$ = INKEY$:IF I$ ="" THEN xxx
```

****

32) IF - THEN statements can be reduced in a programme when the variables alter in a linear fashion. A list of IF - THENs can be cut down to one line. For example:

```
10 IF P = 9  THEN D = 1
20 IF P = 10 THEN D = 2
30 IF P = 11 THEN D = 3
      ETC.
```

This can be changed to:

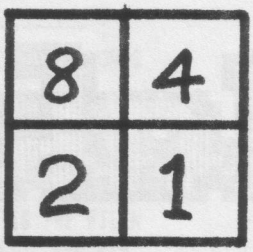
```
10 D = P - 8
```

Watch out for these situations. It saves a lot of memory and is better programming.

****

33) The numeric value of each section of ASCII block characters is shown in figure 1.
Starting Nos.

128 GREEN
144 YELLOW
160 BLUE
176 RED
192 BUFF
208 CYAN
224 MAGENTA
240 ORANGE



Add the starting number of the desired color to the values of the sectors you want to be coloured. For example, to make the upper left and lower right RED, use 176 + 8 + 1 = 185. Thus PRINT CHR$(185) will give you a character like this.

****

34) Here is the method to round a decimal number off to the nearest whole number:

```
INT(3.9 + .5) = 4
INT(3.4 + .5) = 3
```

****

35) Remember NEVER to put quotation marks inside a string like this:

```
xxx PRINT "MARY SAID "HELLO"
```

It will give you an error which is sometimes hard to find.

****

36) When drawing a straight line it sometimes handier to use the DRAW command instead of LINE. It uses less memory and lets you DRAW any colour without the COLOR command.

****

37) If you type in a programme from a magazine it is good practice to put a REM at the beginning of the programme to identify the publication it came from.

****

38) On a 64K machine the string space can be moved to the unused extra 8K. to do this:

```
POKE &H27,&HFE:POKE &H28,&HFF
```

Just make sure that you do not use more than 8K for yor strings.

****

39) RND(n) is not really random. On each start of a programme using RND the sequence starts from the same point quite often giving you the same sequence of numbers each time.
The solution is to start each programme with:

```
X = RND(-TIMER)
```

This sets the timer to a truly random number.

****

40) If you are entering a line of lowercase on screen (reverse video) and don't like the bright green space between words, use CHR$(128). This is a black graphic character.

****

41) If you would prefer to look at an orange text screen for a change then enter:

```
POKE 359.13:SCREEN0,1
```

Entering SCREEN0,0 will return you to the normal screen. This can be incorporated into a programme for a bit of variety.

****

42) The Hi-speed poke can be used to increase the range of the SOUND command. The Hi-speed poke raises the pitch of each SOUND by 130 tones giving an increase in range of 1 - 385 instead of 1 - 255.

****

43) If your Coco has 32K or 64K and you want to know if a certain programme will run on a 16K machine, type in:-

```
CLEAR200,16384
```

then load in your programme. If you get an OM? error then the programme takes up too much memory for 16K.

****

43) Sometimes a programme for your Coco will hang up for no reason. Even after all debugging procedures show that it has been copied in verbatim.
In this case look for these pokes:

```
POKE 1535,106
POKE 359,57
POKE 65314,20
```

Should they appear in your programme.....Delete them.

What these pokes do is access the hidden true lowercase in late model Coco 2's and all Coco 3's. If you have an earlier model Coco 2 or a Coco 1, they do not have this true lowercase and the pokes hang up the machine.

****

44) Radio Shack BW Hi-res Screen Print Utility (26-3121) may have a problem loading into a 64K Coco giving an FC error. If so, try CLEAR 200,31232 <ENTER> then

```
CLOADM"BWDUMP",16384
```

****

Menu Maker

By Paul Edmonds

I wrote the program MENUMKR some time ago after finding I was spending a lot of time formating print statements and setting up menus. This was time I could have used profitably on the main structure of the program and so the idea was born.

A lot of basic programs require some input from the user. This requested input often makes for a more user-friendly program if it is in the form of a menu. The program will then branch to a path dependent upon the user's selection. This program allows the programmer to quickly set up multiple menus which are saved on disk (ascii format) in the form of executable basic subroutines. A single menu subroutine allows up to nine selections to be made, however by stringing menus together one could have a theoretical menu containg over 200 selections.

Inputs requested from the programmer

1) The disk file name under which the menu subroutine will be saved on drive 0. Ensure that the disk in drive 0 is not write protected and note that an extension is not required for the filename.

2) A title for the menu. This title will appear centered and underlined when the user of the completed program is running a completed program.

3) The starting basic line number for the subroutine - the closing line number is calculated and printed to the screen. This facility allows for easy merging of the menu subroutines without overlapping line numbers.

4) Number of selections. This relates to the maxinum number of selections you may have in an individual menu.

5) Input descriptions. Input a brief (less than 3/4 line) description for each individual menu selection.

6) A single letter variable. Here the programmer can input any single letter A-Z which will not interfere with other program variables. The MENUMAKR program will use this variable as a flag within the subroutine so that the programmer can take the necessary action when a menu selection is made. For example if Z was the selected variable, then the menu subroutine would set a flag Z2 = 1 if selection 2 was made by the user of the menu. In a menu with 4 possible selections (Single choice menu), Z1,Z3 and Z4 would be set = 0. The programmer can thus utilise these flags to take the necessary path branching in his main program by using the form 'IF Z2= 1 then etc. etc.'.

7) Single choice menu (Y/N). This allows the programmer to specify (if <Y> is returned) that any choice is mutually exclusive. On the other hand there may be situations where a menu is required where multiple selections must be indicated. An example might be where you set up an educational program for young children. The menu could be entitled "Pick the animals" and the child would have to pick 3 animal selections while the remaining 6 selections relate to plants etc.

General

I wrote the program on a COCO2 and for this reason it does make use of the reverse video for lowercase letters to indicate a menu selection. Therefore COCO3 users should put their COCO3 into the COCO2 mode.

The algorithm I used to increment the basic line numbers was built up in a somewhat unwieldy fashion as I was merging uncompleted sections of the program together and could be written in a better fashion , however it does work and rather than start from scratch I have left it alone.

If someone has the time they may like to convert this for the COCO3 and to run in 80 character width with blinking characters for the selected menu choices.

I have found that the program is a useful utility which does save me considerable time and facilitates better structured and user-friendly programs.

MENUMAKER

```
10 GOSUB 5500'GET PRINT@ VALUES
60 CLS:PRINT@LA+10,"MENU CREATE"
;
62 PRINT@LB+10,"---- ------";
63 PRINT@LD+7,"PAUL T.J. EDMONDS

64 PRINT@LG+7,"WINMALEE N.S.W.
70 GOSUB 5000'KEY PSE
100 CLS:CLEAR 500
105 PRINT"THIS ITEM IS THE FILEN
AME       OF YOUR BASIC PROGRAM
WHICH     WILL BE WRITTEN TO DIS
K. IT     WILL BE AN ASCII FILE
          (A SUBROUTINE) WHICH
CAN       BE MERGED WITH YOUR MA
IN        PROGRAM.":PRINT
110 INPUT"DISK FILENAME";F$:F$=F
$+"/BAS"
112 OPEN "O",#1,F$
115 A1$=""
130 CLS
132 PRINT"THIS ITEM IS THE TITLE
 OR       HEADING OF YOUR MENU A
S IT      WILL APPEAR OVER YOUR
CHOOSEN   MENU SELECTIONS.":PRIN
T
135 INPUT"MENU TITLE";T$
137 CLS:PRINT"THIS ITEM IS THE B
ASIC          STARTING LINE NUMB
ER OF YOUR    SUBROUTINE THAT YO
U WILL MERGE  WITH YOUR MAIN PRO
GRAM.         ENSURE THEREFORE T
HAT LINE      NUMBERS DO NOT OVE
RLAP.":PRINT
138 INPUT"START NO.-SUBROUTINE";
S
141 CLS:PRINT"THIS ITEM IS THE N
UMBER OF      OPTIONS YOU WANT T
O APPEAR      BENEATH YOUR MENU
TITLE.        BECAUSE OF LIMITED
 SCREEN       SIZE, NOT MORE THA
N 9 SHOULD    BE INPUT. IF MORE
REQUIRED ,    YOU COULD STRING M
ENUS TOGETHER ":PRINT
150 INPUT "HOW MANY SELECTIONS";
N
155 CLS:PRINT"NOW INPUT YOUR CHO
OSEN          DESCRIPTIONS. TRY
TO LIMIT      LENGTH TO LESS THA
N 3/4         OF SCREEN WIDTH.":
PRINT
160 FOR K=1 TO N
170 PRINT K;:INPUT"";D$(K)
180 NEXT K
190 GOSUB 6000'REVERSED SELECTIO
NS TO STRING VARIABLES
200 GOSUB 6100'LONGEST SEL=L1
205 CLS:PRINT"YOU MUST NOW SELEC
T A LETTER    VARIABLE. THIS WIL
L BE USED     IN THE SUBROUTINE
AS A FLAG     SO THAT APPROPIATE
 ACTION       CAN BE TAKEN IN YO
UR MAIN       PROGRAM, DEPENDING
 ON THE       CHOICE(S) MADE IN
MENU."
210 PRINT"INPUT A SINGLE LETTER
TO BE":PRINT"USED AS A VARIABLE
FOR FLAGS":PRINT"E.G. IF 2 IS SE
LECTED ON MENU":PRINT" & Z IS TH
E VARIABLE CHOOSEN":PRINT"THEN Z
2 IS SET FROM 0 TO 1":PRINT:INPU
T"VARIABLE";V$
220 FOR K=1 TO N:J$=STR$(K):O=AS
C(MID$(J$,2,1)):V$(K)=V$+CHR$(O)
:NEXT K' IF J$ IS USED FOR ASC.
ONLY SPACE IS RETURNED.
223 CLS:PRINT"YOUR MENU CAN ALLO
W EITHER      MULTIPLE CHOICES
OR YOU        CAN RESTRICT IT TO
 A  SINGLE    CHOICE.":PRINT
226 PRINT"SINGLE CHOICE ONLY-(Y/
N)";:INPUT A1$
300 GOSUB 7000'WRITE MENU PROG.
350 CLS:INPUT"DO YOU WANT ANOTHE
R MENU (Y/N)";A1$:IF A1$="Y" THE
N 115
4998 CLOSE #1
4999 END
5000 REM*** TO PAUSE FOR KEY PRS

5010 FOR K=1 TO 2:FOR T=1 TO 100
:NEXT T:PRINT@458,"(any key)":FO
R T=1 TO 100:NEXT T:PRINT@458,"(
ANY KEY)":NEXT K:EXEC 44539:PRIN
T @ 458,"          ":RETURN
5500 REM SUB TO GET PRNT@VALUES
5510 LA=0:LB=32:LC=64:LD=96:LE=1
28:LF=160:LG=192:LH=224:LI=256:L
J=288:LK=320:LL=352:LM=384:LN=41
6:LO=448:LP=480
5530 RETURN
6000 REM***CHANGE TO COCO2 LOWER
          CASE
6010 CLS
6020 FOR K=1 TO N
6021 N$(K)=D$(K)
6022 L=LEN(N$(K))
6024 FOR P=1 TO L:F$=MID$(N$(K),
P,1):A=ASC(F$):IF A>=65 THEN A=A
+32: :MID$(N$(K),P,1)=CHR$(A)
6028 NEXT P
6040 NEXT K
6050 RETURN
6100 L1=14:FOR K=1 TO N '***LEN
     14 IS MIN. VALUE ("QUIT THI
     S MENU") FOR PRINT@ +SEL.NO
     OR LETTER.
6110 L=LEN(D$(K)):IF L>L1 THEN L
1=L
6120 NEXT K
6130 RETURN
7000 CLS:PRINT@LL," NOW WRITING
YOUR BASIC PROGRAM";
7005 L2=LEN(T$):L2=INT(L2/2):L3=
16-L2
7030 PRINT #1,S;" REM MENU ";T$;
" SUBROUTINE *****"
7031 PRINT #1,S+5;" CLS"
7035 PRINT #1,S+10;" PRINT@";L3;
",";CHR$(34)+T$+CHR$(34);";"
7040 PRINT #1,S+20;" PRINT@";32+
L3;", STRING$(";LEN(T$);",";CHR$
(34)+CHR$(45)+CHR$(34);");"
7045 PRINT #1,S+30;:FOR K=1 TO N
:PRINT #1,V$(K);"=0:";:NEXT K
7050 PRINT #1,""
7090 FOR K=1 TO N
7100 PRINT #1,S+40+(K*10);" PRIN
T@";65+(K*32);",";CHR$(34)+D$(K)
+CHR$(34);":PRINT@";65+(K*32)+L1
+1;",";CHR$(34)+"("+CHR$(34);K;C
HR$(34)+")"+CHR$(34);";"
7150 NEXT K
7160 PRINT #1,S+40+(N+2)*10;" PR
INT@";65+((N+1)*32);",";CHR$(34)
+"CORRECTION"+CHR$(34);":PRINT@"
;65+((N+1)*32)+L1+1;",";CHR$(34)
+"( C )"+CHR$(34);";"
7170 PRINT #1,S+50+(N+2)*10;" PR
INT@";65+((N+2)*32);",";CHR$(34)
+"QUIT THIS MENU"+CHR$(34);":PRI
NT@";65+((N+2)*32)+L1+1;",";CHR$
(34)+"( Q )"+CHR$(34);";"
7180 Z=S+60+(N+2)*10
7183 IF A1$="Y" THEN PRINT #1,Z-
5;"IF XX=1 THEN ";Z+10
7185 PRINT #1,Z;" Q1$=INKEY$"
7186 X=Z+(N+2)*10
7190 PRINT #1,Z+10;" IF Q1$="+CH
R$(34)+CHR$(34)+" THEN ";Z
7195 FORK=1TON:PRINT#1,Z+20+(K*1
0);"IF Q1$=";CHR$(34)+MID$(STR$(
K),2)+CHR$(34);"THEN PRINT@";65+
(K*32);",";CHR$(34)+N$(K)+CHR$(3
4)+CHR$(59);":IF Q1$=";CHR$(34)+
MID$(STR$(K),2)+CHR$(34);"THEN";
V$(K);"=1"
7200 NEXT K
7205 IF A1$="Y" THEN PRINT #1,Z+
30+(N+2)*10;"IF ";V$(1);:FOR K=2
  TO N:PRINT #1,CHR$(43)+V$(K);:
NEXT K:PRINT #1,">1 THEN XX=1 EL
SE XX=0"
7208 IF A1$="Y" THEN PRINT #1,Z+
```

Fourth Generation Languages and OS9

By Ole Eskilsden

Part 2. Sculptor

This is the second part of my three part article on 4GLs. In this part I will describe one of the 4GLs available under OS-9, namely SCULPTOR. Sculptor was developed in the UK by Microprocessor Developments Ltd and has been available for some time for OS-9. Under OS-9 it is a true multi-user system with efficient (and automatic) record locking. The programs are compiled which results in faster running programs. Besides OS-9, Sculptor is available for a great many other operating systems including MS-DOS (single-user or networked multi-user), UNIX and XENIX, DEC VAX, QNX and many, many others. Not only that, but it is object code compatible, which means that a program compiled on ANY version will execute on ANY OTHER version WITHOUT recompilation. Can you imagine: You develop a program or a complete system in Sculptor for instance on a CoCo3 (Sculptor is only available for OS-9 Level 2) and then copy it to a UNIX machine or MS-DOS or whatever and it is instantly ready to run. Of course to achieve this result it is necessary to have a run time version of Sculptor on the target machine because Sculptor compiles its programs into an intermediate object code (I-code) much like BASIC09.

So what do you get when you purchase a copy of Sculptor Development System? You get a good looking A5 size manual in a 4-ring binder in a slip case in the by now familiar PC tradition containing a couple of hundred pages. There are actually three manuals:

1) Installation Manual.
2) An Introduction to Sculptor.
3) Sculptor Reference Manual.

You further get two diskettes with the system and utilities, and the Australian Distributor has added a third diskette with a collection of useful little programs and numerous samples. I found the Sculptor Introduction very easy to go through and after spending only a short time (a coulple of hours) I was able to start experimenting with the language, and what a delight this turned out to be.

But first, what modules does the Sculptor Development System consist of? It consists of:

1) A Data Dictionary.
2) Compilers for interactive Screen (also called Forms) type programs and Report type programs. These are called CF and CR respectively.
3) The run time modules to execute the programs compiled with CF and CR, called Sage (for screen type programs) and Sagerep (for report type programs).
4) Two programs which will automatically generate complete screen or report type programs from your data dictionary specifications. These are called SG for screen generator and RG for report generator.
5) A Menu program to automatically generate menus.
6) Various utility programs to perform file handling tasks, and to specify different screen and printer types, and finally a Query program to perform ad-hoc queries and reports. In other words, a complete development and run time environment with facilities to run on many different computers, terminals and printers without reprogramming.

We will now take a closer look at how all this will work for you. The best way, I think, would be by describing an example, so I will now describe the typical work sequence in a simplified form.

1. Systems Design - You define what the system is going to accomplish.
2. Data Definition - What data do you need to store.
3. Development - Develop the programs, compile and test.
4. Production - Use the program(s).

To use a simple and well worn example. Let us say we want to develop an address book. What data would we need? Probably Name, Street, Town, State, Postcode (zip code for our American friends), telephone number, etc, etc. So

we decide to create a file called address, which we do by using the dictionary program called "Describe", we simply type: "describe address". We now see:

```
Descriptors for address

  For each field enter:
  name,heading,type&size,format;validation
  Type h for help.

  KEY FIELDS
    1:
```

We enter the information about each field, selecting one or more fields to hold the key which must be unique. When we are finished, the record description might look like this:

```
Descriptors for address

  KEY FIELDS
    1:name,NAME,a20,

  DATA FIELDS
    2:street,STREET,a20,
    3:town,TOWN,a10,
    4:state,STATE,a3,
    5:postcode,PCODE,a6,
    6:phone,TELEPHONE,a10,
```

We save the record description which creates a file called "address.d" and we can now generate a standard Screen Form program by typing:

```
sg address
```

and in less than one minute we have a program where the source program is called "address.f" and the compiled program is called "address.g". Before we can execute the program we have to create the actual data file and its associated key file called "address" and "address.k" respectively. This is done by typing "newkf address" and in seconds the empty data file is created. The Screen Form program we created presents the following screen when we type: "sage address", where "sage" is the Screen Form program run-time interpreter:

```
                    "ADDRESS"FILE MAINTENANCE
                                                Today's date [        ]

             NAME [                    ]

                 STREET [                    ]
                   TOWN [          ]
                  STATE [   ]
                  PCODE [      ]
              TELEPHONE [          ]

 i=insert  f=find  n=next  p=prev  m=match  a=amend  d=delete  e=exit

                  Which option do you require?
```

Neat, don't you agree? The second last line shows all the options, such as inserting records, finding records, amending records, and so on. All without any programming. Now go ahead and enter some records, find them again, amend them, delete them, etc.

Now that we have got some data in our file, let us create and print a report. We type: "rg address" and a moment later we have a source program called "address.r" and a compiled object program called "address.q". To run the report we type "sagerep address" where "sagerep" is the run-time interpreter, but you had figured that out, hadn't you?

This is what the report would look like:

"ADDRESS" FILE REPORT

Page: 1 Date: 27/ 9/89

NAME	STREET	TOWN	STATE	PCODE	TELEPHONE
Bentzen, Gordon	8 Odin Street	Sunnybank	QLD	4109	07-3455141
Berrie, Don	25 Irwin Terrace	Oxley	QLD	4075	07-3753236
Devries, Bob	21 Virgo Street	Inala	QLD	4077	07-3727816
Eskildsen, Ole	11 Monarch Street	Kingston	QLD	4114	07-2094322

END OF REPORT

How do you like that? In less than an hour we have created a complete little system td create, store and manipulate our address book and report on it!

Are you begining to see the power of 4GLs? I hope so, otherwise I have failed in describing the benefits to you. Remember, the source programs generated can be modified by the programmer, so what I do most of the time, is to generate the standard program first and then use that to develop the final product. In the extremely simple example above we have only used one file, but you can, of course, use multiple files. You, the programmer, have complete control over everything going on, if you desire. On the other hand, it only takes minutes to generate simple applications. Another point which shows the power of Sculptor: When the Screen Form program is being compiled, the compiler examines the way files are used in the program. If a file is being updated in any way (such as in the example above) then Sculptor creates the necessary record locking, so that two persons cannot update the same record at the same time by accident. Ask the DBase (IBM PC database) programmers if they can do that automatically, or whether they have to meticulously write the record locking every time. If a file is not used for updating then no record locking is being performed. Pretty powerful stuff!

Who is using Sculptor, you might ask? Well, quite a number of individuals and organizations. For instance, Australian Telecom, a Queensland Government Department, numerous software houses, and many others.

The above is only a very small, simple example, space does not permit any more. In Part 3 I will discuss a more complex example and make you an offer you cannot refuse, I hope. Address any correspondence concerning these articles to:

Ole Eskildsen
11 Monarch St
Kingston QLD 4114
Phone: (07) 209 4322

(Reprinted with permission from the National OS9 Users Newsletter).

Matchem

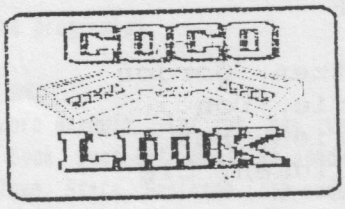
Listing Continued:

```
1050 GOTO1110
1060 IFD(4)=Z ANDK=4GOTO1100ELSE
IFD(4)=Z ANDK<>4THENZ=4:GOSUB70:
GOTO1110
1070 IFD(X)=Z ANDK=N GOTO1110
1080 IFD(1)<>Z ANDD(2)<>Z ANDD(3
)<>Z ANDD(4)<>Z ANDK=1ORK=2ORK=3
ORK=4GOSUB80:GOTO1110
1090 IFD(X)<>Z ANDK=N GOSUB80
1100 R=R+1:B=132:C=160:ZL$=K$+"
IS RIGHT!":GOSUB130:GOTO1120
1110 W=W+1:B=132:C=160:ZL$=K$+"
IS WRONG!":GOSUB130
1120 P$=STR$(R*100/(R+W)):P$=LEF
T$(P$,LEN(STR$(INT(R*100/(R+W)))
)+3):B=0:C=175:ZL$=STR$(R)+" RIG
HT:]"+STR$(W)+" WRONG:]"+P$+"%":
LL=29:GOSUB130:LL=27
1130 RS=RD/10:IFRS=INT(RS)THENPC
$(RS)=P$
1140 IFRS=INT(RS)ANDINT(RS)>1THE
NGOSUB100:PCLSELSE1220
1150 B=10:C=45:ZL$="SCORE, END R
OUND"+STR$(RS-1)+":"+PC$(RS-1)+"
%":GOSUB130
1160 B=10:C=C+30:ZL$="SCORE, END
 ROUND"+STR$(RS)+":"+PC$(RS)+"%"
:GOSUB130
1170 PV=VAL(PC$(RS-1)):PR=VAL(PC
$(RS)):IFPR>PV THENPG$="YOU'RE I
MPROVING!"ELSEIFPR=PV THENPG$="N
O CHANGE!"ELSEIFPR<PV THENPG$="Y
OU'RE SLIPPING!"
1180 C=C+30:ZL$=PG$:B=INT(128-LE
N(PG$)*5):GOSUB130
1190 C=C+30:ZL$="<E>ND OR <C>ONT
INUE?":B=INT(128-LEN(ZL$)*5):GOS
UB130
1200 GOSUB110:IFK$<>"E"ANDK$<>"C
"THEN1200
1210 IFK$="E"THEN1240ELSE1230
1220 GOSUB100
1230 PCLS:B=0:GOTO890
1240 CLS:POKESL,0:END
```

Menumaker

Listing continued:

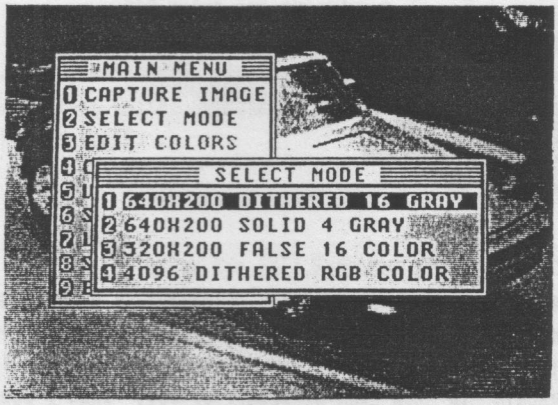
```
35+(N+2)*10;"IF XX=1 THEN";S+30
7210 PRINT #1,Z+40+(N+2)*10;" IF
 Q1$=";CHR$(34)+"C"+CHR$(34);"TH
EN GOTO";S+30;"
7250 PRINT #1,Z+50+(N+2)*10;"IF
Q1$=";CHR$(34)+"Q"+CHR$(34);"THE
N RETURN
7260 PRINT #1,Z+60+(N+2)*10," GO
TO ";Z;"'******END ROUTINE
7300 CLS:PRINT@LA+1,"LAST LINE N
O.=";Z+60+(N+2)*10:FOR A=1 TO 15
00:NEXT A
7400 RETURN
```



AVAILABLE SOON!

RASCAN

VIDEO DIGITIZER



Capture images from a video camera or home VCR (with a good pause-still).

Transfer images to CoCoMax3, ColorMax3 or MV-Canvas for further editing.

Features include:

* No multi-pak rqd. Plugs into joy ports
* Supports full CoCo3 Hi-Res display.
* Slide show program included.
* Easy to use Pop-up-Menu system.
* Made and designed in Australia.

WRITE TO US FOR A FREE DEMO/INFO DISK.

OTHER PRODUCTS:

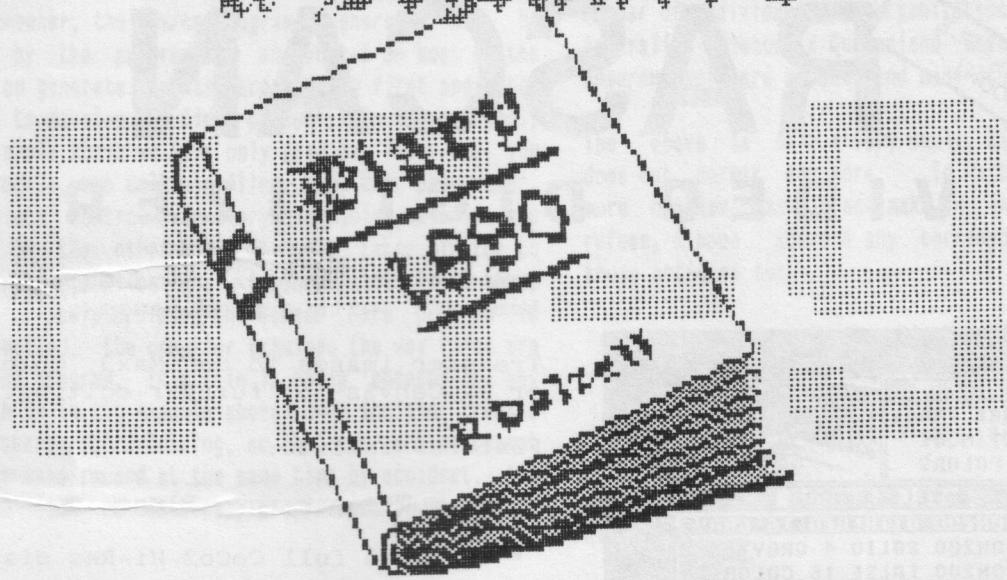
* MV-Canvas Ver.2 (OS-9 Paint program. Rqrs OS-9 & Multiview)...AVAILABLE SOON
* RAS*MAX (Print 4096 colour RASCAN images on CGP-220 & STAR NX-1000)...$20.00
* CGP*MAX (Print CM3, GIF, MGE or HSCREEN2 pictures on CGP-220).....$15.00
* STAR*MAX (As above but for STAR NX-1000).....AVAILABLE SOON
* Z'89 (New CoCo3 version of Zaxxon.).....ONLY 5 COPIES LEFT!..$30.00
* PYRAMIX (Like arcade Q-BERT.).....ONLY 4 COPIES LEFT!..$20.00
* DONUT DILEMMA (10 level platforms game).....SPECIAL...$6.00
* RUPERT RYTHYM (16 level strategy/arcade platform game).....SPECIAL...$6.00
* SPACE INTRUDERS (an old classic re-vitalised).....SPECIAL...$6.00

NOTE - ALL PRODUCTS ARE ONLY FOR THE COCO3 WITH DISK DRIVE EXCEPT DONUT DILEMMA (CoCo 1/2/3, disk or tape) AND RUPERT RYTHYM (CoCo 3, disk or tape).

Cheque or postal money order only

NICKOLAS MARENTES P.O. BOX 6551 UPPER MT.GRAVATT QLD 4122

A Beginner's



The project is to get two seperate programmes to run at the same time under the OS9 windows system.

My first move was to back up the 40 Track DS System disk. I have learned from experience never to experiment with "one-only" disks. This is fatal.
This done, the next move is to transfer the Dynacalc programme from the 35 Track SS Tandy disk and put it on my System disk.

As I was instructed previously (Part 1), I put the Dynacalc disk in /d1 and the System disk in /d0. I then typed on the OS9 prompt:

copy /d1/cmds/dynacalc /d0/cmds/dynacalc <enter>

When this was done I tried to load the programme with:

OS9: dynacalc

After the usual disk whirring I received the equally usual ERROR #216 - Pathway not found.

I ran a 'dir' on /d0 and found that dynacalc did not appear in the root directory. I had thought that the above command would have put it there.
Using the 'dir x' command I found it in the execution directory (The x directs the dir command to the execution directory).
After some advice I looked at the dynacalc disk in /d1 and found that, as well as the file dynacalc (in the execution directory), there was a file DYNACALC.TRM in the root directory.
This file obviously has to be transferred as well.
I tried transferring it by using the command above, substituting 'dynacalc.trm' and as usual got the ERROR #216 on trying to run the programme.
On reflection, I think that it puts the file in the execution directory for the following reasons:

The 'copy /d1/cmds/dynacalc' shows the pathway to the dynacalc file. Look for dynacalc in the CMDS directory of /d1. The second part of the command says to copy it to dynacalc in the CMDS directory of /d0. When I used the same commands with dynacalc.trm I told the system to copy it to the CMDS directory of /d0, which is the current execution directory, and not to the current data directory as I should have. Therefore I changed the command to read:

copy /d1/dynacalc.trm /d0/dynacalc.trm

and voila! It transferred to the data directory and, on the 'OS9: dynacalc' command, the programme was up and running.

On checking /d0 with 'dir x e' (x gets the execution directory and the e gives added information), I now find that I have a surplus 'dynacalc.trm' file in the execution directory which is using up 3200 bytes. I will have to delete it.

A look through the commands list shows that there are two delete commands DEL and DELDIR. DEL is for the deletion of single files and DELDIR deletes whole directories.
The use of -x with the DEL command causes it to assume the file is in the current execution directory. So.....

OS9: del -x dynacalc.trm <enter>

Success! The file has been deleted from the directory.

The above problems show up the need to be fully conversant with the directory system in OS9, what the different types of files are and where they are stored.

****

Next, I intend to run Dynacalc and Basic09 from separate windows. For this exercise I will load Basic09 from /d1 while the System/Dynacalc disk will be in /d0. I will do this by writing a startup file to automatically set up windows and load the programmes from the DOS command.

I am using the "Rainbow Guide to OS9 Level 2" as a guide to assist me. Page 69 lists a startup file (Procedure in OS9 language) on the lines I am looking for. This is followed on page 71 with a 'startapps' procedure to load and run the programmes.

There are already some of the things required in the existing startup file on my system disk. I read this by listing it to the printer.

```
OS9: list startup >/p
```

I am going to attempt to change this file to include the necessary windows and load procedures. I will read up the "Macro Text Editor" section of the OS9 manual before making a start.

****

I have read up on the EDIT function and am keeping the manual open at the ready, So here goes.....

```
OS9: edit startup <enter>
```

****

Using the EDIT system to add, insert and delete lines worked well after I got used to the many little tricks put there to catch you out. For instance, inserting a line means inserting it in front of the last line you are looking at. For example:

```
E: date t
E:
```

The line I type in at the E prompt will be inserted in front of the 'date t' as will any line typed immediately after it. Also you must be careful to put a space before any characters for lines to be included in the procedure file. Commands are put in without first typing the space bar.
I do not feel that the supplied editor is the best but until I find something better it will have to do.

After a fair amount of trial and error, plus much consultation with the manual, I finally managed to produce the following startup procedure. I even remembered to change my current directories for loading Basic09 from /d1.

```
* Echo welcome message
echo * welcome to OS9 *
echo * on the Color Computer 3 *
echo * Robbie's first startup file *
* lock shell and std utils into memory
link shell
display a a
* set time from keyboard
setime </term
date t
echo setting monitor type
montype c
* initializing screen
iniz w7
echo Merging fonts to /w7
merge /dd/sys/stdfonts >/w7
shell i=/w7&
* Now to load our applications
*
echo loading dynacalc
load dynacalc
* as we will be loading Basic09 from /d1
* we must first change the current directories
*
chd /d1;chx /d1/cmds
echo loading RunB
load runb
* this is required if we want to run
* a packed basic09 programme
*
echo loading basic09
load basic09
echo Type CLEAR for 80 columns
```

I ran the file and all went well until it reached the SHELL command. At this it printed 'shell' on screen then displayed the OS9 prompt. It did not load any of the programmes into memory. I tried several times but to no avail.

I was unable to find any reason for this or contact anyone who might have been able to help me so I decided to procede with the 'startapps' file.

****

Using the EDIT function was easier this time although I realised later that I really should have used the BUILD command to write the file from scratch.
The file opens two windows and loads Basic09 from /d1 into one and Dynacalc from /d0 into the other.

```
iniz w6
merge sys/stdfonts >/w6
display 1b 20 7 0 0 50 18 2 0 2 >/w6
chd /d1;chx /di/cmds
basic09 #20K <>>>/w6&
iniz w5
display 1b 20 2 0 0 50 18 1 7 3 >/w5
chd /d0;chx /d0/cmds
dynacalc <>>>/w5&
```

I ran this file from the OS9 prompt and had the pleasure of seeing the programmes appear at the press of the CLEAR key. One problem was that the Basic09 programme showed only rows of dots instead of characters. It was then I added the 'merge sys/stdfonts' line. The stdfonts must be

merged to a graphics window or all characters appear as full stops.

I now have a startup file which initializes /w7 with a shell (this allows all the OS9 modules to be called and gives a screen to home to in case of trouble). On the OS9 prompt I can load my 'startprogs' file and get the two programmes up and running.
I am pleased.....but that is not really what I wanted. I want to put the disk in the drive, type DOS and have my two programmes automatically there in windows ready for use.

So now I will attempt to join the two files into one and have a startup file that does what I want.

****

Using the EDIT function again I have rewritten the startup file. After saving it to the disk using the 'q' command I shut down the computer and started from DOS. All systems go! I now have the two programmes running in windows /w5 and /w6 and a shell running in /w7.

My final startup procedure file looks like this:

```
* Echo welcome message
echo * welcome to OS9 *
echo * on the Color Computer 3 *
echo * Robbie's first startup file *
* lock shell and std utils into memory
link shell
display a a
* set time from keyboard
setime </term
date t
echo setting monitor type
montype c
* Initializing screens
iniz w6
merge sys/stdfonts >/w6
display 1b 20 7 0 0 50 18 2 0 2 >/w6
chd /d1;chx /di/cmds
basic09 #20K <>>>/w6&
iniz w5
display 1b 20 2 0 0 50 18 1 7 3 >/w5
chd /d0;chx /d0/cmds
dynacalc <>>>/w5&
iniz w7
echo Merging fonts to /w7
merge /dd/sys/stdfonts >/w7
shell i=/w7&
```

I have left off the LOAD to memory part of the original procedure until I find out what is to be gained from it. I will also have to find out why my original procedure would not function fully.

Having used the 'Rainbow Guide' book in compiling this procedure, there are a few items I had to read up on to understand. There are quite a few that I still don't understand but I will get there eventually.

'display a a'. This had me puzzled until I realised that the visual display misses two lines on this command. Therefore it was safe to deduce that 'a' is 'A' HEXADECIMAL, which in turn is DECIMAL 10, which in turn is ASCII for a line feed.
'echo' This is the equivalent of Basic's PRINT. It is a good idea to put information lines on screen to indicate what the computer is up to at a given time.
'display 1b 20' etc. This line formats the window size and colours. It is explained in the Windows section of the OS9 manual. I understood the last eight numbers but couldn't work out the '1b 20' part. The Rainbow Guide tells me that this has a name. It is DWSET - or Device Window set. Other similar codes are:

1b 24	DWEND	Display window set
1b 22	OWSET	Overview window set
1b 23	OWEND	Overview window end

These and more codes for use with the display command are described under General Commands in the Windows section of the manual.
WCREATE, which uses decimal numbers instead of the hexadecimal required by DISPLAY, can be substituted for the display command lines. I think I will use it when I compile my next procedure file.
Just one other little thing to remember. If you quit a programme intentionally or unintentionally, you will end up with a blank window. In this case the programme has to be restarted from the shell. ie The green and black window with the OS9 prompt. For example:

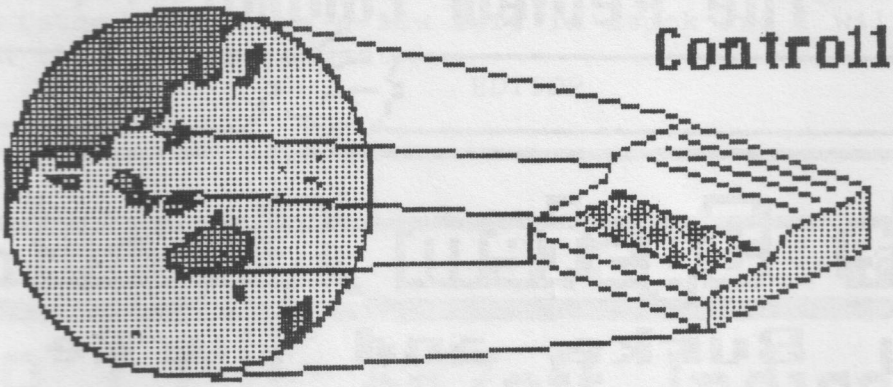
Having quit dynacalc press CLEAR till back to shell.
OS9: dynacalc <>>>/w5& <enter>
Dynacalc should be back in /w5

Another little point. If the data hasn't been saved it will be lost.

It is a good feeling to have OS9 doing something that I planned. Now back to the books.

"Do you suppose it knows something we don't?"

An Introduction to "OUTSIDE WORLD" Controlling



By Darren (Gonzo) Ramsey

In last month's article, I stated that I would base a series of hardware articles around the energy management control system, (E.M.C.S.) I have at home. I should have said "had at home." We (my wife and I) have sold our present home and plan to build a new one in greener pastures.

What this means is that I shall "invent" an entirely new E.M.C.S. for the new home. Although this won't use a Coco, all concepts remain the same. So CoCo and non CoCo users should gain from this information. I shall keep it as easy to read as possible, but it must be kept in mind that a technical subject like this needs to remain as such if it is to be useful to those who wish to use it.

Throughout these articles I shall use extracts from a report I wrote in June 1989 whilst doing my digital techniques certificate at Regency T.A.F.E., titled "Micro Computer Interface through P.P.I".

WHAT IS AN E.M.C.S.?
****

It's an Energy management Control System, a fancy name I dreamt up to describe a bunch of computer controlled switches and sensors to automate some of the more boring and time consuming tasks about the home. In doing so, I aim to save money by more efficiently using electricity and water.

In its prime, it controlled my irrigation system of some seven water circuits, chlorinated and filtered my swimming pool water, and was 9/10ths ready to control the household air-conditioning, emergency lighting system, (for those many blackouts during winter), and monitor the burglar alarm. With thoughts to sell up, the 9/10ths portions came to a sudden halt, but as I shall install them in the new home, I shall describe them for those interested.

OVERVIEW OF COMPUTER EQUIPMENT.
****

There are two sides to the equipment story, that which is essential (bare bones stuff), and that which makes life easy.

* ESSENTIAL - CoCo computer 64k E.C.B. (Minimum), cassette recorder or disk drive, T.V., P.I.A. interface, external equipment driver.
* WHAT I USE - CoCo 2, 256k E.C.B., dual 40 track disk drives, composite mono monitor, audio amplifier, parallel printer, remote terminal, P.I.A. interface, external equipment driver and assorted other junk.

All the extra stuff aids the E.M.C.S., but is not essential, when I started the system I had only the essential gear.

PROGRAMMABLE INTERFACE ADAPTOR.
****

As there are many possible ways to interface a P.C. to the "Outside World", I have chosen the pure digital approach of using a Programmable Interface Adaptor (P.I.A.). Throughout, I also make cross reference to a Programmable Peripheral Interface (P.P.I.), which follows a parallel concept. The P.I.A., which will be discussed later, is a programmable I/O port device whose sole purpose of being is to make the task of I/O to a microprocessor system as simple and flexible as possible.
I shall delve into the theory (based on the Motorola 6281 P.I.A. I.C.) and concept of P.I.A. usage, and provide practical examples of its use.
For the purposes of testing, I have built an I/O simulator unit into which output signals from the P.I.A. can be fed. Also, input signals required by the P.I.A. can be generated by this unit.

In my next article I shall discuss the 6821 P.I.A.

Chain Reaction

The Review Column

File System Repack

By Burke and Burke

An overview by Jacques Laporte

This article has been translated from the original French and is reprinted with permission from LE COCOTIER, the magazine of 'Les Cocophiles' Color Computer Club of Quebec, Canada.

The SYSTEM FILE REPACK by Burke and Burke is a collection of eight utilities for OS9 disk.

Here is a presentation of the eight commands:

BA: Puts in order the BITS of a certain number of logical sectors (LSN) in the Allocation Table.

BD: Removes the BITS of a certain number of logical sectors in the Allocation Table.

CCHECK: Runs through the disk in search of defective sectors and identifies the programmes which use these sectors.
The command verifies the integrity of each sector of the disk.

HDB: Copies a complete Hard disk to floppy disks with the potential to compress the data.
(BACKUP of hard disk to Floppy disks).

HDR: Recoup the data safegaurded on the floppy disks with the HDB command and recreate the hard disk.
(RESTORE the hard disk).

REPACK: Reorganise the diskettes to the hard disk and eliminate fragmentation of the data.
(This is the main programme of the 8 utilities).

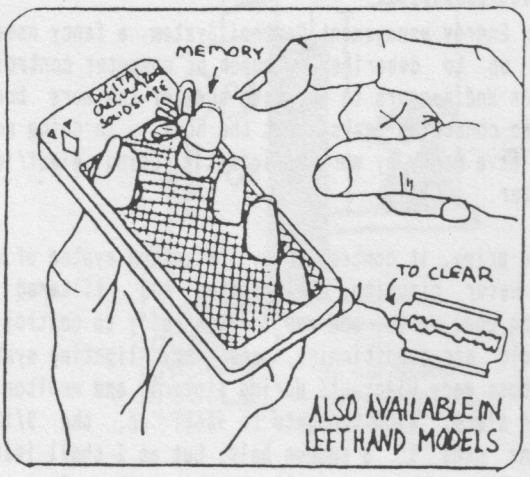
STASH: Identifies the files with directories which must not be touched by the REPACK command.
(Example: OS9 BOOT).
Stash also helps to hide the files which are to be kept guarded secrets.

ZAP: Erase a file without however, disorganising the sectors of the Allocation Table.

I would have to choose the command REPACK as the most important of these utilities.
Burke and Burke recommend the user of the commands DCHECK and CCHECK to localise the defective sectors with the utility command REPACK.
DCHECK is fully compatible with the OS9 original.
CCHECK is only found on the FILE SYSTEM REPACK disk.

I hope you will extract a lot of good from these excellent utility programmes which permit you to reduce disk usage by reconstituting fragmented disks, make their reading much faster and let the drive heads work less when reading and writing to disk.

IRISH DIGITAL CALCULATOR



These additional two pages have been added to rectify the error made in the last magazine where part of this article was inadvertantly left out.

For those people who are interested in completing this small hardware project but did not receive the free gift of the LED and resistor, there are a few left in stock and I will be happy to send them out on request.

EDITOR

A L.E.D. on/off Indicator for your Coco.

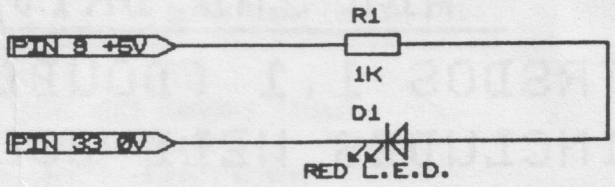
By Darren (Gonzo) Ramsey

Robbie has requested that I write a series of articles on COCO hardware, primarily based around the energy management control system I have installed, (and still installing) within my home. This I will do in forthcoming articles, but to start with I thought a small but useful modification to your COCO was in order. It was indicated to me at the last meeting I attended, that most Club members still didn't have a power on indicator on their computer. Well that will never do; so here we have a means of overcoming this minor problem. Now, before you lose interest and say to yourself "Not another hardware mod", or "I can't do that - I don't know a capacitor from a fuse, even if I got boot from it"!

This is a very easy job and I shall step through it throughout for the benefit of the raw beginner; I urge you to have a go; there is a lot of satisfaction in getting these things to work, and a power indicator is very useful.

THEORY: all COCO's have 5 Volts available, so we will use this to power our indicator. I could use a 5 Volt globe to show when the power is on, but globes are inefficient, use lots of power, and they tend to "blow" when you need them most. There is an electronic device however that can be used for indication which is very efficient. this device is a Light Emitting Diode, (L.E.D. for short).

A diode is like an electronic "non return valve" that is, it allows current to flow in one direction, but not in the other. When the current flows the L.E.D. lights. The nature of L.E.D.'s require that the current through them be limited, and this is done by placing a resistor in series with it. The circuit looks like this:-



CONSTRUCTION: The first thing to do is to open the case of your COCO. If your COCO is less than 3 months old, (if that is still possible), I would suggest you go no further, as opening the case will void your warranty. For the rest of us, carry on.

A hole of diameter 5mm needs to be drilled in the case somewhere. After much personal experience, I would strongly suggest that the best location for this is in the bottom right hand corner of the lower half of the case. Why, well if you ever intend to do further mods, or have the unfortunate need to have your machine repaired, anything fixed to the upper half of the case,

will cause you many problems, and/or much expense. I for one, as with most other techs, take more kindly to a machine that can be worked on in "comfort", rather than fighting with some U.F.O. (Unwanted Fixed Object), which insists on fighting back!

Glue the L.E.D. in place with a small amount of super glue, save the rest of the tube for a rainy day, it's not wanted here.

The resistor supplied is now soldered to the longest lead of the L.E.D. The twin wire should be stripped and soldered with one wire connected to the end of the resistor, and the other to the L.E.D.

A 5 Volt supply is now needed for the L.E.D. this can be found on any COCO on pins 9 and 33 of the Rom port edge connector socket. Pin 9 is +5V and pin 33 is ground. The wire which goes to the resistor is now soldered to PIN 9. The wire which goes to the L.E.D. is now soldered to PIN 33. DO NOT SHORT ANY OF THE EDGE CONNECTOR SOCKET PINS TOGETHER.

The next step is to test the unit; plug the computer in and turn it on. If the L.E.D. glows and the rest of the computer doesn't, then success can be assumed. If the L.E.D. doesn't glow, it may be that the L.E.D. is connected backwards, so try swapping the connections on the L.E.D. If the L.E.D. doesn't glow and the computer doesn't work either, then unsolder the wires to the edge connector socket and try turning on the computer again. If the computer still won't work, then note where the smoke comes from and take it along with you to your next Club meeting.

Good luck and happy hacking. GONZO.

END

COCO-LINK PD SOFTWARE

DISK 001 EDUCATION

1) Australian Geography
2) Australian Explorers
3) Fractutor
4) Decimal
5) Spellit
6) Times Table

DISK 002 EDUCATION #2

BINARY	MATHSMT
COCOHOME	MEMORY
COINDEMO	NUMFUN
FORMULA	PUZZLE
MATCHEM	TRIGSHOW
MATH	WORD

DISK 011 GAME

CoCo Trivia
Trivial Pursuit game.
(Takes up 2 sides of disk)

DISK 012 GAME

Computer Tote
Complete with races and tote betting.
Marvelous for club fund raising!

DISK 013 13 GAMES

21 Card Trick	25 Square
Bobo	Build
Centrit	Cypher
Germ	Life
Max	Maze
Reversi	Tanks
Yancc	

DISK 021 UTILITIES

3CLMLIST	3HBUFF
3PRNTDOC	3QKMEN40
3QKMEN80	3VIPCOCO
CATLOGUE	DIRSORT
DSKDET	GOSUBBER
HASH	MENU
MULTUTIL	PRNTDOC
QKMEN32	

DISK 031 HOME APPLICATIONS

Homehelp	Shoplist
Budget	Loan
Will	

COCO-LINK PRODUCTS

Back Issues.....$2.40each

Software:
Programmers Utility.....$20.00
(Reviewed Coco-Link No.2)

ADVERTISING RATES:

$12.00 per full page
$8.00 per half page
$5.00 per Quarter page
MAX. 3 months

Classified...$2.00 per 20 words.

Write to: COCO-LINK
31 Nedland Cres.
Pt. Noarlunga Sth.
S.A. 5167

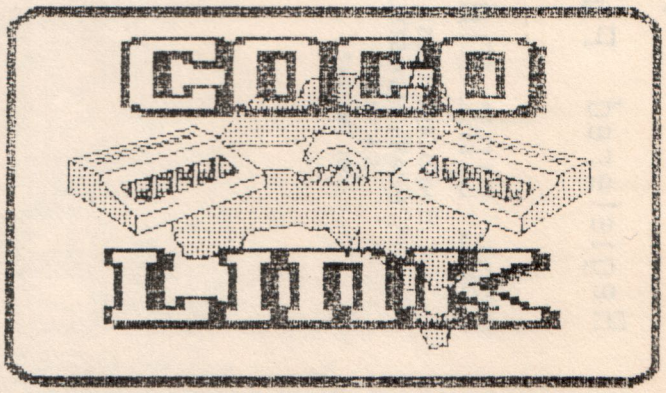
Or phone: (08) 386 1647
(08) 386 1139

CLASSIFIED ADVERTS
****

WANTED

OVERHEAD PROJECTOR. Working or not.
Darren Ramsey (08) 384 6725

SUPER VOICE SONGBOOKS. (Disks).
Vol. 1 (Potpouri)
Vol. 2 (Nursery Rhymes)
Prices to:
Jim Eadsforth (08) 298 2843.



Registered Publication No. SBH 1944

COCO-LINK MAGAZINE

31 NEDLAND CRES.,
PT. NOARLUNGA STH.,
S.A. 5167
(08) 386 1647

Surface Mail

POSTAGE PAID
CHRISTIES BEACH

JACK RAE
PO BOX 2076
MT. ISA
QLD 4825