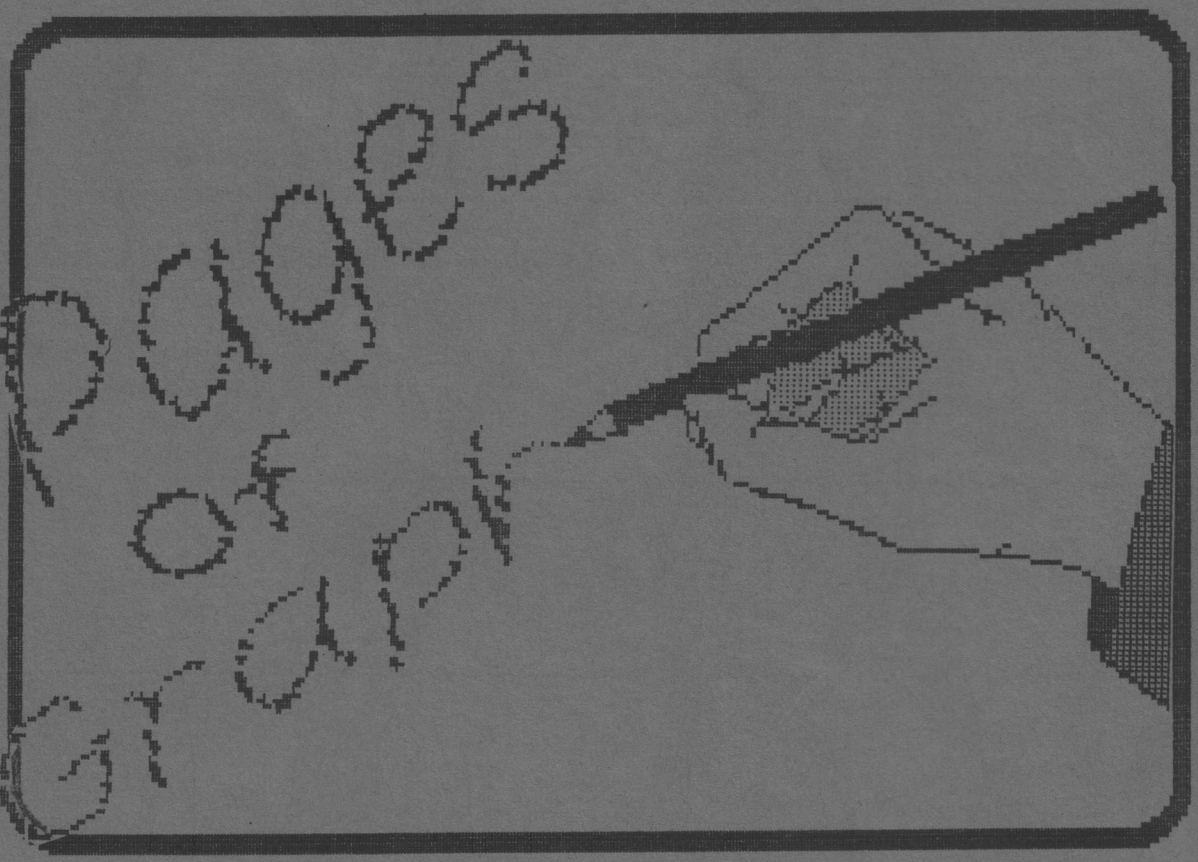
October 1990 Vol 3. No.5

COCO LINK

The Color Computer Magazine



Featuring:

- Pages of Graphics Progs.
- OS9 Beginner's Diary
- More outside world control
- Multivue

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

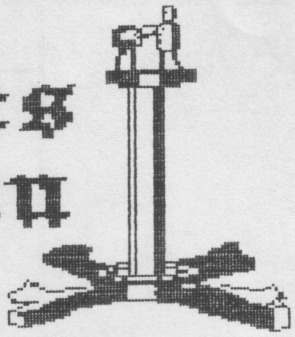
Correspondence:
Garry Holder

Submissions:
Robbie Dalzell

COCO LINK

Copyright Notice

Robbies Column



QUERIES

Whew!! It has been a very busy two months with more queries than usual reaching my desk.

My method of dealing with queries is as follows:

If Garry or I can answer the query right away, I dispatch the answer immediately by post (or on occassion I will phone).

If we are unable to answer the question we try to contact someone who we think will have the answer. If this fails we put the query in the letters pages of COCO-LINK and hope that someone will come up with the answer.

Where the query involves finding errors in programmes, this takes a bit longer and must wait till either Garry or myself has the time to delve into the unknown.

Where we feel that a query and answer would be of interest to other readers we include the query and answer in the letters pages.

****

REVIEWS

I have been asked why, in the June edition of COCO-LINK, I put a review of a programme which is not available in Australia, and why do we not review the programmes which are available here.

The answer to the first question is that a review or overview of any programme is useful information. Programmes not for sale in Australia can be ordered from the US (more on that later).

The answer to the second question is quite simple. To review a programme one first must have a copy of the programme. This is usually sent to the reviewer or magazine by the distributer of the programme. Programme suppliers in Australia do not seem to want the free advertising that this service gives them.

I would be more than happy to receive comprehensive reviews of programmes from subscribers who have a programme they feel other subscribers should know about.

It must always be remembered that a programme review is only a reflection of the writer's view of the programme. Other people may not agree with the reveiwers comments. The main function of the review should be whether the programme does what it sets out to do, is it easy to use, is it worth the price and does the reviewer think it is a good (or bad) programme.

****

PURCHASING FROM THE USA

It is possible to order programmes and hardware from the US. These must be paid in US$ and generally by bank cheque payable through a suitable bank in the US. The appropriate postage and handling must be added to the cost of the item.

If you are using a VISA or MASTERCARD or similar, and the company you are dealing with accepts them, things are much simpler. You simply give them your card number and they fix up the rest. In these cases I find that a phone call to order is the quickest and just as cheap in the long run.

When your hardware or software lands in Australia it could be liable for import duty. I am not definite what this rate is at the moment but think it is between 17% and 25%. I don't know what the actual criteria is for working out if duty is payable as I have found that sometimes they charge it and sometimes they don't. This duty is payable on collection of your parcel from the post office.

By the time you have paid for all these things the cost works out considerably more than the US$ price. This can be anything from about a third to double the US price.

I have found that service from the US is usually very good and if you are willing to pay the extra for airmail you can have your material over in no time at all.

Always try to be sure that the programme etc. you are ordering is not available in Australia. In many cases it is cheaper to buy here rather than send to America.

****

CLUBS

I have the names of a number of clubs which do not appear on our Noticeboard page. I need contact names and phone numbers. These groups are:

Brisbane West
Morwell
Redlands & Birkdale
Penrith & Springwood

Would subscribers who are members of the above clubs please send me the necessary details so as we can add them to our club list.

Any other groups who would like to be placed on the list, please contact me with the details.

****

WHAT'S NEW

A Hong Kong Peripherals developer has just brought out a slimline monitor. This monitor is only 6cms thick and stands on a tilting base. The STL68 VGA crystal display (LCD) monitor weighs less than 2Kg and takes up only 10% of the desk space of a conventional monitor.

It provides a 640 X 480 pixel high-contrast black and white display with colour simulation across 32 shades of grey.

I could certainly do with something that takes up so

little space on my cluttered desk.

****

AUSTRALIA'S FIRST COMPUTER CONVICTION

A 32 year old Melbourne man, Deon Barylak, is the first person to be convicted of computer trespass under laws first introduced in Victoria last year.

Barylak was found guilty of computer trespass and attempting to damage a computer system at the Swinbourne Institute of Technology by the use of a virus programme. He was sentenced to 200 hours' unpaid community work during the next 12 months. A VERY light sentence under the circumstances.

I am glad to see that something is being done at last to curb the idiots in the computer community who are too dumb to realise that their activities only make things worse for everybody else. When these nutcases destroy systems in colleges, banks or wherever the cost of the damage always lands back at the door of the consumer. As, in most cases, the idiots who perpetrate the deeds are also consumers, they add expense on to their own weekly bills as well as everyone elses. How dumb can you get!!

Maybe now that they have laws to cover computer trespass they will do something about updating the laws on property trespass.

****

MULTIVUE

Quite a few months back an article appeared in the American Rainbow with some interesting information on the MULTIVUE programme. The article was written by the Tandy Software Support Group in Fort Worth.

I felt that the article would be of benefit to some of COCO-LINK's subscribers and so wrote to Fort Worth for permission to reprint the article in COCO-LINK. This permission was given and the article appears in this magazine.

Those of you who subscribe to the US Rainbow will have already seen this article but our records show that there are many who do not get this magazine. It is for them that this article appears.

****

THE READERS SURVEY

Survey forms are still coming in at the time of going to print so I have not had time to collate all the material together for publication.

A full rundown of what the survey has revealed will be in the next issue of COCO-LINK.

****

Robbie

PUBLIC DOMAIN SOFTWARE

WINNERS....Disk No.032

This editions Public domain disk is the WINNERS programme by Robbie Dalzell which was presented in a series of articles in this magazine.

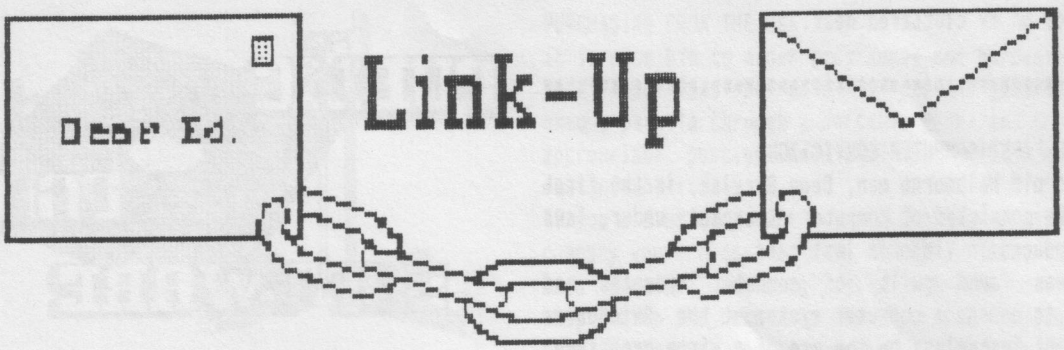
The programme is a full handicapping system for serious punters.

The disk comes complete with ASCII files of the complete series of articles which appeared in COCO-LINK. Also included on the disk is a file of 500 horses taken from Adelaide metropolitan races of 1989. This file has not been kept up to date but should be used for practice only. Once the system has been mastered this file should be deleted and your own file compiled from scratch.

WINNERS requires a double disk system or a double sided drive used under B-DOS or similar using the drive as separate sides 0 and 2.

The disk supplied is a flippy andthe data and programme will have to be configured to suit your particular system. Full instructions will be supplied.

"GENTLEMEN, I SAY RATHER THAN FIX THE BUGS, WE CHANGE THE DOCUMENTATION AND CALL THEM FEATURES."



Dear Robbie,

Please find enclosed my subscription renewal form and cheque for the $14.00 plus my response to your "Reader Survey". I must apologise for not sending this promptly as I had intended, however I am pleased to do my bit to support your efforts with CoCo-link.

The magazine that you and your helpers produce is first class, and I can just imagine the time and effort that it takes. Keep up the good work!

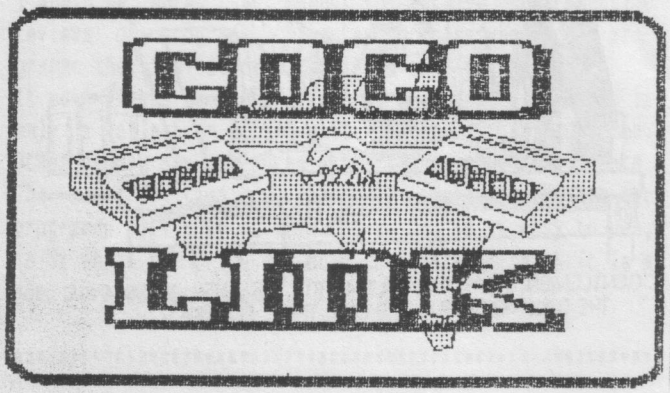
I note your "Column" comments (June 1990) regarding Unix and OS-9, and add my comment for your interest. OS-9 has been described by many as the operating system which supports true multi-tasking and multi-user capabilities, without the hassles of Unix. We believe that OS-9 will continue to gain popularity even though it will probably never reach the same mainstream statistics of MS-DOS for P.C's. Even the later versions of MS-DOS are becoming more OS-9 or Unix like all the time.

Microware have discontinued support of 6809 OS-9, so OS9-68k (OSK) for the Motorola 680XX family will be the future. The Tandy CoCo3 still provides by far the cheapest platform for learning about this very powerful and flexible operating system. The CoCo still provides a simple way of moving between "windows" which is envied by many larger and much more expensive systems.

I am quite sure that any effort put into learning OS-9, OSK and "C" programming will not be a waste.

Gordon Bentzen.
Editor, National
OS9 U.G. Newsletter

****



Dear Ed

I have been meaning to write to Link-up for some time but have not done so. It's called procrastination. Thank goodness the staff of COCO-LINK don't have the same problem or we would not be getting, on a regular basis, what, to me, is an excellent magazine

As a fairly new user of the CoCo 3 I have a good deal of difficulty. Before my CoCo 3 I had a Sinclair Spectrum which I found to be an excellent machine. It was however different in many minor details. I am coming to the conclusion that it is harder to switch from one Computer to another than it is to start from scratch. Or maybe it's my age.

I find'Towards Better Basic' very useful even though I know much of the material. It is useful to know that things I did on my Spectrum can be done on my CoCo. The article on READ-DATA-RESTORE arrived at just the right time. I was in the middle of a programme where, acting on different prompts, I needed to READ from various points in my DATA. The Spectrum has a RESTORE(line number) command which made thing easy. I was pondering what to do when along came COCO-LINK VOL 3 NO 2 presenting me with the simple idea that all I had to do was to READ from the beginning of the DATA and throw the items away until I got to the ones that I wanted. Great!

I should like to thank Sam Thompson for his answer to my problem with my DMP 105 printer. After I had written my first letter I discovered page 60 line 2 in the Scripsit manual. I tried what it said there. I got some fonts to work but not others. Thanks for your list, Sam. It seems that your DMP 107 has more fonts than my DMP 105.

My problem with the tape was solved by Robbie. We decided that I was using a duff tape. Unfortunately I had had it too long to take back to the supplier.

I look at, and try to follow the programmes listed in COCO-LINK. Unfortunately I get stumped when I come across PEEKs and POKEs. Could someone write an article or series dealing with the more useful PEEKs and POKEs.

When I want to LLIST a programme I find that some of the longer lines in my programme run off the edge of the page. With my DMP 105 and my Spectrum I knew a POKE which limited the number of characters printed on a line before the printer went to the next print line. Does anyone know of a POKE that will do the same with my CoCo? If so,could they please let me know?

Keep on Editing. You're doing fine.

Bernard Fletcher SA

Dear Ed,
I am writing to you to ask if you know of a hardware modification to run "THE MASTER KEY 2" copyright 1985 by Donelson & C. Horn; distributed by Computize Inc, on a CoCo3.
My MASTER KEY 2 is in cartridge and runs ok on a CoCo2. On CoCo3 it produces yellow lines on the screen, (green with black background).
Any assistance would be gratefully appreciated.

Desmond Rae. QLD.

Dear Desmond,
I have not been able to dig up any information on the MASTER KEY programme. I hope some of our readers can shed some light on the subject.

****

Dear Ed,
I received my copy of COCO-LINK today (2nd August), and I must write to say it's the best effort yet; congratulations. I particularly like LINK-UP and ROBBIE'S COLUMN each issue. BETTER BASIC "debugging" was excellent, but, as an amateur I was a bit confused, but in reading the article "HINTS AND TIPS" I found a lot of useful information, (A truly wonderful and informative section). The future of the COCO (Nicholas Marentes) is always interesting and a real eye opener, as to the COCO future ??? I also enjoyed Nicholas' advertisement on RASCAN, seems great, but could it be explained in a future issue what you can do with it and how. Also Nicholas, could it be possible to obtain any more programs on tape, as I already have the ones listed in your advert.
Please COCO-LINK would it be possible for you also maybe to produce some programs on tape. Love your magazine, keep up the interesting and informative work.

Graham Elphick. NSW

Dear Graham,
Thanks for the kudos.
We are hoping that Nickolas will be giving us some more news on his RASCAN Digitiser in the near future.
Regards COCO-LINK producing PD material on tape....I am afraid this is not feasible. To turn out good tapes without a VERY good fast dubbing machine takes a very long time. I am sorry to say that COCO-LINK uses up more than its fair share of both Garry and my spare time as it is.....Sorry.

****

Dear Ed.
I am doing a course with I.C.S. on Personal Computing. The CoCo3, and CCR81 comes with the Course which is about programming in BASIC.
I already have Tandy 1000EX, CMS monitor, DMP106 printer and Tandy modem 300 manual as my main computer.
I have not as yet used the modem.
I have a ROM cartridge which I do not have any instructions for. Is it possible that any reader can let me have a copy. The cartridge is MICRO PAINTER: Cat No 26.3077.
Also, do you have a list of members or clubs that have MODEM access. I have a V21 Modem but I do not have anyone to whom I can contact using it.

Mr L.A. Nunn Francis. N.S.W.
Ph (043) 69 1819

Dear Tony,
It is interesting to see that someone is using the Coco in a teaching capacity. Thanks for the information.
Regarding your query on modem use, it is not the policy of COCO-LINK to either distribute or publish the names and addresses of subscribers without their prior authority. I have included your phone number with your letter should some reader wish to contact you regards your cartridge problem or modem communication.
There is a short list of BBSs on the noticeboard page of this magazine. It may be of some help to you.

****

Dear Ed,
Thank you for your letter and review of our program EAGLE POKER. (This was for a Communications project for high school..ed) We are very pleased that you like it and we thank you for the work you did on it. We would be happy if you put it in your magazine or Public Domain disk. Thank you very much for your help.
I have a suggestion for the magazine. I wonder if you would be able to put a list of Coco supporting Bulletin Board Systems (BBSs) in your Noticeboard section. I have attached a list of numbers.
I hope this will be useful for the magazine and for those readers who have modems.
Thank you again for your help.

Mark Funston and Shaun Butterley Vic.

Dear Mark,
I have included your list of BBSs on the Noticeboard page. Thank you for a good suggestion and the list. Anyone wishing to add to this list please send me the relevant details.

****

Dear Ed,
Thank you for my Public Domain disk #021 UTILITIES, but I seem to be having a bit of trouble with it.
Program DSKDET line 0 GOTO 1000, there is no line 1000 !!
Program HASH line 160 ON ZI GOSUB6000,7000,8000,9000,250
There is no line 9000 !! I have tracked this program down in COCO-LINK Oct 1989 page 7, the listing there also does not have a line 9000
Now for some other business. I have a very bad memory, so I now use a method in my programing or with programs that I receive as follows.

CONTINUED ON PAGE 16

Below is a list of the registers in the COCO3 and their use. It is a listing in Basic. The line number is the address in decimal (add 65000). It is followed by the address in Hex, then the contents of that address (one or two bytes) in Hex, either as PEEKED or what I reckon it is initialised to (since some registers all PEEK $7E).
Some of the info I have gleaned from programs published in Rainbow magazine. Comments on the Register List below:

FF99 pokes:
32 column mode: 0=32cols,16lines; $10=64x16; $20=32x16+'status line'; $30=64x17; $60=32x19; $70=64x19.
FF99, 40 column mode: 5=40x24; $11=64x24; $21=32x25; $25=40x25; $31=64x25; $35=80x25; $61=32x28; $65=40x28; $71=64x28; $75=80x28.
Have a play by poking values into the 'Video' registers! But You will have to do it inside a running program, because the Direct Command mode resets all the registers -presumably to display the 40/80 column screens correctly.

Dynamic Address Translation (DAT) Registers:

The COCO3 addresses 512k of RAM, by replacing the 3 most significant bits of an address, with the 8 bits contained in the appropriate DAT register. There are two sets of DAT registers (I'll call them DATRs), the first is used under Basic. I don't know how to flip to the second, which is presumably useful under OS9. The 3 address bits (range 0-7) point to the 8 DATRs in order. So for a machine read or write to address $0000, the GIME chip replaces the 3 highest address bits with the contents of the first DATR (at $FFA0 or 65440) which is $78. This gives an effective address of $0F0000. Because understanding this concept is crucial to using the DATRs (which you need to do to access the 40 or 80 column screens from outside Basic) I will show the process in detail. The spaces between bits allow easy grouping into Hex digits.

```
Micro address:    $0000          xxx0 0000 0000 0000
DAT substitution: $78      0 1111 000
                  -----    -------------------------
                  $0F0000  0 1111 0000 0000 0000 0000
```

The 3 'x' bits (which are 000 in the example) choose the DATR to add in their place. This gives the micro a possible address range of 2 Mbytes! Thus if you PEEK (0), you get the contents of $0F0000, which according to the COCO3 manual, is address $70000 -ie PEEK (0) = LPEEK (&H70000). But the DAT value is $78! Why Tandy computers never do things the obvious way beats me.

Note that Basic in Direct mode keeps resetting the DATRs to the values shown in the list, so if you want to change a DAT value to fiddle, you have to do it inside a running program.

Colour Slots:

The first contents figure shown is the Hex value of what you will PEEK, the figure in backets is the decimal value of what the manual says the slots default to. (For use of the slots, see my article last month.) If you use a green-screen monitor like I do, in 32 column mode, it is useful to poke 63 (buff) into slot 13 ($FF8D), which is the same as PALETTE 13,63. In 40 or 80 column mode I can just type CLS5, which does the same thing. (CLS in 32 column mode doesn't change the background 'attribute' colour.)

```
1 '  *** COCO3 REGISTER USEAGE ***
2 '  Compiled by Ken Wagnitz
3 '  (Add 65000 to line # for the decimal address)
4 '
424 ' FF90 7E GIME EN FIRQ = 5C
425 ' FF91 7E GIME CLOCK
426 ' FF92 40 GIME CLOCK
427 ' FF93 40 GIME EN TIMER IRQ
428 ' FF94 7E7E GIME TIMER COUNT (12 BITS)
430 '
432 ' FF98 VIDEO MODE REGISTER
433 ' FF99 VIDEO RESOLUTION REGISTER
434 ' FF9A BORDER COLOUR
435 '
437 ' FF9D 00 VIDEO VERTICAL OFFSET REGISTER
438 ' FF9E 00   "
439 ' FF9F 00   "
440 ' FFA0 78 DAT REGISTERS (SET 1)
441 ' FFA1 79
442 ' FFA2 7A
443 ' FFA3 7B
444 ' FFA4 7C
445 ' FFA5 7D
446 ' FFA6 7E
447 ' FFA7 7F
448 ' FFA8 78 DAT REGISTERS (SET 2)
449 ' FFA9 70
450 ' FFAA 71
451 ' FFAB 72
452 ' FFAC 73
453 ' FFAD 7D
454 ' FFAE 75
455 ' FFAF 7F
456 ' FFB0 52 (18) SLOT 0 (40/80 BACKGND)
457 ' FFB1 76 (54) SLOT 1  YELLOW
458 ' FFB2 49 (09) SLOT 2  BLUE
459 ' FFB3 64 (36) SLOT 3  RED
460 ' FFB4 7F (63) SLOT 4  BUFF
461 ' FFB5 5B (27) SLOT 5  CYAN
462 ' FFB6 6D (45) SLOT 6  MAGENTA
463 ' FFB7 66 (38) SLOT 7  ORANGE
464 ' FFB8 40 (00) SLOT 8  BLACK (40/80 FOREGND) (32 CHAR
GRAPHICS BACKGND)
465 ' FFB9 52 (18) SLOT 9  GREEN
466 ' FFBA 40 (00) SLOT 10 BLACK
467 ' FFBB 7F (63) SLOT 11 BUFF
468 ' FFBC 40 (00) SLOT 12 BLACK (32 COLUMN FOREGND)
469 ' FFBD 52 (18) SLOT 13 GREEN (32 COLUMN BACKGND)
470 ' FFBE 40 (00) SLOT 14 BLACK
471 ' FFBF 66 (38) SLOT 15 ORANGE
472 '
496 ' FFD8 LOW  SPEED POKE
497 ' FFD9 HIGH SPEED POKE
498 '
504 ' FFE0 9008 SECONDARY VECTORS
506 ' FFE2 C009
508 ' FFE4 C005
510 ' FFE6 F205
512 ' FFE8 F50F
514 ' FFEA F50F
516 ' FFEC F7AF
518 ' FFEE F38F
520 ' FFF0 800B PRIMARY VECTORS
522 ' FFF2 FEEE SWI3
524 ' FFF4 FEF1 SWI2
526 ' FFF6 FEF4 FIRQ
528 ' FFF8 FEF7 IRQ
530 ' FFFA FEFA SWI
532 ' FFFC FEFD NMI
534 ' FFFE 8C1B RESET
```

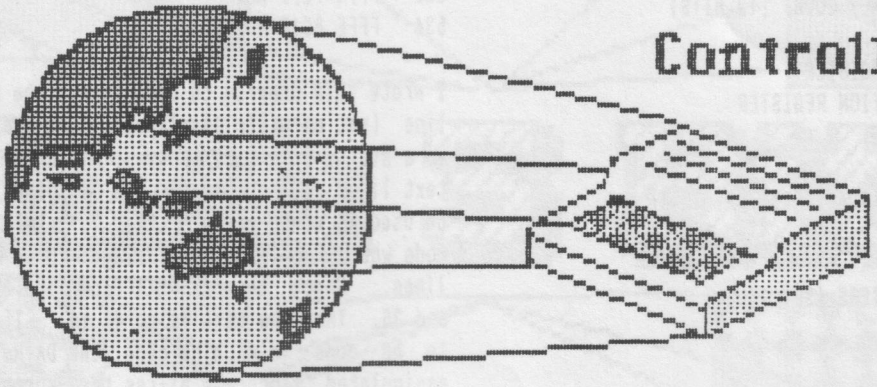
I wrote this program to demonstrate the use of the 25th line (not normally displayed). It could easily be used as a status line in a program, without otherwise losing a text line. I wrote it for the 80 column mode, but it can be used in other modes. Line 20 puts the COCO into a mode where 25 lines are displayed, plus a couple more dot lines. Hence the FOR-NEXT from line 80 blanks lines 25 and 26. Then the text is poked in. If this writing was to be done in machine code, the DATRs would have to be manipulated, since the hi-res text screen is outside the normal 64k space. This is why it does not chew up Basic program memory.

```
1' *** LINE 25 *** by Ken Wagnitz
10 CLS
20 POKE&HFF99,&H35 'SET TO 25 LINES
30 FOR I=1 TO 24
40 PRINTCHR$(13)"LINE "I;
50 NEXT
55 '
60 START = &H6C000+24*80*2
70 FINISH= START+80*2*2
80 FOR I=START TO FINISH STEP 2
90 LPOKE I,32 'BLANK LINES 25 & 26
100 LPOKE I+1,4 'SET CHAR ATTRIBUTES
110 NEXT
120 A$="THIS IS LINE 25!"
130 FORI=1 TO LEN(A$)
140 LPOKE START+2*(I-1),ASC(MID$(A$,I,1))
150 NEXT
160 LPOKE START+2*(I-1)+1,68 'DO A CURSOR
170 LOCATE 0,0 'MOVE REAL CURSOR
180 GOTO180
```

The following program is VITAL for anyone who wants to run OS9 level 1, version 1 on the COCO3.

```
1' *** OS9 PATCH *** Author-Viatel No.352924510 (!)
10 CLEAR500
20 DSKI$0,34,1,A$,B$
30 V=VARPTR(A$)
40 P=PEEK(V+2)*256+PEEK(V+3)
50 POKE P+&H49,&HEF
60 POKE P+&H54,&HEF
70 DSKO$ 0,34,1,A$,B$
```

An Introduction to "OUTSIDE WORLD" Controlling



By Darren (Gonzo) Ramsey

As I mentioned in the last article, I will now discuss the P.I.A. (Peripheral Interface Adapter). Many good articles have been written about P.I.A.'s, in particular the 6821. A very good article was written by Tony Distefano which was published in the August 1986 Australian Rainbow.

Why P.I.A.? Well, for those of us dedicated to putting computers to "useful" purposes, The Peripheral Interface Adapter is a godsend. P.I.A.'s are powerful, easily programmed I/O port devices, and come in all manner of sizes and options. Some have on board RAM and/or timers and have some more operating modes than thought possible. I have chosen the 6821 P.I.A. because if its higher number of I/O ports, and also because it supports the modes of operation that I require to perform my tasks.

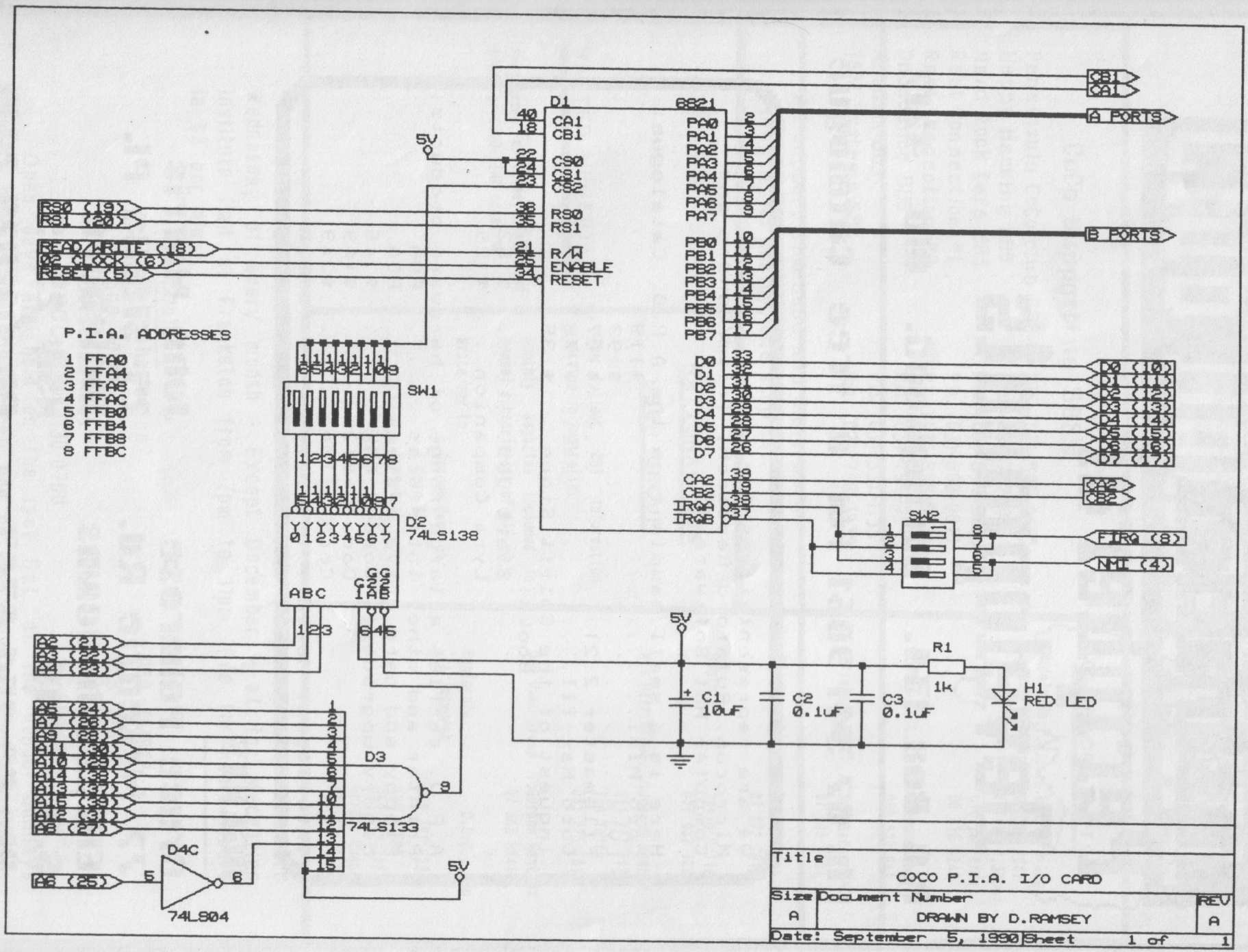
But why P.I.A. and not D-A/A-D (Digital to analogue/ Analogue to digital)? I have an answer to this question too. Many day to day tasks consist of controlling machines/appliances simply to turn them on and off. Lets consider an electric jug. You turn it on, it boils after a time and you turn it off. Lets consider an air conditioner, you turn it on, select hot or cold, often a thermostat regulates the operation by sensing the temperature and turning it on and off. These are but two examples of controlling appliances using the on/off digital type principle. There are many more, but one problem exists. These systems work well, but they're not computer compatible! Enter the P.I.A., an interfacing device specifically designed to turn on and off our appliances and their controls, by means of computer.

In order to control equipment, the P.I.A. needs to be able to initiate outputs to do the switching, but also accept inputs from sensing devices from within the equipment. More often than not, the inputs and outputs need to be active at the same time. The 6821 has basically two halves. Each halve's ports can be operated independant of each other, and each port can be set up as either an input or an output. This affords extreme flexibility in port handling.

Enough said about the P.I.A., albeit the heart of system. To get things moving along I have provided a schematic diagram of the P.I.A. control card. In the next article I will discuss the addressing of it together with some software routines to make use of it. I shall also give some construction tips.

In the article following that, I shall dive directly into the simulator unit. This is not an essential device, but makes life a lot simpler for debugging software. This will include more schematics, software and tips.

SCHEMATIC OVERLEAF.


P.I.A. ADDRESSES
1 FFA0
2 FFA4
3 FFA8
4 FFAC
5 FFB0
6 FFB4
7 FFB8
8 FFBC
D1
6821
CA1
CB1
CS0
CS1
CS2
RS0
RS1
R/W
ENABLE
RESET
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
D0
D1
D2
D3
D4
D5
D6
D7
CA2
CB2
IRQA
IRQB
A PORTS
B PORTS
FIRQ (8)
NMI (4)
SW1
SW2
D2
74LS138
D3
74LS133
D4C
74LS804
R1
1k
C1
10uF
C2
0.1uF
C3
0.1uF
H1
RED LED
5V
Title
COCO P.I.A. I/O CARD
Size
A
Document Number
DRAWN BY D.RAMSEY
REV
A
Date: September 5, 1990
Sheet 1 of 1

Club Noticeboard

CoCo Supporting BBS's

Penninsula CoCo club BBS	Ph. (03)-580-4605
Happy Hacking BBS	Ph. (03)-787-8759
Hard Rock Cafe BBS	Ph. (03)-894-2815
Real Connection #1	Ph. (03)-808-0810
Real Connection #2	Ph. (03)-808-0331
Decadence BBS	Ph. (03)-794-7949
Nemisis BBS	Ph. (03)-331-1155
J&M Systems BBS	Ph. (02)-749-1935

All support 2400 baud and 8 bit word length, No parity, one stop bit.

WANTED URGENTLY

Programmes, articles, hints and tips for COCO-LINK magazine.

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

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
. 07 345 5141

Peninsula C.C.C...Bob Charlesworth
. 059 791 922
. Sue Jones
. 059 773 292
. Gordon Chase
. 059 711 553

The Peninsula Color Computer Club (Inc) meets on the 3rd Wednesday of every month - Except December - at the Mechanics Institute Hall at Frankston from 7pm to 11pm. Fee for non-members is $3 per night.

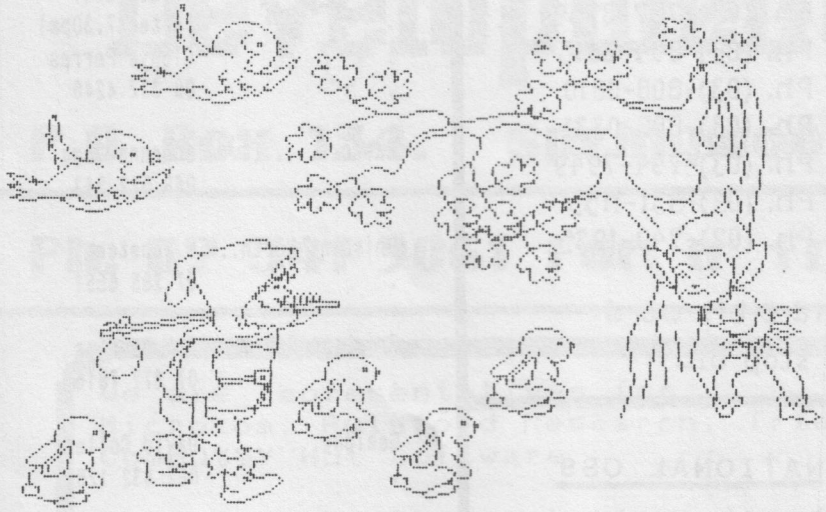
Contact:

Bob Charleston	059 791922
Sue Jones	059 773292
Gordon Chase	059 711553
PCCC-BBS 2130-0700	03 5804605

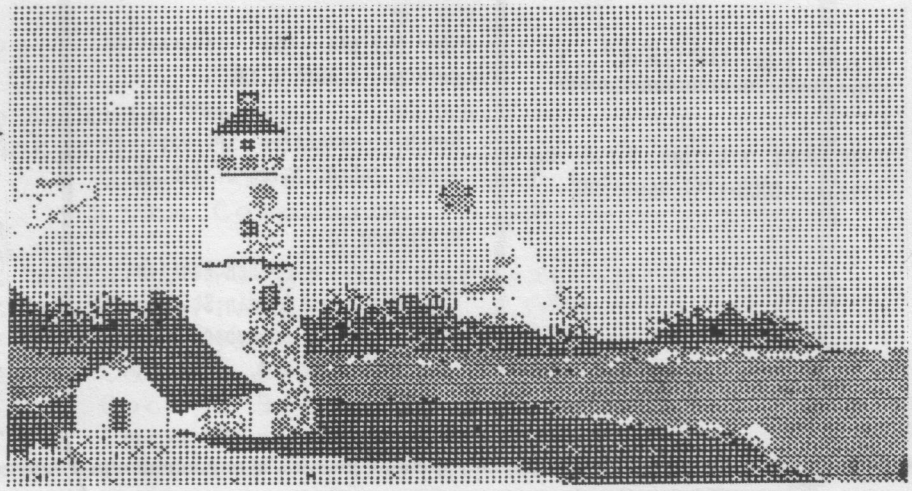
Open days are held on the last Sat. of the school holidays from 12 noon - 5pm. Please call one of the above to confirm dates.

ALL WELCOME

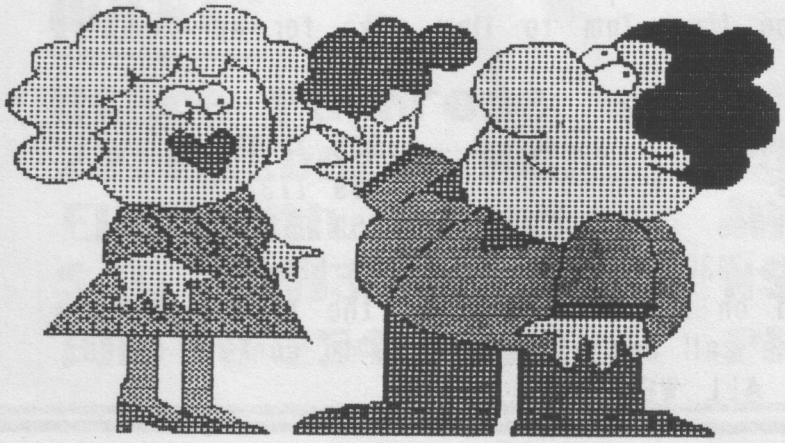
Graphics by.....



This delightful study of elves was drawn by Margaret Bell



"The Lighthouse"
By Arthur Williams
Using CoCoMax III



"Cartoon Characters"
By Arthur Williams
Using CoCoMax III

Multi-Vue and Pre-existing Applications

By

The Tandy Software Support Group

Inasmuch as the documentation for Multi-Vue has been written in a reference style which is more suited to the needs of a programmer than to the needs of the average end-user, we have prepared the following packet of information to help this type of user obtain greater usefulness and pleasure from Multi-Vue.

Basically, the things a person needs to know in order to make use of Multi-Vue for pre-existing software are as follows:

1. The basics of OS9 windows.
2. How to design an AIF (application information file).
3. How to design an icon file and set its attributes.
4. A little about BASIC09 to help facilitate step 3 above.

OS9 WINDOW BASICS

There are six types of OS9 windows with which we need to concern ourselves in our excursion into the world of Multi-Vue. They are, and I quote (from page 9-34 of the BASIC09 reference section of the OS9 level II documentation):

1. A 40 x 24, 16 color text window requiring 1.6K of memory.
2. An 80 x 24, 16 color text window requiring 4K.
5. An 80 x 24, 2 color graphics window requiring 16K.
6. A 40 x 24, 4 color graphics window requiring 16K.
7. An 80 x 24, 4 color graphics window requiring 32K.
8. A 40 x 24, 16 color graphics window requiring 32K.

The 40 x 24 graphics windows have a 320 x 200 dot resolution for graphics purposes and a 40 x 24 character resolution for text purposes. The 80 x 24 graphics windows, similarly, have 640 x 200 and 80 x 24 resolutions.

Two things need to be kept in mind when selecting a window for an application under Multi-Vue. First, text windows cannot be sized by an AIF. That is, a text window will always take up a full screen, no matter what size you specify for it. Second, while graphics windows may be sized to fit on screen with several other windows at the same time, some applications may have a certain minimum size, under which they will not function properly. So be aware of this when we construct an AIF in the next section.

APPLICATION INFORMATION FILES

An application information file (AIF) is a short text file (9 lines, to be exact) which tells Multi-Vue to associate an icon with a particular application, how much memory will be required to run the application and what type and size of window to display the output in.

To create an AIF, use the edit command under OS9. AIF's should be in the directory from which your program may take data, if any. If the program does not need to input data from any particular directory, then the AIF may be anywhere.

A the OS9 prompt, use the chd command to change your data directory to the one in which you want the AIF to appear. Then type 'edit aif.xxx', where 'xxx' stands for any legal triplet of characters which can be used in an OS9 filename. When the E: prompt appears, the computer is waiting for input to tell it what to put in the AIF.

Each AIF requires the following information, in the following order:

Line #1 of the AIF contains the name of the command or program you wish to call.

Line #2 lists any parameters you may need to pass for the application to run properly.

Line #3 is the pathlist to the icon file to be displayed by the AIF. If you do not specify a full pathlist, including the device name, from which to call the icon, Multi-Vue will consider the pathlist as being in the current execution directory, wherever that may be, usually /d0/cmds.

Line #4 tells Multi-Vue how many pages of memory to allocate for the application you are calling. There are 4 pages to 1K of memory, in this case. For most

programs, the standard 32 pages (8K), which OS9 allots by default, will be more than sufficient. However, large programs, such as Rogue, Dynacalc, or BASIC09 require considerably more. 0 will allot the default amount of memory.
Line #5 tells Multi-Vue in what kind of window screen to display the program. This number is the same as that given in the '-s=' specification of OS9's wcreate command.
Line #6 specifies the minimum width of the window for your program in character units. There are two types of window, text and graphics. Text windows will always take up a full screen, whereas you can use lines 6 and 7 of the AIF to tell Multi-Vue to start graphics windows at a smaller size, and make them larger if you want.
Line #7 of the AIF gives the minimum length of the window for your application in lines, with the same notes as for line #6.
Line #8 chooses the palette slot for the window to use for the background color.
Line #9 chooses the palette slot to use for the foreground color. Default colors are:

Slot 0: White
Slot 1: Blue
Slot 2: Black
Slot 3: Green

These can be changed by using the display command.

When you see the E: prompt in edit, press the space bar to insert a line, followed by the information you need on each line, then press the (ENTER) key. For example:

```
E: dynacalc (ENTER)
E: (ENTER)
E: /d1/cmds/icons/icon.dynacalc (ENTER)
E: 100 (ENTER)
E: 7 (ENTER)
E: 39 (ENTER)
E: 11 (ENTER)
E: 2 (ENTER)
E: 0 (ENTER)
```

where (ENTER) means to press the 'enter' key.

Having edited an AIF, you will not need to create an icon file to display so Multi-Vue will let you have access to your program.

ICONS

The following information is included for those who are interested in some of the technical aspects of graphics display. This information is necessary to know in order to write a program like the edic procedure included in this package. It is not necessary to know these things in order to use the program

If you read page 9 - 22 of the Multi-Vue documentation, you will see that an icon file contains a 24 x 24 pixel four color bitmap. What that means is that the file contains data to create a picture which is 24 dots high by 24 dots wide, in a four color display mode. To accomplish this, the computer handles the information contained in a byte in a different fashion than normal. A byte in the computers memory contains eight units of information, called bits. Usually, the byte is treated as a unit. That is, it is considered by the operating system as a single group of eight bits.
Each bit is either on or off, corresponding to a 1 or 0, respectively. A single bit, by itself, can only represent two possibilities. Therefore, by grouping them together in 8's (to form bytes), you can get 2 raised to the 8th power, or 256, different states. Now, what the computer does, in the case of graphics data, is break the bytes into smaller groups of bits.

In a two color mode, the byte is broken into eight groups of one bit each. For a four color mode, the byte is broken into four groups of two bits each and in a sixteen color mode, the byte contains two groups of four bits each. Each group of bits in the byte controls a single dot (pixel) on the screen. Therefore, in a four color mode, with which we are concerned, note, there are two bits in each group. This means that each group can represent 2 raised to the 2nd power, or four different states. (i.e. a choice of four colors may be assigned to any given dot position.)
The computer checks the group in a byte which corresponds to a certain dot on the screen to see what state (0 - 3) is represented there. It then goes to the palette register in memory with the same number, obtains the color data which is stored there and sets the dot to that color.

To draw an icon properly, it is necessary to know how many bytes we need to calculate, and how to assign the number to each byte in order to create the proper pattern of colors.

First, our picture is 24 dots wide. In the four color mode we are using, each byte controls four dots; therefore, we need 24/4, or 6 bytes to control each row. Since there are 24 rows, it requires a total of 6 x 24 = 144 bytes to create the entire icon.
To assign the correct number to the bytes involved, first make a plot or sketch of your icon on graph paper, noting which dots are to be which color. Suppose you had, for example, a pattern of four dots, controlled by the same byte, with colors as follows: 1-0-3-2. The possible bit settings in our two bit groups are as follows:

Binary Value	Decimal Value
00	0
01	1
10	2
11	3

Now, let's put our four groups together into a byte of data. The first group is 1, which corresponds to a group of 01. The next group is 0, a 00. Combined, they give 0100. The next group is 3, or 11. Combined with the previous two groups, this gives 010011. The final group

is 2, or 10. Combined, this yields 01001110.
The values of the bits in a byte, for computational purposes, from left to right, are: 128, 64, 32, 16, 8, 4, 2 and 1 respectively. To obtain the value to put into the byte to create the pattern of colors we mentioned previously, you add values for the bits which are turned on. The bits in the byte we just constructed, which are turned on, correspond to the values 64, 8, 4 and 2. Adding these gives 64 + 8 + 4 + 2 = 78. Therefore, we need to make the value of the byte, to control the four dots we wanted before, 78. Now, repeat this procedure for the other 143 dot sections of your icon.

This is actually very simple, once you have done it a few times, but it is also very time consuming. To that effect, and in accord with the promise to make the package more user-friendly, we have included the listing of edic.

EDIC (THE ICON EDITOR)
The EDIC program is intended to speed the editing of graphics files for icons under Multi-Vue. To use the program, type the supplied listing into the editor in BASIC09 by typing 'edit edic' at the B: prompt. When the E: prompt appears, type in each line of the program, pressing the (ENTER) key after each line. When you have finished entering the listing, the next step is to save and pack the program. Type 'q' to exit the BASIC09 editor to the B: prompt. Type:

save edic (ENTER)
pack edic (ENTER)

Put the system master disk of the operating system back into /d0. Type

chx /d0/cmds (ENTER)
load merge (ENTER)

Put your boot/config/BASIC09 disk back in /d0 and type

merge /d0/cmds/edic /d0/cmds/runb /d0/cmds/inkey
/d0/cmds/gfx2 >/d0/cmds/edic.file (ENTER)

(NOTE: This is one entire command and when typed, will wrap around to the following line. Do not enter as two separate lines.)
If you are using an RGB monitor, Type:
. monotype r (ENTER)
to get a representation of colors as they will appear under Multi-Vue. To run the program, boot OS9. At the OS9: prompt type:

iniz w1 (ENTER)
merge sys/stdfonts >/w1 (ENTER)
shell i=w1& (ENTER)

Now, press the CLEAR key to see the new window. Put the edic.file disk in /d0 and type

chx /d0/cmds; load edic.file (ENTER).
edic (ENTER)

You will be prompted for the background color for your icon. Your choice of responses will be 0, 1, 2 or 3, which correspond to the pallette slots available in a four-color mode, into which the icons are intended to be mapped. 0 is white, 1 is blue, 2 is black and 3 is green. Type in the number corresponding to the color which will be the main color of your icon. This fills in the rectangle with that color. You can then begin editing individual pixels. A blinking dot will appear in the upper left corner of the icon image area. This can be positioned by using the I, comma, J and L keys to move up, down, left and right, respectively. To change the color of a pixel, position the cursor (blinking dot) in that location and press the 'D' key, followed by the number of the pallette slot containing the color you wish to appear there. These numbers are the same ones used to select a background color. When the image appears as you would like, press 'F' to let edic know you are finished. After a few seconds, you will be prompted for the pathlist where the icon information will be filed. Type in the pathlist and press enter. After the file has been written, the OS9: prompt will return. You now need to use the 'attr' command to turn on the execute and public execute attributes of the icon file, so that Multi-Vue will recognize it as a valid icon. For example, suppose you stored your icon as '/d0/cmds/icons/icon.rogue'. Load the 'attr' command by typing 'load Attr (ENTER) at the OS9: prompt. Place the disk with the icon file in /d0 and type:

attr /d0/cmds/icons/icon.rogue e pe (ENTER)

When the OS9: prompt returns, the icon file is ready for use. Now, all you need do is edit an AIF to call the icon, so it will appear on the screen, and you can then use the icon to call the program associated with it.

The listing: edic

```
PROCEDURE edic
 0000       DIM i,ic,q,r,s,path,x,y:INTEGER
 0023       DIM icar(6,24),p(24,24),c:BYTE
 0044       DIM ch:STRING[1]
 0050       DIM fname:STRING[60]
 005C       INPUT "Background color?",c
 0075       FOR q=1 TO 24
 0085         FOR r=1 TO 24
 0095           p(q,r)=c
 00A4         NEXT r
 00AF       NEXT q
 00BA       RUN gfx2("curoff")
 00C8       RUN gfx2("clear")
 00D5       RUN gfx2("defcol")
 00E3       RUN gfx2("color",1)
 00F3       RUN gfx2("box",159,0,401,191)
 010B       RUN gfx2("color",c)
 011D       FOR x=160 TO 390 STEP 10
 0133         FOR y=0 TO 184 STEP 8
 0148           RUN gfx2("bar",x,y,x+7,y+7)
 016B         NEXT y
 0176       NEXT x
 0181       x=160 \y=0
 018F 1     ch=""
 0199       WHILE ch="" DO
 01A5         RUN inkey(ch)
 01AF         RUN gfx2("curhome")
```

```
01BE         RUN gfx2("color",2)
01CE         RUN gfx2("bar",x,y,x+7,y+7)
01F1         RUN gfx2("color",0)
0201         RUN gfx2("bar",x,y,x+7,y+7)
0224        RUN gfx2("color",p((x-150)/10,(y+8)/8))
0248         RUN gfx2("bar",x,y,x+7,y+7)
026B       ENDWHILE
026F       IF ch="j" OR ch="J" THEN
0284         x=x-10
028F         IF x<160 THEN
029B           x=x+240
02A6         ENDIF
02A8       ELSE
02AC         IF ch="l" OR ch="L" THEN
02C1           x=x+10
02CC           IF x>390 THEN
02D9             x=x-240
02E4           ENDIF
02E6         ELSE
02EA           IF ch="," OR ch="<" THEN
02FF             y=y+8
030A             IF y>184 THEN
0316               y=y-192
0321             ENDIF
0323           ELSE
0327             IF ch="i" OR ch="I" THEN
033C               y=y-8
0347               IF y<0 THEN
0353                 y=y+192
035E               ENDIF
0360             ENDIF
0362           ENDIF
0364         ENDIF
0366       ENDIF
0368       IF ch="f" OR ch="F" THEN
037D         GOTO 3
0381       ENDIF
0383       IF ch="d" OR ch="D" THEN
0398         GOTO 2
039C       ELSE
03A0         GOTO 1
03A4       ENDIF
03A6 2     ch=""
03B0       WHILE ch="" DO
03BC         RUN inkey(ch)
03C6         RUN gfx2("curhome")
03D5       ENDWHILE
03D9       IF ch<"0" OR ch>"3" THEN
03EE         GOTO 2
03F2       ENDIF
03F4       c=VAL(ch)
03FE       RUN gfx2("color",c)
0410       p((x-150)/10,(y+8)/8)=c
042B       RUN gfx2("bar",x,y,x+7,y+7)
044E       c=0
0455       GOTO 1
0459 3     FOR q=1 TO 24
046C         FOR r=1 TO 6
047C           icar(r,q)=0
048A           FOR s=1 TO 4
049A             i=1
04A1             IF s=4 THEN
04AD               GOTO 4
04B1             ENDIF
04B3             FOR ic=1 TO 4-s
04C7               i=i*4
04D2             NEXT ic
04DD 4           icar(r,q)=icar(r,q)+p((r-1)*4+s,q)*i
050D           NEXT s
0518         NEXT r
0523       NEXT q
052E       path=1
0535       RUN gfx2("color",2)
0545       RUN gfx2("curon")
0552       INPUT "Pathlist for icon file",fname
0570       CREATE #path,fname:WRITE+DIR
057C       FOR q=1 TO 24
058C         FOR r=1 TO 6
059C           SEEK #path,(q-1)*6+r-1
05B3           PUT #path,icar(r,q)
05C4         NEXT r
05CF       NEXT q
```

****

LINK-UP

CONTINUED FROM PAGE 5

LINE 0 GOTO10 ' self explanatory
1 'REM line used to inform me of what the program is about.
2 CSAVE"PROGRAM"
3 SAVE"PROGRAM/EXT":DIR:END
4 SAVE"PROGRAMA/EXT":KILL"PROGRAM/EXT":RENAME"PROGRAMA/EXT" TO "PROGRAM/EXT":END'Used in initial typing out if program is from a book/map etc,
8 LLIST 'Line used to print elongated title of program and LLIST program to printer
9 'Where program is from e.g.
CL10/90P7 = COCO-LINK Oct 1989 page 7
AR6/87P23 = American Rainbow June 1989 page 6
10 'Full title of program and author
Lines 5,6 & 7 could be used for some other use.
Well that's all for now, hope to hear from you soon.

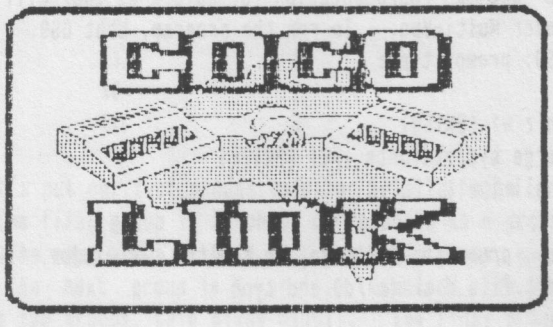
Ron Munro. NSW.

Dear Ron,
DSKDET does not have a line0. I think you have mixed it up with GOSUBBER which does have such a line. The GOTO 1000 directs you to where your main programme should be located. GOSUBBER is purely a listing of commonly used subroutines called from your main programme. This is explained in the instructions for the programe which should have accompanied the disk. (My apologies for ommiting to enclose it).
Regarding the HASH programme. Line 9000 was designated for an option to Modify a Record. As a module for this cannot be written unless the type of rcord is known, the actual code was omitted. You could put a line such as:
9000 PRINT:PRINT"PRESS ANY KEY TO CONTINUE"
9001 I$=INKEY$:IFI$=""THEN9001
9002 GOTO15000
Thank you for your 'Mind Jogger' code. It might be a preferable option to place it at the end of the programme. Line60000 or similar.

****

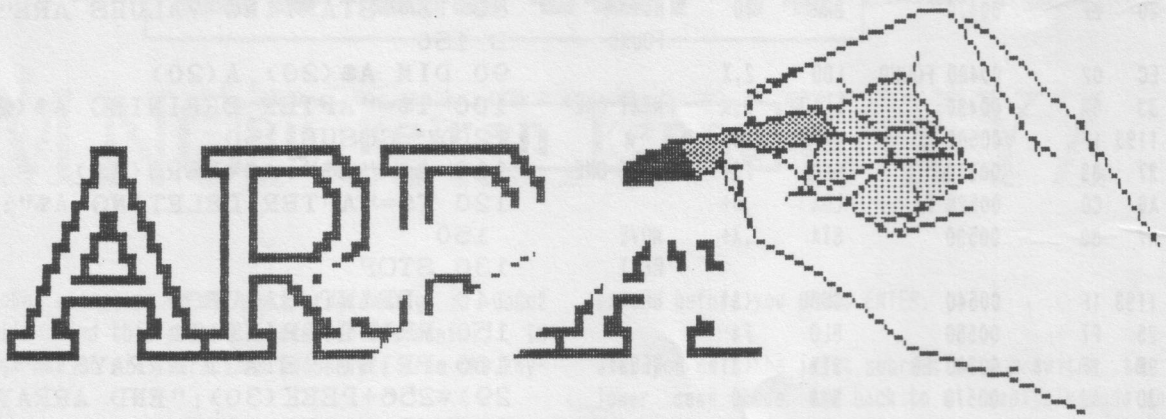
End

****



Erasing Arrays

By George McLintock



This is a small ML routine which allows you to erase an array from a Basic program, while the program is running. This will 'free up' the memory used by it, and/or allow you to define another array with the same name but different dimensions.

The ML routine is relocatable, and can be placed anywhere in memory for execution. The Demo program puts it in the cassette buffer, but this is not necessary. eg if you change M to equal &H400, you can execute it from the 32 column text screen. It could also be incorporated at the end of a Basic program.

The routine is called with a single string parameter, which contains the names of the arrays to be deleted (ie more than one can be deleted with a single call). eg if you had DIM C$(500),T(200),P(60) then to erase these would use A$="C$,T,P". Can use extra spaces in the string if you wish, and the comma is not essential, ie could use spaces instead eg as A$=" T , P C$" will do the same thing.

```
                00010 *CALLED ERASE - TO ERASE  ARRAYS
FROM A BASIC PROGRAM
                00020 * CALLED WITH  A$=USR(A$)
                00030  *  WHERE  A$  CONTAINS NAMES OF
ARRAYS TO DELETE
                00040 * EG A$="A$,D,M"
                00050 *
          0040  00060 DAT     EQU     64
          0042  00070 ENDDAT  EQU     66
          0044  00080 CHR     EQU     68
                00090 *
7D00            00100         ORG     32000
7D00 E6   84    00110 START   LDB     ,X      STR LEN
7D02 AE   02    00120         LDX     2,X     START
                                              DATA
7D04 9F   40    00130         STX     <DAT
7D06 3A         00140         ABX
7D07 9F   42    00150         STX     <ENDDAT END DATA
7D09 9E   40    00160 TO      LDX     <DAT
7D0B A6   80    00170         LDA     ,X+     BUILD
                                              NAME
7D0D 9C   42    00180         CMPX    <ENDDAT FOR
                                              SEARCH
7D0F 22   4F    00190         BHI     FINALL
7D11 81   41    00200         CMPA    #65     A
7D13 25   F4    00210         BLO     TO
7D15 81   5A    00220         CMPA    #90     Z
7D17 22   F0    00230         BHI     TO
7D19 E6   80    00240         LDB     ,X+
7D1B C1   24    00250         CMPB    #36     $
7D1D 27   42    00260         BEQ     T3
7D1F C1   41    00270         CMPB    #65     A
7D21 25   42    00280         BLO     T4
7D23 C1   5A    00290         CMPB    #90     Z
7D25 22   3E    00300         BHI     T4
7D27 DD   44    00310         STD     <CHR
7D29 E6   80    00320         LDB     ,X+
7D2B C1   24    00330         CMPB    #36     $
7D2D 26   06    00340         BNE     FTHS1
7D2F DC   44    00350         LDD     <CHR
7D31 CA   80    00360         ORB     #$80
```

```
7D33 DD   44      00370 FINTHS  STD    <CHR     GOT THIS
                                                ARRAY
7D35 DE   44      00380 FTHS1   LDU    <CHR     NAME
7D37 9F   40      00390         STX    <DAT
7D39 9E   1D      00400         LDX    <29      START
                                                ARRAYS
7D3B 11A3 84      00410 F2      CMPU   ,X
7D3E 27   0A      00420         BEQ    FOUND
7D40 EC   02      00430         LDD    2,X      TO NEXT
7D42 30   8B      00440         LEAX   D,X
7D44 9C   1F      00450         CMPX   <31      END
                                                ARRAYS
7D46 25   F3      00460         BLO    F2
7D48 20   BF      00470         BRA    T0       NOT
                                                FOUND
7D4A EC   02      00480 FOUND   LDD    2,X
7D4C 33   8B      00490         LEAU   D,X      NEXT ONE
7D4E 1193 1F      00500         CMPU   <31
7D51 27   09      00510         BEQ    F6       LAST ONE
7D53 A6   C0      00520 F4      LDA    ,U+
7D55 A7   80      00530         STA    ,X+      MOVE
                                                REST
7D57 1193 1F      00540         CMPU   <31      UP
7D5A 25   F7      00550         BLO    F4
7D5C 9F   1F      00560 F6      STX    <31      RESET
7D5E 20   A9      00570         BRA    T0
7D60 39           00580 FINALL  RTS
7D61 C6   80      00590 T3      LDB    #$80
7D63 20   CE      00600         BRA    FINTHS
7D65 5F           00610 T4      CLRB
7D66 20   CB      00620         BRA    FINTHS
                  00630 *
          7D68    00640 ZZEND   EQU    *
          7D00    00650         END    START
00000 TOTAL ERRORS
```

COCO 2 HINTS

45) SPEED - UP POKE65495,0
NORMAL SPEED POKE65494,0

TRIPLE SPEED POKE65497,0
NORMAL SPEED POKE65496,0

NOTE: These last two POKEs will cause the loss of screen output.

****

46) GET/PUT dimensions can be accomplished by the following method instead of the memory wasting method shown in the TANDY books. Use this formula:

Array = ((X*Y)-1)/n

Where n = 40 for PMODE3 or 4
n = 80 for PMODE1 or 2
n = 160 for PMODE0

****

```
10 'CALLED DEMO - TO DEMONSTRATE
 USE OF ERASE ML ROUTINE
20 'ML POKED TO CASSETTE BUFFER
- COULD BE ANYWHERE IN MEMORY
30 'SET A$= NAME OF ARRAYS TO BE
 DELETED
40 ' THEN EXEC AN A$=USR0(A$) TO
 DELETE THEM
50 '
60 M=&H1DA:GOSUB 190:DEFUSR0=M '
SET ML ROUTINE
70 DIM X,Y,A,B,T,A$,B$,T$  'SOME
 SIMPLE VARIABLES
80 T$="STARTING VALUES ARE":GOSU
B 150
90 DIM A$(20),A(20)
100 T$="AFTER DEFINING A$(20),B$
(20)":GOSUB 150
110 A$="A$":A$=USR0(A$)
120 T$="AFTER DELETING A$":GOSUB
 150
130 STOP
140 'PRINT CALUES
150 PRINT:PRINT T$
160 PRINT "START ARRAYS =";PEEK(
29)*256+PEEK(30);"END ARRAYS =";
PEEK(31)*256+PEEK(32);
170 PRINT "MEM FREE =";MEM
180 RETURN
190 PRINT "SETTING UP ML":LN=260
:T=0:FOR X=0 TO 99 STEP 50:IF X<
49 THEN N=50 ELSE N=54
200 PRINT LN;:A=0:FOR Y=0 TO N-1

210 READ C$:B=VAL("&H"+C$):A=A+B
:POKE M+T,B:T=T+1
220 NEXT Y:READ C$:IF A<> VAL("&
H"+C$) THEN PRINT "ERROR IN LINE
 NO";LN:STOP
230 LN=LN+10:NEXT X:
240 RETURN
250 '
260 DATA E6,84,AE,2,9F,40,3A,9F,
42,9E,40,A6,80,9C,42,22,4F,81,41
,25,F4,81,5A,22,F0,E6,80,C1,24,2
7,42,C1,41,25,42,C1,5A,22,3E,DD,
44,E6,80,C1,24,26,6,DC,44,CA,164
9
270 DATA 80,DD,44,DE,44,9F,40,9E
,1D,11,A3,84,27,A,EC,2,30,8B,9C,
1F,25,F3,20,BF,EC,2,33,8B,11,93,
1F,27,9,A6,C0,A7,80,11,93,1F,25,
F7,9F,1F,20,A9,39,C6,80,20,CE,5F
,20,CB,166A
```

End

Graftext

By Keiran Kenny

I hope CoCo2 users, or CoCo3 users programming in CoCo2 mode, will find this graphic character set useful. It will allow use of all characters accessible from the keyboard.

The string for each character in lines 1000 to 1880 is labelled Z$ plus its ASCII (CHR$) number in brackets. An exception is Z$(95), SHIFT-UP ARROW, which has been given a string for the double quote. In a screen listing it shows as a left-arrow and, in a listing printout, as an underline. But note that, if your are in EDIT mode, it can only be inserted into a line if you are using an imported DOS like ADOS3.

The character set will give you a safe 38 characters per graphic screen line. This value, LL, is established in line 1890 and can be varied if you want shorter lines. You can put into a program line as much text as a line will take.

Subroutine 80 will give you wraparound so that no words are split at the end of a line. The vertical coordinate will be increased by 15 after each line end. The value, C=C+15, in line 110 can be decreased if you would prefer less space between screen lines but make sure you leave enough space for the descenders on lower case letters.

To center a line on the graphic screen, set your horizontal coordinate after your text string, as in line 2060.

If you want input to follow a prompt, as in line 1970, then GOSUB150. When you input text, set your horizontal and vertical coordinates and GOSUB160. Depending on how far left you begin, you can input a single text line of up to 36 characters. In-program text is spaced proportionately while direct-input text is spaced evenly. You can use the left-arrow to backspace and delete typing errors before you press ENTER.

When you call the input subroutine you switch to upper/lower case mode and back to all capitals mode when you exit. Text strings are returned as NM$ and values as V.

Line 10 establishes the values for the hi/lo speed pokes: SP=65497 and SL=65496 if you are using a CoCo3, or SP=65495 and SL=65494 for other CoCos.

Your own program can begin from line 1910. You have space for subroutines between lines 270 and 1000. Generally, subroutines in the beginning of a listing will execute faster.

Listing overleaf

```
0 'GRAFTEXT' COPYRIGHT 1990
   KEIRAN KENNY
10 CLEAR2000
20 IFPEEK(&HFFFE)*256+PEEK(&HFFF
F)=&H8C1B THENSP=65497:SL=65496E
LSESP=65495:SL=65494
30 POKESP,0
40 DIM Z$(122)
50 GOTO1000
60 C=C+15:ZL$="PRESS ANY KEY.":B
=131-INT(LEN(ZL$)*3.5):GOSUB80
70 EXEC44539:K$=INKEY$:B=0:RETUR
N
80 IFLEN(ZL$)<=LL THEN120
90 FORT=LL TO1STEP-1:IFMID$(ZL$,
T,1)=" "THEN110
100 NEXTT:GOTO120
110 P$=LEFT$(ZL$,T):W$=P$:GOSUB1
30:ZL$=RIGHT$(ZL$,(LEN(ZL$))-T):
C=C+15:GOTO80
120 W$=ZL$
130 DRAW"BM"+STR$(B)+","+STR$(C)

140 COLOR0:FORZB=1TOLEN(W$):DRAW
Z$(ASC(MID$(W$,ZB,1)))+"BR3":NEX
T:RETURN
150 LZ=LEN(ZL$)*7:B=B+LZ
160 POKE282,0:BT=B:NM$=""
170 K$=INKEY$:IFK$=""THEN170
180 IFK$=CHR$(8)ORK$=CHR$(13)THE
N190
190 IFB>248ANDK$<>CHR$(8)ANDK$<>
CHR$(13)THEN170
200 IFB=BT ANDK$=CHR$(8)THEN170E
LSEIFK$=CHR$(8)THEN210ELSEIFK$=C
HR$(13)THEN260ELSE230
210 COLOR5:LINE(B-7,C-7)-(B,C+3)
,PSET,BF:B=B-7
220 NM$=LEFT$(NM$,LEN(NM$)-1):GO
TO170
230 NM$=NM$+K$:ZL$=K$:GOSUB80
240 B=B+7
250 GOTO170
260 V=VAL(NM$)
270 B=0:POKE282,255:RETURN
999 'A-Z
1000 Z$(65)="U4E2F2D2NL4D2"
1010 Z$(66)="U6R3FDGNL3FDGNL3BR"

1020 Z$(67)="BR3NEL2HU4ER2FBD5"
1030 Z$(68)="U6R3FD4GNL3BR"
1040 Z$(69)="NR4U3NR3U3R4BD6"
1050 Z$(70)="U3NR3U3R4BD6"
1060 Z$(71)="BR3EU2NLD2GL2HU4ER2
FBD5"
1070 Z$(72)="U3NR4U3BR4D6"
1080 Z$(73)="BRR2LU6NLRBD6BR"
1090 Z$(74)="BRHUBU4BR4D5GNL3BR"
1100 Z$(75)="U6D3RNE3F3"
1110 Z$(76)="NU6R4"
1120 Z$(77)="U6F2E2D6"
1130 Z$(78)="U6F4U4D6"
1140 Z$(79)="BRHU4ER2FD4GNL2BR"
1150 Z$(80)="U6R3FDGNL3BRBD3"
1160 Z$(81)="BRHU4ER2FD3G2NLBU2F
2"
1170 Z$(82)="U6R3FDGL3RF3"
1180 Z$(83)="BRNHR2EH4ER2FBD5"
1190 Z$(84)="BR2U6L2R4BD6"
1200 Z$(85)="BRHU5BR4D5GNL2BR"
1210 Z$(86)="BU6D4F2E2U4BD6"
1220 Z$(87)="NU6E2F2NU6"
1230 Z$(88)="UE4UBL4DF4D"
1240 Z$(89)="BU6DF2ND3E2UBD6"
1250 Z$(90)="BU6R4DG4DR4"
1259 '0-9
1260 Z$(48)="BRHU4ER2FBGNG2BED4G
NL2BR"
1270 Z$(49)="R4L2U6NG2BR2BD6"
1280 Z$(50)="BU5ER2FDGL2GD2R4"
1290 Z$(51)="BU5ER2FDGNL2FDGL2NH
BR3"
1300 Z$(52)="BR3U6G3R4BD3"
1310 Z$(53)="BRNHR2EU2HL3U2R4BD6
"
1320 Z$(54)="BUU3NE2BD3FR2EUHL2G
BD2BR4"
1330 Z$(55)="BU5UR4DG4DBR4"
1340 Z$(56)="BRHUER2EUHL2GDFR2FD
GNL2BR"
1350 Z$(57)="BRNHR2EU4HL2GDFR3BD
3"
1359 'a-z
1360 Z$(97)="BRNR3HER3UHNLFD3"
1370 Z$(98)="NU6R3EU2HL3D4BR4"
1380 Z$(99)="BR3NEL2HU2ER2FBD3"
1390 Z$(100)="BRNR3HU2ER3NU2D4"
1400 Z$(101)="BRNR3HU2ER2FDNL4BD
2"
1410 Z$(102)="BRU3NRNLU2ERFBD5"
1420 Z$(103)="BRNR3HU2ER3D6GNL2E
U2"
1430 Z$(104)="U6D3ER2FD3"
1440 Z$(105)="BRHU2BU2NUBD5BRREB
D"
1450 Z$(106)="BR2D2GNLEU6BU2UBD7
"
1460 Z$(107)="U6D4R2NE2F2"
1470 Z$(108)="NU6"
1480 Z$(109)="U3EFND2EFD3"
1490 Z$(110)="U4DER2FD3"
1500 Z$(111)="BRNR2HU2ER2FD2GBR2
"
1510 Z$(112)="U4R3FD2GL3ND3BR4"
1520 Z$(113)="BRNR3HU2ER3D7U3"
1530 Z$(114)="U4DER2FBD3"
1540 Z$(115)="R3EHL2HER3BD4"
1550 Z$(116)="BR2RNELHU3NRNLU2BR
3BD6"
1560 Z$(117)="BRHU3BR4D3GNL2BR"
1570 Z$(118)="BR2H2U2BR4D2G2BR2"

1580 Z$(119)="NU4E2F2NU4"
1590 Z$(120)="E4BL4F4"
1600 Z$(121)="BRHU3BR4D3GNL2BRNU
D2GNL2EU2"
1610 Z$(122)="BU4R4G4R4"
1619 'SIGNS/PUNCTUATION
1620 Z$(32)="BR" 'SPACEBAR
1630 Z$(33)="UBU2U3BD6" '!
1640 Z$(35)="BRU2NLU2NLU2BR2D2NL
NRD2NLNRD2BR" '#
1650 Z$(36)="BUR3EHL2HERNUND5R2B
D5" '$
1660 Z$(37)="BUE4BL4NUBF4D" '%
1670 Z$(38)="BRNRHUE3HGF3NENG2FB
D"
1680 Z$(39)="BRBU4U2BD6" 'APOSTR
OPHE
1690 Z$(40)="BR2H2U2E2BD6" '(
1700 Z$(41)="E2U2H2BD6BR2" ')
1710 Z$(42)="BU3BR4L2NU2NE2NR2NF
2ND2NG2NL2NH2R2BD3" '*
1720 Z$(43)="BR2BU3ND2NL2NU2NR2B
D3" '+
1730 Z$(44)="BRNGNU" ',
1740 Z$(45)="BU2BR2R2BR2BD2" ' -

1750 Z$(46)="BRNUBR2" '.
1760 Z$(47)="UE4UBD6" '/
1770 Z$(58)="BR3UBU2UBR3BD4" ':
1780 Z$(59)="BR2NGUBU2UBD4" ';
1790 Z$(60)="BU3NE3F3" '<
1800 Z$(61)="BU2BRR2BU2NL2BD4" '
=
1810 Z$(62)="E3NH3BD3" '>
1820 Z$(63)="BR2UBU2REUHLGBD6BR3
" '?
1830 Z$(64)="BRHUER2FU2HL2GBD3FR
2EBD" '@
1840 Z$(91)="NR2U6R2BD6" '[
1850 Z$(92)="BU5NUF4D" '\
1860 Z$(93)="R2U6NL2BD6" ']
1870 Z$(94)="BR2U6NG2F2BD4" '^
1880 Z$(95)="BU5UBR2DBD5" '"
1890 LL=38
1900 PMODE4,1:COLOR0,5:PCLS:SCRE
EN1,1
1910 B=0:C=10:ZL$="This is a sam
ple of what you can do with the
character set in this program. T
o include text in your program l
ines set your horizontal coordin
```

```
ate, B, and your vertical, C, as
 in this line (1910).":GOSUB80
1920 C=C+15:ZL$="Then put your t
ext into a string labelled ZL$ f
ollowed by GOSUB80, as below:":G
OSUB80
1930 C=C+15:ZL$="B=0:C=10:ZL$=_Y
our text_:GOSUB80":GOSUB80
1940 GOSUB60
1950 PCLS
1960 B=0:C=10:ZL$="You can also
ENTER values directly onto the g
raphic screen as below (line 197
0). What you type will follow th
e prompt if you follow it with G
OSUB150.":GOSUB80
1970 C=C+15:ZL$="<ENTER> radius
(max. 45):":GOSUB80:GOSUB150
1980 CIRCLE(128,136),V
1990 C=116+V+15:GOSUB60:PCLS
2000 C=10:ZL$="You can type text
 directly onto the graphic scree
n. Set your horizontal coordinat
e, B, your vertical C and then G
OSUB160 (as in line 2010). Type
a short text below.":GOSUB80
2010 B=0:C=C+15:GOSUB160
2020 C=C+15:ZL$="You typed:":GOS
UB80
2030 C=C+15:ZL$="_"+NM$+"_":GOSU
B80
2040 C=C+15:ZL$="Here are all th
e signs:":GOSUB80
2050 C=C+15:ZL$="! # $ % & ' ( )
 * : = - @ ,  . / \ _Quotes_ < >
 ? [ ]":GOSUB80
2060 C=C+15:ZL$="To END, press a
ny key.":B=131-INT(LEN(ZL$)*3.5)
:GOSUB80
2070 GOSUB70:POKESL,0:CLS:END
```

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.

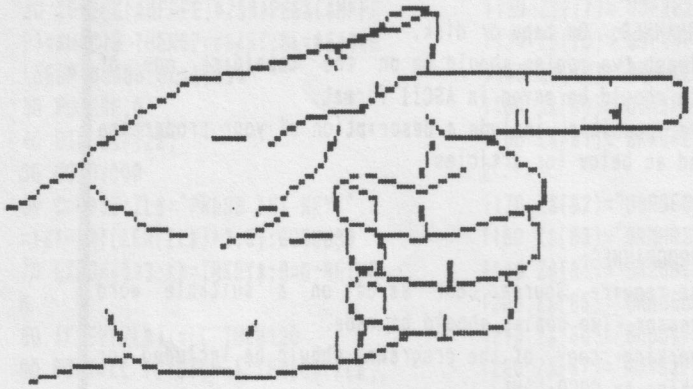
ML PROGRAMMES:
These require Source code saved on a suitable word processor. Two copies should be made.
A working copy of the programme should be included for checking by COCO-LINK.

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.



Better BASIC Part 12

*Random Number Spreads
****
by Keiran Kenny

Recently I wrote to ask our editor's advice on how to secure a better spread of random values in a program, but I have since done some reading on the subject and submit the following short listings by way of illustration. Each will pick seven random numbers out of forty-four.

The simplest version is:

```
0 'RND1
10 CLS
20 FORZ=1TO7
30 PRINTRND(44);
40 NEXT
50 N=N+1:IFN/10=INT(N/10)THEN60ELSEPRINT:GOTO20
60 PRINT:PRINTTAB(6)"PRESS ANY KEY":EXEC44539:GOTO10
```

Using RND(-TIMER) to reset the random number generator each time round the loop is said to give a better spread but any improvement is not very apparent. Compare the results for RND2, below, with that for RND1 above.

```
0 'RND2
10 CLS
20 FORZ=1TO7
30 X=RND(-TIMER)
40 PRINTRND(44);
50 NEXT
60 N=N+1:IFN/10=INT(N/10)THEN70ELSEPRINT:GOTO20
70 PRINT:PRINTTAB(6)"PRESS ANY KEY":EXEC44539:GOTO10
```

One author suggested trying PRINTINT(RND(-TIMER)*44) but this will produce zero, which you probably will not want, and will not produce forty-four. Line 30 of RND3, below, will eliminate that objection.

In the above cases, you might find that the computer seems to pick the same value or consecutive values more often than you might want in a program, and their choices tend to cluster around a range of numbers.
I think that this occurs less in RND3.

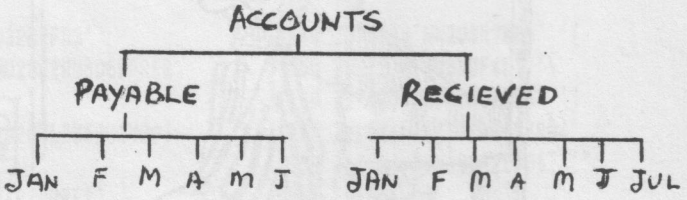
```
0 'RND3
10 CLS
20 FORZ=1TO7
30 PRINTINT(RND(-TIMER)*44)+1;
40 NEXT
50 N=N+1:IFN/10=INT(N/10)THEN60ELSEPRINT:GOTO20
60 PRINT:PRINTTAB(6)"PRESS ANY KEY":EXEC44539:GOTO10
```

Try each listing a few times and decide for yourself. With a little extra code you could turn any of the three examples into a program to pick lotto numbers.

END

OOPS ! ! !

The following should appear on page 29 at the "add example" section. The complete set of pages were printed before this ommission was spotted.



A Typical pathlist to get to the accouns payable in January would be /d0/accounts/payable/jan

Draw 124

By Richard and Johanna Vagg

This programme was originally written by Richard on the Tandy 1000HX. We converted it for the Coco 3. It is intended as a "fun" drawing programme. It can be used 'seriously' to create screens of fancy writing - if you choose the LINE option (in one place), and have lots of patience. (You will probably want to remove the high speed POKE for serious drawing). Th HX version has a 'pen-up' option, and SAVE/LOAD routines. These options would be required for serious work.

The subroutine at line 1000 is there to put some colour into PALETTEs 8, 10, 12 and 14. The computer chooses random colours for these PALETTEs. You begin drawing with COLOR1. Use the up arrow to change to COLOR2, then 3 etc. After COLOR15 comes COLOR0, ie black, the background. Change the subroutine at line 1000 if you want specific colours in the PALETTE slots.

It might seem strange to pick two numbers for the dimensions of a line.... I won't explain.. just experiment, and have fun.

```
10 CLS:POKE65497,0:ON BRK GOTO 6
80
20 ' DRAW124 by Richard and
    Johanna Vagg
25 RGB:WIDTH32:CLS
30 PRINT" Using the joystick, y
ou can      draw in one,two or
four            places at once
."
40 PRINT:PRINT"You can use these
 keys:"
50 PRINT" SPACEBAR - to clear s
creen      M       - to return
to menu    Up arrow - to change
colors"
60 PRINT:PRINT" also try P for
an interesting      effect"
70 INPUT"DRAW WITH..  1 FOR LINE
S                     2 FOR BOXE
S OR                  3 FOR CIRC
LES";N
80 IF N<1 OR N>3 THEN 70
90 PRINT:PRINT:INPUT "DRAW  1, 2
, OR 4 AT A TIME";JV
100 IF JV=3 OR JV>4 THEN 90
110 IF JV<3 THEN X=0:Y=100
120 IF JV=4 THEN X=160:Y=100
130 IF N<3 THEN INPUT"WIDTH AND
DEPTH OF LINE/BOX ..        2 NU
MBERS";W,L:GOTO 150
140 INPUT"RADIUS";W
150 IF N>1 THEN Q=W/2 ELSE Q=1
155 GOSUB1000
160 HSCREEN 2
170 PALETTE 0,0
180 CLS
190 C=1
200 J=JOYSTK(0):K=JOYSTK(1)
210 IF J<18 THEN X=X-Q
220 IF J>40 THEN X=X+Q
230 IF K<18 THEN Y=Y-Q
240 IF K>40 THEN Y=Y+Q
250 IF X<0 THEN X=319
260 IF X>319 THEN X=0
270 IF Y<0 THEN Y=199
280 IF Y>199 THEN Y=0
290 ON N GOSUB 310,470,550
300 GOTO 200
310 HLINE(X,Y)- (X+L,Y+W),PSET
315 GOSUB 380
320 IF JV=1 THEN RETURN
330 HLINE(X,199-Y)-(X+L,199-Y+W)
,PSET
335 GOSUB 380
340 IF JV=2 THEN RETURN
350 HLINE(319-X,199-Y)-(319-X+L,
199-Y+W),PSET
360 HLINE(319-X,Y)-(319-X+L,Y+W)
,PSET
380 A$=INKEY$
390 IF A$=" " THEN HCLS
400 IF A$=CHR$(94) THEN C=C+1
410 IF C>15 THEN C=0
420 HCOLORC
430 IF A$="P" THEN GOSUB630
435 IF A$="M" THEN 25
440 RETURN
450 GOSUB 380
460 RETURN
470 HLINE(X,Y)- (X+L,Y+W),PSET,B

475 GOSUB 380
480 IF JV=1 THEN RETURN
490 HLINE(X,199-Y)-(X+L,199-Y+W)
,PSET,B
495 GOSUB 380
500 IF JV=2 THEN RETURN
510 HLINE(319-X,199-Y)-(319-X+L,
199-Y+W),PSET,B
520 HLINE(319-X,Y)-(319-X+L,Y+W)
,PSET,B
530 GOSUB 380
540 RETURN
550 HCIRCLE(X,Y),W,C
555 GOSUB 380
560 IF JV=1 THEN RETURN
570 HCIRCLE(X,199-Y),W,C
575 GOSUB 380
580 IF JV=2 THEN RETURN
590 HCIRCLE(319-X,199-Y),W,C
600 HCIRCLE(319-X,Y),W,C
610 GOSUB 380
620 RETURN
630 FOR O=1 TO 20
640 FOR P=1 TO 15:PALETTE P,RND(
63):NEXT
650 NEXT
660 GOSUB1000
670 RETURN
680 RGB:POKE65496,0:END
1000 RGB:PALETTE0,0:FOR P=8 TO 1
4 STEP 2:PALETTEP,RND(63):NEXT:R
ETURN
```

SEE 2 * SEE

By Johanna Vagg

George McLintock wrote a very fast programme to TRANSFER PMODE3 or 4 screens to the HSCREEN. Mine is rather slow (in BASIC), but it is different - you get two copies on the one HSCREEN. One copy is a mirror image of the other.

The programme PEEKs the PMODE4 screen and LPOKEs on to the HSCREEN3. I wrote a routine to produce a mirror image of a PMODE4 screen in 1987 (line 300). I tried a similar routine for the HSCREEN. It produced a 'vertical blind' effect. Apparently you can not HGET and HPUT a strip narrower than one byte. On the PMODE screen, you can GET and PUT strips of only one pixel if required. That is why the PMODE4 screen is reversed, then transferred to the HSCREEN.

```
 90 POKE38345,57:POKE65314,48
 95 CLS:PRINT:PRINT"SEEING DOUBLE
 !?                    BY Johanna
  Vagg"
 97 FOR T=1 TO 1000:NEXT
 100 CLS:PRINT:PRINT"THIS PROGRAM
  ASSUMES YOUR PMODE4     SCREEN
 IS LOADED.          THE PROGRAM
 WILL TRANSFER THAT  SCREEN TO TH
 E HSCREEN3 - YOU    SPECIFY HOW
 FAR ACROSS -IN POKES"
 105 PRINT:PRINT"THE PROGRAM WILL
  THEN REVERSE   THE PMODE SCREEN
  AND TRANSFER   THE NEW SCREEN A
 S WELL...       PRODUCING TWINS!
 "
 110 PRINT:PRINT:PRINT"      ANY K
 EY TO CONTINUE"
 120 EXEC44539
 125 CLS:PRINT:PRINT"YOU WILL BE
 GIVEN THE CHANCE    TO SAVE THE
 TWINS.":PRINT:PRINT:PRINT"TO LOA
 D A TWIN FILE OR ANOTHER  HSCREE
 N3 FILE (SAVED IN THE SAMEWAY),
 "RUN500"
 135 PRINT:PRINT:PRINT"      PRESS
  ANY KEY TO BEGIN"
 137 EXEC44539
 140 POKE65497,0
 141 ON BRK GOTO 400
 142 POKE38345,57:POKE65314,48
 143 CLS:INPUT" WHICH DRIVE WILL
 YOU BE USING";D
 144 DRIVE D
 145 HSCREEN3:HCLS1
 150 DIM H(32),A(10),B(10)
 152 ON BRK GOTO 400
 170 PALETTE0,0:PALETTE1,63
 180 S=3:PM=3584
 186 GOSUB 190
 187 GOSUB 295
 188 GOSUB 190
 189 POKE65496,0:FOR T=1 TO 2000:
 NEXT:GOTO 450
 190 HSCREEN0:CLS:INPUT"HOW MANY
 POKES ACROSS            SUGGESTIO
 N: 8 THE FIRST TIME,      40 THE
  SECOND TIME";L
 192 ST=393216+L
 195 POKE&HE6C6,33
 197 HSCREEN3:POKE&HFF9A,63
 200 V=0
 205 FOR H=0 TO 31
 210 PMODE4,1
 215 PP=PM+V*32+H
 220 H(H)=PEEK(PP):NEXT
 225 FOR X=0 TO 31:LPOKEST+X,H(X)
 :NEXT
 230 V=V+1
 235 ST=ST+80
 240 IF V<192 THEN 205
 242 SOUND 100,2
 245 RETURN
 295 HSCREEN0:CLS:PRINT"PATIENCE
 PLEASE...               REVER
 SING PMODE SCREEN"
 300 PMODE4:FOR X=0TO127:GET(X,0)
 -(X,191),A,G:GET(255-X,0)-(255-X
 ,191),B,G:PUT(X,0)-(X,191),B,PSE
 T:PUT(255-X,0)-(255-X,191),A,PSE
 T:NEXTX:RETURN
 400 DRIVE0:RGB:POKE65496,0:END
 450 HSCREEN0:CLS:INPUT"WOULD YOU
  LIKE TO SAVE YOUR       seeing
  double picture";A$
 457 IF A$<>"N" THEN 600
 458 GOTO 680
 500 POKE65496,0:POKE&HE6C6,141:C
 LS:PRINT:PRINTTAB(2)"LOAD FILENA
 ME (NO EXTENSION)":LINEINPUTF$
 510 PALETTE 0,0:PALETTE1,63
 520 HSCREEN 3:POKE&HFF9A,63:FOR
 M=&H70 TO &H71
 530 POKE&HFFA2,M
 540 FI$=F$+"/HR"+HEX$(M-&H70)
 550 LOADM FI$
 560 NEXT
 570 POKE&HFFA2,&H7A
 580 GOTO 580
 600 POKE65496,0:POKE&HE6C6,141:C
 LS:PRINTTAB(2)"SAVE FILENAME (NO
  EXTENSION)":LINEINPUTF$
 610 IF FREE(D)<8 THEN PRINT"NOT
 ENOUGH SPACE - CHANGE DISKS":PRI
 NT"  (PRESS ANY KEY TO CONTINUE)
 ":EXEC44539:GOTO 610
 620 FOR M=&H70 TO &H71
 630 POKE&HFFA2,M
 640 FI$=F$+"/HR"+HEX$(M-&H70)
 650 PRINT"SAVING: "+FI$
 660 SAVEM FI$,&H4000,&H5FFF,&H40
 00
 670 NEXT
 680 POKE&HFFA2,&H7A
 684 POKE&HE6C6,18
 685 HSCREEN3:POKE&HFF9A,63
 686 GOTO 686
```

End

MCVAGG6

By Johanna Vagg

MCVAGG6 is another version of MCVAGG. It has 6 designs instead of the original 3. (McVagg because Mal Mc Lauchlin helped me create the first 3 designs). The new programme is NOT for the DMP110. You can still get the other three designs on your DMP110 by placing the new DATA in the old DATA lines. This would also be the quickest way to get the extra designs on your DMP105/106/130.

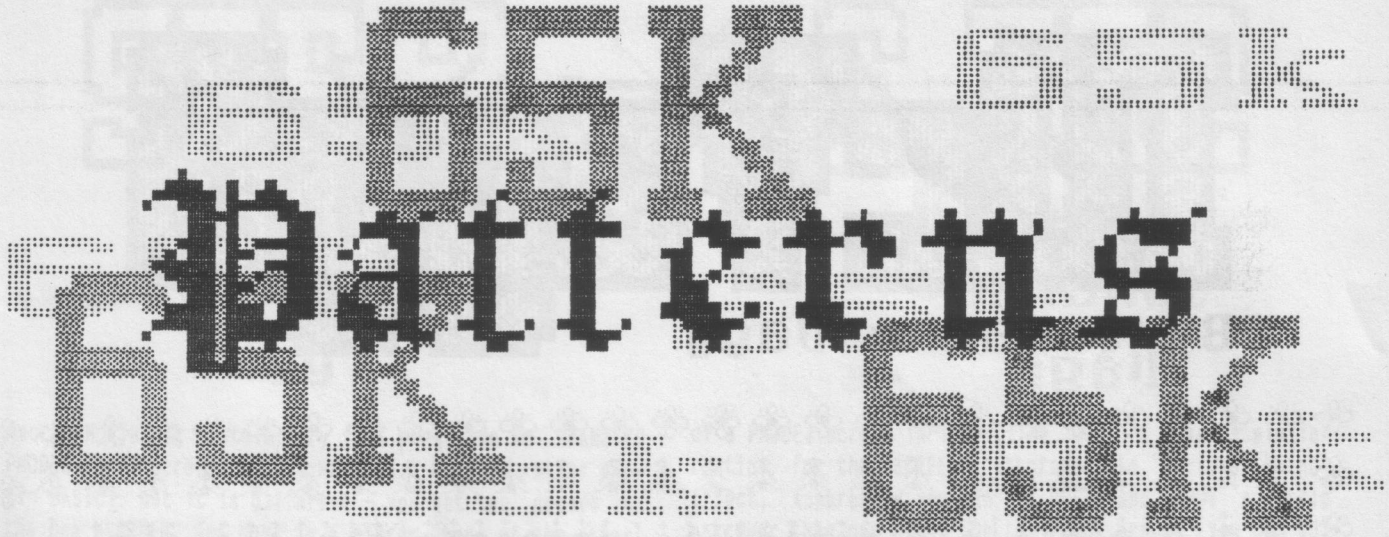
I wanted all six designs in one programme and I saw that there had to be a more efficient way to READ the DATA - see lines 80 to 130 in MCVAGG. They are a bit repetitious! Then look at lines 80 to 120 in MCVAGG6. These few lines handle twice as much DATA.

I have made use of a subroutine to print the top and bottom of the border. You might notice too, that the DATA lines are slightly different. The commas are not mistakes and not unnecessary.

I have added something to make the programme more user friendly - it asks how many repeats you would like down the page.

Listing on

Page 27



By Keiran Kenny

George McLintock's CoCo 3 Screen Dumps (Dec. '89) allow a user to dump hi-res graphic screens in which attractive masks are substituted for the color areas on the graphic screen. If you have a CoCo 2 you could take the POKE178 route to paint the shapes on your graphic screen and improve their appearance in screen dumps. But POKE178 patterns are formed by vertical lines and, as such, are not very attractive and do not offer much variety.

However there is a routine in Microcom's "Supplement to 500 Pokes, Peeks 'n Execs" that offers 65000 paint patterns on the PMODE4 screen. Actually, there is a lot of repetition so that there would be much less than a full 65000 distinguishable patterns. The patterns are also based on vertical lines but in some of them the verticals are barely visible so that they print out like very fine half-tone screens.

I have seen about 4300 of these patterns! As with POKE178, on a PAL TV screen the colors are unpredictable and not always uniform.

When the values in line 20, and the data values in lines 40-50, have been poked into memory, this program will allow you to experiment by inputting values to be poked into positions 254 and 255 and paint a circle (lines 120 and 130). The numbers below the circle are the values poked into 254 and 255 respectively.

For very fine paint patterns in which the vertical lines are barely visible, try number pairs: 1, 84; 9, 246; 45, 210 and 14, 214.
Numbers 26 and 244 will produce an attractive embossed pattern. Press any key to exit the graphic screen. You can then save your picture and/or print it/try another/ end by following the prompts. If you are using a tape recorder substitute CSAVEMPX$,1536,7679,40999 for the save routine in line 1570.

The maximum print position in line 1700 is for my DMP-130A printer and allows for the fact that I can only select standard print in graphic mode. You could change this to suit your printer.

Keep your printouts for reference or note any values that produce interesting patterns and try putting your own artwork into lines 90 - 1490 of the listing. You may find it worthwhile to keep the character set for the numbers (lines 70 and 80). Use the routine in lines 140 - 160 to put your numbers on the screen.

```
0 '65KPATTS' by Keiran Kenny
10 CLS
19 'Lines 20-50: "Supplement to
    500 Pokes Peeks 'n Execs" by
    Microcom Software
20 POKE&H99FF,&HE2:POKE&H9A00,0
30 FORI=&HE200 TO &HE244:READA$:
POKEI,VAL("&H"+A$):NEXT
40 DATA 34,76,1F,10,90,BA,DD,F5,
86,20,97,FD,8D,1F,1F,89,4F,DD,F5
,86,02,97,FD,8D,14,DC,F7,5D,26,0
6
50 DATA 98,FE,97,B5,20,04,96,FF,
97,B5,35,76,7E,93,77,86,08,97,FC
,DC,F5,58,49,91,FD,25,03,90,FD,5
C,0A,FC,26,F3,1E,89,DD,F7,39
60 DIML$(57)
70 L$(48)="BRHU4ER2FBGNG2BED4GNL
2BR":L$(49)="R4L2U6NG2BR2BD6":L$
```

```
(50)="BU5ER2FDGL2GD2R4":L$(51)="
BU5ER2FDGNL2FDGL2NHBR3":L$(52)="
BR3U6G3R4BD3":L$(53)="BRNHR2EU2H
L3U2R4BD6"
80 L$(54)="BUU3NE2BD3FR2EUHL2GBD
2BR4":L$(55)="BU5UR4DG4DBR4":L$(
56)="BRHUER2EUHL2GDFR2FDGNL2BR":
L$(57)="BRNHR2EU4HL2GDFR3BD3"
89 'Your program begins here:
90 INPUT"VAL 1";H
100 INPUT"VAL 2";V
110 PMODE4,1:COLOR0,5:PCLS:SCREE
N1,1
120 CIRCLE(62,96),60
130 POKE254,H:POKE255,V:PAINT(62
,155),0,0
140 B=40:C=166:W$=STR$(H)+","+ST
R$(V)
150 DRAW"BM"+STR$(B)+","+STR$(C)

160 FORZB=1TOLEN(W$):DRAWL$(ASC(
MID$(W$,ZB,1)))+"BR3":NEXT
1500 EXEC44539
1510 CLS:PRINT@128,"SAVE? Y/N"
1520 K$=INKEY$:IFK$=""THEN1520
1530 IFK$="Y"THEN1560
1540 IFK$="N"THEN1580
1550 GOTO1520
1560 INPUT"SAVE PIXNAME";PX$
1570 SCREEN1,1:SAVEMPX$,3584,972
7,40999:K$=INKEY$
1580 CLS:PRINT@128,"(P)RINT/(T)R
Y ANOTHER/(E)ND?"
1590 K$=INKEY$:IFK$=""THEN1590
1600 IFK$="P"THEN1640
1610 IFK$="T"THENCLS:RUN60
1620 IFK$="E"THENCLS:END
1630 GOTO1590
1639 'Screen dump adapted from
     PICPRINT by Geoff Donges
1640 CLS:POKE150,18:' 2400 BAUD
1650 INPUT"sTANDARD eLITE cONDEN
SED";SZ$
1660 IF SZ$="S" THEN MX=224:SZ=1
9
1670 IF SZ$="E" THEN MX=319:SZ=2
3
1680 IF SZ$="C" THEN MX=542:SZ=2
0
1690 SOUND250,2:PRINT
1700 PRINT"MAXIMUM PRINT POSITIO
N ="MX
1710 PRINT@160,"0 --------------
------------"MX:PRINT:PRINT"SET
PRINTER PAPER !!":PRINT
1720 PRINT#-2,CHR$(30)CHR$(27)CH
R$(SZ);
1730 P=256*PEEK(&H25)+PEEK(&H26)
:P=P-&HA4:CLEAR200,P
1740 P=256*PEEK(&H25)+PEEK(&H26)
:P=P-&HA4:FORI=&HE200 TO &HE244:
READA$:NEXT:FORX= 0 TO &HA4:READ
A$:A=VAL("&H"+A$):POKE P+X,A:NE
XT
1750 INPUT"PRINT POSITION";PN
1760 SOUND250,1
1770 IF PN>255 THEN PN=PN-255:PM
=PM+1
1780 SCREEN1,1
1790 POKE&H407,PM:POKE&H408,PN:E
XECP:SOUND250,2:EXEC &H8C1B
1800 DATA86,FE,97,6F,86,12,AD,9F
,A0,2,9E,BA,BF,4,0,30,89,17,A0,B
F,4,5,7F,4,4,10,BE,4,7,10,8C,0,0
,27,A,86,80,AD,9F,A0,2
1810 DATA 31,3F,20,F0,10,8E,80,1
,10,BF,4,2,10,8E,0,0,BE,4,0,86,8
0,E6,84,F4,4,2,26,3,B8,4,3,31,21
,30,88,20,10,8C,0,3,2D
1820 DATA 7,BC,4,5,2D,2,20,9,78,
4,3,10,8C,0,7,26,DB,74,4,2,AD,9F
,A0,2,86,1,B7,4,3,B6,4,2,81,0,26
,BF,BE,4,0,30,1
1830 DATA BF,4,0,B6,4,4,4C,B7,4,
4,81,20,2D,A4,7F,4,4,BE,4,0,30,8
9,0,C0,86,D,AD,9F,A0,2,BC,4,5,2E
,6,BF,4,0,16,FF,75
1840 DATA 39
```

```
0 REM MCVAGG6 BY JOHANNA VAGG
  MC FOR THE HELP GIVEN BY MAL
  MCLAUCHLAN (WITH THE FIRST 3
  DESIGNS)
5 POKE150,18'SETS BAUD RATE
10 CLS:CLEAR1000
35 DIM A$(12)
40 G$=CHR$(30)+CHR$(27)+CHR$(19)
+CHR$(18):GS$=CHR$(30)+CHR$(27)+
CHR$(20)+CHR$(27)+CHR$(83)+CHR$(
0):Y=1:YY=208:TA=122:W=30:D=128
50 T1$=CHR$(27)+CHR$(16)+CHR$(0)
+CHR$(0)
60 TA$=CHR$(27)+CHR$(16)+CHR$(Y)
+CHR$(YY)
70 PRINT:PRINT"    READING DATA.
.."
80 FOR X=1 TO 12
90 READ A:IF A=999 THEN 120 ELSE
 A=A+D
100 A$(X)=A$(X)+CHR$(A)
110 GOTO 90
120 NEXT
140 CLS:PRINT"PLEASE CHOOSE":PRI
NT:INPUT"  1) FLOWERS
           2) TREES
           3) PEOPLE
           4) NOTES
           5) COMPUTERS
           6) HEARTS";Q
150 IF Q>6 THEN 140
160 Q=Q*2
170 INPUT"how many repeats (34 m
aximum)";PP
190 DATA ,,,,96,16,20,16,110,113
,113,110,16,20,16,96,999
200 DATA ,,,,1,2,10,2,29,35,35,2
9,2,10,2,1,999
210 DATA ,,,,,,,64,36,18,127,18,
36,64,,,999
220 DATA ,,,,,64,72,36,18,121,12
7,121,18,36,72,64,999
230 DATA ,,,98,119,98,,,,,,66,11
9,66,,,999
240 DATA 4,6,7,7,63,7,7,6,4,48,5
6,4,3,4,56,48,999
241 DATA ,64,126,6,7,99,127,,,12
4,12,12,70,126,,,999
242 DATA ,1,1,,,,,,3,3,,,1,1,,,9
99
243 DATA 126,126,2,2,2,2,2,2,2,2
,126,,,,,,999
244 DATA 3,7,14,26,50,42,42,42,4
2,42,39,40,48,32,,,999
245 DATA 120,4,2,1,1,2,2,1,1,2,4
,120,,,,,999
246 DATA 1,2,4,8,16,32,32,16,8,4
,2,1,,,,,999
250 PRINT#-2, G$;
260 GOSUB 380
270 FOR P=1 TO PP
280 PRINT#-2:PRINT#-2, T1$;
290 PRINT#-2, A$(Q-1);:PRINT#-2,
 TA$+A$(Q-1)
300 PRINT#-2, T1$;
310 PRINT#-2, A$(Q);:PRINT#-2, T
A$+A$(Q)
320 NEXT
330 PRINT#-2
340 GOSUB 380
350 PRINT#-2, GS$;
360 PRINT#-2, TAB(TA)"A Vagg Pro
duction"
370 GOTO 140
380 PRINT#-2, T1$;
390 FOR P=1 TO W:PRINT#-2, A$(Q-
1);:NEXT
400 PRINT#-2:PRINT#-2, T1$;
410 FOR P=1 TO W:PRINT#-2, A$(Q)
;:NEXT
420 PRINT#-2
430 RETURN
```

A Beginner's

Magazine commitments have kept my OS9 learning down to a minimum. Still, some little progress has been made in setting up my system.

I have bought a new printer, a DMP 133. I wanted to run it at 2400 Baud which is as fast as it will go. My Boot file has the Baud rate set at 600 and this, I thought, meant that I would have to configure a whole new Boot file. This isn't so.

A search of commands shows that XMODE can be used to change things in the Boot file. XMODE displays or changes the initialization parameters of devices such as printer, video display etc. Therefore:

OS9: xmode /p baud=4 ⟨enter⟩

was all that was required to change the Baud rate from 600 to 2400. (The figure 4 for a Baud rate of 2400 is found in the manual on page 6 - 109 of the System Command Description section). I tested it by listing the startup file to the the printer and it worked OK. The only problem with this is that when you reboot OS9 from scratch the Baud rate is again 600.

A bit of information from Ken Wagnitz put me on the right track. You can use the COBBLER command to make the XMODE changes permanent. So.....

OS9: xmode /p baud=4 ⟨enter⟩
OS9: cobbler /d0 ⟨enter⟩

Now my Boot disk sets my printer on 2400 baud ready for work. Terrific!

****

I have just found out that there can be drawbacks in using COBBLER in this way. The description of COBBLER states "If you use COBBLER on a diskette that does not have a storage block large enough to hold the Bootfile, COBBLER destroys the old Bootfile and OS9 cannot boot from that diskette." Whew! That was close.

Therefore the rule should be that, unless COBBLERing to a newly formatted disk, it is preferable to use OS9GEN. This command "creates and links the required OS9 Bootfile to a diskette." In other words it will make the memory allocation suit the bootfile.
The commands would then have looked like:

OS9: xmode /p baud=4 ⟨enter⟩
OS9: os9gen ⟨enter⟩

What this means is that if I had actually added something to the bootfile where it needed more memory than already alloted by the original bootfile I would have been in trouble again.
I wonder what error message that would have given me?

****

I think it is probably about time I cleared up some of the jargon used in OS9. Many of the words used mean the same as other words in RSDOS, so it is only a case of trying to remember these new words. The main thing is to get some of the more prominent words sorted out so as one can understand what the various articles and instruction books are talking about.
There is a very good glossary of terms at the back of the OS9 Manual. It deserves to be read thoroughly. I also like to write things down in simple terms which I can understand more easily.

OS9 BOOT - This means to load OS9. It is basically the equivalent of RUN or LOAD in RSDOS. Coco 3 has the DOS command which does just that.

DIRECTORIES - Store filenames and directory names and their locations.
A PROCEDURE FILE - is contained in the current Data Directory. The file contains a list of OS9 commands that are read and executed by the shell. A Startup file is a good example of a procedure file.
A PROCESS - Is simply a programme that is running.
DATA - is information that a programme uses or creates.
PATHLIST - Is the route or path to a device, directory or file you wish to access. I find that this is what causes most of my ERROR messages.
DEVICE - This covers most of the hardware in OS9. Devices can be disk drives, the terminal, keyboard, a modem or many other things. Device names are preceded by a " /" as in /d0. Disk Drive 0. (Don't forget that space before the slash).

There are also a few basic rules which, once understood, make delving into OS9 a little bit easier. I find that by writing these things down it helps to make them stick in my brain. Here are just a few:

1) After formatting a disk, it has one Directory. This is called the ROOT directory.
2) The Root directory becomes the current Data Directory and CMDS becomes the current Execution Directory. You must instruct OS9 if you wish to change these Directories. This is done using the chd and chx tools.
3) On initial set up, your current Execution Directory will be /d0/CMDS and your current DATA DIRECTORY will be /d0. If you change disks you will need to use the chd and chx tools to change the current Data Directory to the ROOT directory of the new disk and current Execution Directory to the CMDS directory of the new disk.
4) If you ask OS9 to execute a command or programme it automatically looks in the current Execution Directory (CMDS) unless otherwise told.
5) OS9 Directories can contain sub-directories which can contain sub-directories which can contain sub-directories etc., etc., etc. This is called the Heirearchal Directory system. It looks just like the family tree. For example:

Add Example.....

A typical pathlist to get to ..... would be .....
6) You cannot send output from a programme to a window you are using as a terminal. This will be where you see the OS9: prompt.

It is essential to understand these basic principles if one wishes to get anywhere with OS9. I sometimes wonder where I'm heading.

Coco 3 Graphics

Enhanced

Enhanced

Enhanced

Enhanced

By Lindsay Bradford

Have you ever wanted to spruce your Coco3 pictures up just that little bit? How about a border that scrolls colours donwards? Well, that's exactly what my program does. I origionally discovered the scrolling colours when I first began playing with the Coco3's graphics modes in Assembly language.

Though I won't go into just how I set up the program, (it would take a very significant amount of explaining), I will point out that the program will not work properly if you add any other commands into the loop. For those of you who enjoy playing with Machine Language, even placing a NOP instruction inside the loop will ruin the effect of the coloured bars down the screen. Using the high speed poke with this program will have the same effect.

The first few Assembly lines (lines 110-180) will generate a 320 X 200, 16 colour graphics screen at the memory location that is usually reserved for displaying Coco3 graphics screens. I have supplied a basic version of the program for those of you who are not inclined towards Assembly Language.
The program is position independant (it can be moved to any convenient area of memory). Though you can load it into any spot of memory you like, I initially saved the program at memory location $7000 (28672).
To load it somewhere else in memory, type -

LOADM"PROG",XXXX,

where XXXX is the offset from $7000 you want to load it at.
(Please note, if you want to load the program lower than $7000, figure out how much memory there is between $7000 and the end of memory ($FFFF). Also figure out how much memory is required from memory location 0 to the place you want the program loaded. Add these two figures together and use this figure for XXXX in the instruction above.

To start the program,load up your picture and then this program. Type - EXEC &H7000. If you have saved the program somewhere else in memory use EXEC XXXX. When the program executes, you will see the Coco3 graphics screen appear. If the picture you want to view is usually only 192 lines down, there will be a little garbage at the bottom of the screen (the extra 8 lines).

PLEASE NOTE - You cannot get out of the program without pressing the reset button.
I encourage other programmers to disect this program and take what pieces they find useful in it. Hopefully, this will encourage others to get into programming assembly.
Till later, Lindsay.

```
10 'GRAPHICS PROGRAM BY -
    LINDSAY BRADFORD
15 'SHAREWARE CONTRIBUTION,
     $1 WOULD BE GREATLY
        APPRECIATED.
20 'A BASIC VERSION OF MY LITTLE
  GRAPHICS PROGRAM FOR PEOPLE WHO
  DON'T HAVE AN ASSEMBLER.
30 CLEAR 30,&H7000:GOTO50
40 DATA 86,7F,B7,FF,90,86,88,B7,
FF,98,86,3E,B7,FF,99,86,C0,B7,FF
,9D,4F,4C,B7,FF,9A,20,FA
50 FORADD=&H7000 TO&H701A:READIN
F$:POKEADD,VAL("&H"+INF$):NEXT
```

ADVERTISING RATES:

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

Software:

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

Back Issues....$2.50each

CLASSIFIED ADVERTS
****

WANTED

TANDY double sided disk drive for Coco 3 or trade single drive with cash adjustment.
Also wanted. SUPER VOICE song books volume 1 Potpouri and volume 2 Nursery rhymes.
Prices to:
Jim Eadsforth (08) 298 2843.

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	

DISK 032 APPLICATION
WINNERS horse handicapping system

ALL DISKS $5.00 each

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