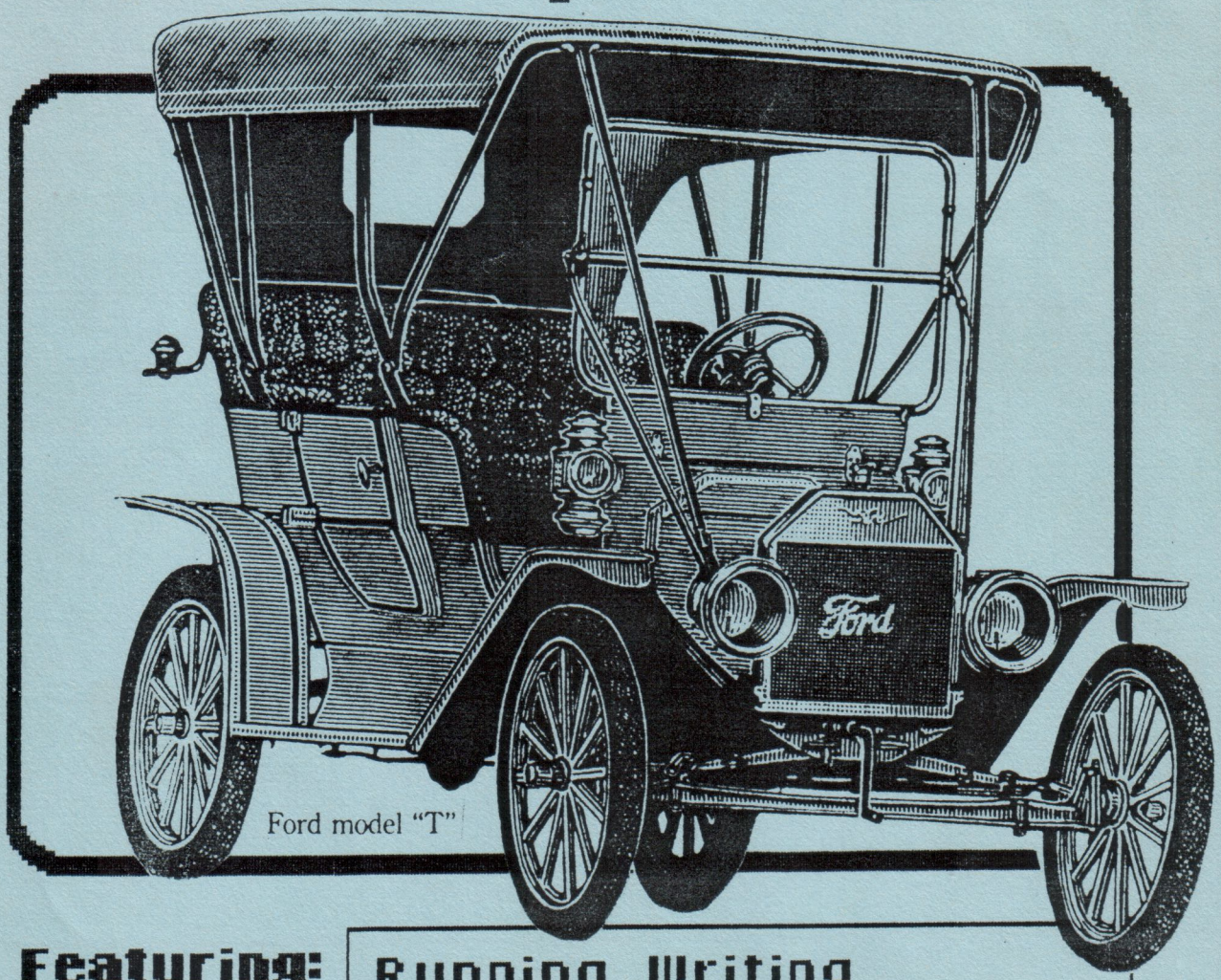
February 1991 Vol 4. No.1

COCO LINK

The Color Computer Magazine



Ford model "T"

Featuring:

Running Writing
Deluxe Control Box
Address Labeler

Contents

Departments

Columns

Features

Advertisers

COCO LINK

EDITOR:
Robbie Dalzell

ASST. EDITOR:
Garry Holder

SUB-EDITORS:
OS9:
Ken Wagnitz

Hardware:
Darren Ramsey

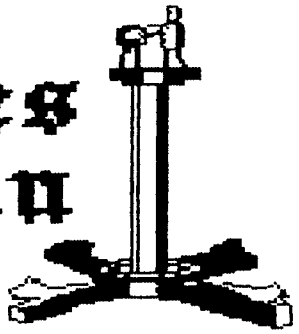
Correspondence:
Garry Holder

Submissions:
Robbie Dalzell

COCO LINK

Copyright Notice

Robbies Column



WHAT ARE THE OPTIONS

As COCO-LINK goes into it's third year of publication some things have to be faced.

1) The Color computer's life span will be limited by the interest of the people who use it.

2) If TANDY discontinue the Coco in the USA there will be a complete drying up of new software.

So, what are the options?

Many will continue to use their Coco's until the keys are worn through to the circuit board. If a machine does all that you want it to do and does it in a proficient manner there is no need to change.

Others will decide to change to a machine which has much more backup from the various companies.

To stay in the same sort of price range this means going to an Amiga or Atari. The most common alternative would be to change to a MS-DOS IBM compatible system

Going the MS-DOS way means learning a new operating system. Contrary to popular belief, MS-DOS takes a bit of learning. Nearly as much as OS9 but without half of the features provided by OS9.

The other two aforementioned machines would probably have a similar lifespan as the Coco and therefore could mean having to move to another system sometime in the future.

Machines like the MM1, if it ever gets released over here, will be a possible move for those who would like to carry on with OS9 as their main operating system.

As I have mentioned before there are many who are more than satisfied with what the Coco does for them and have no need to contemplate moving to a new machine.

Then there are the people whose main interest lies in games and home entertainment. These people will eventually move on to some other machine.

Finally there are the people who think that the Coco has reached the end of its useful life and will be looking for something new to hold their interest.

We all need a change at sometime and it was always too much to expect that the Coco could go on forever....or was it? There are many readers out there who would like to find something new to do to give them that lift similar to the one they got when they first got the Coco. One way to do that is to get into OS9. Believe me it is a challenge that will keep your brain occupied for quite some time but the goal at the end will be worth it.

As most of the above options seem to suggest that you should discard the Coco for some other system, It may seem that I, who have used and supported the Coco in all its various guises for about 10 years, am now trying to get people to desert the cause. This is not the case. To move forward one must appraise the situation honestly. This I have always tried to do by studying the options open to me. The main issue has always been that you should enjoy your hobby to the fullest. If that means leaving Coco behind then so be it.

As the companys' we work for get more and more into modern technology we find that many more of us are working with computers to earn our daily bread. For instance, the oil refinery I work in has upgraded operations over the past couple of years. Now the whole process is computer driven. This has meant that I now spend most of my working day in front of a VDU. I am sure that many more of you, from typists to accountants to process workers are in the same boat.

This, I am sure, is leading to more and more computer hobbyists switching off. I know I am more reluctant to sit down in front of the screen than I was two or maybe three years ago. Is this feeling widespread?

Computing as a hobby has given me a lot of satisfaction over the past nine or ten years. It has given me mental stimulation, fun and been the backbone of this magazine. I would hate to lose all this through feeling that too much of my life is being spent in front of a computer screen.

Probably the only way to prevent this feeling is to look to new horizons. If one finds a new purpose then the fun will come back of its own volition.

I set my new horizons to learning OS9 and then to teaching myself to programme in "C". This has already put new life into my computing and I once again find that I do not have enough time to try all the things I want to. It's quite a while since I felt this way.

The moral of this story is:

If you find that computing as a hobby is becoming stale, try something entirely new. If you haven't already done it, get a copy of OS9 and give it a go. From there you can move on to various new languages. eg Basic09, C, Forth, Pascal etc.

Try it!

****

COCO-LINK IN THE FUTURE

The major issues which will affect this magazine will surface if the number of subscribers drops to a level where the magazine could no longer be sustained.

Another problem COCO-LINK could face in the future is the lack of material to put into the magazine. Therefor it is imperative that we get your continued support in the way of articles and programmes to maintain COCO-LINK as a source of interest, education and enjoyment.

****

THE RUMOUR REPORT

Here is the good news! Lonnie Falk of the RAINBOW magazine has stated in his editorial of December 1990 that the RAINBOW magazine will not be joining with any other magazine and will remain a completely separate entity as it has done for the last 10 years.

I think we should all congratulate and thank them for the service they have provided to the Coco community over this time.

****

CHEAP WAY TO CD RESULTS.

I extracted the following piece of information from a UK computer magazine.

The development of optical data machines using CD is undoubtedly the next big thing. Compact disks are virtually indestructable and no matter what record companies and software vendors say, they can be turned out very cheaply. As with any other computer medium it is the data held which is of value, not the the medium that holds it. This does not prevent greedy companies market- ing these CD disks at premium prices with some business software into the 4 figure region.

One of the main problems in reading the pits and blips on CD disks is the fact that a laser beam can generate unwanted refractions and stray beamlets by bouncing off the machine-cut inner rim and bevelled outer rim areas.

It can cost big bucks to correct these errors but there is a simple and cheap way to do it. First buy one of those green marker pens which are used with wipe-off memo boards. They cost about $1.50. Leaf green is the best colour. Now carefully ink the inside lip of the optical disk and the first 4mm of the outer edge and rim. Thats all that is required.

The same trick can be used on music CD's, improving audio quality by up to 20% (so I'm told).

****

SECURITY TECHNOLOGY

A Pommie computer company has released a security system called Eye Gard. Disguised as a PC monitor, it detects movement in the room and then flashes a large on screen eye at the intruder. It then delivers a warning message while automatically calling up the forces of law and order.

This idea is not very original. A friend of mine has a security system at home which also flashes a wicked eye while giving a vocal warning. This device does not need to call up the police, it deals with the situation in it's own unique way.

My friend's system has only one fault....It requires to be taken for walkies each night regardless of the weather.

****

A CLUB REBORN

The Color Computer section of the Adelaide Micro User Group died about two years ago. Due to the enthusiasm of some Coco enthusiasts the section has restarted. It is good to see that people are getting behind the Coco and forming into groups for self help.

AMUG is one of the oldest Computer clubs in Australia. It was started originally to help the old TRS-80 crowd. It still does that as well as covering all other aspects of computing.

THe vast store of knowledge to be found amongst the many members of this club can only be of great help to new and old Cocoists alike.

Further details can be found on the NOTICEBOARD page.

****

THIS ISSUE

Our survey showed that there are many people out there who wish to improve their BASIC skills so therefor the BETTER BASIC column has returned after a rest.

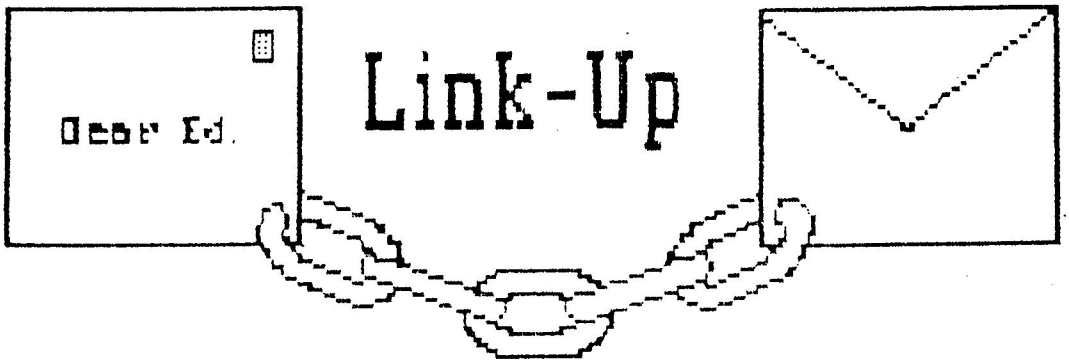
Due to the pressures of trying to get this magazine out in time during the holiday season it is possible that a few editorial errors may have crept in. If so I hope the "good will" of the season will extend to a harried COCO-LINK editor.

Due to this same pressure the "OS9 Diary" will be on holiday but will resume as soon as possible with more earth shattering insights into the OS9 learning process of yours truly.

****

All the best for 1991

Robbie



Dear Ed,
Congratulations! on a great mag. I have only been a subscriber for around 8 months, but I've got something from each and every mag. received. Everything from handy tips to cheap hardware (from the ads).
Along with a lot of your subscribers, I have turned to COCO-LINK for support, as Tandy and other mags. have let us down.
I have a Coco 3(512K) and 2 X 40TDS drives and I'm just starting out on the OS9 journey (nightmare!!!), hence I've followed your 'OS-9 Diary' with great interest. I have a problem for you, in the June90 COCO-LINK, page 26, you refer to an article by Ken Wagnitz on 'Customising your OS9 System Disk' in April89 Vol2 No2. Well, my problem is that I don't have that issue. Could you send me a copy of that issue (cheque enclosed), and maybe reprint it in a future classic, so that others who may be in the same boat as I, can use their drives to the max.
Just as a matter of interest, there are two (that I know of) hackers up here in Tamworth still practicing the ancient art of Cocoing! And loving it.

John McGrath Tamworth NSW

Dear John,
Thanks for the praise. We love it!
I have taken up your suggestion to repeat Ken's article on Customising the OS9 disk. I have included it in this month's issue. It is a very useful article and made my system so much better to use.
The other two Cocoists in Tamworth are not subscribers to COCO-LINK.....See what you can do about it.

****

Dear Ed,
Since my article in CoCo-Link (October, p. 22), I have pursued my relentless quest for the perfect RND statement.

In line 30 of RND3, you could also use PRINTINT(RND(0)*44)+1. Both RND(0) and RND(-TIMER) will produce a value more than 0 and less than 1. Try this shortie and I think you will agree that both function adequately:

```
10 CLS
20 FORX=1TO15
30 PRINTINT(RND(-TIMER)*44)+1,INT(RND(0)*44)+1
40 NEXT
50 PRINT@484,"PRESS ANY KEY";:EXEC44539:GOTO10
```

Consider this statement from a program in Rainbow (Sept. 90):

```
Y=RND(RND(RND(RND(RND(X)))))*RND(X)
```

How random can you get?

If you have any ideas on how to get a better spread of random values, let's hear from you via CoCo-Link.

Keiran Kenny, Cremorne NSW

****

Dear Ed,
On the day of our first big snow storm of the season, I had to use my snowblower to clear the way for the mailman. He handed me, to my surprise, an envelope which read CHRISTIES BEACH S.A. AUSTRALIA. What a time to remind me that the closest beach to me is way down in the Caribbean!
I read with great interest your COCO-LINK games issue. I was especially impressed by the great efficiancy of all the games. They were fast, to the point, errors free and very easy to play even if Ayrton's Puzzle is too tough for me.
I enjoyed very much the animation of Eagle Poker. This brings the idea that you should organise a contest to find the best programmer that could animate the very excellent PATIENCE game. I am sure that Johanna's Kaliedoscope versatility would be a good candidate.
Une Bonne Heureuse, SAINTE et SUCCESSFUL year to you and all my new friends down under with their everlasting summer.

Armand Belanger Quebec Canada

I hate to mention this, but it was 40 degrees centigrade when I opened your letter. I was reclining next to a clear blue swimming pool and thoroughly enjoying a ice cold beer.

****

END

Customising your OS9 System Disk.

By

Ken Wagnitz

This is an article I wrote back in August 87 for the AMUG magazine. It also appeared in Australian Rainbow. Robbie suggested it might be appropriate here, with a little editing. In fact I have tried to make it more informative and simpler (believe it or not!) I deal with customising an OS9 Level 2 disk, but the principles, if not the detail, apply to Level 1 also. And I assume you have two 40track, double sided disk drives.

Here are some things you may like to do with your OS9 System disk. The OS9 disks as supplied, do NOT suit your hardware, if for no other reason than the 60Hz vertical frame frequency going to your TV or monitor, and the associated 60Hz clock, which runs slow on our 50Hz mains.

But first, write protect both supplied OS9 disks then make copies of them and put the originals away in a safe place. You can BACKUP the disks as if they were normal RSDOS 35 track disks, ie using DSKINI to format disks, then BACKUP.

THE BOOT FILE

The BOOT file

When you run the DOS command from Disk Basic to start OS9, it loads and runs a boot file loader on track 34, which in turn loads the boot file called 'OS9Boot' on the OS9 disk. In that boot file are a number of modules which OS9 uses, none of which are commands. It is the contents of that file which need to be changed to suit your hardware. And its the changing of those modules that I deal with here.

1. THE CLOCK:

The clock module is responsible for maintaining the current time, being incremented by the an interrupt from the vertical sync pulse, which in Oz is 50Hz, but 60Hz in the USA. Consequently when you boot the bought OS9 or DESKMATE, the screen may roll, and the clock will lose time. (It loses time with disk accesses also but that's another story). If you have a Philips colour TV you will not be able to correct the rolling if it occurs. Other people can reach for the Vertical Hold. It is important to have the time roughly correct because it date/time stamps files so that a DIR E shows the date/time of last modification of a file.

Fortunately a 50Hz clock module is supplied, along with other modules, on the Boot/Config disk. To use it, run the CONFIG command, as follows:

Assuming two disk drives-

Boot with COPY copy of the system disk in d0. Format a disk in d1, leave it there.

Place your COPY of the CONFIG disk in d0.

Type 'chx /d0/cmds ; chd /d0'. (The ';' is a command separator.)

Type 'config' and follow the prompts as per the 2-drive description in the manual (OS9 Commands, page 6-18).

- Include all of the window descriptors (w.dw - w7.dw), and d0_35s.dd, d1_35s.dd, ddd0_35s.dd drive descriptors, and every other option, and TERM_WIN type terminal. If you have three disk drives, include d2_40d.dd.

Config will write a bootfile on the blank disk in d1 then ask what commands you want on that disk.

Type 'i' to add a couple of command files. Select 'grfdrv' and 'shell' only.

The disk in d1 is now in the same format as the supplied OS9 system disk, ie 35track, single sided, but the clock module in the boot file is 50Hz, and the disk drive descriptors are for double sided drives (assuming that is what you have).

Now replace the disk in d0 with the newly created one from d1, and press the RESET button to boot from the new disk. You should have a 50Hz clock. Then put the system disk back into d0 and

type 'chx /d0/cmds ; chd /d0'.

Type 'setime' and enter the time and date.

2. PRINTER CHARACTERISTICS:

Type 'xmode /p' to see the current printer settings. Change any of these as per the xmode command description (OS9 Commands, page 6-100). In particular, the baud rate will probably need changing.

3. DISK DRIVE CHARACTERISTICS:

When a drive is used, its descriptor is read to see what the drive's characteristics are. So somewhere in the module 'd0' (which is in memory), is a byte which determines what stepping rate should be used for drive 0.

We can modify the descriptors to reflect what the drives can do. (Use the command MDIR to see what modules are in memory.)

Type 'help modpatch' to see the usage of a command which was for some reason not documented in the Commands descriptions. It is used to patch modules in memory.
Now turn to page 5-9 of the Technical Reference section. There you will find a description of the relevant parts of a disk drive descriptor module, referenced by their offset from the start of the module (ie byte address, relative to the 1st byte of the module at address 0). Note that the offsets (Relative Address), and all numbers that follow, are in hexadecimal.

Type 'modpatch'. No prompt is issued. Then type the following, not including the * or past it (comments).

```
I d0        * 'current address' is now the 1st byte of d0.

c 14 0 3    * changes step rate from 0 (30ms) to 3 (6ms).
            * use 1 for 20ms, 2 for 12ms, or leave this
            * line out for 35ms.
c  18  23  28 * changes the number of tracks from 35 to 40
* -use 50 if an 80 track drive, or leave out
            * if correct already.
c 19 1 2    * changes the number of sides from 1 to 2
            * -leave out if drive is single sided,
            * or correct already.
v           * fixes the module crc which is incorrect after
            * changing bytes.

I d1
            * repeat the above 4 lines (after 'I d0') for
            * the 2nd drive (d1), putting in the new bytes
            * applicable.
```

Then repeat for d2, making it the same as d0. Patch d2 also if you have 3 drives.

4. THE TERMINAL SCREEN:
++++++++++++++++++++++++

When you booted, you got a 32 column by 16 line screen. Disappointed? That can be changed, but at the expense of memory. It will cost an extra 16K of RAM to have a 40 or 80 by 24 screen. The device descriptor for the terminal has to be patched ('Term').

Turn to Technical Reference section, pages 6-7,6-8. That is the detail of a device descriptor such as 'Term' which determines the features of the Terminal screen. Now let's customise it!

You should still have 'modpatch' active.

```
I term      * this now points to the descriptor we want to
            * change.
c 2c 28 50  * change number of columns to 80 (use 28 for
            * 40 columns).
c 30 1 2    * change window type to 80 column text
            * (see WCREATEcommand).
* if you wish, the colours can be changed also -here I
choose white on black.
c 33 2 0    * choose white foreground
c 34 3 2    * choose black background
c 35 3 2    * choose black border
v           * fix up the crc.
```

Now press CTRL BREAK (Escape) to end the modpatch session.

Place the new boot disk you created, in d1.
Type 'cobbler /d1'. The current modules will used to replace the bootfile on that disk.
Put that disk into d0 and press RESET. It should boot giving you a nice 80 or 40 column screen. You will have heard disk accesses speed up (if you put in faster stepping rates) after the bootfile loaded, as the shell file and Grfdrv load.
If it didn't work as you expected, start all over again!
If it did work OK, place your system disk copy in d0, and type
'chx /d0:cmds ; chd /d0' then type
'cobbler /d0'.
You now have a customised system disk!

Of course the system disk is not a 40track double-sided. To get that, format a disk (which will be 40TDS), then DSAVE all of the system disk onto it.

MEMORY:
+++++++

Booting on a 128K Coco, leaves 56K of RAM free. The above changes to the initial Terminal display reduce that to 48K. Opening more windows will reduce that more. The answer is to purchase a 512K RAM board. Read the magazines to find out where to get one, or contact me. Contrary to what a Coco magazine suggested, changing the makeup of the file 'shell', which actually contains a lot of utilities, will not increase available memory. As long as a SHELL (memory module) needs to be loaded, it may as well be merged with just less than 8K of utilities, since any file loaded on its own will occupy 8K of RAM space!

DISK SPACE:
+++++++++++

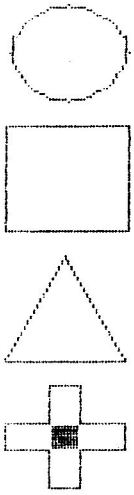
Even with my two 80track drives and a 40track drive connected, I would like more space on my system disk. I like to have all my commands on it, as well as useful procedure files (in a directory called PR), DEFS files, C library files, and SYS files.

Some of the commands in the CMDS directory are not needed, since they are merged in the file called 'shell', (which also happens to contain the module called 'shell').
Do an MDIR to find out what commands you have in memory (which loaded with 'shell'), and delete those commands from the CMDS directory on your system disk.

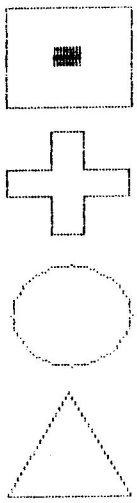
Also, disk space can be saved by MERGEing files. This has

Continued on page 15

SMILER



Shapes



By Keiran Kenny

GROANER

This program is intended as a computer activity for children in the pre-literacy or early literacy stage. Some initial guidance to help the child understand the prompts may be necessary

The graphic screendisplays a column of four easily distinguishable shapes left, and the same shapes in a different order right. All graphic figures are drawn and "got" behind the scene in lines 830 to 890.

A black rectangle appears in one of the shapes left, and a flashing cursor in the top shape right. Press the up/down arrows to move the cursor into the shape that matches the chosen shape and then press the spacebar. A correct choice earns a "smiler" and a musical tinkle, and an incorrect choice earns a "groaner" with appropriate sound.
After each ten tries the score is shown and the child is given an option to play again or end the program.

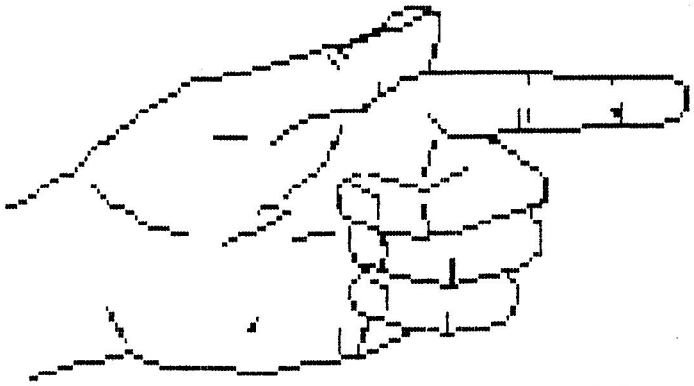
The character set, lines 170 to 740, contains strings for all capitals and numbers and the most-used math and punctuation signs. Each string is labelled Z$ plus its ASCII (CHR$) number in brackets. If you want to use it in your own programs, set the horizontal coordinate, B, the vertical, C, and put your text into a string labelled ZL$ followed by GOSUB100 (as in line 1260). It will give you 27 characters per full screen line with wraparound if your text is longer than 27 characters. The value LL=27 is established in line 750. This can be varied if you want shorter lines.

Line 30 establishes the values for the hi-speed and slow-down pokes according to whether you are using a CoCo 2 or CoCo 3. The program is in hi-speed mode until you press N at the "Play again?" prompt.

Continued overleaf

```
0 'SHAPES'
1 'COPYRIGHT 1990 KEIRAN KENNY
10 CLS:PRINT@232,"ONE MOMENT PLE
ASE"
20 CLEAR500
30 IFPEEK(&HFFFE)*256+PEEK(&HFFF
F)=&H8C1B THENSP=65497:SW=65496E
LSESP=65495:SW=65494
40 POKESP,0
50 DIM Z$(93),A(400),B(400),C(40
0),D(400),E(256),F(256)
60 SL=48:SU=16:GL=48:GU=96
70 GOTO170
80 C=191:ZL$="PRESS ANY KEY.":B=
INT(128-LEN(ZL$)*5):GOSUB100
90 K$=INKEY$:IFK$=""THEN90ELSECO
LOR5:LINE(58,183)-(210,191),PSET
,BF:RETURN
100 IFLEN(ZL$)<=LL THEN140
110 FORT=LL TO1STEP-1:IFMID$(ZL$
,T,1)=" "THEN130
120 NEXTT:GOTO140
130 P$=LEFT$(ZL$,T):W$=P$:GOSUB1
50:ZL$=RIGHT$(ZL$,(LEN(ZL$))-T):
C=C+15:GOTO100
140 W$=ZL$
150 DRAW"BM"+STR$(B)+","+STR$(C)

160 COLOR0:FORZ8=1TOLEN(W$):DRAW
Z$(ASC(MID$(W$,Z8,1)))+"BR4":NEX
T:RETURN
170 Z$(65)="U4E3F3D2NL6D2" 'A
180 Z$(66)="U7R5F1D1G1NL5F1D2G1N
L5BR2" 'B
190 Z$(67)="BRHU5ER4FDBD3DGNL4BR
" 'C
200 Z$(68)="U7R5FD5GNL5BR" 'D
210 Z$(69)="NR6U4NR4U3R6BD7" 'E
220 Z$(70)="U4NR4U3R6BD7"  'F
230 Z$(71)="BRHU5ER4FDBD2NL2D2GN
L4BR" 'G
240 Z$(72)="U7BR6D3NL6D4" 'H
250 Z$(73)="BR1R4L2U7NL2R2BR1BD7
" 'I
260 Z$(74)="BRHUDFR4EU6BD7" 'J
270 Z$(75)="U7D3R2NE3LF4" 'K
280 Z$(76)="NU7R6" 'L
290 Z$(77)="U7F3E3D7" 'M
300 Z$(78)="U7F6NU6D" 'N
310 Z$(79)="BRNR4HU5ER4FD5GBR"'O

320 Z$(80)="U7R5FD2GNL5BR8D3" 'P

330 Z$(81)="BRHU5ER4FD4G2L3RUERF
2" 'Q
340 Z$(82)="U7R5FD2GL5R2F3R1" 'R

350 Z$(83)="BRNHR4EU2HL4HUER4FBD
6" 'S
360 Z$(84)="BR4U7NL4R4BD7" 'T
370 Z$(85)="BRHU6BR6D6GNL4BR" 'U

380 Z$(86)="BU7D4F3E3U4BD7" 'V
390 Z$(87)="NU7E3F3NU7" 'W
400 Z$(88)="UE5UBL5DF5D" 'X
410 Z$(89)="BR3U2H3U2BR6D2NG3BD5
" 'Y
420 Z$(90)="BU7R5DG5DR5" 'Z
430 Z$(48)="BRHNE3U5ER4FNG3D5GNL
4BR" '0
440 Z$(49)="BR2R4L2U7NG2BR2BD7"
'1
450 Z$(50)="BU6ER4FD2GL4GD2R6" '
2
460 Z$(51)="BU6ER4FDGNL3FD2GL4NH
BR5" '3
470 Z$(52)="BR6U7L2G4R6D3" '4
480 Z$(53)="BRNHR4EU2HL5U3R6BD7"
'5
490 Z$(54)="BRHU3NE3D2ER4FDGNL4B
R" '6
500 Z$(55)="BU6UR6G6DBR6" '7
510 Z$(56)="BRHU2EHUER4FDGNL4FD2
GNL4BR" '8
520 Z$(57)="BRNHR4EU5HL4GD2FR4EB
D4" '9
530 Z$(32)="BR" 'SPACEBAR
540 Z$(33)="BR2UBU2U4BR4BD7" '!
550 Z$(36)="BUR3DNU6UR2EHL4HER5B
D5" '$
560 Z$(37)="BU5UERFDGLHBU2BR6DG5
DBR4REUHLGDFBR2" '%
570 Z$(39)="BRBU5NU2BD5BR" '
580 Z$(40)="BR2H2U3E2BD7" '(
590 Z$(41)="E2U3H2BD7BR2" ')
600 Z$(42)="BR3BU7D3NE2NR3NF2ND3
NG2NL3NH2BD4BR3" '*
610 Z$(43)="BR3BUU4D2NL2R2BRBD3"
'+
620 Z$(44)="NUNG" ',
630 Z$(45)="BU3BR2R2BR2BD3" '-
640 Z$(46)="BR2NU1BR4"  '.
650 Z$(47)="UE5UBD7" '/
660 Z$(58)="BR3BUUBU3UBR3BD6" ':

670 Z$(59)="NGUBU4NUBD5" ';
680 Z$(60)="BR3H3E3BD6" '<
690 Z$(61)="BU2BR2R2BU2NL2BR2BD4
" '=
700 Z$(62)="BU6F3G3BR3" '>
710 Z$(63)="BR2UBU2UREUHL2GDBR6B
D5" '?
720 Z$(91)="NR2U7R2BD7" '[
730 Z$(92)="BU7DF5D" '\
740 Z$(93)="R2U7NL2D7" ']
750 LL=27:GOTO820
760 PUT(X,Y)-(X+39,Y+39),A:RETUR
N
770 PUT(X,Y)-(X+39,Y+39),B:RETUR
N
780 PUT(X,Y)-(X+39,Y+39),C:RETUR
N
790 PUT(X,Y)-(X+39,Y+39),D:RETUR
N
800 PUT(SL,SU)-(SL+31,SU+31),E:R
ETURN
810 PUT(GL,GU)-(GL+31,GU+31),F:R
ETURN
820 PMODE4,1:COLOR0,1:PCLS
830 CIRCLE(20,20),19:GET(0,0)-(3
9,39),A
840 LINE(0,48)-(39,87),PSET,B:GE
T(0,48)-(39,87),B
850 DRAW"BM0,135M+19,-39M+19,+39
M-39,+0":GET(0,96)-(39,135),C
860 DRAW"BM14,144R10D14R14D10L14
D14L10U14L14U10R14U14":GET(0,144
)-(39,183),D
870 CIRCLE(63,15),8,,1.5:DRAW"BM
58,11R4BR2R4BL5BDBL3FGHEBR6FGHEB
L3D5BUFGHE":CIRCLE(63,18),4,,1,0
,.5
880 CIRCLE(95,15),8,,1.5:DRAW"BM
90,11R4BR2R4BL5BDBL3FGHEBR6FGHEB
L3D5BUFGHE":CIRCLE(95,22),3,,1,.
5,1
890 GET(48,0)-(79,31),E:GET(80,0
)-(111,31),F
900 PCLS:SCREEN1,1
910 X=0:Y=0:FORXS=1TO4:ONXS GOSU
B760,770,780,790:Y=Y+48:NEXT
920 X=RND(-TIMER)
930 X=216:Y=0
940 FORE=1TO4:T(E)=E:NEXT
950 FORZ=1TO4
960 R=RND(4)
970 IFR=1 ANDZ=1ORT(R)=0THEN960
980 ONR GOSUB760,770,780,790
990 Y=Y+48
1000 N(Z)=15+48*(R-1)
1010 T(R)=0
1020 NEXT
1030 B=100:C=10:ZL$="SMILER":GOS
UB100
1040 B=93:C=88:ZL$="GROANER":GOS
UB100
1050 P=15:S=RND(4)-1:Q=15+48*S
1060 LINE(P,Q)-(P+8,Q+8),PSET,BF

1070 H=235:V=15
1080 IFPEEK(341)=247THENV=V-48:N
R=NR-1
1090 IFPEEK(342)=247THENV=V+48:N
R=NR+1
```

Continued on page 10



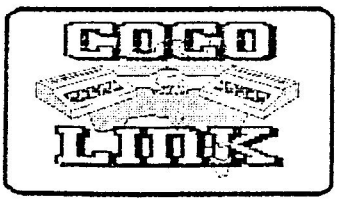
Better BASIC Part 13

Bu Richard Schmidt

Array Sorts

The following are listings of three different sorting methods, and the number of comparisons & exchanges each must make. These listings have been converted over from Pascal, so they may seem to use unnecessarily long variable names, but as a general rule, the longer the variable name, the more readable the program. (Although Microsoft BASIC doesn't encourage this by only reading the first two characters in a variable name).

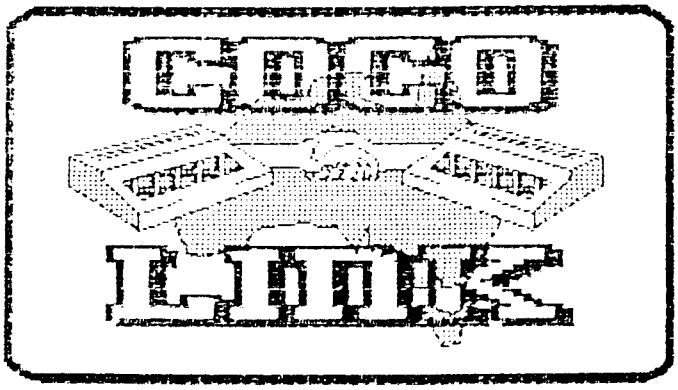
```
 10 REM PROG TO RUN SUBROUTINES.
 20 REM ** SET UP ARRAY **
 30 DIM ARRAY(10)
 40 LASTELEMENT = 5
 50 FOR ELEMENT = 1TO LASTELEMENT
 60 ARRRAY(ELEMENT) = LASTELEMENT
  - (ELEMENT - 1)
 70 PRINT ARRAY(ELEMENT);
 80 NEXT ELEMENT
 90 PRINT "ARRAY IS UNSORTED."
 100 GOSUB 1000' SORT ARRAY
 110 REM ** PRINT OUT ARRAY **
 120 FOR ELEMENT= 1TO LASTELEMENT
 130 PRINT ARRAY(ELEMENT);
 140 NEXT ELEMENT
 150 PRINT "ARRAY IS SORTED."
 160 PRINT "NUMBER OF COMPARISONS
  =";COMPARISON
 170 PRINT"NUMBER OF EXCHANGES ="
 ;E1XCHANGE
 180 END
```




```
1000 REM ** START OF BUBBLE SORT
  **
1010 PASS = 1:COMPARISON = 0:E1X
CHANGE = 0
1020 NOEXCHANGES$ = "TRUE"
1030 FOR ELEMENT = 1 TO (LASTELE
MENT - PASS)
1040 COMPARISON = COMPARISON + 1
1045 FOR X=1 TO LASTELEMENT:PRIN
TARRAY(X);:NEXT:PRINT" PASS NO."
;PASS
1050 IF ARRAY(ELEMENT) <= ARRAY(
ELEMENT + 1) THEN GOTO 1130
1060 REM ** SWITCH NEIGHBOURING
ELEMENTS **
1070 E1XCHANGE = E1XCHANGE + 1
1080 TEMP = ARRAY(ELEMENT)
1090 ARRAY(ELEMENT) = ARRAY(ELEM
ENT + 1)
1100 ARRAY(ELEMENT + 1) = TEMP
1110 REM ** RESET EXCHANGES FLAG
  **
1120 NOEXCHANGES$ = "FALSE"
1130 NEXT ELEMENT
1140 PASS = PASS + 1
1145 PRINT"NOEXCHANGES$ = ";NOEX
CHANGES$
1150 IF NOEXCHANGES$ = "FALSE" T
HEN GOTO 1020
1160 RETURN
```

```
1000 REM ** START OF SELECTION S
ORT **
1010 COMPARISON = 0:E1XCHANGE= 0
1020 FOR C1OUNTER = LASTELEMENT
```

```
TO 2 STEP -1
1030 INDEXOFMAX = C1OUNTER
1040 FOR C2OUNTER = (C1OUNTER -
1) TO 1 STEP -1
1050 COMPARISON = COMPARISON + 1
1060 IF ARRAY(C2OUNTER) > ARRAY(
INDEXOFMAX) THEN INDEXOFMAX = C2
OUNTER
1070 NEXT C2OUNTER
1080 IF INDEXOFMAX = C1OUNTER TH
EN GOTO 1140
1090 REM ** SWITCH ELEMENT AT CO
UNTER WITH ELEMENT AT INDEXOFMAX
 **
1100 E1XCHANGE = E1XCHANGE + 1
1110 TEMP = ARRAY(C1OUNTER)
1120 ARRAY(C1OUNTER) = ARRAY(IND
EXOFMAX)
1130 ARRAY(INDEXOFMAX) = TEMP
1140 NEXT C1OUNTER
1150 RETURN


1000 REM ** START OF INSERTION S
ORT **
1010 COMPARISON = 0:E1XCHANGE= 0
1020 FOR N1EXTPOS = 2 TO LASTELE
MENT
1030 NEXVAL = ARRAY (N1EXTPOS)
1040 REM ** SHIFT ALL VALUES > N
EXVAL DOWN ONE ELEMENT **
1050 N2EXTPOS = N1EXTPOS
1060 COMPARISON = COMPARISON + 1
1070 IF ARRAY (N2EXTPOS - 1) <=
NEXVAL THEN GOTO 1110
1080 E1XCHANGE = E1XCHANGE + 1
1090 ARRAY (N2EXTPOS) = ARRAY (N
2EXTPOS - 1)
1100 N2EXTPOS = N2EXTPOS - 1
1110 IF N2EXTPOS > 1 THEN GOTO 1
060
1120 ARRAY (N2EXTPOS) = NEXVAL
1130 NEXT N1EXTPOS
1140 RETURN
```



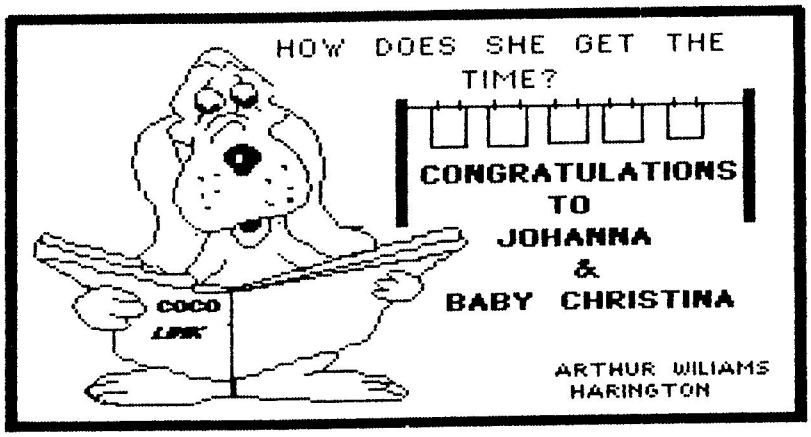
The "Bubble Sort" method works by comparing neighboring elements of the array, and exchanging them if necessary. The "Insertion Sort" method works by comparing the next element in the array with an already sorted part of the array (initially, the first element), and inserting it into the appropiate place.
The "Selection Sort" method works by finding the largest element in the array, and then places it at the end of the already sorted part of the array (initially at the beginning of the array).
As you can see, each of these sorting methods take different times to process the array (due to the different number (and complexity) of comparisons and exchanges), and so you may choose one of them over the other. I have coded these for a numeric array (a list of numbers), but they will work equally well for string arrays (lists of alphabetic characters).

END

Continued from page 8

```
1100 IFNR<1THENNR=1
1110 IFNR>4THENNR=4
1120 IFV<20THENV=20
1130 IFV>164THENV=164
1140 LINE(H-4,V-4)-(H+4,V+4),PSE
T,BF:FORD=1TO200:NEXT
1150 IFINKEY$=CHR$(32)THEN1180
1160 LINE(H-4,V-4)-(H+4,V+4),PRE
SET,BF
1170 GOTO1080
1180 IFN(NR)=Q THENRT=RT+1:FORSN
=128TO158STEP10:SOUNDSN,1:NEXT:G
OSUB800ELSE1210
1190 SL=SL+32:SL>176THENSL=48:SU
=48
1200 GOTO1230
1210 WR=WR+1:FORSN=31TO1STEP-10:
SOUNDSN,1:NEXT:GOSUB810ELSE1230
1220 GL=GL+32:IFGL>176THENGL=48:
GU=128
1230 IFRT+WR=10THEN1260
1240 GOSUB80:K$=INKEY$:NR=0
1250 GOTO910
1260 B=74:C=160:ZL$="SMILERS:"+S
TR$(RT):GOSUB100
1270 C=172:ZL$="GROANERS:"+STR$(
WR):GOSUB100
1280 B=56:C=191:
9395 "PLAY AGAIN? Y/N":GOSUB100
1290 K$=INKEY$:IFK$<>"Y"ANDK$<>"
N"THEN1290
1300 IFK$="Y"THENPCLS:NR=0:RT=0:
WR=0:SL=48:SU=16:GL=48:GU=96:GOT
0910
1310 IFK$="N"THENPOKESW,0:CLS:EN
D
```

Graphics by.....

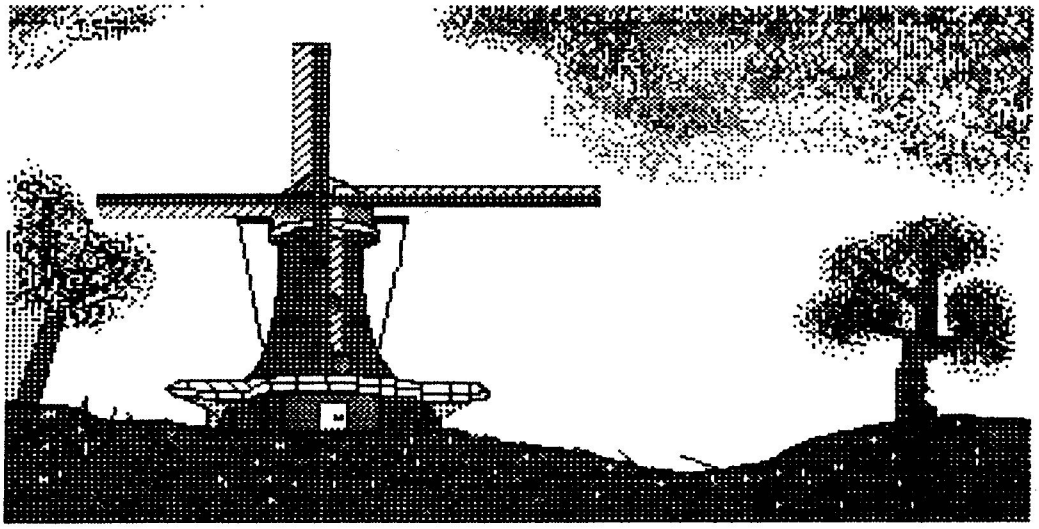


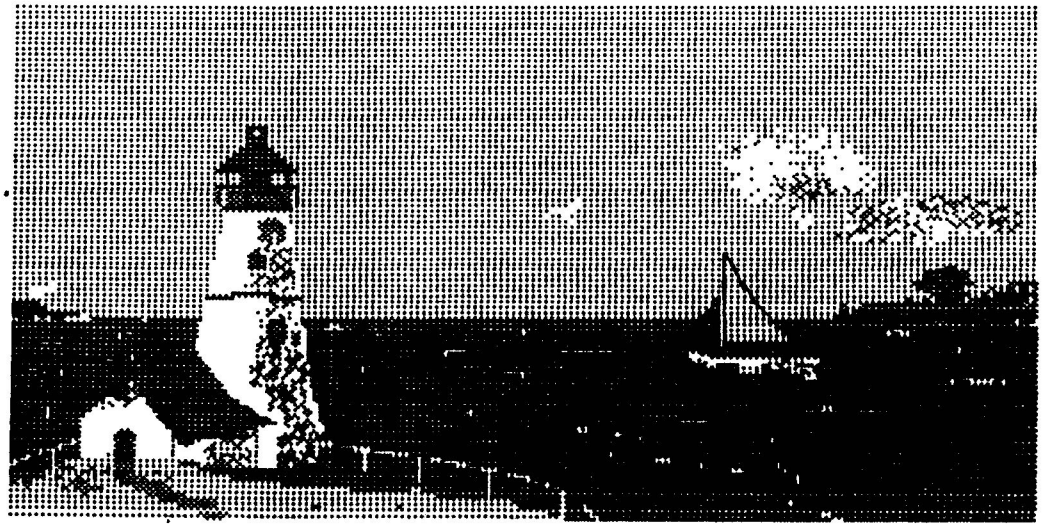
A belated card

for Johanna

Two

Scenes

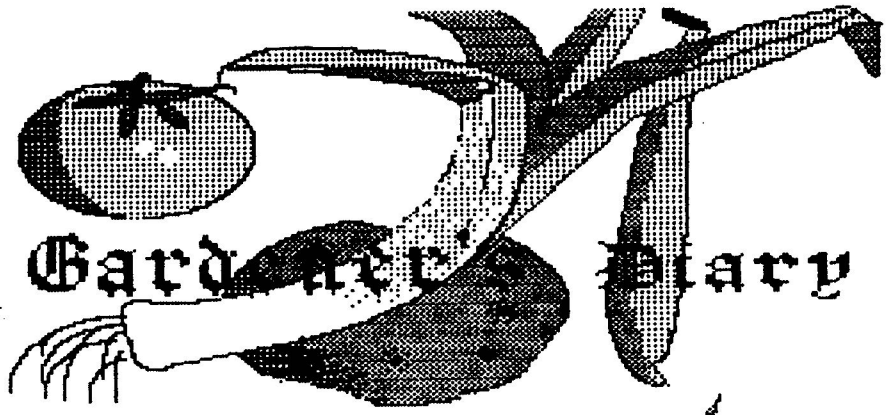




By

Arthur

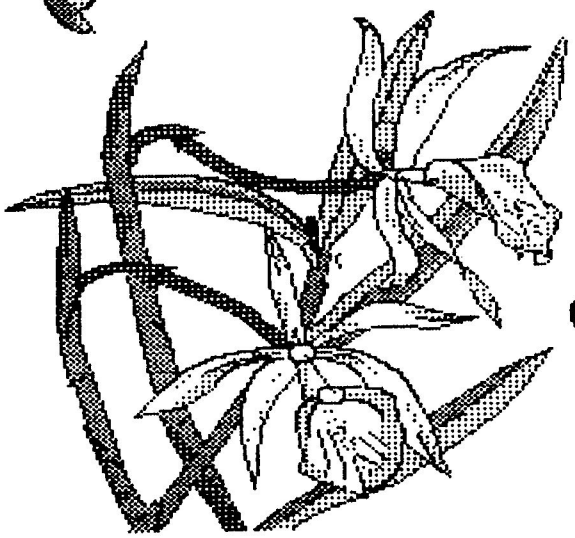
Williams



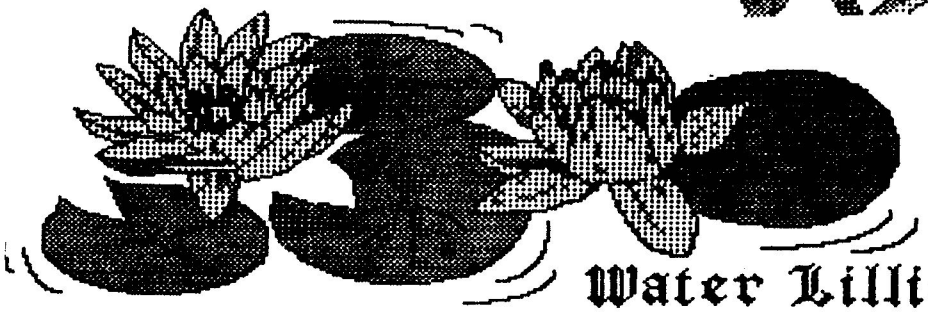
Gardener's Diary

SOME

PREVIOUS



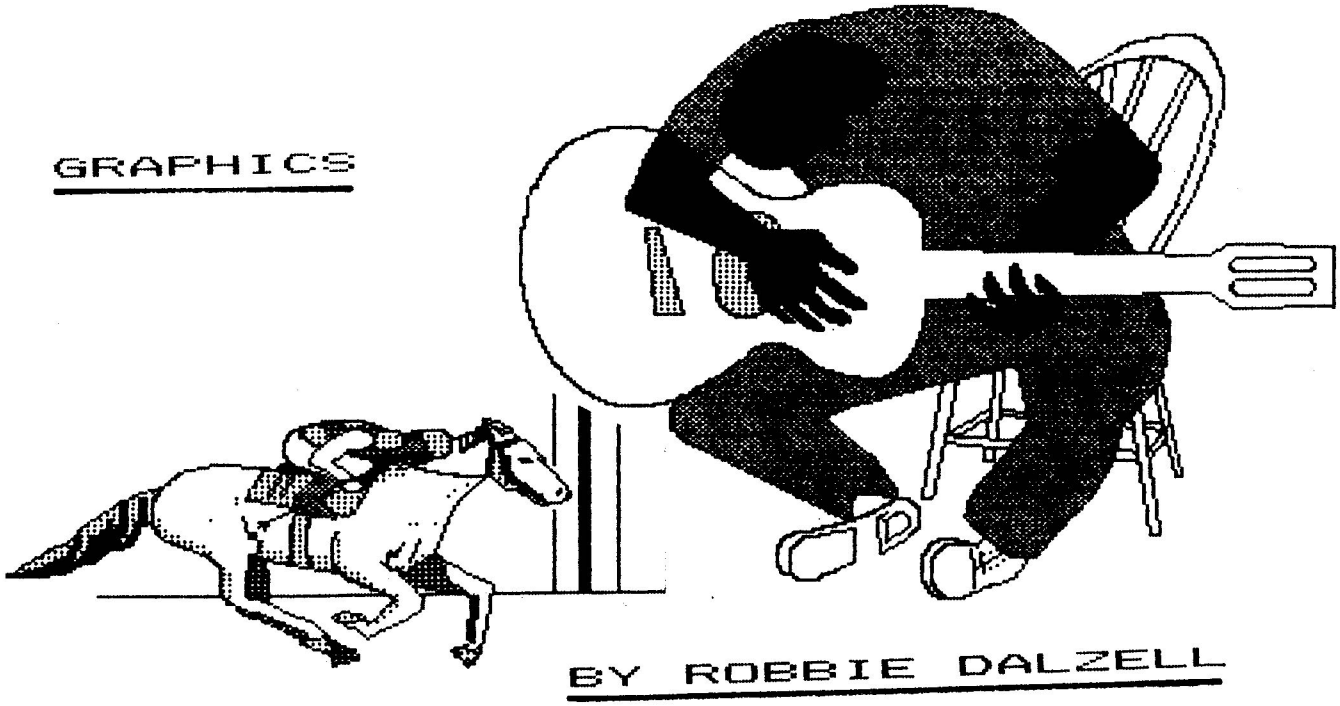
Orchids in Bloom



COCO-LINK

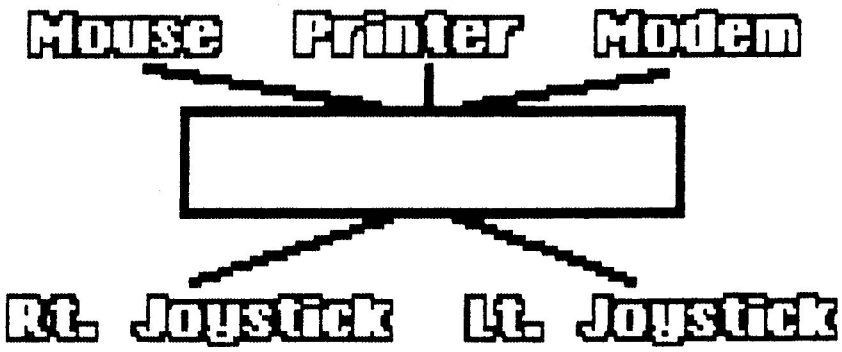
Water Lillies

GRAPHICS



BY ROBBIE DALZELL

Deluxe Switch-box



By Kevin Gowan

Are you sick of having to unplug your MOUSE and replace it with a JOYSTICK when you want to play a game, and vice-versa when you need the Mouse again. Have you ever played a game using a Joystick, only to find it is plugged into the Left-hand socket, while the game requires it in the Right-hand one (or the other way round)? Are you tired of unplugging the PRINTER to connect your MODEM to the Serial I/O port? Isn't it annoying to have to unplug your cassette and Right-hand joystick in order to use the Hi-Resolution Joystick Interface for those special programs? Well, here's the answer to all those problems in a nutshell!

Build the Deluxe Switch Box, and never have to plug-unplug again! The Switch Box is wired to the Color Computer by a 21-way Ribbon Cable (like the type that connects to the disk drives) connected to four DIN Plugs (L-H joystick, R-H joystick, Cassette and Serial I/O ports). You plug your Cassette Player into the DIN Socket at the Switch Box. The Printer also plugs into a DIN Socket provided (as well as your Modem). The two Joysticks plug into their respective DIN Sockets, and there is an extra socket for the Mouse.

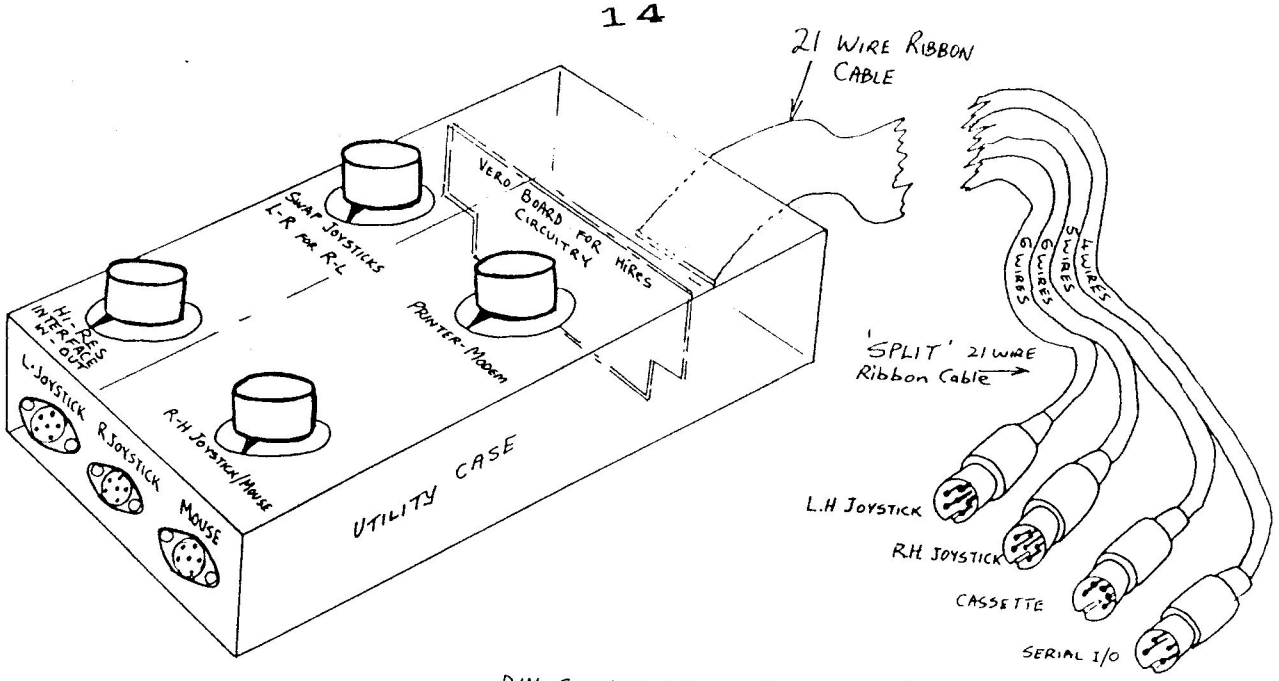
The Hi-Resolution Joystick Interface is wired into the Right-hand Joystick circuit (used by programs like MAX-10 and CoCoMax III), and is switched in-to and out-of the circuit by a two-position rotary switch. The switch has 6 poles; two of them switch the X-axis and Y-axis joystick inputs to the Operational Amplifiers (1/2 of Integrated Circuit, LM3900); another two poles switch those Op-Amp outputs towards the CoCo joystick inputs; one pole connects power (+5 Volts) to the IC (LM3900), and the other pole connects the Cassette Output lead to the buffer Op-Amp (which in turn is connected to the 'ramp' Op-Amp circuit). NOTE: When the Hi-Res Interface is switched 'IN', the Cassette circuitry has a 'hole' in in, and shouldn't be used until the Hi-Res I/F is switched 'OUT'. The circuitry for the Hi-res I/F can be built on Vero board, which can slide into the grooves inside the Utility-box. I cut out the lower corners (see diagram) to allow room for the wiring to pass from the front to the rear of the box unimpeded.

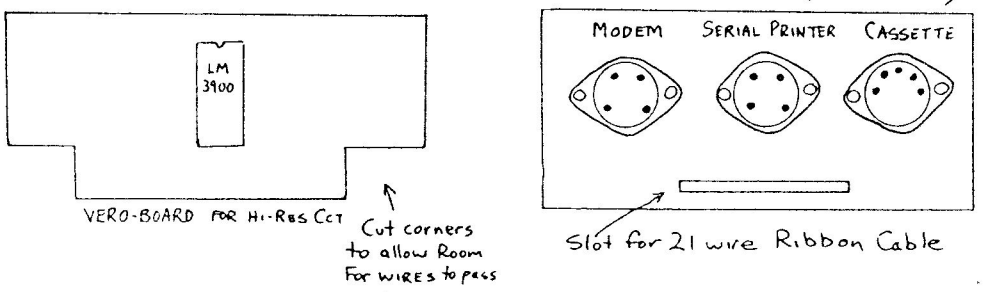
The Joystick-Mouse option rotary switch is also a 6-pole 2-position device, using 5 of the 6 poles. This allows you to alternate between a Joystick or the Mouse. Note: they are wired into the Right-hand Joystick circuit to allow the Mouse to be used with the Hi-Res Joystick Interface. The Power Rail (+5 volt - pin 5) is switched to lessen the power drain when both the R-H Joystick and Mouse are plugged in.

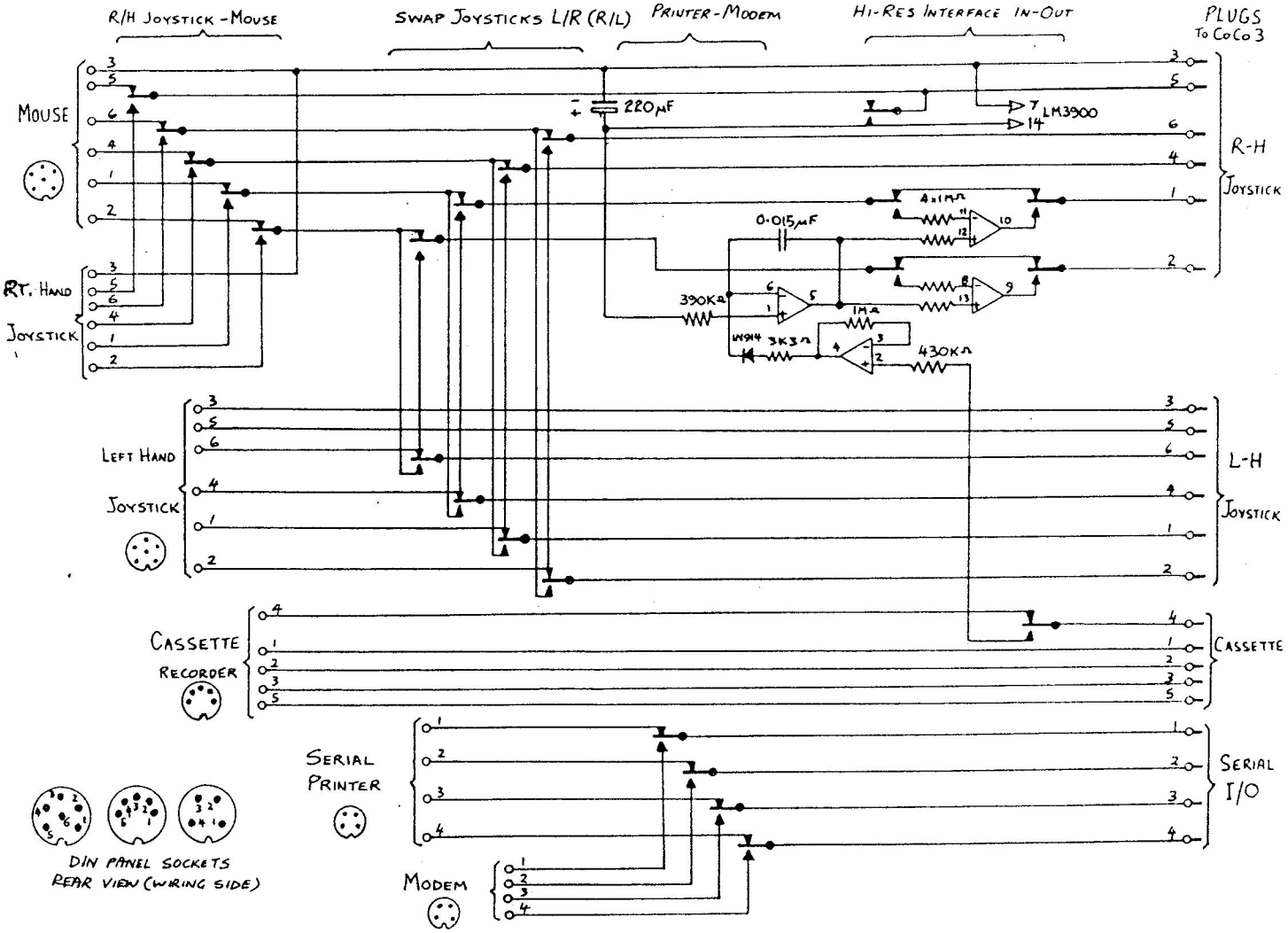
The Serial Printer - Modem rotary switch is a 6-pole 2-position switch, and only 4 poles are used, to physically switch all 4 leads used from the Serial I/O port of the CoCo.

The Joystick-Swap rotary switch is an 8-pole 2-position switch, which I had great difficulty in obtaining. I had to resort to cannibalising some old communications equipment to obtain an 'OAK' switch with two wafers, each of which contained 4-poles (2-positions). It seems that these types of switches are not readily obtainable at your corner hobby electronics store (Tandy - Dick Smith). You can pay big bucks to buy a switch kit and wafers to suit from a professional electronics store, so it may pay to scrounge around for a second hand one. The switch 'swaps' the X-axis and Y-axis potentiometer wiper connections of the Left-hand and Right-hand Joysticks. With this switch, you cam play games meant for two players (each having their own joysticks), and use the same joystick, switching it from the Left to the Right (or vice-versa) as necessary.

You may wish to use DYMO tape to label the DIN Panel


21 WIRE RIBBON CABLE
VERO BOARD FOR HIRES CIRCUITRY
SWAP JOYSTICKS L-R FOR R-L
HI-RES INTERFACE OUT
PRINTER-MODEM
L. JOYSTICK
R JOYSTICK
MOUSE
R-H JOYSTICK/MOUSE
UTILITY CASE
'SPLIT' 21 WIRE Ribbon Cable
6 WIRES
5 WIRES
6 WIRES
4 WIRES
L.H JOYSTICK
R.H JOYSTICK
CASSETTE
SERIAL I/O


LM 3900
VERO-BOARD FOR HI-RES CCT
Cut corners to allow Room For WIRES to pass
DIN SOCKET LAYOUT (OTHER END)
MODEM
SERIAL PRINTER
CASSETTE
Slot for 21 wire Ribbon Cable


R/H JOYSTICK - MOUSE
SWAP JOYSTICKS L/R (R/L)
PRINTER-MODEM
HI-RES INTERFACE IN-OUT
PLUGS To CoCo 3
MOUSE
RT. HAND JOYSTICK
220µF
LM3900
0.015µF
4x1MΩ
390KΩ
1MΩ
3K3Ω
430KΩ
R-H JOYSTICK
LEFT HAND JOYSTICK
L-H JOYSTICK
CASSETTE RECORDER
CASSETTE
SERIAL PRINTER
SERIAL I/O
MODEM
DIN PANEL SOCKETS REAR VIEW (WIRING SIDE)

Sockets and Rotary Switches, or use stick on labels.

Here is a parts list for those of you wanting to make this project:-

Description	Quantity
Panel Socket (6 pin DIN)	3
Utility box (Zippy Box 60x113x196mm)	1
Panel Socket (4 pin DIN)	2
Rotary Switch 6-pole 2-position	3
Panel Socket (5 pin DIN) 180 degrees	1
Rotary Switch 8-pole 2-position	1
Plug (4 pin DIN)	1
Knobs (for rotary switches)	1
Plug (5 pin DIN) 180 degrees	1
I.C. LM3900 Quad Op-Amp	1
Plug (6 pin DIN)	2
14 Pin IC socket	1

Resistors (5 @ 1 Meg, 1 @ 430 Kohm, 1 @ 390 Kohm).
Diode 1 @ 1N914 Silicon.
Capacitors (1 @ 0.015 uF 50 Volt Greencap
1 @ 220uF 10 Volt Electrolytic)
Misc: 1.5 Metres 34-way Ribbon Cable - Hookup wire - 12 Bolts & nuts (for panel sockets)
Vero Board - Solder - Labels.

END

Continued from page 6

a number of advantages:
-it reduces disk space by reducing the number of files on the disk (each file has an overhead, no matter how small);
-it reduces disk space by packing the files more efficiently (disk space is allocated in 256byte sectors);
-and for modules which are loaded into memory, each file uses a multiple of 8k bytes of memory, so merge small files into just under 8k byte (less 256 bytes I think) files (or multiples thereof).

On a two drive (or more) setup, use the second drive to hold data files or program source files -eg your Basic09 programs which are under development. When a Basic09 program is working OK, then PACK it into the CMDS directory (if there is room).

END

Housie

By John Mc Grath

Housie, Bingo or Lotto. These names will be familiar to many people. In its many guises the game has been popular since well before WW2. It was a very popular parlour game after the war and was played by many families in the time before TV changed the way of our home lives. Lotto, as it was then known was to grow into the vastly popular Bingo of today.

Versions of this game can still be bought for home entertainment but instead of picking the numbers out of a bag, this little programme will select them for you electronically.

OK. Eyes Down!.....All the ones.....Legs Eleven!!!

```
10 DIMA(90)
11 X=RND(-TIMER)
20 ON WAIT GOTO 140
25 FORX=1TO90
30 A(X)=X:NEXTX
40 CLS3
50 Y=0:B=0
60 N=RND(90)
70 X=N
80 IFN<>A(X)THENGOTO60
85 A(X)=0
90 PRINT@Y,N;
91 '?#-2,N;
92 B=B+1
95 Y=Y+3:IFY=30 OR Y=31 OR Y=62
OR Y=94 OR Y=126 OR Y=158 OR Y=1
90 OR Y=222 OR Y=254 OR Y=282 TH
EN Y=Y+2
96 PRINT@352,"PRESS BREAK AT ANY
 TIME FOR NEW GAME";
97 PRINT@290,B" NUMBERS";
98 IF B=90 THEN GOTO130
99 'GOTO60
100 PRINT@480,"ANY KEY FOR NEXT
NUMBER";
110 EXEC44539
120 GOTO60
130 PRINT@480,"ANY KEY FOR ANOTH
ER GAME";:EXEC44539
140 RUN
```

Running Writing

By George McLintock

This programme explains and demonstrates how to print letters in running writing script. It was written by George McLintock for the Tandy Color Computers 1, 2 and 3.

You can have quite a bit of fun with the graphics mode on your printer once you have an understanding of how it does graphics. The use of graphics printing is normally associated with screen dumps, but this is not its only use. This submission provides a procedure for printing text files in 'running writing' The general procedure can be applied to other types of script, and in a follow up article I will describe a procedure for designing your own script characters.
The procedure here consists of two basic steps. Defining the running writing script to be printed in graphics mode, and a small program to actually print a text file. The actual printing process is fairly simple in itself, the relatively large text description follows from the number of different concepts involved.

PRINTER GRAPHICS:
Tandy printers do graphics in a similar way to Epsons etc, except that each column of dots is 7 dots high instead of 8, and the top dot in the column corresponds to the low order bit in the byte (binary 1) instead of the high order. An additional requirement is that the high order bit for graphic must be on. ie all graphics characters to be printed are greater than 127 decimal.

Tandy printers have two distinctly separate modes of operation. A 'text' mode and a 'graphics' mode. Some control codes can have a different meaning depending on the mode the printer is in, and many control codes have no effect in graphics mode
The control code to start graphics mode on 'newer' Tandy printers is CHR$(18), with CHR$(30) used to go to text mode. The equivalent codes for older Tandy printers (ie most of the old 7 pin ones) are CHR$(8) to start graphics, and CHR$(15) to end them.
Horizontal dot density for these printers is established by the text density in effect when the printer is put into graphics mode. ie 10 characters per inch = 60 dots per inch, 12 chars / ins = 72 dots / ins, 16.7 chars / inch = 100 dots per inch. Horizontal dot density cannot be changed in graphics mode, you have to go back to text mode to do it.
The CR/LF codes operate 'normally' in graphics mode, except that the LF distance is fixed at the height of 7 dots. ie 7/72 of an inch, or 7/60 of an inch depending on the vertical density of the printer. The only way to get a different LF distance is to go back to text mode, do the LF and then return to graphics mode again.
When in graphics mode, all characters received which are greater than 127 are printed as a graphics column of dots. A limited number of control codes are recognised and operated on normally. All others are ignored. These printers will do an automatic CR/LF when a graphics character is printed in the last dot position in a line. so they automatically 'wrap round' at the end of a line. A problem with this is that if you print a full width of graphics (eg 576 columns at 72 dots per inch) then you should not send a CR to start the next line.

CONTROL CODES:
Dot matrix printers can be used to produce quite a range of different types of output on the paper, like condensed print, underlined text etc. The printer is told when to start doing these different types of things by control codes which are sent from the computer to the printer. The printer recognises these control codes and adjusts its internal operations to change the way that it prints dots on the paper. In this context, graphics mode is just another control code sequence which tells the printer how to print dots on the paper.
Printer control codes vary a lot between different printer types, and they can also vary between different models from the same supplier. While they generally follow a similar structure, they vary in the detail.If

the control codes used here don't work, you will have to check your printer manual.

DEFINING SCRIPT CHARACTERS:
The characters defined here are a bit of a mixture.. The general shape of the letters are roughly as I remember being taught in a Queensland Primary School in the mid forties. However, that was a long time ago, so I can't be too sure. As a script, it has a number of weaknesses, but it is intended to reflect an individual style so it can't be perfect.

The following is an example of how the characters are formed

```
.....  ....xxx.....  low order bit, first row
.....  ....x.x.....
.....  ...x..x.....
.....  ..x..x......
.....  ..x.x.......
.....  ..x.x.......
.....  ..xx........  high order bit, first row
...xxx..x...  ..xx.xxx....  middle line
..x...xxx...  ..x.x..x....
.x.....x....  ..xx...x....
.x.....x....  .xx....x....
xx.....x....  xxx....x....  join point. leading edge
..x....x..xx  .xx....x..xx  join point, trailing edge
..xxxxxx.x..  .x.....xxx..  base line
......xxx...  .....  low order bit, third row
.....xx.....  .....
...x.x......  .....
..x..x......  .....
.x..x.......  .....
.x.x........  .....
..x.........  .....  high order bit, third row
```

In the program these characters are defined in DATA statements which are read into strings for processing. The sequence of values in the Data statements are: first value = number of graphic rows per character. Second value = number of bytes per graphics row. The values which follow are the Hex values for each graphic byte that defines the character, in row order sequence. Because there are so many zero bytes involved in a definition, I've compressed these in the Data statements. The code =12 means 12 zero bytes. See the appropriate DATA statements to see how these two characters are defined.

Hex digits come from a base 16 number system, where each digit has a value of 0 to 15 (Decimal). This in turn represents the state of 4 binary bits which are called a nibble. eg

```
0000 = 0   0100 = 4   1000 = 8   1100 = C
0001 = 1   0101 = 5   1001 = 9   1101 = D
0010 = 2   0110 = 6   1010 = A   1110 = E
0011 = 3   0111 = 7   1011 = B   1111 = F
```

Each byte consists of 8 bits or two nibbles, and the state (on or off) of each bit in a byte can be represented by a 2 digit Hex number
As you go further up a logic path, the contents of a byte can represent a range of different things. eg Hex 41 can represent the ASCII character A, a decimal number 65, the op code INC CL etc. At the basic level, all that each Hex digit represents is the state of 4 binary bits.
I won't attempt a description of how to design a script character, my approach is strictly intuitive. I start with something, and modify it until I like it. If you don't like it, you can change it. With running writing, the essential criteria is that all letters have the same base line, and that they join together in a reasonable way.
The letters here 'join' at the rows base line plus 1 and 2, and the start of most lower case letters provide a 'joining point' at that level.
The upper case letters here follow the same general rule for joining, which works OK where the letter following it is a lower case one. However, when an upper case letter joins another upper case letter it doesn't always work too good. For these, you can get an acceptable looking result by putting a space in your text to separate all upper case letters that follow each other.

THE SCRIPT PROGRAM:
The Script program will print any 'normal' ASCII text file from disk using any graphics type script style that is defined according to the above general rules.

CoCo Basic has a problem with its LINE INPUT command from disk so that it will not handle lines longer than about 250 bytes correctly. It simply 'loses' all characters after that. I've never bothered to find out why this is so, nor do I know if it was fixed with the 1.1 ROM. I've always worked around it in one way or another.
While it is not unduely difficult to do this for a traditional type file, it would require a complete re-write of the 'read file from disk' routine, and get a little complicated to handle all types of files with the same routine. Hence, as set up, you CAN'T handle text files with more than about 240 characters between CR's. It is a bit of a problem because the traditional type file is more common on CoCo's than on MS-DOS machines. But it is a fact of life with this one

The program follows a fairly standard procedure for formatting a text file, and is relatively small in itself. Most of the program is made up of the DATA statements used to define the script. This part of the code will be used with other programs, and starts from line # 2800 to be consistant with these other programs

The script defined here is variable width, ie the width is not the same for all characters. Individual characters range from 7 to 23 dots wide (Lower case i to upper case M), or as small as 5 dots wide for the single quote if you count these. This means that you cannot use a single

standard line width to format the text to be printed. The script program has to be able to establish its own line width while printing. This is much easier to achieve with the files produced by the traditional style of word processors, but as the other types are around, and fairly popular, I've included a procedure to handle these as well.

The GOSUB at Line # 530, together with the switch FX is used to convert any style of text file into a single stream of text characters for the print loop. This routine is called to read a string from the input file. A string is returned in A$, with all formatting CR's removed. Basic's LINE INPUT command will read a continuous stream of characters into a string until it is terminated by either a CR in the file, or it has read 255 characters. Any CR encountered is not returned in the string, however, the LF/CR sequence (this order only) is returned and will not stop the input stream. FX is used as a switch to tell the GOSUB what sort of text file it is reading, so that it can provide the text to the print loop in the format required.

FX is set to a value of zero or more to handle all the other funnies that can occur with 'normal' text files. This effectively reads files as a two stage process. When called, it will normally only return a non null string. It does this by continually reading records from the file until it gets one which has valid characters in it. This is the record that is returned to the print loop. The value of FX is used to determine when an end of paragraph is reached, and this is defined to be when it has read FX records without any valid characters in them. The paragraph breaks are handled by this GOSUB as well.

If you are not sure of the sort of file produced by your word processor, then just try increasing values of FX, until it produces the correct result. A value of zero to three will normally work, with zero for a traditional type file and a three for a print file.

Two other problems can occur here. Because each individual line that is separated by a CR, won't have a blank on the end to separate the last word in this line from the first word in the next line. To compensate for this, a blank is put on the end of all lines that are not a paragraph end. (But only when FX is greater than zero, this doesn't happen for traditional type files). The other problem is that the start of a new line can contain some 'extra' characters and blanks that are not wanted. There is an extra little routine included that simply removes these. If you want to retain leading formatting blanks, then remove the test for blanks in line # 570. If this general approach doesn't suit your requirements, then you can recode this GOSUB to suit the specifics of your own word processor.

PRINT LOOP:

After all that, we now have a 'normal' type input stream of characters that we can print. To handle the variations that can occur later, the print loop first builds up information about each line to be printed in some arrays. The L$() array contains the characters to be printed across the line, where each element in the array contains a single character. The corresponding element number in the other arrays contain information about that character. The L() array contains the width of the character in number of dots, and the K() array contains its position in the character set. ie it is the element number in the K$() array which defines the actual character in graphics format.

As it goes, the program accumulates (in T) the total number of dots required to print all the characters on the paper. When T exceeds the value of LL (width of a line in dots) then we have got the character that would go past the end of the line, but we still need to have this character for the word wrap test.

The GOSUB from line 240 does the word wrap adjustments at the end of each line. This consists of two steps. The first is to search back through the print line (in L$()) until we find a blank character. If there are none, then we break the line at the last character that will fit in the line. The second step is to eliminate any blanks immediately following the last character printed in a line. Both these steps are necessary to allow for all possible combinations that might occur when doing a word wrap.

There are two pointers involved in this procedure. 'X' points to the position of the 'next' character in the input stream, while 'Q' points to the position of the 'next' character in the line to be printed. The 'next' character in each case need not be the same one because of the different logic involved for each case. We are tracking two different things here, and they control different operations independently. X follows the characters comming from the input stream and determines when we have to go back to the input file to get more characters. Q follows the characters going to the output stream to be printed, and determines when a line is to be printed.

The line of text to be printed has now been defined and is set up in the L$() array. It can now be printed by the actual print loop starting at line # 380. There are two entry points provided here, one at line 360 and the other at 380. As set up here both do the same thing because the second option has not been included here.

PRINTING THE LINE:

The program now builds up the actual graphics rows to be printed, using the information contained in the arrays set up earlier. As noted, each row of text actually requires 3 graphic rows of dots, and these have to be set up in the sequence required

The loop FOR Y = 0 TO CX (Where CX = 2 here) sets up each of the three graphic rows to be printed in sequence from the top down. A single Basic string can have a maximum length of 255 bytes, which is not enough to hold a full row of dots across the page, so more than one string element is required. The complete row is built up in the array PL$(9). The pointer G is used to 'step' from one element to the next as a string is filled. (As a matter of interest, the extra large arrays in the program are there to suit a Laser jet at 300 dots per inch)

The first thing put into the string to be printed is the left margin (in LM$). This is then followed by the appropriate part of each individual character to be included in the line. The variable Y is used to 'pick out' the subset of the definitions in K$() which is required for each character across the line. The actual code for doing this can be compressed into a single line (#420) because of the way that the information is built up in the arrays earlier on. The variable K provides the essential link for this procedure.
The last thing done before printing is to set the graphics printer control codes required. The first part of this was set up at the start of the program. That bit set graphics mode and density. To complete the control code we have to add the bit that specifies the number of graphic columns to be printed. The complete control code required is put into PL$(0). In fact, this could have been done outside the Y loop, it is included here instead for a reason which will come later.

PAGE LENGTH:
Another feature of the script here concerns the space allocated for a blank line to separate paragraphs. For normal text, the line feed distance for a blank line is the same as a line feed for any other line. However, a line of text here requires 3 graphics rows and I feel that this is excessive for a blank line in this script, so I have made a blank line equal to a single row of graphics print (7 dots), and this is sufficient to provide an obvious paragraph break.
This means that the actual number of lines of text printed on a page will also vary (and why shouldn't it). To handle this we require a procedure to count 'rows of dots' down the page rather than rows of text. This is done with the variable LD. After each row is printed, the program checks if there is still sufficient space on the page to hold another row of text. If not, then it skips to the start of the next page.
There is no parameter here to specify the size of the top or bottom margin on a page. The only one provided is for the number of dot rows to print per page. For this one, you set the top margin manually when you switch on the printer. The 'top of form' that the printer will skip to when it 'counts out' the remaining lines required to get the paper to the required starting position on the next page will be the same as the page above.

OTHER OPTIONS:
The program contains parameters for various other options that can be applied. These are set in the program code during initialisation. Most changes for 60 dot/inch printers would be made here. Parameters involved are

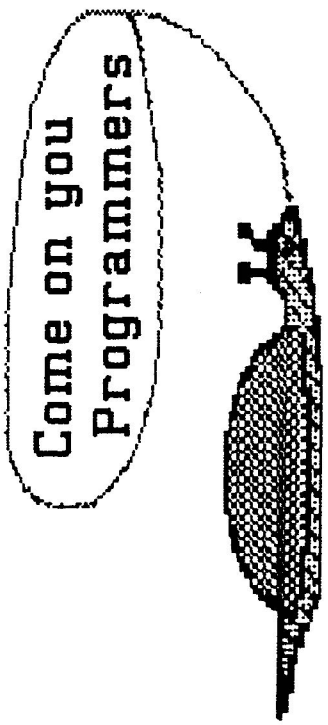
LM size of left margin in dots
LL number of dots per line to be used for text
LD number of dots deep per page for printing

In the next issue of COCO-LINK I will submit a program for changing the script.

```
5 POKE149,0:POKE150,18:'SET BAUD
 RATE TO 2400
10 'Called TANDYSCP - To print t
ext file in Running Writting
20 ' By George McLintock AND AS
MODIFIED FOR COCO
25 ' IS CHANGE MODIFIED FOR TAND
Y PRINTERS
30 PCLEAR 1:CLEAR 6000
40 DIM K$(90),N$(90),K(500),L(50
0),L$(500),PL$(9)
50 CLS: PRINT "PRINT TEXT FILE I
N RUNNING WRITTING - BY GEORGE M
CLINTOCK"
60 PRINT:PRINT "FX IN LINE # 100
 SPECIFIES FILE TYPE" : PRINT "N
OTE THAT FX=0 WILL NORMALLY NOT
WORK CORRECTLY ON COCO"
70 PRINT: LINE INPUT "ENTER FILE
 NAME TO PRINT ";N$
80 LM = 10: LM$="" : IF LM > 0 T
HEN LM$ = STRING$(LM,128) 'Left
margin
90 LL = 450: LD = 780 'Width & D
epth of print area in dots
100 FX = 0 'Varies with file typ
e 0=Traditional 1&2=Other 3=Prin
t
110 PRINT #-2,CHR$(30)+CHR$(27)+
CHR$(23); 'Set 72 dots / inch
120 PRINT #-2,CHR$(18); 'Start g
raphics
130 ' Not required CoCo
140 PRINT:PRINT "DOING SCRIPT":
GOSUB 2800 'Set up Graphic strin
gs
150 OPEN "I",#1,N$: T=0: Q = 1
160 IF EOF(1) THEN 670 ELSE GOSU
B 530
170 FOR X = XT TO LEN(A$) 'Proce
ss each character
180 L$(Q) = MID$(A$,X,1) : K(Q)
= INSTR(K$,L$(Q)) 'Next characte
r
190 L(Q) = ASC(MID$(K$(K(Q)),2,1
)): T = T + L(Q) 'Its length
200 IF T > LL THEN GOSUB 240: GO
SUB 360: Q=0'Word wrap and Print
 Line
210 Q = Q + 1: NEXT X
220 GOTO 160 'Continue next line
 of print
230 'Word wrap at end of line of
 print
240 G=1: FOR Y = Q TO 1 STEP -1
250 IF L$(Y) = " " THEN 280 'Fou
nd blank
260 G = G + 1: NEXT Y: G = 1 'Is
 none
270 'Move pointers back to last
blank or character
280 FOR Y = 1 TO G
290 T = T - L(Q): Q = Q - 1: X =
 X - 1: IF X < 1 THEN 320
300 NEXT Y
310 'Eliminate any blanks at sta
rt of next line
320 FOR Y = X + 1 TO LEN(A$)
330 IF MID$(A$,Y,1) <> " " THEN
RETURN
340 X = X + 1: NEXT Y: RETURN
350 'Print full line
360 'Any justify to right margin
 would go in here
370 'Print partial line (No just
ification)
380 IF T = 0 THEN Q=1: RETURN
390 FOR Y = 0 TO CX 'Graphics ro
ws per line
400 G = 1: PL$(G) = LM$: FOR K =
 2 TO 9: PL$(K) = "": NEXT K
410 FOR K = 1 TO Q 'For each let
ter
420 PL$(G) = PL$(G) + MID$(K$(K(
K)),3 + Y*L(K),L(K))
430 IF LEN(PL$(G)) > 200 THEN G
= G + 1
440 NEXT K
450 IF LEN(PL$(G))> 0 THEN G = G
 + 1 'Adjust
```

```
470 FOR GX=0 TO G: PRINT#-2,PL$(
GX);:NEXT GX
480 PRINT #-2 'Do CR + LF
490 NEXT Y: T = 0: Q = 1
500 DY = DY + CD: IF DY + CD > L
D THEN FOR Y = DY TO LD STEP 7:
PRINT #-2: NEXT Y: DY = 0 'Page
end
510 RETURN
520 'Get next input line
530 LINE INPUT #1,A$: IF FX > 0
THEN A$ = A$ + " "
540 GOSUB 580
550 IF XT > LEN(A$) THEN FY = FY
+1: IF FY > FX THEN GOSUB 620 EL
SE 530
560 FY=0: RETURN
570 'Exclude unwanted characters
 at start of line
580 XT=1: FOR XT = 1 TO LEN(A$):
 X$ = MID$(A$,XT,1)
590 IF X$ = " " OR X$ = CHR$(13)
 OR X$ = CHR$(10) THEN 600 ELSE
RETURN
600 NEXT XT: RETURN
610 'Do end of paragraph
620 Q=Q-1: GOSUB 380 'Print any
left
630 PRINT #-2 'Blank line
640 DY = DY + 7: IF DY + CD > LD
 THEN FOR Y = DY TO LD: PRINT #-
2: NEXT Y: DY = 0 'Page end
650 RETURN
660 'End of file
670 CLOSE: Q=Q-1: GOSUB 380: LPR
INT CHR$(30)+CHR$(27)+CHR$(19);
'Reset printer
680 STOP
2790 'Set up script characters
2800 K$ = " ABCDEFGHIJKLMNOPQRST
UVWXYZabcdefghijklmnopqrstuvwxyz
1234567890-=,.?" + CHR$(34) + "'
!@#$%&*()+:;/"
2810 PRINT "Setting up character
s": FOR X = 0 TO LEN(K$)
2820 READ T,Q: K$(X) = CHR$(T) +
 CHR$(Q): Q = T * Q
2830 FOR Y = 1 TO Q
2840 READ A$: T = VAL("&H" + A$)

2850 IF LEFT$(A$,1) <> "=" THEN
K$(X) = K$(X) + CHR$(T): GOTO 28
80
2860 T = VAL(MID$(A$,2)): K$(X)
= K$(X) + STRING$(T,128)
2870 Y = Y + T -1
2880 NEXT Y: READ N$(X): PRINT N
$(X);" ";
2890 NEXT X: CX = ASC(MID$(K$(1)
,1,1)) - 1: CD = (CX+1)*7 'Lines
 deep
2900 RETURN
3000 DATA 3,10,=2,DE,D2,D2,D2,D0
,=5,FF,84,84,8C,94,E7,=4,BE,92,9
2,92,B2,9C,=2,Not recognised
3010 DATA 3,8,=24,Blank
3020 DATA 3,14,=5,E0,90,88,86,82
,FF,=3,C0,E0,B8,9E,87,82,82,C2,C
2,FE,FF,C0,C0,A0,=14,A
3030 DATA 3,15,=1,E0,D0,CE,C3,C1
,C1,C1,C1,E1,91,91,9F,=2,FF,C0,C
0,C0,C0,C0,C0,C0,C0,E1,DE,C0,
C0,A0,=15,B
3040 DATA 3,13,=1,C0,B0,88,86,=1
,81,81,99,91,91,91,8E,90,8F,B0,C
0,C0,C0,C0,C0,C0,A0,A0,A0,A0,=13
,C
3050 DATA 3,19,=7,FE,87,99,E1,81
,81,81,FE,=4,F0,D0,D0,D0,A0,D8,C
7,C0,C0,C0,C0,C1,B2,8E,87,84,88,
90,A0,=19,D
3060 DATA 3,14,=2,C0,DF,E1,C1,C1
,81,85,89,89,86,=2,9E,A1,C0,C0,C
0,C0,C0,C0,C0,A0,A0,A0,A0,A0,=14
,E
3070 DATA 3,15,=3,82,85,89,C9,C1
,C1,FD,C3,C1,C1,81,81,98,E4,C4,C
8,C8,C0,B0,9C,A3,A0,A0,A0,A0,A0,
A0,=15,F
3080 DATA 3,13,E0,9C,86,81,81,81
,81,81,81,81,89,86,97,9C,A0,E
0,C0,C0,C0,F0,88,C0,C0,A0,A0,=1,
B8,C4,C2,82,A1,9D,83,81,=4,G
3090 DATA 3,17,=2,86,89,91,81,F1
,CE,C0,A0,A0,F0,9C,8A,89,85,83,F
0,C0,C0,A0,B1,8F,81,=3,FF,C0,C0,
E0,A0,A0,A0,=17,H
3100 DATA 3,13,=2,F0,C8,C4,C0,C2
,C0,C1,FF,C0,C0,=1,98,E4,C4,C0,C
0,C0,A0,98,87,=17,I
3110 DATA 3,12,=1,E0,90,88,84,82
,=1,C1,BD,83,=3,83,82,82,82,82,F
E,C3,82,A2,A2,A0,A0,D8,C4,A2,9C,
83,81,=5,J
3120 DATA 3,17,=3,81,FF,E0,E0,E0
,E0,90,88,84,82,81,81,81,81,E0,C
0,C0,A0,9F,=3,87,98,E0,C0,C0,C0,
A0,A0,=18,K
3130 DATA 3,14,=6,C0,F8,A6,A1,91
,9F,=2,F0,D0,D0,D1,D1,B1,C8,C7,C
0,C0,C0,C0,A0,A0,=14,L
3140 DATA 3,23,86,89,81,81,81,FF
,9C,82,81,81,81,E1,FE,9C,82,81,8
1,81,E1,9E,=5,E0,98,86,81,=3,E0,
9C,87,=5,FC,C3,C0,E0,A0,A0,=23,M
3150 DATA 3,17,86,81,81,81,E1,FE
,9C,82,81,81,81,81,E1,9E,=5,E0,9
8,87,=5,F8,C6,C1,C0,A0,A0,A0,=17
,N
3160 DATA 3,13,C0,B0,88,86,=1,87
,99,E1,81,81,=1,FE,=1,8F,B0,C0,C
0,C0,C0,C0,C0,B1,8E,87,84,98,=13
,O
3170 DATA 3,15,=2,C0,B8,86,81,81
,81,81,C1,C1,C1,C1,A1,9E,C0,F8,C
7,81,81,81,81,81,81,=21,P
3180 DATA 3,16,=2,E0,98,84,82,81
,81,81,81,81,81,82,FC,=2,BE,D1,8
8,88,88,88,98,90,B0,A0,C0,C0,FC,
C3,C0,C0,=2,81,81,81,81,81,81,81
,81,81,=5,Q
3190 DATA 3,15,=2,C1,BD,83,81,81
,C1,C1,C1,C1,C1,C1,BE,=1,C0,F8,C
7,81,81,81,81,=1,FF,C0,C0,C0,A0,
A0,A0,=15,R
3200 DATA 3,14,=5,86,D9,B1,CF,C0
,=4,C6,D8,B0,CC,C2,C1,C0,C0,C0,C
0,A3,BC,A0,A0,=14,S
3210 DATA 3,15,=3,82,85,81,81,81
,81,F9,87,81,81,81,81,98,E4,C4,C
8,C0,C0,A0,98,87,=21,T
3220 DATA 3,16,8E,81,81,81,F1,8E
,=5,C0,BC,83,=5,9C,E3,C0,C0,C0,E
0,98,FE,C3,C0,C0,A0,A0,=16,U
3230 DATA 3,16,86,89,81,81,F1,8E
,=4,9F,E1,F9,87,=5,BC,C3,C0,C0,A
0,90,8C,82,81,=1,83,84,84,=16,V
3240 DATA 3,20,86,89,91,81,81,FE
,=2,E0,98,86,FF,=1,C0,B0,88,84,8
2,81,81,=4,FC,B3,8C,83,=2,FF,98,
86,81,=26,W
3250 DATA 3,16,=3,86,89,89,81,81
,FF,98,86,=1,81,81,81,83,F0,C0,C
0,C0,C0,B0,8C,9F,E1,C0,C4,C4,C4,
B8,=18,X
3260 DATA 3,15,86,81,81,81,F1,8E
,=5,C0,BC,83,=4,9C,E3,C0,C0,C0,C
0,C0,B8,C7,C0,A0,A0,=4,F8,C4,A0,
9A,86,81,81,=4,Y
3270 DATA 3,12,=3,FE,C0,C1,A1,99
,81,81,C1,BE,F0,D0,D0,D0,D0,E0,A
0,B0,9C,92,89,88,A0,D8,84,82,A1,
9F,=6,Z
3280 DATA 3,13,=13,90,98,E6,C1,C
1,C1,C2,FE,C7,C0,A0,A0,A0,=13,a
3290 DATA 3,11,=2,F8,C6,B1,89,87
,=4,90,88,FF,C1,C0,A0,90,8F,90,A
0,A0,=11,b
3300 DATA 3,9,=9,90,98,E6,C1,C1,
C1,A2,A0,A0,=9,c
3310 DATA 3,13,=8,C0,B0,8E,81,=1
,90,98,E6,C1,C1,C1,C2,FE,C7,C0,C
0,A0,A0,=13,d
```

```
3320 DATA 3,10,=10,A0,A0,BC,E2,D
1,D1,CE,C0,A0,A0,=10,e
3330 DATA 3,10,=5,F0,CE,A1,99,87
,90,90,88,EC,FE,C3,C0,C0,A0,A0,A
0,D8,86,BF,=6,f
3340 DATA 3,12,=12,90,9C,E2,C1,C
1,C1,C2,FE,83,C0,A0,A0,=1,A0,D8,
84,A0,9E,83,81,81,=3,g
3350 DATA 3,12,=2,F8,C6,B1,89,87
,=5,90,F8,BF,85,82,81,81,FF,C0,C
0,A0,A0,=12,h
3360 DATA 3,7,=3,90,90,=2,90,FC,
C7,C0,C0,A0,A0,=7,i
3370 DATA 3,11,=7,90,90,=2,90,90
,90,88,88,FC,87,=1,C0,C0,A0,98,A
4,C2,A2,9E,83,81,81,=3,j
3380 DATA 3,12,=2,F8,C6,B1,89,87
,=5,90,FC,8F,81,88,99,E9,C5,C2,C
0,A0,A0,=12,k
3390 DATA 3,8,=2,F8,C6,B1,89,87,
=1,90,88,FF,C1,C0,C0,A0,A0,=8,l
3400 DATA 3,19,=19,90,88,86,E1,B
F,84,82,81,C1,BF,84,82,81,F1,CE,
C0,C0,A0,A0,=19,m
3410 DATA 3,13,=13,90,88,86,C1,B
F,84,82,81,81,FF,C0,C0,A0,=13,n
3420 DATA 3,12,=12,90,98,E6,C1,C
7,D9,D2,BE,90,90,A0,A0,=12,o
3430 DATA 3,14,=5,E0,B8,=7,90,90
,88,F8,BE,AD,C2,C1,E1,D2,CE,C0,A
0,A0,C0,B0,8E,81,=10,p
3440 DATA 3,12,=12,90,9C,E2,C1,C
1,C1,E2,9E,83,=1,E0,A0,=3,B8,C6,
81,A0,98,84,83,=2,q
3450 DATA 3,11,=3,C0,=7,90,88,86
,83,82,E2,DE,C0,C0,A0,A0,=11,r
3460 DATA 3,10,=3,C0,=6,90,F8,C6
,C3,CC,D0,F0,E0,C0,A0,=10,s
3470 DATA 3,8,=2,90,D0,F0,9E,91,
90,90,88,FE,C7,C0,C0,A0,A0,=8,t
3480 DATA 3,12,=12,90,FC,C7,C0,C
0,A0,90,FC,C3,C0,A0,A0,=12,u
3490 DATA 3,14,=14,98,86,81,81,B
E,C0,C0,C0,A0,9E,8F,90,A0,A0,=14
,v
3500 DATA 3,16,=16,90,B8,C6,C1,C
0,A0,F8,C7,C0,C0,A0,9E,8F,90,A0,
A0,=16,w
3510 DATA 3,14,=14,90,88,84,C6,C
1,C1,B1,FE,C2,C1,C1,A2,A0,A0,=14
,x
3520 DATA 3,11,=11,90,FC,C7,C0,C
0,A0,F0,CC,C3,A0,A0,=1,A0,D8,84,
A2,9F,81,=4,y
3530 DATA 3,12,=12,90,88,C6,C1,C
1,A1,92,CE,C0,C0,A0,A0,=1,A0,D8,
85,BF,81,81,=5,z
3540 DATA 3,8,=2,84,82,=1,FF,=7,
FF,=10,1
3550 DATA 3,15,=3,84,82,81,81,81
,81,C1,A1,9E,=5,C0,E0,D0,C8,C4,C
2,C1,C0,C0,C0,C0,=17,2
3560 DATA 3,15,=4,81,81,81,91,99
,97,A1,E1,=5,90,A0,C0,C0,C0,C0,C
0,C0,C0,B0,8F,=17,3
3570 DATA 3,16,=4,E0,90,88,86,=1
,FF,=8,82,83,82,82,82,82,82,FF,8
2,82,82,82,=18,4
3580 DATA 3,15,=3,B8,97,91,91,91
,91,91,91,E1,=5,90,A0,C0,C0,C0,C
0,C0,C0,C0,B0,8F,=17,5
3590 DATA 3,14,=2,F0,CE,A1,91,91
,91,A1,A1,C3,=5,87,99,E0,C0,C0,C
0,C0,C0,B0,8F,=16,6
3600 DATA 3,15,=2,81,81,81,81,81
,81,81,E1,99,87,81,=7,C0,B0,8C,8
3,=21,7
3610 DATA 3,15,=3,86,D9,D1,B1,A1
,B1,D1,D1,9F,=5,9E,A1,C0,C0,C0,C
0,C0,C0,C0,E1,9E,=17,8
3620 DATA 3,14,=2,98,E6,=1,81,81
,81,81,81,C0,FE,=5,90,E1,C1,C2,C
2,C1,C1,A0,9F,=16,9
3630 DATA 3,15,=2,F0,8C,82,81,81
,81,81,81,81,8E,F0,=4,83,9C,A0,C
0,C0,C0,C0,C0,C0,B8,87,=17,0
3640 DATA 3,18,=20,81,81,81,81,8
1,81,81,81,81,81,81,81,81,81,=20
,-
3650 DATA 3,18,=2,A0,A0,A0,A0,A0
,A0,A0,A0,A0,A0,A0,A0,A0,A0,A0,=4,8
4,84,84,84,84,84,84,84,84,84,84,
84,84,84,=20,=
3660 DATA 3,7,=9,E0,E0,C0,=4,84,
82,81,=2,Comma
3670 DATA 3,6,=8,E0,E0,=8,Full s
top
3680 DATA 3,14,=2,84,82,81,81,81
,C1,A1,A1,90,8E,=8,E0,E3,=20,?
3690 DATA 3,8,=2,9F,=2,9F,=18,Do
uble quote
3700 DATA 3,5,=2,9F,=12,Single Q
uote
3710 DATA 3,6,=3,FF,=4,E0,E3,=8,
!
3720 DATA 3,14,=2,F0,88,88,E4,A4
,A4,E8,88,90,E0,=4,87,88,90,93,9
2,92,93,92,82,81,=16,@
3730 DATA 3,16,=3,A0,A0,A0,F8,A7
,A0,A0,F8,A7,A0,A0,=4,84,84,E4,9
E,85,84,E4,9E,85,84,84,=6,9C,83,
=2,9C,83,=7,#
3740 DATA 3,15,=2,86,98,91,A1,FF
,C1,C1,FF,81,81,82,=4,90,A0,C0,C
0,FF,C0,C0,FF,C1,C3,BC,=8,87,=2,
87,=5,$
3750 DATA 3,13,=2,BC,BC,BC,9C,C8
,A8,98,88,84,=4,90,88,84,82,81,9
C,9E,9E,9C,=15,%
3760 DATA 3,19,=5,C0,C6,B9,E1,91
,8E,=3,F0,90,B0,=4,9C,A2,C1,C0,C
0,C0,C0,C3,AC,90,AC,C3,C0,C0,E0,
=21,&
3770 DATA 3,12,=2,A0,C0,C0,=1,F8
,C0,C0,A0,=4,84,82,82,81,9F,82,8
2,84,=14,*
3780 DATA 3,9,=2,E0,9E,=2,81,=4,
87,F8,=9,83,84,=3,(
3790 DATA 3,9,=2,81,81,=1,82,FC,
=7,C0,BF,=4,84,82,82,81,=3,)
3800 DATA 3,18,=8,FE,=11,81,81,8
1,81,81,81,FF,81,81,81,81,81,81,
81,=20,+
3810 DATA 3,6,=2,B0,B0,=4,E0,E0,
=8,
3820 DATA 3,7,=2,B0,B0,=5,E0,E0,
C0,=4,84,82,81,=2,;
3830 DATA 3,14,=9,C0,E0,B0,98,88
,=2,C0,E0,B0,98,8C,86,83,81,=4,8
2,83,81,=11,/
```



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.

CLUB CONTACTS

Adelaide.....Laurie O'Shea
. 08 363 2647
. (after 7.30pm)
. Glenys Ferres
. 08 332 4264

Basic.....Johanna Vagg
. 066 522 943

Brisbane North....M.Webster
. 07 285 6551

Brisbane S/W.....Bob Devries
. 07 372 7816

Geelong.....Alan Murrells
. 052 753 065

Moe User Group....Joseph Hester
. 051 277 817
. Ian Taffs
. 051 275 751

OS9 User Group....Gordon Bentzen
. 07 354 5141

Peninsula CCC...Bob Charleston
. 059 791 922
. Sue Jones
. 059 773 292
. Gordon Chase
. 059 711 553

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

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

Coco 1,2&3 Disk/Tape **Application**

Address

NATIONAL OS9 USER GROUP
8 ODIN ST.
SUNNYBANK
QLD 4109

By Robbie Dalzell

This programme originated back in 1983 and was written for cassette use by Edwin P. Meiners of the U.S.A. I have used the programme since that time but over the years it has seen some changes. Some of Edwin P. Meiners code remains but has been shifted around and added to. From a basic address list programme, "Address" has become a full-blown address data base and label printer with many features. These include:

1. Disk or Cassette usage.
2. Files can be appended.
3. Records can be added at any time.
4. All records can be changed or deleted.
5. Records can be sorted alphabetically.
6. Labels can be printed out in single or double rows.
7. Individual labels can be printed.
8. Labels can be printed by type of record.
9. Labels cfan be printed from a particular record number to end of file.
10. A complete list of all records can be printed.

As listed ADDRESS is written for the Coco 3. To use it on a Coco 2 requires only minimal alterations. Any changes required are listed at the end of this article.

The 200 max records recorded in line 40 is a purely arbitrary figure. As the records are stored sequentially and are of varying lengths it is impossible to give an exact figure.
"ADDRESS" is very user friendly and should be easy to follow from the menus.

Reference names should preferably be the surname of the person to be recorded. This is the information which is used for sorting the file

ENTERING A RECORD:
Line 1: Full christian and surname.
Line 2: Number and street name.
Line 3: Your city or suburb.
Line 4: State and postcode.
Reference name.
Address type.
Phone No.
Comment.

Type of file alludes to whether you want to separate addresses into types such as club, business, Christmas list etc.
The Comment line allows you to add any pertinent details you consider necessary. For instance, I use it to make a note of subscription renewal dates.

The label on your Coco-Link magazine is printed using this programme. I use computer type labels, but it is possible to use "Impact" brand rolls or similar, which are available at most stationery stores for about $5.00 per 200 roll. These labels require your printer to have friction drive.

CHANGES
Some changes may be necessary to allow the programme to work on your system. The following are the most likely changes which may be required.
1. Different size labels require a different number of linefeeds between labels. Should you require to alter the spacing, this can be done in line 730. Just alter "FORX=1 TO4" to your requirements.
2. Coco 2 users should change the POKES in line 60 to POKE 65494,0 and lines 820 and 80 to POKE 65495,0. These are the speed-up and return to normal p9okes.
3. The printer band rate is found on line 25. This should be changed to suit the band rate of your printer. Line 800 records the baud rate in string form and this should also be changed to suit.
4. Line 30 contains PALEITE CODES to change my MONO monitor screen to my preference. If you use a Coco 2 or do not like what it does to your screen, delete it.
5. The codes to print the complete listing in condensed form are found on line 390. in my case these are CHR$(27);CHR$(20). Check your printer manual for the codes necessary for your printer, they may have to be altered.

```
30 PALETTE12,63:PALETTE13,0:CLS
40 PMODE0:PCLEAR1:CLEAR19000:TM=
200:DIMT$(TM)
59 '***MENU ROUTINE***
60 POKE&HFFD9,0:GG=0:CLS:PRINT"
  ADDR MENU":PRINT:PRINT" 1 SAVE
 FILE":PRINT" 2 LOAD FILE":PRINT
" 3 APPEND FILE":PRINT" 4 INSERT
 ADDR":PRINT" 5 REVIEW & EDIT AD
DR":PRINT" 6 SEARCH FOR ADDR":PR
INT" 7 LIST FILE":PRINT" 8 PRINT
 LABELS"
70 PRINT" 9 SORT FILE":PRINT"10
STOP"
80 PRINT@481,"MAX ADDR"TM"   ADD
R USED"MX;:PRINT@384:INPUT" ENTE
R NUMBER";A$:IFA$="1"THENGOSUB17
0:GOTO60ELSEIFA$="2"THEN110ELSEI
FA$="4"THENTL=0:GOSUB130:GOTO60E
LSEIFA$="10"THENCLS:POKE&HFFD8,0
:END:GOSUB810:GOTO60
90 IFA$="6"THENGOSUB370:GOTO60EL
SEIFA$="7"THENGOSUB390:GOTO60ELS
EIFA$="8"THENGOSUB590:GOTO60ELSE
IFA$="9"THENGOSUB640:GOTO60ELSEI
FA$="3"THENGOSUB200:GOTO60ELSEIF
A$="5"THENGOSUB270:GOTO60
100 PRINT@416:PRINT" 1-10 PLEASE
!":GOTO80
110 IFMX=0THEN120ELSECLS4:PRINT@
170,"> WARNING <";:PRINT@224,"TH
E FILE IN MEMORY WILL BE LOST":I
NPUT"DO YOU WISH TO CONTINUE";A$
:IFLEFT$(A$,1)<>"Y"THEN60
120 GOSUB820:GOSUB800:I=1:GOSUB2
10:MA=MX:GOTO60
130 GOSUB540:IFTL>TM THENRETURNE
LSECLS:PRINT"ADDRESS SETUP
 ^ FOR MENU":PRINT:S=0
140 S=S+1:IFS>4THEN142ELSEPRINT"
LINE";:PRINTUSING"##";S;:PRINT"?
 ";:LINEINPUTA$:IF S=1 AND A$=""
"THENRETURNELSEIFA$=""THEN142ELS
ET$(TL)=T$(TL)+LEFT$(A$,30)+"]":
IF S=1 THEN R1$=A$
141 GOTO 140
142 PRINT:LINEINPUT"REFERENCE NA
ME, OR <ENTER>      FOR FIRST 10
 CHARACTERS OF      LINE 1: ";A$
:IF A$=""THEN A$=LEFT$(R1$,10):P
RINT"REFERENCE NAME IS: ";A$:ELS
EPRINT
143 T$(TL)=LEFT$(A$,20)+"["+T$(T
L):R1$=""
150 PRINT:LINEINPUT"ADDRESS TYPE
? ";A$:T$(TL)=T$(TL)+"\"+LEFT$(A
$,7)+"\":IFTL>MA THENMA=TL
155 PRINT:LINEINPUT"TELEPHONE?";
A$:T$(TL)=T$(TL)+"*"+LEFT$(A$,12
)+"*"
160 PRINT:LINEINPUT"COMMENT? ";A
$:T$(TL)=T$(TL)+LEFT$(A$,100):MX
=MX+1:GOTO130
169 '***SAVE FILE***
170 CLS:GOSUB820:PRINT"FILE OUTP
UT INFO":GOSUB230:PRINT@384," WR
ITING FILE";:GOSUB260:OPEN"O",DV
,F$
180 K=0:FORI=0TOMA:IFT$(I)<>""TH
ENK=K+1:PRINT#DV,T$(I):PRINT@448
," AT ADDR"K"OF"MX;
190 NEXT:CLOSE#DV:RETURN
199 '*** APPEND FILE ***
200 I=MA+1:GOSUB210:RETURN
209 '***LOAD FILE***
210 CLS:PRINT"FILE INPUT INFO":G
OSUB230:PRINT@384," SEARCH FOR F
ILE";:GOSUB260:OPEN"I",DV,F$:CLS
0:PRINT@384," LOADING FILE";:GOS
UB260
220 IFEOF(DV)ORI>TM THENCLOSE#DV
:RETURNELSELINEINPUT#DV,T$(I):MA
=I:MX=MX+1:I=I+1:PRINT@448," AT
ADDR"MA;:GOTO220
230 GOSUB820:PRINT:INPUT"FILE NA
ME";F$:F$=LEFT$(F$,8):IFPEEK(188
)=6THENDV=-1ELSEGOSUB250
240 IFDV=1THENCLS0:RETURNELSEPRI
NT:PRIN"POSITION CASSETTE.":GOTO
690
250 PRINT:INPUT"USE DISK";A$:IFL
EFT$(A$,1)<>"Y"THENDV=-1:RETURNE
LSEDV=1:F$=F$+"/DAT":RETURN
260 PRINT" ";:IFF$=""THENRETURNE
LSEPRINT">"F$" ";:RETURN
269 '*** SINGLE LABEL & EDIT SEL
ECT ***
270 IFMX=0THENRETURNELSECLS:PRIN
T"EDIT SETUP          ^ FOR MENU
":PRINT:LINEINPUT"REF NAMES? ";A
$:IFA$="^"THENRETURNELSEGOSUB770
:I=0:IFGG=1THENCL$="P, ^, OR <EN
TER>"ELSECL$="C, D, -, ^, OR <EN
TER>"
280 I=I+1:IFI>MA THEN270ELSEK=IN
STR(T$(I),"["):IFK=0THEN280ELSEG
OSUB740:IFA$<>"T"THEN280
290 GOSUB560:IFCMD$="P"THENA=1:U
L=A:GOSUB605
295 IFCMD$="-"THEN300ELSEIFCMD$=
"D"THENMX=MX-1:T$(I)="":GOTO280
296 IFCMD$="^"THENRETURNELSEIFCM
D$="C"THEN310ELSE280
300 I=I-1:IFI<=0THENI=0:GOTO280E
LSEK=INSTR(T$(I),"["):IFK=0THEN2
80ELSE290
310 S=0:CLS:PRINT"CHANGE    "CHR
$(95)" DELETES A ADDR LINE":K=IN
STR(T$(I),"["):PRINT"REF NAME: "
;MID$(T$(I),1,K-1):LINEINPUT"???
?????? ";A$:IFA$=""THENNT$=MID$(
T$(I),1,K-1)+"["ELSENT$=LEFT$(A$
,20)+"["
320 S=S+1:IFS>4THEN340ELSEJ=K+1:
K=INSTR(J,T$(I),"]"):IFK=0THENGO
SUB360:GOTO340ELSEPRINT"LINE";:P
RINTUSING"##";S;:PRINT": "MID$(T
$(I),J,K-J):LINEINPUT"??????? ";
A$:IFA$=CHR$(95)THENS=S-1:GOTO32
0ELSEIFA$=""THENA$=MID$(T$(I),J,
K-J)
330 NT$=NT$+LEFT$(A$,30)+"]":GOT
0320
340 GOSUB760:PRINT"ADDRESS TYPE:
 "MID$(T$(I),J,K-J):LINEINPUT"??
???????????? ";A$:IFA$=""THENA$=M
ID$(T$(I),J,K-J)
342 NT$=NT$+"\"+LEFT$(A$,7)+"\"
344 GOSUB765:PRINT"TELEPHONE: "M
ID$(T$(I),J,K-J):LINEINPUT"?????
???????";A$:IFA$=""THENA$=MID$(T
$(I),J,K-J)
346 NT$=NT$+"*"+LEFT$(A$,12)+"*"

350 J=K+1:PRINT"COMMENT: "MID$(T
$(I),J,LEN(T$(I))-J+1):LINEINPUT
"???????? ";A$:IFA$=""THENT$(I)=
NT$+MID$(T$(I),J,LEN(T$(I))-J+1)
:RETURNELSET$(I)=NT$+A$:RETURN
360 PRINT"LINE";:PRINTUSING"##";
S;:PRINT"? ";:LINEINPUTA$:IFA$="
"THENRETURNELSENT$=NT$+LEFT$(A$,
30)+"]":S=S+1:IFS>4THENRETURNELS
E360
370 IFMX=0THENRETURNELSECLS:PRIN
T"SEARCH SETUP          ^ FOR MENU
":PRINT:LINEINPUT"TARGET? ";M$:I
FM$="^"THENRETURNELSEI=0:CL$="^"
OR <ENTER>"
380 I=I+1:IFI>MA THEN370ELSEK=IN
STR(T$(I),M$):IFK=0THEN380ELSEK=
INSTR(T$(I),"["):GOSUB560:IFCMD$
="^"THENRETURNELSE380
389 '*** LIST RECORDS ***
390 POKE155,132:PRINT#-2,CHR$(27
);CHR$(20):XX=9:S=0:SS=50:KK=0:G
OSUB550:GOSUB520:FORI=1TOMA:IFT$
(I)=""THENNEXTELSEPRINT#-2,TAB(2
);I;:GOSUB410:NEXT
400 IFL<66THENL=L+1:GOTO400ELSER
ETURN
410 IA=0:XX=9:KA=INSTR(T$(I),"["
):S=S+1
420 L=L+1:IA=IA+1:IFKA=0THEN430E
```

```
LSEJA=KA+1:KA=INSTR(JA,T$(I),"]"
):IFKA=0THEN430ELSEPRINT#-2,TAB(
XX)MID$(T$(I),JA,KA-JA);:XX=XX+2
0
430 ON IA GOSUB450,460,470,465,4
80,480,480
440 IFIA<6THEN420:PRINT#-2:RETUR
N
450 K=INSTR(T$(I),"["):RETURN
460 GOSUB760:RETURN
465 GOSUB765:PRINT#-2,TAB(XX+3);
MID$(T$(I),J,K-J);TAB(XX+20);RIG
HT$(T$(I),3)
467 IFS=SS THENSS=SS+50:GOSUB482
:RETURN
470 RETURN
480 RETURN
482 FORX=1TO12:PRINT#-2:NEXT:GOS
UB530
490 IA=6:IFKA=0THENRETURNELSEPRI
NT#-2:L=L+1:RETURN
520 FORL=0TO2:PRINT#-2:NEXT:PRIN
T#-2,TAB(9)"ADDRESS LIST: "F$
530 PRINT#-2:PRINT#-2,STRING$(11
7,"-")
531 PRINT#-2,TAB(10);"NAME";TAB(
30);"ADDRESS";TAB(50);"TOWN";TAB
(70);"STATE";TAB(90);"PHONE";TAB
(110);"RENEWAL":PRINT#-2,STRING$
(117,"-")
532 RETURN
540 TL=TL+1:IFTL>TM THENRETURNEL
SEIFT$(TL)=""THENRETURNELSE540
550 CLS:PRINT"PRINTER SETUP":PRI
NT:PRINT" 1 POSITION PAPER":PRIN
T" 2 SET BAUD RATE TO "B$:PRINT:
GOTO690
560 CLS:PRINT"ADDRESS DISPLAY"TA
B(29);:PRINTUSING"###";I;:PRINT:
PRINT"REF NAME: "MID$(T$(I),1,K-
1):PRINT
570 J=K+1:K=INSTR(J,T$(I),"]"):I
FK=0THEN580ELSEPRINT"   "MID$(T$
(I),J,K-J):GOTO570
580 PRINT:GOSUB760:PRINT"ADDRESS
 TYPE: "MID$(T$(I),J,K-J):GOSUB7
65:PRINT:PRINT"TELEPHONE: "MID$(
T$(I),J,K-J):PRINT:J=K+1:PRINT"C
OMMENT: "MID$(T$(I),J,LEN(T$(I))
-J+1):PRINT@480,"CMD: "CL$;:INPU
TCMD$:CLS:RETURN
589 '***LABEL CHOICE***
590 CLS:PRINT"LABEL SELECT
  ^ FOR MENU":PRINT:LINEINPUT"IN
DIVIDUAL LABELS (Y/N)? ";A$:IFA$
="^"THENRETURNELSEIFA$="Y"THENGG
=1:GOSUB599:GOTO270
591 IFA$="Y"THENGG=1:GOTO595
592 PRINT:LINEINPUT"ADDRESS TYPE
S? ";A$
595 INPUT"1 OR 2 UP LABELS";A:IF
 A=1 OR A=2 THEN UL=A:ELSE595
596 PRINT:INPUT"START AT WHAT FI
LE NUMBER ";II:INPUT"END AT WHAT
 FILE NUMBER   ";JJ$
597 IFJJ$=CHR$(13)THENJJ=TM ELSE
JJ=VAL(JJ$)
598 GOSUB599:GOSUB770:GOTO600
599 PRINT:INPUT"AT WHAT TAB POSI
TION WOULD YOU  LIKE SECOND LABE
L";L8:RETURN
600 GOSUB550:I=II
605 IFGG=1 AND M$<=LEFT$(T$(I),L
EN(M$)) THENA$="T":TL=I:GOTO620
610 I=I+1:IFI>JJ THENRETURNELSEI
FT$(I)=""THEN610ELSEGOSUB750:IFA
$<>"T"THEN610ELSETL=I
620 IF UL=1THEN 630ELSE I=I+1:IF
I>TM THENTN=0ELSEIFT$(I)=""THEN6
20ELSEGOSUB750:IFA$<>"T"THEN620E
LSETN=I
630 GOSUB700:IFGG=1THENRETURN
632 GOTO610
635 IFGG=1THEN640ELSECLS0
639 '*** SORT FILE ***
640 IM=0:IFMA<2THENRETURN
650 IM=IM+1:S=0:I=MA:GOSUB660:IF
S<>0THEN650ELSERETURN
660 J=I-1:IFT$(J)=""ANDT$(I)<>""
THENGOSUB680:GOTO670ELSEIFT$(J)>
T$(I)THENGOSUB680
670 I=I-1:IFI<=IM THENRETURNELSE
660
680 A$=T$(I):T$(I)=T$(J):T$(J)=A
$:S=1:RETURN
690 LINEINPUT"PRESS <ENTER> WHEN
 READY ";A$:CLS0:RETURN
699 '***PRINT ROUTINES***
700 L=0:PRINT#-2:JF=INSTR(T$(TL)
,"["):KF=INSTR(T$(TN),"[")
705 IF UL=1THENKF=0
710 IFJF=0THEN720ELSEJ=JF+1:JF=I
NSTR(J,T$(TL),"]"):IFJF=0THEN720
ELSEPRINT#-2,TAB(5)MID$(T$(TL),J
,JF-J);
720 IFKF=0THEN730ELSEK=KF+1:KF=I
NSTR(K,T$(TN),"]"):IFKF=0THEN730
ELSEPRINT#-2,TAB(L8)MID$(T$(TN),
K,KF-K);
730 PRINT#-2:L=L+1:IFL<4THEN710E
LSE FORX=1TO4:PRINT#-2:NEXT:RETU
RN
740 IFM$<=MID$(T$(I),1,K-1)ANDN$
>=MID$(T$(I),1,K-1)THENA$="T":RE
TURNELSEA$="F":RETURN
750 GOSUB760:IFM$<=MID$(T$(I),J,
K-J)ANDN$>=MID$(T$(I),J,K-J)THEN
A$="T":RETURNELSEA$="F":RETURN
760 K=INSTR(T$(I),"\"):J=K+1:K=I
NSTR(J,T$(I),"\"):IFK<>0THENRETU
RNELSEK=LEN(T$(I)):J=K:RETURN
765 K=INSTR(T$(I),"*"):J=K+1:K=I
NSTR(J,T$(I),"*"):IFK<>0THENRETU
RNELSEK=LEN(T$(I)):J=K:RETURN
770 IFA$=""THENM$="":N$=STRING$(
20,255):RETURNELSEK=INSTR(A$,"-"
):IFK=0THENM$=LEFT$(A$,20):N$=M$
:RETURNELSEIFK=1THENM$="":N$=LEF
T$(MID$(A$,2,LEN(A$)-1),20):RETU
RN
780 M$=LEFT$(MID$(A$,1,K-1),20):
IFK=LEN(A$)THENN$=STRING$(20,255
):RETURNELSEN$=LEFT$(MID$(A$,K+1
,LEN(A$)-K),20):RETURN
800 POKE&HFFD8,0:POKE149,0:POKE1
50,40:B$="2400":RETURN
820 POKE&HFFD8,0:RETURN
```

A Review by Desmond Rae

REQUIREMENTS: 128k or 512k CoCo3. 1 Drive. Printer a must, but not necessary to run (only if you want print outs). Mouse or Joystick required. A Multi-Pak or Y-cable is NOT required as Max10 has a hardware copy-protection scheme consisting of a clicker in the cassette port, and a modified Hi-res Interface to put a Mouse/Joystick into.

Max 10 is a very good wordprocessor for the CoCo 3 and is the best I have seen and used. It runs quite well on a 128k Machine. The only thing is that it takes a little while to load the font you select. This does not happen on a 512k CoCo 3.

SET UP: The program comes on one disk which is Non-protected. This makes life easier for making a backup up of the master disk for storage in a safe place.
After the backup sequence has occurred, place the copy in the drive and run CONFIG. It will ask your monitor type, set the foreground and background colours, printer type and baud rate. It will then save this to disk. If the Config is not correct you may exit and run config again.

Printing is a little slow as it does a double strike on a DMP-132 but the quality is good. On a DMP-200 it prints the four times and it is REAL dark.
The scroll is quite fast for a graphic word processor, but slow compared to a wordprocessor, say Telewriter 128. But no Wordprocessor comes even close to Max 10 when it comes to putting Pictures and Text together.

FONTS: When you buy Max 10 you get 6 fonts with it. These are Script, Prospect, Topeka, Ft Worth 10, Ft Worth 24, and Venice. All of these fonts can be mixed on one line or even in one word.
As an extra you can buy 2 extra font disks (which I have) They really add to the Pizzaz of your documents.
You can have different styles of writing as well. They are bold, underlined, italic, subscript, and superscript.
PLEASE NOTE: I have found that you can NOT run CoCoMax 3 fonts on Max 10. (if you can, I'd be interested on how to get them to work, as I have tried unsuccessfuly).

OPTIONS: Max 10 has a Cut and Paste feature, handy for moving parts of your document around. It also has a clipboard where your cutting goes until pasted.
It is possible to preview a document, while working on one. You can add another document in 3 easy steps.
There is a Find/Change icon. This allows you to find a word and change it or just find it. It also tells you how many times the word occurs.
If you speficy a page number, the document will change occordingly, and can even go straight to a page number in 2 easy steps.
Max-10 has a Key Click/Beep Toggle and a Picture Perfect toggle. This allows you to get a WYSIWYG. Short for What You See Is What You Get. This function is very handy for Picture and Text placing as you can see what it will look like when it is finally printed.
The About.. function tells you about your document. It tells you the size of the document, how much memory is left, how much memory you have used, how much disk space it will take (in Granuals) and finally, how many Words and Paragraphs are in the document.
There is a page preview function where you can view all the pages of your document by clicking on the arrows in the direction in which you want to travel.
Max-10 includes a 40,000 word spelling checker. If it finds a word that is not in its dictionary, it tells you. You can then change it or look it up. It even gives you the option of adding the word to the dictionary. Mind you, you dont have to check all text. You can select a part of the document to be checked and it will do just that section.
There is also a graphic representation of the Rulers. They specify how the text is to be printed. You can turn these on or off, but either way, they are not printed.
You can have a Headers and/or Footers on your pages. These can contain text or Graphics or both.
There is an insert page break switch. You can also set your columns. It is VERY important when you want to put columns in your document to set where each column is to be placed or the columns will write over each other and you wont be able to read it. The maximun number of

columns is 3.
One last thing it has is a special switch that allows you to read in documents from other wordproessors. The switch turns off the linefeeds so that the document can be read. You can delete these by copying one of the spaces (Represented by a full black box) into the Find/Change box and useing 'Change all' to delete them from the document.

PRINTERS: Printers Supported are DMP 105R, DMP 130R (or the DMP 132) DMP 106 in the IBM mode, CGP 220, Gemini, OKIData 92 and IBM/Epson Compatibles.
I have tried the DMP 200 as a DMP 130 and works fine. The DMP 132 I also set up as a DMP 130 and it ran faster when printing in the Double Strike. Whereas, the DMP 200 printed Four Times and took quite a while.

The disk includes a convert program, Pixtrans. It allows you to convert pictures from CoComax 3, ColorMax 3, Cocomax 2, Pmode 4 pictures in memory and HSCREEN pictures in memory. It converts these into a format it can read and you can then use these in any document you want.

SPECIFICATIONS: On 128k CoCo 3's the Buffer size is 32k, and your document can be 44 pages in length.
On 512k CoCo 3's your buffer size is 64k, and your document can be 88 pages in length. (And if you can fill that your doing good!)

MAX 10 also comes with a well presented 67 page manual which highlights all the programmes many wonderful features in an easy to understand format.

Documents may also be SAVEed/LOADed in ASCII but they takes up more disk space. The one good feature of using ASCII is for writing Basic programs.

One final word. This wordprocessor is designed for when you need varied fonts and graphic representations. If you are constantly doing School assignments (like me) or Club news letters where you use Posters or pictures that are drawn or digitized, Max-10 is ideal. I would NOT use it if I were running a business as the printout is too slow. For that type of fast work it pays to use a different Wordprocessor such as Telewriter 128 or Wordpower.

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.

★ "We're thinking of computerising your department, but we can't find out what it does."

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

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
Wax		Maze
Reversi		Tanks
Yancc		

DISK 014	11 GAMES	
3Boxes		3Vagas
About		King Tut
Memory		Nausea
Patience		Pong
Puzzle		Slither
Wigworm		

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		

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