THE Magazine for experienced TANDY Color Computer Users !

$4.50

AUSTRALIAN COCO MAGAZINE

VOL3NO8
APRIL87

Programs Featured Include:

CoCo Oscillascope

Color Dump for the PMODE 3 Screen

Joystick Simulator

Building Speaker Enclosures

Plus

Helpful Peeks and Pokes

Game, Music and More

WHAT'S ON THE BEST OF CoCoOz

Best of CoCoOz #1. EDUCATION

ROADQUIZ ..... ROB WEBB
SHARE MARKET ..... ALEPH DELTA
HANGMAN ..... ALEPH DELTA
AUSTQUIZ ..... P. THOMAS
ALPHABET ..... RON WEBB
SPELLING TUTOR..... IAN LOBLEY
TANK ADDITION ..... DEAN HODGESON
FRACTION TUTOR ..... ROBBIE DALZELL
TABLES ..... BARRIE GERRAND
ICOSA ..... BOB WALTERS
KIDSTUFF ..... JOHANNA VAGG
TAXMAN ..... TONY PARFITT
FLAGQUIZ ..... ROB WEBB

Best of CoCoOz #2 part 1 16K GAMES

PYTHON ..... W.ARMSTRONG
COCOMIND ..... STEVE COLEMAN
POKERMCH ..... GRAHAM & MATTHEWS
OILSLICK ..... JEREMY GANS
SPEEDMATHS ..... DEAN HODGESON
CCMETEOR ..... BOB THOMSON
BATTACK ..... JEREMY GANS
SKIING ..... JOSHUA GANS
PROBDICE ..... BOB DELBOURGO
RALLY ..... TONY PARFITT
CHECKERS ..... J & J GANS
FOURDRAW ..... JOHANNA VAGG

Best of CoCoOz #2 part 2 32K GAMES

TREASURE ..... DAVIDSON & GANS
SHOOTING GALLERY ..... TOM DYKEMA
MASTERMIND ..... GRAHAM JORDAN
GARDEN OF EDEN ..... DAVE BLUHDORN
ANESTHESIA ..... MIKE MARTYN
YAHTZEE ..... KEVIN GOWAN
OREGON TRAIL ..... DEAN HODGSON
BATTLESHIP ..... CHRIS SIMPSON
ADVENTURE + ... STUART RAYNER
ANDROMEDIA ..... MAX BETTRIDGE
LANDATTACK ..... ALDO DEBERNADIS

Best of CoCoOz #3 UTILITIES

SCREEN PRINT ..... TOM DYKEMA
RAMTEST ..... TOM DYKEMA
PRINT SORT ..... PAUL HUMPRIES
BEAUTY ..... BOB THOMPSON
DATAGEN ..... ROBIN BROWN
PCOPY ..... BRIAN DOUGAN
FASTEXT ..... OZ-WIZ
MONITOR + ... BRIAN FERGUSON
COPYDIR ..... THOMAS SZULCHA
LABELLER ..... FRED BISSELING
SPEED CONTROL ..... PAUL HUMPRIES
2BC ..... WARREN WARNE
CREAT-A-TITLE ..... BRIAN FERGUSON
DISKFILE ..... BRIAN DOUGAN
BIG REMARKS ..... BOB THOMSON
LABELLER ..... GORDON BENTZEN
DIR ..... MORRIE SINGER
HI ..... ALEX.HARTMANN

Best of CoCoOz #4 Business

HI ..... ALEX. HARTMANN
(disk; Disk Directory Manager)
PERSMAN ..... PAUL HUMPHREYS
(Personal Finance Management)
BANKSTAT ..... BARRY HATTAM
(Annual & Store Statement)
CC5 ..... GRAHAM MORPHETT
(tape; Sales Invoicing)
INSURE ..... ROY VANDERSTEEM
(Analyse Home Contents)
COCOFILE ..... BRIAN DOUGAN
(tape; database)
DPMS ..... PAUL HUMPREYS
(disk; Disk Program Management Sys)
DATABASE ..... PAUL HUMPREYS
(tape; THE tape database)
RESTACC ..... DUNG LY
(tape; Restaurant Accounts)
SPDSHEET ..... GRAHAM MORPHETT
(disk; 22 column spreadsheet)
PRSPDSHT ..... GRAHAM MORPHETT
(disk; prints out "SPDSHEET")
ACS3 ..... GREG WILSON
(disk; Multi disk database)

Best of CoCoOz #5 ADVENTURES

ADV 32K ..... S. RAYNER
QUEST ..... TONY PARFITT
LABYRINTH ..... JAMES REDMOND
ADV + ... SEAN LOWE
CRYSTAL ..... C & K SPRINGETT
PRISON ..... TIM ALTON
OPALTON ..... IAN CLARKE
WIZARD ..... DARRELL BERRY
TREASURE ..... C. DAVIDSON
LOST ..... ALEX. HARTMANN

Best of CoCoOz #6 PRESCHOOL

ALPHABET ..... STUART DAWSON
HATDANCE ..... JOHANNA VAGG
AUSTSONG ..... McDERMOTT FAMILY
ADVANCE ..... McDERMOTT FAMILY
WALTZING ..... McDERMOTT FAMILY
TIMEKANG ..... McDERMOTT FAMILY
BAND ..... McDERMOTT FAMILY
KIDSTUFF ..... JOHANNA VAGG
MATCHER ..... ?
LETTERS ..... JACK FINNEN
BABYSIT ..... JOHANNA VAGG
SPELLING ..... JOHANNA VAGG
SPEEDTAB ..... DEAN HODGSON
10 FACES ..... JOHANNA VAGG

Please Note : Some of the programs on Best of Cocooz # 3 and #4 will not work on the Coco 3.

Best of CoCoOz #7 GRAPHICS

LIL'COCO ..... ANDREW WHITE
THE ROOM ..... HERMANN FREDRIKSON
BACK STREET ..... JOY WALLACE
LOCO ..... MIKE D'ESTERRE
COCO ART ..... SANDY McGREGOR
KANGA ..... JOHANNA VAGG
THE BOAT ..... SANDY McGREGOR
SAD COCO ..... P. BOLLE
TOWER ..... C.A.SYMS
WINDY DAY ..... SARAH LAW
SAILING ..... STEVE YOUNGBERRY
OUTHOUSE ..... STEVE YOUNGBERRY
SMURF ..... JOHANNA VAGG
SUNSTATE ..... STEVE YOUNGBERRY
HELICOPTER ..... ANDREW WHITE
MARTHA ..... ANDREW WHITE
BAD MOON ..... STEVE YOUNGBERRY
MCC ..... JOY WALLACE
EAGLE ..... ?
BLASTER ..... PAUL YOULD
FOGHORN ..... PAUL STEVENSON

Best of CoCoOz #8 16K GAMES

ALIEN ..... STUART SANDERS
QWERL ..... DARRELL BERRY
SHOOTOUT ..... CRAIG STEWART
SHUTTLE ..... CRAIG STEWART
FROG ..... DARREN OTTERY
FROGRACE ..... TOM LEHANE
KIMMAT ..... TOM LEHANE
GRANDPRI ..... DOUG GREY
WATER WARS ..... JUSTIN LIPTON
CATERPILLER ..... JUSTIN LIPTON
DETECTIVE ..... VAL STEPHENSON
BREAKOUT ..... WHY/BILT

Best of CoCoOz #9 32K GAMES

TRIOMINO ..... BOB DELBOURGO
MATCHEM ..... CHARLES BARTLETT
GO ..... BOB DELBOURGO
NARZOD ..... MAX BETTRIDGE
CHOMPER ..... MAX BETTRIDGE
POPBALL ..... MAX BETTRIDGE
LUDO ..... WHY/BILT
SABRE ..... ANDREW SIMPSON
MOVEABOUT ..... KEVIN GOWAN
JIGSAW ..... JAMES REDMOND
LABYRINTH ..... JAMES REDMOND
TANK ..... CRAIG STEWART

Best of CoCoOz #10 Education II

METEOR ..... DEAN HODGSON
DRIVERS TEST ..... ANDREW SIMPSON
SALE ..... JUSTIN LIPTON
TABLES ..... PAT KERMODE
OPALTON ..... IAN CLARKE
CAPITAL LETTERS ..... BOB HORNE
TEST MATCH ..... JEFF SHEEN
SENTENCE ENDINGS ..... BOB HORNE
ESCAPE ..... DEAN HODGSON
RAILMATH ..... BOB HORNE
COUNTDOWN ..... DEAN HODGSON
WHATZIT ..... BOB HORNE
HOMOPHONES ..... BOB HORNE
COMPOUND WORDS ..... BOB HORNE

Best of CoCoOz #11 Education III
This is a DISK only issue!!

CHATWIN MANOR ..... BOB HORNE

TAPE $16 each

DISK $16 each

inside COCO

Utilities

Music

Graphics

Games

Applications

Business

Hardware Mod.

Plus: In a Nutshell, Data structures in Forth, Frickers Follies
M/L Programming, Fields in a Disk Record and more!

Founder: Greg Wilson.
Managing Editor: Graham Morphett.
Editor: Alex Hartman.
Accounts: Annette Morphett.
Production: Paul Wynne
Advertising: Graham Morphett.
Art: Jim Bentick.
Subeditors: John Redmond, Fred Bisselling, Jack Fricker.

Special thanks to: Geoff Fiala, Martha Gritwhistle, Mike Turk, Brian Dougon.
Deadlines: 7th of preceeding month.
Phone: 075 51 0577 or contact on Viatel *64213#
Registered Publication QBG 4009.

IN A NUT SHELL

OS–9 Level 2 Realeased!

We have been telling you all to withhold judgement on the CoCo 3, and having just seen the OS 9 Level 2 package demonstrated by Blaxland Computer Services at the meet we held at Ringwood in Melbourne, I can say with all certainty that it is going to blow your mind!!

What we have here is an operating system which is fully capable of running EIGHT programs at once with NO speed depreciation!

We sat in awe as Jerome Siappy from Blaxland ran a demo which supplied readouts on 4 windows on the one screen, whilst it also operated 4 other windows off screen!!

It means that you will be able to backup disks whilst writing to a wordprocessor, whilst operating a terminal program on line, whilst playing a game, whilst printing a file... AND MORE!!

If this isn't the biggest news in home computing, I'll go back to Melbourne!

CoCoTex 3.1

To cap a top night, Ron Wright showed version 3.1 of his CoCoTex program which provides access to Videotex services such as Viatel, Telgraf and Prestel.

The new version reflects Ron's work in getting to understand how the GIME chip operates.

He has been able to speed up the operation of the CoCo 3 on Viatel to the point where it is faster than our dedicated Sony Terminal! There are additional features too which make this program, which is available as a free upgrade to CoCoTex 3 owners, a most valuable possession!

Competitions

Don't forget the programming competitions are in full swing. The Tandy prize looks like being something really worth the effort.

For those of you who do not know about this competition, Tandy will purchase from the authors, the best CoCo 3 & the best CoCo 2 programs submitted to us by 31st July, 1987, for resale at Christmas this year.

The prize looks like being in the $1000 region - not bad for doing something you want to do anyhow!

Programs & Articles Needed!

Speaking of programs, we need some urgently! If you have some work that you've been preparing for us, we'd really appreciate getting it real soon! Feeding two mags is a bit different to feeding one!

Conf '87

We're still fiddling with Conf '87, but should have the date set by next month, and hopefully the venue too! It is certain however, that the conference will be in Sydney and that it will be in either the July holidays or in August, as previously stated. If you have any reaction to these dates, good or bad, please phone me!

Alex will be back next month with "In a Nut Shell". We figured this month, as I got to see OS 9 Level 2, that I might as well tell you all about it!

A handy Business Utility

BOOK LABEL PRINTER

16K ECB
BUSINESS

by Wim De Puit

BOOK LABEL PRINTER WAS written to print lots of sticky address labels to be used as names for my children's school books.

Of course it can be also used to print just about anything you want on a standard address label.

I have written it for a Gemini printer running at 4800 baud, but that can be changed quite easily.

The printer codes are in the first few lines; Line 20 sets the baud rate, DW$ is the code for double width, RE$ resets the printer, PI$ is for pica, EL$ is elite and CO$ is the compressed type code.

Put the codes for your printer in, and you're away!!

The first line of the label is always printed as bold as possible, with the remaining lines fitted in without going to double width. Long lines are changed to a smaller font to make them fit.

You can print any line centered, flush left or flush right as you choose.

My kids reckon they're the bees knees with printed lables on their books! I hope you can get some use out of it too!

The Listing:

```
0 GOTO10
1 '******** BOOK LABELLER ******
   ****** WIM DE PUIT **********
3 SAVE"170:3":END
10 CLEAR 500
20 ' PRINT NAME LABELS
30 CLS:PRINT@108,"BOOK"
40 PRINT@172,"LABEL"
50 PRINT@235,"PRINTER"
60 PRINT@325,"BY WIM DE PUIT":RE
M PO BOX 112 KINGSTON TAS
70 FOR I=1 TO 1000:NEXT I
80 POKE150,7 : REM 4800 BAUD
90 DIM L(5),S$(5),SP(5),PR$(5)
100 DW$=CHR$(14):EM$=CHR$(27)+"E
"
110 RE$=CHR$(27)+CHR$(64)
120 PI$=CHR$(18):DS$=CHR$(27)+CH
R$(71)
130 EL$=CHR$(27)+CHR$(66)+CHR$(2
)
140 CO$=CHR$(27)+CHR$(66)+CHR$(3
)
150 CLS:PRINT"YOU HAVE 5 LINES O
N THE LABEL.  THE FIRST LINE WIL
L BE PRINTED  IN DOUBLE WIDTH IF
 POSSIBLE.    OTHER LINES WILL B
E PRINTED IN  NORMAL OR SMALLER
PRINT AS      NECESSARY TO FIT T
HE WORDS IN.":PRINT
160 PRINT:PRINT"DO YOU WANT TO:"
:PRINT"       1. PRINT A SAMPLE L
ABEL           (TO LINE UP PRINTE
R)            2. PRINT YOUR OWN L
ABELS.        3. END.":PRINT:PRIN
T
170 PRINT"WHICH (1-3)"
180 Z$=INKEY$:IF Z$<>"1" AND Z$<
>"2" AND Z$<>"3" THEN 180
190 CLS:Z=VAL(Z$)
200 ON Z GOTO 740,220,210
210 END
220 FOR L=1 TO 5
230 PRINT"LINE";L;:INPUT S$(L)
240 L(L)=LEN(S$(L)):IF L(L)>59 T
HENPRINT"SORRY, TOO LONG.":PRINT
:GOTO 230
250 PRINT:PRINT"DO YOU WANT THE
LINE :":PRINT:PRINT"          1. F
LUSH LEFT                     2. C
ENTRED                        3. F
LUSH RIGHT"
260 PRINT:PRINT"WHICH (1-3)"
270 W$=INKEY$:IF W$<>"1" AND W$<
>"2" AND W$<>"3" THEN 270
280 W=VAL(W$)
290 PRINT
300 IF L(L)>41 THEN PR$(L)=CO$ E
LSE 350
310 IF W=1 THEN SP(L)=0
320 IF W=2 THEN SP(L)=FIX((60-L(
L))/2)
330 IF W=3 THEN SP(L)=60-L(L)
340 GOTO 590
350 IF L(L)>34 THEN PR$(L)=EL$ E
LSE 400
360 IF W=1 THEN SP(L)=0
370 IF W=2 THEN SP(L)=FIX((42-L(
L))/2)
380 IF W=3 THEN SP(L)=42-L(L)
390 GOTO 590
400 IF L(L)>30 OR L>1 THEN PR$(L
)=PI$ ELSE 450
410 IF W=1 THEN SP(L)=0
420 IF W=2 THEN SP(L)=FIX((35-L(
L))/2)
430 IF W=3 THEN SP(L)=35-L(L)
440 GOTO 590
450 IF L(L)>20 THEN PR$(L)=DW$+C
O$ ELSE 500
460 IF W=1 THEN SP(L)=0
470 IF W=2 THEN SP(L)=FIX((30-L(
L))/2)
480 IF W=3 THEN SP(L)=30-L(L)
490 GOTO 590
500 IF L(L)>17 THEN PR$(L)=DW$+E
L$ ELSE 550
510 IF W=1 THEN SP(L)=0
520 IF W=2 THEN SP(L)=FIX((21-L(
L))/2)
530 IF W=3 THEN SP(L)=21-L(L)
540 GOTO 590
550 PR$(L)=DW$+PI$
560 IF W=1 THEN SP(L)=0
570 IF W=2 THEN SP(L)=FIX((17-L(
L))/2)
580 IF W=3 THEN SP(L)=17-L(L)
590 NEXT L
600 GOSUB 620
610 GOTO 160
620 INPUT"HOW MANY LABELS DO YOU
 WANT";N
630 CLS:PRINT"POSITION LABELS (1
-WIDE) IN     PRINTER.  HIT ANY
KEY WHEN READY"
640 Q$=INKEY$:IF Q$="" THEN 640
650 FOR J=1 TO N
660 FOR L=1 TO 5
670 PR$(L)=EM$+DS$+PR$(L)
680 PRINT#-2,RE$;
690 PRINT#-2,PR$(L)+STRING$(SP(L
),"")+S$(L)
700 NEXT L
710 PRINT#-2
720 NEXT J
730 RETURN
740 RESTORE:FOR L=1 TO 5
750 READ Q1,Q2,Q3,Q4,Q5,SP(L),S$
(L)
760 PR$(L)=CHR$(Q1)+CHR$(Q2)+CHR
$(Q3)+CHR$(Q4)+CHR$(Q5)
770 NEXT L
780 GOSUB 630
790 GOTO 160
800 DATA 14,27,69,27,71,1,Marcus
 de Puit
810 DATA 18,,,,,14,Grade 2
820 DATA ,,,,,6,Calvin Christian
 School
830 DATA ,,,,,,
840 DATA 27,66,3,,,1,20 Sophia S
treet  Kingston  Tasmania  7150
 Phone 294950
```

■

'...bandits at eleven o'clock'

DOGFIGHT

32K ECB + Joysticks **GAME** **by Craig Stewart**

DOGFIGHT WAS ONE of the first Machine Language programs I created. It is a two player game in control of two bi-planes. The idea is to shoot each other out of the sky.

Being hit five times means your bi-plane goes out of control.

The bi-plane controls are elementary: pulling your joystick back will move your plane in an upward direction while pushing your joystick forward will move your plane downwards.

Note: If you want to save this program as a machine language file, type in DEL40 and type in RUN. When the 'OK' prompt comes back, type in:

```
(C)SAVEM"DOGFIGHT",24576,26837
,24576
```

The Listing:

```
0 GOTO10
1 '********* PLANE FIGHT *******
   ***** CRAIG STEWART *********
3 SAVE"146H:3":END
10 REM CLEAR MEMORY IF NECESSARY
20 FOR I= 24576TO 26837
30 READ Q$:POKEI,VAL("&H"+Q$):NE
XTI
40 EXEC 24576
50 DATA BD,68,A2,BD,68,CB,7E,61,
A3,B6,62,5F,C6,20,3D,B7,62,61,86
,8,3D,81,0,26,4,C6,80,20,8,C6,40
,4A,27,3,54,20,FA,F7,62,64,B6,62
,60,C6,20,3D,FD,62,62,F6,62,61,B
E,62,62,3A,30,89,D,A3,BF,62,65,1
0,BE,62,69,30,1D,A6,A0,F6,62,64,
3D
60 DATA A7,80,E7,84,A6,A0,F6,62,
64,3D,B7,62,67,A6,84,BA,62,67,A7
,80,E7,84,A6,A4,81,FF,27,5,30,88
,21,20,D7,39,B6,62,60,43,B7,0,8C
,5F,BD,A9,51,39,B6,62,6E,81,0,26
,5,7F,62,6E,20,5,7A,62,6E,20,3B,
AD,9F,A0,A,B6,62,6F,81,0,26,16,8
6
70 DATA 16,B7,62,6E,B6,62,6D,81,
1,26,7,86,8,B7,62,6D,20,3,7A,62,
6D,81,3F,26,16,86,F,B7,62,6E,B6,
62,6D,81,8,26,7,86,1,B7,62,6D,20
,3,7C,62,6D,B6,62,6D,81,1,26,A,1
0,8E,62,D6,7C,62,5F,16,0,C5,81,2
,26,1C,10,8E,62,F7,7C,62,5F
80 DATA F6,62,60,C1,4,22,B,BD,65
,DA,C6,B0,F7,62,60,16,0,A8,7A,62
,60,81,3,26,19,10,8E,63,18,F6,62
,60,C1,4,22,B,BD,65,DA,C6,B0,F7,
62,60,16,0,8B,7A,62,60,81,4,26,1
B,10,8E,63,39,7A,62,5F,F6,62,60,
C1,4,22,A,BD,65,DA,C6,B0,F7,62
90 DATA 60,20,6C,7A,62,60,81,5,2
6,7,10,8E,63,5A,7A,62,5F,81,6,26
,1D,10,8E,63,7B,7A,62,5F,F6,62,6
0,C1,B0,25,A,BD,65,DA,C6,4,F7,62
,60,20,42,7C,62,60,20,3D,81,7,26
,1A,10,8E,63,9C,F6,62,60,C1,B0,2
5,A,BD,65,DA,C6,4,F7,62,60,20,24
100 DATA 7C,62,60,20,1F,81,8,26,
1B,10,8E,63,BD,7C,62,5F,F6,62,60
,C1,B0,25,A,BD,65,DA,C6,4,F7,62,
60,20,3,7C,62,60,10,BF,62,69,BD,
60,9,39,B6,62,74,B7,62,6D,B6,62,
6B,B7,62,5F,B6,62,6C,B7,62,60,B6
,62,70,B7,62,6E,B6,1,5B,B7,62,6F
,BD
110 DATA 60,79,B6,62,5F,B7,62,6B
,B6,62,60,B7,62,6C,B6,62,6E,B7,6
2,70,B6,62,6D,B7,62,74,AD,9F,A0,
0,81,8,26,5,86,1,B7,62,D5,81,9,2
6,3,7F,62,D5,B6,62,D5,81,0,26,3,
BD,60,6D,B6,62,75,B7,62,6D,B6,62
,72,B7,62,5F,B6,62,73,B7,62,60,B
6
120 DATA 62,71,B7,62,6E,B6,1,5D,
B7,62,6F,BD,60,79,B6,62,5F,B7,62
,72,B6,62,60,B7,62,73,B6,62,6E,B
7,62,71,B6,62,6D,B7,62,75,B6,62,
D5,81,0,26,3,BD,60,6D,BD,63,FF,B
D,64,6E,BD,65,1A,BD,66,5A,BD,67,
6,7E,61,A3,34,2,10,8E,63,DE,10,B
F,62
130 DATA 69,BD,60,9,35,2,39,80,8
0,0,0,0,0,0,0,0,0,0,0,80,80,1,0,
0,0,0,3C,3C,1,3,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0
140 DATA 0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,A,0,0,0,0,0,0,0,0,0,3,F0,
40,20,63,42,7F,FE,7F,FE,7,FE,0,3
0,0,10,0,18
150 DATA 0,18,0,0,0,0,0,0,FF,0,0
,0,10,0,88,1,9C,2,BE,4,F8,8,F0,3
,F0,7,FE,7,8E,F,4,7E,0,38,0,10,0
,0,0,0,0,FF,0,0,3,E0,1,C0,1,CC,9
,FC,D,E0,B,C0,9,C0,B,C0,B,C0,1,C
0,1,80,1,80,3,80,7,80,0,0,FF
160 DATA 0,0,8,80,19,C0,3C,80,5F
,80,F,80,7,80,3F,C0,11,E0,9,F0,4
,F0,2,38,0,38,0,30,0,20,0,0,FF,0
,0,0,0,0,0,18,0,18,0,8,0,44,0,7F
,E0,7F,FE,7F,FE,42,C6,4,2,F,C0,0
,0,0,0,0,0,FF,0,0,0,0,0,8,0,1C,0
170 DATA 7E,0,F0,61,E0,7F,E0,F,C
0,F,10,1F,20,7D,40,39,80,11,0,8,
0,0,0,FF,0,0,1,E0,1,C0,1,80,1,80
,3,80,3,D0,3,D0,3,90,3,D0,7,B0,3
F,90,33,80,3,80,7,C0,0,0,FF,0,0,
4,0,C,0,1C,0,1C,40,F,20,F,90,7,8
8,3,FC
180 DATA 1,E0,1,F0,1,FA,1,3C,3,9
8,1,10,0,0,FF,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,FF,B6,62,6B,B0,62,
72,81,6,25,5,81,FA,22,1,39,B6,62
,6C,B0,62,73,81,6,25,5,81,FA
190 DATA 22,1,39,86,FA,B7,62,D2,
BD,66,25,BD,66,25,BD,66,25,BD,66
,25,86,A,B7,62,D2,8E,1,F4,30,1F,
8C,0,5,25,26,34,10,7F,62,6F,B6,6
2,6B,B7,62,5F,B6,62,6C,B7,62,60,
7F,62,6E,BD,60,79,B6,1,13,B7,0,8
C,5F,BD,A9,51,35,10,7E,64,36,86,
64
200 DATA BB,62,6B,B7,62,6B,7E,61
,A3,B6,62,76,81,0,27,4,7A,62,76,
39,B6,FF,0,84,1,81,0,27,1,39,86,
7,B7,62,76,8E,62,78,C6,0,A6,80,8
1,0,27,6,5C,C1,5,26,F5,39,86,1,3
0,1F,A7,84,F7,62,77,8E,62,7E,3A,
B6,62,6B,80,C,A7,84,30,6,B6
210 DATA 62,6C,8B,8,A7,84,8E,62,
C6,3A,86,2D,A7,84,B6,62,74,8E,62
,8A,3A,81,1,26,7,86,3,C6,0,16,0,
40,81,2,26,6,86,3,C6,FD,20,36,81
,3,26,6,86,0,C6,FD,20,2C,81,4,26
,6,86,FD,C6,FD,20,22,81,5,26,6,8
6,FD,C6,0,20,18,81,6,26
220 DATA 6,86,FD,C6,3,20,E,81,7,
26,6,86,0,C6,3,20,4,86,3,C6,3,A7
,84,30,6,E7,84,BD,66,25,39,8E,62
,78,C6,0,A6,80,81,1,27,9,5C,BD,6
6,54,C1,6,26,F2,39,BF,62,96,F7,6
2,98,8E,62,7E,3A,BF,62,99,A6,84,
30,6,E6,84,B7,62,9B,F7,62
230 DATA 9C,BD,65,9E,F6,62,98,8E
,62,C6,3A,A6,84,4A,27,3B,A7,84,8
E,62,8A,3A,F6,62,9C,C1,A,25,2E,C
1,B4,22,2A,B6,62,9B,AB,84,30,6,E
B,84,BE,62,99,A7,84,30,6,E7,84,B
7,62,9B,F7,62,9C,BD,65,EA,BD,67,
8A,BE,62,96,7C,62,98,F6,62,98,16
,FF,8E
240 DATA 8E,62,78,F6,62,98,3A,86
,0,A7,84,20,E4,B6,62,9B,C6,20,3D
,B7,62,9D,86,8,3D,81,0,26,4,C6,8
0,20,8,C6,40,4A,27,3,54,20,FA,53
,F7,62,64,B6,62,9C,C6,20,3D,FD,6
2,9F,BE,62,9F,30,89,D,A3,F6,62,9
D,3A,B6,62,64,A4,84,A7,84,39,34,
36
250 DATA 10,8E,63,DE,10,BF,62,69
,BD,60,9,35,36,39,B6,62,9B,C6,20
,3D,B7,62,9D,86,8,3D,81,0,26,4,C
6,80,20,8,C6,40,4A,27,3,54,20,FA
,F7,62,64,B6,62,9C,C6,20,3D,FD,6
```

```
2,9F,BE,62,9F,30,89,D,A3,F6,62,9
D,3A,B6,62,64,AA,84,A7,84,39,B6,
FF
260 DATA 1,84,F7,B7,FF,1,B6,FF,3
,84,F7,B7,FF,3,B6,FF,23,8A,8,B7,
FF,23,8E,0,0,A6,80,B7,FF,20,8D,6
,8C,0,C8,26,F4,39,B6,62,D2,4A,26
,FD,39,86,5A,4A,26,FD,39,B6,62,A
1,81,0,27,4,7A,62,A1,39,B6,FF,0,
84,2,81,0,27,1,39,86,7,B7
270 DATA 62,A1,8E,62,A3,C6,0,A6,
80,81,0,27,6,5C,C1,5,26,F5,39,86
,1,30,1F,A7,84,F7,62,A2,8E,62,A9
,3A,B6,62,72,80,C,A7,84,30,6,B6,
62,73,8B,8,A7,84,8E,62,CC,3A,86,
2D,A7,84,B6,62,75,8E,62,B5,3A,81
,1,26,7,86,3,C6,0,16,0,40,81
280 DATA 2,26,6,86,3,C6,FD,20,36
,81,3,26,6,86,0,C6,FD,20,2C,81,4
,26,6,86,FD,C6,FD,20,22,81,5,26,
6,86,FD,C6,0,20,18,81,6,26,6,86,
FD,C6,3,20,E,81,7,26,6,86,0,C6,3
,20,4,86,3,C6,3,A7,84,30,6,E7,84
,BD,66,25,39,8E,62
290 DATA A3,C6,0,A6,80,81,1,27,9
,5C,BD,66,54,C1,6,26,F2,39,BF,62
,C1,F7,62,C3,8E,62,A9,3A,BF,62,C
4,A6,84,30,6,E6,84,B7,62,9B,F7,6
2,9C,BD,65,9E,F6,62,C3,8E,62,CC,
3A,A6,84,4A,27,3B,A7,84,8E,62,B5
,3A,F6,62,9C,81,5,25,2E,C1,B4,22
,2A
300 DATA B6,62,9B,AB,84,30,6,EB,
84,BE,62,C4,A7,84,30,6,E7,84,B7,
62,9B,F7,62,9C,BD,65,EA,BD,67,CA
,BE,62,C1,7C,62,C3,F6,62,C3,16,F
F,8E,8E,62,A3,F6,62,C3,3A,86,0,A
7,84,20,E4,F6,62,72,C0,A,F0,62,9
B,C1,A,25,1,39,F6,62,9C,5A,F0,62
,73
310 DATA C1,A,25,1,39,8E,62,78,F
6,62,98,3A,86,0,A7,84,86,28,B7,6
2,D2,BD,66,25,BD,66,25,86,A,B7,6
2,D2,B6,62,D3,4C,81,5,27,44,B7,6
2,D3,39,F6,62,6B,C0,A,F0,62,9B,C
1,A,25,1,39,F6,62,9C,5A,F0,62,6C
,C1,A,25,1,39,8E,62,A3,F6,62,C3
320 DATA 3A,86,0,A7,84,86,28,B7,
62,D2,BD,66,25,BD,66,25,86,A,B7,
62,D2,B6,62,D4,4C,81,5,27,50,B7,
62,D4,39,86,B4,B7,62,D2,BD,66,25
,BD,66,25,BD,66,25,86,A,B7,62,D2
,F6,1,13,8E,0,0,3A,30,1F,8C,0,5,
25,1F,34,10,7F,62,6F,B6,62,72,B7
330 DATA 62,5F,B6,62,73,B7,62,60
,7F,62,6E,BD,60,79,BD,66,25,35,1
0,7E,68,24,B6,62,6D,B7,62,75,86,
0,7F,62,D3,39,86,DC,B7,62,D2,BD,
66,25,BD,66,25,BD,66,25,86,A,B7,
62,D2,F6,1,13,8E,0,0,3A,30,1F,8C
,0,5,25,1F,34,10,7F,62,6F,B6,62,
6B
340 DATA B7,62,5F,B6,62,6C,B7,62
,60,7F,62,6E,BD,60,79,BD,66,25,3
5,10,7E,68,70,B6,62,6D,B7,62,74,
86,0,7F,62,D4,39,7F,FF,C0,7F,FF,
C3,7F,FF,C5,B6,FF,22,84,7,8A,F8,
B7,FF,22,7F,FF,C7,7F,FF,C9,7F,FF
,CB,7F,FF,CC,7F,FF,CE,7F,FF,D0,7
F,FF,D2
350 DATA 39,8E,E,0,6F,80,8C,26,0
,25,F9,39
```

Here's something to shoot for

STARS

16K ECB
GRAPHICS

by Craig Stewart

STARS IS A SMALL graphics experiment that moves 200 odd stars on four different perspective planes, giving the illusion of moving through space.

About the program: once run it automatically self-executes. If you want to save the machine language program to tape or disk, replace line 40 with:

```
40 INPUT"SAVE TO TAPE OR DISK
  (T/D)";A$
41 IF A$="T"THEN 43
42 SAVEM"STARS",14848
  ,15661,14848:END
43 CSAVEM"STARS",14848
  ,15661,14848:END
```

The Listing:

```
0 GOTO10
1 '********** STARS ***********
   ****** CRAIG STEWART ********
3 SAVE"146G:3":END
10 REM CLEAR MEMORY IF NECESSARY
20 FOR I= 14848TO 15661
30 READ Q$:POKEI,VAL("&H"+Q$):NE
XTI
40 EXEC 14848
50 DATA 17,2,F5,8E,82,0,10,8E,3C
,30,CE,3C,94,A6,80,A7,A0,A6,80,A
7,C0,10,8C,3C,94,25,F2,10,8E,3B,
B2,CE,3B,DA,A6,80,A7,A0,A6,80,A7
,C0,10,8C,3B,DA,25,F2,10,8E,3C,2
,CE,3C,11,A6,80,A7,A0,A6,80,A7,C
0,10,8C,3C,11,25,F2,10,8E,3C,20,
CE,3C
60 DATA 28,A6,80,44,44,44,A7,A0,
A6,80,A7,C0,10,8C,3C,28,25,EF,BF
,3B,B0,17,0,1D,17,0,75,17,0,3D,1
7,0,6F,17,0,37,17,0,69,17,0,B,17
,0,83,17,0,60,17,0,28,20,E0,8E,3
B,B2,10,8E,3B,DA,A6,80,B7,3B,9E,
A6,A4,6C,A0,B7,3B,9F,34,30,17
70 DATA 0,BA,7A,3B,9F,17,0,C6,35
,30,8C,3B,DA,25,E2,39,8E,3C,2,10
,8E,3C,11,A6,80,B7,3B,9E,A6,A4,6
C,A0,B7,3B,9F,34,30,17,0,94,7C,3
B,9E,17,0,8E,7A,3B,9F,7A,3B,9F,1
7,0,97,7A,3B,9E,17,0,91,35,30,8C
,3C,11,25,D3,39,8E,3C,20,10,8E,3
C
80 DATA 28,A6,80,B7,3B,9E,A6,A4,
6C,A0,B7,3B,9F,34,30,17,0,2E,35,
30,8C,3C,28,25,E8,39,8E,3C,30,10
,8E,3C,94,A6,80,B7,3B,9E,A6,A4,6
C,A0,B7,3B,9F,34,30,17,0,3F,7A,3
B,9F,17,0,4B,35,30,8C,3C,94,25,E
2,39,B6,3B,9F,C6,20,3D,1F,1,30,8
9,E
90 DATA 0,F6,3B,9E,3A,B6,3B,9F,8
1,BF,22,19,4F,A7,88,E0,10,8E,3B,
A9,C6,5,A6,A0,A7,84,30,88,20,8C,
25,C0,22,3,5A,26,F1,39,B6,3B,9F,
81,BE,22,A,17,0,1B,B6,3B,AF,AA,8
4,A7,84,39,B6,3B,9F,81,BE,22,B,1
7,0,9,B6,3B,AF,43,A4,84,A7,84,39
100 DATA B6,3B,9E,C6,20,3D,B7,3B
,AE,86,8,3D,8E,3B,A0,30,86,E6,84
,F7,3B,AF,B6,3B,9F,C6,20,3D,1F,1
,F6,3B,AE,3A,30,89,E,0,39,80,A,8
0,40,20,10,8,4,2,1,0,8,1C,3E,1C,
8,0,0,0,0,81,2D,27,3C,81,3E,27,4
2,81,3C,27,39,81,3D,27,3F
110 DATA BD,90,AA,25,24,5F,80,30
,97,D7,86,A,3D,4D,26,19,DB,D7,25
,15,D,D8,27,17,BD,9B,98,BD,90,AA
,24,E6,C,D8,9E,D9,30,1F,9F,D9,39
,7E,B4,4A,5C,27,FA,39,5D,27,F6,5
A,39,5D,27,F1,54,39,5D,2B,EC,58,
39,34,60,8D,16,BD,B7,E,35,E0,BD,
9C,1B
120 DATA C6,2,BD,AC,33,D6,D8,9E,
D9,34,14,7E,9A,32,9E,D9,34,10,BD
,9B,98,BD,B3,A2,25,C4,BD,9B,98,8
1,3B,26,F9,35,10,DE,A6,34,40,9F,
A6,BD,B2,84,35,10,9F,A6,39,4F,1F
,8B,DC,E3,10,27,D,74,93,D5,DD,E3
,22,D,F,E3,F,E4,35,2,10,EE,67,84
,7F
130 DATA 34,2,3B,A,C,1,3,5,6,8,1
,A8,1,90,1,7A,1,64,1,50,1,3D,1,2
B,1,1A,1,A,0,FB,0,ED,0,DF,0,D3,0
,C7,0,BB,0,B1,0,A6,0,9D,0,94,0,8
B,0,83,0,7C,0,75,0,6E,A6,9C,93,8
B,83,7B,74,6D,67,61,5B,56,51,4C,
47,43,3F
140 DATA 3B,37,34,31,2E,2B,28,26
,23,21,1F,1D,1B,19,18,16,14,13,1
2,9E,8A,C6,1,34,14,D7,C2,9F,D5,B
D,95,9A,BD,B1,56,BD,B6,54,20,8,B
D,9B,98,7E,9B,BE,35,14,D7,D8,27,
FA,9F,D9,10,27,0,EA,D,D8,27,F0,B
D,9B,98,81,3B,27,F5,81,27,27,F1,
81,4E
150 DATA 26,4,3,D5,20,E9,81,42,2
6,4,7F,FF,C0,7F,FF,C3,7F,FF,C5,B
6,FF,22,84,7,8A,F8,B7,FF,22,7F,F
F,C7,7F,FF,C9,7F,FF,CB,7F,FF,CC,
7F,FF,CE,7F,FF,D0,7F,FF,D2,8E,E,
0,4F,5F,ED,81,8C,26,0,25,F9,39,3
F
```

BUILDING SPEAKER ENCLOSURES

16K/16K MC-10 | UTILITY | by Les Thurbon

I HAVE RE-WRITTEN this program mainly to operate on the Coco but will also run on the MC-10 with some alterations and they are...

For MC-10 owners, delete lines 50, 60, 70, 80, 90, and all of line 40 except 'DIM M(200)'.

Also, delete all POKE'S throughout the program and then replace all 'PRINT#-2,' statments to 'LPRINT' statements. You will also need an 80 column printer for graphs and printouts.

Coco owners who do not have or use a disk drive should change the POKE359,57 in line 60 to POKE359,13.

I found this program in the back of Tandy's book "Building Speaker Enclosures, by David Weems." (Cat. No. 62-2309)

Unless you are an Audio Engineer you WILL need a this book to go with this program.

I think the program was designed for the model one or two, however I'm not really sure and if anyone would like to upgrade the program, with on screen graphics be my guest. I'm sure Tandy will have no objection either.

The program is fully prompted and should operate on most 80 column printers as well as the screen.

The program calculates specifications for Drivers, Closed Box Speakers and Vented Box Speakers for Stereo HIFI, P.A. and general musical applications.

There is little more that I can say here except read the above mentioned book, ALL of it, and GOOD LUCK.

The Listing:

```
0 GOTO10
1 '********** HIFI ************
   ***** LES THURBON **********
3 SAVE"199:3":END
10 PMODE4,1:PCLS:SCREEN1,1
20 FORX=0TO191STEP2:LINE(X,P)-(2
55-X,191-P),PSET,B:P=P+2:NEXT:LI
NE(0,96)-(255,96),PRESET:PAINT(1
28,96),5,5:DRAW"C0BM20,86ND20R17
F3D4G3L17BD10BR50NR5R3U20NL2R3BR
100NR20D10NR15D10R20BR40NU20R20
30 DRAW"BM106,180NR4D2R4D2NL4BR2
U4R4D4NL4BR2U4NR4D2R3BR3BU2R4L2D
4BR4NU4R2NU3R2NU4BR2U4R4D2NL4D2B
R2U4R4D2L4R2F2BR2U4NR4D2NR3D2R4
40 DRAW"BM106,190U4R4D2NL4BR2BU2
ND4R4D2NL4L2F2BR2NR4U2NR3U2R4BR2
NR4D2R4D2NL4BR2NR4U2NR3U2R4BR2ND
4F4U4BR2R4L2D4BR4R4U2L4U2R4
50 DRAW"BM128,96NE17NF17NG17NH17
60 FORX=1TO2000:NEXT:CLS:POKE359
,57:SCREEN0,1:PRINT@226,"DESIGNI
NG SPEAKER ENCLOSURES";:PRINT@29
4,"COPYRIGHT (C) 1987";:PRINT@3
58,"L.W. & D.W. THURBON.";:FORX=
1TO3000:NEXT:SCREEN0,0
70 ' LOUDSPEAKER DESIGN.
     WRITTEN BY ROBERT L CLAUD.
80 ' EQUATIONS OF D.B. KEELE
     AND RICHARD SMALL.
     (C) TANDY (TM) CORPORATION.
90 ' REWRITTEN FOR THE TANDY
     COLOR COMPUTER BY
     D. W. AND L. W. THURBON.
     (C) PIXEL SOFTWARE P/L.
100 DIMM(200):CLS:PRINT:PRINT"
ENTER YOUR PRINTER BAUD RATE
1 TO 5  (NOT BAUD RATE!)":PRINT:
PRINT"  1 = 600":PRINT"  2 = 120
0":PRINT"  3 = 2400":PRINT"  4 =
 9600":PRINT"  5 = NO PRINTER":P
RINT:INPUT"  ";BR
110 IFBR=1 THEN BR=87
120 IFBR=2 THEN BR=40
130 IFBR=3 THEN BR=18
140 IFBR=4 THEN BR=1
150 IFBR=5 THEN BR=87
160 POKE149,0:POKE150,BR:CLS:PRI
NT"   LOUDSPEAKER SYSTEM DESIGN"
:PRINT
170 PRINT"   ENTER YOUR SELECTIO
N (1-4)"
180 PRINT"   1. DRIVER PARAMETER
S"
190 PRINT"   2. VENTED BOX DESIG
N"
200 PRINT"   3. CLOSED BOX DESIG
N"
210 PRINT"   4. END
220 PRINT:INPUT"   ";P
230 ON P GOTO 240,640,1570,2300
240 CLS:PRINTTAB(6)" DRIVER PARA
METERS"
250 L$="N":R$="N"
260 INPUT" ENTER DRIVER NAME
          ";D$
270 INPUT" ENTER D.C. RESISTANCE
 OF VOICE  COIL ";RE
280 INPUT" ENTER FREE AIR RESONA
NCE";FS
290 INPUT" ENTER IMPEDANCE AT FR
EE AIR     RESONANCE";ZMAX
300 RO=ZMAX/RE
310 RF=SQR(RO)*RE
320 PRINT" ENTER FREQ. BELOW FRE
E AIR =   ";RF;"OHMS";
330 INPUT" ";F1
340 PRINT" ENTER FREQ. ABOVE FRE
E AIR      =";RF;"OHMS";
350 INPUT" ";F2
360 QMS=FS*SQR(RO)/(F2-F1)
370 QES=QMS/(RO-1)
380 QTS=QMS*QES/(QMS+QES)
390 INPUT" ENTER TEST BOX VOLUME
 IN         CUBIC FEET ";TVB
400 INPUT" ENTER DRIVER RESONANC
E IN TEST  BOX ";TFS
410 VAS=TVB*(1.149*(((TFS/FS)^2)
-1))
420 CLS:PRINTTAB(7)"DRIVER PARAM
ETERS":PRINT:PRINT
430 PRINT" ";D$
440 PRINT" VOICE COIL RESISTANCE
(RE) =    ";RE;" OHMS"
450 PRINT" FREE-AIR RESONANCE(FS
) =       ";FS;" HZ"
460 PRINT" QMS = ";QMS
470 PRINT" QES = ";QES
480 PRINT" QTS = ";QTS
490 PRINT" VAS = ";VAS
500 PRINT:INPUT" LINE PRINTER OU
TPUT (Y/N) ";L$
510 IFL$<>"Y"THEN610
520 POKE150,BR:' BAUD RATE POKE.
530 PRINT#-2,CHR$(27);CHR$(15);T
AB(12)"DRIVER PARAMETERS":PRINT#
-2,CHR$(27);CHR$(14);CHR$(13)
540 PRINT#-2:PRINT#-2," ";D$
550 PRINT#-2," VOICE COIL RESIST
ANCE (RE) =";RE;" OHMS"
560 PRINT#-2," FREE-AIR RESONANC
E (FS) =";FS;" HZ"
570 PRINT#-2," QMS = ";QMS
580 PRINT#-2," QES = ";QES
590 PRINT#-2," QTS = ";QTS
600 PRINT#-2," VAS = ";VAS
610 INPUT" ANOTHER DRIVER (Y/N)
";R$
620 IFR$="Y"THEN240
630 GOTO160
640 CLS:PRINTTAB(7)"VENTED BOX D
ESIGN":PRINT
650 L$="N"
```

```
660 INPUT" DRIVER NAME ";D$
670 INPUT" ENTER QTS";QTS
680 INPUT" ENTER VAS";VAS
690 INPUT" ENTER FS";FS
700 VB=15*((QTS^2.87)*VAS)
710 FB=.42*((QTS^-.9)*FS)
720 FH=.26*((QTS^-1.4)*FS)
730 CLS:R$="N"
740 PRINTTAB(9)"B4-ALIGNMENT":PR
INT
750 PRINT" VB = ";VB
760 PRINT" FB = ";FB;" HZ"
770 PRINT" F3 = ";FH;" HZ"
780 H=0
790 PRINT:INPUT" CHANGE BOX SIZE
 (Y/N) ";R$
800 IFR$<>"Y"THEN850
810 PRINT:INPUT" ENTER NEW VB ";
VB
820 FB=FS*((VAS/VB)^.32)
830 FH=FS*(SQR(VAS/VB))
840 H=20*(LOG(2.6*(QTS*((VAS/VB)
^.35)))/LOG(10))
850 CLS:PRINTTAB(7)"VENTED BOX D
ESIGN":PRINT
860 B$=" VENT":R$="N":L$="N"
870 PRINT" ";D$
880 PRINT" VB = ";VB
890 PRINT" FB = ";FB;" HZ"
900 PRINT" F3 = ";FH;" HZ"
910 PRINT" PEAK OR DIP IN RESPON
SE        = ";H;" DB"
920 PRINT:INPUT" CHANGE BOX SIZE
 (Y/N) ";R$
930 IFR$="Y"THEN810
940 INPUT" LINE PRINTER OUTPUT (
Y/N) ";L$
950 IFL$<>"Y"THEN1020
960 PRINT#-2,CHR$(27);CHR$(15);T
AB(10)"VENTED BOX DESIGN";CHR$(2
7);CHR$(14);CHR$(13)
970 PRINT#-2," ";D$
980 PRINT#-2," VB = ";VB
990 PRINT#-2," FB = ";FB;" HZ"
1000 PRINT#-2," F3 = ";FH;" HZ"
1010 PRINT#-2," PEAK OR DIP IN R
ESPONSE = ";H;" DB"
1020 INPUT" VENTED BOX DESIGN GR
APG          (Y/N) ";K$
1030 IFK$="Y" THEN1150
1040 INPUT" CUSTOM DESIGN (Y/N)
";Y$
1050 IFY$<>"Y"THEN1520
1060 CLS:PRINTTAB(4)"CUSTOM VENT
ED BOX DESIGN"
1070 B$=" ":R$="N"
1080 PRINT:INPUT" CHANGE VB (Y/N
) ";R$
1090 IFR$<>"Y"THEN1110
1100 INPUT" ENTER NEW VB ";VB
1110 R$="N"
1120 INPUT" CHANGE BOX TUNING (Y
/N) ";R$
1130 IFR$<>"Y" THEN1150
1140 INPUT" ENTER NEW FB ";FB
1150 CLS:PRINT:PRINT" CALCULATIO
NS IN PROGRESS"
1160 A=(FB^2)/(FS^2)
1170 B=A/QTS+(FB/(7*FS))
1180 C=1+A+(FB/(7*FS*QTS))+(VAS/
VB)
1190 D=1/QTS+(FB/(7*FS))
1200 FORF=20TO200STEP5
1210 F9=F/FS:F5=F9^2
1220 F4=F9^4:F3=F9^3
1230 F6=(F4-C*F5+A)^2
1240 F7=(B*F9-D*F3)^2
1250 M(F)=20*(LOG(F4/((F6+F7)^.5
))/LOG(10))
1260 NEXT
1270 IFY$="Y" THEN 1300
1280 IFK$= "Y" THEN GOTO1510
1290 IFB$=" VENT"THEN850
1300 CLS:PRINTTAB(4)"CUSTOM VENT
ED BOX DESIGN"
1310 R$="N"
1320 PRINT:PRINT" ";D$
1330 PRINT" VB = ";VB
1340 PRINT" FB = ";FB;" HZ"
1350 PRINT" QTS = ";QTS
1360 PRINT" FS = ";FS;" HZ"
1370 PRINT" VAS = ";VAS
1380 PRINT:INPUT" CHANGE VB OR F
B (Y/N) ";R$
1390 IFR$="Y"THEN1060
1400 INPUT" LINE PRINTER OUTPUT
(Y/N) ";L$
1410 IFL$<>"Y"THEN1490
1420 PRINT#-2,CHR$(27);CHR$(15);
TAB(8)"CUSTOM VENTED BOX DESIGN"
;CHR$(27);CHR$(14);CHR$(13)
1430 PRINT#-2," ";D$
1440 PRINT#-2," VB  = ";VB
1450 PRINT#-2," FB  = ";FB;" HZ"
1460 PRINT#-2," QTS = ";QTS
1470 PRINT#-2," FS  = ";FS;" HZ"
1480 PRINT#-2," VAS = ";VAS
1490 INPUT" CUSTOM VENTED BOX DE
SIGN GRAPH  (Y/N) ";G$
1500 IFG$<>"Y" THEN1520
1510 GOSUB2140
1520 INPUT" ANOTHER VENTED BOX D
ESIGN (Y/N) ";R$
1530 IFR$="Y"THEN640
1540 INPUT" ANOTHER CUSTOM VENTE
D BOX       DESIGN (Y/N) ";K$
1550 IFK$="Y" THEN1060
1560 GOTO160
1570 CLS:PRINTTAB(7)"CLOSED BOX
DESIGN":PRINT
1580 L$="N":R$="N"
1590 INPUT" DRIVER NAME ";D$
1600 INPUT" ENTER QTS ";QS
1610 INPUT" ENTER VAS ";VAS
1620 INPUT" ENTER FS ";FS
1630 INPUT" ENTER VB ";VB
1640 A=VAS/VB
1650 FC=FS*(SQR(A+1))
1660 QTC=(FC*QS)/FS
1670 F3=FC*SQR((((1/QTC^2)-2)+SQ
R((((1/QTC^2)-2)^2)+4))/2)
1680 CLS:PRINTTAB(7)"CLOSED BOX
DESIGN"
1690 R$="N"
1700 PRINT:PRINT" ";D$
1710 PRINT" F3 = ";F3;" HZ"
1720 PRINT" QTC = ";QTC
1730 PRINT" VB = ";VB
1740 PRINT:INPUT" CHANGE BOX SIZ
E (Y/N) ";R$
1750 IFR$<>"Y" THEN1780
1760 PRINT:INPUT" ENTER NEW VB "
;VB
1770 GOTO1640
1780 CLS:PRINTTAB(8)"CLOSED BOX
DESIGN"
1790 R$="N":L$="N"
1800 PRINT:PRINT" ";D$
1810 PRINT" QTS = ";QS
1820 PRINT" QTC = ";QTC
1830 PRINT" VAS = ";VAS
1840 PRINT" VB = ";VB
1850 PRINT" FS = ";FS;" HZ"
1860 PRINT" ALPHA = ";A
1870 PRINT" FC = ";FC;" HZ"
1880 PRINT" F3 = ";F3;" HZ"
1890 PRINT:INPUT" LINE PRINT OUT
 (Y/N) ";L$
1900 IFL$<>"Y" THEN2000
1910 PRINT#-2,CHR$(27);CHR$(15);
TAB(11);"CLOSED BOX DESIGN";CHR$
(27);CHR$(14);CHR$(13)
1920 PRINT#-2," QTS = ";QS
1930 PRINT#-2," QTC = ";QTC
1940 PRINT#-2," VAS = ";VAS
1950 PRINT#-2," VB = ";VB
1960 PRINT#-2," FS = ";FS;" HZ"
1970 PRINT#-2," ALPHA = ";A
1980 PRINT#-2," FC = ";FC;" HZ"
1990 PRINT#-2,"F3=    ";F3;" HZ"
2000 PRINT:INPUT" CLOSED BOX RES
PONCE GRAPH (Y/N) ";R$
2010 IFR$<>"Y" THEN2090
2020 CLS:PRINTTAB(4)"CALCULATION
S IN PROGRESS"
2030 FORF=20TO200 STEP5
2040 FH=F/FC:FQ=FH^2:MAG=FQ/(SQR
(((FQ-1)^2)+((FH/QTC)^2)))
2050 M(F)=20*(LOG(MAG)/LOG(10))
2060 NEXT
2070 GOSUB2140
2080 '
2090 R$="N"
2100 INPUT" ANOTHER CLOSED BOX D
ESIGN (Y/N) ";R$
2110 IFR$="Y"THEN1570
2120 GOTO160
2130 '
2140 DATA20,30,40,50,60,70,80,90
,100,120,140,160,180,200
2150 DATA200,150,100,80,60,40,30
,20
2160 '
2170 POKE149,0:POKE150,BR:PRINT#
-2,CHR$(27);CHR$(15)
2180 PRINT#-2,TAB(9)"-40"TAB(19)
"-30"TAB(29)"-20"TAB(39)"-10"TAB
(50)"0"TAB(59)"+10"TAB(69)"Db)
2190 PRINT#-2,TAB(10)"+"TAB(20)"
+"TAB(30)"+"TAB(40)"+"TAB(50)"+"
TAB(60)"+":PRINT#-2,"HZ.";CHR$(1
3)
2200 FORI=1TO22
2210 READF
2220 PRINT#-2,F;TAB(9)"I";
2230 IFM(F)<-39THEN2260
2240 PRINT#-2,TAB(50+M(F))"*"
2250 GOTO2270
2260 PRINT#-2,""
2270 NEXT
2280 PRINT@482,;:INPUT" PRESS EN
TER TO CONTINUE ";R$
2290 RESTORE:RETURN
2300 END
```

HALLELUJAH CHORUS

CoCo 1/2 + Orchestra–90CC MUSIC

by Harvey Smith

PRESENTING ONE OF THE many music competition entrants. The title of this fine work of art is called the "Halleluyah Chorus" and is just great!

This work of art can be found on the end of the CoCoOz Tape this month.

The Listing:

```
/HALLELUJAH CHORUS
/GEO. F. HANDEL
JASF8080000F/ORGAN - LOUD
JBSF80800008/ORGAN - MID
JCSF80800001/ORGAN - SOFT
JDSFDA822028/DIAPASON -MID
JESFDA822021/DIAPASON -SOFT
K2#
NQ=B0
V1YBV2YBV3YDV4YD
M *V1Q.1I565Q$
@V2I432*I111$$
@V3I65463642
V4IDCBDADB9
M *Q8I-2343$5
V2Q1I-2-2-3-2$1
@V3I65465 6$4
V4IDCBDCD$B
M *I43Q2I1Q567
@V2I32Q2I4*342
@V3I56Q9I6$Q$
V4ICDQ9ID$Q$
M *Q.8I565Q$
V2I3345 43Q$
V3H1I11Q$
@V4I665436Q$
M *Q.8I565I$S88
V2I3345 43$S55
V3H.1Q1
@V4I665436$4
M *I88$S88I88$S88
V2I43$S55I43$S55
@V3Q1H1Q1
V4I36$436$4
M *I78878543
V2I43Q2I3101
V3I01101@432
V4I56Q2I6456
M *Q.9I5A9Q$
V2Q.5I555Q$
V3Q.2I232Q$
V4@Q.2I224Q$
M *Q.9I5A9$S99
V2Q.5I555$S55
V3Q.2I232$2
V4@H2Q22
M *IA9$S99IA9$S99
V2I55$S55I55$S55
V3I32$232$2
V4@H2Q22
M *IA9$S99IA9$S99
V2I55$S55I55$S55
V3I32$232$2
V4@Q2H2Q2
M *IA9Q8I7S99IA9
V2Q.5I4#5S55I55
V3I32Q1@I2$$$
V4Q.2I3#2$$$
M *Q.5I5Q67
V2@Q.2I2Q10
M *I81H8Q7
V2I1-6H1Q0
M *Q.6I6Q5I$S55
V2@Q.1I1Q2I$*S22
V3@Q$$I$2
M *I87$S55I87$S55
V2I32$S22I32$S22
V3I10$-210$-2
V4@I22$222$2
M *I87$S55I87Q$
V2I32$S22I32Q$
V3I1S00I0-210Q$
V4@I2S22I2222Q$
M *Q.1I1Q23
V2@Q.6I6Q54
M *I4-3H4Q3
V2@I3AH3Q4
M *Q.2I2Q1I$SCC
V2Q$$$I$S88
V3Q$$$I$5
V4Q$$$I$1

M *IDC$SCCIDC$SCC
V2I88$S88I88$S88
V3I65$565$5
V4I11$111$1
M *IDC$SCCIDCQ$
V2I88$S88I88Q$
V3I6S551 5565Q$
V4I1S1111111Q$
M *Q.8I8Q9A
V2H35
V3@I+16$+102$+1
M *IB4HBQA
V2I84Q6H5
V3@I11$-202-1
M *Q.9I9Q8I$8
V2I4255Q55
V3Q1001-61-6
V4@+10246+16
M *I75$7Q89
V2Q2105$S43I2S54
V3@Q.2I2Q10
V4Q.9I9Q87
M *Q5I$S87I6259
V2Q3I$S65I4#255
V3I1-6H1Q0
V4@I6DH6Q7
M *I878S98I7797
V2Q54#I5$Q2
V3@H1Q2I$2
V4H8Q9$
M *I5S55I85$797
V2I$S11I33Q25
V3@I86$+10202
M *I$5A8$785
V2I$3A8Q23
V3Q.1I10-2$1
V4Q.6I679$6
M *I$699$788
V2I4-3H4Q3
V3@I13$522$1
V4I8A$C77$8
M *Q8I77Q88
V2Q.2I2Q11
V3@I35I.2S3Q4I$6
V4IACI.9SAQBI$6
M *Q88I8CA8
V2Q1I123853
V3@I1310Q.+1I+1
V4I8A87Q.6I6
M *IBBDDQC
V2I4466Q5

V3@I1133Q-1
V4I88AAQ6
V1YCV2YCV3YEV4YE
M *Q5543I.2S1
V2@Q221*1I.0S1
V1YBV2YBV3YBV4YB
M *H.1Q5
V2H.1Q-2
V3@H.2Q4
V4H.4Q6
V1YAV2YAV3YAV4YA
M *Q543I.2S1H.1
V2@Q21-12H.2
V3@Q5323H.4
V4Q7899H.6
V1YAV2YAV3YDV4YD
M *QC;C;B;AI.9S8
V2Q5;5;6;55
V3Q3;2;4;3I.2S1
V4Q1;0;@1;22
M *Q.8I9QA;B#;
V2Q.5I4Q5;6;
V3Q.1I2Q3;4#;
V4@Q.2I3Q2;1;
M *Q.C;IC;QD;E;
V2Q.7;I8;Q8;9;
V3Q.5;I5;Q6;7;
V4Q.0;I1;Q1;2;
M *Q.F
V2Q.A
V3Q.8
V4Q.3
V1YBV2YBV3YBV4YB
M @Q163I45
M *H$Q$1
V2H5Q6$
V3YDV4YD
M *Q503-2
V3Q$I$2-16$4
M *Q1I.0S-1Q0@I.1S2
V3@I13#Q2I$565
M *I-2$Q583
V2@I2Q0+1+3
V3@I7232$2Q+1
M *Q614I32
V2Q6I$-120Q1
V3Q1$$@I$3
M *Q3I.2S1Q18
V2Q1013
V3@I29$46$+1
M *QC7A5
V2Q5230
V3I02$0+16$4
M *Q8I76Q7I.6S5
V2I1223Q22
V3@I13#265C$5
M *H5QCC
V2Q20CC
V3@Q79*55
M *Q.CSCCIFE$SCC
V2Q.CS99IA9$S99
V3W5
V4Q$I$01-2$0
M *IFE$SCCIFE$SCC
V2IA9$S99IA9$S99
V3W5
V4I1@2$0+12$0
M *IFEI.CSCQCI.CSC
V2IA9Q$H$
V3Q5I.5S5Q5I.5S5
V4I1-2Q$H$
M *Q.CSCCIFE$SCC
V2H$IA9$S99
V3W5
V4Q$I$01-2$0
M *IFE$SCCIFE$SCC
V2IA9$S99IA9$S99
V3W5
V4I1-2$01-2$0
M *IFESCBA9Q88
V2IA9Q$$$
V3Q5S5432Q11
V4I1-2Q$$$
M
V2W8
V3Q$I$565$5
```

Continued overleaf

SPEAKER / HEADPHONE SELECTOR

A recent bout of the flu gave me the opportunuity to try out an idea that I had been thinking about for quite some time.

Earlier this year, I had an Audio/Video Amp fitted to my CoCo for use with an BMC monitor and although I don't play a large number of games, I thought that the Audio one was a good choice. I did notice on a few occasions that the little speaker left a lot to be desired when, say an adventure like Dungeons of Daggorath needs the sound as much as the picture. In this game probably more so because it indicates the impending arrival of a nasty that can KILL with one swift blow. I thus found out that the beating heart drowned out all other sounds.

With that in mind I set about to make an adaptor to take the fitting of standard stereo headphones (keeping in mind that I am not an electronics expert).

The adaptations are as follows

First, disconnect both leads going to the speaker taking one to the outside connector of a two position switch. Take the other to a female stereo plug socket.

Now add another wire to the first section disconnected from the speaker and connect it to the centre position of the switch.

Next, connect the remaining speaker terminal up to the second Amp wire previously disconnected and finally connect the remaining switch terminal to the second connector of the headphones socket.

Now you have an adaptor to select between sound from the inbuilt speaker or headphones.

For fear of drilling humungous holes in the CoCo's case, I encased the entire setup in one of those little tin boxes that sits not too untidily nearby.

Like I said, I'm no electronics expert, so if any of this procedure is the long way round, I would like to hear about it.

(The device really adds realism to the shoot em up type arcade games where you can now feel like you're actually in the cockpit).

Jeff Wetzig
12 Cromwell St
Wooloowin 4030

HALLELUJAH CHORUS

From previous page

```
V4W1
M *IDC$SCCIDC$SCC
V2W8
V3I65$565$5
V4W1
M *IDCQ$$$
V2Q8I.8S8Q9I.9S9
V3165Q$$$
V4Q1I.1S1Q2I.2S2
M *Q$I$SDDIED$SDD
V2W9
V3Q$I$676$6
V4W2
M *IED$SDDIED$SDD
V2W9
V3I76$676$6
V4W2
M *IEDQ$$$
V2Q99AA
V3I76Q$$$
V4Q2233
M *Q$I$SEEIFE$SEE
V2WA
V3I87$787$7
V4W3
M *QFI.ASAQBB
V2QA8$4
V3Q8I.3S3Q4$
V4Q31$$
M *WB
V2H4Q66
V3H$4
V4H$@Q11
M *Q.BIBA9AB
V2Q.5I5Q5I55
V3Q.4I43234
V4@Q.2I2Q2I22
M *H9I$789
V2H5Q$$
V3H25
V4@H20
M *I5567$AB#C
V2H$Q$5
V3H3Q32
V4H0Q10
M *I878S98I7$QC
V2Q.54#I59CB%
V3H1I2$Q$
V4@H1I2$Q$
M *QFAD8
V2QAIA9Q8165
V3@I$2+101634
M *QBIA9QAI.9S8
V2Q4555
V3@Q5I43Q2I32
M *H8Q88
V2Q53H$
V3Q46+1+1
M *Q$I$SCCIDC$SCC
V2W8
V3Q$I$565$5
V4W1
M *IDC$8Q88
V2Q8$$$
V3I65Q$$$
V4Q1I$1Q11
M *Q$I$SCCIDC$SCC
V2W8
V3Q$I$565$5
V4W1
M *IDC$5Q58
V2Q8I$3Q33
V3I65$1Q11
V4Q1$$$
M *QA885
V2Q3541
V3Q111-2
V4@Q1236
M *Q7887
V2Q23H2
V3Q0110
V4@Q54H5
M *Q8$CC
V2Q3$88
V3Q1$55
V4@Q4$*11
M *QFI$CQCC
V2QAI$8Q88
V3I8S34I55Q55
V4I3S12I31Q11
M *IF$$S67I8SABCBA9
V2IA$Q$$$
V3I8S34I5S43Q33
V4I3S12I3S42Q3-2
M *Q8A68
V2Q$565
V3Q3341
V4@Q1212
M *QBIA9H9
V2Q45H5
V3Q2I32H2
V4@Q32H2
M *Q$I$SCCIDC$SCC
V2W8
V3Q3$I565$5
V4Q2$$$
M *IDC$SCCIDC$SCC
V2W8
V3I65$565$5
V4W1
M *IDCQ$$F
V2Q8$$8
V3I65Q$$5
V4Q1$$-2
M *H.FQF
V2H.8Q8
V3H.4Q4
V4@H.1Q1
M *WFHF
V2W8H8
V3W5H5
V4@W2H2
```

SCREEN DUMPS with PLOTTERS

by George McLintock

FOR SOME TIME now I have been inclined towards the purchase of a colour printer, but put off by the cost of them for what you get. I recently spent some time experimenting with screen dumps to the CGP-115, and convinced myself that plotters actually produce a superior product. So I went out and purchased an A4 sized plotter, a Pixy 3. This has been available at a 'sell out' price for some time and appears to be excellent value. The only real problem with the CGP is its half size paper width.

I have developed two programs which take different approaches to screen dumps and both include a range of options to obtain maximum flexibility from the capabilities of the plotter.

Colours with a Plotter Single Lines

The BASIC source of colour is the ink in the pen, and which pen you use. The CGP has four colours; black, red, blue and green. The Pixy has these four colours plus brown, orange, rose and purple. In addition to these basic pen colours you can create others with overprinting, eg red over blue to get mauve.

As well as these standard colours, plotters also provide an option of offset overprinting. For example, the following are different ways of producing a mauved coloured line, all of which a different visual effect.

The initial line is (0,0)-(50,0) ie from the point 0,0 to the point 50,0.

The normal mauve is (0,0)-(50,0) in blue followed by (0,0)-(50,0) in red.

- blue over red gives a slightly different colour. A single offset mauve is (0,0)-(50,0) followed by (0,1)-(50,1) in red

Other variations are:
(0,0)-(50,0) in blue,
(0,1)-(50,1) in red,
(0,2)-(50,2) in blue.
... which is different to:
(0,0)-(50,0) in red,
(0,1)-(50,1) in blue,
(0,2)-(50,2) in red.

With its smaller step size the Pixy provides other effects because each line drawn is actually wider than its step size, so that its single offset line is actually a mixture of overprinting and offset printing, eg (0,0)-(0,50) in blue followed by (0,1)-(50,1) in red is different to (0,0)-(0,50) in blue followed by (0,2)-(50,2) in red.

There is no noticable gap between these last two lines, just no overprinting which gives a different visual effect.

Colour Blocks

With a plotter, a single block of colour on the screen can be transformed to any mixture of colours in the dump that you might think of.

For example, the following is an outline of the various ways that a single square of colour on the screen can be dumped to a plotter. The colour block on the screen is produced by LINE (0,0)-(50,50),,PSET,BF.

The simple dump is to draw a series of lines across the page to fill this area using a single pen, eg if the colour is red, then by drawing the following lines (0,0)-(50,0) in red followed by (0,1)-(50,1) in red followed by (0,2)-(50,2) in red, up to line 50.

A lighter red can be produced by skipping alternative lines, ie by only drawing every second line in the sequence above, eg (0,0)-(50,0) in red followed by (0,2)-(50,2) in red up to line 50.

To produce mauve by this procedure you then repeat these same lines by using the blue pen.

Offset overprinting allows for a greater variations in colours. For example a single offset mauve is achieved with the following lines (0,0)-(50,0) in red followed by (0,1)-(50,1) in blue followed by (0,2)-(50,2) in red followed by (0,3)-(50,3) in blue with the sequence repeated up till line 50.

Again skipping alternative lines will give a different effect. To change the references a bit, line 0 is red, line 1 is white (or paper colour), line 2 is blue, line 3 is white, line 4 is red, etc up till line 50.

With this approach, any combination of colours becomes possible, with any number of skips between each line.

Stripes can be produced in various widths, ie line 0 to 3 in red, lines 4 to 7 as white, line 8 to 11 as red (or blue).

With additional program control (as with Colour-Dump), it is possible to treat red on the screen in the top left corner different to red in the top right corner.

Across or Down or Both

The other option with plotters is that you can draw the lines across the page or down the page. For example the discussion above was based on drawing lines across the page. However the simple dump (first example) will work just as well by drawing lines down the page instead of across it, ie to use (0,0)-(0,50) in red followed by (1,0)-(1,50) in red followed by (2,0)-(2,50) in red up till line 50.

All of the variations for across the page have a similar effect if drawn on the page, but the visual effect of each option is different.

The next option of course is to draw the lines in both directions. With solid colours this produces a similar effect to overprinting (but different). But with white (skipped lines) you can produce another range of patterns, eg stripes become checks and tartan effects can be obtained.

Obtaining these Effects

The screen dumps submitted obtain these effects by repeating the same dump several times with different coloured pens and different skip factors (and or offsets).

There are two skip factors to be specified. The number of lines to skip before drawing any lines, and the number of lines to skip between each line drawn.

For example, alternating red and blue lines are obtained by drawing the first dump with the red pen, skip zero at the start and skip one between each line drawn. The second dump is with the blue pen, skip one at the start and skip one between each line drawn.

Stripes are obtained in a similar fashion. The example used above can be obtained with three repeats. The first skips zero at the start and skip 6 between each line drawn. The second skips one at the start and 6 between, while the third is skip 2 at the start and 6 between.

Alternating red and blue stripes require 6 repeats the first three with the red pen, skip zero at start and 12 between, one at start and 12 between and skip 2 at start and 12 between, the next three repeats are with the blue pen, skip 6 at start and 12 between, skip 7 at start and 12 between and 8 at start and 12 between.

For single lines, LINE DUMP also provides an additional option to move the whole dump on the paper between each repeat. This is called an off-set. If the second dump is off-set one or two dots from the previous one, then you obtain the effects described for single lines.

The programs are written to allow you to set it up for any number of repeats before the dump actualy starts. When the setup is complete, the plotter will work away to the end without any further action on your part.

Of the options available; size, rotation, pmode, position etc are set only once for the whole dump. The other options of pen colour, screen colour (PMODE 3), direction of lines, skip factors etc are set seperately for each repeat of the dump. Other options are also available which are specific to each program.

Note that the first repeat is numbered zero, and zero should be entered at the menu for a single pen conventional dump.

With this approach, virtually any style of screen dump can be obtained, and you can experiment with any combination of options that might occur to you.

Aspects Common to both Programs Menu

Both programs contain a number is options which are set up through a menu arrangement. The programs contain instructions on how to use the menu.

For all menus the left arrow key will provide more information about each option (a form of help key). The right arrow key allows you to access other options where these might apply.

The menus are set up for conventional data entry, and when moving down it, the cursor will skip over some cells which are not normally required. To access these options you need to move back to them with the up arrow key.

The basic setup arrangement is designed mainly to experiment with the different options on different screen pictures. It has a number of weakinesses when you come to more regular useage of the program for standard screen dumps.

I have added a couple of features to overcome the more obvious problems, but a better and neater menu setup arrangement could be devised. The major problem (which is common with these sort of exercises), is that with so much of the logic (and code) already in existance, a major change can require a complete re-write of the whole thing.

While I may get round to doing this eventually, I prefer to use it as it is for awhile to develop a better understanding of what I actually want to do. The existing version is the result of two major revisions without a complete re-write (ie re-using existing code where possible), and the code as it now exists certainly reflects this. A complete re-write is required for the next major revision.

The nature of the problem is indicated by the setup of predetermined colour/patterns (discussed later). When you actually set up some examples (as done for Colour Dump), it suggests that you should set up the patterns independantly of the pen colour. But then how far do you go, the same logic would apply to the skip factors within a pattern. Therefore, it may be preferable to re-structure the whole approach to how to specify the repeats. At this stage I prefer to wait and see, and use it as it is.

A possibility of couse is that someone else might like the programs enough the re-write this aspect themselves - send me a copy. I have included a description of the parameters actually required for the dumps. Providing these are set before you enter the dump, any setup procedure will work.

Standard Dumps

The menu arrangement can become tediuos if all you want is a normal standard screen dump. I have therefore included an option with each program to allow this (for the CGP plotter). The parameters required are set up in lines 94 to 96 for Line Dump and lines 83 to 85 for Colour Dump.

You can re-define these if you wish by changing these lines. Or you can add other standard dumps by following the same general logic.

Pre-defined Patterns

After experimenting for a while, you may find that you like certain colour/pattern combinations. If these require a number of repeats to obtain, it can become tedious to have to re-entr them everytime you want that combination in a dump.

I have included a facility with Colour Dump which allows you to pre-define up to 26 colour/patterns which can be

generated automatically by using letters 'A' to 'Z' for the pen colour in the menu. The use of this facility is described in the section specific tro Colour Dump.

A similar procedure could be applied to Line Dump, but I have not attempted it at this stage. Line Dump uses more memory, and its use of memory would have to be tidied up a bit to fit this procedure in.

Save and Restore the Dump Parameters

I feel it would be a desireable option to be able to save and restore the dump parameters for a particular picture, particularly for the more complicated ones. If you try a few different effects, and get one that you like, it would certainly be convenient to be able to reuse those parameters without having to re-enter them.

This option is not included with either program. As a partial solution, I have included an option to print out the parameter values used at the end of the dump.

The values printed could be saved, and re-loaded into another version of the program which by-passed the menu setup completely and re-loaded these values and goes directly to the dump.

Memory Requirements

As set up, both programs require a 32K machine, but most of this space is required for the menu and setup operations. The actual screen dumps themselves are quite small and will fit into a 16K machine if the menu is deleted and the parameters required setup as part of the basic code.

I have included a list of the parameters and program line numbers required for each dump.

Both programs will work without modification with either disk or tape systems.

Machine-Language Routines

Both programs contain machine language (ML) routines. The programs are set up to include the ML routines at the end of the basic program following the procedure described in 'Charlies Machine', Australian Rainbow, Jan 83, and as used for Art Liner.

When you run the program the first time, the ML code will be incorporated within the basic program. You can (C)SAVE this version and use it in this format, or simply RUN the program again to get the dumps.

With Line Dump, the first time you run it after incorporting the ML routine, you may get an ?UL error in line 94. If you do, simply RUN it again and the error should go away.

Before saving it you should change the first line to GOTO the line number in line 650, and delete lines 490 and 650. The program was set up in this way so that it will continue to work OK if you RUN the version with the DATA statements in it a number of times without re-loading the program.

The machine code is relocatable and can be POKE'd into any other area of memory you might prefer. If you do locate it elsewhere, remember to change the value to 'M' in the program to correspond to the start of the routine.

Size of Dumps and Rotation

This depends on your plotter. The Pixy 3 has 2450 x 1800 dot positions (other A4 plotters would be similar). This gives a maximum of 9 times the size of the screen with normal rotation, and six times the size if rotated 90 degrees. This appears sufficient to not require any special treatement for size and rotation.

The only special option for the Pixy is to allow the dump to be positioned anywhere on the pge. The default position is the top left corner, but this can be altered to anywhere on the page.

The CGP provides only 480 dots across the page and 999 down it in any one quadrant. These screen dumps position it in the second quadrant (+x,-y with possible extension to Y+), and some special treatement of size is provided. (If comparingdot widths, the step size of the CGP is twice that of the pixy.)

A single size dump is drawn with normal rotation in the center of the page, although this can be altered in the menu. A double sized dump is drawn with 90 degree rotation, again in the center of the page (which again can be altered).

Options are also provided for 3x and 4x sizes for screens dumps, with either rotation (compared with the Pixy, these are equilivant to 6x and 8x size).

With these sizes, not all the screen will fit on the paper. But there are occasions when the whole screen is not required, anyway, eg Cooch, CoCo Feb/Mar 86 will fit on the paper at 3x size and will fit a 4x size if either Pew or the caption is excluded.

The menu set-up provides for these options and allows for the area of the screen to be dumped to be selected. Use the right arrow at the size option to access this facility.

Plotter Controls and Other Plotters

The programs are organised to be relatively independant of the type of plotter being used. The start and end points for each line drawn is calculated with reference to the CoCo screen and are only translated to plotter reference points at the time that the commands are actually sent to the plotter.

The code for doing this is the same in both programs (line 9 to 13 in Colour Dump, and lines 15 to 18 in line dump). The code invloved is effectively a single line of basic code for each plotter type, so that it should be relatively easy to convert these programs to work with any other plotter.

A possible complication is the fixed value reference point used to reverse the mirror effect that occurs with the Pixy co-ordinate system, and with the 90 degree rotation on the CGP. However these are specific to the particular plotter and have to be allowed for anyway.

The setting up of the fixed reference values is part of the dump size routines and should be relatively easy to find in the code, as should be the selection of the pen colour.

○

Hint...

Edit command - extra features

Q - Quit the current edit session without making any changes (Returns to an OK prompt).

A - Abort the current edit -remains in edit mode and re-displays the line as it was originally.

E - End the current edit session (same effect as pressing ENTER).

NOTE: These EDIT commands are in addition to those explained in the Tandy handbooks.

Kevin Gowan

Across or Down or Both

The other option with plotters is that you can draw the lines across the page or down the page. For example the discussion above was based on drawing lines across the page. However the simple dump (first example) will work just as well by drawing lines down the page instead of across it, ie to use (0,0)-(0,50) in red followed by (1,0)-(1,50) in red followed by (2,0)-(2,50) in red up till line 50.

All of the variations for across the page have a similar effect if drawn on the page, but the visual effect of each option is different.

The next option of course is to draw the lines in both directions. With solid colours this produces a similar effect to overprinting (but different). But with white (skipped lines) you can produce another range of patterns, eg stripes become checks and tartan effects can be obtained.

Obtaining these Effects

The screen dumps submitted obtain these effects by repeating the same dump several times with different coloured pens and different skip factors (and or offsets).

There are two skip factors to be specified. The number of lines to skip before drawing any lines, and the number of lines to skip between each line drawn.

For example, alternating red and blue lines are obtained by drawing the first dump with the red pen, skip zero at the start and skip one between each line drawn. The second dump is with the blue pen, skip one at the start and skip one between each line drawn.

Stripes are obtained in a similar fashion. The example used above can be obtained with three repeats. The first skips zero at the start and skip 6 between each line drawn. The second skips one at the start and 6 between, while the third is skip 2 at the start and 6 between.

Alternating red and blue stripes require 6 repeats the first three with the red pen, skip zero at start and 12 between, one at start and 12 between and skip 2 at start and 12 between, the next three repeats are with the blue pen, skip 6 at start and 12 between, skip 7 at start and 12 between and 8 at start and 12 between.

For single lines, LINE DUMP also provides an additional option to move the whole dump on the paper between each repeat. This is called an off-set. If the second dump is off-set one or two dots from the previous one, then you obtain the effects described for single lines.

The programs are written to allow you to set it up for any number of repeats before the dump actualy starts. When the setup is complete, the plotter will work away to the end without any further action on your part.

Of the options available; size, rotation, pmode, position etc are set only once for the whole dump. The other options of pen colour, screen colour (PMODE 3), direction of lines, skip factors etc are set seperately for each repeat of the dump. Other options are also available which are specific to each program.

Note that the first repeat is numbered zero, and zero should be entered at the menu for a single pen conventional dump.

With this approach, virtually any style of screen dump can be obtained, and you can experiment with any combination of options that might occur to you.

Aspects Common to both Programs Menu

Both programs contain a number is options which are set up through a menu arrangement. The programs contain instructions on how to use the menu.

For all menus the left arrow key will provide more information about each option (a form of help key). The right arrow key allows you to access other options where these might apply.

The menus are set up for conventional data entry, and when moving down it, the cursor will skip over some cells which are not normally required. To access these options you need to move back to them with the up arrow key.

The basic setup arrangement is designed mainly to experiment with the different options on different screen pictures. It has a number of weakinesses when you come to more regular useage of the program for standard screen dumps.

I have added a couple of features to overcome the more obvious problems, but a better and neater menu setup arrangement could be devised. The major problem (which is common with these sort of exercises), is that with so much of the logic (and code) already in existance, a major change can require a complete re-write of the whole thing.

While I may get round to doing this eventually, I prefer to use it as it is for awhile to develop a better understanding of what I actually want to do. The existing version is the result of two major revisions without a complete re-write (ie re-using existing code where possible), and the code as it now exists certainly reflects this. A complete re-write is required for the next major revision.

The nature of the problem is indicated by the setup of predetermined colour/patterns (discussed later). When you actually set up some examples (as done for Colour Dump), it suggests that you should set up the patterns independantly of the pen colour. But then how far do you go, the same logic would apply to the skip factors within a pattern. Therefore, it may be preferable to re-structure the whole approach to how to specify the repeats. At this stage I prefer to wait and see, and use it as it is.

A possibility of couse is that someone else might like the programs enough the re-write this aspect themselves - send me a copy. I have included a description of the parameters actually required for the dumps. Providing these are set before you enter the dump, any setup procedure will work.

Standard Dumps

The menu arrangement can become tediuos if all you want is a normal standard screen dump. I have therefore included an option with each program to allow this (for the CGP plotter). The parameters required are set up in lines 94 to 96 for Line Dump and lines 83 to 85 for Colour Dump.

You can re-define these if you wish by changing these lines. Or you can add other standard dumps by following the same general logic.

Pre-defined Patterns

After experimenting for a while, you may find that you like certain colour/pattern combinations. If these require a number of repeats to obtain, it can become tedious to have to re-entr them everytime you want that combination in a dump.

I have included a facility with Colour Dump which allows you to pre-define up to 26 colour/patterns which can be

generated automatically by using letters 'A' to 'Z' for the pen colour in the menu. The use of this facility is described in the section specific tro Colour Dump.

A similar procedure could be applied to Line Dump, but I have not attempted it at this stage. Line Dump uses more memory, and its use of memory would have to be tidied up a bit to fit this procedure in.

Save and Restore the Dump Parameters

I feel it would be a desireable option to be able to save and restore the dump parameters for a particular picture, particularly for the more complicated ones. If you try a few different effects, and get one that you like, it would certainly be convenient to be able to reuse those parameters without having to re-enter them.

This option is not included with either program. As a partial solution, I have included an option to print out the parameter values used at the end of the dump.

The values printed could be saved, and re-loaded into another version of the program which by-passed the menu setup completely and re-loaded these values and goes directly to the dump.

Memory Requirements

As set up, both programs require a 32K machine, but most of this space is required for the menu and setup operations. The actual screen dumps themselves are quite small and will fit into a 16K machine if the menu is deleted and the parameters required setup as part of the basic code.

I have included a list of the parameters and program line numbers required for each dump.

Both programs will work without modification with either disk or tape systems.

Machine–Language Routines

Both programs contain machine language (ML) routines. The programs are set up to include the ML routines at the end of the basic program following the procedure described in 'Charlies Machine', Australian Rainbow, Jan 83, and as used for Art Liner.

When you run the program the first time, the ML code will be incorporated within the basic program. You can (C)SAVE this version and use it in this format, or simply RUN the program again to get the dumps.

With Line Dump, the first time you run it after incorporting the ML routine, you may get an ?UL error in line 94. If you do, simply RUN it again and the error should go away.

Before saving it you should change the first line to GOTO the line number in line 650, and delete lines 490 and 650. The program was set up in this way so that it will continue to work OK if you RUN the version with the DATA statements in it a number of times without re-loading the program.

The machine code is relocatable and can be POKE'd into any other area of memory you might prefer. If you do locate it elsewhere, remember to change the value to 'M' in the program to correspond to the start of the routine.

Size of Dumps and Rotation

This depends on your plotter. The Pixy 3 has 2450 x 1800 dot positions (other A4 plotters would be similar). This gives a maximum of 9 times the size of the screen with normal rotation, and six times the size if rotated 90 degrees. This appears sufficient to not require any special treatement for size and rotation.

The only special option for the Pixy is to allow the dump to be positioned anywhere on the pge. The default position is the top left corner, but this can be altered to anywhere on the page.

The CGP provides only 480 dots across the page and 999 down it in any one quadrant. These screen dumps position it in the second quadrant (+x,-y with possible extension to Y+), and some special treatement of size is provided. (If comparingdot widths, the step size of the CGP is twice that of the pixy.)

A single size dump is drawn with normal rotation in the center of the page, although this can be altered in the menu. A double sized dump is drawn with 90 degree rotation, again in the center of the page (which again can be altered).

Options are also provided for 3x and 4x sizes for screens dumps, with either rotation (compared with the Pixy, these are equilivant to 6x and 8x size).

With these sizes, not all the screen will fit on the paper. But there are occasions when the whole screen is not required, anyway, eg Cooch, CoCo Feb/Mar 86 will fit on the paper at 3x size and will fit a 4x size if either Pew or the caption is excluded.

The menu set-up provides for these options and allows for the area of the screen to be dumped to be selected. Use the right arrow at the size option to access this facility.

Plotter Controls and Other Plotters

The programs are organised to be relatively independant of the type of plotter being used. The start and end points for each line drawn is calculated with reference to the CoCo screen and are only translated to plotter reference points at the time that the commands are actually sent to the plotter.

The code for doing this is the same in both programs (line 9 to 13 in Colour Dump, and lines 15 to 18 in line dump). The code invloved is effectively a single line of basic code for each plotter type, so that it should be relatively easy to convert these programs to work with any other plotter.

A possible complication is the fixed value reference point used to reverse the mirror effect that occurs with the Pixy co-ordinate system, and with the 90 degree rotation on the CGP. However these are specific to the particular plotter and have to be allowed for anyway.

The setting up of the fixed reference values is part of the dump size routines and should be relatively easy to find in the code, as should be the selection of the pen colour.

○

Hint...

Edit command - extra features

Q - Quit the current edit session without making any changes (Returns to an OK prompt).

A - Abort the current edit -remains in edit mode and re-displays the line as it was originally.

E - End the current edit session (same effect as pressing ENTER).

NOTE: These EDIT commands are in addition to those explained in the Tandy handbooks.

Kevin Gowan

You could end up with egg on your face

TOAD in the HOLE

16K ECB
GAME

by John Day

TOAD IN THE HOLE is the first program I wrote in ECB, soon after I got the CoCo. It is a typically Queensland game, if the media are anything to go by, which they are probably not. Anyway toads and Queensland seem to go together, so here it is.

TOAD IN THE HOLE is a game for 4 players. All can play at the same time and the aim is to get your frog out of the hole. Actually, the frog hops any which way, all you do is tap the spacebar to say, "Hop, frog!". Maybe you'll be luck if the frog gets out of the hole at all!

Here it is, then, the latest from the warped mind of the mad Sailor from the South. Who? John Day, ninny, who else?

The Listing:

```
1 '?? TOAD IN THE HOLE ??
2 '
3 '  A DUBIOUS GAME BY
4 '     JOHN DAY
5 '
6 '  HIS FIRST IN ECB!
7 '
8 '
10 SCREEN1,0:PCLS
20 R$(1)="O1T255V5AV>AV>AV25AV30
AV+A"
30 R$(2)="O1T255V5CP5V10C#P5V15C
P5V20C#P5V25CP5V30C#"
40 R$(3)="O1T255V5EP1EP1V10EP1EP
1V15EP1EP1V20EP1EP1V25EP1EP1V30E
"
50 R$(4)="O2T255V5AV10A#V15BV200
3CV25C#V30D"
60 GOSUB520:GOSUB770
70 CLS:PRINT@224,"      HANG ABOU
T - CATCHING            A TOAD OR
 TWO FOR THE                  R
ACE."
80 J$="O4T255AA#BCC#DD#EFF#GF#FE
D#DC#CBA#A"
90 N$(1)="C2BM11,6E2D12L3R4"
100 N$(2)="C2BM5,174U2E2R8F2D4G1
0D2R12U2"
110 N$(3)="C2BM237,174U2E2R8F2D4
G2L3R3F2D4G2L8H2U2"
120 N$(4)="C2BM244,19U14G10R12"
130 PL(1)=11:PR(1)=28:PL(2)=11:P
R(2)=161:PL(3)=243:PR(3)=161:PL(
4)=243:PR(4)=28
140 LINE(2,2)-(20,20),PSET,B:PAI
NT(4,4),3,4
150 LINE(2,189)-(20,169),PSET,B:
PAINT(4,171),3,4
160 LINE(253,2)-(233,20),PSET,B:
PAINT(235,4),3,4
170 LINE(233,169)-(253,189),PSET
,B:PAINT(235,171),3,4
180 FOR N=1TO4:DRAW N$(N):NEXT
190 CIRCLE(128,96),10,4:PAINT(12
8,96),2,4
200 CIRCLE(128,96),95,4:PAINT(12
8,20),3,4
210 'INITIAL POSITION OF TOADS
220 DRAW"BM110,85":X(1)=110:Y(1)
=85:CIRCLE(X(1),Y(1)),4,2:PAINT(
X(1),Y(1)),2,2
230 DRAW"BM+0,25":X(2)=110:Y(2)=
110:CIRCLE(X(2),Y(2)),4,2:PAINT(
X(2),Y(2)),2,2
240 DRAW"BM+35,0":X(3)=145:Y(3)=
110:CIRCLE(X(3),Y(3)),4,2:PAINT(
X(3),Y(3)),2,2
250 DRAW"BM+0,-25":X(4)=145:Y(4)
=85:CIRCLE(X(4),Y(4)),4,2:PAINT(
X(4),Y(4)),2,2
260 PMODE3,1
270 SCREEN1,0
280 FOR L=1TO4
290 CIRCLE(PL(L),PR(L)),4:PAINT(
PL(L),PR(L)),4,4
300 RR=RND(4)
310 PLAY R$(RR)
320 IF INKEY$=""THEN340
330 GOTO360
340 FOR D=1TO RND(5)*100:NEXT
350 IF INKEY$=""THEN300
360 PAINT(128,96),3,3:PAINT(X(L)
,Y(L)),3,3
370 PLAY J$
380 P=RND(8)+6:IF RND(10)>5THEN
P=-P
390 X(L)=X(L)+P
400 Q=RND(6)+6:IF RND(10)>5THEN
Q=-Q
410 Y(L)=Y(L)+Q
420 IF PPOINT(X(L),Y(L))=1ORPPOI
NT(X(L)+7,Y(L))=1OR PPOINT(X(L)-
7,Y(L))=1OR PPOINT(X(L),Y(L)+7)=
1ORPPOINT(X(L),Y(L)-7)=1OR PPOIN
T(X(L)+7,Y(L)+7)=1OR PPOINT(X(L)
-7,Y(L)-7)=1THEN 470
430 CIRCLE(X(L),Y(L)),4,2:PAINT(
X(L),Y(L)),2,2
440 PAINT(PL(L),PR(L)),1,1
450 NEXT
460 CT=CT+1:GOTO280
470 FOR W=1TO9:PLAY R$(4):NEXT:C
LS:SCREEN0,0:PRINT@96,"   TOAD #
"RIGHT$(STR$(L),1)" IS THE FIRST
 OUT OF              THE HOLE!!"
480 PRINT@228,"IN"CT"MOVES"
490 PRINT@448,"  ANOTHER RACE? <
Y/N>"
500 A$=INKEY$:IF A$=""THEN500
510 IF A$="Y"THEN RUN ELSE CLS:P
RINT"T.I.T.H. LOADED":END
520 CLS3:PRINT@32:PRINT@448
530 FORX=1TO481STEP32:PRINT@X,"
";:PRINT@X+29," ";:NEXT
540 FOR X=1TO48
550 READ TD:PRINT@TD,CHR$(128);:
NEXT
560 PRINT@123,CHR$(164);:PRINT@2
19,CHR$(161);
570 DATA101,102,103,104,105,107,
108,109,110,111,113,114,115,116,
117,119,120,121,122
580 DATA135,139,143,145,149,151,
155
590 DATA167,171,175,177,178,179,
180,181,183,187
600 DATA199,203,204,205,206,207,
209,213,215,216,217,218
610 PRINT@270,"IN THE";
620 FOR X=1TO41
630 READ HO:PRINT@HO,CHR$(128);:
NEXT
640 DATA294,298,300,301,302,303,
304,306,311,312,313,314,315
650 DATA326,330,332,336,338,343.
660 DATA358,362,364,368,370,375
670 DATA390,394,396,397,398,399,
400,402,403,404,405,407,408,409,
410,411
680 PRINT@327,CHR$(172);:PRINT@3
28,CHR$(172);:PRINT @329,CHR$(17
2);:PRINT@344,CHR$(172);:PRINT@3
45,CHR$(172);
690 PRINT@359,CHR$(163);:PRINT@3
60,CHR$(163);:PRINT @361,CHR$(16
3);:PRINT@376,CHR$(163);:PRINT@3
77,CHR$(163);
700 RR=RND(4)
710 PLAYR$(RR)
720 IF INKEY$="" THEN740
730 RETURN
740 FOR D=1TORND(5)*100:NEXT
750 IF INKEY$=""THEN700
760 RETURN
770 CLS:PRINT@32," "CHR$(34)"TOA
D IN THE HOLE"CHR$(34)" IS A RAC
ING  TOAD GAME SIMULATING THE MO
VES  OF REAL TOADS IN A REAL PON
D -"780 PRINT" NOT ALWAYS AS CO-
OPERATIVE AS  THE RACING OWNER
WOULD PREFER!
790 PRINT" EACH TOAD JUMPS IN TU
RN AT THE  PRESS OF THE SPACE BA
R, IN THE  DIRECTION, AND FOR A
```

Continued on page 22

COLOUR DUMP

32K ECB + PLOTTER GRAPHICS UTILITY

by George McLintock

COLOUR DUMP WORKS by scanning across (or down) the screen, one line at a time, and drawing lines across (or down) the page on the plotter as required by the size and options selected. Effectively this is the normal approach to screen dumps.

It is designed for a PMODE 3 screen, but includes a ML routine to convert a PMODE 4 screen to a PMODE 3 one, so that it can be used for a PMODE 4 screen as well.

ML routines are used to return one of the screen co-ordinates to the Basic program for the dump. When drawing lines across the page, the Basic program keeps track of the Y co-ordinate (line number), and the ML routine returns the move to and draw to, X co-ordinate values (column number) for all the lines to be drawn across that line.

The role of the X and Y co-ordinates are reversed when drawing lines down the page. This is achieved by the values assigned to P and P1 in line 18. The values from the ML routine are returned to Basic in page 5 of the graphic screen.

When the dump is scaled up, additional lines are drawn as required to fill in the spaces between each line generated from the screen. The skip factors apply to the lines as drawn on the paper, not to the lines generated from the screen.

This program has been designed and coded specifically to allow it to work equally as well using the large graphic screen of ARTLINER. By changing a few parameters in it, the program can be used to dump that large screen directly.

This also means it should be relatively easy to convert it to work with the new CoCo 3 screens.

PARAMETERS REQUIRED FOR THE SCREEN

The Basic lines required for the actual screen dump are 1 to 26 and 75 to 80. The ML routines are also required. If the setup code is excluded, then the following parameter values should be set in Basic code following line 80, and the program should then branch to line 16.

Include a SCREEN statement if you want to see the graphic screen during the dump.

RP is number of repeats (starting from zero)
B(RP,8) must be set with a DIM statement, and the following 9 parameters set for each repeat
F is the counter used in the FOR .. NEXT loop for the dump

B(F,0) = screen colour to be used
B(F,1) = Pen colour
B(F,2) = Direction for lines
  1 for across
  2 for down
B(F,3) = Skip at start
B(F,4) = Skip between lines
B(F,5) = Row start (0-190)
B(F,6) = Row end (1-191)
B(F,7) = Column start (0-254)
B(F,8) = Column end (1-255)

Other parameters required are
PT = Plotter type
  0 CGP
  1 Pixy
K = Size of dump
RT = Rotation
  0 for normal
  1 for rotated 90 degrees

Q and R are the constants required to position the dump in the required position on the page. Values required can be derived from lines 12/13, or from the size part of the set-up routine where these values are set.

To convert a PMODE 4 screen to PMODE 3, EXEC MC

To initialise the routine for disk or tape EXEC M+256.

OTHER OPTIONS

The program has an option to allow you to reproduce the same colour on the screen in a number of different colours on the paper. For example, red in the top left corner can be dumped in red to the plotter, while the same colour, red, in the top right corner of the screen can be made green on the paper. With any number of variations of the same idea.

While this may seem a little complicated, there are occasions when being limited to four colours on the screen can be restrictive, particularly when the plotter can produce a far greater range of colours and patterns in the hard copy dump.

For example, with a scene, you can have one style of blue for the sky, another for water, and possibly a third for mountains, or the colour of a house or car.

Another variation might be for different coloured clothing on people. Two female figures might have the same coloured dress on the screen, but you could dump one with a red and white striped dress, and the other with a tartan skirt and a blouse with stripes the other way.

While this sort of variation requires another level of planning and set-up for the dump, it does provide a significant extension to the graphic capabilities of the CoCo with plotter. Some very elaborate coloured pictures on paper can be produced with this combination.

This option is obtained through parameters 5 to 8. Each repeat of the dump can be restricted to a specific area of the screen. The area to be dumped is specified by the row and column start and stop positions. Only lines within this range are drawn. The default is the full screen.

The setup routine also contains an option which allows you to set these values directly with reference to the actual graphic screen. If you use this option, the values selected are entered directly in to the repeat array for that repeat.

You can of course use this option to find all the reference points you want at the same time, and then key the values in directly.

This option operates completely independently of the

skip factors. The skip factors will always apply in a constant way from the top (or side) of the screen.

To reduce pen changes, the normal setup routine sorts the repeat array into ascending pen number sequence. If you want the lines drawn in a special sequence, then the menu provides an option to suppress this sort.

The sort used is Shell-Metzner, which is very compact (9 lines of code - 28 to 36), and one of the fastest Basic sorts available.

PRE-DEFINED PATTERNS

COLOUR DUMP includes a facility to pre-define a number of colour/patterns which can be used by entering A to Z instead of the normal pen colour. These are defined in lines 325 to 361 and can be modified when desired.

Patterns pre-defined in the listing provide examples of what can be produced. They are:

A to D correspond to pens 0 to 3 and produce a square pattern for each pen colour
: the pattern is draw lines across the page with skip zero at start and skip 4 between lines
: then draw lines down the page with the same skip factors and pen colours

E produces an off set colour mix drawing lines across the page
: is pen 1 skip zero at start and one between
: and pen 2, skip one at start and one between

F as for E but draws lines down page

G as for E, with pens 2 and 4

H as for G, but down page

I as for E, with pens 3 and 4

J as for I, but down page.

K draws a narrow stripe pattern across the page
: stripes are 3 lines deep, skip 3 between
: using pen 1

L as for I, but down page.

M draws a wider stripe pattern
: 9 lines deep, skip 9
: using pen 1

N as for M, but down page.

O draws narrow alternating stripes across page
: stripes are 3 deep with pen 1, skip 3
: then 3 deep with pen 2, skip 3

P as for O, but down page

Q draws alternating wide stripes across page
- uses pens 1 and 2 (as for O) but stripes 9 lines deep.

R as for Q, but down page

S Provides a 3 colour narrow striped pattern
: with pens 1,2 and 3

T as for S, but down page

U combines K & L

V combines M & N

W combines O & P

X combines Q & R

Y combines S & T

Z duplicates A.

These patterns are defined by the strings in A$(26), where A$(0) is the number of patterns defined (must be a string eg "26") A$(1) corresponds to A, and A$(26) to Z etc

Within each string (using A$(1) as an example)

- the first 2 characters specify how many repeats are required for the definition
: eg 02 for 2 repeats required

- For each repeat, you then require 6 characters in the string which are used to define the parameters which may change for each repeat

Within each group

The first character is the pen colour
eg 0 in position 3 and position 9

The second character is the direction of paint
eg 1 in position 4 (for across) and 2 in position 10 (for down)

The third and fourth character is the skip at start value
eg 00 in position (5 & 6) and in (11 & 12)

The fifth and sixth character is the skip between lines value
eg 04 in position (7 & 8) and in (13 & 14)

The intention was to generate these definitions by typing in the list of numbers required to define the patterns. But typing long lists of repetitive numbers is a pain, while string manipulation in basic can be fun.

Hence some definitions have been set up using small bits of Basic code. To see what is in the strings, break the program and print them

The repeat array generated from the menu is set up in A. The repeat array for the dump is in B. The routine to set up the predefined patterns generates B from A.

LINEDUMP

(Screen Dump by following lines)

This program works by following lines on the screen and then drawing the equivalent line on the plotter. It is a destructive dump in that it resets the points as it goes.

To allow the dump to be repeated, the program uses PCLEAR 8, and saves pages 1-4 in pages 5-8 before it starts, and PCOPY's them back for each repeat.

It is designed for a PMODE 4 screen, but includes a ML routine to convert a single colour in a PMODE 3 screen to a PMODE 4 one that can be dumped. This occurs separately for each repeat so that the program can be used to dump a complete PMODE 3 screen if desired. (Requires 4 repeats, one for each colour)

The ML routines used to find the start of the next line to be drawn requires the bits defining these to be on.

So that it will not work directly with an inverse screen, ie pictures drawn using COLOUR 0,1 or 0,5. It does, however, contain another ML routine which will convert these screens to the format required.

If you are not sure of the COLOUR used, a PEEK of the background colour should equal 0. If it is 255 then the screen is inverse. The first graphic byte (Hex E00 for disk, or Hex 600 for tape) is normally the background colour.

If you try to dump an inverse screen, the program will draw one dot at a time, starting from the top left corner of the screen.

The dump itself is coded mainly in Basic, the ML routines are used only to find the start of the next line to be drawn, or the start of the next block to be painted.

This particular operation is painfully slow if coded in Basic, and is the main reason why screen dumps coded entirely in Basic are so slow.

PARAMETERS REQUIRED FOR THE SCREEN DUMP

The Basic lines required for the actual screen dump are 2 to 57, 85 to 91, and 210 to 219. The ML routines are of course required. If the setup code is excluded, then the following parameter values should be set in Basic code following line 91, and the program should branch to line 213. (The picture to be dumped must be copied to pages 5 to 8 before branching to line 213.)

Include a SCREEN statement if you want to see the graphic screen during the dump, and watch the progress of it to estimate how much longer it might take. Remember you must set the value of F to suit.

RP is the number of repeats (starting from zero)

RC(RP,6) must be set with a DIM statement, and the following 7 parameters set for each repeat.

RX is this repeat number and is zero at the start.

RC(RX,0) = offset from previous dump
RC(RX,1) = pen colour
RC(RX,2) = switch to control the operations performed
0 for both paint and draw
1 for paint only (no draw)
2 for draw only (no paint)
Painted areas are still drawn (is equivalent to no paint option)
RC(RX,3) = direction of lines in paint
0 Don't use the paint option
1 draw lines across the page
2 draw lines down the page
RC(RX,4) = skip at start (for paints)
RC(RX,5) = skip between lines (for paints)
RC(RX,6) = if PMODE 3 this is the colour dumped for this repeat, plus one.

0 if PMODE 4 or
1 plus the PMODE 3 colour number (0-3 + 1)

Other parameters required are

F value returned by PPOINT when the dot is on
1 for SCREEN 1,0
5 for SCREEN 1,1
PT plotter type (as before)
K size of dump (as before)
RT rotation (as before)
FP option to draw single dots or not
0 for no - non-zero for yes
PX preferred direction to start searching for the continuation of a line
0 for across or non-zero for down
Q and R are the constants to position the dump on the paper. They operate as
for COLOUR DUMP.

To invert an inverse screen EXEC MI
To convert a PMODE 3 screen, POKE the PMODE 3 colour (0-3) to be dumped into MP and EXEC MC.

PROGRAM OPERATION

The basic operation of this program revolves around its line search procedures. A ML routine is used to find the start of the next line to be drawn, or colour block to be painted, if painting.

The ML search starts at the top left corner of the screen, and scans across each line. It moves down the screen, line by line, until it finds a starting point. The X,Y co-ordinates of the start of the graphic screen byte that contains this starting point are then returned to the Basic program, which performs the rest of the dump.

The Basic program then finds the actual bit co-ordinate to start from, and then enters its line draw or paint cycle.

The line draw routine will continue to draw a line on the paper for as far as it can without lifting the plotter pen. It resets all points as it goes, so that each point is only drawn once

From its initial starting point, the program searches around this point until it finds an adjoining bit that is also on. It will then follow this direction, in a straight line, until the next bit in that direction is not on.

At this stage, it draws that line on the paper and returns to its search routine around this last point. It will continue this cycle until it reaches a point which has no adjoining bit on, when it will return to the ML routine to find the next new starting point.

The sequence in which the program searches around each end point for a new line to follow is determined by lines 5 to 8 of the program. The basic sequence (in degrees) is 90, 180, 270, 0, 135, 225, 315, and 45.

Without the paint option, the sequence in which this search occurs can have a significant impact on the final appearance of some dumps. The paint option overcomes most of the earlier problems I had with this aspect.

The only option still in the program is to start the search with either 90 then 180, or with 180 and then 90. Again with the paint option, this is not as significant as I thought it might be, but I've left it in. The setup routine sets this so that the first search is in the opposite direction to the paint direction.

PAINT OPTION

The program without the paint option, will work quite well with smaller 'sized dumps. But any larger sized dumps which includes an area of solid colour on the screen may require this option for an acceptable result.

The problem can be indicated by seeing what happens if you dump a solid square of colour scaled up by 9. The program will work around this block, from the outside in, and draw corresponding lines on the paper.

But the lines on the page will be 9 dots apart, which does not give the appearance of a solid block of colour.

If the dump is not scaled up then this doesn't matter, because the lines on the paper overlap anyway to give the appearance of a solid block of colour.

When used, the paint opteration is performed before the line draw. The ML search routine is extended to search for a minimum block of 4 bits on, as two adjoining rows and two adjoining columns.

When painting across the page, the program finds the start of the first line in the block, and follows it across to its end, in a straight line. It then draws this line, plus as many additional lines below it as required to fill the gap between this line and the next line from the screen. (Allowing for skip factors as it goes)

It then returns to the starting point of this line and searches for the starting point of the line below it. When it finds that point it then repeats the operation for the next line.

This is repeated until the colour block is painted out. The program then returns to the ML routine to find the start of the next block.

When all colour blocks are painted out, the program resets the ML routine for normal line search, and starts its normal line draw routine.

When painting, the skip factors apply independently to each colour block (They are reset at the start of each block) and determine which lines are actually drawn on the paper.

The same general procedure is applied when painting down the page, and the same code is used for both directions. The different dirctions are obtained by the value of X1 and Y1 being equal to zero or one.

When painting down the page, the program may not perform as expected. The initial ML search routine is designed for across the page. The equivalent search for down the page is rather complicated and has not been incorporated here.

The effects produced may be different for mixed colour paints, but can provide interesting variations.

The paint option will be applied to all areas on the screen where there are 4 adjoining bits on in a square. With some pictures this can occur in places which you might not consider to be solid blocks of colour, and you might think the program is not working properly. It does work, just give it time.

COLOUR OPTIONS

Colour options for this dump are obtained by using similar procedures as for COLOUR DUMP. It includes an option to have the repeat do the paint only, without following lines. With this combination you can use different skip factors to mix colours in the painted areas, while having a single colour for non-painted areas.

With a PMODE 3 screen, you can dump each colour as a separate repeat and get a normal 4 colour dump to correspond with the screen colours. Or mix colours with any particular PMODE 3 colour.

The additional colour option involves the offset. This moves the whole dump a number of dot positions for that repeat. So that you can obtain an offset overprinting for single lines as well as for colour blocks. eg do the first dump in blue and the second in red, offset one, to get an offset mauve for all single lines.

Skip zero at start, and one between each line drawn for both repeats to get the same effect for colour blocks.

Note that the offset also moves the colour blocks, so that an offset of one effectively adds one to the start skip specified for that repeat, relative to the original dump position on the paper.

OTHER OPTIONS

The paint option can leave single dots on the screen. The normal operation of this program is not to draw single dots. However, if you want to draw them, this option will do so.

The Listing:

```
1 GOTO 490 'SCREEN DUMP - PMODE
3 - BY GEORGE MCLINTOCK
2 N=PEEK(N2)*NK+PEEK(N2+1):IF N=
0 THEN RETURN 'NO LINES
3 W=N3:V1=0:V2=0:FOR Y9=1 TO N
4 X8=PEEK(W)*NK+PEEK(W+1):Y8=PEE
K(W+2)*NK+PEEK(W+3):W=W+4
5 IF X8>=B(F,P1) THEN GOSUB 8:IF
 Y8<=B(F,P1+1) THEN GOSUB 10:GOT
O 7 ELSE Y8=B(F,P1+1):GOSUB 10:G
OTO 7 'LINE STARTS IN BOUNDS
6 IF Y8>B(F,P1) THEN X8=B(F,P1):
GOSUB 8:GOSUB 10 'STARTS OUTSIDE
 FINISHES INSIDE
7 NEXT Y9:GOSUB 9:RETURN
8 IF B(F,2)=1 THEN X=X8:Y=X9:V2=
SW ELSE X=X9:Y=X8:V1=SW
9 IN$="M": IF PT=0 THEN 12 ELSE
13 'MOVE TO
10 IF B(F,2)=1 THEN X=Y8:Y=X9:V2
=SW ELSE X=X9:Y=Y8:V1=SW
11 IN$="D":IF PT <> 0 THEN 13 'D
RAW TO
12 IF RT=0 THEN PRINT#-2,IN$;Q+K
*X+V1;",";R-K*Y-V2;T$;:RETURN EL
SE PRINT#-2,IN$;Q-K*Y-V2;",";R-K
*X-V1;T$;:RETURN 'CGP
13 IF RT=0 THEN PRINT#-2,IN$;Q-K
*X-V1;",";R+K*Y+V2;T$;:RETURN EL
SE PRINT#-2,IN$;Q-K*Y-V2;",";R-K
*X-V1;T$;: RETURN 'PIXY
14 FOR SW=0 TO K-1:SC=SC-1:IF SC
 <0 THEN SC=B(F,4)
15 NEXT SW:RETURN
16 FOR F=0 TO RP:POKE N1,B(F,0):
IF PT=0 THEN IN$="C" ELSE IN$="J
"
17 PRINT#-2,IN$;B(F,1);T$
18 IF B(F,2)=1 THEN M1=M+26:M2=M
+20:M3=M+107:P=5:P1=7 ELSE M1=M+
79:M2=M+48:M3=M+141:P=7:P1=5 'SE
T PARAMS
19 SC=B(F,3):EXEC M1 'INITIALISE
20 IF B(F,P)>0 THEN FOR X=0 TO B
(F,P)-1:EXEC M2:GOSUB 14:NEXT X
'SKIP AT START
21 FOR X9=B(F,P) TO B(F,P+1):EXE
C M3 'SCREEN LINE
22 FOR SW=0 TO K-1:IF SC=0 THEN
GOSUB 2 'DO DUMP LINE
23 SC=SC-1:IF SC<0 THEN SC=B(F,4
)
24 NEXT SW,X9,F
25 IF PT=0 THEN IF RT=0 THEN X=1
91 ELSE X=255
26 IF PT=0 THEN PRINT#-2,"M0,";-
K*X;T$; ELSE PRINT#-2,"H";T$;
27 GOTO 380
28 IF QR$="N" THEN RETURN ELSE F
=RP 'SORT ON PEN NO
29 F=INT(F/2):IF F=0 THEN RETURN
30 FOR W=0 TO F-1
31 X=W:Y=W+F:P=0
32 IF B(X,1)<=B(Y,1) THEN 34
33 P=1:FOR P1=0 TO 8:E(1)=B(X,P1
):B(X,P1)=B(Y,P1):B(Y,P1)=E(1):N
EXT P1
34 X=Y:Y=Y+F:IF Y<=RP THEN 32
35 IF P<>0 THEN 31
36 NEXT W:GOTO 29
37 X9=1:V$=V$(X):PRINT@32*X+Y1,V
$; 'PARAMETERS FOR MENU
38 A$=INKEY$:IF A$<>"" THEN 41
39 X9=X9+1:IF X9>9 THEN X9=1:PRI
NT@X*32+Y1,V$;:IF V$=CHR$(191) T
HEN V$=V$(X) ELSE V$=CHR$(191)
40 GOTO 38
41 IF A$=CHR$(94) THEN PRINT@X*3
2+Y1,V$(X);:X=X-1:IF X<0 THEN X=
Y8:GOTO 37 ELSE 37
42 IF A$=CHR$(10) OR A$=CHR$(13)
 THEN PRINT@X*32+Y1,V$(X);:X=X+1
:IF X>Y8 THEN X=0:GOTO 37 ELSE 3
7
43 IF A$=CHR$(8) THEN 45 ELSE IF
 A$=CHR$(9) THEN 51
44 V$(X)=A$:PRINT@X*32+Y1,A$;:RE
TURN
45 CLS:IF F=1 THEN ON X+1 GOSUB
176,245,247,248,244,258,260 ELSE
 ON X+1 GOSUB 264,251,270,253,25
6,272,275,260
46 GOSUB 154
47 CLS:FOR SW=0 TO Y8:PRINT@SW*3
2,J$(SW);
48 PRINT@SW*32+Y1,V$(SW);:NEXT S
W
49 IF F <> 1 THEN PRINT@448,"FOR
 REPEAT NO";Y;
50 GOTO 37
51 IF F = 1 THEN 54 ELSE IF X=5
THEN GOSUB 60:X=7:GOTO 47 ELSE I
F X=6 THEN GOSUB 277:X=7:GOTO 47
52 IF X=3 THEN CLS:INPUT"START S
KIP";V$(X):X=4:GOTO47 ELSE IF X=
4 THEN CLS:INPUT"SKIP BETWEEN";V
$(X):X=5:GOTO47
53 GOTO37
54 IF X=5 THEN CLS:INPUT"ENTER N
UMBER OF REPEATS";V$(5):X=6:GOTO
 47
55 IF X=3 THEN GOSUB 172:X=4:GOT
O 47
56 IF X <> 2 THEN 37
57 V$(2)="F":IF V$(1)="0" THEN G
OSUB 87 ELSE GOSUB 181
58 IF RT=0 THEN V$(3)="0" ELSE V
$(3)="1"
59 X=X+2:GOTO 47
60 V$(5)="F":CLS:PRINT"ENTER SCR
EEN CO-ORDINATES FOR   ROWS TO B
E DUMPED (0-191)":PRINT"AS START
 ROW  , STOP ROW"
61 INPUT"IF USE ,<ENTER> WILL DE
FAULT TO FULL SCREEN";A(Y,5),A(Y
,6):IF A(Y,6)=0 THEN A(Y,6)=191
62 IF A(Y,5)<0 OR A(Y,6)>191 OR
A(Y,5)>=A(Y,6) THEN PRINT "INVAL
ID VALUES":GOTO 61
63 PRINT:PRINT"ENTER SCREEN CO-O
RDINATES FOR   COLUMNS (0-255)":
PRINT"AS START COLUEN , STOP COL
UMN":INPUT A(Y,7),A(Y,8):IF A(Y,
8)=0 THEN A(Y,8)=255
64 IF A(Y,7)<0 OR A(Y,8)>255 OR
A(Y,7) >= A(Y,8) THEN PRINT"INVA
LID VALUES":GOTO 63
65 RETURN
66 SW=1:PRINT@416,"INVALID OPTIO
N": PRINT "HAVE ANOTHER GO":GOTO
 69 'OTHER PARAMS
67 'COMMON GET OPTION
68 V1=PEEK(&H88):V2=PEEK(&H89)
69 IF SW <> 0 THEN PRINT@480,STR
ING$(30," ");:SW=0
70 PRINT@480,"ENTER OPTION ?";
71 A$=INKEY$:IF A$="" THEN 71
72 PRINT A$;:POKE &H88,V1:POKE &
H89,V2:RETURN
73 'SELECT OPTIONS ALL AND SUNDR
Y
74 CLEAR 1500:PMODE 3,1:SCREEN 1
,1:PCLEAR 5
75 CLS:PRINT "SCREEN DUMP TO PLO
TTER":PRINT"FOR PMODE 3 SCREEN":
PRINT:PRINT"WRITTEN BY GEORGE MC
LINTOCK"
76 DIM A1(28),A2(28),E(5),B1(28)
,B2(28)
77 DIM X,Y,F,X8,Y8,X9,Y9,W
78 NK=256:M=PEEK(27)*256+PEEK(28
)-313:EXEC M+256
79 N1=M+18:N2=M+14:N3=PEEK(M)*NK
+PEEK(M+1):MC=M+275
80 DIM K,P,P1,V1,V2,Q,R,RT,PT,IN
$,T$,SW,SC
81 A1$="Y"+CHR$(8)+CHR$(9)+CHR$(
94)+CHR$(10)+CHR$(21)+CHR$(93)+C
HR$(95)+CHR$(91)+"(),."
82 PRINT:PRINT "MENU OPERATIONS
ARE":PRINT"UP/DOWN ARROWS TO NEX
T CELL":PRINT"LEFT ARROW PROVIDE
S MORE INFO":PRINT"RIGHT ARROW A
LLOWS MORE OPTIONS   - WHERE AP
```

```
PLICABLE"
83 PRINT "OTHER KEYS ENTERED INT
O CELL":PRINT" ** YOU MUST ENTER
 Y AT OK CELL TO EXIT **":PRINT:
PRINT"USE   N  FOR STANDARD DUMP
":PRINT"OR PRESS ENTER FOR MENU"
;:GOSUB 155:CLS:IF A$ <> "N" THE
N 194 ELSE DIM B(2,8):RP=2
84 INPUT"ENTER SIZE OF DUMP";X:I
F X=1 THEN K=1:Q=112 ELSE IF X=2
 THEN K=2:Q=480-48:RT=1 ELSE PRI
NT"INVALID SIZE":GOTO 84
85 FOR X=0 TO 2:B(X,0)=X+1:B(X,1
)=X+1:B(X,2)=1:B(X,6)=191:B(X,8)
=255:NEXT X:T$=CHR$(13):PRINT#-2
,CHR$(18);T$;"I";T$;:SCREEN 1,1:
GOTO 16
86 '
87 CLS:PRINT"ENTER SIZE OF DUMP"
88 S1=VAL(V$(2)):E(1)=X:E(2)=Y:E
(3)=X1:E(4)=Y1:E(5)=X9
89 PRINT:PRINT"1 NORMAL SIZE":PR
INT "2 DOUBLE SIZE":PRINT "3 3X
SIZE (PART SCREEN)":PRINT "4 4X
SIZE (PART SCREEN)":GOSUB 68
90 K=VAL(A$):IF K<1 OR K>4 THEN
GOSUB 66:GOTO 90
91 ON K GOSUB 96,104,112,112
92 X=E(1):Y=E(2):X1=E(3):Y1=E(4)
:X9=E(5)
93 IF A$="C" THEN 87
94 IF P=-1 THEN IF S1=0 THEN COL
OR 4,1 ELSE COLOR 8,5
95 RETURN
96 CLS:PRINT "NORMAL SIZE DUMP":
RT=0
97 PRINT:PRINT "PRESS ENTER TO H
AVE DUMP IN      CENTER OF PAPER"
:PRINT"OR 0-224 FOR SIZE OF LEFT
 MARGIN":PRINT "ZERO WILL BE NO
MARGIN"
98 INPUT A$
99 IF A$="" THEN Q=112:GOTO 101
100 Q=VAL(A$):IF Q<0 OR Q>224 TH
EN 97
101 PRINT:PRINT"THIS DUMP WILL B
E IN NORMAL      ROTATION":PRINT"
PRESS ENTER IF OK":INPUT "OR 1 T
O ROTATE 90 DEGREES";A$:IF A$="1
" THEN RT=1:Q=480-Q ELSE RT=0
102 RETURN
103 '
104 CLS:PRINT "DOUBLE SIZE DUMP"
:PRINT"SCREEN IS ROTATED 90 DEGR
EES":RT=1
105 PRINT:PRINT "PRESS ENTER TO
HAVE DUMP IN      CENTRE OF PAPER
":PRINT " OR ENTER SIZE OF RIGHT
  (TOP)    MARGIN (0-96)"
106 INPUT A$
107 IF A$="" THEN Q=480-48:RETUR
N
108 Q=VAL(A$):IF Q<0 OR Q>96 THE
N 105
109 Q=480-Q
110 RETURN
111 '
112 CLS:PRINT K;" X SIZE":GOSUB
118
113 IF RT=0 THEN Q=-K*X:RETURN
114 Q=480+K*X:IF K=4 THEN R=24
115 RETURN
116 '
117 '
118 PRINT:PRINT "REQUIRES PARTS
OF THE SCREEN TO BE EXCLUDED":PR
INT "WHICH ARE SHOWN AS OUTSIDE
THE  LINES ON THE SCREEN"
119 PRINT:PRINT "THESE CAN BE AL
TERED BY THE      ARROW KEYS TO M
OVE THE LINES IN THE DIRECTION O
F THE ARROW"
120 GOSUB 154:CLS
121 PRINT "NORMAL CASE ARROW KEY
S MOVE      8 LINES AT A TIME":PR
INT"UPPER CASE ARROW KEYS MOVE O
NLY ONE LINE AT A TIME"
122 PRINT"PRESSING  Y  WILL SELE
CT THE      AREA WITHIN THE LINES
FOR THE    DUMP"
123 PRINT"ANY OTHER KEY WILL RET
URN TO     THIS MENU (TO CHANGE T
O ROTATIONOR THE SIZE)"
124 GOSUB 154
125 CLS:PRINT"ENTER THE ROTATION
 FOR THE       PICTURE":PRINT" 0
FOR NORMAL":PRINT" 1 FOR 90 DEGR
EES"
126 PRINT:PRINT" OR C TO CHANGE
THE SIZE":PRINT" OR X TO CHANGE
THE COLOR OF THELINES ON THE SCR
EEN"
127 GOSUB 68:IF A$="X" THEN PRIN
T:INPUT"ENTER VALUES FOR COLOR S
TATEMENT";P,SC:COLOR P,SC:P=-1:G
OTO 125
128 IF A$="C" THEN RETURN ELSE I
F A$="1" THEN RT=1:GOTO 134
129 RT=0
130 IF K=3 THEN X=48:X1=160 ELSE
 X=66:X1=120
131 Y=X+X1:GOSUB 139
132 IF Y1 <> 1 THEN 125 ELSE RET
URN
133 '
134 IF K=3 THEN X=16:X1=160 ELSE
 X=66:X1=120
135 Y=X+X1:GOSUB 157
136 IF Y1 <> 1 THEN 125 ELSE RET
URN
137 '
138 'SELECT FOR NORMAL SCREEN
139 SCREEN 1,S1
140 GET (X-2,0)-(X+2,191),A1,G
141 LINE (X,0)-(X,191),PSET
142 GET (Y-2,0)-(Y+2,191),A2,G
143 LINE (Y,0)-(Y,191),PSET
144 GOSUB 155
145 Y1=INSTR(A1$,A$):IF Y1=0 THE
N PUT (X-2,0)-(X+2,191),A1,PSET:
PUT (Y-2,0)-(Y+2,191),A2,PSET:RE
TURN
146 IF Y1=1 THEN LINE (0,0)-(X,1
91),PRESET,BF:LINE (Y,0)-(255,19
1),PRESET,BF:RETURN
147 IF Y1=2 THEN X9=-8 ELSE IF Y
1=3 THEN X9=8 ELSE IF Y1=6 THEN
X9=-1 ELSE IF Y1=7 THEN X9=1 ELS
E 144
148 PUT (X-2,0)-(X+2,191),A1,PSE
T
149 PUT (Y-2,0)-(Y+2,191),A2,PSE
T
150 X=X+X9:IF X<2 THEN X=2
151 Y=X+X1:IF Y>253 THEN Y=253:X
=253-X1
152 GOTO 140
153 '
154 PRINT@480,"PRESS ENTER TO CO
NTINUE";
155 A$=INKEY$: IF A$="" THEN 155
 ELSE RETURN
156 'SELECT FOR ROTATED SCREEN
157 SCREEN 1,S1
158 GET (0,X-1)-(255,X+1),A1,G
159 LINE (0,X)-(255,X),PSET
160 GET (0,Y-1)-(255,Y+1),A2,G
161 LINE (0,Y)-(255,Y),PSET
162 GOSUB 155
163 Y1=INSTR(A1$,A$):IF Y1=0 THE
N PUT(0,X-1)-(255,X+1),A1,PSET:
PUT(0,Y-1)-(255,Y+1),A2,PSET:RET
URN
164 IF Y1=1 THEN LINE (0,0)-(255
,X),PRESET,BF: LINE (0,Y)-(255,1
91),PRESET,BF:RETURN
165 IF Y1=4 THEN X9=-8 ELSE IF Y
1=5 THEN X9=8 ELSE IF Y1=8 THEN
X9=-1 ELSE IF Y1=9 THEN X9=1 ELS
E 162
166 PUT (0,X-1)-(255,X+1),A1,PSE
T
167 PUT (0,Y-1)-(255,Y+1),A2,PSE
T
168 X=X+X9:IF X<1 THEN X=1
169 Y=X+X1:IF Y>190 THEN Y=190:X
=190-X1
170 GOTO 158
171 '
172 CLS:PRINT"THE PROGRAM WILL N
ORMALLY SORT  THE REPEATS INTO P
EN COLOR":PRINT"SEQUENCE":PRINT:
INPUT"ENTER N TO STOP IT";QR$
173 '
174 GOTO 47
175 ' ADJUST A PMODE 3 SCREEN
176 PRINT "THIS SCREEN DUMP IS D
ESIGNED FORA PMODE 3 SCREEN"
177 PRINT "HOWEVER IT IS POSSIBL
E TO CHANGEA PMODE 4 SCREEN TO A
 PMODE 3   PICTURE WHICH CAN BE
DUMPED BY  THIS PROGRAM"
178 PRINT "THE RESULTS PRODUCED
WILL BE A  RED/ORANGE FOREGROUND
 ON A      GREEN/BUFF BACKGROUND
"
179 PRINT:PRINT"ENTER 3 OR 4":RE
TURN
180 'DO PARAMETERS FOR PIXY PLOT
TER
181 CLS:PRINT "PARAMETERS FOR PI
XY PLOTTER"
182 PRINT:PRINT "ENTER 1-9 FOR S
IZE OF PLOT":GOSUB 68:K=VAL(A$)
183 PRINT:PRINT"ENTER 0 FOR NORM
AL":PRINT "  OR  1 TO ROTATE 90
DEGREES":GOSUB 68
184 RT=VAL(A$):IF RT <> 0 THEN R
T=1
185 IF RT=1 AND K>6 THEN PRINT:P
RINT"MAX SIZE 6 WHEN ROTATED":GO
TO 182
186 PRINT:PRINT "ENTER X,Y CO-OR
DINATES FOR THE  PLOT POSITION O
F THE TOP LEFT   CORNER OF THE S
CREEN"
187 PRINT:PRINT "USE , OR 0, FOR
 TOP LEFT CORNER":PRINT "OR +X,Y
 FOR DISPLACEMENT FROM   OR  X,Y
 FOR ABSOLUTE VALUES"
188 INPUT A$,V2:IF A$="" OR A$="
0" THEN V1=2450:IF RT=1 THEN V2=
K*256:GOTO191 ELSE 191
189 IF LEFT$(A$,1)="+" THEN V1=2
450 - VAL(A$):IF RT=1 THEN V2=K*
256+V2:GOTO191 ELSE 191
190 V1=VAL(A$)
191 Q=V1:R=V2:IF R>1800 THEN PRI
NT "R=";R:INPUT"ENTER Y TO CONFI
RM";A$:IF A$<>"Y" THEN 181
192 RETURN
```

```
193 '
194 DIM J$(7),V$(7):X8=207:Y1=25
:PD=0
195 J$(0)="PMODE (3,4)"
196 J$(1)="TYPE PLOTTER (0,1)"
197 J$(2)="SIZE DUMP"
198 J$(3)="ROTATION (0,1)"
199 J$(4)="SCREEN (0,1)"
200 J$(5)="NO OF REPEATS"
201 J$(6)="  OK (Y/N)"
202 CLS:FOR X=0 TO 6
203 V$(X)=CHR$(X8)
204 NEXT X
205 F=1:X=1:Y8=6:V$(0)="3":GOSUB
 47
206 IF X=6 AND A$="Y" THEN 209
207 X=X+1:IF X>6 THEN X=0
208 GOSUB 37:GOTO 206
209 RP=VAL(V$(5)):DIM A(RP,8)
210 S1=VAL(V$(4)):IF S1 <> 1 THE
N S1=0
211 IF V$(0)="4" THEN EXEC MC
212 IF V$(1)="1" THEN PT=1:T$=CH
R$(10)+CHR$(13) ELSE PT=0:T$=CHR
$(13)
213 IF V$(2)="F" THEN 221
214 K=VAL(V$(2)):IF PT=0 AND K>4
 THEN PRINT "MAX SIZE IS 4X":PRI
NT "ENTER SIZE":INPUT V$(5):GOTO
 214
215 IF V$(3)="1" THEN RT=1 ELSE
RT=0
216 IF PT=0 AND K=2 THEN RT=1
217 IF V$(2)="F" THEN 221
218 IF PT=1 THEN Q=2450:IF RT=0
THEN R=0:GOTO221 ELSE R=256*K:GO
TO221
219 IF K=1 THEN Q=112 ELSE IF K=
2 THEN Q=48 ELSE IF K=3 THEN RT=
1:Q=-48 ELSE IF K=4 THEN RT=1:Q=
-4*66:R=24
220 IF RT=1 THEN Q=480-Q
221 J$(0)="SCREEN COLOR (0-3)"
222 J$(1)="PEN COLOR (0,3)"
223 J$(2)="ACCROSS OR DOWN (1,2)
"
224 J$(3)="SKIP AT START"
225 J$(4)="SKIP BETWEEN LINES"
226 J$(5)="SCREEN LIMITS"
227 J$(6)="FIND POSITION"
228 J$(7)="   OK (Y.N)"
229 FOR Y=0 TO RP:A(Y,5)=0:A(Y,6
)=191:A(Y,7)=0:A(Y,8)=255
230 CLS:FOR X=0 TO 7
231 V$(X)=CHR$(X8)
232 NEXT X
233 F=2:X=0:Y8=7:GOSUB 47
234 IF X=7 AND A$="Y" THEN 237
235 X=X+1:IF X>7 THEN X=0 ELSE I
F X>4 THEN X=7
236 GOSUB 37:GOTO 234
237 FOR X=0 TO 4:A(Y,X)=VAL(V$(X
)):NEXT X
238 IF A(Y,1)>3 THEN PRINT:INPUT
"INVALID PEN COLOR - TRY AGAIN";
A$:A(Y,1)=VAL(A$):GOTO 238
239 IF V$(1)>="A" THEN A(Y,1)=AS
C(V$(1)):PD=1
240 IF A(Y,2) <> 2 THEN A(Y,2)=1
241 NEXT Y
242 IF PT=0 THEN PRINT#-2,CHR$(1
8);T$;"I";T$;
243 GOSUB 323:SCREEN 1,S1:GOSUB
28:GOTO 16
244 PRINT"USE 0 IF PICTURE IS SC
REEN 1,0":PRINT"  OR 1 IF SCREEN
 1,1":RETURN
245 PRINT"0 FOR CGP-115":PRINT"1
 FOR PIXY":PRINT"  OR SIMILAR A4
 PLOTTER"
246 RETURN
247 PRINT"THE SIZE OPTIONS VARY
WITH THE  PLOTTER TYPE":PRINT:PR
INT"USE THE F OPTION TO FOLLOW T
HESETHROUGH":PRINT:PRINT"SET PLO
TTER TYPE FIRST":GOTO 262
248 PRINT"0 FOR NORMAL":PRINT"1
TO ROTATE 90 DEGREES":PRINT:PRIN
T"THIS IS SET AS PART OF THE":PR
INT"F OPTION FOR SCREEN SIZE"
249 PRINT:PRINT"THE F OPTION IN
THIS CELL ALLOWSYOU TO SUPPRESS
THE PEN COLOR   SORT":GOTO 262
250 RETURN
251 PRINT"USE 0 TO 3 FOR CGP-115
":PRINT"1 TO 3 FOR PIXY"
252 RETURN
253 PRINT"THE DUMP CAN SKIP ONE
OR MORE   LINES AT THE START OF
EACH      REPEAT BEFORE DRAWING
ANY LINES"
254 PRINT:PRINT"USE 0 FOR NONE":
PRINT"OR THE NUMBER OF LINES TO
SKIP"
255 PRINT:PRINT"USE THE F OPTION
 FOR HIGHER":PRINT"NUMBERS":GOTO
 262
256 PRINT"THE DUMP ALLOWS ONE OR
 MORE     LINES TO BE SKIPPED BE
TWEEN EACHLINE DRAWN"
257 GOTO 254
258 PRINT"DIFFERENT CLOLRS CAN B
E OBTAINEDBY REPEATING THE DUMP
USING     DIFFERENT COLORED PENS
"
259 PRINT"AND DIFFERENT SKIP FAC
TORS ETC":PRINT:PRINT"0 DOES A S
INGLE COPY":PRINT"OR 1-9 FOR NUM
BER OF REPEATS":PRINT"USE F FOR
HIGHER NUMBERS":GOTO 262
260 PRINT"YOU MUST USE  Y  TO EX
IT"
261 RETURN
262 PRINT:PRINT"THE F OPTION IS
OBTAINED BY     PRESSING THE RIG
HT ARROW KEY"
263 RETURN
264 PRINT"SELECT THE SCREEN COLO
R TO BE   PRINTED FOR THIS REPEA
T":PRINT
265 PRINT"0 - GREEN/BUFF":PRINT"
  = 00 ON DISPLAY SCREEN"
266 PRINT"1 - YELLOW/CYAN":PRINT
"  = 01 ON DISPLAY SCREEN"
267 PRINT"2 - BLUE/MAGENTA":PRIN
T"  = 10 ON DISPLAY SCREEN"
268 PRINT"3 - RED/ORANGE":PRINT"
  = 11 ON DISPLAY SCREEN"
269 RETURN
270 PRINT"YOU CAN HAVE THE LINES
 DRAWN    EITHER ACCROSS OR DOWN
 THE PAGE":PRINT:PRINT"USE 1 FOR
 ACCROSS":PRINT" OR 2 FOR DOWN"
271 RETURN
272 PRINT"YOU CAN RESTRICT THIS
REPEAT TO DUMP A SELECTED PART O
F THE     SCREEN ONLY":PRINT"THI
S IS SPECIFIED BY THE SCREEN CO-
ORDINATES AT WHICH TO START  AND
 STOP THE DRAWING OF LINES"
273 PRINT:PRINT"USE THE F OPTION
 TO ACCESS THESEOPTIONS"
274 GOTO 262
275 PRINT"THIS ALLOWS YOU TO FIN
D ACTUAL  GRID POSITIONS ON THE
SCREEN":PRINT"TO GET ROW AND COL
UMN START AND STOP POSITIONS":PR
INT"  - WHICH ARE AUTOMATICALLY
    ENTERED INTO THE SETUP"
276 PRINT:PRINT"USE THE F OPTION
 TO ACCESS IT":GOTO 262
277 CLS:PRINT"ARROW KEYS MOVE TH
E LINES 8 DOTSAT A TIME":PRINT"U
PPER CASE ARROW KEYS MOVE 1 DOTA
T A TIME":PRINT"<>,. MOVE 48 DOT
S AT A TIME"
278 PRINT:PRINT"USE SPACE BAR TO
 CHANGE THE     LINES TO BE MOVE
D":PRINT"USE Y TO EXIT":PRINT"
    - WITH VALUES SET":PRINT" OR
 P TO DISPLAY RESULTS":GOSUB 154
279 V$(6)="P"
280 V2=1:A(Y,5)=48:A(Y,6)=144:A(
Y,7)=64:A(Y,8)=192:SCREEN1,S1
281 IF V2=1 THEN P=5:P1=7 ELSE P
=6:P1=8
282 GET(0,A(Y,P+V2)-1)-(255,A(Y,
P+V2)+1),A2,G
283 GET(A(Y,P1+V2)-2,0)-(A(Y,P1+
V2)+2,191),B2,G
284 LINE(0,A(Y,P+V2))-(255,A(Y,P
+V2)),PSET
285 LINE(A(Y,P1+V2),0)-(A(Y,P1+V
2),191),PSET
286 GET(0,A(Y,P)-1)-(255,A(Y,P)+
1),A1,G
287 GET(A(Y,P1)-2,0)-(A(Y,P1)+2,
191),B1,G
288 LINE(0,A(Y,P))-(255,A(Y,P)),
PSET
289 LINE(A(Y,P1),0)-(A(Y,P1),191
),PSET
290 GOSUB 155:IF A$="P" THEN GOT
O 304
291 IF A$=" " THEN GOSUB 299:IF
V2=1 THEN V2=-1:GOTO 281 ELSE V2
=1:GOTO 281
292 V1=INSTR(A1$,A$):IF V1=0 THE
N 290 ELSE IF V1=1 THEN GOSUB 29
9:RETURN
293 PUT(A(Y,P1)-2,0)-(A(Y,P1)+2,
191),B1,PSET
294 PUT(0,A(Y,P)-1)-(255,A(Y,P)+
1),A1,PSET
295 ON V1-1 GOSUB 309,310,311,31
2,313,314,315,316,317,318,319,32
0
296 IF A(Y,P)<1 THEN A(Y,P)=1 EL
SE IF A(Y,P)>190 THEN A(Y,P)=190
297 IF A(Y,P1)<2 THEN A(Y,P1)=2
ELSE IF A(Y,P1)>253 THEN A(Y,P1)
=253
298 GOTO 286
299 PUT(A(Y,P1)-2,0)-(A(Y,P1)+2,
191),B1,PSET
300 PUT(0,A(Y,P)-2)-(255,A(Y,P)+
1),A1,PSET
301 PUT(A(Y,P1+V2)-2,0)-(A(Y,P1+
V2)+2,191),B2,PSET
302 PUT(0,A(Y,P+V2)-1)-(255,A(Y,
P+V2)+1),A2,PSET
303 RETURN
304 CLS:PRINT"CURRENT VALUES ARE
":A$="######":PRINT
305 PRINT"     START  STOP"
306 PRINT"ROW";:PRINT USING A$;A
(Y,5);A(Y,6)
307 PRINT"COL";:PRINT USING A$;A
(Y,7);A(Y,8)
308 GOSUB 154:SCREEN 1,S1:GOTO 2
90
```

```
309 A(Y,P1)=A(Y,P1)-8:RETURN
310 A(Y,P1)=A(Y,P1)+8:RETURN
311 A(Y,P)=A(Y,P)-8:RETURN
312 A(Y,P)=A(Y,P)+8:RETURN
313 A(Y,P1)=A(Y,P1)-1:RETURN
314 A(Y,P1)=A(Y,P1)+1:RETURN
315 A(Y,P)=A(Y,P)-1:RETURN
316 A(Y,P)=A(Y,P)+1:RETURN
317 A(Y,P1)=A(Y,P1)-48:RETURN
318 A(Y,P1)=A(Y,P1)+48:RETURN
319 A(Y,P)=A(Y,P)-48:RETURN
320 A(Y,P)=A(Y,P)+48:RETURN
321 SCREEN 1,1
322 GOTO 322
323 CLS:PRINT"SETTING UP PRE-DEF
INED COLORS"
324 IF PD=0 THEN DIM B(RP,8):FOR
 X=0 TO RP:FOR Y=0 TO 8:B(X,Y)=A
(X,Y):NEXT Y,X:RETURN
325 DIM A$(26):A$(0)="26"
326 A$(1)="02010004020004"
327 A$(2)="02110004120004"
328 A$(3)="02210004220004"
329 A$(4)="02310004320004"
330 A$(5)="02110001210101"
331 A$(7)="02210001410101"
332 A$(9)="02310001410101"
333 FOR X=6 TO 10 STEP 2
334 A$(X)=A$(X-1):MID$(A$(X),4)=
"2":MID$(A$(X),10)="2"
335 NEXT X
336 A$(11)="03110006110106110206
"
337 A$(12)="03120006120106120206
"
338 A$="110018":A$(13)="09"
339 FOR X=1 TO 9
340 A$(13)=A$(13)+A$:MID$(A$(13)
,2+X*6-3)=STR$(X-1)
341 NEXT X
342 A$(14)=A$(13):FOR X=1 TO 9
343 MID$(A$(14),2+X*6-4)="2":NEX
T X
344 A$(15)="06110012110112110212
210612210712210812"
345 A$(16)=A$(15):FOR X=1 TO 6
346 MID$(A$(16),2+X*6-4)="2":NEX
T X
347 A$(17)=A$(13):FOR X = 1 TO 9
348 MID$(A$(17),2+X*6-1)="36":NE
XT X
349 A$(17)=A$(17)+MID$(A$(17),3)
:MID$(A$(17),1)="18":FOR X=1 TO
9
350 MID$(A$(17),56+X*6-3)=MID$(S
TR$(17+X),2):MID$(A$(17),56+X*6-
5)="2":NEXT X
351 A$(18)=A$(17):FOR X=1 TO 18
352 MID$(A$(18),2+X*6-4)="2":NEX
T X
353 A$(19)="09110018110118110218
210618210718210818311218311318311
1418"
354 A$(20)=A$(19):FOR X=1 TO 9
355 MID$(A$(20),2+X*6-4)="2":NEX
T X
356 FOR X=1 TO 5
357 A$(20+X)=A$(9+2*X)+MID$(A$(1
0+2*X),3)
358 A$=STR$(VAL(LEFT$(A$(9+2*X),
2))*2):IF LEN(A$) > 2 THEN A$=MI
D$(A$,2)
359 MID$(A$(20+X),1)=A$
360 NEXT X
361 A$(26)=A$(1)
362 '
363 X1=-1:FOR X=0 TO RP:IF A(X,1
) < 4 THEN X1=X1+1 ELSE X1=X1+VA
L(LEFT$(A$(A(X,1)-64),2))
364 NEXT X:DIM B(X1,8):PRINT:PRI
NT"TOTAL OF";X1" REPEATS":X8=-1
365 FOR X = 0 TO RP:IF A(X,1) <
4 THEN X8=X8+1:FOR Y=0 TO 8:B(X8
,Y)=A(X8,Y):NEXT Y:GOTO 374
366 X9=A(X,1)-64:Y1=VAL(LEFT$(A$
(X9),2))
367 FOR Y=0 TO Y1-1:X8=X8+1
368 B(X8,0)=A(X,0):FOR Y8=5 TO 8
:B(X8,Y8)=A(X,Y8):NEXT Y8
369 B(X8,1) = VAL(MID$(A$(X9),2+
6*Y+1,1))
370 B(X8,2) = VAL(MID$(A$(X9),2+
6*Y+2,1))
371 B(X8,3) = VAL(MID$(A$(X9),2+
6*Y+3,2))
372 B(X8,4) = VAL(MID$(A$(X9),2+
6*Y+5,2))
373 NEXT Y
374 NEXT X:RQ=RP:RP=X1
375 FORX=0 TO 26:A$(X)="":NEXT X
376 RETURN
380 CLS:PRINT"ENTER P TO PRINT P
ARAMETERS":INPUT"OR PRESS ENTER
TO FINISH";A$:IF A$<>"P" THEN ST
OP
381 IF PT=0 THEN PRINT#-2,CHR$(1
7)
382 PRINT:INPUT"ENTER NAME OF DU
MP";A$:PRINT#-2,"PARAMETERS FOR
";A$
383 PRINT#-2," SCN PEN DIR SKS S
KB  RS  RE  CS  CE":FOR X=0 TO R
Q:PRINT#-2,USING "####";A(X,0);:
IF A(X,1) < 4 THEN PRINT#-2,USIN
G "####";A(X,1);  ELSE PRINT #-2
,"   ";CHR$(A(X,1));
384 FOR Y=2 TO 8:PRINT#-2,USING"
####";A(X,Y);:NEXT Y:PRINT#-2
385 NEXT X:STOP
490 'ADD ML FIRST TIME
500 M$="9E1B308901366F806F806F80
9F1B39":Y=&H01DA 'ML ROUTINE TO
ADJUST POINTERS
501 B=0: FOR X=1 TO 30 STEP 2: N
=VAL("&H"+MID$(M$,X,2)): B=B+N:
POKE Y,N:Y=Y+1: NEXT X 'POKE TO
MEMORY
502 IF B <> &H0569 THEN PRINT"ER
ROR IN LINE 500":STOP
503 EXEC &H01DA:CLEAR 'EXEC ML
504 B=PEEK(27)*256+PEEK(28)-313:
LN=600:PRINT"LINE NO";
505 FOR X=0 TO 310 STEP 25 'POKE
 ML TO MEMORY
506 PRINT LN;
507 IF X<299 THEN N=25 ELSE N=10
508 A=0:FOR Y=0 TO N-1
509 READ C$:POKE B,VAL("&H"+C$):
B=B+1:A=A+VAL("&H"+C$)
510 NEXT Y:READ C$:IF A <> VAL("
&H"+C$) THEN PRINT "ERROR IN LIN
E NO";LN:STOP
511 LN=LN+1
512 NEXT X
513 CLS:PRINT"THE MACHINE LANGUA
GE ROUTINE IS NOW ADDED TO THE B
ASIC PROGRAM  (C)SAVE THIS VERSI
ON FOR LATER  USE":PRINT"RUN THE
 PROGRAM TO USE IT"
600 DATA 26,0,28,0,E,20,26,20,0,
20,0,C0,E,0,0,0,28,C0,3,0,8D,51,
AE,56,20,49D
601 DATA 4,8D,4B,AE,5C,AF,54,E6,
59,3A,86,8,3D,AF,56,E3,52,ED,40,
39,6C,58,39,8D,35,A8C
602 DATA 6D,58,27,F7,6F,58,6C,43
,86,4,A1,43,26,E,6F,43,AE,56,30,
1,AF,56,AE,54,30,919
603 DATA 1,AF,54,39,8D,16,AE,5C,
AF,54,6F,43,6F,58,10,AE,5A,E6,59
,3A,31,3F,26,FB,EC,A74
604 DATA 5A,20,BF,33,8C,A6,39,8D
,FA,AE,54,10,AE,52,86,4,A7,58,E6
,80,4F,59,49,59,49,AF2
605 DATA A7,A0,A7,A0,6A,58,26,F3
,AC,56,26,E9,8D,93,20,30,8D,D8,A
E,54,10,AE,52,E6,84,CCB
606 DATA A6,43,27,5,59,59,4A,26,
FB,59,49,59,49,A7,A0,E6,59,3A,AC
,56,26,E8,6D,58,27,A32
607 DATA 3B,6F,58,6C,43,86,4,A1,
43,26,3,17,FF,83,6F,5E,6F,5F,AE,
52,10,AE,50,A6,42,972
608 DATA A1,80,27,5,AC,40,2D,F8,
39,8D,1C,8D,24,A6,42,A1,80,26,8,
AC,40,2D,F8,8D,E,9D4
609 DATA 20,ED,8D,A,31,3E,8D,A,2
0,DB,6C,58,20,CE,1F,10,A3,52,83,
0,1,ED,A1,39,6C,932
610 DATA 5F,26,2,6C,5E,39,33,8D,
FF,C,DC,BA,ED,5C,C3,18,0,ED,50,C
3,2,0,ED,52,39,A89
611 DATA 33,8D,FE,F9,AE,5C,C6,4,
E7,5E,4F,59,59,69,84,49,69,84;49
,4D,27,2,CA,3,6A,AEA
612 DATA 5E,26,EE,E7,80,AC,50,26
,E4,39,518
640 DEL 500-640
650 GOTO 74
```

■

TOAD in the HOLE

Continued from page 15

```
DISTANCE,  CHOSEN BY THE TOAD IT
SELF."
800 PRINT" AS THE GAME BEGINS, T
HE LILY    PAD DISAPPEARS, AND S
TAYS OUT   OF THE WAY FOR WHOLE
GAME."
810 PRINT@448,"    PRESS A KEY T
O CONTINUE"
820 A$=INKEY$:IF A$=""THEN820
830 CLS
840 PRINT@32," SHOULD A TOAD JUM
P ON TOP OF    ANOTHER, THE NEXT
 OF THOSE TO   JUMP WILL SUBMERG
E THE OTHER,   WHICH WILL NOT SU
RFACE UNTIL    ITS TURN COMES AR
OUND."
850 PRINT" THE FIRST TOAD TO LEA
P OUT OF   THE POND, (PRESUMING
THAT ONE   ACTUALLY DOES), IS TH
E WINNER!"
860 PRINT@448,"    PRESS A KEY T
O CONTINUE"
870 IF INKEY$=""THEN 870
880 RETURN
```

CHECKSUMS FOR ML ROUTINES

32K ECB
UTILITY

by George McLintock

ONE OF THE major problems with magazine articles containing ML routines as DATA statements within a Basic program is the lack of checksums to assist those who actually type them in from the listing.

I have recently typed in the CGP-115 screen dump from the June CoCo, and feel that it is an extreme example of the problem. The ML routine is 766 bytes long, around 1500 Hex characters to be typed in, and not a single checksum to guard against typing errors. If it doesn't work, what do you do next?

Another problem with this particular program is that you might also notice that the ML entry points set by the DEFUSR's in line 20 of listing 1, are not contained within the addresses in line 2 of listing 2 which POKE the ML code into memory.

Your program is still not working, what do you check next?

I have submitted a number of programs which contain ML routines as DATA statements in Basic, and normally use another Basic program (Listing 1) to produce the DATA statements to be used. This program includes the calculation of a checksum at the end of each DATA line.

Listing 2 provides an outline of how these checksums can be handled in the program which POKE's the ML code back into memory.

These programs are submitted in the hope that other contributors might adopt a similar procedure to assist those of us who still type in programs from magazine listings.

TO CREATE THE DATA LINES

The Basic program, Listing 1, produces an ASCII file of DATA statements which corresponds to a ML program which has been (C)LOADM'ed into memory. The listing provided is the one used to create the DATA statements for REMOVE. This ML routine was ORG'ed at 32000, and the END statement occurred at 32168.

The DATA statements are produced with line numbers starting at 1000, and increments of 10. Each DATA line (except the last), contains 25 bytes of ML code with a checksum on the end.

To alter this for another program, change lines 20 and 40 to suit the start and end addresses of the ML routine. To modify for Extended Basic, also change lines 10 and 110 to File #-1 instead of #1, and delete the "/BAS" in the file name.

The file produced by this program has the same structure as would be produced by a (C)SAVE "MLREMVE",A. The PRINT in line 100 allows you to check that the last byte is correct.

TO PRODUCE LISTING 2

Listing 2 is the Basic program which is used to read the DATA lines produced from listing 1, and POKE the ML code back into memory. For different ML routines, all that needs to be altered is line 20, for the start address, and lines 40, 50 and 120 to correspond to the length of the program.

For Disk Basic you can load this program from some previous use of it, and then MERGE the DATA statements to it. For Extended Basic, if you don't have a tape merge procedure, CLOAD the file produced by listing 1, and then type in listing 2.

In either case, before submitting it to a magazine, RUN the program from the copy to be submitted, to make sure that it works.

The Listing:

```
0 GOTO5
1 '******* LISTING 1 ***********
3 SAVE"29:3":END
5 'BASIC PROGRAM TO CREATE DATA
STATEMENTS FOR REMOVE
10 LN=1000:OPEN"O",#1,"MLREMVE/B
AS"
20 FOR X=32000 TO 32168 STEP 25
30 A$=MID$(STR$(LN),2)+" DATA ":
B=0
40 IF X<32149 THEN N=25 ELSE N=1
8
50 FOR Y=0 TO N-1
60 B=B+PEEK(X+Y)
70 A$=A$+HEX$(PEEK(X+Y))+","
80 NEXT Y
90 A$=A$+HEX$(B):LN=LN+10
100 PRINTA$
110 PRINT#1,A$
130 NEXT X
140 CLOSE:STOP
```

The Listing:

```
0 GOTO5
1 '********.LISTING 2 **********
3 SAVE"29A:3":END
10 'BASIC PROGRAM FOR REMOVE UTI
LITY
15 CLEAR 500,32000
20 M=32000
30 LN=1000: Z=M
40 FOR X=0 TO 168 STEP 25
50 B=0:IF X < 149 THEN N=25 ELSE
 N=18
60 FOR Y=0 TO N-1:READ C$
70 C=VAL("&H"+C$):POKE Z,C
80 B=B+C: Z=Z+1
90 NEXT Y: READ C$
100 IF B <> VAL("&H"+C$) THEN PR
INT "ERROR IN LINE NO";LN:STOP
110 LN=LN+10:NEXT X
120 PRINT "REMOVE NOW AVAILABLE
IN MEMORY FROM";M;" TO";M+168-1
999 'LINES 1000 ON ARE DATA LINE
S CREATED BY LISTING 2
```

Just tell 'em, Joe sent ya!

AL'S HOUSE

GAME

by Charles Bartlett

AL AND HIS BOYS don't like no visitors, especially FEDS wid rods. Dey aint no dummies these boys, so yus better be good or yus ul be dead !!!

Al's boys appear randomly at the windows, but they aint so dumb as to stand in front of a window that any lousy fed, (YOU), already has his sights on.

They will appear at any OTHER window and you must move your gun sight, (using the joystick), to that window. Ya gotta be quick though.

When you have one of Al's boys in your sights, press the fire button and your machine gun will go off with a five round burst, (and machine gun sound). A single tone is heard for a hit and one point is awarded.

Al don't want you to knock off all his boys, so just to keep you on your toes, he appears in person, in the doorway.

Sometimes he's just come to make you jump, but watch out, cos sometimes he pauses a bit longer to take aim and if you don't get him ... he will fill ya full of lead.

The game is over when you are dead, this fun happening can occur either when you run out of ammo, (of which you start with 500 rounds) or when you are not quick enough in shooting Al.

In other words, no matter what your final score, you end up dead ... Al likes this game !!!!

Should you not like the colors as they appear on your screen, the PALETTE colors have been labeled to enable you to know which ones it is you are changing.

```
0 GOTO10
3 SAVE"168:3":END
10 '        AL'S HOUSE
 (C)  1/1/87  CHARLES BARTLETT
20 ON BRK GOTO800
30 ON ERR GOTO810
40 POKE65497,0:GOSUB670:HSCREEN2
:HCLS0:HBUFF1,520:HBUFF2,3560:HB
UFF3,660:HBUFF4,660:HBUFF5,1800:
DIM BP(12,1),PC(5):FOR Q=1 TO 5:
READ PC(Q):NEXT:DATA35,38,54,63,
0
50 GOSUB430:HCLS0
60 HCOLOR1:FOR Y=1 TO 25 STEP 5:
HLINE(0,Y)-(60,Y),PSET:NEXT:HLIN
E(60,1)-(60,20),PSET
70 FOR X=0 TO 51 STEP 10:FOR Y=1
 TO 5:HLINE(X,Y)-(X,Y),PSET:HLIN
E(X,Y+10)-(X,Y+10),PSET:HLINE(X+
5,Y+15)-(X+5,Y+15),PSET:HLINE(X+
5,Y+5)-(X+5,Y+5),PSET:NEXTY,X
80 FOR X=0 TO 51 STEP 10:FOR Y=1
 TO 20 STEP 5:HPAINT(X+1,Y+1),RN
D(2)+2,1:NEXTY,X
90 HGET(0,1)-(50,20),1:GOSUB100:
GOTO110
100 HCOLOR15:HLINE(148,65)-(202,
191),PSET,BF:HGET(148,65)-(202,1
28),5:RETURN
110 Z=1:FOR X=10 TO 300 STEP 50:
HLINE(X,10)-(X+30,50),PSET,BF:HL
INE(X,90)-(X+30,130),PSET,BF:BP(
Z,0)=X:BP(Z,1)=10:Z=Z+1:NEXT:FOR
 Z=7 TO 12:BP(Z,0)=BP(Z-6,0):BP(
Z,1)=90:NEXT
120 FOR Y=1 TO 190 STEP 20:FOR X
=1 TO 251 STEP 50:HPUT(X,Y)-(X+4
9,Y+19),1,OR:NEXTX,Y
130 HCOLOR14:HLINE(142,61)-(208,
191),PSET,B:HLINE(148,65)-(202,1
91),PSET,B:HPAINT(145,62),14,14
140 HGET(10,10)-(40,50),4
150 AM=500:GOSUB420:HCOLOR9:HPRI
NT(2,20),"AL'S HOUSE":HCOLOR0:HL
INE(10,180)-(100,190),PSET,BF:HL
INE(220,180)-(290,190),PSET,BF:H
COLOR9:HPRINT(2,23),"HITS":GOSUB
270:HPRINT(28,23),"AMMO":GOSUB35
0
160 PS=0:Z=1:PZ=1:GOSUB950
170 OZ=Z
180 Z=RND(13):IF Z=PZ THEN 180
190 IF Z=13 AND PZ=10 THEN180
200 IF Z=13 THEN Z=10:GOTO910
210 GOSUB280
220 GOSUB820
230 IF AM<=0 THEN OZ=Z:Z=10:GOSU
B280:GOTO370
240 GOSUB250:GOTO170
250 OK=0:IF BUTTON(0)=1 THEN FOR
 Q=1 TO 5:POKE146,10:EXEC43345:P
OKE146,102:EXEC43345:HCOLOR0:GOS
UB360:AM=AM-1:GOSUB350:FOR DL=1
TO 30:NEXTDL,Q:IF PZ=Z THEN HCOL
OR0:GOSUB270:PS=PS+1:SOUND100,1:
OK=-1:RETURN
260 HCOLOR9
270 HPRINT(6,23),PS:RETURN
280 IF Z=10 THEN GOSUB310:GOSUB3
30:RETURN
290 IF OZ=10 THEN GOSUB340:GOSUB
320:RETURN
300 GOSUB310:GOSUB320:RETURN
310 HPUT(BP(OZ,0),BP(OZ,1))-(BP(
OZ,0)+30,BP(OZ,1)+40),4,PSET:RET
URN
320 HPUT(BP(Z,0),BP(Z,1))-(BP(Z,
0)+30,BP(Z,1)+40),3,PRESET:GOSUB
820:RETURN
330 HPUT(146,65)-(200,191),2:GOS
UB820:RETURN
340 HPUT(148,65)-(202,128),5,PSE
T:HPUT(148,129)-(202,191),5,PSET
:RETURN
350 HCOLOR9
360 HPRINT(32,23),AM:RETURN
370 HCOLOR6:HDRAW"BM155,110S16FE
DRGRGFLGHGUHELEUFU":HPAINT(160,1
20),6,6
380 FOR L=1 TO 10:FORQ=1 TO 5:PO
KE146,10:EXEC43345:POKE146,102:E
XEC43345:PALETTE6,PC(Q):FOR DL=1
TO20:NEXTDL,Q,L
390 PC=0:GOSUB900:GOSUB330
400 GOSUB420:HCOLOR9:HPRINT(2,20
),"Play Again"
410 I$=INKEY$:IF I$=""THEN410 EL
SE IF I$="Y" THEN PC=0:GOSUB900:
GOSUB670:GOSUB100:GOTO150 ELSE I
F I$="N" THEN 800 ELSE 410
420 HCOLOR0:HLINE(10,158)-(100,1
68),PSET,BF:RETURN
430 HCOLOR1:HCIRCLE(160,50),8,1,
2,.90,.64
440 HLINE(152,47)-(145,47),PSET:
HLINE(145,46)-(174,40),PSET:HLIN
E-(168,44),PSET:HLINE(153,43)-(1
60,38),PSET:HLINE(161,37)-(165,3
7),PSET:HPAINT(160,50),9,1
450 HLINE(166,37)-(167,40),PSET:
HLINE(140,71)-(155,62),PSET:HLIN
E-(160,78),PSET:HLINE-(165,62),P
SET:HLINE-(180,71),PSET:HLINE(15
0,66)-(159,79),PSET:HLINE(161,79
)-(170,66),PSET

460 HLINE(158,69)-(160,66),PSET:
HLINE-(162,69),PSET:HLINE(140,72
)-(136,89),PSET:HLINE(180,72)-(1
74,89),PSET:HLINE-(152,98),PSET:
HLINE-(150,92),PSET:HLINE-(166,8
```

Continued on page 30

CoCo 3 Program

Here's a game to test you wits

HOW?

GAME

by Charles Bartlett

THE REASON FOR the name of this game is in HOW it should be played and HOW the devil can you win. At first glance the screen looks like a PACMAN type layout, with one token, (you), and three other tokens, (them).

The screen has lots of dots in a maze pattern and your first inclination will be to charge around the screen collecting all the dots and dodging (THEM), well if you do that you will lose.

This is a game of strategy, (THEM) will try to get at you in an "as the crow flies" fashion. You must collect the dots AND try and get them to trap themselves at the same time.

There are 99 dots per screen, once you get all 99, they will be replaced and you can keep going.

You will ask your self HOW can I get all the dots without getting GOT ... all I can say is it can be done

The Listing:

```
0 GOTO10
3 SAVE"168C:3":END
10 '          H O W
   (C)  1/1/87  CHARLES BARTLETT

20 CLEAR1000:ON BRK GOTO820
30 POKE65497,0:DIM I,X,Y,K,XX,YY
,Q,BOX(10,10):GOSUB810:PLAY"V31T
255L25504"::HSCREEN2:HCLS2:GOSUB
660
40 DATA32,31,55,9,40,5,19,8,63
50 FOR I=1TO 4:READ DR$(I),BIT(I
):NEXT:DATA"BD18NR18BU18",1,"ND1
8",2,"BR18ND18BL18",4,"NR18",8
60 HCOLOR4:HDRAW"BM0,0"
70 FOR X=1 TO 10:PX(X)=PX:PX=PX+
18:FOR Y=1 TO 10:READ BOX(X,Y):P
Y(Y)=PY:PY=PY+18
80 FOR I=1 TO 4:IF NOT(BOX(X,Y)
AND BIT(I))=BIT(I) THEN HDRAW DR
$(I)
90 NEXTI:PLAY STR$(X)
100 HDRAW"BR18":NEXTY:PY=0:N=N+1
8:N$="BM0,"+STR$(N):HDRAWN$:NEXT
X:GOSUB750:GOSUB490:GOSUB510
110 HCOLOR4:HLINE(186,115)-(310,
155),PSET,B:HPAINT(190,120),8,4:
GOSUB710:HPAINT(255,191),1,4:GOS
UB680
120 OK=0:FORK=1 TO 3:GOSUB230:NE
XTK:GOSUB180
130 FOR Q=1 TO 2:GOSUB250:GOSUB4
40:IF C=99 THEN C=0:GOSUB490:GOS
UB510
140 P2=RND(2):IF P2=1 THEN C2=19
 ELSE C2=56
150 P1=RND(2):IF P1=1 THEN C1=16
 ELSE C1=29
160 PALETTE6,C1:PALETTE7,C2
170 NEXT Q:GOSUB260:GOTO130
180 HCOLOR3:HCIRCLE(PX(X)+10,PY(
Y)+10),7:HPAINT(PX(X)+10,PY(Y)+1
0),3,3:RETURN
190 HCOLOR2:HLINE(PX(X)+2,PY(Y)+
2)-(PX(X)+17,PY(Y)+17),PSET,BF:R
ETURN
200 HCOLOR2:HLINE(PX(GX(K))+2,PY
(GY(K))+2)-(PX(GX(K))+17,PY(GY(K
))+17),PSET,BF
210 IF (BOX(GX(K),GY(K)) AND 16)
=16 THEN HSET(PX(GX(K))+9,PY(GY(
K))+9,0)
220 RETURN
230 HCOLOR4:HCIRCLE(PX(GX(K))+10
,PY(GY(K))+10),7
240 HPAINT(PX(GX(K))+10,PY(GY(K)
)+10),4,4:RETURN
250 J1=JOYSTK(0):J0=JOYSTK(1):RE
TURN
260 FOR K=1 TO 3:IF GX(K)<X THEN
 GOSUB320
270 IF GX(K)>X THEN GOSUB340
280 IF GY(K)<Y THEN GOSUB360
290 IF GY(K)>Y THEN GOSUB380
300 IF GX(K)=X AND GY(K)=Y THEN
 FOR Q=1 TO 5:PLAY"O1N12N11N10N9
N8N12N11N10N9N8N7N6N5N4N3N2N1":N
EXT:RUN
310 NEXTK:RETURN
320 IF (BOX(GY(K),GX(K)) AND 4)=
4 THEN GOSUB400:IF A1=2 OR A1=3
 THEN GOSUB200:GX(K)=GX(K)+1:GOS
UB230
330 RETURN
340 IF (BOX(GY(K),GX(K)) AND 2)=
2 THEN GOSUB410:IF A2=2 OR A2=3
THEN GOSUB200:GX(K)=GX(K)-1:GOSU
B230
350 RETURN
360 IF (BOX(GY(K),GX(K)) AND 1)=
1 THEN GOSUB420:IF A3=2 OR A3=3
THEN GOSUB200:GY(K)=GY(K)+1:GOSU
B230
370 RETURN
380 IF (BOX(GY(K),GX(K)) AND 8)=
8 THEN GOSUB430:IF A4=2 OR A4=3
 THEN GOSUB200:GY(K)=GY(K)-1:GOS
UB230
390 RETURN
400 A1=HPOINT(PX(GX(K)+1)+10,PY(
GY(K))+10):RETURN
410 A2=HPOINT(PX(GX(K)-1)+10,PY(
GY(K))+10):RETURN
420 A3=HPOINT(PX(GX(K))+10,PY(GY
(K)+1)+10):RETURN
430 A4=HPOINT(PX(GX(K))+10,PY(GY
(K)-1)+10):RETURN
440 IF J0>58 AND (BOX(Y,X) AND 1
)=1 THEN GOSUB190:Y=Y+1:GOSUB180
:GOSUB530:RETURN
450 IF J1>58 AND (BOX(Y,X) AND 4
)=4 THEN GOSUB190:X=X+1:GOSUB180
:GOSUB530:RETURN
460 IF J1<5 AND (BOX(Y,X) AND 2)
=2 THEN GOSUB190:X=X-1:GOSUB180:
GOSUB530:RETURN
470 IF J0<5 AND (BOX(Y,X) AND 8)
=8 THEN GOSUB190:Y=Y-1:GOSUB180:
GOSUB530:RETURN
480 RETURN
490 FOR XX=1 TO 10:FOR YY=1 TO 1
0:BOX(XX,YY)=BOX(XX,YY)+16:NEXTY
Y,XX
500 BOX(Y,X)=BOX(Y,X)-16:RETURN
510 FOR XX=1 TO 10:FOR YY=1 TO 1
0:HSET(PX(XX)+9,PY(YY)+9,0):PLAY
 STR$(XX)
520 NEXTYY,XX:RETURN
530 PLAY"N5N4N3N2N1"
540 IF (BOX(X,Y) AND 16)=16 THEN
 BOX(X,Y)=BOX(X,Y)-16:PLAY"N6N7N
8N9":SC=SC+1:C=C+1:GOSUB680:RETU
RN
550 RETURN
560 DATA5,6,6,6,6,6,6,6,6,3
570 DATA13,7,6,6,2,4,6,6,7,11
580 DATA9,12,6,7,6,6,7,6,10,9
590 DATA12,6,3,9,5,3,9,5,6,10
600 DATA5,6,10,9,9,9,9,12,6,3
610 DATA13,3,5,14,10,12,14,3,5,1
1
620 DATA9,9,12,6,6,6,6,10,9,9
630 DATA9,12,6,7,3,5,7,6,10,9
640 DATA13,6,6,10,13,11,12,6,6,1
1
650 DATA12,6,6,6,10,12,6,6,6,10
660 N$(0)="BRGD4FR2EU4HNL2BR2":N
$(1)="BR2NGD6NLRBU6BR2":N$(2)="B
DER2FDG4R4BU6BR1":N$(3)="BDER2FD
GNLFDGL2HBR5BU5":N$(4)="BD3NR4E3
ND6BR2":N$(5)="NR4D3R3FDGL2HBR5B
U5":N$(6)="BDNED4FR2EUHNL3BU3NL2
NFBR2":N$(7)="BD6UE4UNL4BR"
670 N$(8)="BRGDFNR2GDFR2EUHEUHNL
2BR2":N$(9)="BD5FR2EU4HL2GDFR3BU
3BR1":RETURN
```

Continued on page 30

COME TO

at Bundee

Conf '87 this year is to be held at the Uniting Church's campsite in Bundeena NSW.

This is a particularly pretty area of Sydney, situated on the northern tip of the Royal National Park, in Port Hacking.

The water views are fabulous, and the bushwalks are amoungst the best in Australia.

Not that you'll have anytime during conference for these things, because as usual, the conference will be jam packed with all sorts of things to see and do!

The big news this year will obviously be the growing use of OS-9 Level 2 on the CoCo 3's; and Conf '87 will be the definitive place to see this excellent system.

By that time initial users will have had time to sort the system out and create some really interesting stuff.

But it is not just OS-9 that is of interest this year.

With the release of the new T1000 EX and SX, interest in these machines has never been higher. We'll have a number of these computers at the conference, as well as their big brothers, the T3000 series, which we'll be putting through their paces.

We've had continuing interest in some of the more diverse subjects covered in the magazine at past conferences, so again this year we'll have tutorials on hardware mods and on Forth.

There'll be Basic Basic and Advanced Basic courses, and an Assembly Language tutorial as well.

Other computers will be discussed, principally the 68000 series of computers, and of course, we'll be showing Goldlink 642 on Viatel - and Videotex in general.

Conference is a place to meet old friends, to meet the people behind the names in the magazine, to learn a lot of new information, to see the latest Tandy equipment.

We hope you'll come. We're sure you'll be glad you did. But please hurry your booking, because accommodation (which is not obligatory) and places at the conference, are both limited by the size of the centre.

The cost is increased over previous years due entirely to the fact that we are doing it in Sydney which is a good deal more expensive than the Gold Coast!

On the other hand, many of you will save by not having the additional travelling expenses associated with getting to the Gold Coast.

We aim to make the conference a family affair, and the location is a good one for people with families who are less interested in computers, but who would still like to be with dad or mum for the weekend.

The family can take a ferry trip, go for bush walks, or just laze on the beach, whilst you do your thing at the conference.

CONF '87

na N.S.W.

CONF '87

Rates

Accommodated (1) $87.00
Family of 2, + $68.00 = $155.00
Additional family members $52.00 ea
Includes supper Friday evening, breakfast lunch and dinner on Saturday and breakfast and lunch on Sunday plus all accommodation.

Non Accommodated Rates

	One day	Two Days
One person	$40.00	$58.00
Sat Evening Meal	$12.00	$12.00
	$52.00	$70.00

Additional family $31.50 $45.50 /person
Includes morning / afternoon tea and lunch.

$20.00 deposit required with booking;
final payment to be made by 15th July 1987.

LOCATION:-
Uniting Church's campsite
Bundeena NSW

DATE:- 8th & 9th August, 1987

REGISTER NOW!!
We can only accept a limited number of people this year. DON'T MISS OUT! on a top weekend of FUN, FRIENDSHIP and LEARNING.

Name: .....

Address: .....

Phone: .....

No. People attending: .....

SPEAK UP!:- Now is your chance to suggest your ideas for any tutorials we may not have mentioned. (participants only).

Tutorials likely to attend: .....

.....

Please find enclosed:
chq/money order/bankcard/visa/mastercard

Card No. .....

Signature: .....

Mary Poppins, eat your heart out

COCO RUNNER

32K ECB Tape Only

by Max Bettridge

THE IDEA OF CoCo Runner is to run through cities, forests, seasides, etc and at the same time avoiding all obstacles in your path.

Use the right joystick button to jump over any obstacles that might get in your way.

DISK USERS: load in CoCo Runner and save it to tape. Turn off your computer, unplug your disk controller, turn your computer back on again and type in: PCLEAR5:CLOAD

TAPE USERS: Type in PCLEAR5:CLOAD

The Listing:

```
0 GOTO10
3 SAVE"72:3":END
10 CLS0:POKE65495,0:GOTO30
20 'COCO RUNNER FOR 32K ECB
    BY MAX BETTRIDGE.
25 'note: tape users only
30 PMODE4:PCLS:PCOPY1TO5
40 DATA142,8,164,198,23,166,132,
73,73,73,132,3,52,2,166,132,72,7
2,52,2,166,1,73,73,73,132,3,170,
224,167,128,90,38,236,166,132,72
,72,170,224,167,132,48,136,9,140
,11,36,38,209,57,142,8,185,198,1
,166,128,167,136,191,90,38,248,4
8,136,31,140,11,89,38,238,57
50 A=30000:FORI=A TO A+50:READJ:
POKEI,J:NEXT:RESTORE:B=30052:FOR
I=B TO B+50:READJ:POKEI,J:NEXT:R
ESTORE:C=30103:FORI=C TO C+50:RE
ADJ:POKEI,J:NEXT:D=30155:FORI=D
TO D+21:READJ:POKEI,J:NEXT
60 POKE30053,7:POKE30054,100:POK
E30098,8:POKE30099,164:POKE30063
,3:POKE30104,6:POKE30105,36:POKE
30149,7:POKE30150,100:POKE30011,
0
70 GOTO510
80 FORR=1TO2:POKE2873,192:EXECD:
NEXT:RETURN
90 O=2873:POKEO,16:EXECD:POKEO,5
6:EXECD:POKEO,16:EXECD:RETURN
100 POKEO,84:EXECD:POKEO,124:EXE
CD:POKEO,16:EXECD:RETURN
110 POKEO,30:EXECD:POKEO,62:EXEC
D:POKEO,126:EXECD:RETURN
120 POKEO,254:EXECD:POKEO,130:EX
ECD:POKEO,254:EXECD:RETURN
130 O=RND(8):ON O GOTO140,150,16
0,170,180,190,200,210
140 PUT(180,32)-(190,42),O1,PSET
:RETURN
150 PUT(180,32)-(190,42),O2,PSET
:RETURN
160 PUT(180,32)-(190,42),O3,PSET
:RETURN
170 PUT(180,32)-(190,42),O4,PSET
:RETURN
180 PUT(180,32)-(190,42),O5,PSET
:RETURN
190 PUT(180,32)-(190,42),O6,PSET
:RETURN
200 PUT(180,32)-(190,42),O7,PSET
:RETURN
210 PUT(180,32)-(190,42),O8,PSET
:RETURN
220 IFPPOINT(49,40-I)>0THEN320EL
SERETURN
230 FORI=0TO8:PUT(40,30-I)-(48,4
0-I),M8,PSET:EXECA:GOSUB220:PCOP
Y1TO3:NEXT:FORI=0TO6:PUT(40,22+I
)-(48,32+I),M9,PSET:EXECA:PCOPY1
TO3:GOSUB220:NEXT:PUT(40,30)-(48
,40),DM,PSET
240 JP=JP+1:IFJP=3THEN330
250 P=PEEK(65280):IFP=254THEN230
260 PMODE4:M=M+1:IFM>10THENM=1
270 P=PEEK(65280):IFP=254ORP=126
THEN230
280 DL=DL+1:IFDL>21THENDL=21
290 IFDL>15-ST THENIFRND(20)=1TH
ENDL=0:ON ST GOSUB0,80,90,100,11
0,120
300 IFDL>15-ST THENIFRND(20)=9TH
ENDL=0:GOSUB130
310 IFPPOINT(49,39)>0THEN320ELSE
340
320 FORT=1TO5:PUT(40,30)-(48,40)
,DA,PSET:EXECA:PCOPY1TO3:NEXT:PU
T(40,30)-(48,40),DM,PSET:PCOPY1T
O3:FORT=1TO300:NEXT:PUT(40,30)-(
48,40),M3,PSET:PCOPY1TO3:FORT=1T
O300:NEXT:PUT(40,30)-(48,40),M3,
PSET:PCOPY1TO3:FORT=1TO500:NEXT
330 JP=0:F=F+32:POKE6586+F,0:PLA
Y"O1EFDGEFA":IFF=352THEN950ELSE4
10
340 ON M GOTO350,360,370,380,390
,400,390,380,370,360
350 PUT(40,30)-(48,40),M1,PSET:G
OTO410
360 PUT(40,30)-(48,40),M2,PSET:G
OTO410
370 PUT(40,30)-(48,40),M3,PSET:G
OTO410
380 PUT(40,30)-(48,40),M4,PSET:G
OTO410
390 PUT(40,30)-(48,40),M5,PSET:G
OTO410
400 PUT(40,30)-(48,40),M6,PSET:G
OTO410
410 PMODE4,2:SCREEN1,1:EXECA
420 TI=TI+1:IFTI=10THENEXECB:TI=
0:IFST>4THENEXECC
430 IFST<4THENTA=TA+1:IFTA=35THE
NEXECC:TA=0
440 FT=FT+1:IFFT=10THENPUT(TP,14
9)-(TP+8,159),M2,PSET:TP=TP+6-ST
:FT=0
450 IFTP=75THEN850
460 IFTP=124THEN850
470 IFTP=173THEN850
480 IFTP=219THEN850
490 IFTP>225THEN1230
500 PCOPY1TO3:GOTO260
510 DATA U2ERFDNL2D,U3R2FDNL2DL2
,NR3U2ERF,U3R2FDGL,NR2UNRUERF,UN
RUERF,U2ERFBDBLNLDL2,U3BR3D2NL2D
,U3,BUFRU3NL2R,U2NUBRRNEFD,NU3R3
,U3FRED3,U3F3U3,BUUERFDGLH,U3R2F
GL,BUUERFDGLHBRF2,U2ERFDLNLF,R3H
L2UERF,BR2U3NL2R,NU3R3U3,BU3D2FR
EU2,NU3ERFU3
520 DATAEUHBR3GDF,BU3FRNED2LU2,N
R3E3L3
530 DIML$(57):FORJ=32TO57:READR$
:L$(J)=R$:NEXT
540 PMODE4:SCREEN1,1:PCLS
550 AA=30:BB=30:W$="COCO RUNNER
        BY MAX BETTRIDGE    FOR
 THIRTY TWO K    COLOUR EXTENDED
    IF YOU REQUIRE     INSTRUCTI
ONS PRESS    Y  ELSE  N":GOSUB60
0
560 A$=INKEY$:IFA$="Y"THEN570ELS
EIFA$="N"THEN650ELSE560
570 PCLS:AA=20:BB=10:W$="YOU ARE
 RUNNING A   MARATHON OF FIVE
 STAGES   EACH STAGESAPS YOUR ST
RENGTH AND IS MORE DIFFICULTUSE
YOUR RIGHT FIRE BUTTON TO JUMP T
HE OBSTICALS AS THEY   APPROACH
        PRESS ANY KEY":GOSUB60
0:EXEC44539:AA=30:BB=10
580 PCLS:W$="EACH TIME YOU FALL
OVER AN OBSTICAL YOUTAKE A SIP F
ROM YOURENERGY CUP    EMPTYTHE C
UP AND YOU    HAVE LOST    YOU
RCUP IS REFILLED AT  EACH STAGE
       GOOD RUNNING
            PRESS ANY KEY":GO
SUB600:EXEC44539:PCLS
590 AA=20:BB=10:W$="THE DISTANCE
 COVERED IS INDICATED BY THE POS
SITION OF THE    RUNNER AT THE
    BOTTOM OF THE      SCREEN
            PRESS ANY KEY":GOS
UB600:EXEC44539:PCLS:AA=50:BB=90
:W$="STAND BY":GOSUB600:GOTO650
600 FORL=1TOLEN(W$):P$=MID$(W$,L
,1):IFP$=" "THEN630
610 DRAW"BM"+STR$(AA)+","+STR$(B
B)+";S12":DRAWL$(ASC(P$)-33)
620 IFP$="I"THENAA=AA-9
630 AA=AA+12:IFAA>240THENAA=20:B
B=BB+15
640 NEXT:DRAW"S4":RETURN
650 PMODE4
660 DATA6,6,0,28,18,24,20,52,1,0
,0,0,6,6,0,28,18,24,20,98,66,0,0
,0,6,6,0,28,18,24,20,20,100,0,0,
```

```
0,6,6,0,28,48,24,28,24,52,0,0,0,
6,6,0,14,24,12,10,18,34,0,0,0,6,
6,0,4,24,12,10,1,1,0,0,0,24,24,0
,60,60,60,24,24,24,28,0,0,24,24,
2,28,48,28,18,19,32,0
670 DATA0,0,3,3,0,15,16,24,31,1,
2,0,0,0,0,0,3,3,15,12,58,25,12,6
,0,0,0,0,32,51,13,12,12,34,45,12
,12,12,12,12,12,12,12,0,0,255
,255,3,255,255,192,255,255,0,0,2
55,255,195,15,15,195,255,255,0,0
,192,204,204,204,255,255,12,12,0
,0,255,255,192,255,255,3,255
680 DATA255,25,4,32,8,3,12,10,80
,4,38,2,40,129,16,121,144,14,248
,51,204,33,195,65,192,1,192,1,22
4,3,240,3,248
690 FORT=1TO130:READJ:POKE1568+X
,J:X=X+32:NEXT
700 DIMM1(2),M2(2),M3(2),M4(2),M
5(2),M6(2),M7(2),M8(2),M9(2),DM(
2),DA(2)
710 GET(0,0)-(8,10),M1,G:GET(0,1
2)-(8,22),M2,G:GET(0,24)-(8,34),
M3,G:GET(0,36)-(8,46),M4,G:GET(0
,48)-(8,58),M5,G:GET(0,60)-(8,70
),M6,G:GET(0,73)-(8,83),M7,G:GET
(0,85)-(8,95),M8,G:GET(0,96)-(8,
106),M9,G:GET(0,108)-(8,118),DM,
G:GET(0,120)-(8,130),DA,G
720 X=0:PCLS:FORR=1TO48:READJ:PO
KE1568+X,J:X=X+32:NEXT
730 DIMS1(2),S2(2),S3(2),S4(2),S
5(2):GET(0,0)-(9,9),S1,G:GET(0,1
0)-(9,19),S2,G:GET(0,20)-(9,29),
S3,G:GET(0,30)-(9,39),S4,G:GET(0
,40)-(9,49),S5,G:PCLS
740 PCLS:X=0:FORR=1TO16:READJ,K:
POKE1568+X,J:POKE1569+X,K:X=X+32
:NEXT:DIMT(7):GET(0,1)-(16,17),T
,G:PCLS:DIMO1(3),O2(3),O3(3),O4(
3),O5(3),O6(3),O7(3),O8(3)
750 DATA0,0,33,255,126,62,18,18,
0,0,0,0,0,0,34,65,42,28,0,0,16,2
4,21,19,25,21,3,1,0,0,8,28,54,28
,8,8,8,8,0,0,0,56,104,84,42,23,1
3,7,0,0,0,192,64,32,20,62,255,10
2,0,0,0,66,36,24,24,36,66,129,0,
0,0,192,176,172,150,75,36,24
760 X=0:I=1568:FORR=1TO78:READJ:
POKEI+X,J:X=X+32:NEXT:GET(0,0)-(
10,10),O1,G:GET(0,10)-(10,20),O2
,G:GET(0,20)-(10,30),O3,G:GET(0,
30)-(10,40),O4,G:GET(0,40)-(10,5
0),O5,G:GET(0,50)-(10,60),O6,G:G
ET(0,60)-(10,70),O7,G:GET(0,70)-
(10,80),O8,G:PCLS
770 LINE(31,0)-(225,41),PSET,B:L
INE(31,41)-(225,45),PSET,BF:K=33
:ST=1:POKE178,2:LINE(36,47)-(230
,47),PSET:LINE(227,5)-(231,47),P
SET,BF:LINE(36,143)-(231,148),PS
ET,BF
780 DATA252,252,192,192,192,192,
252,252,252,252,204,204,204,204,
252,252,0,0,0,0,0,0,24,24,252,25
2,192,192,192,192,252,252,252,25
2,204,204,204,204,252,252,0,0,0,
0,0,0,24,24,248,252,204,204,248,
216,204,204,204,204,204,204,204,
204,252,252,204,236,236
790 DATA252,252,220,220,204,204,
236,236,252,252,220,220,204,252,
252,192,240,240,192,252,252,248,
252,204,204,248,216,204,204,0,0,
0,0,0,0,0,24,24,192,192,192,192,19
2,192,252,252,252,252,192,240,24
0,192,252,252,198,198,238,124,56
,56,16,16,252,252,192,240,240
800 DATA192,252,252,192,192,192,
192,192,192,252,252,0,0,0,60,60,
0,0,0,252,252,192,240,240,192,25
2,252,204,236,236,252,252,220,22
0,204,252,252,192,240,240,192,25
2,252,248,252,204,204,248,216,20
4,204,252,252,204,192,220,204,25
2,252,204,204,252,252,48,48
810 DATA48,48,0,0,0,60,60,0,0,0,
3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
252,255,199,195,195,195,255,255
820 T=3401:FORR=1TO13:FORZ=1TO8:
READJ:POKET,J:T=T+32:NEXT:T=T-25
5:NEXT:T=6500:FORR=1TO6:FORZ=1TO
8:READJ:POKET,J:T=T+32:NEXT:T=T-
255:NEXT:T=6513:FORR=1TO7:FORZ=1
TO8:READJ:POKET,J:T=T+32:NEXT:T=
T-255:NEXT:T=T+1:FORR=1TO16:READ
J:POKET,J:T=T+32:NEXT
830 POKET-63,255:POKET-31,255:T=
T-30:FORR=1TO8:POKET,192:T=T-32:
NEXT:FORR=1TO8:READJ:POKET,J:T=T
-32:NEXT:PCOPY1TO3:GOTO960
840 PCOPY1TO3:PLAY"T255O1ACFDEB"
:GOTO20
850 TP=TP+1:EN=0:PMODE4,2:SCREEN
1,1:PUT(40,78)-(48,88),M7,PSET:F
ORR=1TO12:PLAY"T200O5EDCAF":POKE
6938+EN,255:EN=EN-32:FORZ=1TO100
:NEXT:NEXT:PMODE4
860 F=0:ONST GOTO870,880,890,900
,910,1220
870 PUT(90,153)-(99,162),S1,PSET
:PUT(31,0)-(225,40),VA,PSET:ST=2
:GOTO260
880 PUT(90,153)-(99,162),S2,PSET
:PUT(31,0)-(225,41),VB,PSET:ST=3
:GOTO260
890 PUT(90,153)-(99,162),S3,PSET
:PUT(31,0)-(225,19),VD,PSET:ST=4
:GOTO260
900 PUT(90,153)-(99,162),S4,PSET
:PUT(31,0)-(225,19),VC,PRESET:ST
=5:GOTO260
910 PUT(90,153)-(99,162),S5,PSET
:PUT(31,1)-(224,19),VE,PSET:ST=6
:GOTO260
920 CLS0:PRINT"YOU ARE RUNNING A
 MARATHON       THE CITY LIGHTS A
RE BEHIND YOU. YOU MUST JUMP OBS
TICALS ON THE  WAY.  IF YOU FALL
 YOU TAKE A SIPFROM YOUR CUP. IF
 YOUR CUP IS   EMPTY YOU LOSE. U
SE YOUR RIGHT  JOYSTK BUTTON TO
JUMP."
930 PRINT:PRINT"EACH NEW LEVEL (
5 IN ALL) WILL  REFILL YOUR ENER
GY CUP":PRINT"GOOD RUNNING..PRES
S ANY KEY";:EXEC41393:CLS0:GOTO9
60
940 GOTO940
950 ST=1:CLS0:PRINT@130,"YOUR EN
ERGY DRINK IS USED UP.   ANOTHER
 GAME Y/N. ?":A$=INKEY$:IFA$="Y"
THENTP=30:GOTO850ELSEIFA$="N"THE
NENDELSE950
960 '
970 POKE178,RND(255):LINE(K+RND(
10),20)-(K+RND(10)+10,20-RND(8))
,PSET,B:K=K+9:IFK<200THEN970
980 POKE178,RND(255):LINE(33,20)
-(223,20),PSET:POKE178,3
990 DATA8,8,8,8,8,8,8,8,8,8,8,7,
7,7,7,7,7,7,7,7,6,6,6,6,6,6,6,6,
6,6,6,6,6,5,5,5,5,5,5,5,5,5,5,5,
4,4,4,4,4,4,4,4,4,3,3,3,3,3,2,2,
2,2,3,3,3,3,4,4,4,4,4,4,4,4,4,5,
5,5,5,5,5,5,5,5,6,6,6,6,6,6,6,7,
7,7,7,7,7,7,8,8,8,8,8,8,8,8,8,8,
7,7,7,7,7,6,6,6,6,6,5,5
1000 DATA5,5,4,5,5,5,6,6,6,6,6,6
,6,6,6,6,6,6,7,7,7,7,7,7,7,7,7,7
,7,7,8,8,8,8,8,8,8,7,7,7,7,7,6,6
,6,6,6,6,6,6,6,6,6,6,6,6,6,7,7,7
,7,8,8,8,8,8,8,8,8,8,8,8,7,7,238
,132,228,36,228,238,170,238,172,
170,224,64,64,64,64
1010 II=32:FORI=1TO194:READJ:PSE
T(II,J):II=II+1:NEXT:PAINT(35,5)
:X=0
1020 FORR=1TO3:FORI=1TO5:READJ:P
OKE2292+X,J:X=X+32:NEXT:X=X-159:
NEXT
1030 DIMVA(204):GET(31,0)-(225,4
1),VA,G:LINE(32,1)-(224,40),PRES
ET,BF
1040 DATA8,8,8,8,8,8,8,8,8,8,7,7
,6,6,6,6,5,5,5,5,6,6,6,5,5,5,4,4
,4,4,3,3,3,3,4,4,4,5,5,5,6,6,6,7
,7,7,8,8,8,8,8,8,8,8,8,8,8,8,8,8
,7,7,7,7,6,6,6,5,5,5,5,5,6,6,6,6
,7,7,7,7,8,8,8,8,8,8,8,8,8,8,7,7
,7,6,6,6,5,5,4,4,4,3,3,4,4,4,5,5
,4,4,4,5,5,5,6,6,6,7,7,7,8
1050 DATA8,8,8,8,8,8,7,7,7,6,6,5
,5,4,4,4,5,5,5,6,6,6,7,7,7,8,8,8
,8,8,8,8,8,7,7,7,6,6,5,5,4,4,4,5
,5,6,6,5,5,5,4,5,5,6,6,7,7,7,8,8
,8,7,6,5,4,4,3,3,3,2,2,1,1,255,2
47,243,209,145,247,0,129,1,1,3,3
,7,158,254,60
1060 II=32:FORI=1TO194:READJ:PSE
T(II,J):II=II+1:NEXT:LINE(31,10)
-(225,10),PSET:PAINT(32,4):LINE(
31,18)-(225,18),PSET:PAINT(34,15
):FORX=1TO7:POKE178,RND(2):LINE(
32,9+X)-(224,9+X),PSET:NEXT:FORR
=1TO8:READJ:POKE1910+Q,J:POKE186
2+Q,J:Q=Q+32:NEXT
1070 DIMVB(204):GET(31,0)-(225,4
1),VB,G:LINE(32,1)-(224,40),PRES
ET,BF:X=0
1080 PUT(32+X,3)-(48+X,19),T,PSE
T:X=X+RND(30):IFX<175THEN1080
1090 PUT(32+Y,4)-(48+Y,20),T,PSE
T:Y=Y+RND(30):IFY<175THEN1090
1100 DIMVC(100):GET(31,0)-(225,1
9),VC,G:LINE(32,1)-(224,40),PRES
ET,BF
1110 X=0:FORR=1TO200:PSET(RND(19
4)+31,RND(10)):NEXT:FORR=1TO8:RE
ADJ:POKE1620+X,J:X=X+32:NEXT
1120 LINE(32,12)-(224,18),PSET,B
F:POKE178,3:FORR=1TO6:LINE(35+R,
19-R)-(100-R,19-R/2),PRESET:LINE
(130+R,19-R/3)-(180-R,19-RND(R))
,PRESET:NEXT
1130 DIMVD(204):GET(31,0)-(225,4
1),VD,G:LINE(32,1)-(224,19),PSET
,BF:X=0
1140 DATA0,117,69,101,69,69,0,12
7,0,21,149,85,52,21,0,255,1,213,
21,221,85,213,0,253
1150 PUT(31+X,12)-(48+X,19),T,PR
ESET:X=X+RND(20):IFX<160THEN1150
1160 X=0:FORR=1TO3:FORI=1TO8:REA
DJ:POKE1911+X,J:X=X+32:NEXT:X=X-
255:NEXT:DIMVE(95):GET(31,1)-(22
4,19),VE,G:LINE(32,1)-(224,40),P
RESET,BF
1170 DATA238,132,228,36,228,238,
170,238,172,170,224,64,64,64,64,
```

```
233,141,203,139,233,112,72,72,72
,112
1180 X=0:PMODE4,2:SCREEN1,1:LINE
(30,160)-(225,164),PSET,BF:POKE1
78,2:LINE(35,166)-(230,170),PSET
,BF:LINE(227,164)-(231,170),PSET
,BF:POKE178,3:FORR=1TO3:FORI=1TO
5:READJ:POKE7587+X,J:X=X+32:NEXT
:X=X-159:NEXT
1190 X=0:FORR=1TO2:FORI=1TO5:REA
DJ:POKE7611+X,J:X=X+32:NEXT:X=X-
159:NEXT:PUT(30,149)-(38,159),M2
,PSET
1200 TP=30:DRAW"BM78,150;D5BR48N
R4U2R4U3L4BR48R4D3NL3D2NL4BR48NU
5R4L2U2D4":GOTO850
1210 DATA192,192,192,192,192,204
,204,255,255,204,204,204,204,204
,204,204,204,204,243,251,219,219
,219,203,203,207,207,60,62,54,54
,54,50,50,51,51,207,207,204,204,
207,204,204,207,207,159,155,27,3
1,30,31,27,153,153
1220 FORT=1TO6:PMODE4:FORR=1TO9:
READJ:POKE2297+X,J:X=X+32:NEXT:E
XECA:EXECA:EXECA:EXECA:X=X-288:P
COPY1TO3:PMODE4,2:SCREEN1,1:NEXT
:POKE30011,3
1230 FORT=1TO200:PMODE4:EXECA:PM
ODE4,2:PCOPY1TO3:NEXT
1240 PCLS:AA=30:BB=50:W$="ANOTHE
R RUN   YES OR NO":GOSUB600
1250 A$=INKEY$:IFA$="Y"THENRUNEL
SEIFA$="N"THENENDELSE1250
```

■

ARTIFACT

16K ECB

UTILITY

by Justin lipton

ARTIFACT IS A very short program. All it does is display the artifact colours on the PMODE 4 screen using the '178' POKE. For those who have not seen them before it could be interesting while others may be put to sleep!

The Listing:

```
0 GOTO10
1 '********* ARTIFACT **********
2 '******* JUSTIN LIPTON *******
3 SAVE"86A:3":END
10 PMODE 4,1
20 SCREEN 1,1
30 PCLS
40 FOR A= 1 TO 255
50 POKE 178,A
60 LINE(0,0)-(255,193),PSET,BF
70 NEXT A
```

AL'S HOUSE

Continued from page 24

```
4),PSET:HLINE-(170,78),PSET
470 H$="BM151,93G2L5GFNR3GFNR3GF
NR3GFR4EU2E2":HCIRCLE(145,95),10
,1,1,.05,.90:HCIRCLE(145,90),3:H
PAINT(140,97),0,1:HDRAW H$
480 HLINE(146,84)-(147,78),PSET:
HLINE(160,85)-(160,80),PSET:HLIN
E(174,90)-(184,133),PSET:HLINE-(
166,133),PSET:HLINE-(160,112),PS
ET:HLINE-(154,133),PSET:HLINE-(1
36,133),PSET:HLINE-(143,104),PSE
T:HLINE(160,96)-(160,114),PSET
490 HLINE(148,134)-(146,150),PSE
T:HLINE-(156,150),PSET:HLINE-(16
0,115),PSET:HLINE-(164,150),PSET
:HLINE-(174,150),PSET:HLINE-(172
,134),PSET
500 HCIRCLE(150,154),5,1,.9:HCIR
CLE(140,155),5,1,.5
510 HCIRCLE(170,154),5,1,.9:HCIR
CLE(180,155),5,1,.5
520 M$="BM154,47S4C0ER2FGL2NHBR4
ERFD2FDGL3HUEU2BR4BUER2FGL2HBD8R
F2D3H3L5G3U3E2R6":HDRAW M$
530 HPAINT(152,140),13,1:'LEFT L
EG
540 HPAINT(166,140),13,1:'RIGHT
LEG
550 HPAINT(166,109),13,1:'BR COA
T
560 HPAINT(152,109),13,1:'BL COA
T
570 HPAINT(152,83),13,1:HPAINT(1
62,81),13,1:'TL & TR COAT
580 HPAINT(149,97),9,1:'HAND
590 HPAINT(159,72),12,1:'TIE
600 HPAINT(158,65),11,1:HPAINT(1
61,65),11,1:'SHIRT
610 HPAINT(154,67),10,1:HPAINT(1
65,67),10,1:'COLLAR
620 HPAINT(163,39),13,1:HPAINT(1
66,42),13,1:HPAINT(150,46),13,1:
'HAT
630 HPAINT(150,154),8,1:HPAINT(1
40,155),8,1:HPAINT(168,155),8,1:
HPAINT(178,155),8,1:'SHOES
640 HGET(133,35)-(187,161),2
650 HGET(145,35)-(175,75),3
660 RETURN
670 PALETTE RGB:PALETTE3,60
680 PALETTE4,56
690 PALETTE0,0
700 PALETTE7,60
710 PALETTE14,35:'DOOR FRAME
720 PALETTE15,0:'WINDOW BLACK
730 PALETTE13,32:'COAT, HAT AND
PANTS
740 PALETTE12,44:'TIE
750 PALETTE11,63:'SHIRT
760 PALETTE10,35:'COAT COLLAR
770 PALETTE9,61:'FACE AND HAND
780 PALETTE8,0:'SHOES
790 RETURN
800 PALETTE RGB:POKE65496,0:END
810 CLS:PRINT"YOU MADE A TYPO":P
RINT"ERROR CODE ";ERNO;" REPORTE
D":PRINT"IN LINE ";ERLIN:GOTO800
820 FOR J=0 TO 3:J1(J)=JOYSTK(J)
:NEXT
830 IF AM<=0 THEN RETURN
840 PC=0:GOSUB900
850 IF J1(0)<5 THEN PZ=PZ-1:IF P
Z<1 THEN PZ=12
860 IF J1(0)>55 THEN PZ=PZ+1:IF
PZ>12 THEN PZ=1
870 IF J1(1)<5 THEN PZ=PZ-6:IF P
Z<1 THEN PZ=PZ+6
880 IF J1(1)>55 THEN PZ=PZ+6:IF
PZ>12 THEN PZ=PZ-6
890 PC=9:GOSUB900:RETURN
900 HCOLOR PC:HCIRCLE(BP(PZ,0)+1
5,BP(PZ,1)+20),10,PC:HLINE(BP(PZ
,0)+15,BP(PZ,1)+15)-(BP(PZ,0)+15
,BP(PZ,1)+25),PSET:HLINE(BP(PZ,0
)+10,BP(PZ,1)+20)-(BP(PZ,0)+20,B
P(PZ,1)+20),PSET:RETURN
910 GOSUB280
920 FOR F=1 TO 4:GOSUB820:GOSUB2
50:IF AM<=0 THEN 230
930 IF OK THEN 170
940 NEXT F:OZ=Z:Z=10:GOSUB280:GO
TO370
950 T$="T6L4O3N1N6N9N8N6N9N6N8N6
N2N4L1N1;P4L4N1N6N9N8N6N9N6N8N6N
1O2N12N11L1N11L4P8N11O3N2N5L1N8L
4P8O2N11O3N2N5L1N6P4N4L4N2N1P4N4
N2N1P4N1O2L1N9":PLAY T$:RETURN
```

■

HOW?

Continued from page 25

```
680 S$=RIGHT$("0000"+RIGHT$(STR$
(SC),LEN(STR$(SC))-1),4):FOR V=1
 TO 4:S(V)=VAL(MID$(S$,V,1)):NEX
TV
690 SC$="BM200,125S16"+N$(S(1))+
"BR2"+N$(S(2))+"BR2"+N$(S(3))+"B
R2"+N$(S(4))
700 HCOLOR8:HDRAW OS$:HCOLOR5:HD
RAW SC$:OS$=SC$:RETURN
710 HCOLOR7:HLINE(185,10)-(320,9
0),PSET,BF:HCOLOR4:HDRAW"BM185,2
0;S31ERGNLED3NRGNU3R2U3ERGNLD7EN
U7GLU3L2D3ENU2GLU7BR7;R2NEFNED5G
L2HU5E2R2FD5GBL1U5L2D5NR2ENRU4BU
1BR4;ERGLD7E2F2EU7LGRNEND7LD5HGU
5ED5E"
720 HPAINT(250,30),1,4
730 HPAINT(285,30),6,4:HPAINT(21
0,30),6,4:HPAINT(235,30),6,4
740 RETURN
750 GX(1)=RND(10):GY(1)=RND(10)
760 GX(2)=RND(10):GY(2)=RND(10):
IF (GX(1)=GX(2) AND GY(1)=GY(2))
 THEN 760
770 GX(3)=RND(10):GY(3)=RND(10):
IF ((GX(1)=GX(3) AND GY(1)=GY(3)
) OR (GX(2)=GX(3) AND GY(2)=GY(3
))) THEN 770
780 X=RND(10):Y=RND(10)
790 FOR T=1 TO 3:IF (X=GX(T) AND
 Y=GY(T)) THEN 780
800 NEXTT:RETURN
810 FOR X=0 TO 8:READ CC:PALETTE
 X,CC:NEXT:RETURN
820 PALETTE RGB:POKE65496,0:END
```

■

ISLANDS

16K ECB
APPLICATION

by Craig Stewart

ISLANDS is a short M/L display program that gives the effect of flying at great speed past a multitude of different shaped islands in low-res. - that's all it does, so don't go grabbing your joysticks. It may have been the basis of a game, but I think it looks just okay as a display program.

PS: For those who are interested, the islands represent the CoCo's memory map. As the program passes over the memory, it will eventually hit lots of blank RAM (represented by open water or flat topped islands), and when this eventually passes, the process will repeat.

The Listing:

```
0 GOTO5
1 '********* ISLAND ************
   ****** CRAIG STEWART ********
3 SAVE"146:3":END
5 CLEAR 200, 16127
10 CLS:PRINT "loading data .. pl
ease wait..
20 LI=50:FOR X= 16128 TO  16278
STEP 70:LI=LI+10:T=0
30 READ T$:FOR Y=X TO X+69:READ
A$:A=VAL("&H"+A$):T=T+A:POKE Y,A
:NEXT Y:IF HEX$(T)=T$ THEN 40
35 PRINT "error in line ";LI:END
40 NEXT X:READ T$:T=0:LI=LI+10:F
OR Z=X-70 TO  16278:READ A$:A=VA
L("&H"+A$):T=T+A:POKE Z,A:NEXT Z
:IF HEX$(T)<>T$ THEN 35 ELSE PRI
NT:PRINT"data loading complete"
45 PRINT "press enter to save":L
INEINPUTZ$
50 SAVEM "ISLAND", 16128, 16278,
 16128 :END
60 DATA 1EB4,7E,3F,85,8E,4,0,C6,
1F,30,1,A6,84,A7,1F,5A,26,F7,C6,
80,E7,80,8C,6,0,25,EC,39,BE,3F,8
E,A6,80,BF,3F,8E,81,B9,25,9,C6,F
F,86,B7,B7,3F,96,20,14,81,3C,25,
9,C6,0,86,BF,B7,3F,96,20,7,C6,1,
86,BB,B7,3F,96,F7,3F
70 DATA 1CD1,94,B6,3F,95,BB,3F,9
4,B7,3F,95,81,2,22,2,86,3,81,10,
25,2,86,F,B7,3F,95,C6,20,3D,FD,3
F,90,BE,3F,90,30,89,4,1F,B6,3F,9
6,30,88,20,A7,84,8C,5,DF,22,4,86
,BF,20,F2,8E,5,FF,86,AF,A7,84,39
,BD,3F,1B,BD,3F,3,7E
80 DATA 1BBC,3F,85,80,E8,0,0,0,0
,0,8,0,3F,28,D6,5C,C1,8,26,4,4C,
5F,84,7,7E,9E,FD,39,9E,CF,EC,C4,
27,7,83,0,1,8D,3,1F,21,39,34,76,
6R,64,A6,63,3D,ED,66,EC,61,3D,EB
,66,89,0,ED,65,E6,E4,A6,63,3D,E3
,65,ED,65,24,2
90 DATA 234,3F,85,80,E8,0,0,0,0,
0,8,0
```

JOYSTICK SIMULATOR

16–64K ECB
UTILITY

by John Carmichael

JOYSTICK SIMULATOR allows you to simulate a joystick by using the arrow keys to move, and the space bar to fire. It can be used in two ways

1. To speed up games currently using INKEY$ or PEEKs to read the keyboard.
2. To convert joystick dependent games to keyboard driven games.

It should work on either the 16K or 64K CoCo. It is a position independent machine language program, which checks all the arrow keys and the space bar. Three keys can be held down at once. After doing an EXEC JJ the horizontal movement will be placed in variable H.

If left, H=0.
If nothing, H=32.
If right, H=63

The vertical movement will be placed in variable V.

If up, V=0.
If nothing, V=32.
If down, V=63

Fire button (space bar) will be placed in variable FX.

If nothing, FX=32.
If fire, FX=0.

All three variables will change depending on which keys are being held down.

Add these lines to the program to watch it work ...

```
10 GOSUB 10000
20 ?@32,H;V;FX:EXEC JJ
 :GOTO20
```

The utility replaces the following line in a joystick program:

```
10 H=JOYSTK(0):V=JOYSTK(1)
 :FX=PEEK(65280)
```

In the fire routine, replace

```
IF FX=126 OR FX=254 THEN ...
```

... with ...

```
IF FX=0 THEN ...
```

The variable names can be changed (see below) but they must be initialised before doing EXEC JJ, failure to do so results in an ?FC ERROR.

To initialise, make sure your program starts with a GOSUB 10000. Put any DIMs immediately after the CLEAR. If the original program CLEARs space for one of it's own machine programs, then do it's CLEAR before doing the GOSUB10000. Line 10000 causes the machine code to be located under any existing machine code.

Changing the variable names

To change the variable names so that they suit any particular program, change line 10030. Single letter variables must have the CHR$(0) added to them, as all variable names are in fact two bytes long in CoCo's memory.

If the simulator doesn't work it may be that you have a conflict with other DATA statements in the program, especially if a RESTORE command exists.

The Listing:

```
0 GOTO10000
3 SAVE"85:3":END
10000 JJ=PEEK(39)*256+PEEK(40)-1
13:CLEAR200,JJ:FX=0:V=0:H=0:JJ=P
EEK(39)*256+PEEK(40)+1
10010 DATA 0,0,0,0,0,0,0,33,8C,F
0,31,8C,F4,9E,1B,EC,84,26,3,7E,B
4,4A,10,A3,C4,26,E,86,86,A7,2,DC
,8A
10020 DATAED,3,ED,5,33,42,AF,A1,
30,7,6D,C4,26,E0,C6,FB,30,8C,CD,
58,5C,F7,FF,2,B6,FF,0,43,48,27,2
,8D,5,C1,7F,26,EB,39,C1,7F,27,19
,30,2,C1,EF,25,6,C1,F7,27,F,20,6
,30,2,C1,DF,27,7,EE,84,86
10025 DATA 7C,A7,43,39,EE,84,6F,
42,39
10030 FIRE$="FX":VERTICAL$="V"+C
HR$(0):HORIZONTAL$="H"+CHR$(0) '
VARIABLE NAMES
10035 NAMES$=FI$+VE$+HO$:V=1:FOR
N=JJ TOJJ+5:POKEN,ASC(MID$(NA$,V
,1)):V=V+1:NEXT
10040 FORN=JJ+6 TOJJ+110:READA$:
POKEN,VAL("&H"+A$):NEXT:JJ=JJ+13
:RETURN    'EXEC JJ TO SIMULATE
JOYSTICK
10050 'NOT PRESSED- FX=32 SPACE
      V=32    H=32
10060 'PRESS- FX=0 SPACE, H=0 _
H=63 RIGHT, V=0 ^, V=63 DOWN
```

TANDY...®

Save $50

Convenient Computing With Our Multi-Pak Interface

Computer Not Included

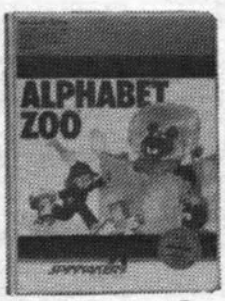
129.95 Reg 179.95

The Multi-Pak Interface enables you to connect up to four Program Pak® cartridges to your Color Computer at once! Instead of tedious plugging and unplugging you can change cartridges under program control or with selector switch. 26-3124

Software for Fun and Education from the Tandy Library

A. 59.95

B. 59.95



C. 29.95 Reg 69.95

D. 21.95 Reg 49.95

E. 19.95 Reg 34.95

F. 19.95 Reg 34.95

A. Fraction Fever. Doubles as both a fun game and educational tool. Children hop along on a pogo stick to match correct fractions and zap incorrect ones 26-3169

B. Kindercomp is a compilation of learning tasks. Eg. match shapes, write names, and draw pictures. Kids develop various skills 26-3168

C. Alphabet Zoo encourages children to associate letters of the alphabet with the sounds they represent in 2 exciting maze games 26-3170

D. You must take control of **The Reactoid** in order to contain the atoms and radiation released as a result of a fusion reactor meltdown 26-3092

E. Taxi. A team oriented game in which the responsibility of earning a fare is shared. You drive anywhere from New York to Shanghai 26-2509

F. Children learn to analyse the things they see with **Ernie's Magic Shapes** by matching shapes and recognizing similarities and nuances 26-2524

Star Trap is a dynamic maze game where the challenge is to trap a shooting star by blocking the paths and using special maze gates 19.95 | Reg 34.95 26-2510

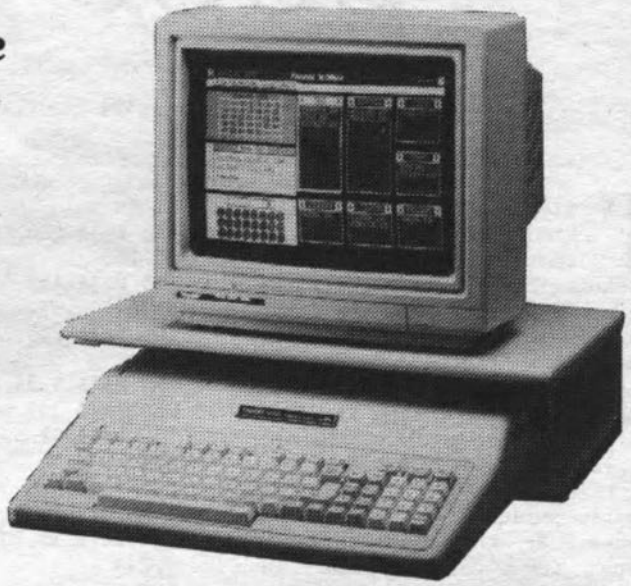
* Some may need recorder and/or joysticks.

Better Again™

The Tandy 1000 EX for Excellence in Home and School Computing

$1499

Monitor and Stand Not Included



The Tandy 1000 EX is our lowest priced MS DOS based Personal Computer. The 1000 EX has a clock speed of 7.16/4.77 Mhz which is 50% faster than the IBM PC ™. Includes a 13.3cm disk drive and 256K RAM that is expandible to 640K. A viable educational aid! 25-1050

CM-5 RGBI Color Monitor. A 33cm screen displays 80 × 25 text and 320 × 200 graphics. Ideal for elementary color graphics. 25-1023 **$599**

TM - Trademark International Business Machines Corporation.

Tandy 1000 SX — the Optimal Way to Business Efficiently

$2299

Monitor Not Included

Tandy 1000 SX. With the 8088 chip at it's core you are assured of compatibility with industry standard MS DOS software and with a selectable 4.77 or 7.16 Megahertz clock speed you'll have your work done faster than ever. Tandy 1000 SX has a 384K RAM that's expandible to 640K and 2 built-in 13.3cm drives. 25-1051

CM-10 RGBI Color Monitor. A 33cm VDU with 640 × 200 high resolution graphics. 25-1022 **$899**

Welcome to the Next Generation of Personal Computing

WE SERVICE WHAT WE SELL!

Tandy ELECTRONICS

A DIVISION OF TANDY AUSTRALIA LIMITED INC IN N.S.W.

Nearly 350 Stores Australia-Wide

Available From 350 Stores Australiawide Including Tandy Computer Centres

or Order On VIATEL *642614#

Independent Tandy Dealers may not be participating in this ad or have every item advertised. Prices may also vary at individual Dealer Stores

Every picture tells a story

COCO 3 SCAN

for Amateur radio enthusiasts only!

by G. Thurston

THIS PROGRAM will receive black and white pictures from amatuer radio transmissions. It uses the hi-res 640x225x4 screen. There are 24 levels of grey. It will only work on the CoCo 3.

Type in "CC3SCAN" and save it. Type "TVC351/BAS", save it. Then run it,. to produce the ML program. When "CC3SCAN" is RUN, it loads the "TVC351/BIN" program produced.

When RUN, the menu appears. There are four speeds of transmission available from the menu from 7 seconds to 36 seconds per frame. By changing the brightness, you can often get a better picture from an off frequency transmission. The Sync crossing level can be changed as well.

Most slow-scans transmissions can be found on 14.230 MHz upper side-band. Tune in the voice, and the slow-scan should then be tuned properly. Tape the slow-scan audio, and you can look at the pictures at a later date by playing them into the computer again.

The only extra needed is a cable from the speaker output of the receiver to the black cassette plug into the computer.

The Listing:

```
0 GOTO10
1 'RUN THIS PROGRAM FIRST TO GET
   A BINARY FILE CALLED 'TVC351'
   THEN RUN '3SCAN' (THE NEXT
   PROGRAM).
2 'IF YOU ARE RUNNING A TAPE-
   BASED PROGRAM, CHANGE LINE 40
   TO READ:
  40 CSAVEM"TVC351",30000,&H77EE
,&H7532
3 SAVE"158AA:3":END
10 CLEAR200,29999
20 FORA=30000TO&H7EE
30 READP:POKEA,P:NEXT
40 SAVEM"TVC351",30000,&H77EE,&H
7532
50 END
100 DATA 116,99,142,0,1,48,31,30
110 DATA 136,38,250,26,80,134,25
5,183
120 DATA 255,2,134,76,183,255,14
4,134
130 DATA 59,183,255,165,16,255,1
17,48
140 DATA 126,181,83,134,48,142,2
55,160
150 DATA 167,128,76,129,53,38,24
9,16
160 DATA 206,149,47,22,0,43,70,2
2
170 DATA 0,34,0,0,0,160,0,0
180 DATA 140,160,140,160,4,0,1,0
190 DATA 92,39,21,16,33,255,249,
181
200 DATA 255,32,39,244,92,39,9,1
6
210 DATA 33,255,237,181,255,32,3
8,244
220 DATA 57,26,80,48,141,2,24,19
1
230 DATA 254,245,134,126,183,254
,244,182
240 DATA 255,33,132,254,183,255,
33,182
250 DATA 255,35,132,254,183,255,
35,134
260 DATA 2,183,255,147,134,0,183
,255
270 DATA 2,127,255,32,182,255,14
7,28
280 DATA 191,134,136,183,255,152
,134,125
290 DATA 183,255,153,134,92,183,
255,144
300 DATA 127,255,154,183,255,217
,134,192
310 DATA 183,255,157,127,255,158
,134,0
320 DATA 183,255,176,134,7,183,2
55,177
330 DATA 134,56,183,255,178,134,
63,183
340 DATA 255,179,182,255,35,138,
8,183
350 DATA 255,35,182,255,3,132,24
7,183
360 DATA 255,3,182,255,1,138,8,1
83
370 DATA 255,1,51,141,255,88,142
,0
380 DATA 0,236,141,255,93,49,139
,16
390 DATA 175,74,31,16,163,70,31,
1
400 DATA 175,72,236,70,174,72,48
,139
410 DATA 172,74,16,36,1,32,49,13
9
420 DATA 16,175,72,182,255,0,132
,64
430 DATA 16,39,1,65,111,69,95,13
4
440 DATA 1,230,68,111,68,23,255,
48
450 DATA 231,66,225,196,16,34,1,
8
460 DATA 134,22,167,65,134,4,167
,78
470 DATA 109,79,16,38,0,175,109,
200
480 DATA 16,16,38,0,203,109,200,
17
490 DATA 16,38,0,134,230,66,235,
69
500 DATA 231,69,16,43,0,161,230,
66
510 DATA 192,45,193,24,47,4,198,
24
520 DATA 32,3,33,1,18,193,0,42
530 DATA 3,95,32,4,33,247,33,245
540 DATA 49,141,1,40,88,49,165,1
66
550 DATA 164,167,132,166,33,52,2
,31
560 DATA 16,227,70,31,2,53,2,167
570 DATA 164,48,1,172,74,16,39,0
580 DATA 157,172,72,16,39,0,107,
166
590 DATA 69,160,67,167,69,42,21,
32
600 DATA 31,111,79,108,200,16,19
8,0
610 DATA 235,68,247,117,106,30,1
36,30
620 DATA 136,22,254,189,198,9,23
5,68
630 DATA 231,68,18,18,33,246,32,
150
640 DATA 198,12,235,68,231,68,18
,16
650 DATA 140,170,170,16,140,170,
170,22
660 DATA 255,77,106,200,17,230,6
8,203
670 DATA 7,231,68,18,18,22,255,6
3
680 DATA 230,68,203,3,231,68,16,
140
690 DATA 170,170,22,255,75,230,6
8,203
700 DATA 5,231,68,18,22,255,40,2
30
710 DATA 68,203,8,231,68,30,136,
22
720 DATA 255,29,230,68,203,15,23
1,68
730 DATA 108,200,16,30,136,22,25
5,15
740 DATA 230,68,203,6,231,68,109
,65
750 DATA 22,255,4,236,72,163,70,
31
760 DATA 1,230,68,203,8,231,68,3
0
```

Continued on page 37

Have fun with...

COCO OSCILLASCOPE

16K ECB + EDTASM+ UTILITY/HARDWARE MOD. by D. Thurston

THIS IS A little program that is sure to make a few people happy. There has been a lot of interest shown in a program to make the computer into an oscillascope. This program does just that. There is a capacitor, a connector, two resistors, and a cable needed to interface to the joystick port.

There is an assembly listing included which should help others to understand a bit more about it. It's fully commented, so I'll just give an outline of the program here.

The first attempt used a real time effort which would erase the last trace, and put the new one up at the same time. However it was too slow to be of use, so this one stores a line of samples, and then displays them. This appears real-time, but isn't quite.

The 5V on the joystick port is divided in half by the two resistors to start in the middle. The program then waits for a negative to positive crossing.

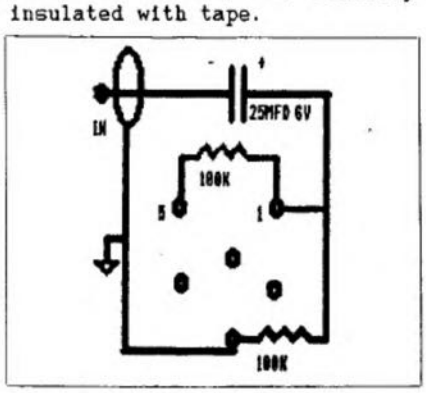
Then the PIA is tested. If it is too high, the digital to analogue converted is increased one step, and the PIA bit is tested again. This process is repeated many times, and the DAC tends to track the voltage on the joystick port.

When the end of the allotted memory is reached, the samples are put to screen. Because the DAC can only be increased or decreased one step at a time, the rise time is limited - therefore, it oversamples, and will show a fast rise better.

The original program used a 256 PMODE4, but it was slow to reach the end of the screen, and to keep the pixels from seperating, it only converted a small amount of the screen with the scan. The cursor PMODE gives a more spectacular display.

The program is written in BASIC, and the ML is poked into memory for convenience. Be sure to save it before running, as it might run rampant over your hard work if there is an error.

The circuit diagram is included, and if small components are used, will fit into the DIN plug neatly. They must, of course, be suitably insulated with tape.



A word of caution!!! The input will not handle voltages over 5 volts, peak to peak.

It is only meant for audio frequencies, and is suitable for the audio out of a radio, amplifier, tape recorder, etc. Just be sure to keep the audio down until you see it on the screen. If there are strange looking lines from the top of the trace, the volume is too high.

The frequency response is limited, but it will give a fair representation of the signals, if care is taken to adjust the oversampling to suit the frequency and rise time. This is not a laboratory-quality instrument, but it can be a lot of fun.

I have a successive approximation routine in mind, and time permitting, will write a program using the same interfacing.

Type in the program, save it, and when it's working, there will be a trace across the screen. There must be an AC input, or the screen will remain blank until the signal is input.

The routine checks the keyboard for an input at the end of each screen. It will ask for the oversampling delay. A higher number will help it follow a faster rise time but will lower the speed. Experiment.

Have fun!

```
00100 ***************************
00110 *******SCOPE***************
00120 ***************************
00130          ORG  13616          PROGRAM BEGINS AT 13616
00140 DAC      EQU  COMPAR+$20     ADR OF DAC PIA
00150 COMPAR   EQU  $FF00          ADR OF COMPARATOR PIA
00160 BEGSCR   RMB  2              SAVE CALCULATED ADR OF BEGINNING OF SCREEN
00170 EOS      RMB  2              SAVES ADDR TO END OF SCREEN CLEARING
00171 EOL      RMB  2              HOLDS COMPUTED END OF LINE
00180 BIT      FCB $80             SET MSB, WHICH IS ROTATED THROUGH BYTE
00190 DELAY    RMB  1              THIS BYTE SETS PRINT DELAY
00200 DELAY2   RMB  1              THIS BYTE TIMES THE DELAY
00210 START    LDA  $BC            GET START OF GRAPHICS PAGE
00220          CLRB MUL            IT BY 256
00225          INCA                ADD 256 TO CENTER THE SCREEN
00230          STD BEGSCR          SAVE CALCULATED SCREEN START
00240          ADDD #$400          NUMBER OF BYTES TO CLEAR
```

```
00250          STD  EOS          SAVE CALCULATED END OF SCREEN
00251          LDD  BEGSCR       GET SCREEN START
00252          ADDD #$10         COMPUTE END OF LINE
00253          STD  EOL          SAVE IT
00260          ORCC #$50         MASK INTERUPTS FOR BETTER TIMING
00270          LDA  #128         SETS DAC TO HALF VOLTAGE
00280          LDB  #128         SETS MSB WHICH IS USED TO TEST COMPARATOR BIT
00290          STA  DAC          INIT DAC
00300          LDX  #BUFF        POINT TO START OF BUFFER
00310          LDU  #COMPAR      POINT TO COMPARATOR
00320          LDA  DELAY        GET DELAY CONSTANT
00330          STA  DELAY2       INIT DELAY
00340          LDA  #128         START SAME AS DAC
00350          POS  BITB,U       TEST COMPARATOR
00360          BNE  POS          WAIT UNTIL SIGNAL GOES NEGATIVE
00370          NEG  BITB,U       TEST COMPARATOR
00380          BEQ  NEG          WAIT UNTIL SIGNAL GOES POSITVE
00390          ATOD BITB,U       START A TO D ON POSITVE SLOPE. TEST COMPARATOR
00400          BNE  HIGHER       IF BIT SET, ADD 4 (ONE STEP OF DAC)
00410          SUBA #4           BIT NOT SET, SO SUBTRACT 4
00420          STA  DAC          DECRIMENT DAC
00430          BRA  SAVE
00440 HIGHER   ADDA #4           BIT SET, SO ADD 4
00450          STA  DAC          INC DAC
00460          BRA  SAVE
00470 SAVE     STA  ,X+          STORE CURRENT LEVEL
00480          CMPX #$4000       TEST FOR END OF SAMPLE
00490          BNE  ATOD         GO READ SOME MORE
00500          LDX  BEGSCR       POINT TO START OF SCREEN
00510          CLR  A            CLEAR ALL BITS
00520 CLS      STA  ,X+          CLEAR PORTION OF SCREEN USED
00530          CMPX EOS          TEST FOR END OF LINE
00540          BNE  CLS          GO CLEAR SOME MORE
00550          LDX  BEGSCR       POINT TO BEGINNING OF SCREEN
00560          LDA  DELAY        INIT DELAY
00570          STA  DELAY2
00580          LDY  #BUFF        POINT TO BUFFER
00590 LOOP     LDA  ,Y+          GET READING
00600          LSRA              DIVIDE BY 2
00610          LSRA              DIVIDE BY 2
00620          LDB  #$10         NUMBER OF BYTES PER LINE
00630          MUL               CALCULATE OFFSET
00640          LEAU  D,X         ADD OFFSET
00650          LDA  ,U           GET BYTE
00660          ORA  BIT          SET BIT
00670          STA  ,U           PUT IT ON SCREEN
00680          DEC  DELAY2       TEST FOR END OF DELAY
00690          BNE  LOOP         GO SET ANOTHER PIX ON THIS X COORDINATE
00700          LDA  DELAY        INIT DELAY AGAIN
00710          STA  DELAY2
00720          ANDCC #$FE        CLR CARRY BIT FROM CC REGISTER
00730          ROR  BIT          MOVE TO NEXT X POSITION
00740          BCC  LOOP         NOT READY FOR NEXT BYTE YET
00750          ROR  BIT          CARRY SET - ROTATE IT INTO MSB
00760          LEAX 1,X          POINT TO NEXT BYTE
00770          CMPX EOL          TEST FOR END OF LINE
00780          BNE  LOOP         GO SET MORE
00790          JSR  [$A000]      TEST FOR KEY ENTRY
00791          LBEQ START        GO DO IT AGAIN
00792          RTS               RETURN TO BASIC IF KEY PRESSED
00800 BUFF     FCB  1            MARK START OF BUFFER
00810          END  START        TELL ASSEMBLER TO END PROGRAM AND EXEC AT START.
```

DETACH

32K DECB
UTILITY
D.W Thurbon

THE PROGRAM switches off the disc rom and with a press of the reset button returns you to extended basic interal rom. This means that you no longer need to remove the disc controller to return to normal extended basic.

Therefore, no wear and tear is placed upon the controller.

The Listing:

```
0 GOTO10
3 SAVE"24:3":END
10 ' DISC CONTROLER "DETACH"
     (C) PIXEL SOFTWARE. 1985.
      BY D. W. THURBON.
20 CLS:CLEAR200,&H6FFF
30 POKE&H7000,&H12
40 FORN=1TO16
50 POKE&H7000+N,PEEK(&HA073+N)
60 NEXT
70 POKE&H7000+17,&H8E
80 POKE&H7000+18,&H7F
90 POKE&H7000+19,&HFE
100 FORN=20TO78
110 POKE&H7000+N,PEEK(&HA073+N+1
2)
120 NEXTN
130 FORN=1TO167
140 POKE&H7000+78+N,PEEK(&H8001+
N)
150 NEXTN
160 POKE&H7000+246,&H7E
170 POKE&H7000+247,&H80
180 POKE&H7000+248,&HB0
190 POKE&H72,&H70
200 POKE&H73,&H00
210 CLS3:PRINT@64,"   PRESS RESE
T TO DETACH DISK  "
```

■

Hint...

CoCoMax Fix

Want to run CoCoMax on the CoCo 3? Resave your CoCoMax system program again and you'll have no problems!

This is how you do it:

```
LOADM"COCOMAX/SYS":SAVEM"COCOMAX
/SYS",&HE00,&H18F0,0
```

COCO 3 SCAN

Continued from page 34

```
770 DATA 136,33,240,22,254,241,1
08,79
780 DATA 22,254,62,32,249,22,254
,57
790 DATA 106,78,106,65,16,39,255
,97
800 DATA 109,78,38,156,111,200,1
6,172
810 DATA 72,16,37,255,206,230,68
,203
820 DATA 10,231,68,134,6,167,200
,17
830 DATA 109,71,22,254,165,26,80
,182
840 DATA 255,147,134,59,183,255,
163,126
850 DATA 119,138,16,254,117,48,1
42,2
860 DATA 24,191,254,245,134,24,1
83,254
870 DATA 244,127,255,147,134,56,
142,255
880 DATA 160,167,128,76,140,255,
168,38
890 DATA 248,189,246,121,28,175,
57,127
900 DATA 255,147,48,141,255,199,
175,97
910 DATA 59,126,119,125,255,255,
255,254
920 DATA 239,254,239,238,238,238
,174,238
930 DATA 174,234,170,234,170,170
,170,169
940 DATA 154,169,154,153,153,153
,89,153
950 DATA 89,149,85,149,85,85,69,
84
960 DATA 69,84,68,68,4,68,4,68
970 DATA 0,64,0,0,0,0,0,0


1 WIDTH80
2 GOTO5
3 SAVE"158A:3":END
5 IFPEEK(&H7530)=116THEN25
10 CLEAR200,29999
20 LOADM"TVC351"
22 POKE&HFF40,0
25 SYNC=&H7566:SPCING=&H7569:BRI
GHTNESS=&H7679:KEYENABLE=&H75B5
26 ONBRKGOTO1000
30 CLS
40 PRINT"SPEED
50 PRINT"1 -12"
60 PRINT"2 -24
70 PRINT"3 -36
80 PRINT"7 -7
90 PRINT"8 -8
92 PRINT"C -CLEAR RUNFLAG"
93 PRINT"S -CHANGE SYNC LEVELS"
94 PRINT"W -WAIT FOR SYNC"
95 PRINT"B -CHANGE BRIGHTNESS"
100 A$=INKEY$:IFA$=""THEN100
101 IFA$="C"THENPOKE30069,0:GOTO
175
102 IFA$="S"THENGOSUB1010:GOTO17
5
103 IFA$="W"THENPOKE30069,1:GOTO
175
104 IFA$="B"THENCLS:PRINT"BRIGHT
NESS NORMALLY 45;BRIGHTNESS NOW
IS";PEEK(&H7679):INPUT"NEW BRIGH
TNESS";B:POKE&H7679,B:GOTO175
110 K$="12378A"
120 I=INSTR(1,K$,A$)
130 IFI=0THEN30
140 FORA=1TOI
150 READP
160 NEXT
170 POKE&H7569,P
175 FORA=0TO100:NEXT
180 EXEC&H7532
185 PALETTERGB:POKE&HFFD8,0
190 CLEAR:GOTO25
200 DATA54,54,86,30,34
300 'SYNC AT &H7566
310 'SPCING AT &H7569
320 'BRIGHTNESS AT &H7678
330 'KEYBOARD LINE ENABLE AT &H7
5B5
1000 PALETTERGB:POKE&HFFD8,0:STO
P
1010 CLS:PRINT"SYNC NORMALLY 70.
 SYNC = ";PEEK(&H7566):INPUT"NEW
 SYNC";S:POKE&H7566,S:RETURN
```

■

TAPE CHECKER

16K ECB
UTILITY
by Allan Thompson

TAPECHECKER IS A small utility designed to verify whether or not the programs on your tape have been saved properly or not.

All that is required of you is to enter the number of programs on that particular tape.

After it has checked that number of files, it will return an 'ok' message.

```
0 GOTO10
1 REM   TAPE FILE CHECKER
2 REM   BY ALLAN THOMPSON
3 REM   17 LARKDALE CRES.
4 REM    O'HALLORAN HILL
5 REM   SOUTH AUST.  5158
8 SAVE"100:3":END
10 CLS8:PRINT@39,"TAPE FILES CHE
CKER";:PRINT@103,"BY  ALLAN THOM
PSON";:PRINT@193,"ENTER NUMBER O
F FILES ON TAPE";:PRINT@289,"ELS
E ENTER 36, & <BREAK> STOP";:C$=
CHR$(255):PRINT STRING$(48,C$);:
INPUTN:CLS0
20 Y=1:FORX=1TO N:PRINT@Y,X;:SKI
PF:PRINT@Y+3,"checked";
30 Y=Y+16:NEXTX:END
40 KILL"TAPECHKR/BAS":SAVE"TAPEC
HKR/BAS":END
```

How to get large numbers of FIELDS in a Disk Record

32K Disk ECB
UTILITY

by John Carmichael

THIS ARTICLE IS orientated to those programmers using direct access files, who want to create much larger records with lots more FIELDs than they can currently get.

The OPEN "D",#1,"ACCOUNTS/DAT" defaults to a record size of 256. You can gain a larger record size by using the FILES command. It must come before the OPEN.

If unspecified, FILES defaults to two buffers, with a total of 256 bytes. FILES 2,900 however, would give you two buffers with a combined size of 900 bytes. This would allow you to have for example, two records, the first 20 bytes and the second up to 880 bytes long.

E.G.;

```
10 FILES 2,900
20 OPEN"D",#1,"COMPANY/DAT",20
30 OPEN"D",#2,"FIGURES/DAT",880
```

The problems come when you would like to have say, 88 fields of 10 bytes each in the large record. As best as I can make out, this is impossible given the way the FIELD statement works.

All FIELD specifications have to be made in a single line of BASIC. The limit of 250 bytes to a line of BASIC, means you can have a maximum of 22 fields if you also wanted the convenience of arrays.

A statement like the one below doesn't work.

```
40 FOR N=1TO88:FIELD#2,10AS
F$(N) :NEXT
```

It results in a single field of length 10 being allocated to F$(88).

So, for example, in Graham's ACCOUNTS Spreadsheet in Feb, 1986 and May CoCo, he is limited to 15 transaction types.

The following program is designed to overcome this limitation of the FIELD statement by creating a BASIC line as long as is necessary to specify all your field requirements.

Whilst the BASIC input buffer only allows you up to 250 characters per line, execution of the BASIC program is not dependent on any particular line length. (See the articles in JULY 1986 CoCo, p.28 for a fuller explanation.)

This utility should be typed up seperately and SAVEd as an ASC file for future MERGEing. (If you have B-DOS you can eliminate GOSUB10045 by using PPEEK())

When typing the FIELD statement in your program keep going until you can fit no more AS s e.g.

```
40 FIELD#2,10ASF$(1),10ASF$(2)
  ,10ASF$(3), .. etc,  ASF$(22)
```

Then keep typing on the next line as if you were continuing, being sure to start the line with a comma.

```
50 ,10ASF$(23),10ASF$(24),
10ASF$(25), ..etc, 10ASF$(44)
```

Continue doing this on successive lines until you are finished.

Now MERGE the utility, and RUN 10000. Answer the line number question with the number of the FIELD statement that needs extending, in the above example, 40.

The program will reply with LINE 50 ADDED TO LINE 40. Repeat the procedure until all the AS lines are "lumped together" on line 40. When done, DEL10000- to remove the utility. When you LIST 40 you will see only the first 255 characters, the rest of the line will be hidden. Once this has been done you cannot EDIT the line without chopping its length back down to normal.

This method still requires a lot of typing to create a lot of FIELDs, but at least they can now be created!

HOW IT WORKS:

Subroutine 10045 gets the value of the two bytes starting at V, and returns that value in V.

Subroutine 10050 points S to the address of the first byte in the line number specified.

A = the value for the FIELD token as it is stored in BASIC
P = the address of the start of the line which was just searched
B = the address of the start of the next line

Line 10040 gets the address for the line following the one to be concatonated, and stores it in the next-line-pointer of the line with the FIELD statement. It then places 5 spaces where the old BASIC pointers were. That is, it joins the two lines together.

COPY ROM

64K ECB

UTILITY

by D.W. Thurbon

COPY ROM COPIES ROM into high ram (64K only) and allows the use of the ram and is disk compatible. It does this to allow use of the memory locations between 49152 to 65279 for ML programs. Just LOAD and RUN the program then load your ML program into that area and GO FOR IT.

The Listing:

```
0 GOTO10
3 SAVE"24A:3":END
10 ' COPYROM. BY D. W. THURBON.
     CURTESY OF PIXEL SOFTWARE.
     PUBLIC DOMAIN SOFTWARE.
20 FORX= 16384TO 16402:READA:POK
EX,A:NEXT:END
30 DATA 142, 128, 0, 127, 255, 2
22, 236, 132, 127, 255, 223, 237
, 129, 140, 254, 254, 37, 241, 5
7
```

The vagaries of the variable

VARIABLE LISTER

16K ECB
UTILITY

by Russ Nelson

WHEN DEVELOPING BASIC programs or modifying existing BASIC, it is helpful to know what variables have been used and where they have been used in the program. Some variable listers have been published but none have satisfied me.

This effort is a modification of a program by John Carmichael published in August 1985 Australian Rainbow Magazine. It is saved in ASCII (on the monthly CoCoOz tape) and can be accessed anytime by RUN 50000. Make sure the program to be searched has not got any lines above 50000.

When RUN it will display the first line number of each variable, and store all the next lines that variable appears on. After all lines have been searched, the full list can be displayed on screen or printed in order of line number, or variable name. A list of REM line numbers, bytes used by spaces and total number of program lines is also printed.

Being BASIC, this program is slow so I have used the speedup POKE to save time. If your computer will not accept fast speed, delete line 50040. With fast speed it takes 10 minutes to scan 150 lines, but after saving the variables the display or printing is without delay.

The Listing:

```
0 END
1 '********** VARILISTER ******
3 SAVE"197:3":END
50000 'VARIABLE LISTER**MERGE ON
TO END OF PROGRAMME AND RUN 5000
0
50010 'AUS RAINBOW AUGUST 85 BY
JOHN CARMICHAEL
50020 'MODIFIED BY RUSS NELSON N
OVEMBER 1986
50030 PCLEAR1:CLEAR2000:DIMA$(10
0),RE$(70):CLS:C=0:PRINT"LINE VA
RIABLE
50040 POKE65495,0
50050 S=PEEK(25)*256+PEEK(26)+2
50060 NLA=PEEK(S-2)*256+PEEK(S-1
):IFNLA=0 THEN50280 ELSELS=LS+1
50070 LN=PEEK(S)*256+PEEK(S+1):N
=S+2:IFLN=50000 THEN50280
50080 A=PEEK(N):IFA=0 THENS=N+3:
GOTO50060 ELSEIFA=134 THENS=NLA+
2:GOTO50060' DATA
50090 IFA=34 THENGOSUB50200:GOTO
50080 ELSEIFA=32 THENSP=SP+1
50100 IFA=130 ORA=131 THENRR$(R)
=STR$(LN):R=R+1:L=L+NLA-N-1:S=NL
A+2:GOTO50060
50110 IFA>90ORA<65 THENN=N+1:GOT
O50080
50120 A$="":GOSUB50230
50130 FORB=1TOC:LV=LEN(A$):LA=LE
N(A$(B))
50140 L$=STR$(LN):Z=LEN(L$)
50150 IFA$+" "<>LEFT$(A$(B),LV+1
) THEN50180
50160 IF INSTR(1,A$(B),L$)=0THEN
A$(B)=A$(B)+L$
50170 B=98
50180 NEXTB:IFB=99 THEN50080 ELS
EC=C+1:GOSUB50400
50190 PRINTA$(C):GOTO50080
50200 N=N+1:IFPEEK(N)=34 ORPEEK(
N)=0 THEN50210 ELSE50200
50210 IFPEEK(N)=0 THEN50080 ELSE
N=N+1
50220 RETURN
50230 A$=A$+CHR$(A):N=N+1:A=PEEK
(N)
50240 IFA<90ANDA>47 THENIFA<58OR
A>64 THEN50230
50250 IFA=40 THENA$=A$+"(":N=N+1
:RETURN
50260 IFA=36 THEN50230
50270 RETURN
50280 POKE65494,0:PRINT"PRINTOUT
/SCREEN (P/S) ":PRINT
50290 A$=INKEY$:IFA$="" THEN5029
0 ELSEIFA$="P" THEND=-2 ELSED=0
50300 PRINT"SORTED/UNSORTED (S/U
) ":PRINT
50310 A$=INKEY$:IFA$="" THEN5031
0 ELSEIFA$="S" THEN50330 ELSE503
20
50320 FORZ=1TOC:PRINT#D,"    "A$
(Z):NEXTZ:GOTO50390
50330 A=65
50340 FORZ=1TOC
50350 IF LEFT$(A$(Z),1)=CHR$(A)
THENPRINT#D,"    "A$(Z)
50360 NEXTZ
50370 A=A+1
50380 IF A=94 THEN50390 ELSE5034
0
50390 PRINT#D,CHR$(13)"REM LINES
"TAB(13);:FORB=0TOR:PRINT#D,RE$(
B);:NEXT:PRINT#D,CHR$(13)L"BYTES
 USED BY REMARKS"CHR$(13)SP"BYTE
S USED BY SPACES"CHR$(13)LS-1"PR
OGRAMME LINES":GOTO50280
50400 B$="":FORX=LV TO6:B$=B$+"
":NEXTX
50410 A$(C)=A$+B$+L$:RETURN
```

■

CORRECTION

DISASSEMBLER

Ah, the joys of missing listings. We've accidentally left out part of the listing to "Disassembler" (March 1987 CoCo, page 8) by Wilfred Tritscher.

Sorry about any inconveniences, folks! Here is the rest of that listing:

```
271 'STRING$

272 ZS$="":FOR ZJ=1TO ZL:ZS$=ZS$
+CHR$(ZC):NEXT:RETURN

275 'HEX-DEC HX$=HEX(HX)

276 HX$=""

278 Z2=INT(HX/16):Z1=HX-Z2*16:HX
$=MID$("0123456789ABCDEF",Z1+1,1
)+HX$:IF Z2=0 THEN RETURN ELSE H
X=Z2:GOTO278

279 'DEC - HEX  ZN=VAL(ZH$)

280 ZN=0:ZF=1:FOR ZJ=1TO LEN(ZH$
):ZM$=MID$(ZH$,LEN(ZH$)+1-ZJ,1):
IF ASC(ZM$)>64 THEN ZI=ASC(ZM$)-
55 ELSE ZI=VAL(ZM$)

282 IF ZI>15 THEN PRINT"FC ERROR
":STOP ELSE ZN=ZN+ZI*ZF:ZF=ZF*16
:NEXT:RETURN
```

■

They don't write songs like this anymore

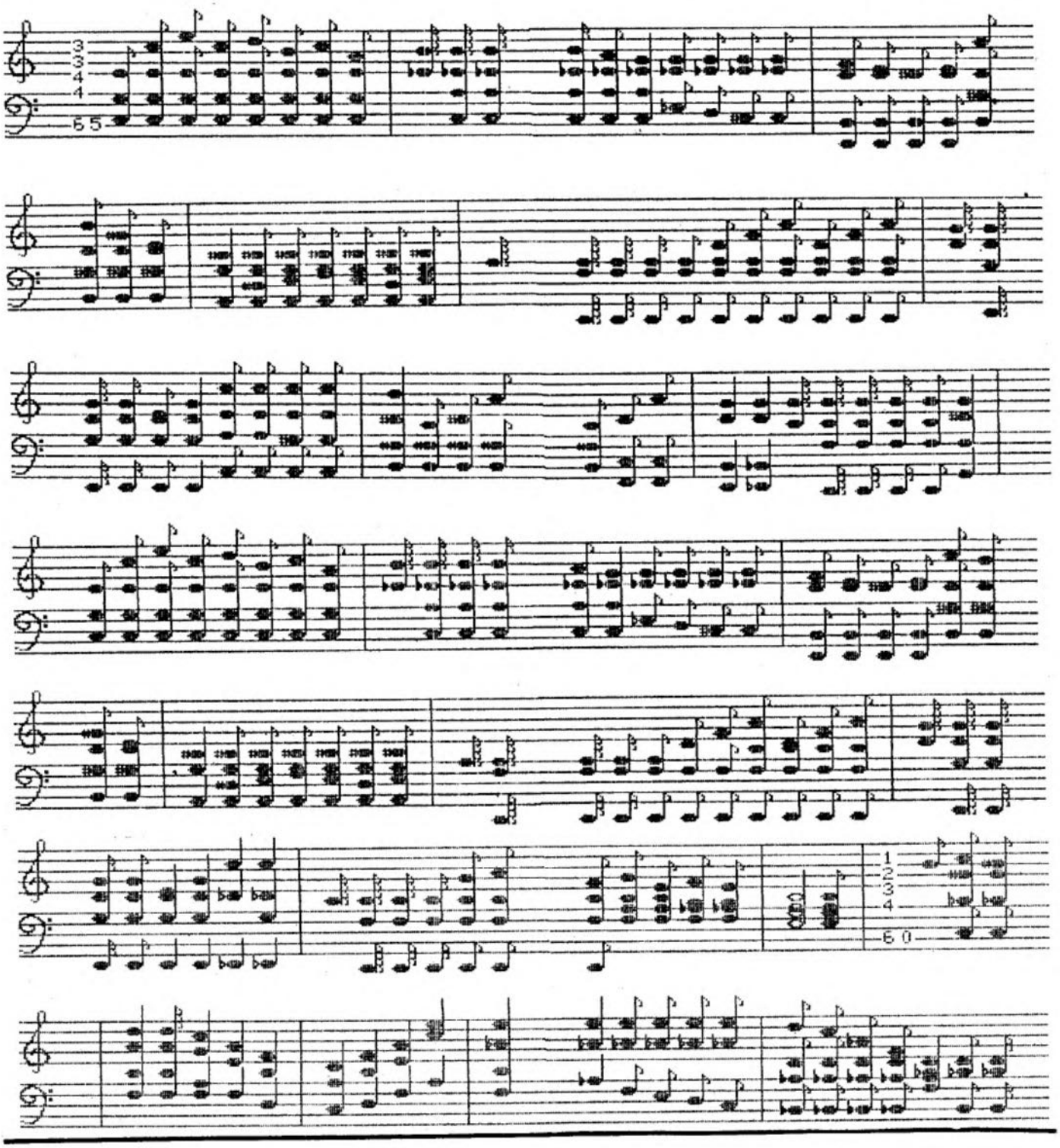
STARDUST

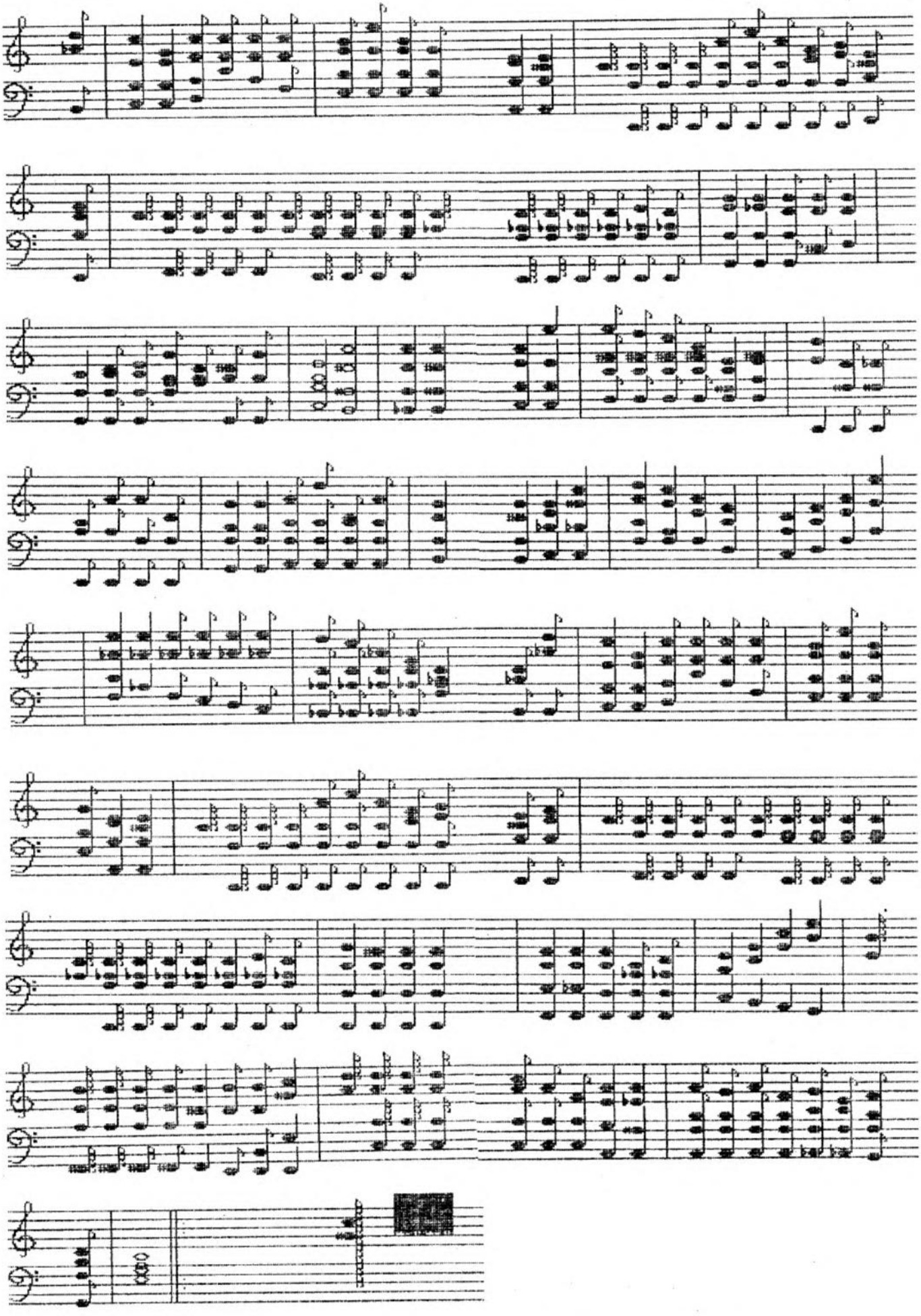
CoCo 1/2 & Musica II
MUSIC

by Harvey Smith

STARDUST IS YET ANOTHER one of the excellent entrants to the music competition. The original artist of the song was Hoagy Carmichael and was cleverly compiled by Harvey Smith using the Musica II program.

Note: It is possible to play this piece using Music (original) for those people who get the CoCoOz on Disk monthly.





For those who need one, here's a ...

DATABASE

32K ECB tape only
BUSINESS

(and Sticky Label Utility)

by Michael J. Hartmann

IT WAS ONLY until recently that the only type of cockatoo seen on our property were the ones that flew over the New England farm every spring in search of feed. August 1984, however, saw the emergence of a new breed of white cockatoo.

My parents had been shopping for computers for quite a while. They were aware that computers were progressing toward becoming an important piece of farm equipment and to stay in the farm game one must move with the times. Finally, a cassette - based 64K ECB CoCo 2 was purchased. It was then up to me to learn and ultimatly design and write programs for use on the family farm.

At that time I was a year 10 student and so had plenty of time to spend learning the new language. Computer programming became my hobby and after about a year I felt that my 'finer' results were worthy of submittion to the crew of 'Aussie Rainbow'.

Prospectivley it was time to think about farm programs. I approached the situation systematically, initially asking myself, "How could CoCo make farm-life easier?"

For an answer I reviewed the farm calender. I decided that the foremost use would be as a source for quick, easy and efficient storage and retrieval of various types of information.

"A DATABASE", thought I. Enthusiastically I charged through the pages of'Rainbow' and 'CoCo' searching for such a program. It was then that I really felt the devestating effects of the rural crisis.

The shortage of money means that no disk-drive will, for the meantime at least, supersede the humble cassette-player.

So the databases I did find just took too long to load and even when they were loaded many didn't satisfactorily fulfill my requirements. So, I was faced with having to write my own database.

When I write a program I like to initially review other authors programs and then modify or adapt them to suit my needs.

The end result should be short, simple and efficient (sounds a lot like a girl I know!!).

'LABKMAKR' (CoCo Nov'86) is an example of the result of this approach. If a program is too fancy I find that the time taken to load it in is often greater than the time taken to perform the job manually. Length must be kept to a minimum.

The database I wrote (LISTING 1) fulfills the above criteria. It is a simple shell-type program and can be modified with little fuss and bother.

Once the ball was rolling there seemed no stopping it. Needs were appearing faster than were the solutions. A new set of stockyards were to be built, 'GRAFPLOT' (CoCo Nov'86) was born. Although a bit clumsy, it was a far better system of designing the yards than we'd ever had before.

All was going well, and I was really progressing with this application programming. But this was all to change. The ball that had been rolling so well fell into a hole. Year 12 and the HSC stopped it in its tracks. My programming was sacrificed for study. The Coco, however, was not left to gather dust. The printer (DMP110) and word processor (TELE64) purchased earlier in 1986 meant that I could write essays and assignments quicker than ever before. The teachers couldn't even complain about my handwriting, although some typo's led to some remarks!

With the welcome demise of my school-life, I have been active with my computing. There are indeed many needs to be covered.

For example, the CoCo has now entered the fire-fighting business. The system is now utilised to print reminders to all members of the local Bushfire Brigade, TELE64 records the minutes of general meetings, my database stores the names and numbers of members and my short, simple and efficient program 'STIKLABL' (Listing 2) addresses stickers for mailing envelopes.

So, effective business computing CAN be carried out more than adequatly by the CoCo, even with a simple cassette system. The possibilities are almost endless, from data storage to word processing and beyond. Technology on the farm is ever-changing, and in modern agriculture where speed is a vital factor, the CoCo is proving itself a worthy addition to the farming enterprise.

The Listing:

```
0 '*****************************'
1 '**********DATABASE**********'
2 '****GENERAL SHELL PROGRAM****'
3 '****BY MICHAEL J HARTMANN****'
4 '*******MOUNT VIEW GUYRA******'
5 '************NSW 2365*********'
6 '*****************************'
7 GOTO10
8 SAVE"194:3":END
9 '
10 GOTO35
11 CSAVE"DATABASE"
12 I$=INKEY$:IFI$=""THEN12
20 '
25 '  -=>64K CLEAR:CHANGE?<=-
30 '
35 CLEAR10000:DIM S$(2000)
40 '
45 '-=>MENU FROM J.CARMICHAEL<=-
50 '
55 CLS:PRINT"MENU":PRINT:PRINT"1
: REPLCE DATA"
60 PRINT"2: CREATE FILE":PRINT"3
: APPEND FILE"
65 PRINT"4: SAVE DATA":PRINT"5:
LOAD DATA":PRINT"6: PRINT DATA":
PRINT"7: SORT DATA":PRINT:PRINT"
PRESS ENTER TO MAKE SELECTION
ARROWS OR NUMBERS TO MOVE"
70 D=1:X=64
75 GOSUB115
80 K$=INKEY$:IFK$=""THEN80
85 K=VAL(K$):IFK>7THEN80 ELSE GO
SUB115
90 IF K$=""ANDD>1 THEND=D-1:K=D
:GOTO110
95 IFK$=CHR$(10)ANDD<7THEND=D+1:
K=D:GOTO110
100 IF K$=CHR$(13)THEN ON D GOTO
 140,195,205,255,325,415,545
105 IF K=0THEN75
110 D=K:X=64+(K-1)*32:GOTO75
```

```
115 FORN=1024+X TO 1037+X:IFPEEK
(N)>63 THEN A=-64 ELSEA=64
120 POKEN,PEEK(N)+A:NEXT:RETURN
125 '
130 ' -=> REPLACE ROUTINE <=-
135 '
140 N=0
145 CLS:PRINT"REPLACE DATA":PRIN
T
150 PRINT"PRESS <ENTER> WHEN DON
E"
155 PRINT:INPUT"FILE NO. TO REPL
ACE";N
160 IF N=0 OR N>B THEN 55
165 PRINT S$(N)
170 INPUT"REPLACEMENT";S$(N)
175 GOTO140
180 '
185 '-=>CREATE/APPEND ROUTINE<=-
190 '
195 B=0:CLS
200 B=B+1
205 CLS:PRINT@0,"INPUT/ADD ITEMS
":PRINT@22,"MEM";MEM
210 PRINT"PRESS <ENTER> WHEN DON
E"
215 PRINT:PRINT""B;
220 INPUT S$(B)
225 IF S$(B)=""THEN 55
230 B=B+1
235 GOTO205
240 '
245 ' -=> SAVE ROUTINE <=-
250 '
255 CLS
260 PRINT"SAVE DATA ONTO TAPE":P
RINT:PRINT:PRINT"READY TAPE-PRES
S PLAY/REC,"
265 INPUT"FILENAME";F$
270 PRINT@138,"-CHECK COUNTER,"
275 PRINT@170,"-PRESS <ENTER>."
280 I$=INKEY$:IF I$=""THEN280
285 OPEN "O", #-1, F$
290 FOR X=1 TO B-1
295 PRINT#-1, S$(X)
300 NEXT X
305 CLOSE #-1:GOTO55
310 '
315 '-=> LOAD ROUTINE <=-
320 '
325 CLS:PRINT"LOAD DATA FROM TAP
E":PRINT
330 INPUT"FILENAME=";F$
335 PRINT"READY TAPE-PRESS PLAY"
340 PRINT@106,"-PRESS <ENTER>"
345 I$=INKEY$:IF I$=""THEN345
350 AUDIOON
355 OPEN "I", #-1, F$
360 B=1
365 IF EOF(-1) THEN 390
370 INPUT #-1,S$(B)
375 PRINT S$(B)
380 B=B+1
385 GOTO365
390 CLOSE #-1
395 PRINT@256,"ANY KEY FOR MENU"
:EXEC44539:GOTO55
400 '
405 '-=> PRINT ROUTINE <=-
410 '
415 CLS:PRINT"PRINT ROUTINE":PRI
NT@160,"":INPUT"SCREEN OR PRINTE
R <S/P>";R$
420 IF R$<>"S" AND R$<>"P" THEN
415
425 IF R$="P" THEN495
430 '
435 '*SCREEN
440 '
445 FOR X=1 TO B-1 STEP15
450 FOR Z=X TO X+13
455 PRINT Z; S$(Z)
460 NEXT Z
465 PRINT"any key to continue":E
XEC44539
470 NEXT X
475 GOTO55
480 '
485 '*PRINTER
490 '
495 FOR X=1 TO B-1 STEP 15
500 FOR Z=X TO X+10
505 PRINT#-2,S$(Z)
510 NEXT Z
515 PRINT"any key to continue":E
XEC44539
520 NEXT X
525 GOTO55
530 '
535 '-=> SORT ROUTINE <=-
540 '
545 CLS:PRINT"SORT ROUTINE":PRIN
T@160,"":INPUT"ALPHABET/NUMERICA
LLY <A/N>";R$
550 IF R$<>"A" AND R$<>"N" THEN5
45
555 FOR I=B-1 TO 1 STEP -1
560 FOR J=1 TO I
565 IF R$="A" THEN 595
570 IF S$(J)<=S$(J+1) THEN 655
575 GOTO640
580 '
585 '*ALPHABETICALLY
590 '
595 FOR C=1 TO LEN(S$(J))
600 IF MID$(S$(J),C,1) < "A" THE
N 630
605 FOR C1=1 TO LEN(S$(J+1))
610 IF MID$(S$(J+1),C1,1) < "A"
THEN 625
615 IF MID$(S$(J),C,LEN(S$(J))-C
+1) <= MID$(S$(J+1),C1,LEN(S$(J+
1))-C1+1) THEN 655
620 GOTO 640
625 NEXT C1
630 NEXT C
635 GOTO 655
640 SV$ = S$(J)
645 S$(J)=S$(J+1)
650 S$(J+1)=SV$
655 NEXT J
660 NEXT I
665 FOR I=1 TO B:PRINT S$(I):NEX
T I
670 GOTO 55
```

■

The Listing:

```
0 '*****************************'
1 '********** STIKLABL *********'
2 '***** BY M.J.HARTMANN *****'
3 '**** MOUNT VIEW, GUYRA ****'
4 '********** NSW 2365 *********'
5 '*****************************'
6 GOTO20
7 '
8 SAVE"194A:3":END
9 I$=INKEY$:IFI$=""THEN9
10 '
15 '<<<= INPUT DATA ROUTINE =>>>
20 CLS:S=0
25 LINEINPUT "INPUT name (32 LET
TERS MAX)      ";N$
30 LINEINPUT "INPUT address
               ";A$
35 LINEINPUT "INPUT town
               ";T$
40 LINEINPUT "INPUT state AND po
stcode         ";SP$
45 INPUT"HOW MANY LABELS";L
50 '<<= LIST DATA AND CHECK =>>'
55 CLS
60 PRINT"NAME= ";N$:PRINT
65 PRINT"ADDRESS= ";A$:PRINT
70 PRINT"TOWN= ";T$:PRINT
75 PRINT"STATE AND POSTCODE= ";S
P$:PRINT
80 INPUT"  is this correct <Y/N>
";C$
85 IF C$<>"N" THEN 155
90 IF C$="N" THEN INPUT"WHICH IS
 WRONG <1/2/3/4>";WR
95 ON WR GOSUB 110,120,130,140
100 GOTO55
105 '     <<= CHANGE NAME =>>
110 LINEINPUT"INPUT NEW NAME
               ";N$:RETURN
115 ' <<= CHANGE ADDRESS =>>
120 LINEINPUT"INPUT NEW ADDRESS
               ";A$:RETURN
125 '    <<= CHANGE TOWN =>>
130 LINEINPUT"INPUT NEW TOWN
               ";T$:RETURN
135 '<<=CHANGE STATE AND CODE=>>
140 LINEINPUT"INPUT NEW STATE AN
D POSTCODE     ";SP$:RETURN
145 '
150 '
155 IFPEEK(65314)/2<>INT(PEEK(65
314)/2)THENPRINT"PRINTER NOT REA
DY!":INPUT"WHEN READY,PRESS <ENT
ER>";PR$
160 IFPEEK(65314)/2<>INT(PEEK(65
314)/2)THEN155
165 PRINT#-2,"":PRINT#-2,"":PRIN
T#-2,""
170 '
175 PRINT#-2,"";N$
180 PRINT#-2,"";A$
185 IF LEN(T$)+LEN(SP$)+12 >30 T
HEN X=30-LEN(T$)-LEN(SP$) ELSE X
=12
190 PRINT#-2,"";T$+STRING$(X," "
)+SP$
195 S=S+1:IF S=L THEN 20 ELSE 15
5
200 END
```

■

For the mathematician

ANOTHER PIE

32K ECB
APPLICATIONS

by Jim Jacobs

BILL BARDEN'S article "Pi to 10 000 Digits" in the May RAINBOW intrigued me. The ability of a home computer to do things which were once so difficult is fascinating.

As Bill mentions in his article it is possible to improve and speed up the program. This prompted me to try and the one I ended up with is nearly 800 times faster.

While the calculation of Pi to many decimal places may not have much direct application, my work on the program has helped me understand better how computers do arithmetic and to utilise the capabilities of the 6809 (the CPU chip) of the CoCo more effectively.

As well the Basic program to calculate Pi ("NUMBERPI") I have included the machine language part of the program with notes which should be useful to those who wish to delve deeper.

LIMITATIONS

Checking a few numbers gives this timing formula for the program ..

Time = 1.05 D*D seconds

... where D is the number of decimal places divided by one hundred - just under three hours for 10 000 places. I cannot see any more ways to increase the speed (except the high speed poke), but no doubt some ingenious reader will.

About 416 bytes are required for each 1000 decimal places held in multibyte variables and as two are needed a 16K machine could handle almost 20 000 places!

There is no allowance for overflow in the odd number division routine which limits the number of decimal places to about 20 000. Otherwise two byte arithmetic could handle up to 40000 places in about 50 hours on a 32K COCO!

IMPROVEMENTS

I imagine the multibyte variables as strings of bytes stretching from left to right the first two bytes are the integer part and the rest are the fractional part.

I used Bill Barden's program as a guide or check list and made the following changes to speed up the calculation:

1. By multiplying by the argument the series becomes ...

$$\text{arctan1/x} = x \cdot \sum_{i=1}^{\infty} (-1)^{(i-1)} / (2i-1) \cdot x^{2i}$$

... avoiding the initial division by x.

2. There is no need to prescale the variables. The sum of the series ...

Pi/4 = 4.arctan1/5 - arctan1/239

... is a fraction less than one and after the initial multiplication by 4 to get Pi; repeated multiplication by 10 will produce the digits to the left of the decimal point one by one.

3. Only two multibyte variables are required. One for the sum of the series (the sum variable) and the other for the powers of (1/5**2) or (1/239**2) (the power variable).

4. Shifting all the bits of the power variable during division is not neccessary. Each byte of the quotient ends up in the same position and so can be simply produced from left to right.

5. During division it is only neccessary to shift one byte at a time from the dividend into the remainder even for word (two byte) division. This gives a 30% speed up.

6. Only one byte division is required for 5**2 giving a further 25% speed up.

7. The two divisions and the summation can be done at the same time. As the words of the power variable are produced they are divided by the odd numbers and incorporated in the series. The operations can be imagined proceeding from left to right.

8. As the process goes on the numbers get smaller. Leading zeros appear in the power variable allowing the length to be reduced. Also this allows the summation to be terminated when the values are small enough there is no need to count the bytes.

9. Division can be speeded up by adding the negative of the divisor instead of subtracting a carry is generated when the quotient bit is one and can be shifted into the quotient; at the same time a one is shifted through the quotient to count the bits.

LISTINGS

The Basic program "NUMBERPI" is given in Listing 2. It contains a machine language routine for the actual calculations this is given in assembly language in Listing 1.

When "NUMBERPI" is RUN the machine language program which is held in the data lines is poked into memory just above the Basic area. This takes a few seconds.

Then Pi can be calculated to a chosen number of decimal places and the results displayed one line at a time. The display can be repeated or a new number of places chosen.

The assembly language used in Listing 1 is to explain the machine language and would need to be adapted to suit the assembler being used.

The addresses given are notional only the program is automatically located just above the Basic program.

Note that the code is self-modifing ...

- The two series use different power division routines the branch offset at 0E56 changes.
- The series summation is addition and subtraction for alternate terms. The operations

are changed at addresses 0E5B & 0E65 .

- The odd number divisor at addresses 0E3E & 0E43 is changed for each term of the series.

These zero page addresses are used:

```
0040-1  ST  Sum pointer  (start of sum variable)
  42-3  PT  Power pointer  (..... power ...)
  44-5  END  End of multibyte storage
  46-7  Q   Odd division remainder
    48  FLG  Hi/lo byte flag
    49  N   Power division quotient
  4A-B  R   Power   ..    remainder
  4C-D  M   Odd     ..    quotient.
```

Listing 1 Pi Calculation

```
        Divide by 239*239
0E00 9749  :  2ND   ST A N     count & quotient
   2 DC4A  :         LD D R     get remainder

   4 6884  :  LP2   ASL ,X      shift dividend
   6 5949  :        ROL D       into remainder
   8 2405  :        BCC UNDER   overflow?
   A 83DF21:        SUB D'239*239'  generate
   D 2008  :        BRA BT2       carry

   F C320DF: UNDER ADD D'-239*239'  subtract
  12 2503  :        BCS BT2
   4 8320DF:        SUB D'-239*239' correction

   7 0949  :  BT2   ROL M      bit into quotient
   9 24E9  :        BCC LP2     & count

   B DD4A  :         ST D R    replace remainder
   D 9649  :         LD A M    get quotient
   F 2010  :        BRA ODD

        Divide by 5*5
  21 D64A  :  1ST   LD B R     remainder

   3 6884  :   LP   ASL ,X     shift dividend
   5 59    :        ROL B       into quotient
   6 CBE7  :        ADD B '-25'   subtract
   8 2502  :        BCS BT
   A C0E7  :        SUB B '-25'  correction

   C 49    :    BT  ROL A     bit into quotient
   D 24F4  :        BCC LP      & count


   F DD49  :         ST D M,Rh save qtnt & rmdr

        Odd number division
  31 A780  :  ODD   ST A ,X+     result byte
   3 8601  :        LD A '1'     counter
   5 974D  :        ST A M1       quotient
   7 DC46  :        LD D Q       remainder

   9 0849  :  DLP   ASL M       shift dividend
   B 5949  :        ROL D        into remainder
   D C30000:        ADD D -ODD1   subtract
  40 2503  :        BCS BIT
   2 830000:        SUB D -ODD2  correction

   5 094D  :  BIT   ROL M1    bit into quotient
   7 24F0  :        BCC DLP     & count

   9 DD46  :         ST D Q     save remainder
   B 0A48  :        DEC FLG      hi / lo?
   D 2B0B  :        BMI OUT

   F 964D  :         LD A M1    move hi byte
  51 974C  :         ST A Mh    in quotient

      Begin division routine
   3 8601  : PDVM   LD A '1'    counter
   5 2000  :        BRA 1ST/2ND   CA/A9

      Summation
   7 0048  :  OUT_  NEG FLG      reset
   9 ECC4  :         LD D ,U    get sum
   B 004C  :        OPD1 M      +/- result
   D EDC1  :         ST D ,U++   replace
   F 240B  :        BCC TERM
  61 315E  :        LEA Y -2,U

   3 ECA3  :CARRY    LD D ,--Y
   5 000001:        OPD2 '1'    +/- carry
   8 EDA4  :         ST D ,Y
   A 25F7  :        BCS CARRY

   Do all words for each term of the series
   C 9C44  : TERM  CMP X END      term end?
   E 25E3  :        BCS PDVM
```

```
  Continue until the terms are small enough
  70 301C  :        LEA X -4,X
   2 9C42  :        CMP X PT    end of series?
   4 256D  :        BCS RTN

        Next term
   6 8D65  :        BSR CLEAR      remainders
   8 AE8CC3:         LD X -ODD1,P
   B 8640  :         LD A'change'
   D A88CDB:        EOR A OPD1,P   +/-

  80 A78CD8: INTO    ST A OPD1,P   direct
   3 8010  :        SUB A '16'
   5 A78CDD:         ST A OPD2,P  immediate
   8 301E  :        LEA X -2,X     next odd
   A AF8CB1:         ST X -ODD1,P
   D AF8CB3:         ST X -ODD2,P

  90 DE40  :         LD U ST     reset
   2 9E42  :         LD X PT      pointers
   4 EC84  :         LD D ,X     zero?
   6 26BB  :        BNE PDVM

      Remove leading zeros
   8 3002  :        LEA X 2,X
   A 9F42  :         ST X PT
   C 3342  :        LEA U 2,U
   E DF40  :         ST U ST
  A0 20CA  :        BRA TERM

    Start here  4.arctan1/5 first
   2 8D2A  :START   BSR RESET

   4 ED83  :  BCLR   ST D ,--X     clear
   6 9C40  :        CMP X ST          both
   8 26FA  :        BNE BCLR       variables

   A CCCA14:         LD D'1st,4*5'  offset,dvdnd
   D 8ED301:         LD X'Add,1'   addition

  B0 8D0E  :        BSR SERIES

        Then do -arctan1/239
   2 8D1A  :        BSR RESET

   4 ED83  :  PCLR   ST D ,--X     clear
   6 9C42  :        CMP X PT         power
   8 26FA  :        BNE PCLR       variable

   A CCA9EF:         LD D'2nd,239'  offset,dvdnd
   D 8E9301:         LD X'Sub,1'   subtraction

  C0 A78C93: SERIES ST A PDVM+3,P
   3 B741  :         ST B 1,U  initial dividend
   5 1F10  :        TFR X D
   7 D748  :         ST B FLG
   9 8E0001:         LD X '1'   for first odd
   C 20B2  :        BRA INTO

      Setup the pointers etc.
   E 338C77: RESET LEA U START,P  of variables
  D1 DF40  :         ST U ST       sum pointer
   3 EC5E  :         LD D -2,U     byte count
   5 33CB  :        LEA U D,U     end of sum
   7 DF42  :         ST U PT      power pointer
   9 30CB  :        LEA X D,U
   B 9F44  :         ST X END     of variables

      Remainders
   D 5F4F  : CLEAR CLR D
   F DD4A  :         ST D R
  E1 DD46  :         ST D Q
   3 39    :        RTS


    Conversion and display routines
   4 8D88  :ENTRY   BSR RESET

   6 ECC3  :  TRFR   LD D ,--U   transfer sum
   8 ED83  :         ST D ,--X   for conversion
   A 9C42  :        CMP X PT
   C 26F8  :        BNE TRFR

   E 8D48  :        BSR x2
  F0 8D46  :        BSR x2       multiply by 4

   2 8607  :         LD A '7'    1st line
   4 8C    :        CMP X         skip 2

      For each row after the first
   5 8608  :  NEXT   LD A '8'    line count
   7 C611  :         LD B'Add4'   x5 count
   9 DD48  :         ST D CT8,CT4

   B 8DD1  :        BSR RESET

   D 8603  : GROUP   LD A '3'
   F 9747  :         ST A CT3

 0F01 9F44  :  ZERO   ST X END     remove
   3 EC83  :         LD D ,--X    trailing
   5 27FA  :        BEQ ZERO       zeros

   7 6F41  : DIGIT CLR 1,U
   9 8D2D  :        BSR x2   10 = 2 x 5

   B EC84  :    x5   LD D ,X
```

```
   D 8384  :  ADD   ADD D ,X
   F 2406  :        BCC SKIP
  11 3184  :        LEA Y ,X

   3 6CA2  :  C'RY  INC ,-Y
   5 27FC  :        BEQ C'RY

   7 0849  :  SKIP  ASL CT4
   9 24F2  :        BCC ADD

   B 0C49  :        INC CT4     reset
   D ED81  :         ST D ,X++
   F 9C44  :        CMP X END
  21 25E8  :        BCS x5

   3 A641  :         LD A 1,U
   5 8B30  :        ADD A '0'    ascii
   7 BDA30A:        JSR DISPLAY
   A 0A47  :        DEC CT3
   C 26D9  :        BNE DIGIT

   E 8620  :         LD A " "
  30 BDA30A:        JSR DISPLAY
   3 0A48  :        DEC CT8
   5 26C6  :        BNE GROUP
   7 39    :        RTS

   8 9E44  :   x2   LD X END
   A 4F    :        CLR A       carry 0

   B 47    : LOOP   ASR A       restore carry
   C 6982  :        ROL ,-X
   E 6982  :        ROL ,-X
  40 49    :        ROL A       save carry
   1 9C42  :        CMP X PT
   3 26F6  :        BNE LOOP
   5 39    :        RTS

   6 0000  :    number of bytes
   8  >>>>> start of variable area
```

The Listing:

```
1 PRINT@489,"NUMBERPI":PRINT:PRI
NT@492,"BY JIM JACOBS 27FE87"
2 GOTO10
3 SAVE"198:3":END
10 PRINT"POKING":A=(PEEK(31)+2)*
256:FORA=A TOA+325:READB:POKEA,B
:NEXT
20 INPUT"NEW - N OR DISPLAY - D"
;A$:IFA$="D"THEN50ELSEIFA$<>"N"T
HEN20
30 INPUT"NUMBER OF DECIMAL PLACE
S";N:IFN<1ORN>9999THEN30
40 B=2*INT(N/4.82)+6:POKEA,INT(B
/256):POKEA+1,B-INT(B/256)*256:E
XECA-164
50 PRINT"END - E  OR ANY KEY TO
GO ON            PI TO"N"PLACES":P
RINT"  3.";:EXECA-98:B=N-21
60 A$=INKEY$:IFA$=""THEN60ELSEIF
A$="E"THEN20
70 EXECA-81:B=B-24:IFB<0THEN20EL
SE60
100 DATA151,73,220,74,104,132,89
,73,36,5,131,223,33,32,8,195,32,
223,37,3,131,32,223,9,73,36,233,
221,74,150,73,32,16,214,74,104,1
32,89,203,231,37,2,192,231,73,36
,244,221,73
101 DATA167,128,134,1,151,77,220
,70,8,73,89,73,195,,,37,3,131,,,
9,77,36,240,221,70,10,72,43,8,15
0,77,151,76,134,1,32,,,72,236,19
6,,76,237,193,36,11,49,94,236,16
3,,,1,237,164,37,247
102 DATA156,68,37,227,48,28,156,
66,37,109,141,101,174,140,195,13
4,64,168,140,219,167,140,216,128
,16,167,140,221,48,30,175,140,17
7,175,140,179,222,64,158,66,236,
132,38,187,48,2,159,66,51,66,223
,64,32,202
```

Continued on page 52

SNAKES and LADDERS

CoCo 3 Program

GAME

by Charles Bartlett

THIS IS PLAYED by the traditional rules with the following changes in the interest of more INTEREST. The player that starts first is selected by the highest throw, not the first to throw a six.

Two tokens are NOT allowed to occupy the same square, if a move would land you on the opponents square you will be placed on square back.

This applies in ALL cases, including going up ladders and down snakes, (you don't get put back down the ladder, just moved back one space at the top of the ladder).

An EXACT throw is required to reach 100. An extra turn is awarded for each SIX that is thrown. The computers messages, token and dice are all one colour, your's are all another colour.

Press any key to start your dice rolling, press any key to stop it rolling ... the number showing when you press a key is NOT the number that the dice will stop at, it stops at the next number ... just in case anybody thought of cheating, not that you would ... would you?

The Listing:

```
0 GOTO10
3 SAVE"168A:3":END
10 '    SNAKES AND LADDERS
   (C) CHARLES BARTLETT  1/1/87
20 CLEAR2000:ON BRK GOTO930
30 POKE65497,0::HSCREEN2:GOSUB94
0:HBUFF 1,500:HBUFF 2,500:DIM M(
8,1),XA(11),YA(11),BP(100,1):PLA
Y"O3T255L255":FOR X=1 TO 8:READ
M(X,0),M(X,1):NEXT:DATA 8,35,22,
87,48,70,63,82,92,19,84,68,60,28
,51,7

40 N$(0)="BRGD4FR2EU4HBR2":N$(1)
="BR2NGD6NLRBU6BR2":N$(2)="BDER2
FDG4R4BU6BR1":N$(3)="BDER2FDGNLF
DGL2HBR5BU5":N$(4)="BD3NR4E3ND6B
R2":N$(5)="NR4D3R3FDGL2HBR5BU5":
N$(6)="BDNED4FR2EUHNL3BU3NL2NFBR
2":N$(7)="BD6UE4UNL4BR"
50 BX$="URDL":N$(8)="BRGDFNR2GDF
R2EUHEUHNL2BR2":N$(9)="BD5FR2EU4
HL2GDFR3BU3BR1":LR$="NH3BEHE2FBE
H3BGFG2HBG":LL$="NE3BFEF2GBFE3BH
GH2EBH":LE$="EBE2EBG4":RE$="FBF2
FBH4":GOTO100
60 SX$="":SC$="":S$=RIGHT$("0000
"+RIGHT$(STR$(SC),LEN(STR$(SC))-
1),LS):FOR V=1 TO LS:S(V)=VAL(MI
D$(S$,V,1)):NEXTV
70 SC$="BM"+PO$+SS$:FOR V=1 TO L
S:SX$=SX$+N$(S(V))+"BR2":NEXT:SC
$=SC$+SX$:IF OK THEN HCOLOR0:HDR
AW OS$
80 HCOLOR FG:HDRAW SC$:OS$=SC$:R
ETURN
90 HCOLOR3:Z=1:FOR X=0 TO 250 ST
EP 25:HLINE(X,0)-(X,190),PSET:XA
(Z)=X:Z=Z+1:NEXT:Z=0:FOR Y=0 TO1
90 STEP 19:HLINE(0,Y)-(250,Y),PS
ET:YA(Z)=Y:Z=Z+1:NEXT:RETURN
100 HCIRCLE(200,94),90,1,1,.50,.
75:HCIRCLE(200,94),80,1,1,.50,.7
5:HCIRCLE(200,10),10,1,.66,.75,.
25:HCIRCLE(50,96),60,1,1,.99,.25
:HCIRCLE(50,96),70,1,1,.99,.25:H
CIRCLE(50,160),25,1,.22,.25,.75:
HPAINT(50,160),1,1
110 HCIRCLE(188,100),40,1,2,0,.2
5:HCIRCLE(190,100),50,1,1.7,0,.2
5:HCIRCLE(234,101),8,1,2,.50,0:H
CIRCLE(190,180),30,1,.10,.25,.75
:HPAINT(234,101),1,1
120 HCIRCLE(190,100),180,8,.24,.
25,.50:HCIRCLE(192,100),170,8,.2
2,.25,.50:HCIRCLE(16,100),8,8,2.
4,.50,0:HCIRCLE(190,140),3,8,1,.
75,.25:HPAINT(190,140),8,8
130 HCIRCLE(90,60),100,9,.33,.75
,0:HCIRCLE(90,60),90,9,.30,.75,0
:HCIRCLE(90,31),10,9,.44,.25,.75
:HCIRCLE(185,60),4,9,2.4,0,.50:H
PAINT(185,60),9,9
140 HDRAW"BM180,180S12C10":HDRAW
 LE$:FOR L=1 TO 5:HDRAW LR$:NEXT
:HDRAW LE$:HPAINT(178,174),10,10
150 HDRAW"BM34,140C10":HDRAW RE$
:FOR L=1 TO 13:HDRAW LL$:NEXT:HD
RAW RE$:HPAINT(38,138),10,10
160 HDRAW"BM190,100":HDRAW RE$:F
OR L=1 TO 4:HDRAW LL$:NEXT:HDRAW
 RE$:HPAINT(195,100),10,10
170 HDRAW"BM60,70":HDRAW LE$:FOR
 L=1 TO 4:HDRAW LR$:NEXT:HDRAW L
E$
180 HPAINT(60,66),10,10
190 GOSUB90
200 FG=6:LS=2:XA=1:YA=9:FOR X=1
TO 10:E=0:GOSUB690:GOSUB720:XA=X
A+1:NEXT:XA=XA-1:FOR X=11 TO 20:
E=1:GOSUB690:GOSUB720:XA=XA-1:NE
XT
210 FG=6:LS=3:OK=-1:PO$=STR$(BP(
100,0)+2)+","+STR$(BP(100,1)+6):
SC=100:GOSUB60
220 HPAINT(256,191),14,3
230 PP=0:CP=0:PN=1:CN=1:GOSUB400
:GOSUB410
240 HCOLOR12:HLINE(254,10)-(312,
40),PSET,BF
250 OS$="":LS=1:SS$="S12":FG=7:H
COLOR2:HPRINT(32,2),"SNAKES":HCO
LOR5:HPRINT(32,3),"AND":HCOLOR11
:HPRINT(32,4),"LADDERS":GOSUB870
:M$(1)="Press":M$(2)="To":M$(3)=
"Play":GOSUB860
260 GOSUB910
270 GOSUB280:GOTO290
280 IK$=INKEY$:IFIK$=""THEN280EL
SE RETURN
290 FG=15:GOSUB870:M$(1)="Roll":
M$(2)="For":M$(3)="Start":GOSUB8
60:GOSUB420:PP=SC
300 FG=4:GOSUB870:M$(1)="My":M$(
2)="Roll":M$(3)="Now":GOSUB860:G
OSUB430:CP=SC
310 IF CP=PP THEN 290 ELSE IF CP
>PP THEN 330 ELSE 320
320 GOSUB350:IF PN=100 THEN GOSU
B870:M$(1)="YOU WIN":GOSUB860:FO
R OO=1 TO 1000:NEXT:GOSUB880:IF
II$="Y" THEN 900 ELSE GOTO930
330 GOSUB370:IF CN=100 THEN GOSU
B870:M$(1)="I WIN":GOSUB860:FOR
OO=1 TO 1000:NEXT:GOSUB880:IF II
$="Y" THEN 900 ELSE GOTO930
340 GOTO320
350 FG=15:GOSUB870:M$(1)="Your":
M$(2)="Turn":GOSUB860:GOSUB420:G
OSUB450:GOSUB460:GOSUB490:IF SC=
6 THEN 350
360 RETURN
370 FG=4:GOSUB870:M$(1)="My":M$(
2)="Turn":GOSUB860:GOSUB430:GOSU
B460:GOSUB450:GOSUB590:IF SC=6 T
HEN 370
380 RETURN
390 GOTO290
400 HGET(BP(PN,0),BP(PN,1))-(BP(
PN,0)+24,BP(PN,1)+18),1:RETURN
410 HGET(BP(CN,0),BP(CN,1))-(BP(
CN,0)+24,BP(CN,1)+18),2:RETURN
420 PO$="270,160":I$=INKEY$:GOSU
B730:IFI$=""THEN420ELSE GOTO440
430 PO$="270,160":WA=RND(5)+5:FO
R WB=1 TO WA:GOSUB730:NEXT
440 SOUND100,1:SOUND100,1:FOR OO
=1 TO 1000:NEXT:RETURN
450 HCIRCLE(BP(PN,0)+12,BP(PN,1)
+9),8,15:HPAINT(BP(PN,0)+12,BP(P
N,1)+9),15,15:RETURN
460 HCIRCLE(BP(CN,0)+12,BP(CN,1)
```

Continued on page 52

The next installment

CASHBOOK

OS-9
BUSINESS

by Ian Lobley

(The following program is in Basic09 source code and is quite long. Several Procedures will be published each month and will then be followed by Ian's articles explaining the operation of the program. As Ian uses the HIRES screen, those who wish to use Wordpak will have to alter the Modules accordingly.

This is the second part of Ian Lobley's CASH BOOK program written in Basic09. If you haven't already done so, may I suggest that you start with last months magazine as it will give you an introduction to these articles. The following is just a brief guide to the operation of Cash Book. Next month we will explain how the whole system works and what to do if you get into trouble.

Having typed in all the procedures and SAVED them under the correct NAMES we are ready to PACK them into the CMDS directory (don't forget to keep the Basic09 source code on another disk). As per all PACKED procedures, RUNB must be in the cmds direcory for them to work. The next step is to use the MAKDIR command and create a directory called CASH. This directory will hold the following data files for the program; Items_list
Company_name
Bank
Start_bal
Chq_record

When running the procedure RUN_CASH, a memory size of 9000 is required to Run, and store data files during operation. The 'CASH' directory will always be the working directory (chd /d0/cash)

If problems are encountered, go back and check your source code. Some adjustments may have to made in the layout of the screen. Ian uses OPAK in the sixtyfour character by nineteen line mode.

```
RUN CASH includes the
following procedures :-
menu  Read_Bank  Title
title2  items  Change_chq
get_chq search_chq
create_chq  print_t
print_chq  set_date
input_chq  plus

CREATE_CASH includes the
following procedures :-
create_menu  create_chq
create_bank  create_name
input_items create_items

*REM create_bank creates
bank and start_bal

PRINT_CASH
search_month_p  print_chq_p
title_p  macau_title  items

OTHERS
Search_month
delete_chq_record  del_all
see_bank  recon
recalc  change_start_bal
```

PLEASE NOTE
that lines indented by two (2) characters are in fact a continuation of the previous line.

```
PROCEDURE search_month
PRINT CHR$(12)
DIM line:STRING[64]
line="--------------------
  ------------------------
  -----------------"
PRINT TAB(15); "******
  P R I N T   M O N T H
  ******"
PRINT  \ PRINT
TYPE types_all=code
  :STRING[4]; name:STRING
  [20]
DIM types(30):types_all
TYPE record=date:STRING[8];
   number:INTEGER; payee
  :STRING[25]; ammt:REAL;
  item:STRING[4]
DIM chq_rec:record
DIM type_total(30):REAL
FOR x=1 TO 30
type_total(x)=0
NEXT x
RUN items(types)
DIM path:BYTE
DIM choice:STRING[1]
DIM search:STRING[2]
DIM count:INTEGER
DIM total:REAL
DIM total2:REAL
DIM deposits:REAL
total=0
total2=0
count=0
deposits=0
chq_rec.date=""
chq_rec.number=0
chq_rec.payee=""
chq_rec.ammt=0
chq_rec.item=""
INPUT "What Month to PRINT
  <eg 06> ? ",search
IF LEN(search)<2 THEN
search="0"+search
ENDIF
PRINT CHR$(12)
RUN title
OPEN #path,"chq_record"
WHILE NOT(EOF(#path)) DO
GET #path,chq_rec
IF MID$(chq_rec.date,4,2)
  =search THEN
IF LEFT$(chq_rec.item,3)
  ="dep" THEN
deposits=deposits+chq_
  rec.ammt
RUN print_chq(chq_rec)
ELSE
total=total+chq_rec.ammt
FOR x=1 TO 30
IF chq_rec.item=types(x)
  .code THEN
type_total(x)=type
  _total(x)+chq_rec.ammt
ENDIF
```

```
NEXT x
count=count+1
RUN print_chq(chq_rec)
ENDIF
ENDIF
ENDWHILE
PRINT
PRINT line
PRINT TAB(17); "Totals ";
PRINT TAB(37);
PRINT USING "r10.2^";
  deposits;
CLOSE #path
PRINT TAB(48);
PRINT USING "r10.2^"; total
PRINT
PRINT line
FOR x=1 TO 30
IF type_total(x)<>0 THEN
total2=total2+type_total(x)
PRINT TAB(17); types(x).
  name;
PRINT TAB(48);
PRINT USING "r10.2^"; type
  _total(x)
ENDIF
NEXT x
PRINT
PRINT TAB(48);
PRINT USING "r10.2^"; total2
PRINT
PRINT  \ PRINT
INPUT "<enter> to Return
  ",choice
END
```

****PLEASE NOTE****

that lines indented by two (2) characters are in fact a continuation of the previous line.#########

```
PROCEDURE del_chq_record
DIM choice:STRING[1]
PRINT CHR$(12)
PRINT TAB(20); "**
  W A R N I N G **"
PRINT  \ PRINT
PRINT "ABOUT TO DELETE ALL
  CHEQUE RECORDS"
PRINT  \ PRINT
INPUT "Are you SURE <Y> or
  <N> ??? ",choice$
IF choice$="y" OR choice$
  ="Y" THEN
PRINT "****** Deleting Chq
  Records ******"
DELETE "chq_record"
RUN create_chq
ENDIF
PROCEDURE del_all
DELETE "bank"
DELETE "chq_record"
DELETE "company_name"
DELETE "items_list"
DELETE "start_bal"
END
```

****PLEASE NOTE****

that lines indented by two (2) characters are in fact a continuation of the previous line

```
PROCEDURE see_bank
DIM choice:STRING[1]
DIM balance:REAL
PRINT CHR$(12)
DIM path:BYTE
balance=0
OPEN #path,"bank"
GET #path,balance
CLOSE #path
PRINT TAB(20);
PRINT "Balance = $";
PRINT USING "r10.2^";
  balance
PRINT  \ PRINT
INPUT "<enter> to Return
  ",choice
END
```

****PLEASE NOTE****

that lines indented by two (2) characters are in fact a continuation of the previous line

```
PROCEDURE recon
PRINT CHR$(12)
PRINT "**** BANK
  RECONCILLIATION ****"
PRINT
TYPE record=date:STRING[8];
  number:INTEGER; payee:
  STRING[25]; ammt:REAL;
  item:STRING[4]
DIM chq_rec:record
DIM path:BYTE
DIM choice:STRING[1]
DIM dep_tot:REAL
DIM chq_tot:REAL
DIM uchq:REAL
DIM utot:REAL
DIM udep:REAL
DIM udtot:REAL
DIM old_bal:REAL
DIM diff:REAL
DIM bank_bal:REAL
DIM calc_bal:REAL
calc_bal=0
bank_bal=0
diff=0
old_bal=0
udep=0
udtot=0
utot=0
uchq=0
chq_tot:=0
dep_tot:=0
chq_rec.date=""
chq_rec.number=0
chq_rec.payee=""
chq_rec.ammt=0
chq_rec.item=""
INPUT "Input Last Bank
  Balance as per Bank
  Statement ",bank_bal
PRINT
PRINT "Input Ammount of
  Unpresented Cheques or
  Total "
LOOP
INPUT " <zero> to end  ?
  ",uchq
EXITIF uchq=0 THEN
ENDEXIT
utot=utot+uchq
ENDLOOP
PRINT
PRINT "Input Ammount of
  Unpresented Deposits or
  Total "
LOOP
INPUT " <zero> to end ?
  ",udep
EXITIF udep=0 THEN
ENDEXIT
udtot=udtot+udep
ENDLOOP
PRINT
OPEN #path,"start_bal"
GET #path,old_bal
CLOSE #path
PRINT CHR$(12)
PRINT "Statement Bank
  Balance ";
PRINT USING "t48,r10.2^";
  bank_bal
PRINT
PRINT "Start Balance ";
PRINT USING "t38,r10.2^";
  old_bal
OPEN #path,"chq_record"
WHILE NOT(EOF(#path)) DO
GET #path,chq_rec
IF LEFT$(chq_rec.item,3)
  ="dep" THEN
dep_tot=dep_tot+chq_rec.ammt
ELSE chq_tot=chq_tot+chq_rec
  .ammt
ENDIF
ENDWHILE
CLOSE #path
PRINT
PRINT "Plus Deposits ";
PRINT USING "t38,r10.2^";
  dep_tot
PRINT
PRINT "Less Cheques &
```

```
  Expenses ";
PRINT USING "t38,r10.2^";
  chq_tot
PRINT
PRINT "Plus Outstanding
  Deposits";
PRINT USING "t38,r10.2^";
  udtot
PRINT
PRINT "Less Unpresented
  Cheques ";
PRINT USING "t38,r10.2^";
  utot
PRINT
calc_bal=old_bal+dep_tot
  +udtot-chq_tot-utot
PRINT "Calculated Balance ";
PRINT USING "t48,r10.2^";
  calc_bal
PRINT
diff=calc_bal-bank_bal
PRINT "Difference ";
PRINT USING "t48,r10.2^";
  diff
PRINT
INPUT "<enter> to Return
  ",choice$
END
```

****PLEASE NOTE****

that lines indented by two (2) characters are in fact a continuation of the previous line.#########

```
PROCEDURE recalc
PRINT CHR$(12)
PRINT "**** BANK
  RECALCULATION ****"
PRINT
TYPE record=date:STRING[8];
  number:INTEGER; payee:
  STRING[25]; ammt:REAL;
  item:STRING[4]
DIM chq_rec:record
DIM path:BYTE
DIM choice:STRING[1]
DIM dep_tot:REAL
DIM chq_tot:REAL
DIM old_bal:REAL
DIM bank_bal:REAL
DIM calc_bal:REAL
calc_bal=0
bank_bal=0
old_bal=0
chq_tot:=0
dep_tot:=0
chq_rec.date=""
chq_rec.number=0
chq_rec.payee=""
chq_rec.ammt=0
chq_rec.item=""
OPEN #path,"start_bal"
GET #path,old_bal
CLOSE #path
PRINT "Start Balance ";
PRINT USING "t38,r10.2^";
  old_bal
OPEN #path,"chq_record"
WHILE NOT(EOF(#path)) DO
GET #path,chq_rec
IF LEFT$(chq_rec.item,3)
  ="dep" THEN
dep_tot=dep_tot+chq_rec.ammt
ELSE chq_tot=chq_tot+chq_rec
  .ammt
ENDIF
ENDWHILE
CLOSE #path
PRINT
PRINT " Deposits ";
PRINT USING "t38,r10.2^";
  dep_tot
PRINT
PRINT " Cheques & Expenses
  ";
PRINT USING "t38,r10.2^";
  chq_tot
PRINT
calc_bal=old_bal+dep_tot
  -chq_tot
PRINT "Calculated Balance ";
PRINT USING "t48,r10.2^";
  calc_bal
PRINT
PRINT "To Make This Balance
  Your New Bank Balance
  press <y> "
INPUT "<enter> to Return
  ",choice
IF choice="Y" OR choice="y"
  THEN
OPEN #path,"bank"
PUT #path,calc_bal
CLOSE #path
ENDIF
END
```

****PLEASE NOTE****

that lines indented by two (2) characters are in fact a continuation of the previous line.#########

```
PROCEDURE change_start_bal
DIM path:BYTE
DIM begin_bal:REAL
begin_bal=0
OPEN #path,"start_bal"
GET #path,begin_bal
PRINT "Start Balance = ";
PRINT USING "t20,r10.2^";
begin_bal
PRINT
INPUT "Input New Balance
",begin_bal
SEEK #path,0
PUT #path,begin_bal
CLOSE #path
```

****PLEASE NOTE****

that lines indented by two (2) characters are in fact a continuation of the previous line

```
PROCEDURE menu
LOOP
PRINT CHR$(12)
PRINT TAB(15); "OS9 CASH
  BOOK for the CoCo"
PRINT TAB(14); "Version 1.01
  86/07/30 Ian Lobley"
PRINT TAB(13); "The Current
  Date Is ";
SHELL "date"
DIM choice:STRING[2]
DIM balance:REAL
FOR x=1 TO 2
PRINT
NEXT x
PRINT TAB(20); "<1> Input
  Cheques and Deposits"
PRINT TAB(20); "<2> See
  Cheques"
PRINT TAB(20); "<3> Search &
  Change"
PRINT TAB(20); "<4> Print
  Month  <44> to > printer"
PRINT TAB(20); "<5> Set New
  Date"
PRINT TAB(20); "<6> Input
  More Catagories"
PRINT TAB(20); "<9> Delete &
  Create Chq List"
PRINT TAB(20); "<y> Yearly
  Printout"
PRINT
PRINT TAB(20); "<n> For Next
  Menu"
PRINT TAB(20); "<e> To
  E*N*D"
INPUT choice
IF choice="e" OR choice="E"
  THEN
END
ENDIF
IF choice="n" OR choice="N"
  THEN
PRINT CHR$(12)
PRINT "****** M E N U  No  2
  ******"
PRINT  \ PRINT  \ PRINT
PRINT TAB(20); "<10>
  Reconcilliation of Bank
  statement"
PRINT TAB(20); "<11>
  Recalculate Bank Balance"
PRINT TAB(20); "<12> Correct
```

```
Codes & Catagories List "
PRINT TAB(20);"<13> Change
  Starting Bank Balance"
PRINT
PRINT TAB(20); "<enter> to
  Return"
PRINT
INPUT choice
IF choice="11" THEN
RUN recalc
ENDIF
IF choice="10" THEN
RUN recon
ENDIF
IF choice="12" THEN
RUN correct_items
ENDIF
IF choice="13" THEN
RUN change_start_bal
ENDIF
ENDIF
IF choice="y" OR choice="Y"
  THEN
RUN year_print
ENDIF
IF choice="5" THEN
RUN set_date
ENDIF
IF choice="6" THEN
RUN input_items
ENDIF
IF choice="$" THEN
RUN see_bank
ENDIF
IF choice="9" THEN
RUN del_chq_record
ENDIF
IF choice="44" THEN
RUN search_month_p
ENDIF
IF choice="3" THEN
RUN search_chq
ENDIF
IF choice="4" THEN
RUN search_month
ENDIF
IF choice="1" THEN
RUN input_chq
ENDIF
IF choice="2" THEN
RUN get_chq
ENDIF
ENDLOOP
END
PROCEDURE read_bank
PARAM balance:REAL
DIM path:BYTE
OPEN #path,"bank"
GET #path,balance
CLOSE #path
END
PROCEDURE title
DIM top_title:STRING[64]
PRINT "Date";
PRINT TAB(10); "Chq No";
PRINT TAB(17); "Payee /
  Depositor";
PRINT TAB(38); "Deposits";
PRINT TAB(49); "Payments";
PRINT TAB(60); "Item"
END
PROCEDURE title2
DIM top_title:STRING[64]
top_title:="Date      Chq No
   Payee          Ammount
Type Balance"
PRINT top_title
END
PROCEDURE items
TYPE types_all=code:
  STRING[4]; name:STRING[20]
DIM path:BYTE
PARAM types(30):types_all
DIM x:INTEGER
OPEN #path,"items_list"
FOR x=1 TO 30
GET #path,types(x)
NEXT x
CLOSE #path
PROCEDURE change_chq
TYPE record=date:STRING[8];
  number:INTEGER; payee:
  STRING[25]; ammt:REAL;
  item:STRING[4]
PARAM chq_rec:record
DIM choice:STRING[10]
DIM new_bal:REAL
DIM old_bal:REAL
DIM path:BYTE
old_bal=0
new_bal=0
PRINT "Input change to be
  made "
PRINT "Either Date , Number
  , Payee , Ammt or Type "
INPUT choice
IF choice="date" OR
  choice="Date" THEN
INPUT "New Date
  ?",chq_rec.date
ENDIF
IF choice="Number" OR
  choice="number" THEN
INPUT "New Number ?
  ",chq_rec.number
ENDIF
IF choice="Payee" OR
  choice="payee" THEN
INPUT "New Payee ?
  ",chq_rec.payee
ENDIF
IF choice="ammt" OR
  choice="Ammt" THEN
OPEN #path,"bank"
GET #path,old_bal
PRINT "Current Bank Balance
  ";
PRINT USING "t35,r10.2^";
  old_bal
IF chq_rec.item<>"deps" THEN
old_bal=old_bal+chq_rec.ammt
ELSE old_bal=old_bal
  -chq_rec.ammt
ENDIF
PRINT "Balance w/o this
  transaction";
PRINT USING "t35,r10.2^";
  old_bal
INPUT "New Ammount ?
  ",chq_rec.ammt
IF chq_rec.item<>"deps" THEN
new_bal=old_bal-chq_rec.ammt
ELSE new_bal=old_bal+chq_rec
  .ammt
ENDIF
PRINT "New Bank Balance ";
PRINT USING "t35,r10.2^";
  new_bal
SEEK #path,0
PUT #path,new_bal
CLOSE #path
ENDIF
IF choice="Type" OR
  choice="type" THEN
INPUT "New Type ?
  ",chq_rec.item
ENDIF
END
PROCEDURE get_chq
PRINT CHR$(12)
TYPE record=date:STRING[8];
  number:INTEGER; payee
  :STRING[25]; ammt:REAL;
  item:STRING[4]
DIM chq_rec:record
DIM path:BYTE
DIM choice:STRING[1]
DIM dep_tot:REAL
DIM chq_tot:REAL
chq_tot:=0
dep_tot:=0
chq_rec.date=""
chq_rec.number=0
chq_rec.payee=""
chq_rec.ammt=0
chq_rec.item=""
RUN title
OPEN #path,"chq_record"
WHILE NOT(EOF(#path)) DO
GET #path,chq_rec
IF LEFT$(chq_rec.item,3)
  ="dep" THEN
dep_tot=dep_tot+chq_rec.ammt
ELSE chq_tot=chq_tot+chq_rec
  .ammt
ENDIF
RUN print_chq(chq_rec)
ENDWHILE
PRINT  \ PRINT  \
CLOSE #path
PRINT "   TOTALS  ";
PRINT USING "t38,r10.2";
  dep_tot;
PRINT USING "t49,r10.2";
```

```
 chq_tot
INPUT "<enter> to Return
 ",choice$
END
PROCEDURE search_chq
PRINT CHR$(12)
TYPE record=date:STRING[8];
 number:INTEGER; payee:
 STRING[25]; ammt:REAL;
 item:STRING[4]
DIM chq_rec:record
DIM path:BYTE
DIM choice:STRING[1]
DIM search:INTEGER
DIM count:INTEGER
count=0
chq_rec.date=""
chq_rec.number=0
chq_rec.payee=""
chq_rec.ammt=0
chq_rec.item=""
INPUT "What Chq No to Search
 for ? ",search
PRINT CHR$(12)
RUN title
OPEN #path,"chq_record"
WHILE NOT(EOF(#path)) DO
GET #path,chq_rec
count=count+1
IF search=chq_rec.number
 THEN
RUN print_chq(chq_rec)
INPUT "Change <Y> or enter
 ",choice
IF choice="y" THEN
RUN change_chq(chq_rec)
count=count-1
SEEK #path,SIZE(chq_rec)
 *count
PUT #path,chq_rec
ENDIF
ENDIF
ENDWHILE
CLOSE #path
PROCEDURE create_chq
PRINT "****** Creating new
 Chq List ******"
TYPE record=date:STRING[8];
 number:INTEGER; payee:
 STRING[25]; ammt:REAL;
 item:STRING[4]
DIM chq_rec:record
DIM path:BYTE
chq_rec.date=""
chq_rec.number=0
chq_rec.payee=""
chq_rec.ammt=0
chq_rec.item=""
CREATE #path,"chq_record"
PUT #path,chq_rec
CLOSE #path
PROCEDURE print_t
TYPE types_all=code:
 STRING[4]; name:STRING[20]
DIM types(30):types_all
DIM choice:STRING[2]
DIM x:INTEGER
RUN items(types)
PRINT "Check your item TYPE
 - List as below"
PRINT
FOR x=1 TO 30
IF types(x).code<>" " THEN
PRINT types(x).code,
ENDIF
NEXT x
PRINT
PRINT "<Enter>to Continue
 <I> to insert new Item"
INPUT choice
IF choice="i" OR choice="I"
 THEN
RUN input_items
ENDIF
END
PROCEDURE print_chq
TYPE record=date:STRING[8];
 number:INTEGER; payee:
 STRING[25]; ammt:REAL;
 item:STRING[4]
PARAM chq_rec:record
PRINT chq_rec.date;
PRINT TAB(11); chq_rec
 .number;
PRINT TAB(17); chq_rec
 .payee;
IF LEFT$(chq_rec.item,3)
 ="dep" THEN
PRINT TAB(37);
ELSE PRINT TAB(48);
ENDIF
PRINT USING "r10.2^";
 chq_rec.ammt;
PRINT TAB(60); chq_rec.item
PROCEDURE set_date
PRINT "The Current Date is"
SHELL "date"
SHELL "setime"
PROCEDURE input_chq
PRINT CHR$(12)
PRINT "*** I N P U T
 C H E Q U E S  $
 D E P O S I T S ***"
PRINT
TYPE types_all=code:
 STRING[4]; name:STRING[20]
DIM types(30):types_all
TYPE record=date:STRING[8];
 number:INTEGER; payee:
 STRING[25]; ammt:REAL;
 item:STRING[4]
DIM chq_rec:record
DIM found:BOOLEAN
DIM errnum:INTEGER
DIM path:BYTE
DIM cdate:STRING[8]
DIM choice:STRING[2]
DIM temp:STRING[6]
DIM chqno:INTEGER
DIM count:INTEGER
DIM balance:REAL
RUN items(types)
balance=0
count=0
chqno=0
temp=""
cdate=""
chq_rec.date=""
chq_rec.number=0
chq_rec.payee=""
chq_rec.ammt=0
chq_rec.item=""
RUN read_bank(balance)
OPEN #path,"chq_record"
WHILE NOT(EOF(#path)) DO
GET #path,chq_rec
ENDWHILE
INPUT "What number month are
 you doing <eg 06>",choice
IF LEN(choice)<2 THEN
choice="0"+choice
ENDIF
PRINT
PRINT "Last Chq was ....."
RUN title
RUN print_chq(chq_rec)
PRINT  \ PRINT
PRINT "Current Balance is ";
PRINT USING "r12.2"; balance
PRINT  \ PRINT
INPUT "What is next Chq No ?
 ",chqno
PRINT CHR$(12)
cdate=LEFT$(DATE$,3)+choice
 +"/"
temp=MID$(cdate,4,2)+MID$
 (cdate,1,2)
cdate="/"+MID$(temp,1,2)+
 "/"+MID$(temp,3,2)
RUN title
LOOP
INPUT " Input DATE  or <e>
 to End Session ",chq
 _rec.date
IF chq_rec.date="E" OR
 chq_rec.date="e" THEN 100
IF LEN(chq_rec.date)=1 THEN
chq_rec.date="0"+chq_rec
 .date
ENDIF
PRINT CHR$(9);
PRINT "

 "
PRINT CHR$(9);
chq_rec.date=chq_rec.date
 +cdate
PRINT chq_rec.date;
PRINT "    ";
INPUT "<Enter,s,d>",choice
IF choice="s" OR choice="d"
 THEN
chqno=chqno-1
ENDIF
chq_rec.number=chqno
```

```
IF choice="d" THEN
chq_rec.number=0
PRINT CHR$(9);
PRINT TAB(50)
PRINT CHR$(9);
PRINT chq_rec.date;
PRINT TAB(12);
PRINT "DEP";
ELSE
PRINT CHR$(9);
PRINT TAB(50)
PRINT CHR$(9);
PRINT chq_rec.date;
PRINT TAB(12);
PRINT chq_rec.number;
ENDIF
PRINT "  ";
INPUT " ",chq_rec.payee
IF chq_rec.number=0 THEN
PRINT TAB(38);
PRINT CHR$(9);
ELSE PRINT TAB(48);
PRINT CHR$(9);
ENDIF
INPUT " ",chq_rec.ammt
IF chq_rec.number=0 THEN
PRINT TAB(37);
ELSE PRINT TAB(48);
ENDIF
PRINT CHR$(9);
PRINT USING "r10.2^";
  chq_rec.ammt;
PRINT TAB(60);
50 INPUT " ",chq_rec.item
found=FALSE
FOR x=1 TO 30
IF chq_rec.item=types(x)
  .code THEN
found=TRUE
ENDIF
NEXT x
IF found=FALSE THEN
RUN print_t
RUN items(types)
PRINT CHR$(12)
RUN title
RUN print_chq(chq_rec)
PRINT TAB(59);
PRINT CHR$(9);
GOTO 50
ENDIF
IF LEFT$(chq_rec.item,3)<>
  "dep" THEN balance
  =balance-chq_rec.ammt
ELSE balance=balance+chq_rec
  .ammt
ENDIF
PUT #path,chq_rec
chqno=chqno+1
ENDLOOP
CLOSE #path
100 CLOSE #path
OPEN #path,"bank"
PUT #path,balance
CLOSE #path
END
```

SNAKES and LADDERS

Continued from page 45

```
+9),8,4:HPAINT(BP(CN,0)+12,BP(CN
,1)+9),4,4:RETURN
470 HPUT(BP(PN,0),BP(PN,1))-(BP(
PN,0)+24,BP(PN,1)+18),1,PSET:RET
URN
480 HPUT(BP(CN,0),BP(CN,1))-(BP(
CN,0)+24,BP(CN,1)+18),2,PSET:RET
URN
490 NA=PN+SC:IF NA=CN THEN NA=NA
-1:IF NA=PN THEN RETURN
500 IF NA>100 THEN RETURN
510 GOSUB470:PN=PN+1:GOSUB400:GO
SUB450:SOUND PN,1:IF PN<NA THEN
510
520 FOR J=1 TO 4:IF PN=M(J,0) TH
EN GOSUB470:PN=M(J,1):GOSUB700:G
OTO570
530 NEXT
540 FOR J=5 TO 8:IF PN=M(J,0) TH
EN GOSUB470:PN=M(J,1):GOSUB710:G
OTO570
550 NEXT
560 RETURN
570 IF PN=CN THEN PN=PN-1
580 GOSUB400:GOSUB450:SOUND PN,1
:RETURN
590 NP=CN+SC:IF NP=PN THEN NP=NP
-1:IF NP=CN THEN RETURN
600 IF NP>100 THEN RETURN
610 GOSUB480:CN=CN+1:GOSUB410:GO
SUB460:SOUND CN,1:IF CN<NP THEN
610
620 FOR J=1 TO 4:IF CN=M(J,0) TH
EN GOSUB480:CN=M(J,1):GOSUB700:G
OTO670
630 NEXT
640 FOR J=5 TO 8:IF CN=M(J,0) TH
EN GOSUB480:CN=M(J,1):GOSUB710:G
OTO670
650 NEXT
660 RETURN
670 IF CN=PN THEN CN=CN-1
680 GOSUB410:GOSUB460:SOUND CN,1
:RETURN
690 FOR T=0 TO 80 STEP 20:BP(X+T
,0)=XA(XA):BP(X+T,1)=YA(YA-E):E=
E+2:NEXT T:RETURN
700 PLAY"N1N2N3N4N5N6N7N8N9N10N1
1N12":RETURN
710 PLAY"N12N11N10N9N8N7N6N5N4N3
N2N1":RETURN
720 SS$="S4":FOR T=0 TO 80 STEP
20:PO$=STR$(BP(X+T,0)+6)+","+STR
$(BP(X+T,1)+6):SC=X+T:GOSUB60:NE
XTT:RETURN
730 SC=RND(6):GOSUB810:ON SC GOS
UB750,760,770,780,790,800:GOSUB8
20
740 RETURN
750 C(5)=FG:RETURN
760 C(1)=FG:C(9)=FG:RETURN
770 C(3)=FG:C(5)=FG:C(7)=FG:RETU
RN
780 C(1)=FG:C(3)=FG:C(7)=FG:C(9)
=FG:RETURN
790 C(5)=FG:GOTO780
800 C(2)=FG:C(8)=FG:GOTO780
810 FOR X=1 TO 9:C(X)=14:NEXT:RE
TURN
820 DI$="C"+STR$(FG)+"U7R7D7L7BE
C"+STR$(C(1))+BX$+"BU2C"+STR$(C(
2))+BX$+"BU2C"+STR$(C(3))+BX$+"B
R2C"+STR$(C(4))+BX$+"BD2C"+STR$(
C(5))+BX$+"BD2C"+STR$(C(6))+BX$+
"BR2C"+STR$(C(7))+BX$+"BU2C"+STR
$(C(8))+BX$+"BU2C"+STR$(C(9))+BX
$
830 DD$="BM"+PO$+"S16"+DI$
840 HDRAW DD$
850 RETURN
860 HZ=1:FOR HY=10 TO 14 STEP 2:
HPRINT(32,HY),M$(HZ):HZ=HZ+1:NEX
T:RETURN
870 HCOLOR14:GOSUB860:FOR HD=1TO
 3:M$(HD)="          ":NEXT:HCOLOR
 FG:RETURN
880 GOSUB870:M$(1)="Play":M$(2)=
"again":M$(3)="Y/N":GOSUB860
890 II$=INKEY$:IFII$=""THEN890EL
SEIF II$="Y" OR II$="N" THEN RET
URN ELSE 890
900 PP=0:CP=0:GOSUB470:GOSUB480:
CN=1:PN=1:GOSUB400:GOSUB410:GOTO
290
910 FOR PR=0 TO 15:READ PC:PALET
TE PR,PC:NEXT:RETURN
920 DATA55,39,32,4,5,45,13,8,43,
32,20,13,61,60,63,36
930 PALETTE RGB:POKE65496,0:END
940 FOR X=0 TO 15:PALETTE X,0:NE
XT:PALETTE13,60:HCOLOR13
950 HPRINT(32,2),"SNAKES":HPRINT
(32,3),"AND":HPRINT(32,4),"LADDE
RS":HPRINT(32,19),"Just":HPRINT(
32,20),"A":HPRINT(32,21),"Moment
":HPRINT(32,22),"Please":RETURN
```

ANOTHER PIE

Continued from page 46

```
103 DATA141,42,237,131,156,64,38
,250,204,202,20,142,211,1,141,14
,141,26,237,131,156,66,38,250,20
4,169,239,142,147,1,167,140,147,
231,65,31,16,215,72,142,,1,32,17
8,51,140,119,223,64,236,94,51,20
3,223,66,48,203,159,68,95,79,221
,74,221,70,57
104 DATA141,232,236,195,237,131,
156,66,38,248,141,72,141,70,134,
7,140,134,8,198,17,221,72,141,20
9,134,3,151,71,159,68,236,131,39
,250,111,65,141,45,236,132,227,1
32,36,6,49,132,108,162,39,252
105 DATA8,73,36,242,12,73,237,12
9,156,68,37,232,166,65,139,48,18
9,163,10,10,71,38,217,134,32,189
,163,10,10,72,38,198,57,158,68,7
9,71,105,130,105,130,73,156,66,3
8,246,57
```

Keep this one handy

HELPFUL PEEKS and POKES

APPLICATIONS

16K ECB

by Greg Dennis

HELPFUL PEEKS AND POKES is a program I devised so that I could keep a record of Peeks and Pokes in one program.

It covers areas such as getting more memory, speed pokes, screen management, input / output problems, disability controls and character controls.

The Listing:

```
0 GOTO10
3 SAVE"201:3":END
10 '***************
20 '*  HELPFULL   *
30 '*   PEEKS     *
40 '*    'N'      *
50 '*   POKES     *
60 '*             *
70 '*    BY       *
80 '*  GREGORY    *
90 '*   DENNIS    *
100 '*   1986      *
110 '***************
120 ' DRAWS TITLE PAGE
130 PMODE4,1:SCREEN1,1:PCLS
140 DRAW"BM22,20D20R5U8R6D8R5U20
L5D8L6U8L5"
150 DRAW"BM47,40U20R16D5L11D3R10
D4L10D3R11D5L16"
160 DRAW"BM72,40U20R5D15R11D5L16
"
170 DRAW"BM97,40U20R16D13L12C0U5
C1R6U3L6D4C0D4C1D7L4"
180 DRAW"BM122,40U20R16D5L11D3R6
D5L6D7L5"
190 DRAW"BM147,40R16U20L5D15L6U1
5L5D20"
200 DRAW"BM172,40U20R5D15R11D5L1
6"
210 DRAW"BM197,40U20R5D15R11D5L1
6"
220 DRAW"BM60,80U20R16D13L12C0U5
C1R6U3L6D4C0D4C1D7L4"
230 DRAW"BM85,80U20R16D5L11D3R10
D4L10D3R11D5L16"
240 DRAW"BM110,80U20R16D5L11D3R1
0D4L10D3R11D5L16"
250 DRAW"BM135,80U20R5D9E8R5G10F
9L5H7D7L5"
260 DRAW"BM160,80U5R11U2L11U13R1
6D5L11D3R11D12L16"
270 DRAW"BM105,115U20R5F16U16R5D
20L10H10D10L5"
280 DRAW"BM97,97H5R5F5L5"
290 DRAW"BM134,97E5R5G5L5"
300 DRAW"BM60,145U20R16D13L12C0U
5C1R6U3L6D4C0D4C1D7L4"
310 DRAW"BM85,145U20R16D20L11U1C
0U4C1U10R6D10L6D1C0D4C1L5"
320 DRAW"BM110,145U20R5D9E8R5G10
F9L5H7D7L5"
330 DRAW"BM135,145U20R16D5L11D3R
10D4L10D3R11D5L16"
340 DRAW"BM160,145U5R11U2L11U13R
16D5L11D3R11D12L16"
350 ' DRAWS CONTINUE MESSAGE
360 DRAW"BM40,189U8R4D4NL4BR4"
370 DRAW"ND4R4BR4"
380 DRAW"D4R4BU2NL4U2NL4BR4"
390 DRAW"NR4D2R4D2NL4BR4"
400 DRAW"R4U2L4U2R4BR8"
410 DRAW"R4D2L4D2R4U4BR4"
420 DRAW"ND4R4D4BR4"
430 DRAW"R4U2L4U2BR4NDBR8"
440 DRAW"NU4D1ND3NE3NF3E1BR6"
450 DRAW"D4R4BU2NL4U2NL4BR4BD4"
460 DRAW"R4U2L4U2BR4ND2BR8"
470 DRAW"BU1R2NR2NU2D5BR4"
480 DRAW"U4R4D4NL4BR8"
490 DRAW"NR4U4R4BR4"
500 DRAW"ND4R4D4NL4BR4"
510 DRAW"U4R4D4BR4"
520 DRAW"BU5R2NR2NU2D5BR5"
530 DRAW"U4BU2U1BD7BR4"
540 DRAW"U4R4D4BR4"
550 DRAW"NU4R4NU4BR4"
560 DRAW"NR4U2NR4U2R4D2"
570 EXEC 44539
580 CLS
590 ' CREDIT LINES
600 PRINTTAB(6)"BY GREGORY DENNI
S"
610 PRINTTAB(12)"1986"
620 PRINT"PRESS ANY KEY TO CONTI
NUE"
630 EXEC 44539
640 '****MAIN MENU****
650 CLS
660 PRINT"PLEASE MAKE A CHOICE A
S TO WHICH SECTION YOU WOULD LIK
E TO SEE"
670 PRINT TAB(6)"(1) MEMORY HINT
S"
680 PRINT TAB(6)"(2) SPEED POKES
"
690 PRINT TAB(6)"(3) SCREEN MANA
GING
700 PRINT TAB(6)"(4) I/O
710 PRINT TAB(6)"(5) DISABLE FUN
CTIONS"
720 PRINT TAB(6)"(6) CHARACTER C
ONTROL"
730 PRINT TAB(6)"(7) TO EXIT TO
BASIC"
740 PRINT:PRINT"ENTER 1,2,3,4,5
OR 6"
750 PRINT:PRINTTAB(5)"WHICH ONE"
:INPUTA
760 ON A GOSUB 999,1799,2299,289
9,3599,4399,4999
770 GOTO 650
999 REM START OF MEMORY HINTS
1099 CLS:PRINT" POKE 25,6 :NEW (
31K FREE BUT NO HIGH RES)"
1199 PRINT" POKE 25,12:POKE26,1:
POKE3072,0:NEW (29K FREE BUT ONL
Y PMODE 0)"
1299 PRINT" POKE 25,18:POKE26,1:
POKE4608,0:NEW (27K FREE AND PMO
DE 2)"
1399 PRINT" POKE 25,24 :POKE 26,
1 :POKE 6144,10:NEW (26K FREE AN
D PMODE 2)"
1499 PRINT" POKE113,3:EXEC 40999
(CLEAR ALL MEMORY AND COLD START
 COMPUTER)"
1599 PRINT"PRESS ANY KEY TO RETU
RN TO MAIN MENU":EXEC 44539:RETU
RN
1699 REM END OF MEMORY HINTS
1799 REM START OF SPEED POKES
1899 CLS:PRINT" POKE 65495,0:DOU
BLE SPEED POKE(POKE65494,0 RETUR
NS YOU TO NORMAL SPEED)"
1999 PRINT" POKE 65497,0(TRIPLE
SPEED POKE N.B LOSS OF SCREEN BU
T COMPUTER STILL FUNCTIONS(POKE6
5496,0 TO RETURN TO NORMAL)"
2099 PRINT" POKE 359,60 :SLOW PO
KE (CAN NOT BE MADE BACK TO NORM
AL)":PRINT:PRINT"PRESS ANY KEY T
O RETURN TO MAIN MENU":EXEC 4453
9:RETURN
2199 REM END OF SPEED POKES
2299 REM START OF SCREEN MANAGIN
G
2399 CLS:PRINT" POKE 359,57(ORAN
GE SCREEN)"
2499 PRINT" POKE 178,N(N=1 TO 25
5) (TO GET MORE COLORS IN PMODE
4 E.G.DRAW A CIRCLE WITH CIRCLE(
100,100),50 THEN PAINT (100,100)
,,1";
2599 PRINT" THIS POKE MAKES THE
COMPUTER STORE A SPECIAL NUMBER
IN ITS PAINTS REGISTER SO BY ,,1
 THIS MAKES THE COMPUTER USE WHA
T EVER IS THERE)"
2699 PRINT:PRINT"PRESS ANY KEY T
O RETURN TO MAIN MENU":EXEC44539
:RETURN
2799 REM END OF SCREEN MANAGING
2899 REM START OF I/O
2999 CLS:PRINT" PRINTPEEK(129)(A
FTER AN I/O ERROR :1=THE TAPE:2=
OUT OF MEMORY)"
```

Continued on page 55

MACHINE LANGUAGE PROGRAMMING

LESSON TWO
32K CoCo with EDTASM+

by Malcolm Patrick

IN THIS LESSON you will learn how to write the source listing, check for any errors while in the edit command, and check for any errors while in the zbug command.

Instruction Mnemonics is how the computer gets its information. There are two parts to the Mnemonic, the Opcode and the Operand.

Comment Lines start with an asterisk or are placed in column five. Remember to always put in the comment lines because you might pick up the program next year and you will not remember what it is all about.

Labels are found in column two of the text (listing).

Look at figure 1-1 on page 2 and particularly at line 220; you will find the instruction

* BNE BUB020 *.

This means Branch if Not Equal to BUB020 in line 140 and continue on.

The Assembly–Language process

The 64K Colour Computer has 65,536 Bytes of memory available to the system. Each byte is made up of 8 Binary Ones or Zeros.

I.E. 00101011 = 1 BYTE or 8 BITS. If a BIT = (1) then it is on, if it = (0) then it is off.

The 6809 Microcomputer (computer) will read each Byte and convert it to an instruction (Binary code).

The EDTASM+ has a built-in assembler that will translate the text of the "assembly language" into "machine language", the binary ones and zeros.

A/IM

After the Assembly language (listing) has been typed in it is then (ASSEMBLED) by the edtasm+ command A/IM , Assemble / In Memory.

If you type in figure 1-2 on page 4 of Barden's book, and then do A/IM ,you will note the extra columns added to the source listing. This is referred to as the object code.

On the top of the first column of the object code is 0A6A. This will more than likely be different to your listing. It is the address in memory where the machine language is to be stored.

The second column of the object code begins with 7F. This will always be the same as your listing. It is the machine code for CLR found in the third column of the source code (opcode). So this is the OPCODE for the OBJECT CODE..

The third column of the object code begins with 0A95. This is sometimes different to your listing. This is the OPERAND for the OBJECT CODE.

COMMENT lines are ignored and do not generate any Object Code.

Below the listing is TOTAL ERRORS, this must always be Zero, otherwise the program will not run properly. If it is not zero then there is a mistake in the source listing so go through and correct any mistakes.

Last of all to scroll onto the screen will be the LABEL ADDRESSES. You will note that BUB010 from line number 120 in the source code, is assigned to the address of 0A6D in the Object Code.

Whatever address your Edtasm+ assembles BUB010 in line 120 to, it will always give you that address at the end of the program.

If there is any letter beside the OBJECT CODE LABEL ADDRESS, then there is a mistake with one of the labels either in the label column or the operand column.

A/LP A/IM/LP

To get a complete listing of the Object & Source code to the PRINTER, type in A/IM/LP. = Assemble In Memory to the Line Printer. If you are running EDTASM+ on disk and wish only to get the source listing, then you only need the letter (T).

A/IM/WE

When typing in large programs it is easy to make mistakes. One easy way to find them is to Assemble / In Memory / and WAIT on all ERRORS. I.E. A/IM/WE.

Load figure 1-2 from page 4 and change the (P) in line 110 to (D).

Now do an A/IM/WE. You will note that the assembling program will stop at line 110 and print the words UNDEFINED SYMBOL. It is telling you that there is an error in your program. You could stop the assembler there and correct the line, but at this stage we will continue by pressing enter.

The assembler will continue until the end and added to your Label Addresses is the word DASSNO followed by the letter (U).

This is also a way of finding if there is an error.

ZBUG

Zbug is another part of the EDTASM+ assembler package. It is another means of scrutinizing the source listing in the most intimate way. You will learn much more on this with each lesson.

EXECUTING THE PROGRAM IN ZBUG

To EXECUTE the the listing of figure 1-2, first make sure that the assembler runs through without any errors. Next go into ZBUG then make sure that the program will break out by typing (XLOOP), meaning EXIT at the word (command) LOOP.

Next start the execution (or run) by typing (GBUBSRT), meaning GO at the word (command) BUBSRT.

So what we have done is this.

```
*A/IM  rem assembled in memory
*Z     rem gone into zbug
           (note the change
           from asterisk to
           the pound sign)
#XLOOP rem exit at loop
#GBUBSRT  rem go bubsrt
```

You will see the letters being sorted at the top of the screen and the numbers at the bottom. To execute this program again

you will have to go back to the editing mode with the letter (E), assemble the source code again and continue the whole procedure again.

Now let's change the listing so that it will display better on the screen. Change the 400 in line 120 to 432, this is the beginning of the screen location. Now change the 511 in line 210 to 454, this will place the sorted numbers up a few lines.

To finish with run the assembled program in Zbug and note the difference.

EDTASM+ EDIT Commands

Q = Quit the program and return to basic
N = renumber, N200,20 = renumber from 200 and increase by 20
I = Insert, I200,2 = insert at line 200 and increase by 2

QUESTION FOR LESSON TWO

Q1 Why is a LABEL used ?

A1: .....

Q2 What is Mnemonics and what does it do ?

A2: .....

Q3 What is a BYTE and what does it represent ?

A3: .....

Q4 Why is each BYTE converted to either a one or a zero ?

A4: .....

Q5 What command is used to Assemble the source code ?

A5: .....

Q6 After you have assembled the source code what code is generated to the left of the line numbers and what are the three columns called ?

A6: .....

Q7 What is the easiest way to find an error in the source code?

A7: .....

Q8 If you changed the opcode in line 190 to CTA and then assembled the source code with A/IM/WE, what error code will appear?

A8: .....

Q9 What is the ZBUG and what does it do ?

A9: .....

Q10 What are the 4 commands to test by running a source code in zbug ?

A10: 1.....2.....

A10: 3.....4.....

Answers for Lesson Two (to be looked at AFTER you have completed the questionnaire!!)

Q1 Why is a LABEL used?
A1 It is easier to say goto label BUB020 than to find the address of the line and say go to address 0A74 and continue.

Q2 What is Mnemonics and what does it do?
A2 The mnemonics is how the computer gets its information by adding, subtracting, and comparing information found in the opcode and the operand.

Q3 What is a BYTE and what does it represent?
A4 A byte is 8 BITS of memory that is made up of ones or zeroes.

Q4 Why is each BYTE converted to either a one or a zero?
A4 The 6809 microcomputer can only read (understand) Binary code.

Q5 What command is used to Assemble the source code?
A5 A/IM

Q6 After you have assembled the source code what code is generated to the left of the line numbers and what are the three columns called?
A6 The OBJECT code is generated, and the three columns are 1 the address in memory 2 the opcode 3 the operand.

Q7 What is the easiest way to find an error in the source code?
A7 With the command A/IM/WE, Assemble in Memory Wait on Errors.

Q8 If you changed the opcode in line 190 to CTA and then assembled the source code with A/IM/WE, what error code will appear?
A8 Bad opcode

Q9 What is the ZBUG and what does it do?
A9 The zbug is part of the EDTASM+ program, and it allows you to examine the source code to see if there are any bugs in it.

Q10 What are the 4 commands to test by running a source code in zbug?
A10 1 A/IM/WE 2 Z
3 XLOOP 4 GBUBSRT

HELPFUL PEEKS and POKES

Continued from page 53

```
3099 PRINT" POKE143,8:POKE144,24
:POKE145,4 (READS A TAPE IN AT N
ORMAL SPEED)"
3199 PRINT" POKE 143,13:POKE144,
24:POKE145,6 (READS A TAPE IN AT
 HIGH SPEED)"
3299 PRINT" IF YOU GET CONTINUED
 I/O ERRORS WITH NO REASON FOR T
HEM HAPPENING TRY TURNING YOUR C
ASSETTE RECORDER UPSIDE DOWN WHI
LE THE PROGRAM IS LOADING"
3399 PRINT"PRESS ANY KEY TO RETU
RN TO MAIN MENU":EXEC44539:RETUR
N
3499 REM END OF I/O
3599 REM START OF DISABLE FUNCTI
ON SECTION
3699 CLS:PRINT" POKE383,57 (LIST
 PROTECT)"
3799 PRINT" POKE383,126(DISABLES
 LIST PROTECT)"
3899 PRINT" A=PEEK(116)*256+PEEK
(117)-20:X=INT(A/256):Y=A-(X*256
):POKE113,85:POKE114,X:POKE115,Y
:FORI=A TO A+17:READB:POKEI,B:NE
XT I :DATA 18,182,255,3,138,1,18
3,255,3,189,173,33,189,172,239,1
26,173,158 (DISABLES RESET)"
3999 PRINT" POKE65315,54 (DISABL
ES ROMPAK START UP)"
4099 PRINT" POKE65315,55 (STARTS
 ROMPAK)"
4199 PRINT"PRESS ANY KEY TO RETU
RN TO MAIN MENU":EXEC 44539:RETU
RN
4299 REM END OF DISABLE FUNCTION

4399 REM START OF CHARACTER CONT
ROL
4499 CLS:PRINT" EXEC44539 (PAUSE
S COMPUTER UNTIL KEY PRESS)"
4599 PRINT" POKE 383,126:POKE384
,161:POKE 385,177 (LISTS ONE LIN
E AT A TIME)"
4699 REM END OF CHARACTER CONTRO
L
4799 PRINT:PRINT"PRESS ANY KEY T
O RETURN TO MAIN MENU":EXEC 4453
9:RETURN
4899 REM RETURN TO BASIC
4999 POKE113,3:EXEC 40999
```

DATA STRUCTURES in FORTH

Languages

by John Redmond

IT HAS BECOME common for computing languages to require data to be declared at the beginning of a program, such as in Pascal or C. In truth, it's to give the compiler a chance to make sense of what the programmer has written, but it has become an important, and useful, part of the discipline of programming.

The absence of anything like this requirement in Basic programming certainly does make things simpler, but I'm convinced that it's an important contributor to the disorganized state of most Basic code. It IS possible to write clear, elegant Basic code, but it happens all too rarely.

One of the most conventional characteristics of Forth is the requirement to predefine constants and variables, e.g.,

```
4 CONSTANT LEGS
VARIABLE FROGS
```

Now, when you use LEGS in a program, or enter it from the keyboard, the constant value of 4 is placed on the stack. LEGS is a VALUE. When you use FROGS, however, things are very different.

FROGS places on the stack the ADDRESS of a reserved part of memory. You can then get the VALUE currently in that memory with the @ word (pronounced 'fetch').

When you do this, the address on the stack is replaced by the 16-bit value at the address.

This is a very natural way to do things, and we don't need to get involved with pointers and with talk of left and right values, as in C.

Such complications are red herrings and just get in the way of understanding what programming is about. Just remember that invoking a variable returns an address, while invoking a constant returns a value.

Beginners start to get confused, however, when a constant is used for an address. If you are familiar with assembly language, you will recognize that this is just the same as an equate - and done for the same reason, to make the code more readable and easier to maintain.

Programming would be very limited in Forth if the only data types were constants and variables. What about arrays, strings and the records of Pascal? Forth can go a lot further than more conventional languages in designing and defining specialist data types and the point of this article is to explore some simple examples. But first we must recap the structure of a Forth word, to see how memory is used.

A word in memory consists of a header and a body. To make the discussion clearer, I will discuss only the simplest type of header. In this, the first four bytes of the header contain the length of the name (first byte value), then the first three letters of the name. (If the name is shorter than three characters, the extra bytes contain garbage.) There are variations in header structure, but both the great polyForth and A*Forth version 3.69 do it in this way. The next two bytes are the link field and point to the header of the next word in the dictionary (this need not concern us any more in this article). The last two bytes contain the pointer to the code being used in the word (again, let's not worry too much here, but it's all in Brodie's 'Starting Forth').

All very complicated, but it's all constructed automatically by the very smart word CREATE. So, then, if we type in:

```
CREATE POND
```

... we get a complete header corresponding to POND. POND can now be found in the dictionary and, if we now type in POND, what happens? An address is placed on the stack. This address is that of the NEXT byte after the code pointer (see above). The jargon for this address is the pfa (parameter field address) and that of the code pointer is the cfa (code field address).

Getting complicated? A little, but the previous two paragraphs are worth re-reading. Then let's introduce another standard Forth word: ALLOT. ALLOT allots a designated number of bytes after the header. So, now are in a position (believe it or not!) to define the defining word VARIABLE. It is simply:

```
: VARIABLE CREATE 2 ALLOT;
```

In other words, VARIABLE uses CREATE to make a header with a cfa which points to the 2 bytes which have been allotted.

CONSTANT is very similar, except that the cfa points to different code, so that the CONTENTS of the two bytes, rather than their address, are returned on the stack.

Two more words, related to ALLOT, are , ('comma') and C, ('c-comma'). These introduce a 16-bit (comma) or 8-bit (c-comma) at the position after the header. In fact A*Forth always initializes variables to 0 when they are defined, and its definition is equivalent to:

```
: VARIABLE CREATE 0 , ;
```

Using CREATE and ALLOT, it's very simple to define an array. Suppose we need an array which holds 24 byte values - and we want to refer to it by name, e.g. ECOSYSTEM:

```
CREATE ECOSYSTEM 24 ALLOT
```

Now then, what happens when we type in ECOSYSTEM? We get the address of the first byte after the header, of course. It is the address of the first of the 24 ALLOTed bytes. It is the BASE ADDRESS of the array, i.e., the address of the 0th element. Want the address of the 9th element (i.e., with an offset of 8)? Just add 8 to the base address! You can fetch the byte value with C@ (c-fetch) and you

are there - or you can change its value with a C! (c-store). Not hard, is it?

In fact, defining an array in this way is trivially simple. Trivially simple, yes, but the data structure we have constructed is primitive. No checking for index range (what happens if we fall off the end?) or for element type (what happens if we forget that it contains bytes and try to poke it 16-bit values?).

This is part of the strength of Forth and part of the advantage in designing your own data structures: YOU can decide whether you need to have protective code.

If you do the range checking and data type generation somewhere else in a program, it is superfluous and time-wasting to repeat the check when you access the array.

If you really do need a smarter data structure, such as general ARRAY word to keep in your Forth library, it is still easy - once you have the hang of how Forth accesses data.

But let's not move too quickly but rather stick to the simpler examples for the moment - this time strings (the great strength of Basic). The simplest string type is the dimensioned string (just like the first four bytes of the header discussed above).

The length of the string is in the first byte, and the string itself follows. If we want a string variable, we need a string arranged in just this way directly after the header. We need, of course, to ALLOT some memory for it. But how much?

Well, the length of the string is in the first byte; so we need to ALLOT one more byte than the content of the first byte. Think about it!

But we still have to get the string into position, after the header. Forth has a special word to do exactly this - a word called WORD.

This is a brilliantly versatile word which is used over and over in all sorts of contexts. It expects on the stack a delimiter, which might be 32 (for a space) or 34 (for a "), and accepts a string of characters from the keyboard or disk screen up to, but not including, that delimiter.

WORD returns on the stack the address of the length byte of the string. Let's try to get these words working together with STRING:

```
: STRING CREATE 34 WORD C@ 1+
   ALLOT ;
```

Isn't the definition short?

Quite normal for Forth. Remember that STRING is a DEFINING word which is used to define other words, e.g.,

```
STRING PREACHER John Redmond"
```

Note that STRING extracted TWO strings from the command line. CREATE extracted PREACHER by using WORD with 32 as delimiter, then the act was repeated with 34 as delimiter. A matter of having the right tools for the job. Then the job's simple. Now, again, we have a header constructed with CREATE; so what happens when we type in PREACHER?

Yes, we get an address on the stack, as before. This is the address of the length byte of the string (the string proper starts at the next address). But what can we do with the address? Why COUNT it, of course!

The word COUNT takes the address and returns on the stack the byte value in the address and, underneath on the stack, the next address (i.e., that of the first character of the string). Again, the right tool for the job. We'll finish with another right tool: TYPE.

This word expects on the stack just the two entries put there by COUNT. Therefore, if we type in:

```
PREACHER COUNT TYPE
```

Forth responds with 'John Redmond' (who else?). But preachers don't usually count and type (do they?). The command line doesn't feel right, does it? Forth has unique power to make commands feel right (yes, I DO mean unique!). Let's define a very small word:

```
: IS COUNT TYPE ;
```

NOW we can type in PREACHER IS and get a perfectly reasonable answer. That's all for now.

What we've looked at this time is a selection of the simplest (and dumb) data types. There is just so much more that we can do with data in Forth and next time we will start to look at some of the more sophisticated structures and start to nibble at concepts of smart data types which know their own functions and know their place in the world (those interested in OOPLs will know what I'm talking about). Until then.

PROGRAM FIX

32K ECB
UTILITY

by D.W Thurbon

PROGFIX RELOCATES ML programs that normally reside in graphics screen memory pages 1-4 to a higher position in memory so that they may be used with disc systems. The program has user prompts.

The Listing:

```
0 GOTO10
3 SAVE"24B:3":END
10 ' PROGFIX. BY D. W. THURBON.
     COURTESY PIXEL SOFTWARE.
     PUBLIC DOMAIN SOFTWARE.
20 CLS:PCLEAR1
30 PRINT"PRESS ANY KEY WHEN TAPE
 READY"
40 IFINKEY$=""THEN40
50 INPUT"FILE NAME";F$
60 IFLEN(F$)>8THENPRINT"TOO BIG"
:GOTO50
70 CLOADMF$,&H2000
80 SL=PEEK(487)*256+PEEK(488)
90 EN=PEEK(126)*256+PEEK(127)-1
100 EX=PEEK(157)*255+PEEK(158)-&
H2000
110 NN=EN+42:XX=EN+1
120 FORI=EN+1 TO NN
130 READX$:X=VAL("&H"+X$)
140 POKEI,X
150 NEXTI
160 ST=SL+&H2000
170 S1=INT(SL/256):S2=SL-256*S1
180 N7=EN+27:POKEN7,S1:N8=EN+28:
POKEN8,S2
190 EL=EN-&H2000
200 S3=INT(ST/256):S4=ST-S3*256
210 POKEEN+23,S3:POKEEN+24,S4
220 E3=INT(EN/256):E4=EN-E3*256
230 POKEEN+34,E3:POKEEN+35,E4
240 X1=INT(EX/256):X2=EX-X1*256
250 POKEEN+41,X1:POKEEN+42,X2
260 INPUT"DISK FILE NAME";F$
270 IFLEN(F$)>8THENPRINT"TOO BIG
":GOTO260
280 SAVEMF$,ST,NN,XX
290 SOUND125,3:PRINT"READY":END
300 DATA 4F,B7,FF,40,86,34,B7,FF
310 DATA 03,8E,04,00,86,80,A7,80
320 DATA 8C,06,00,2D,F9,8E,26,00
330 DATA 10,8E,06,00,A6,80,A7,A0
340 DATA 8C,47,10,2D,F7,0F,71,7E
,18,38
```

The continuing saga of.....

FRICKERS FOLLIES

by Jack Fricker

This month I am going to continue with the subject of Public Domain Software. At the end of this article is a listing of the help files for the software I have been talking about. As you can see there is quite a number of them. The help files alone take up 1 disk

I am assured by INTERTAN (Tandy) That Level II will be released at the end of March which means that you should be able to purchase it by the time you read this. Hopefully I will have mine soon and will be able to report to you on it.

```
Directory of /d0/HELP
20:30:45
ACIA.MapIn          ACIA.MapOut
ANSI.gotoxy         ASCIIfy
Adj                 Adr
Advent              Amort
Antenna             Ap
Append              Asc
Asm
Average_StDev       Bessel
Bin2BCD             BinCom
Blank0              BootSplit
Break               Build
CCgotoxy            Cal
Cat                 ChVolNam
Char_to_Int_to_Char
Check               CheckBook
Check_File          Chown
Clock               Cmp
CoCo_Configurations
Col                 Com
Comm                Comm_bas
Commands            Compress
Crypt               D
DCopy               DDir
DDisplay            DList
DateLib             Date_Cvt
Dates               DeASCII
Del.a               Del.p
Dir                 DirLister
DisAsm              DiskCat
DiskID              DiskLock
Dl                  DnLoad
DocGen2             Dollar_Print
DumpMem             Eps
EquFix              Erase
ErrCmd              Erreport
Error               Error.a
Error.c             FM
Fast_Fourier        Field
Finance             FlexBin.a
FlexBin.b           FlexEx
Forms2.GNX          GetNumb
GfxDump2            Graph1
HCopy               HDel
HDir.a              HDir.b
Hangman             Help.a
Help.c              Hexify
Hx                  ISAM
Ident               Ileav
InKey               Iniz
Inkey_Hal           InputE
Insert              Install
Jerrybench          Kalah
Kermit              Kill13
Kimtron             LLoad
LineFit             Link
Lisa                List
ListN               Load
LoadMem             LowUp
MRename             MakArg
Make                MakeTopics
MapMem              Maze
Merge               ModBuild
ModLinkB            ModList
ModMem              Modem
Mortgage            Music
NInsert             Network
NewID               NewStrip
New_Hex_Dump        Normal
Nroff               Oki
OneLine             P1
POpen               Pad
PagPrt              Page
Peek                Pf
Plotter             Post
PrSet               PrSet_10x
Print.a             Print.b
Print.c             Purge.a
Purge.b             PutDos
Pwd                 Pwd_Name
QDir                RMLocate
RMnew               Randomize
Ratmaze             ReHook
Remote              Replace
ResRat1HC           RidSect
Rof                 S1Flex
S1Intel             S1Load
S1UnFlex            S1UnLoad
Save                SaveMem
Scan                Serial
SetMem              SetParam_CoCo
Seterm              Setime.new
Shape               ShoMem
ShoRegs             Sieve
Sine                Sled
SortDir             Spaces
Spint               Sqsh
StriParity          Strip.a
Strip.c             StripNum
StripREM            StripZ
SysCall             SysCall_Hal
SysTest             TVI970_Configurations
Term_Ctl            TexCom
Three_D_Graphics
Tmode               Today
Trans_Lib.A         Trans_Lib.C
Translit            Univariate
Unlink              UpLoad
UpLow               Upper
Uppercase           VerMod
Wc                  Wc1
WordGame            Words
Words.a             XCom9
XLisp               alias
arc                 basutil
cb
changepassword      changeterm
chkng               crypt.c
delw                dict
dirw                emd
extract             fcat
fcopy               findfunc
finds               go
graft               grep
head                intruder
latest              lib
listpasswords       listusers
makdir.a09          module
mv                  owner
patch               ppc
rc                  rm
setparam            showc
sort                spell
splif               split
sst                 tab
tail                tc
tcmp                term_utils
tr                  tree
tube                uniq
untab               unwords
words.c             xc

Directory of
/d0/HELP/CoCo_Configurations
20:32:51
CoCo_Char           RMS_WordPak
dc_WordPak          ds_WordPak

Directory of /d0/HELP/Sled
20:32:58
Intro               Sled
SledCard            SledMan

Directory of
/d0/HELP/TVI970_Configuratio
ns 20:33:08
DcTVI_970           DsTVI_970
```

A Pascal version of...

GAME of LIFE

The Game of life is a facinating game invented by the Mathematician, John Conway, which attempts to simulate a collection of living organisms. The game was originally designed to be played on a grid board of specified size on which counters are to be placed. These counters are born, survive or die during a generation according to the following rules:-

(a) Counters with 2 or 3 neighbouring counters survive to the next generation.

(b) Counters with 4 or more neighbours die from overcrowding and are removed from the next generation.

(c) Counters with 1 or less neighbours die from isolation and are also removed.

(d) Empty locations which have 3 counters in the neighbouring locations is a birth location. A counter is placed in the location for the next location.

The program is implemented in OS-9 Pascal. However, it can be translated into Basic09 easily if you understand both Pascal and Basic09 (well the two languages are compatable to a degree in that no further alteration of the alogrithm is required - unless you have a better one).

The program presented here used a grid board of 20 X 20, you can make it larger by changing the constant size. But remember the larger the board the slower the program. I do not recommend it.

If you have an IBM compatible machine and a Standard Pascal compiler then delete all the lines with the Shell commands. The program should run (even better) as the program is implemented in standard Pascal.

After you have typed in the program, compiled it using 'PASCAL < program name'. When you have succesfully compiled it without errors, you should PCODEF in your current data directory.

You mav execute the program at this stage, However I reccomend that you continue to compile this into native code (6809 Machine Language) for higher execution speed.

When you run the program, it will ask 'how many counters you wish to start with'. This is the number of stars in your initial pattern. Then it will ask you to enter the co-ordinate of each counter. Remember to draw up initial at the centre of the grid board to allow for expansion.

Note that you must NOT enter a comma between the X and Y co-ordinate. For example to enter a co-ordinate od x=10 and y=6 (10 accross, 6 down), type 10 6<enter>.

Please note that you must have the file 'SUPPORT' (one of the files included in the CMDS directorv of the Pascal Compiler) in the CMDS directory when running the program in either PCODE or in native code.

For the program to run properly you must have the following utilities in your execution directory:
TMODE , LOAD , DISPLAY

(For further details on this game see Scientific American, 1970 ,P 120) If you have any question please contact me on viatel 677282790.

Some initial patterns for you to try out :-

A straight line with either 3, 4, 5, or 6 counters. Or

```
***
* *
* *
```

for about 30 generations.

Please note: PCODE file must be executed by PASCALS, Native code can be executed by simpy typing its name.

```
Program game_of_life
(input,output) ;
   (* By David ( Dung ) Ly -
1986 *)
   (*  8 Awaba ave Wagga
Wagga NSW 2650 *)
   (* This is a public
domain software you may copy
      or make any
alteration if you wish *)

Const
     size = 20 ;
Type
     board =
array[0..size,0..size] of
integer ;
     temp_life =
array[0..size,0..size] of
boolean ;

Var

reply,system,generation,coun
t,count1 : integer ;
      again : char ;
      temp,grid:board ;
      die : temp_life ;

Procedure init ( var grid :
board ) ;

Var

number_of_counter,count,coun
t1,hh,vv:integer;

Begin

   Writeln('Please enter
inital pattern by Catesian
coordinates');
   Writeln('It is wise to
draw your pattern at the
middle of the board');
   Writeln('How many counter
do you wish to start with');

Readln(number_of_pattern);

   For count := 1 to
number_of_pattern do
     Begin
       Writeln('Enter [x,y]
coordinate of counter number
',count:2);
       readln(hh,vv);
       grid[vv,hh] := 1

     end
end;


 Procedure update ( var
temp:board ;
                    var die
: temp_life );
 Var
    count,count1 :integer ;

 Begin
     for count := 0 to size
do
```

```
    Begin
     for count1 := 0 to
size do
      Begin

       if die[count,count1]
= true then
        Begin
         temp[count,count1]
:= 0 ;
         die[count,count1]
:= false
        end
     end
      end;
end;


Procedure print (var grid :
board ) ;

Var
   home,count,count1 :
integer ;
Begin
   for count := 0 to size -
3   do
    Begin
      write('       ');
      for count1 := 0 to
size do
       Begin
         if
grid[count,count1] = 1 then
          write('*')
        else
         write(' ')
       end;
       Writeln;
     end;
     home := shell('display
1');
end ;


Procedure life ( var
grid,temp:board ;
                 var die :
temp_life ) ;
Var
   neighbours,count,count1
: integer ;

Begin

  neighbours := 0;
  for count := 1 to size -1
do
   for count1 := 1 to size -
1 do
    Begin
     neighbours :=
grid[count-1,count1-
1]+grid[count,count1-1]

+grid[count+1,count1-
1]+grid[count+1,count1]

+grid[count+1,count1+1]+grid
[count,count1+1]

+grid[count-
1,count1+1]+grid[count-
1,count1];

     if grid[count,count1] =
1 then
      Begin
        if (neighbours <=1 )
or ( neighbours >=4 ) then
         die[count,count1] :=
true
      end
     else
      if grid[count,count1]
= 0 then
       begin
        if neighbours = 3
then
         temp[count,count1]
:= 1
       end
    end
end;


        (* MAIN PROGRAM *)
Begin
  system := shell('tmode -
pause');
  REPEAT
    for count := 0 to size
do
    for count1 := 0 to size
do
     Begin
       grid[count,count1] :=
0;
       temp[count,count1] :=
0;
       die[count,count1] :=
false;
     end;
    system := shell('load
display');
    init ( grid );
    temp := grid;
    Writeln('How many
generation do you wish to
run');
    readln(reply);
    print(grid);
    for generation := 1 to
reply do
      Begin

       writeln('Generation
==> ':10,generation:2);
       life(grid,temp,die);
       update(temp,die);
       grid := temp ;
       system :=
shell('display c');
       print(grid);
  end;
  Writeln;
  Writeln('Would you like to
input another pattern');
  Readln(again)

UNTIL again = 'n'

end.
```

■

CORRECTIONS

"National Anthems" (Page 54, Australian CoCo, February 1987)

There is a printing error in line 50. Apparently some magazines have got this error while others don't.

But for those of you who do, here is the solution:

```
50 DATA169,191,182,207,201,228,2
17,239,233,255,249,207,207,207,2
07,207,207,207,207,207,207,249,2
55,233,239,217,228,201,207,182,1
91,169
```

***

"23" (Page 55, Australian CoCo, December 1986)

It seems that line 14 in the listing was accidentally typed into line 13. This is what it should look like:

```
13 LO$="BRNR4HU6ER4FD6GBR5":AP$=
"BU8NH2BR4BD8":N2$="NR5UE5UHL3GB
F6GBR4":N3$="BUFR4EU2HNL3EU2HL4B
F8":BB$="BR5":LY$="BR3U5NH3E3BR4
BD8"
14 LL$="NU8R5BR4":SL$="NG2E10BD1
0BR4"
```

***

"Disassembler" (Page 8, Australian CoCo, March 1987)

I forgot to acknowledge the authorship of the subroutines I used in 'the "Disassembler" program and consequently some of them weren't listed in the hardcopy.

Line 255 should read:

```
255  'HEX  -  DEC  ZH=HEX(ZN)
THESE SUBROUTINES ARE COURTESY
OF H. SCHNEIDER FROM FEBRUARY
1984 OF HOT COCO
```

The same applies for lines 271 to 282 inclusive.

Also, line 171 has got the last three characters missing. At present it reads ... 'GOT'. It should read ... 'GOTO 12'.

W. Tritscher

Hint...

LLIST @ 32 CPL on a DMP105

The below program will print out your program listings on a DMP 105 printer at 32 characters per line. Simply load and RUN the program before loading the program to be listed.

```
10 FOR X=32710 TO 32765
15 READ A:POKE X,A:NEXT X
20 EXEC 32710:CLEAR200,32709
25 POKE155,33
30 NEW
35 DATA 182,1,103,167,141,0,46
40 DATA 190,1,104,175,141,0,40
45 DATA 134,126,183,1,103,48
50 DATA 141,0,4,191,1,104,57
55 DATA 52,2,150,111,129,254
60 DATA 38,16,150,156,139,1
65 DATA 145,155,37,8,15,156
70 DATA 134,13,173,159,160,2
75 DATA 53,2,18,18,18
```

Draw your own conclusions
with CoCo 3 graphics.....

COCO 3 DRAWING MACHINE

CoCo 3 + Joystick
GRAPHICS UTILITY

REPRINT!!!

by Andrew McLintock

<Ed's note: This program appeared in last month's edition of Australian CoCo Magazine. It appears that the listing was subject to a change in the CoCo 3's DOS. We listed the program with a Rainbow Bits 1.4 DOS residing in the CoCo 3's memory. Sorry folks!>

A WHILE AGO dad got himself one of these flashy new colour computer three's. At first I didn't want too much to do with it probably because I didn't understand half the new commands.

After trying a few different ideas I soon found out that the graphics are very impressive.

That's what Drawing Machine is all about - a way of using the graphics in the easiest way possable. Nearly all the commands can be controlled by the joystick and this was the best way I found to make Drawing Machine easy to use.

There are a complete set of instructions with the program but if they look to be a bit long to put in and you don't want them then DEL 1150 - 1360.

You also need to change these lines:-

1140 RETURN

Get rid of the last command in the last HPRINT command in Line 1130. It is important that line 1370 stays the same line number.

Line 10 is needed if you have BDOS.

You can save your drawings on tape or disk but you will need a CoCo 3 and a joystick.

The Listing:

```
0 GOTO 10
1 '**DRAWING MACHINE**
2 '**BY ANDREW MCLINTOCK**
3 SAVE"148:3":END
10 POKE &H13E,0:POKE&H143,0 'FOR
 BDOS.
20 Y=1:HBUFF 4,3700:GOSUB 1110
30 PCLEAR 6:HSCREEN 2:HBUFF 1,66
:HBUFF 2,66:HBUFF 5,3700:C1=1:C2
=1:U=1
40 PALETTE 0,27:PALETTE 1,12:PAL
ETTE 3,20:PALETTE 4,0:PALETTE 5,
36:PALETTE 6,32:PALETTE 7,18:PAL
ETTE 8,55:PALETTE 9,63:PALETTE 1
0,44:PALETTE 11,40:PALETTE 12,45
:PALETTE 13,56:PALETTE 14,2:PALE
TTE 15,43
50 HCOLOR 2,0:HLINE(105,100)-(10
0,100),PSET:HLINE-(100,105),PSET
:HDRAW"BM104,102;F5G2H5G2E3C0E1C
2E3":HPAINT(101,101),2,2:HCOLOR
1,0
60 HGET(100,100)-(110,110),1:HSC
REEN 2
70 GOSUB 300
80 A=128:B=96:A1=A:B1=B
90 HGET(A,B)-(A+10,B+10),2
100 HPUT(A,B)-(A+10,B+10),1
110 A1=A:B1=B:A=JOYSTK(0)*5:B=JO
YSTK(1)*3
120 IF K=6 AND PD=1 THEN GOSUB93
0:HLINE-(A,B),PSET:GOSUB940
130 IF A>4 AND A<46 AND B<190 AN
D B>130 AND BUTTON(0)=1 THEN GOS
UB930:C=HPOINT(A,B):GOSUB940:IF
K=4 THEN 600 ELSE IF K=7 THEN GO
SUB 700
140 IF BUTTON(0)=1 AND A<52 THEN
 GOSUB 910:GOTO 210
150 IF BUTTON(0)=1 AND B<182 AND
 A>52 THEN GOSUB 920:ON K GOSUB
440,470,500,580,640,670,700,800,
820:GOTO 180
160 IF K=4 AND A>52 AND B>182 AN
D BUTTON(0)=1 THEN 580
170 IF K=7 AND A>241 AND B>181 A
ND BUTTON(0)=1 THEN 730
180 IF A=A1 AND B=B1 THEN 110
190 HPUT(A1,B1)-(A1+10,B1+10),2
200 T=0:GOTO 90
210 IF A>44 THEN 180
220 IF B<4 OR B>124 THEN 180
230 IF A>4 AND A<24 THEN 270
240 B2=INT((B-4)/20):B2=B2+1
250 ON B2 GOSUB 630,660,690,750,
790,810
260 GOTO 180
270 B2=INT((B-4)/20):B2=B2+1
280 ON B2 GOSUB 430,460,490,520,
560,570
290 GOTO 180
300 FOR X=4 TO 40 STEP 20:FOR Y=
4 TO 119 STEP 20
310 HLINE(X,Y)-(X+20,Y+21),PSET,
B

320 NEXTY:NEXTX
330 FOR X=4 TO 39 STEP 10:FOR Y=
130 TO 189 STEP 15
340 HLINE(X,Y)-(X+10,Y+15),PSET,
B

350 HPAINT(X+1,Y+1),C,1:C=C+1
360 NEXTY:NEXTX
370 C=1
380 HLINE(52,0)-(52,191),PSET:HP
AINT(2,2),1,1:HCIRCLE(14,14),7:H
LINE(28,8)-(40,20),PSET,B:HLINE(
20,28)-(8,40),PSET:A$="U4E9F3G7H
3F3G2L4"
390 HDRAW"BM28,41;"+A$:HPAINT(32
,35),1,1:A$="E5U4G5D4H6U4E5F6G5H
6":HDRAW"BM15,61;"+A$:A$="R8U8H2
U2L4D2G2R8L8D8":HDRAW"BM33,61;"+
A$:HLINE(27,51)-(35,51),PSET:HLI
NE-(28,48),PSET:HLINE(28,54)-(35
,51),PSET
400 HDRAW"BM8,81;R8E2U2H2L6H2U2E
2R8":HDRAW"BM28,81;U12D12R12":HD
RAW"BM 18,101;L8H2U8E2R8":HDRAW"
BM27,101;U10E1R3F1D5L5R5D5":HDRA
W"BM35,101;U11R4F1D3G1L4R4F1D4G1
L4"
410 HDRAW"BM9,121;U12R7F1D5G1L7"
:HDRAW"BM29,121;U10R4U3R2D3L3R7D
3L10R10D7L10":HPAINT(34,109),1,1
:HLINE(50,192)-(320,182),PSET,B:
HLINE(50,181)-(320,181),PSET
420 RETURN
430 K=1:L=0:HPRINT(18,23),"-CIRC
LE-":RETURN
440 IF L=1 THEN 450 ELSE GOSUB93
0:HSET(A,B,U):GOSUB940:L1=A:L2=B
:L=1:RETURN
450 R=SQR((A-L1)^2)+((B-L2)^2))
:GOSUB930:HCIRCLE(L1,L2),R,U:GOS
UB940:L=0:HRESET(L1,L2):K=0:GOSU
B910:RETURN
460 K=2:L=0:HPRINT(19,23),"-LINE
-":RETURN
470 IF L=1 THEN 480 ELSE GOSUB93
0:HSET(A,B,U):GOSUB940:L1=A:L2=B
:L=1:RETURN
480 GOSUB930:HLINE(A,B)-(L1,L2),
PSET:GOSUB940:L=0:RETURN
490 K=3:L=0:HPRINT(18,23),"-ERAS
ER-":RETURN
500 IF L=1 THEN 510 ELSE GOSUB93
0:HSET(A,B,U):GOSUB940:L1=A:L2=B
:L=1:RETURN
510 GOSUB930:HCOLOR 0,1:HLINE(A,
B)-(L1,L2),PSET,BF:GOSUB940:HCOL
OR U,0:RETURN

520 HPRINT(8,23),"SAVE PICTURE (
Y/N)":GOSUB1370:IF A$="Y"THEN 53
0 ELSE GOSUB910:RETURN
530 GOSUB910:HPRINT(8,23),"SAVE
TO (D)ISK OR (T)APE
540 GOSUB1370:IF A$="D" OR A$="T
" THEN D$=A$:GOTO 550 ELSE 540
550 GOSUB910:HPRINT(8,23),"FILEN
AME-":A2=17:GOSUB890:GOSUB930:GO
SUB950:GOSUB1000:GOSUB940:GOSUB9
```

```
10:RETURN
560 HPRINT(8,32),"-CLEAR- ARE YO
U SURE(Y/N)":GOSUB1370:IF A$="Y"
 THEN HCOLOR 0,5:HLINE(53,0)-(32
0,180),PSET,BF:HCOLOR 1,0:GOSUB9
10:RETURN ELSE RETURN
570 K=4:HPRINT(11,23),"COLOUR":H
PRINT(28,23),"PALETTE":HLINE(182
,192)-(182,182),PSET:RETURN
580 IF A<182 OR L=1 THEN 600 ELS
E GOSUB 910:HPRINT(10,32),"SLOT-
":A2=15:GOSUB 890:IF VAL(B$)<0 O
R VAL(B$)>15 THEN A=190:B$="":GO
TO 580
590 P1=VAL(B$):B$="":HPRINT(23,2
3),"COLOUR-":A2=30:GOSUB 890:IF
VAL(B$)<0 OR VAL(B$)>63 THEN B$=
"":GOTO 580 ELSE PALETTE P1,VAL(
B$):B$="":GOSUB 910:GOTO 180
600 GOSUB910:IF L=1 THEN 610 ELS
E HPRINT(18,23),"-COLOUR-":L=1:G
OTO 180
610 HPRINT(16,23),"CORRECT (Y/N)
":GOSUB1370:IF A$="N" GOSUB910:L
=0:GOTO 180
620 U=C:L=0:GOSUB 910:HCOLOR U,0
:GOTO 180
630 K=5:L=0:HPRINT(20,23),"-BOX-
":RETURN
640 IF L=1 THEN 650 ELSE GOSUB93
0:HSET(A,B,U):GOSUB940:L1=A:L2=B
:L=1:RETURN
650 GOSUB930:HLINE(A,B)-(L1,L2),
PSET,B:GOSUB940:L=0:K=0:GOSUB910
:RETURN
660 K=6:L=0:HPRINT(18,23),"-PENC
EIL-":RETURN
670 IF L=1 THEN 680 ELSE HLINE(A
,B)-(A,B),PSET:L=1:PD=1:RETURN
680 PD=0:K=0:L=0:GOSUB910:RETURN
690 K=7:L=0:HPRINT(8,23),"CHANGE
 PAINT COLOURS-  YES  NO":HLINE(
278,182)-(278,191),PSET:HLINE(24
1,182)-(241,191),PSET:RETURN
700 IF L=3 THEN 720 ELSE IF L=2
THEN 710 ELSE C1=C:L=2:GOSUB910:
HPRINT(14,23),"-BOUNDRY COLOUR-"
:RETURN
710 C2=C:L=3:GOSUB910:HPRINT(18,
23),"-PAINT-":RETURN
720 GOSUB930:HPAINT(A,B),C1,C2:G
OSUB940:L=0:K=0:GOSUB910:RETURN
730 IF A<278 THEN 740 ELSE GOSUB
910:HPRINT(18,23),"-PAINT-":L=3:
GOTO 180
740 GOSUB910:HPRINT(15,23),"-PAI
NT COLOUR-":L=1:GOTO 180
750 HPRINT(8,23),"LOAD PICTURE (
Y/N)":GOSUB1370:IF A$="Y" THEN 7
60 ELSE GOSUB910:RETURN
760 GOSUB910:HPRINT(8,23),"LOAD
FROM (D)ISK OR (T)APE"
770 GOSUB1370:IF A$="D" OR A$="T
" THEN D$=A$:GOTO 780 ELSE 770
780 GOSUB910:HPRINT(8,23),"FILEN
AME-":A2=17:GOSUB890:GOSUB950:GO
SUB1040:GOSUB910:RETURN
790 K=8:HPRINT(7,32),"*":A2=8:GO
SUB890:HPRINT(A2,23),"*":RETURN
800 GOSUB930:HPRINT(A/8,B/8),B$:
GOSUB940:B$="":K=0:GOSUB910:RETU
RN
810 K=9:HPRINT(17,23),"CUT & PAS
TE":L=0:RETURN
820 IF L>0 THEN 830 ELSE GOSUB93
0:HSET(A,B,U):GOSUB940:L=1:L1=A:
L2=B:RETURN
830 IF L=2 THEN 860 ELSE G1=ABS(
A-L1):G2=ABS(L2-B):G=INT(G1/2)+1
:IF G2*G>3700 THEN RETURN ELSE :
GOSUB930:HGET(A,B)-(L1,L2),4:HLI
NE(A,B)-(L1,L2),PSET,B:GOSUB 910
:HPRINT(16,23),"CORRECT (Y/N)"
840 GOSUB1370:IF A$="N" THEN HPU
T(A,B)-(L1,L2),4:GOSUB940:GOSUB
910:L=0:HRESET(L1,L2):RETURN
850 L=2:HPUT(A,B)-(L1,L2),4:HRES
ET(L1,L2):GOSUB940:GOSUB910:HPRI
NT(15,23),"-POSITION OBJECT-":RE
TURN
860 GOSUB930:HGET(A,B)-(A+G1,B+G
2),5:HPUT(A,B)-(A+G1,B+G2),4:GOS
UB 910:HPRINT(16,23),"CORRECT (Y
/N)"
870 GOSUB1370:IF A$="N" THEN HPU
T(A,B)-(A+G1,B+G2),5:GOSUB940:GO
SUB 910:L=0:RETURN
880 L=0:GOSUB940:GOSUB 910:RETUR
N
890 A$=INKEY$:IF A$="" THEN 890
ELSE IF A$=CHR$(13) THEN RETURN
900 B$=B$+A$:HPRINT(A2,32),A$:A2
=A2+1:IF A2>39 THEN RETURN ELSE
890
910 HCOLOR 0,1:HLINE(53,183)-(32
0,192),PSET,BF:HCOLOR U,0:RETURN
920 IF BUTTON(0)=1 THEN 920 ELSE
 RETURN
930 HPUT(A1,B1)-(A1+10,B1+10),2:
RETURN
940 HGET(A1,B1)-(A1+10,B1+10),2:
HPUT(A1,B1)-(A1+10,B1+10),1:RETU
RN
950 RESTORE:D=0:FOR X=&HE00 TO &
HE44:READ E$:E=VAL("&H"+E$):D=D+
E:POKE X,E:NEXTX:IF D<>&H183C TH
EN PRINT"ERROR IN DATA STATMENTS
":STOP
960 POKE &HE6E1,&H7E:POKE &HE6E2
,&H0E:POKE &H0E02,&H20:IF D$="T"
 THEN GOSUB 910:HPRINT(8,23),"SE
T UP TAPE RECORDER."
970 IF D$="D" THEN GOSUB910:HPRI
NT(8,23),"PRESS <ENTER>"
980 GOSUB 1370
990 RETURN
1000 POKE &H0E04,0:FOR X=1 TO 8:
N1$=B$+MID$(STR$(X),2):HCLS:IF D
$="D" THEN SAVEMN1$,&H1000,&H1FF
F,0:GOTO 1020
1010 CSAVE N1$,&H1000,&H1FFF,&H1
000
1020 NEXTX
1030 B$="":POKE &HE6E1,&H8E:POKE
 &HE6E2,&H20:RETURN
1040 POKE &H0E04,1:FOR X=1 TO 8:
N1$=B$+MID$(STR$(X),2):IF D$="D"
 THEN LOADM N1$ ELSE CLOADM N1$
1050 HCLS:NEXTX:GOTO 1030
1060 DATA 20,0D,20,0,0,10,0,20,0
,0,0,0,0,0,0
1070 DATA 30,8C,F0,EF,07,10,AF,0
9,ED,0B,1F,13
1080 DATA 10,AE,40,AE,43,6D,42,2
6,17,A6,A0,A7,80,AC,45,26,F8,10,
AF,40
1090 DATA 1F,31,EE,07,10,AE,09,E
C,0B,7E,E6,EB
1100 DATA A6,80,A7,A0,AC,45,26,F
8,20,E7
1110 WIDTH 40:PALETTE 11,36:PALE
TTE 12,63:HSCREEN 2:PALETTE 0,63
:PALETTE 1,0:P$="L20U3R20D3U18L2
0D15U15E5R10F5H5U3L2D3L2U3L2D3L2
U3L2D3U3R10":HDRAW"BM30,30"+P$:H
LINE(10,23)-(30,15),PSET,B:HGET(
10,3)-(30,30),4:FOR X=10 TO 320
STEP 40:HPUT(X,3)-(X+20,30),4
1120 HPUT(X,190)-(X+20,163),4:PA
LETTE 0,63:HCOLOR Y,0:HLINE(X+1,
22)-(X+19,16),PSET,BF:HLINE(X+1,
182)-(X+19,176),PSET,BF:Y=Y+1:NE
XTX:HPRINT(16,5),"COCO 3"
1130 HPRINT(11,9),"DRAWING  MACH
INE.":HPRINT(18,17),"By":HPRINT(
11,19),"Andrew Mclintock":HPRINT
(13,13),"Instructions"
1140 GOSUB 1370:IF A$="Y" THEN 1
160 ELSE RETURN
1150 ATTR 5,4
1160 WIDTH 80:Q$="-INSTRUCTIONS-
":LOCATE 33,0:PRINTQ$:LOCATE 2,2
:PRINT"Drawing Machine is a Hi-R
es graphic's programme and requi
res a single joystick.":ATTR 3,4
:LOCATE 0,4:PRINT"Display-":ATTR
 5,4:LOCATE 8,4
1170 PRINT"The screen is divided
 into four areas, these are:-The
 programme commands, Colour grid
s, Drawing page and the Operatio
n display. The commands are show
n in to rows of six squares each
 with its own diagram. The colou
r grid is below the"
1180 LOCATE 0,7:PRINT" commands
and are showm as sixteen differe
nt coloured squares. The drawing
 page is the large square in the
 middle of the screen. The opera
tions display is the rectangle a
t the bottom of the screen and i
t gives all messages and"
1190 LOCATE 67,9:PRINT"informati
on. The cursor is the small arro
w and is controled by the right
joystick. To select a command po
sition the arrow over the symbol
 and press the fire button.":ATT
R 7,4:LOCATE 0,12:PRINT"PROGRAMM
E COMMANDS":ATTR 3,4
1200 LOCATE 0,13:PRINT"Circle-":
ATTR 5,4:LOCATE 7,13:PRINT"Draws
 a cicle: Position arrow and pre
ss button, a small dot will show
 the centre of the circle. Move
the arrow and push button and a
circle will be drawn with a radi
us distance of the arrow and"
1210 LOCATE 42,15:PRINT"the cent
re. The small dot will disappear
.":ATTR 3,4:LOCATE 0,17:PRINT"Bo
x-":ATTR 5,4:LOCATE 5,17:PRINT"D
raws a box: Position arrow and p
ress button, a small dot will ma
rk one cornor. Move arrow to oth
er cornor and press button."
1220 ATTR 3,4:LOCATE 0,19:PRINT"
Line-":ATTR 5,4:LOCATE 5,19:PRIN
T"Draws a line: Position arrow a
nd press button and one end of t
he line will be marked. Position
 arrow at the other end and pres
s button.":ATTR 3,4:LOCATE 0,21:
PRINT"Pencil-":ATTR 5,4
1230 LOCATE 8,21:PRINT"Draws a c
ontinous line: Press button to l
```

Continued next page


```
POKE111,254:DIR
    Prints out your disk
directory to your printer
POKE25,14:POKE26,1:POKE3584,0:NEW
   No graphics, lots of memory.
POKE25,20:POKE26,1:POKE5120,0:NEW
    Same as PCLEAR1
POKE25,26:POKE26,1:POKE6656,0:NEW
    Same as PCLEAR2
POKE25,32:POKE26,1:POKE8192,0:NEW
    Same as PCLEAR3
POKE25,38:POKE26,1:POKE9728,0:NEW
    Same as PCLEAR4
POKE2439,255
     Verify on
POKE2439,0
     Verify off
POKE65344,0
     Turns off disk drive motors
POKE298,0:POKE303,0
     Turns off disk BASIC commands
POKE298,25:POKE303,14
     Returns the above to normal
POKE111,254:DIR
     Hard copy of directory onto
printer
PEEK(235)
     Returns drive number
PEEK(236)
     Returns track number
PEEK(237)
     Returns sector number
PEEK(2439)
     = 0 if verify on
     = 255 if verify off

PEEK(49152)
     68 = If disk system present
EXEC49364
     Warmstart DECB 1.0
EXEC49383
     Warmstart DECB 1.1
EXEC52175
     Same as DIR in DECB 1.0
EXEC52393
     Same as DIR in DECB 1.1
POKE359,60
     Slow poke for Extended
Colour BASIC
POKE359,60:POKE361,37
     Slow poke for Disk Extended
 Colour BASIC
POKE359,19:POKE360,19:POKE361,57
     Even slower poke
POKE65495,0
     Double speed
POKE65494, 0
     Normal speed
POKE65497,0
     Triple speed (loss of screen)
     Triple spped for CoCo 3
POKE65496,0 Returns to normal
```

COCO 3 DRAWING MACHINE

Continued from previous page

```
ower penceil. Draw with joystick
 and press button to lift it.":G
OSUB 1360:CLS 5:LOCATE 33,0:ATTR
 5,4:PRINTQ$:ATTR 3,4:LOCATE 0,1
:PRINT"Eraser-":ATTR 5,4:LOCATE
7,1
1240 PRINT"Clears a certain area
: Works the same as box command
but uses background colour.":ATT
R 3,4:LOCATE 0,3:PRINT"Paint-":A
TTR 5,4:LOCATE 7,3:PRINT"Paints
an enclosed area: When you selec
t this command you have the opti
on to change the paint"
1250 LOCATE 21,4:PRINT"and bound
ry colours or to use the previou
s colours set. To select new opt
ions place the arrow over the bo
x marked <Yes> and press button.
 Move the arrow to the colour gr
id and place it over the colour
you want to paint with and"
1260 LOCATE 9,7:PRINT"press butt
on. Do the same for the boundry
colour. Move the cursor to the D
rawing page and press button. Th
e screen will be coloured the pa
int colour until it hits the bou
ndry colour. If you wish to use
the same colours selected the"
1270 LOCATE 1,10:PRINT" box mark
ed <No> and position the arrow o
n the Drawing page and press but
ton.":ATTR 3,4:LOCATE 0,11:PRINT
"Save-":ATTR 5,4:LOCATE 6,11:PRI
NT"Allows you to save a picture
to tape or disk.":ATTR 3,4:LOCAT
E 0,12:PRINT"Load-"
1280 LOCATE 6,12:ATTR 5,4:PRINT"
Allows you to load a picture fro
m tape or disk.":ATTR 3,4:LOCATE
 0,13:PRINT"Clear-":ATTR 5,4:LOC
ATE 6,13:PRINT"Clears the drawin
g page.":ATTR 3,4:LOCATE 0,14:PR
INT"Letters-":ATTR 5,4:LOCATE 8,
14
1290 PRINT" Draws letters on the
 drawing page: Type in message f
rom keyboard and it will appear
on the operation display. Press
<Enter> when finished. Position
the arrow on the screen and pres
s button. The message will appea
r on the drawing page."
1300 ATTR 3,4:LOCATE 0,18:PRINT"
Palette-":ATTR 5,4:LOCATE 9,18:P
RINT"Controls drawing colours an
d Palette changes: To change dra
wing colour place arrow over box
 marked <Colour> and press butto
n. Then move the arrow to the co
lour grid and select a colour.
1310 LOCATE 35,20:PRINT"All mess
ages will apper in the drawing c
olour.":GOSUB 1360:CLS 5:LOCATE
33,0:ATTR 5,4:PRINTQ$:LOCATE 0,1
1320 ATTR 3,4:PRINT"Cut and Past
e-":LOCATE 14,1:ATTR 5,4:PRINT"A
llow you to get an area of the s
creen and put that image elsewhe
re: Position arrow and press but
ton, this marks the one cornor o
f the area. Move arrow amd press
 button and a box will"
1330 LOCATE 37,3:PRINT" mark the
 area to be moved. Reposition ar
row and press button and the ima
ge is put mack.":ATTR 3,4:LOCATE
 0,5:PRINT"Colour Grid:-"
1340 LOCATE 0,6:ATTR 5,4:PRINT"W
hen ever selecting a colour on t
he colour grid make sure that th
e point of the arrow is well on
the square.The first box holds t
he background colour, the second
 is the foreground colour (at th
e start of the"
1350 LOCATE 48,8:PRINT"programme
). The third holds the arrow col
our.":LOCATE 1,10:PRINT"The prog
ramme has several other bits a p
ices but these are not major and
 you will find them as you go al
ong. Have Fun !!!":GOSUB 1360:AT
TR 3,4:RETURN
1360 ATTR 3,4:LOCATE 25,23:PRINT
"PRESS <ENTER> TO CONTINUE.";:AT
TR 5,4
1370 A$=INKEY$:IF A$="" THEN 137
0 ELSE RETURN
```

■

GOLDSOFT

P.O. BOX 1742, SOUTHPORT. QLD. 4215 Phone (075) 510 015

Goldsoft Price list as at March, 1987 Please tick (✓) your requirements.

HARDWARE

CoCoConnection: $206.00 ( )
Video Amp: With Sound - $35.00 ( )
Without Sound - $25.00 ( )
The Probe: $49.95 ( )

GOLDLINK
Access Goldlink *642# on Viatel with a 1200/75 baud modem. Annual subscription: $39.75 ( )

SOFTWARE

Magazines, Tapes & Disks
Australian CoCo (Advanced Programs for your CoCo):

Magazines:	Tape ( ) or Disk ( )
12 Months $39.95 ( )	12 Months $123.75 ( )
6 Months $24.95 ( )	6 Months $ 74.25 ( )
1 Month $ 4.50 ( )	1 Month $ 16.50 ( )

Softgold (Programs for your CoCo):

Magazines	Tape ( ) or Disk ( )
12 Months $39.95 ( )	12 Months $123.75 ( )
6 Months $24.95 ( )	6 Months $ 74.25 ( )
1 Month $ 4.50 ( )	1 Month $ 16.50 ( )

Gold Disk - Available Quarterly:
1 Month: $16.00 ( )

MicoOz Tape (MC-10 Programs for your Mico):
12 Months $75.00 ( )
6 Months $42.00 ( )
1 Month $10.00 ( )

The CoCo3 Tape/Disk:
1 - Tape: $10.00 Disk: $16.00 ( )
2 - Tape: $10.00 Disk: $16.00 ( )

"Say the Wordz":
Two Curriculum based speller programs for your Tandy Speech/Sound Pack: $29.95
Req. 32K + Tandy Speech Pack ( )

Best of CoCoOz - $16.00
A selection of programs from Australian CoCo Magazine.

	Tape:	Disk:
# 1 - Education:	( )	( )
# 2 - Games 16K:	( )	( )
# 2 - Games 32K:	( )	( )
# 3 - Utilities:	( )	( )
# 4 - Business :	( )	( )
# 5 - Adventure:	( )	( )
# 6 - Preschool:	( )	( )
# 7 - Graphics :	( )	( )
# 8 - Games 16K:	( )	( )
# 9 - Games 32K:	( )	( )
#10 - Education:	( )	( )
#11 - Education:	-	( )

Special offer: Buy any 2 "Best of" and pay only $28.00!

BRIC-A-BRAC

Blank Tapes:	12 @ $18.00 ( )
(C-30)	1 @ $ 2.00 ( )
Tape Cases:	12 @ $ 5.00 ( )
Disks DSDD:	10 @ $20.00 ( )
	1 @ $ 2.50 ( )

BOOKS
Help (for your CoCo): $9.95 ( )
Mico Help (for your MC-10): $9.95 ( )

BACK ISSUES
Australian CoCo: Sep 84 - Dec 85: $2.00 ( )
Australian CoCo: Jan 86 - Feb 87: $3.75 ( )
Australian Mico: Aug 84 - Dec 85: $2.00 ( )

ADDITIONAL REQUIREMENTS

.....

.....

.....

.....

New Subscription: ( ) Renewal: ( )

Sub No: .....

Name: .....

Address: .....

.....

.....P/Code.....

Phone: (....) .....

Please find enclosed:
a. Cheque ( )
b. Money Order ( )
c. Credit Card ( )

Credit Card Type & Number:

Bankcard ( ): ..... - ..... - ..... - .....

Visa ( ): ..... - ..... - ..... - .....

Mastercard ( ): ..... - ..... - ..... - .....

Expiry Date: ...../.....

Authorised Amount: $.....

Signed: .....

(Stop between numbers = b.h. else a.h.; but, hyphen between = both)

user contacts

ACT:
- CANBERRA NTH — JOHN BURGER 062 58 3924
- CANBERRA STH — LES THURBON 062 88 9226

NSW:

SYDNEY:
- BANKSTOWN — CARL STERN 02 649 3793
- BLACKTOWN — KEITH GALLAGHER 02-627-4627
- CARLINGFORD — ROSKO MCKAY 02 624 3353
- CHATSWOOD — BILL O'DONNELL 02 419 6081
- COLYTON — HERMAN FREDRIKSSON 02 623 6379
- FAIRFIELD — ARTH PITTARD 02 72 2881
- GLADESVILLE — MARK ROTHWELL 02 817 4627
- HILLS DIST — ARTHUR SLADE 02 622 8940
- HORNSBY — ATHALIE SMART 02 848 8830
- INGLEBURN — STEPHEN RIDGEWAY 02 605 7382
- KENTHURST — TOM STUART 02 654 2178
- LEICHHARDT — STEVEN CHICOS 02 560 6207
  or GORGE ECHEGARAY 02 560 9664
- LIVERPOOL — LEONIE DUGGAN 02-607-3791
- MACQUARIE FIELDS — BARRY DARNTON 02 618 1909
- MINTO — GRAHAM POLLOCK 02 603 5028
- SUTHERLAND — IAN ANNABEL 02 528 3391
- SYDNEY EAST — JACKY COCKINOS 02 344 9111

- ALBURY — RON DUNCAN 060 43 1031
- ARMIDALE — DOUG BARBER 067 72 7647
- BLAXLAND — BRUCE SULLIVAN 047 39 3903
- BROKEN HILL — TERRY NOONAN 080 88 2382
- CAMDEN — KEVIN WINTERS 046.66.8068
- COFFS HARBOUR — BOB KENNY 066 51 2205
- COOMA — ROSS PRATT 0648 23 065
- COORANBONG — GEORGE SAVAGE 049 77 1054
- COOTAMUNDRA — CHERYL WILLIS 069 42 2264
- DENILIQUIN — WAYNE PATTERSON 058 81 3014
- DUBBO — GRAEME CLARKE 068 89 6549
- FORBES — JOHANNA VAGG 068 52 2943
- GOSFORD — PETER SEIFERT 043 32 7874
- GRAFTON — PETER LINDSAY 066 42 2503
- GUYRA — MICHAEL J. HARTMANN 067 79 7547
- JUNEE — PAUL MALONEY 069 24 1860
- KEMPSEY — RICK FULLER 065-62-7222
- LEETON — BRETT WALLACE 069-53-2081
- LISMORE — ROB HILLARD 066 24 3089
- LITHGOW — DAVID BERGER 063 52 2282
- MAITLAND — BILL SNOW 049 66 2557
- MOREE — ALF BATE 067 52 2465
- MUDGEE — BRIAN STONE 063-72-1958
- NARROMINE — GRAEME CLARKE 068 89 6549
- NEWCASTLE — LYN DAWSON 049 49 8144
- NOWRA — ROY LOPEZ 044 48 7031
- ORANGE — JIM JAMES 063 62 8625
- PARKES — DAVID SMALL 068 62 2682
- PORT MACQUARIE — RON LALOR 065 83 8223
- SPRINGWOOD — DAVID SEAMONS 047 51 2107
- TAHMOOR — GARY SYLVESTER 046 81 9318
- UPPER HUNTER — TERRY GRAVOLIN 065 45 1698
- URALLA — FRANK MUDFORD 067 78 4391
- WAGGA WAGGA — CES JENKINSON 069 25 2263
- WYONG — JOHN WALLACE 043 90 0312

NT:
- DARWIN — BRENTON PRIOR 089.81.7766

QLD:

BRISBANE:
- BIRKDALE — COLIN NORTH 07 824 2128
- CLAYFIELD — JACK FRICKER 07 262 8869
- COLL'WOOD PK — AND'W SIMPSON 07 288 5206
- IPSWICH — NICK MURPHY 07 271 1777
- PINE RIVERS — BARRY CLARKE 07 204 2806
- SOUTH WEST — BOB DEVRIES 07 372 7816
- SANDGATE — MARK MIGHELL 07 269 3846
- SCARBOROUGH — PETER MAY 07 203 6723
- WOODRIDGE — BOB DEVRIES 07 375 3161

- AIRLIE BEACH — GLEN EVANS 079 46 1264
- BIGGENDEN — ALAN MENHAM 071 27 1272
- BOWEN — TERRY COTTON C/O 077 86 2220
- BUNDABERG — RON SIMPKIN 071 71 5301
- CAIRNS — JEFF LARSEN 070 54 7127
- DALBY — MERRICK IANSKY 074.62.3228
- GLADSTONE — CAROL CATHCART 079 78 3594
- GOLD COAST — GRAHAM MORPHETT 075 51 0577
- GYMPIE — BERT LLOYD 071 8219100
- HERVEY BAY — LESLEY HORWOOD 071 22 4989
- MACKAY — LEN MALONEY 079511333x782
- MARYBOROUGH — JOHN EFFER 071 23 1369
- MT ISA — JACK RAE 077 43 3486
- MURGON — PETER ANGEL 071 68 1628
- ROCKHAMPTON — KEIRAN SIMPSON 079 28 6162
- TARA — DEBBIE DORFIELD 074 65 3177
- TOOWOOMBA — LEN GERSEKOWSKI 076 35 8264
- TOWNSVILLE — JOHN O'CALLAGHAN 077 73 2064

SA:
- ADELAIDE — JOHN HAINES 08 278 3560
- NORTH — STEVEN EISENBERG 08 250 6214
- MORPHETTVALE — KEN RICHARDS 08 384 4503
- PORT NOARLUNGA — ROB DALZELL 08 386 1647
- SEACOMBE HTS — GLENN DAVIS 08 296 7477
- PORT LINCOLN — BILL BOARDMAN 086 82 2385
- PORT PIRIE — VIC KNAUERHASE 086 32 1230
- WHYALLA — MALCOLM PATRICK 086 45 7637

TAS:
- DEVONPORT — JEFF BEST 004 24 6759
- HOBART — BOB DELBOURGO 002 25 3896
- KINGSTON — WIM DE PUIT 002 29 4950
- LAUNCESTON — BILL BOWER 003 44 1584
- WYNYARD — ANDREW WYLLIE 004 35 1839

VIC:

MELBOURNE:
- MELBOURNE CCC — JOY WALLACE 03 277 5182
- DANDENONG — DAVID HORROCKS 03 793 5157
- DONCASTER — JUSTIN LIPTON 03 857 5149
- FRANKSTON — BOB HAYTER 03.783.9748
- WARRE WARREN — LEIGH EAMES 03 704 6680
- NTH EASTERN — PETER WOOD 03 435 2018
- MELTON — MARIO GERADA 03 743 1323
- RINGWOOD — IVOR DAVIES 03 758 4496
- SUNBURY — JACK SMIT 03.744.1355
- SUNSHINE — IAN BUTTRIS 03 314 3240
- UPR F'TREE GLY — RORY DOYLE 03 758 2671

- BAIRNSDALE — COLIN LEHMANN 051 57 1545
- BALLARAT — MARK BEVELANDER 053 32 6733
- CHURCHILL — GEOFF SPOWART 051 22 1389
- DAYLESFORD — DANNY HEDJI 054 24 8329
- GEELONG — DAVID COLLEN 052 43 2128
- MAFFRA — MAX HUCKERBY 051 45 4315
- MOE — JOSEPH HESTER 051 27 7817
- MOE — JIMMY WELSH 051 27 6984
- MORNINGTON — MICHAEL MONCK 03 789 7997
- MORWELL — GEORGE FRANCIS 051 34 5175
- SALE — BRYAN McHUGH 051 44 4792
- SHEPPARTON — ROSS FARRAR 058 25 1007
- SMYTHESDALE — TONY PATTERSON 053 42 8815
- SWAN HILL — BARRIE GERRAND 050.32.2838
- TONGALA — TONY HILLIS 058 59 2251
- TRARALGON — LEIGH DAVIES 051 74 5552
- WONTHAGGI — LOIS O'MEARA 056 72 1593
- YARRAWONGA — KEN SPONG 057 44 1488

WA:
- PERTH — IAIN MACLEOD 09 448 2136
- GIRRAWHEEN — HANK WILLEMSEN 09 342 7639
- KALGOORLIE — TERRY BURNETT 090.21.5212

CANADA - CoCo:
- Ontario — Richard Hobson 416 293 2346

special interest groups

TEACHERS' INTEREST GROUP
- BRISBANE — BOB HORNE 07 281 8151

BUSINESS:
- BRIZBIZ — BRIAN BERE-STREETER 07 349 4696

OS9 GROUPS:
- NATIONAL OS9 USERS' GROUP — GRAEME NICHOLS 02 451 2954

NSW
- SYDNEY
  - BANKSTOWN — CARL STERN 02 646 3619
  - CARLINGFORD — ROSKO MCKAY 02 624 3353
  - GLADESVILLE — MARK ROTHWELL 02 817 4627
  - SYDNEY EAST — JACKY COCKINOS 02.344.9111
- COOMA — FRED BISSELING 0648 23263

QLD
- BRISBANE — JACK FRICKER 07 262 8869

VIC
- LATROBE VLY — GEORGE FRANCIS 051 34 5175

WA
- KALGOORLIE — TERRY BURNETT 090.21.5212

MC-10 GROUPS:
- LISMORE — ROB HILLARD 066 24 3089
- SYDNEY — GRAHAM POLLOCK 02 603 5028
- WARRNAMBOOL — GARY FURR 055 62 7440

TANDY 1000 / MS DOS:
- NSW:
  - GLADESVILLE — MARK ROTHWELL 02 817 4627
  - SYDNEY WEST — ROGER RUTHEN 047.39.3903
  - WYONG — JOHN WALLACE 043 90 0312
- QLD:
  - BRISBANE
    - NORTH — BRIAN DOUGAN 07 30 2072
    - SOUTH — BARRY CAWLEY 07 390 7946
  - GOLD COAST — GRAHAM MORPHETT 075 51 0015
- SA
  - PORT LINCOLN — BILL BOARDMAN 086 82 2385
- VIC:
  - MELBOURNE — TONY LLOYD 03 882 4664

FORTH:
- PORT LINCOLN — JOHN BOARDMAN 086 82 2385
- SYDNEY — JOHN REDMOND 02 85 3751

ROBOTICS:
- BOWEN — TONY EVANS 077 86 2220
- GOLD COAST — GRAHAM MORPHETT 075 51 0015
- SYDNEY — GEOFF FIALA 02 84 3172
- WAGGA WAGGA — CES JENKINSON 069 25 2263

CHRISTIAN USERS' GROUP:
- COLLIE — RAYMOND L. ISAAC 097 34 1578

24 HOUR 300 BAUD BULLETIN BOARD SYSTEMS

SYDNEY:
- INFOCENTRE — 02 344 9511
- TANDY ACCESS — 02 625 8071
- THE COCOCONNECTION — 02 618 3591

HOBART:
- HUB — 002 49 4.05

VIDEOTEX SYSTEMS
- MOUSETEX — 059 42 5528
- VTX 4000 — 03 741 3295

TANDY INFO ON VIATEL
- GOLDLINK — VIATEL #642#
- BLAXLAND COMPUTER SERVICES — VIATEL #64263#
- COMPUTER HUT SOFTWARE — VIATEL #64262#
- PARIS RADIO — VIATEL #64268#
- POWER CODE — VIATEL #64265#
- TANDY — VIATEL #64261#

ALLAN BEALE	726353300
FRED BISSELING	648232630
JACK FRICKER	726288690
JOHN GRIGSBY	945872030
STUART HALL	939765790
BOB KENNY	665122050
JEFF LARSEN	705471270
IAIN MACLEOD	944821360
CHRIS NAGLE	689523360
RICHARD PANKHURST	280717870
ROSS PRATT	648230650
RICCAY SCHMAHL	298151500
ARTHUR SLADE	262289400
DARCY O'TOOLE	755105770
RON WRIGHT	352924510

■

SOFTGOLD MAGAZINE
Registered by Australia Post –
Publication No. QBG 4009
AUSTRALIAN CoCo
Publication No. QBG 4007
P.O. BOX 1742
SOUTHPORT. QLD. Australia. 4215.

GOLDLINK

COM. STATION 692

ON

Telecom VIATEL
AUSTRALIA'S NATIONAL VIDEOTEX SERVICE

BULLETIN BOARDS
TANDY COMPUTERS
ATARI COMPUTERS
COMMODORE COMPUTERS
FAST FLORIST
REVIEWS
HARDWARE & SOFTWARE
SHOP

YES WE EVEN TALK BACK !!