Registered by Australia Post: Reg. No. QBG 4009

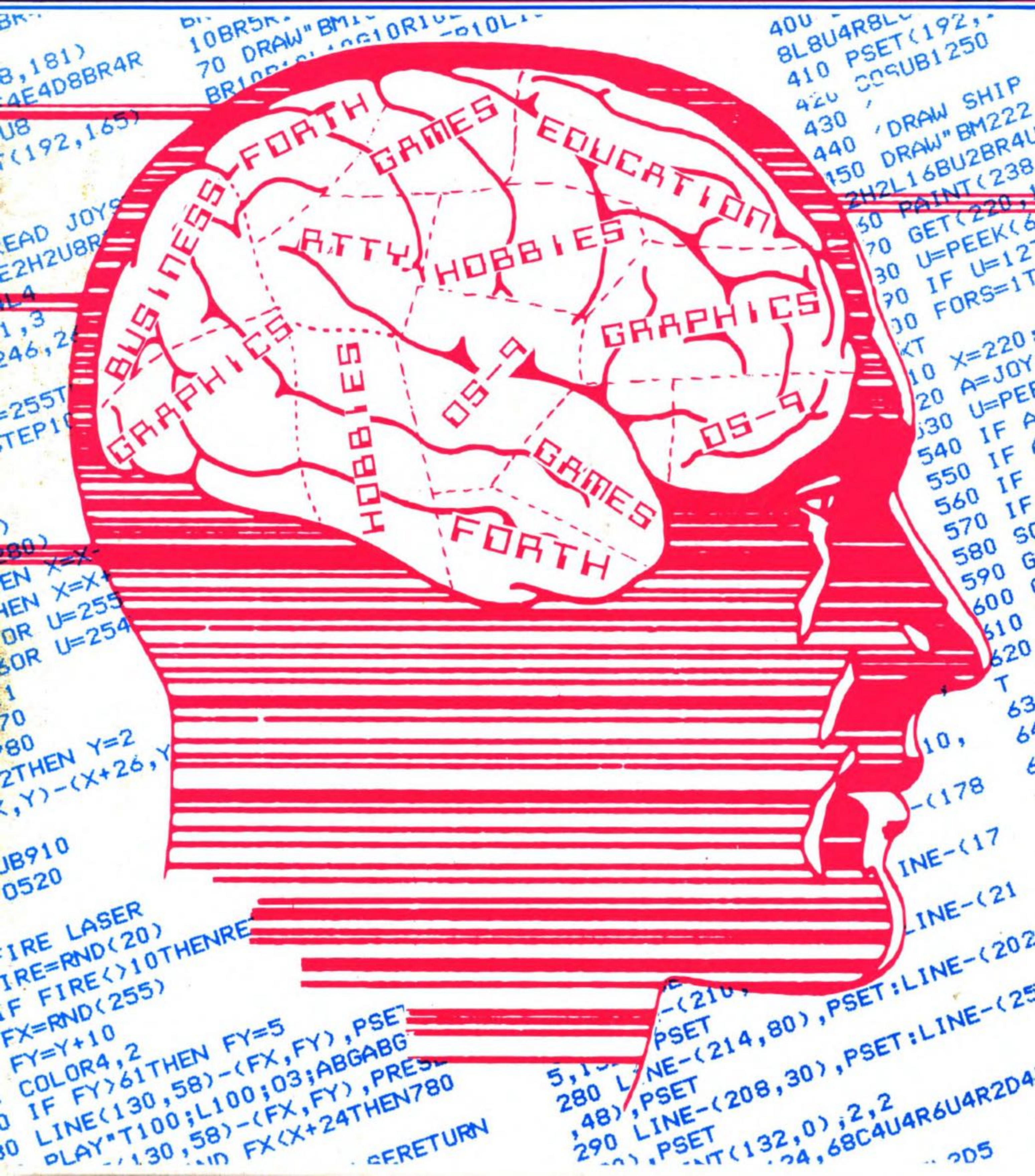
For your TANDY Color Computer

$3.95

AUSTRALIAN RAINBOW

March, 1985

No.45



SOFTWARE SPECTRUM

mail order: **P.O. Box 2101, Adelaide, South Australia, 5001.**

deliveries: **21 Compton Street, Adelaide**
Phone: (08) 211 8344
Orders Only (08) 211 8763

DISC DRIVES

DRIVE & PACKAGE:
40 Track Double sided Double Density slimline drive, power supply and case, HDS Disc Controller 1-1 Tandy DOS. + cable .... **$499 + p&h**

DRIVE 1 PACKAGE:
40 Track Double sided Double Density slimline drive power supply and case + cable .... **$310 + p&h**

DISC CONTROLLERS:
HDS Disc Controllers with Tandy 1-1 DOS. Gold Edge Connector .... **$199 + p&h**

J DOS:
An enhanced DOS. Implements all basic commands found in RS DOS plus new commands and features such as Auto for automatic line numbering: DOS to Boot 0S9: ERL, ERR and ERROR. To trap and determine error conditions, Flex to boot Flex: Ram to transfer J DOS to upper 32K ram and execute from it: Rate to change drive step rate: RUNM to run machine code program: DSKINI allows formatting 40 track single or double sided as well as the RS 35 track: up arrow, down arrow keys are used to scroll through text making debugging and program entry easier .... **$69.95 + p&h**

HJL KEYBOARD:
The best replacement keyboard for the CoCo .... **$139 + p&h**

PRINTER INTERFACE:
300-9600 switch selectable board rate, serial out put and printer switch .... **$99.95 + p&h**

VIDEO TO MONITOR INTERFACE:
Beat the RF Interference fit a Monitor .... **$35.00 + p&h**

TAXAN GREEN SCREEN MONITORS: .... **$189 + p&h**

Memory Upgrades
64K Ram Upgrades CoCo E, F, and CoCo II .... **$85.00 + p&h**
64K Ram Upgrade Board (new CoCo II) .... **$85.00 + p&h**
6809E .... **$25.00 + p&h**
SAM Chip .... **$30.00 + p&h**
PIA's .... **$15.00 + p&h**

PRINTERS & PACKAGES

PRINTER PACKAGE ONE:
CP80A Dot Matrix 100cps 80 column printer
- Parallel/Serial Printer Interface
- VIP Writer (the best word processor for the CoCo)
- Graphicom (the best graphic drawing and print dump program) .... **$449 + p&h**

PRINTER PACKAGE TWO:
ENP1091 Dot Matrix 120cps 80 column printer
55cps near letter quality
90cps proportional
- Parallel/Serial Printer Interface
- VIP Writer
- Graphicom .... **$549 + p&h**

PRINTER PACKAGE THREE:
VCM 20cps Daisy Wheel printer (uses Diablo print wheels)
- VIP Writer
- Parallel/Serial Printer Interface .... **$549 + p&h**

SOFTWARE

Star Trader — Graphic Space Adventure 32K ECB .... **$29.95 + p&h**
Middle Kingdom — Graphic Dungeons: Dragon Type Adventure .... **$29.95 + p&h**
Bomber Command — War Game .... **$29.95 + p&h**
Company Command — War Game .... **$29.95 + p&h**

SOFTWARE SPECIALS

MYSTERY PACK .... **$20.00 + p&h**
Six program tapes for $20.00. Contains a mixture of programs from utilities — adventure arcades in both Basic and Machine Code. While stocks last.

All products come with an unconditional Guarantee. If defective return within 15 days for replacement. **Dealer inquiries invited**

✂- -

SEND SELF ADDRESSED ENVELOPE FOR FREE CATALOGUE
To: SOFTWARE SPECTRUM, Box 2101 GPO, Adelaide 5000. **New Zealand Bankcard welcome**

Mr Mrs Miss/Ms .... Address .... P/code ....

Phone .... Please charge Bankcard No ☐☐☐☐ ☐☐☐ ☐☐☐ ☐☐☐☐☐☐ Exp Date .... Sig ....

128K *RAM CARD*

Price not available

INCREASE YOUR 64 K Co-Co OR Co-Co II TO 128 K RAM

- ☐ FITS COMPLETELY INSIDE YOUR COMPUTER.
- ☐ SWITCHES TWO NEW 32 K BANKS OF RAM IN AND OUT OF MEMORY.
- ☐ BANKS CAN BE MAPPED IN THE UPPER HALF OR LOWER HALF, OR CAN ALSO BE A SECOND COMPLETE 64 K BANK.
- ☐ SWITCH TABLES INCLUDED.
- ☐ SIMPLE INSTALLATION AND DOCUMENTATION.
- ☐ A MUST FOR OS-9 USERS.
- ☐ COMPLETE WITH 8 (4164) RAM CHIPS.
- ☐ PAL CHIP HANDLES ALL BANKING COMMANDS.

OS9 SOFTWARE

Opack — Utilix — Filter Kit — SDisk & Bootfix — OS9 — Basic09 — Dynamite+ — Color Sleuth — RMS (Record Management System) — Stylograph Mailmerge — Spelling Checker Accs Receiveable and Pavable Inventory (Control system for retailer and distributor)

AGENTS:
J Fricker Qld 07 262 8869
K Enright 09 36 26472
S Wilson Act 06 25 12339
Bayne & Trembath Vic & Tas
Ian Lee South Australia
L Wrigley 75 Molesworth St
Lower Hutt NZ

PHONE NOW

Blaxland Computer Services

(THE COLOR COMPUTER SOFTWARE SHOP)

P.O. Box 125 BLAXLAND 2774

047 39 3903

ToTo Advertising

For all advertising in CoCo and Rainbow. . . .

phone 075 39 2003

TIRED OF GAMES??

Coco Accountant
Domestic and Small Business accounts program with spreadsheet. Tape or Disk $49.50

Small Business Accounts Receivable
Up to 500 Debtors with auto ageing and one key billing. 16K Disk only.$49.50

The Expert
An intelligent data base matching inputs to stored data as 'perfect match' or 'percentage match.' We can't describe it here - send for information. Useful to Farmers,Nurserymen or any small business.$35 16K Disk only.

You're Joking!
Your computer tells jokes at random intervals using your Tandy Speech-Sound Pak. Hilarious fun at parties and you can add YOUR jokes to the existing data base. $35 Tape

WORDZ series
Vocabulary building educational game with FRENCH/ENGLISH, SPANISH/ENGLISH, ENGLISH or !BLUE! vocabularies. Specify Language and Tape or Disk. $27.50

Psytest
Psychological profile test determines and reports on peoples character analysis. Not a game. Tape or Disk $27.50

Programmers Apprentice
More than 30 of the best BASIC routines used by the Coco, including INVERSE video, BREAK disable, LIST disable, sound effects and auto screen formats. Tape only. $19.50

Coco Handiman
Does the materials estimating and costing for painting, papering,tiling, panelling etc. Disk only. $35

THATSLOTTO Popular Lotto number generator and Pools Pik $15

Chip Picker Punters pal makes quick selections from daily Form guide. $15

BlackJaq. Popular game that teaches counting as well. $15

— PROFESSIONAL RACING PROGRAMS —

"PASTMASTER" Making a profit as usual. Send for details. Tape or Disk. $75

"HARNESS MASTER" Trotting equivalent of "Pastmaster" $49.95

Phone 055 23 3793 or write for information on these programs

SILICON SYSTEMS – P.O. BOX 392 - PORTLAND - VIC 3305

Founder GREG WILSON

AUSTRALIAN EDITOR AND PUBLISHER
Graham Morphett

CO-EDITOR
Kevin Mischewski

EDITOR'S ASSISTANT
Sonya Young

AND GRATEFUL ASSISTANCE FROM
Brian Dougan
Peggy Annabel
Richard and Judy
Bob Thompson
Paul Humphries
Alex Hartmann
Andrew Simpson
Jim & Sheryl Bentick
Patric Simonis
Annette Morphett
Glen Mischewski

COVER DESIGN
Jim Bentick

All Programs in this issue of RAINBOW are available on cassette tape

ADVERTISING
ToToAdvertising P.O.Box 5730 Gold Coast Mail Centre, Bundall, Qld.

DEADLINES
The 7th of the preceeding month of publication

OS-9
Kevin Holmes is the contact for OS-9 information. He also has access to OS-9 Software from the U.S.
His address is:— 39 PEARSON ST., NARARA, N.S.W. 2250

Printed by:
Australian Rainbow Magazine
P.O. Box 1742
Southport Qld. 4215

PRINT #-2,

As I write this, it is late in the evening.
The frantic work of the past week is over, and I've had time to enjoy some telephone conversations again as I 'buttoned up'.
We've grown to be quite a family. I am continually amazed by your warmth and friendship, even when we stuff up.
The reason I've been working back tonight is that there were a lot of invoices and other bits of paper inadequately followed up during the hectic days of 1984. As we weren't sure where we were, we sent letters to a number of people regarding these invoices, and the replies have been coming back, and creating a minor flood of paper!

As you are aware, although we are soon to get a brace of Tandy 1000's, we currently use CoCos exclusively in the office, and have never been dissatisfied with their performance. Even Greg's one - the only one that plays up at all, (we're sure he's trying to get at us!), is only a problem because the keys stick more than usual, because of the amount of cigarette ash he used to pour down on the keys! The CoCoLink CoCo and one other, run 24 hours a day - faultlessly.

We have a couple of old grey cases, a couple of new grey cases, a white long case which has two switchable ROMs, switchable reverse video, two styles of lower case, and reads more tapes than any other CoCo in the country! And we have a short case.

In fact we will probably get another short case soon - there's a variant due soon with some nice little extras!

Which is best? No doubt about it, the later grey cases had the most going for them, especially if you wanted to indulge in a little hacking. The white long case wasn't a computer that I could warm to, particularly because of the keyboard, but the new computer - the short case, is great as long as you dont want to hack. I like the keyboard, although I still think that the grey case keyboard is perfectly satisfactory. But the big plus for short case is the steady picture. Used with our Sony KV-1430AS TV, a TV with a computer RF input at the front, in addition to the normal input at the rear, we get a picture of unerring quality - virtually monitor standard.

INDEX

Blaxland Computer Services have just supplied a GM-1211 monitor and monitor mod. For a day or two we've had the pleasure of working with a very professional looking and performing unit! Telewriter even looks acceptable on it!

We have hitched a computer that has reverse video built-in, to the monitor, and I couldn't be more pleased with the overall effect.

Now some quick news:

1. The electricty strike in Queensland could have cost us the magazine this month but for some very dedicated effort by Jim, Sheryl and Sonya; and also by our printers, Assembly Press of Springwood in Brisbane, who bought a generator to keep their employees in a job, and us happy!

2. We have a games contest going in Aust CoCo - submit your game before May 31st, and be eligible to win a disk drive from Software Spectrum, or a 128K upgrade from Blaxland Computer Services, or a Tandy voice pack from Bayne and Trembath. Games submitted since August 1984 are eligible.

Our gratitude goes to the three suppliers who enthusiastically sought to be a part of the contest. Their prizes will be presented at CoCoConf.

3. CoCoConf - Tutorials are being finalized and details will be announced next month. So far the choice includes:

* One or two on OS9.
* a MS DOS / Tandy 1000 tutorial.
* a 128K tutorial.
* a tutorial for MC 10 users.
* Computers in Education.
* Hardware Hacking.
* At least one on BASIC.

Lonnie Falk (American Rainbow) is also talking about coming!

Please book early. I need to know of your intent - even if you are not in a position to pay at present.

4. The OS8 chips arrived this week and will be distributed free to subscribers only, with April's edition. Thanks go to Bayne and Trembath and Blaxland Computer Services, again (!), for making the supply of

continued on p 16

COCOCONF

15 - 16 JUNE, 1985

9.00 AM Rotary Hall
Lawson St
Southport. Qld.

* TUTORIALS
* FREE ticket to the Computer Expo
* See and operate the latest in Hard and Software.
* Pick up a bargain.
* Catch up on old friends.

PROGRAM

SAT:

9.00 AM Welcome!
9.30 AM Tutorials. Choice of 4, or head off to the Computer Expo.
11.00 AM Morning Tea.
11.30 AM Return to Tutorials.
12.30 PM Lunch.
2.00 PM Tutorials. Choice of 4, or head off to the computer Expo.
3.30 PM Afternoon Tea.
4.00 PM Return to Tutorials.
5.00 PM Break to prepare for Dinner.

8.00 PM Dinner (Venue to be announced).

SUN:

10.00 AM Spend today with the Software Agents. Try out the new Programs, or join in the games contests.

Tutorials subjects are yet to be finalised, however it is likely that Tutorials on 128K, Educational use of the Computer, the Basic Language (Beginners), The Basic Language (Experienced), Hardware Hacking, and more will be available.

Apply now. We need to know if you are coming. Cost of ticket includes entry to CoCoConf, Computer Expo, and entry to the Tutorials.

COCOCONF

Name:.....................................

Address:..................................

....

.... PC.....

I enclose full price $ 39.95

I enclose part price $ 9.95

and will pay the rest off before CoCoConf.

Please bill my Mastercard / Visa / Bankcard NO ....

Please find Cash / M.O. / Cheque enclosed. Signed..................

LETTERS

To Graham,

I think that Australian Rainbow is one of the best computer magazines around. Keep up the good work!

Grant Menner,
Parkwood, WA

Grant,
We agree!
Graham.

Dear Graham,
VIP Calc: My problem appeared in the Dec/Jan Rainbow. I am still hopeful that someone will have a satisfactory solution. The error doesn't always occur. Since I wrote, Peter Turner (Adelaide Micro Users) spent some time on it for me and found a solution which works even if it is a cumbersome overkill. Special thanks to Peter.
The solution was to change from default 'SINGLE' (8 digit) to 'DOUBLE' (16 digit) precision ... why this should be necessary for dollars & cents (two decimal places) is a mystery to us. In that mode, it is S-L-O-W especially when loading or printing (definately a 'mow-the-lawn-while-you-wait' job!) Fortunately, one can use 'SINGLE' and 'M'anual calculation until absolutely necessary. If the occasional error has not occured, printing can be done on 'SINGLE' too.
Junkfood: - Dec/Jan Rainbow.
I typed the four data listings for this machine language game, loaded and successfully ran the first three; - however the fourth would not run. It gave me "FC ERROR IN 10".
The first three parameters of x appears consistant with the other three listings so I suspect the second one - "15988". Is there an error there?
Sopwith CoCo:- Dec/Jan Rainbow.
Other readers probably found this error too. Something is missing between lines 1835 and 1850. Another Adelaide Micro User Group member, Robert Dalzell hunted up the earlier version of this updated program and found that the line 1835 is not the same and 1839 and 1840 come before 1850. I won't quote them here because it didn't work successfully. The article does not instruct the new Rainbow reader HOW to run it, either. I determined from line 56, where 'S$=INKEY$' and 'S=ASC(S$)-64' that if S is to be '-16', one has to press zero.
Well I have an instrument panel, I can use the joysticks for throttling and maintaining a heading and trim, but cannot get anywhere. I even ran out of fuel and nothing happened. The Altimeter keeps rotating like a clock. Do I need to get the earlier copy of Rainbow or can the instructions & corrections be published for the benefit of new subscribers? I'm no pilot!!! Maybe you can help me get my wins.
Allan Thompson.
O'Halloran Hill, SA.

Allan,
As far as we know 'Junkfood' is OK.
'Sopwith' was wrong, and the corrections are in this magazine. Unfortunately, you got the entire article, there were no other instructions.
I sympathize with you over the problems you are experiencing with VIP Calc. It was for reasons such as yours, that I stopped using prepared programs, particularly from the VIP range. Usually you can write a basic program which will do your job more simply, and often faster.
I have asked Kevin, (who is a pilot), to prepare an article that will more fully explain the two simulators available, (Sopwith, and Worlds of Flight).
Graham.

Dear Graham,
It seems last week wen I reseved your magazen (November issue) Australian Rainbow, I reseved two of them. They are exectly the same, I think you should look at thies matter, becouse I don'nt want to reseve two magazens which are the same. I would like to ask is, Austraian CoCo-the same as Australian Rainbow? Was I soppose to get one of each of thous magazens? and you've made a mistake and sended two the same? Anywhow could you please look into it,-I like reading you magazens,even thouw it dos'nt look like a million dollor magazen, its very informative. Keep up the good work.
Gordon El Nahi.
Coburg, VIC.

Gordon,
Australian CoCo is written by Australians and New Zealanders. The programs have been coming in for some time as source material for CoCoOz. There were things one couldn't do with CoCoOz, like hardware articles and reviews, and Greg decided that it was time to start an Australian Magazine, based on CoCoOz.
Australian Rainbow, conversely, is loosely based on American Rainbow, but has an increasing input from talented Australians who have specialist interest in Education, Business, OS9, and 128K upgrades. Australian Rainbow aims to have something for everyone but aims to meet the needs of the user who has had a CoCo for a little while.

Like many others, your subscription was caught up in the confusion created in November when we used our 'upgraded' Data Base for the first time. We knew we were going to stuff up, but it was unavoidable, and so we sent a second magazine where we were uncertain as to whether or not the first one actually went.
If this seems unbelieveably wasteful, and stupid, and even pointless, well, the point was that we didn't want you to miss out! It was the only thing we could do at the time. And yes it was wasteful!
Since then, we have been able to get the new Data Base fully operational and we have, in fact, been discussing it in this magazine.

I enjoyed reading your letter. We didn't correct your spelling because we thought your letter looked great the way it was! Besides, we could at least read yours and could tell who wrote it, which is more than I can say for quite a few of the letters we get here!
Graham.

Dear Graham,
Saw your article on the Brother EP44, and thought I'd just add that I use one and think that it is a good printer cum typewriter.
The only complaint I have is the cost of ribbons! A big advantage is that you can type information into the Brother's memory, and when it is ready, it can be transfered to CoCo for storage, and reprinting at a later stage.
Conceptually, it could be used as a terminal to the Color Computer, but I haven't tried any programing for that, I've only done the bit to transfer from the Brother to CoCo.
George T McLintock
Narrabundah, ACT.

George,
Brian Bere-Streeter (BrisBiz), swears by the EP44. But then they say he'll swear at most things!
(Thats not true.) We were impressed by the final product, of both Brian's and your machines. Guess we'll have to get one in here and have a play ourselves!
Graham.

Dear Graham,
I am looking for a good word-processing program to run on a TRS-80 colour computer, (basic model only - not extended basic), that I can load from cassette to use as part of my own programs for estimating routines and calculations.
I have the TRS 80 Scripsit cartridge but of course, but it can not be used with other programs - as far as I know.
I would appreciate if you could let me know of any programs available and cost etc.
Gordon Glashier.
Nth Sydney, NSW.

Gordon,
Using SCRIPTSIT, if you save text to tape using the PRINTER routine, you save to ASCII.
It is then possible to write a small program to then modify text, or insert it into something else.
Such a program need be only a few lines long. Something like this would suffice:

```
10 OPEN"I",#-1,
20 FORT=1TO500
30 IFEOF (-1) THEN 60
40 INPUT#-1,A$(T)
50 NEXTT
60 CLOSE#-1:FORT=1TO500
70 PRINTA$(T)
80 NEXTT
90 END
```

You can see how readily this program could be adapted to provide multiple copies and merged letters.
So I wouldn't put SCRIPTSIT down yet, it is very useful - even if the ASCII files from it wont load into TELEWRITER!
Graham.

EDUCATION PAGE

The CoCo Grade Book became available recently, and adds an additional dimension to CoCo's work in the classroom.

Unfortunately, the program is available only on disk. This is because the program calls the files in varying ways during it's operation, a process which would make a taped version quite clumsy.

The program is in two parts.

SET UP automatically clears old files from the disk, and gets the initial data. Data can also be added during main program use.

GRADE BOOK records the results of 15 tests against the names of 50 pupils in 10 classes. Cumulative grades can be obtained, which are reported according to the teacher's preference, ie numeric, percentage, alphabetical, weighted.

There are other functions available too, which call your attention to students who fall below standard, and which provide print-outs of results.

All in all, a handy program to have around, and at $49.95, including manual, not expensive.

The Timetable program the same company, Silicon Systems, had hoped to have ready for CoCo in February, has met with a few problems, and is now expected mid year.

The nice thing about dealing with a Software company like Silicon is that they really want to make the program work for you. Paul Worden assures me that if you want additional features, or even if you just want parts of programs you buy from him modified, then you only have to yell, .. well, something like that!

SUGAR No 2 arrived and has reviews of 4 programs for Apples, 2 programs for BBCs, 4 programs for Commode 64s, 2 programs for Microbees, 4 general use programs, and 4 for CoCo.

The 4 CoCo programs reviewed are:

Timebound - found to be satisfactory, but not of specific support to the current curriculum,

Cookie Monster's Letter Crunch - found to be generally satisfactory,

Musica - found to be of use to the enthusiast, but of little help to the teacher,

Quiz Maker - found to have special value in that it doesn't depend on the school having a lot of computers - the tests can be printed, once formulated on CoCo.

The reviews are brief but fair. Of the brace of CoCo programs mentioned, I feel that Quiz Master and perhaps Letter Crunch are the only ones that would get repeated use in the average classroom.

We are starting to get feed-back from a number of schools, which have started their own Computer Awareness Projects.

One of the local schools with an Awareness Project has listed the following as its aims:

1. All students in the school will be able to demonstrate an elementary understanding of :

i. What a computer is,

ii. Common applications of computers; eg, cash registers, automated bank tellers, telephone equipment, traffic control.

2. All students in the school will show an informed appreciation of the significance of computers in everyday life:

i. Benefits to society; eg, through improved information handling, better methods of communication, automation, the cashless society.

ii. Problems to be solved; eg, redundancy and retraining, methods of ensuring personal privacy.

Karel Davey, Tandy's Education contact in Sydney, drew my attention to the recent review of Logo in CREATIVE COMPUTING. This magazine found that Tandy's Logo for CoCo, (alliteration intended!), had some minor short-comings, but that the documentation was excellent.

Which brings me to my pulpit - the others are often pretty, but then so is CoCo; sometimes they have features that we develop later; and occasionally they are technically superior; but none have the documentation and the SUPPORT CoCo has. And therefore they don't last, and therefore, in the washup, don't do as good a job!

(Here endth the first lesson!)

You will note from our CoCoLink section, that we have had some fun, and some difficulty, in maintaining our bulletin board during the first month of operation. Nonetheless, we are confident that by the time you read this, any remaining bugs will be of little consequence.

We invite educators to use the Education section as a forum.

Speaking of forums, there will be a 3 to 6 hour Tutorial (depending on demand), on the use of computers in Education at CoCoConf in June. Just another reason to be there!

In addition to the taped version of "Best of CoCoOz", announced last month, we have just released a disk version. The disk version has the advantage of loading each program simply from the menu program, and of course, loads faster than the tape, a big advantage for an harrassed teacher! 'Best of' was conceived as a starter pack for schools or teachers with new CoCos to get to know. Included are 14 of the best Education programs to appear on CoCoOz over the past three years. Best of CoCoOz #1 is $10.00 (Taped) or $21.95 (Disk).

TANK ADDITION

Dean Hodgson

Three years ago I wrote a whole clutch of educational games on the color computer for children at my school. Many were designed to help them gain confidence in themselves as learners while practicing tables. "Tank Addition" is one of those programs. It was part of a set of arcade-style games that included the now infamous "Maths Invaders".

Once typed in, the program can be CSAVEd and RUN as normal. The idea behind the game is to shoot moving tanks by firing correct answers to addition problems.

To begin you must first indicate which level to start at. Each level determines the set of facts to be tested; eg, level 5 will give you +5 facts. If you do well enough during a wave, the level will be automatically advanced. Many mistakes will have the level lowered. This is an important aspect in good teaching - adjust the difficulity to suit where the learner is at.

There are two parts to each wave. Part one has a cannon on the left (that's you) and five tanks on the right. Each tank carries an addition problem and they move toward the left, toward you! You position your cannon using the up and down arrow keys. When in place, type in the answer to the problem straight across and press either ENTER or SPACEBAR to fire.

If you were right, the tank explodes and vanishes. If you made an error, your answer is erased and you had better try again quickly! The tanks keep coming. If a tank reaches the green line, the game ends. Points for tanks vary with the difficulty level.

After each wave of tanks you will also face a large "bomber". This vehicle carries a much larger addition problem. You have to destroy it before proceeding to the next wave (which is faster).

The game can be used by children as something to play in their spare time, or organised into a supplimentary practice program. If so, I suggest each child keep a record card showing date, level worked, score, and level to do next time. My own research shows that if this type of program is used as a suppliment (which is its intention), then for best results, children should play every day but no more than two games at a time.

I would like to comment on a few interesting aspects of the program.

Paging techniques are used to create smooth animation. The trick is this:

1. Copy the viewing screen from pages 1 & 2 to pages 3 & 4 (PCOPY).
2. Switch working screen to pages 3 & 4 (PMODE1,3).
3. Pick tank and erase. (LINE, PRESET, BF).
4. Update tank's position and store.
5. Display tank at new position. (PUT).
6. Copy working screen back to pages 1 & 2 (PCOPY).
7. Switch back to working on pages 1 & 2. (PMODE1,1).

You do the erasing and positioning on unseen pages then copy them to the pages being viewed. This is more difficult in PMODEs 3 & 4 because you need way more memory and the copying process is slower. Hence my use of PMODE1.

The subroutine starting at line 9000 puts leading zeroes into the strings containing the scores (SC$ & HS$). This makes for a more interesting display.

The text on graphics display routine is at line 20. Data for the characters starts at line 50000. This particular character set was designed to allow the smallest sized readable characters in any PMODE. The displayed method is simple. The message to be displayed is put into variable QZ$. Position the cursor, give the text colour and size using a DRAW"BM x,y;Cn;Sn" statement. Then GOSUB20. Data for each character is in an array. Feel free to use this routine in your own programs.

Lines 3000 to 3390 handle the "monster bomber problem". They can easily be deleted or bypassed if you wish. Note that the problem gets harder with each wave and has a maximum answer of 99. The cannon wont hold more than two digits.

There is something else worth thinking about. This is the issue of violence and racism inherent in these types of games.

Frequently "invaders" are pictured as slanty-eyed little beasties coming in hoards that the lone fighter must stave off. Is this the type of environment we want our children to inherit? How do girls feel about it? These are issues we need to address when selecting educational or other software for our children. "Tank Adddition" and simular games do have a measure of simulated violence.

At any rate, many children have found the game of benefit, and of course, enjoyable.

THE LISTING:

```
1 '******TANK ADDITION***********
   *****BY DEAN HODGSON**********
10 GOTO10000
20 POKE65495,0:FORQZ=1TOLEN(QZ$)
:DRAWAZ$(ASC(MID$(QZ$,QZ,1))-32)
:NEXT:POKE65494,0:RETURN
90 A$="":POKE65495,0
100 I$=INKEY$:IF I$="" THEN200
110 IF I$=CHR$(13) OR I$=" " THE
NIF LEN(A$)>0 THEN600 ELSE 100
115 IF I$="^" OR I$=CHR$(10) THE
N400
120 IF I$=CHR$(8) OR (I$>"@" AND
 I$<"[") THEN IF LEN(A$)>0 THEN5
00 ELSE 100
130 IF I$<"0" OR I$>"9" OR LEN(A
$)=2 THEN100
140 A$=A$+I$:DRAW"BM4,"+STR$(GY+
8)+"S8C2":QZ$=A$:GOSUB20:SOUND20
0,1:GOTO100
200 Z=RND(5)-1:IF N(Z)=0 THEN200
205 X=A(Z,0):Y=A(Z,1)
210 PCOPY1TO3:PCOPY2TO4:PMODE1,3
220 GET(X,Y)-(X+62,Y+28),0,G:LIN
```

```
E(X,Y)-(X+62,Y+28),PRESET,BF
230 A(Z,0)=A(Z,0)-JD:X=A(Z,0)
240 PUT(X,Y)-(X+62,Y+28),O,PSET
245 REMDRAW"BM"+STR$(A(Z,0)+18)+
","+STR$(A(Z,1)+16)+"C2S8":QZ$=P
$(Z):GOSUB20
250 PCOPY3TO1:PCOPY4TO2:PMODE1,1
260 IF A(Z,0)>45 THEN100
300 POKE65494,0:COLOR4:FORZ=0TO4
:IF N(Z)=0THEN302
301 LINE(A(Z,0),A(Z,1)+4)-(38,GY
+12),PSET:PLAY"T255V31O2;CGCGCG"
302 NEXTZ
310 PUT(0,GY)-(62,GY+28),E,OR:SC
REEN1,0:PLAY EX$:PLAY EX$:PLAY E
X$:PLAY EX$
315 CLS3:PRINT@0,STRING$(32,246)
;:PRINT@480,STRING$(31,246);:POK
E1535,246
316 FORI=31TO479STEP32:PRINT@I,S
TRING$(2,246);:NEXT
320 AC=INT(HT/(HT+MI)*100):PRINT
@74,"SCORE";AC;"%";:PRINT@165,"Y
OU EARNED";SC;"POINTS";
330 PRINT@264," NEXT START AT
";:PRINT@296," SKILL LEVEL";SK;"
.";
360 PRINT@426,"press enter";:SCR
EEN0,1
365 R$=INKEY$:IF SC>HS THEN HS=S
C
370 IF INKEY$<>CHR$(13) THEN370
380 CLS4:PRINT@204,"BYE!!";
390 PLAY"L2;T30;O3;N1;2;3;4;5;6;
7;8;9;10;11;12;O4;T2;N1":GOTO200
0
400 REM
410 IF I$="^" THEN MV=-1 ELSE MV
=1
420 GX=GP+MV:IF GX<0 OR GX>4 THE
N100
430 PCOPY1TO3:PCOPY2TO4:PMODE1,3
440 LINE(0,GY)-(38,GY+24),PRESET
,BF
450 GY=GY+34*MV:GP=GX
460 PUT(0,GY)-(38,GY+24),C,PSET
465 IF A$<>"" THEN DRAW"BM4,"+ST
R$(GY+8)+"S8C2":QZ$=A$:GOSUB20
470 PCOPY3TO1:PCOPY4TO2:PMODE1,1
:GOTO100
500 LINE(2,GY+6)-(22,GY+18),PRES
ET,BF:GOTO90
600 IF N(GP)=0 THEN A(GP,0)=255
605 POKE65494,0:COLOR3:LINE(40,G
Y+13)-(A(GP,0)+6,GY+13),PSET:PLA
Y"V31T255O5GCGCGC"
610 LINE-(40,GY+13),PRESET:COLOR
2:LINE(44,0)-(44,170),PSET
615 IF N(GP)=0 THEN680
620 IF VAL(A$)=N(GP) THEN700 ELS
E NM=NM+1
670 LINE(44,0)-(44,170),PSET
680 LINE(4,GY+6)-(22,GY+18),PRES
ET,BF:GOTO90
700 PUT(A(GP,0),A(GP,1))-(A(GP,0
)+62,A(GP,1)+28),E,OR
710 PLAY EX$:LINE(A(GP,0),A(GP,1
))-(A(GP,0)+62,A(GP,1)+28),PRESE
T,BF
720 SC=SC+SK*W:HT=HT+1
730 LINE(54,178)-(118,191),PRESE
T,BF:DRAW"BM60,178C2S8":GOSUB900
0:QZ$=SC$:GOSUB20:IFSC<=HS THEN7
40
735 HS=SC:GOSUB9000:QZ$=HS$:LINE
(196,178)-(255,191),PRESET,BF:DR
AW"BM196,178":GOSUB20
740 LINE(2,GY+6)-(22,GY+18),PRES
ET,BF
750 N(GP)=0:NK=NK+1
760 IF NK<5 THEN20 ELSE3000
1000 PCLS1:COLOR2:LINE(44,0)-(44
,170),PSET
1010 GOSUB9000:QZ$="SCORE "+SC$:
DRAW"BM0,178C2S8":GOSUB20:QZ$="H
IGH "+HS$:DRAW"BM152,178":GOSUB2
0
1020 GP=2:GY=74:PUT(0,GY)-(38,GY
+24),C,PSET:W=W+1
1100 JD=W*2:NK=0:NM=0
1110 FORI=0TO4:A=RND(9):B=SK:IF
SK=10 THENB=RND(9)
1115 N(I)=A+B
1120 P$(I)=RIGHT$(STR$(A),1)+"+"
+RIGHT$(STR$(B),1):NEXTI
1125 SCREEN1,1
1130 FORI=0TO4:A(I,0)=RND(50)+14
0:A(I,1)=I*34+4
1140 PUT(A(I,0),A(I,1))-(A(I,0)+
62,A(I,1)+28),R,PSET
1150 DRAW"BM"+STR$(A(I,0)+18)+",
"+STR$(A(I,1)+16)+"C2S8":QZ$=P$(
I):GOSUB20
1160 PLAY"V31L255O3CBO4EO5CG"
1170 NEXT:GOTO90
2000 PCLS1:LINE(0,0)-(255,191),P
SET,B:QZ$="TANK":DRAW"BM70,8;S24
C3":GOSUB20:QZ$="ADDITION":DRAW"
BM24,40":GOSUB20
2010 PUT(186,156)-(248,184),R,PS
ET:PUT(10,156)-(48,180),C,PSET
2020 QZ$="HIGH SCORE "+HS$:DRAW"
BM50,80;S8C4":GOSUB20
2025 QZ$="PRESS":DRAW"BM98,100;S
8C2":GOSUB20:QZ$="ANY KEY":DRAW"
BM88,116":GOSUB20:QZ$="TO START"
:DRAW"BM86,130":GOSUB20
2030 PCOPY1TO3:PCOPY2TO4
2040 SCREEN1,1:DL=0
2050 A$=INKEY$:IFA$<>""THEN2100
```

```
2055 DL=DL+1:IFDL<100THEN2050
2060 COLOR4:LINE(176,162)-(38,16
4),PSET:PLAY"V31T25502;CGCGCG":L
INE-(176,162),PRESET:PUT(0,154)-
(62,182),E,OR:SCREEN1,0:PLAYEX$
2070 A$=INKEY$:IFA$<>""THEN2100
2080 DL=DL+1:IFDL<170THEN2070
2090 PCOPY3TO1:PCOPY4TO2:GOTO204
0
2100 SCREEN0,0:CLS:PRINT@41,"you
r controls ":PRINT@98,"UP & DOWN
 ARROW KEYS TO MOVE.   PRESS ENT
ER KEY TO SHOOT."
2105 PRINT@228,"ENTER SKILL LEVE
L (1-10)";:PRINT@331,;:LINEINPUT
"--> ";LE$:SK=VAL(LE$):IF SK<1 O
R SK>10 THEN2000
2110 PCLS1:SCREEN1,0:HT=0:SC=0:M
I=0:W=0:GOTO1000
3000 POKE65494,0:LINE(0,0)-(255,
176),PRESET,BF
3010 PUT(0,74)-(38,98),C,PSET:PA
INT(10,86),2,4
3020 PUT(188,62)-(252,110),M,PSE
T
3030 A=RND(W*10)+10:B=RND(W*10)+
10:C=A+B:IFC>99THEN3030 ELSEA$=S
TR$(A):B$=STR$(B)
3040 P$=RIGHT$(A$,LEN(A$)-1)+"+"
+RIGHT$(B$,LEN(B$)-1):DRAW"BM198
,82C3S8":QZ$=P$:GOSUB20
3050 PCOPY1TO3:PCOPY2TO4:PMODE1,
1:SCREEN1,0:TP=188
3060 A$="":PAINT(10,86),2,4
3100 I$=INKEY$:IF I$="" THEN3140
3110 IF I$=CHR$(13) OR I$=" " TH
EN IF LEN(A$)>0 THEN3200 ELSE 31
00
3115 IF I$=CHR$(8) AND LEN(A$)>0
 THEN3060
3120 IF I$<"0" OR I$>"9" OR LEN(
A$)>=2 THEN3100
3130 A$=A$+I$:DRAW"BM4,82C3S8":Q
Z$=A$:GOSUB20:SOUND200,1:GOTO310
0
3140 PCOPY1TO3:PCOPY2TO4:PMODE1,
3:LINE(TP,62)-(TP+64,110),PRESET
,BF:TP=TP-JD
3150 PUT(TP,62)-(TP+64,110),M,PS
ET:DRAW"BM"+STR$(TP+10)+",82C3":
QZ$=P$:GOSUB20:PCOPY3TO1:PCOPY4T
O2
3160 PMODE1,1:SOUNDTP,1:IF TP>45
 THEN3100
3180 POKE65494,0:COLOR3:FOR I=0
TO 30:LINE(TP,86)-(40,86),PSET:P
LAY"V31T25501DEFG":LINE-(TP,86),
PRESET:NEXTI
3190 GY=74:GOTO310
3200 POKE65494,0:COLOR3:LINE(40,
86)-(TP,86),PSET:PLAY"V31T25505G
CGCGC":LINE-(40,86),PRESET
3210 IF VAL(A$)<>C THEN3060
3300 PUT(TP,66)-(TP+62,94),E,OR:
PLAY EX$
3310 J=SC:SC=SC+C*10:HT=HT+1:IF
NM<1 THEN SK=SK+1 ELSE IF NM>3 T
HEN SK=SK-1
3312 IF SC>HS THENHS=SC
3315 IF SK>10 THEN SK=10 ELSE IF
 SK<1 THEN SK=1
3330 COLOR2:LINE(44,0)-(44,170),
PSET:GOTO1000
9000 A$=MID$(STR$(SC),2):IFSC>99
99THENSC$=A$:GOTO9020
9010 SC$=LEFT$("00000",5-LEN(A$)
)+A$
9020 A$=MID$(STR$(HS),2):IFHS>99
99THENHS$=A$:RETURN
9030 HS$=LEFT$("00000",5-LEN(A$)
)+A$:RETURN
10000 POKE65494,0:CLEAR100:PMODE
1,1:DIM C(12),R(23),E(23),M(44),
O(23),A(5,1)
10005 CLS3:PRINTSTRING$(33,150);
:PRINT@480,STRING$(31,150);:FORI
=63TO479STEP32:PRINT@I,STRING$(2
,150);:NEXT:POKE1535,150
10010 PRINT@104," TANK  ADDITION
 ";:PRINT@168," BY DEAN HODGSON"
;:PRINT@264," COPYRIGHT 1982 ";:
PRINT@328,"PYRAMID SOFTWARE";:SC
REEN0,1
10020 GOSUB50000
10030 PCLS1:DRAW"BM0,8;S8C4;D6F1
R11U1E2R5U2L5H2U1L11G1;BM4,4;C2;
R5BD10L5;BM8,2C3R1BD12L1":PAINT(
12,14),4,4:LINE(2,8)-(22,20),PRE
SET,BF:GET(0,2)-(38,26),C,G
10040 DRAW"BM110,118;C2S8NR9U1R9
C3;BM142,112L3G1L2D3L2D1R16U2H2L
1H1L3H1;BM126,124R16;BM122,126;C
4;NR20G1L1G2D1F2R1F1R20E1R1E2U1H
2L1H1L1":PAINT(136,136),4,4:PAIN
T(136,120),3,3:LINE(126,126)-(15
8,138),PRESET,BF:GET(110,112)-(1
72,140),R,G
10050 FORI=1TO99:PSET(RND(62)+10
0,RND(28),RND(4)):NEXTI:GET(100,
0)-(162,28),E,G
10060 PCLS1:DRAW"BM130,86S8C2;R1
E4R5E6U1R6D3F5R4D6L4G5D3L6U1H6L5
H4":PAINT(168,86),2,2:DRAW"BM140
,78;C3R5E6;BM140,94R5F6;BM154,62
C4R11;BM154,110R11;BM162,78NR9U5
E1R4D2F4R7D8L6NL9G4D2L4H1U5":PAI
NT(168,72),4,4:PAINT(168,100),4,
4
10070 GET(130,62)-(194,110),M,G
10080 EX$="T255V3101;6;4;2;5;7;9
```

```
;2;9;4;3;V22;8;2;8;4;1;7;6;3;2;7
;5;V14;7;7;9;2;5;3;1;8;V6;3;7;2;
1;1;7;3;7;9;6;2"
10085 HS$="00000"
10090 GOTO2000
50000 DIM AZ$(58):RESTORE:FORZ=0
TO58:READ AZ$(Z):NEXT:RETURN
50010 DATA BR4,BR2D2BD2D0BU4BR2,
BR1D1BR2U1BR2,BR1D4BR2U4BF1L4BD2
R4BU3BR2,BR2D4NL2R1E1H1L2H1E1R3B
R2,R0BF4R0BL4E4BR2,D4R1U4R1D4R1U
4BR2,BR2D1BU1BR2,BR2G1D2F1BR2BU4
,BR1F1D2G1BR3BU4,F2NU2NR2NF2ND2N
G2NL2E2BR2,BD2R2NU2ND2R2BU2BR2,B
R2BD3D1G1BE3BU2,BD2R4BU2B
50020 DATA BR2BD4D0BR2BU4,BD2R4B
H2D0BD4D0BU4BR4,D4R3U4L3BR5,BR1N
D4BR2,R3D2L3D2R3BU4BR2,R3D2NL3D2
L3BU4BR5,D2R2NU2ND2R1BU2BR2,NR3D
2R3D2L3BU4BR5,NR3D4R3U2L3BU2BR5,
R3D1G3BE4BR1,D4R3U2NL3U2L3BR5,D2
BD2R3U2NL3U2L3BR5,BR2BD1D0BD2D0B
R2BU3,BR2BD1D0BD2D1G1BE3B
50030 DATA BD2NF2E2BR2,BD3R4BU2L
4BU1BR6,F2G2BU4BR4,E1R2FD1G1L1D2
BU4BR4,BR4,ND4R3D2NL3D2BU4BR2,D4
R2E1H1NL2E1H1L2BR5,D4R3BU4L3BR5,
D4R2E1U2H1L2BR5,D2NR2D2R3BU4L3BR
5,NR3D2NR2D2BE4BR1,NR3D4R3U2L1BE
2BR1,D2ND2R3D2U4BR2,R2L1D4L1R2BR
2BU4,BD3D1R3U4BR2,D4U2RNF2E2BR2
50040 DATA D4R3BU4BR2,ND4F2E2ND4
BR2,ND4F3D1U4BR2,D4R3U4L3BR5,ND4
R3D2L3BR5BU2,D4R1E1NF1E1U2L3BR5,
ND4R3D2L3R1F2BR2BU4,NR3D2R3D2L3B
R5BU4,R2L1D4BR3BU4,D4R3U4BR2,D2F
2E2U2BR2,D4E2F2U4BR2,F4BL4E4BR2
50050 DATA F2ND2E2BR2,R4G4R4BU4B
R2
50070 DATA BD1NR3D1R1F1R1D1L3BU4
BR5,BR2D4BL1BU3R2BR2BU1,BD1D3R3U
3BU1BR2,BD1D1F2E2U1BU1BR2,BD1D3R
2NU3R2U3BU1BR2,BD1F3BL3E3BU1BR2,
BD1D1F2G2E4U1BU1BR2,BD1R3G3R3BR2
BU4
```

CORRECTION

We managed to omit the following lines from December's Rainbow. They are part of the 'Sopwith CoCo' program and were omited from pages 48-49.

THE LINES:

```
1839 IF SD/5280>10THEN1710
1840 CA=SB-(RB(S)-1.5708):DB=COS
(CA):IFABS(DB)>.2588 THEN 1710 E
LSE IF DB>.0523 THEN DB=.0523 EL
SE IF DB<-.0523 THEN DB=.0523
```

SILICON SYSTEMS SOFTWARE — CoCo Grade Book

TEACHERS TEACHERS TEACHERS

The "Coco Grade Book" is a valuable aid for teachers, eliminating the tedious work of grade calculations and report writing. It tracks and analyses grades for up to 50 students in 10 classes. 15 tests can be stored for each class, but using the 'accumulate function' a whole years grades can be stored and assessed. You need a 32K Coco with one disk.$49.95 with manual and P/P postage

Function Summary

Test grades and descriptions stored for every student.
Students names and Class descriptions are stored.
Add or delete Classes/Students/Tests.
Calculates cumulative grade points for each student.
Calculates percentage grades.
You can assign letter grades and set percentage cut-offs.
Calculates test mean scores.
Calculates the standard deviations of tests.
Drops the lowest test grades (optional).
Optional weighting of scores.
Generates hard-copy reports with optional message.
Complete edit control with options to add/delete/change all functions.
Flags students below your set cut-off

TEACHERS TEACHERS TEACHERS

```
ANALYSIS MENU
<1> DISPLAY TEST GRADES
<2> DISPLAY TEST DESCRIPTIONS
<3> DISPLAY ACCUMULATED POINTS
<4> SET LETTER GRADES AND GET % TOTALS
<5> STUDENTS NAMES/CODES/NUMBERS
<6> TEST MEAN SCORES
<7> STANDARD DEVIATION
<8> SYSTEM DATE
<9> HEADER MESSAGE FOR REPORTS
OR DO
<A> DROP A TEST
<B> ACCUMULATE/ ERASE AND GET TOTALS
<C> ADJUST SCORES
<D> RETURN TO MAIN MENU
```

```
MAIN MENU
---------
<1> ENTER GRADES
<2> CHANGE GRADES
<3> CHANGE TEST DESCRIPTIONS
<4> CHANGE STUDENT NAME/CODES
<5> CHANGE NUMBER OF STUDENTS
<6> ANALYSIS (MENU)
<7> GET CLASS INFORMATION
<8> SAVE DATA AND QUIT
<9> QUIT WITHOUT SAVING DATA
```

```
OPTIONS MENU
------------
<1> CHANGE NO. OF CLASSES
<2> CHANGE CLASS TITLES
<3> GET GENERAL CLASS INFORMATION
<4> RETURN TO SELECT CLASS
```

bankcard
welcome here

FOR MORE INFORMATION ON THIS PROGRAM CONTACT:-
SILICON SYSTEMS SOFTWARE - P.O. Box 392 - PORTLAND - Victoria 3305

16K ECB

An Open-Ended Exploration

By Joseph Kolar

The more you learn, the more you realize how little you know. That is true as far as the CoCo is concerned, and that is why every session at the keyboard is an adventure in learning.

The new CoCo owner bought his versatile machine for its graphics capabilities, among other things. We will explore the *POKE* and *PEEK* BASIC Statements as they apply to the text screen page.

I can't be sure what we'll do, but fire up CoCo and let us proceed line by line and investigate whatever comes to mind. It is a good, open-end way to learn and still have fun.

The text screen is what you see when you turn on the CoCo. It is your working area. *PRINT@* locations 0 to 511 cover all 512 locations on the text screen. Key in:

```
1 CLS
10 PRINT@ 10, CHR$(128)
100 GOTO 100
```

Line 10 tells CoCo to print at the eleventh space of the top row, a black square. This is due to the first upper left-hand location being designated as 0. To verify that this is so, add and *RUN:*

```
11 PRINT@0,"12334567890";
```

Don't forget to add the semicolon. Now, delete the semicolon and see what happens. To help yourself learn, say to yourself, "Having deleted or omitted the semicolon, the black block, *CHR$(128)*, vanished. When the semicolon was restored, the black box was visible. So, what have I discovered?"

Insert an apostrophe or *REM* marker in front of 'P' in Line 11 and insert Line 9, copying the information in Line 11 ending with a semicolon. *RUN* and observe. Press BREAK, then delete the semicolon in Line 9 and *RUN.*

BREAK places the apostrophe (') marker in front of 'P' in Line 9. Delete the apostrophe in Line 11 and recheck both with and without the apostrophe.

You should have noticed that when Line 9 precedes Line 10, it is not necessary to add the semicolon. However, if you placed the information in Line 9 following Line 10, it is a different story! It is left to you to mull it over in your mind and figure out why this is so. There is no better way for a beginner to learn something than to work it out for himself.

When you are finished, you may *DEL9*, or if you prefer, keep it as a *REM* line in your program. It is harmless.

Please note that using *PRINT@* allows you to print a string of characters, such as 'RAINBOW' when enclosed in quote marks and separated from the location value by a comma. You can print the ASCII character codes using *CHR$(x).* The characters from 128 through 255 will create block graphics.

If you are unfamiliar with these graphics blocks, key in the following routine:

```
5 GOTO 200.
```

This line gets us around our routine, which we will refer to later.

```
200 FOR X=128 TO 255
210 PRINT@240, CHR$(X)
220 FOR Z =1 TO 200: NEXT
250 NEXT X
299 GOTO 299
```

Each *CHR$* character from 128 through 255 will be printed, in rotation, in the middle of the display screen. Add:

```
211 PRINT@270,X
```

This will give the numerical value of each shape displayed. It will also help you visualize each shape which will be directly above the second digit.

If you want to see the other characters, change Line 200:

```
200 FOR X= 33 to 127
```

CHR$(32) is a blank space, creating a space just as the space bar does. The low numbers are control codes and do not generate a visible display.

To keep this routine for later reference, put an apostrophe marker in Line 5. You will hold the routine harmless, but available.

Get in the habit of using the *REM* marker to hide or uncover program lines and routines. You will get lots of mileage out of this handy tool when you are experimenting or creating your own original work.

This *PRINT@* Text Screen can be accessed using *POKEs*. The memory locations of the Text graphics page begin at 1024. This memory location is equivalent to *PRINT@ 0*. The memory locations continue just as the *PRINT@* location and ends as memory location 1535, which is in the lower right-hand corner. It can also be called with *PRINT@ 511*.

Note that *PRINT@ 511-0* and memory location 1535-1024 both equal 511. Allow 1 for the location you are subtracting and you get 512, the total number of all possible locations.

Each one of these text screen locations may be accessed by means of *POKEx,y*, where 'x' is a specific location from 1024 to 1535 and 'y' is a value from 0 to 255.

So, what is *POKE* anyway? It is a statement that allows CoCo to place into a designated location whatever text screen character you desire. It has other uses not within the scope of this article. Press BREAK and add:

```
20 POKE 1066, 255 RUN.
```

This placed an orange block directly underneath the *PRINT@ 10*, black block.

PEEK allows you to look at a specified memory location to see what information, if any, resides there.

Press BREAK, *PRINT PEEK(129)*, and ENTER. This memory location is checked on I/O Error message when *CLOADing* a program from cassette. A zero means that memory is no good and if a one is returned, it signifies that the tape is no good. Try this:

```
PRINT PEEK(1066) ENTER.
```

The value of 96 is returned. This 96 represents 'blank' (empty). The reason for this is that we are not in the program, having broken out and location 1066 reverts to its original state, 'blank'.

The observant newcomer will notice that *CHR$(96)* is a reversed '@'. Verify this by unmasking Line 5. (Remove the apostrophe.) Mask Line 200 with '. Then add:

```
201 FOR X=96 TO 96 RUN.
```

This is a lazy person's way to substitute a single value in a *FOR TO* statement. If you used 201 X=96 you would still get the correct answer, but you'd also get an NF Error in 250. If this was an integral part of a real program, it would bomb out unless Line 250 was deleted.

This 'one value' hint is valuable when you may be experimenting with different values. OK! Press BREAK, mask lines 5 and 201 and unmask Line 200.

A disturbing fact remains. *CHR$(96)* and the 96 that was revealed by *PEEKing* at memory location 1066 are different. There are some differences between the ASCII characters using *CHR$* and the characters that CoCo recognizes from 0 to 255.

To compare the *POKEd* characters with the *CHR$* characters, change Line 200 and add Line 211:

```
200 FOR X=0 TO 127
211 POKE1269,X RUN
```

Let's make it neater. Press BREAK and change lines 210 and 211:

```
210 PRINT@234, CHR$(X)
211 PRINT@238, X
```

The graphics blocks from 128 to 255 are the same. If you want to check this out change Line 200 to include whatever values you care to compare.

Remember, the character displayed at the left, if any, is the ASCII code, and the character on the right is what CoCo will read for the same value when it is *POKEd* into a memory location.

You are urged to make a reference table of the two sets of characters, side by side, insofar as they differ.

Now press BREAK, and mask Line 5 again.

To demonstrate that one set can be substituted over the other, Line 23 will *POKE* an orange box over the black box at *PRINT@ 10* and Line 24 will superimpose a black box over the *POKEd* orange box, using *PRINT@*. Add and *RUN*.

```
23 POKE 1034, 255
24 PRINT@42, CHR$(128)
```

POKEing graphics characters is one quick way to cover large areas. Press BREAK and add:

```
6 GOTO 300
300 FOR L=1024 TO 1055
310 POKE L,255
320 NEXT L
500 GOTO 500
```

This creates an orange line that covers all the text screen locations on the top row. Add:

```
330 FOR M=32 to 63
340 PRINTM, CHR$(175)
350 NEXT M
```

Using blue, *CHR$(175)*, we can use *PRINT@ M*, all locations in the second row to fill them in. *RUN* BREAK and to make a left border add:

```
360 FOR L=1024 to 1504 STEP 32
370 POKEL,255
380 NEXT L
```

Since we want only one vertical column, in Line 360, we start at 1024 and skip 31 columns to put a dab of color in each 32nd, or left-hand, row. Purists will note that we should begin with location 1056, but it is easier to go over the corner block in the top row.

Using *PRINT@*, we will create an orange border on the right-hand side. Press BREAK and add:

```
385 FOR M=479 TO 31 STEP-32
390 PRINT@M, CHR$(255);
395 NEXT M
```

We went from bottom to top for a change of pace. Note that we were unable to use *FOR M=511* etc. because filling in this corner box would cause the screen to scroll up one row. Omit the semicolon at the end of Line 390 and watch a disaster area. We could fill that corner location safely with a *POKE* to avoid that pesky scroll. We do so when we create the bottom border. Press BREAK, add and *RUN*:

```
400 FOR L=1505 TO 1535
410 POKEL, 255
420 NEXT L
```

We can *PEEK(x)* a value while we are in the program. We will ask CoCo to check if memory location 1503 is orange, (255); *PRINT@ 237, "ORANGE"* and go ahead and create the bottom border. If 1503 is not orange, forget about the bottom border and skip to the end of the program. Press BREAK, add and *RUN*:

```
399 IF PEEK(1503)=255 THEN PRINT237,"ORANGE"; ELSE 500.
```

To verify that this works, substitute 255 in Line 399 with another value and try it. As an alternate, pick an arbitrary *POKE* location, from 1024 to 1535 to see if it is orange.

You can *POKE* characters, other than the graphics values but, except for an asterisk or plus sign, which create neat borders or accents, it is silly to create a border of reversed @. Listing 1 will show an example of *POKEing* alphabetic characters.

At this time your mind is racing ahead with projects to try out. Before you do, put in the three missing blue sides of the inner border.

And, let's have some fun! Create a half-screen full of reversed @. Adjust the *POKEd* locations so the display is centered horizontally on the screen with a green band on the top and bottom. Open a partial row in the middle of the screen, leaving one blank space at each end and *POKE* your first name into the cleared space. Create a pause so your name may be read and then blank out the name slot with some graphics block.

Doing this exercise will give you ideas to either modify and improve what you have created, or go off into a frenzy of creativity in another direction.

Listing 1 is an example of using all *POKEs* to create a demonstration program which is somewhat similar to the exercise above.

One advantage of using the graphics characters, 128-255, is that you get to use all the colors available on your palette. You need not be an artist to have fun creating whatever your mind's eye conceives. You may wind up with some pretty impressive concoctions.

Some notes on Listing 1. Line 140 puts the top row of graphics characters on the screen the hard way — one at a time with an appropriate pause. Compare Line 140 with lines 180-210, which create the bottom segment.

There is no *RETURN* after Line 310, a *GOSUB* routine. This was a boo-boo. Since a similar routine follows, this effectively makes the pause 230 instead of 200. Can you see why? No harm was done and I failed to notice it.

Line 160 has no *GOSUB* pause between the two *POKEs* because they go onto the display as one unit. Line 230, the left border has a small pause between units so it blends nicely with lines 250-280, which override the text.

Line 100 does not have the 'short' pause. It seemed to look better to have 0! come on as a single unit. The 'long' pause is used only before and after HELLO!

Read the listing and figure out what each program line does. Except for the two pause routines at the end, it is a linear program and each routine follows exactly as it appears on the screen.

Hopefully, you will have some ideas to modify, expand or enhance this listing, so what are you waiting for?

Note Listing 2 should not be keyed in. Just compare it with Listing 1. It is the same as Listing 1 except it is tightened up using multiple program lines. Two changes, the missing *RE TURN* was added to Line 310 and in Line 10, 20 was changed to 10 due to deletion of Line 20 from Listing 1 and subsequent UL Error message. Which listing would you rather key in?

Listing 1:

```
Ø '<LISTING1>
1Ø CLS
2Ø C=RND(255)
3Ø IF C<144 THEN 2Ø
4Ø GOSUB31Ø
5Ø POKE 126Ø,96
6Ø POKE 1261,72:GOSUB32Ø
7Ø POKE 1262,69:GOSUB32Ø
8Ø POKE 1263,76:GOSUB32Ø
9Ø POKE 1264,76:GOSUB32Ø
1ØØ POKE 1265,79
11Ø POKE1266,97
12Ø POKE1267,96
13Ø GOSUB31Ø
14Ø POKE1226,C:GOSUB32Ø:POKE1227
,C:GOSUB32Ø:POKE1228,C:GOSUB32Ø:
POKE1229,C:GOSUB32Ø:POKE123Ø,C:G
OSUB32Ø:POKE1231,C:GOSUB32Ø:POKE
1232,C:GOSUB32Ø:POKE1233,C:GOSUB
32Ø:POKE1234,C:GOSUB32Ø:POKE1235
,C:GOSUB32Ø:POKE1236,C:GOSUB32Ø:
POKE1237,C
15Ø GOSUB32Ø
16Ø POKE 1268,C:POKE 1269,C
17Ø GOSUB32Ø
18Ø FOR X=13Ø1 TO 129Ø STEP-1
19Ø POKEX,C
2ØØ GOSUB32Ø
21Ø NEXT X
22Ø GOSUB32Ø
23Ø POKE 1258,C:GOSUB32Ø:POKE125
9,C
24Ø GOSUB32Ø
25Ø FOR X=126Ø TO 1267
26Ø POKEX,C
27Ø GOSUB32Ø
28Ø NEXTX
29Ø GOSUB32Ø
3ØØ GOTO 1Ø
31Ø FOR Z=1 TO 2ØØ:NEXT
32Ø FOR Z=1TO 3Ø:NEXT
33Ø RETURN
```

Listing 2:

```
Ø '<LISTING2>
1Ø CLS:C=RND(255):IFC<144 THEN 1
Ø:GOSUB31Ø
5Ø POKE 126Ø,96:POKE1261,72:GOSU
B32Ø:POKE1262,69:GOSUB32Ø:POKE12
63,76:GOSUB32Ø:POKE1264,76:GOSUB
32Ø:POKE1265,79:POKE1266,97:POKE
1267,96:GOSUB31Ø
14Ø POKE1226,C:GOSUB32Ø:POKE1227
,C:GOSUB32Ø:POKE1228,C:GOSUB32Ø:
POKE1229,C:GOSUB32Ø:POKE123Ø,C:G
OSUB32Ø:POKE1231,C:GOSUB32Ø:POKE
1232,C:GOSUB32Ø:POKE1233,C:GOSUB
32Ø:POKE1234,C:GOSUB32Ø:POKE1235
,C:GOSUB32Ø:POKE1236,C:GOSUB32Ø:
POKE1237,C:GOSUB32Ø
16Ø POKE 1268,C:POKE 1269,C:GOSU
B32Ø
18Ø FOR X=13Ø1 TO 129Ø STEP-1:PO
KEX,C:GOSUB32Ø:NEXT:GOSUB32Ø
23Ø POKE 1258,C:GOSUB32Ø:POKE125
9,C:GOSUB32Ø
25Ø FOR X=126Ø TO 1267:POKEX,C:G
OSUB32Ø:NEXT:GOSUB32Ø:GOTO1Ø
31Ø FCR Z=1 TO 2ØØ:NEXT:RETURN
32Ø FOR Z=1TO 3Ø:NEXT:RETURN
```

Software Review

Pak-Panic — The Old Game With A New Twist

With centipedes, monsters, invisible mazes and ghosts that can go through walls, *Pak-Panic* from Tom Mix Software is unique compared to all of the competition. *Pak-Panic* is a 32K 100 percent machine language, arcade-style game that uses the left joystick and firebutton.

The scenario is as follows: You are Pakman. Your job is to go around the screen eating dots, power pills, and bonus prizes while avoiding monsters.

A power pill is one of the larger dots on the screen. Seven are on levels one through four. Six are on levels five through nine. When a power pill is eaten, Pakman has the power to eat all of the monsters he pleases. Whenever Pakman eats a monster, his ghost appears at the top of the screen. When seven ghosts have appeared at the top of the screen one of two things will happen. Either one of the ghosts will come out and float around the screen (even through the walls) hunting for Pakman, or the seven ghosts will link together to form a centipede that will do the same thing. Even with power pills, Pakman cannot overpower ghosts or centipedes.

Bonus prizes appear in the middle of the screen about twice a board. When bonus prizes are eaten they are stored in a box below the screen. When 14 prizes have been eaten you get a bonus of 14,000 points. This can only happen twice.

Four more tricks the programmer threw in to make the program better are invisible mazes every four rounds, the ability to store power pills, a selection of difficulty at the beginning of the game, and a high scores board.

You can store power pills by eating a pill while a previous pill is in effect. Stored power pills can be used by pushing your button while no power pill is in effect. A maximum of six power pills can be stored at any time.

I liked *Pak-Panic* and I think many other people will like it.

(Tom Mix Software, 4285 Bradford NE, Grand Rapids, MI 49506, tape $24.95, disk $27.95)

— Pat Downard

We mean business...

Last issue, we began a series of articles detailing the system programs used here.

The following two programs work in concert with last month's program "ACS3".

"SURNM3" examines each name on the file disk (set up by ACS3 on drives 0 and 2) and writes segmented surname master files to drive 1 which can be then utilized by our second program this month, GT3, to find names and numbers quickly.

"SURNM3" suffers from the problem we discussed briefly last month, in that it will do one file, and the first half of the next one, but hangs up when it tries to switch to drive 2 the next time. The program is too useful to let this worry us, so we just cold start it before we start a new file.

A scan of the listing shows that "SURNM3" looks for spaces in the Name file, assuming everything left of the space to be christian name, and everything right of the space to be surname.

The surname only is stored, along with its record number for future use by GT3.

GT3 when run, asks one question : SURNAME?

Any name that begins with the input letters will be displayed, so it is possible to find names that you might not otherwise find,due to poor handwriting or whatever.

For example, if I input MOR, GT3 loads the M file and extracts all the names starting with MOR. If I am looking for MORPHETT, then I will also see the 2.5 million MORRIS's that we have on data base, but eventually, I will also find MORPHETT.

If I press (ENTER) instead of inputing a name, then GT3 RUNS ACS3.

SURNM3 in particular, is a clumsy program, which takes a lot of time to execute, and therefore we tend to make up new files from it only once a month. GT3 is better, and works fine. It could be faster - we could for example, have a master file for each letter of the alphabet, but then that would be a real drag when it came to getting those files updated by SURNM3!

So we compromise, and accept what we have. Perhaps now you can see why we like to see your subscription number on everything you write to us! If your number is supplied we can go straight into ACS3, otherwise we have to find you first on GT3.

What are the alternatives? Well most of the alternatives revolve around upgrading to 80 tracks, or going to a program written in BASIC09, or to a ready made data base such as RMS, (in OS9).

Greg's solution was to transfer to another, larger computer, and he was actually going through that process in June.

We felt that by going to the double sided double drives, we could stave off that day, and gain experience in complex programing and data handling within CoCo. (We certainly achieved that!)

The other way is to work with a prepared program like RMS, but I am not a fan of prepared data bases. It is my view that the data base you write yourself, whilst nearly always being a trade off in some department, usually does the job more fully, and more quickly, than the other programs.

Next month we will show the postcode sort program as it has become. It is a prime example of the reasons why it is good to be able to retain control of your data and handle it according to your own whim.

Listing 1:

```
1 '***GT3***12/1/85**NAME SEARCH
   *BY GREG WILSON & G. MORPHETT
   ******* COPYRIGHT ***********
2 CLS0:GOTO10
3 SAVE"GT3/BAS:3":DIR3:PRINTFREE
(3)
4 I$=INKEY$:IFI$=""THEN4
10 CLEAR15500:DIM A(900),A$(900)
20 INPUT"SURNAME";X$:CLS0:GOTO23
0
30 IA$=IB$:PRINTX$:X=LEN(X$):Z=A
SC(X$):IFZ=47THEN60
40 FORJ=1TOI:IFZ=ASC(A$(J))THENG
OTO70
50 NEXTJ:GOTO20
60 CLOSE:END
70 IFX$<>LEFT$(A$(J),X)THEN50 EL
SE B=A(J):GOTO110
80 GOTO50
90 GET#1,A(J):PRINTB;" ";E$;:PRI
NTTAB(6)N$:PRINTTAB(6)S$;:PRINTT
AB(6)T$;:GOSUB150:PRINTST$;" ";P
$:PRINT"B/CARD:";
100 PRINTLEFT$(Z$,3);"-";MID$(Z$
,4,2);"-";MID$(Z$,6,3);"-";RIGHT
$(Z$,6):PRINTSTRING$(32,223);:CL
OSE:GOTO50
105 STOP
110 IFA(J)<=1450 THEN OPEN"D",#1
```

continued on page 16

CFM

COMPUTERWARE FOR MICROS.

bankcard welcome here

VISA

P.O. Box 81
Brooklyn Park . S.A 5032
Phone (08) 336 6588

LET'S GET DOWN TO BUSINESS

PRO-COLOR FILE *Enhanced* 2.0 $79.95

An all new version of **PRO-COLOR-FILE** will once again leave its mark as the most flexible database in its price range for the Color Computer.

• 60 Data Fields • 1020 BYTE RECORDS • TRUE MULTI DRIVE SUPPORT • 4000+ RECORD CAPACITY • 4 USER DEFINED DATA ENTRY SCREENS • 28 MATH EQUATIONS • IF-THEN-ELSE FUNCTIONS IN EQUATIONS • FILE-WIDE RECALCULATION • 8 USER DEFINED REPORT FORMATS • 6 USER DEFINED LABEL FORMATS • TOTAL FIELDS ON REPORTS • SUMMARIZE FIELDS • SEND REPORTS TO PRINTER, SCREEN OR TEXT FILE • FAST ML SORT (750 RECORDS IN LESS THAN 5 MINUTES) • CREATE UP TO 16 INDEXES FOR SORTING OR REPORTING RECORDS • AUTO KEY REPEAT • KEYBOARD CLICK • STORES FORMATS FOR REPEATED USE • CUSTOM SELECTION MENUS • PASSWORD PROTECTION • CREATES FILES COMPATIBLE WITH DYNACALC® •

Because of **PRO-COLOR-FILES** ability to send reports to a text file, this means you can use your favorite communications program to transmit reports to other computers or read them in with your favorite word processor for creating customized reports. You can also convert ASCII files from your favorite spread sheet program into data files that can be accessed for further reporting and analyzing. **PRO-COLOR-FILE** is also supported by the **PRO-COLOR-FILE** National Users Group with quarterly newsletters. Join the rest of the world and discover for yourself what you've been missing.

PRO-COLOR-BUSI $199.95

As most Businessmen do not want to learn how to run a Database and format it, we have Established a small business package. This package has, RETAIL INVOICE, WHOLESALE INVOICE, ACCOUNTS PAYABLE, REPORTS, LABLE, Plus a supplementary set of instructions to walk a newcomer through. The package includes the full **PRO-COLOR-SERIES** as well, so that, when the need arises, you can have someone build your own special Database. We will be establishing further up and running Databases as the need arises.

DYNACALC® $129.95

This is the CoCo version of **VISICALC**. **DYNACALC** is not a toy. Sort up to eleven times faster than other top line spreadsheets. If you are really serious about using your CoCo for Business then there is no other spreadsheet to consider.

DYNACALC is designed to run with **PRO-COLOUR-SERIES**.

TELEWRITER-64 DISK — $75.00 CASSETTE — $66.00

COMPUTERWARE researched the word processors available for the Color Computer. This is the best. **Telewriter-64** is a truly sophisticated system that is marvelously easy to use. It works with any 16K, 32K or 64K system and any CoCo compatible printer.

TOP-RATED COCO WORD PROCESSOR

MASTER DESIGN $39.95

This graphics program does more for you than just hi-res graphic editing. It will generate lettering in hi-res graphics that can be different sizes, skinny, bold, textured, drop shadowed, raise shadowed or tall. It will also interface with the Telewriter-64 word processor for printing hi-res displays with your letters.

As a graphics editor, it takes full advantage of all the extended BASIC hi-res graphic commands. Create boxes, circles, lines, copy displays and utilize GET and PUT features. Some added commands include mirror reflection, turn displays backwards or upside down, Squish displays, create dot patterns for shading or diagonal lines for creative backgrounds.

Special text files created with the Letter Head Utility allow you to access hi-res graphics from Telewriter-64, your own BASIC programs or **PRO-COLOR-FORMS**.

MASTER DESIGN comes with its own screen dump routine which interfaces with all popular dot matrix printers that have dot addressable graphic ability.

See reviews in:
July '84 *Rainbow*, Oct. '84 *Hot CoCo* Telewriter-64 © 1983 by Cognitec

PRO-COLOR-FORMS 2.0 $39.95

PRO-COLOR-FORMS will access data files you create with **PRO-COLOR-FILE** and merge them with a letter or place them on pre-printed forms such as statements. Any field of information from your data file can be placed anywhere and repeated as many times in the letter or on the form. You can use the built in ML text editor for creating the form or use your favorite word processor.

• DESIGN UP TO 6 FORMATS AT ONE TIME • USER DEFINED PAGE SIZE • SUPPORTS SPECIAL PRINTER CONTROL CODES • RIGHT JUSTIFICATION • PASSWORD PROTECTION • MERGES WITH GRAPHICS FROM MASTER DESIGN •

If you use our graphics program **MASTER DESIGN**, you can merge graphics with your forms for added enhancements. Have your graphic letter head printed at the top of each letter or incorporate designs, bar graphs or any display created within the form itself.

PRO-COLOR-DIR $24.95

Need to organize all your diskettes so you know where each program is? **PRO-COLOR-DIR** will read your directories and create a master data file that can be accessed by **PRO-COLOR-FILE** for sorting and reporting. 1000 + records can be stored on one diskette with valuable information about each program.

• DISK ID NAME • FILENAME/EXT • TYPE OF FILE • DATE CREATED • DATE UPDATED • NUMBER OF GRANS ALLOCATED • NUMBER OF SECTORS ALLOCATED AND USED • MACHINE LANGUAGE ADDRESSES •

A diskettes directory can be re-stored in the data file with old entries deleted and new ones appended automatically. You can obtain hard copies of the information and create labels of the filenames for placing on the diskette itself.

CFM

COMPUTERWARE FOR MICROS.

bankcard welcome here

VISA

P.O. Box 81
Brooklyn Park, S.A 5032

Phone (08) 336 6588

BACK IT UP! $59.95
(Supplied on Disk)

SUPER BACK-UP UTILITY©

SUPER BACK-UP UTILITY WILL PERFORM ALL OF THE FOLLOWING FUNCTIONS

1. **TAPE TO TAPE** (Regardless of most protection schemes)
2. **TAPE TO DISK** (Move Cassette programs to Disk)
3. **AUTO RELOCATE** (For those Cassette programs that conflict WITH Disk operating systems)
4. **DISK TO TAPE** (Place Disk programs onto Cassette)
5. **DISK TO DISK** (Our powerful Spit-N-Image Program. Regardless of protection schemes)

- **MENU DRIVEN!**
- **REQUIRES 32K EXTENDED COCO**
- **REQUIRES 1 OR 2 DRIVES** (For Disk Functions)
- **ALL MACHINE LANGUAGE!!!**

COMPARE WITH OTHER INDIVIDUAL PROGRAMS COSTING IN EXCESS OF $100.00 OR MORE!!!

COLORPACK $39.95

ROM/RAM pack. (specify configuration)

COLOR QUAVER $29.95

Software Music Synthesizer on tape (requires 32/64K).

COLOR BURNER $99.95

EPROM Programmer (2716/32/32A/64/64A/128, 68764/66) with software.

LOWERKIT III $89.95

- Full-time upper and lowercase installs in 15 minutes.
- Normal and reverse video standard.
- Fully compatible with all Alpha and Graphic modes.
- Assembled and tested.

Important! Specify Color Computer or Color Computer II.

DOUBLE DOS II $44.95
(DISK ONLY) 64K required

Double DOS II—Now use **35, 40,** or **80** track (double or single sided) drives, all on one system, all at the same time. All regular disk commands are supported with Double DOS II and are totally transparent to your BASIC programs! You can get up to **158 granules** on a disk using an 80 track drive. These are the added commands:

BAUD 1-6 ... change the BAUD rate.
TRACK 35, 36, 40, 80 ... change number of tracks.
DOUBLE ... enable the double sided option.
PDIR ... print your directory to printer.
DUMP ON/OFF ... send programs without a terminal program.
RATE 6,35 ... change the head stepping rate.
VIDEO ON/OFF ... reverse video without a hardware mod.
SCROLL 1-255 ... change your screen scrolling speed.
COMMAND ... will list all new commands.
DUP 0, 1, 2 ... will allow copy & backup from one side of a drive to another!
DATE ... you can enter the month, day and year as an extension to your programs when they are displayed during a DIR command.

We guarantee that this program will work using the above commands, with all types of 35, 40 or 80 track drives!

DRIVE O PACKAGES $549
$7 shpg

More storage, Less cost!

... Our single-sided disk package gives 23,040 bytes more for a dollar less!

179, 712 Bytes: DD-1 SSDD Drive
DC-1 40 Track Controller
CA-1 Cable

GRAPHICOM PART II Introductory pack $39.95

Graphicom Part II is a video processing package that provides many functions that are missing in Graphicom. Here are just a few of the features provided by Graphicom Part II:

ENLARGE/REDUCE/ROTATE
Enlarge or reduce any portion of a screen by any amount, just like a photographic enlarger! Independent of the enlargement or reduction, rotate by any degree or fraction of a degree about any point on the screen.

PAN & ZOOM
"Zoom in" x2, x4, or x8 on any portion of the screen to do fine pixel work. Allows editing of Graphicom character sets with ease!

TYPESETTER & FONT EDITOR
Add text in 16 different sizes with several display modes to choose from including COLORED FOREGROUND & BACKGROUND text! Edit 8x8 characters for use in the typesetter. Over 30 character sets supplied on disk. "GRAB" function allows transfer of some Graphicom character sets to Graphicom Part II format.

PIXEL BLASTER
Allows the user to easily substitute or remove colors. Widen lines, swap BLUE & RED without effecting BLACK & WHITE, etc.

GRAPHICOM PART II DOES NOT REQUIRE GRAPHICOM TO RUN!
Graphicom Part II requires a 64K extended disk basic system, it will load and save both standard BIN files and Graphicom screens, and supports 1 to 4 disk drives with keyboard or joystick (analog or switch type). All functions support color or Hi-Res operation, as well as 4 screen display modes.

AUTOTERM

TURNS YOUR COLOR COMPUTER INTO THE

WORLD'S SMARTEST TERMINAL!

YOU'LL ALSO USE AUTOTERM FOR SIMPLE WORD PROCESSING & RECORD KEEPING

EASY TO USE
ON-SCREEN EDITING via cursor. Full prompting and error checking. Key Beep and Error Beebop. Scroll bkwd/fwd while on line. Save/load files while on line. Maintain a disk copy of session. Automatic graphics. True lower case. Screen widths of 32, 40, 42, 50, 64. No split words on screen/printer. Print all or part of text. Search for strings. Well written manual goes step-by-step and has many KSM examples. Back cover is a cheat sheet.

RECOMMEND 32K to 64K

PLEASANTLY POWERFUL
Total communications ability, 128 ASCII chars. 1200 baud, etc. Send text, graphics, BASIS, ML Scan/Edit current data while receiving more data. Any modem. Fully supports D.C. Hayes and others. any printer, page size, margins, etc. Override narrow text width of received data. Examine/change parameters, KSMs and disk directories at any time. Handles files which are larger than memory.

TRULY AUTOMATIC
Create, edit, print, save and load Keystroke Multipliers (KSMs). KSMs automate almost any activity. dial via modem, sign-on, interact, sign-off. Perform entire session. Act as a message taker. KSM may include parameter changes, disk operations, editing, time delays, looping, execution of other KSMs, waiting for part-specified responses, branching based upon responses.

CASSETTE—$69.95 **DISKETTE—$69.95**

Nrth Qld Colour Software, 9 Durham Court, KIRWAN, TOWNSVILLE, QLD. Phone (077) 73-2064.
Rainbow Valley Comuters, RMB 6680, MAFFRA, VIC. Phone (051) 454-315
Blaxland Computer Service, P.O. BOX 2774, BLAXLAND, N.S.W. Phone (047) 39-3903
Paris Radio Electronics, 161 Bunnerong Road, KINGSFORD, N.S.W. Phone (02) 344-9111.
Geoff Tolputt, P.O. Box 140, WOOLOONGABBA, QLD. Phone (07) 446084.
Crystal Blade Software, P.O. Box 256, ROSEVILLE, N.S.W. Phone (02) 467-1619
The Computer Hut, 21 William Street, BOWEN, QLD. Phone (077) 86-2220.

COMPUTERWARE FOR MICROS
Peter and Jillian Collison
11 Grantley Avenue,
Rostrevor S.A. 5073
PHONE: (08) 336 6588

continued from page 13

```
,"ACS2/DAT",103:D=0:GOTO130
120 OPEN"D",#1,"ACS2/DAT:2",103:
A(J)=A(J)-1450:D=2
130 FIELD#1,25AS E$,17AS N$,26AS
 S$,17AS T$,4AS P$,14AS Z$
140 GOTO90
150 W=VAL(LEFT$(P$,1)):IFW=2THEN
ST$="NSW"
160 IFW=3THENST$="VIC"
170 IFW=4THENST$="QLD"
180 IFW=5THENST$="SA"
190 IFW=6THENST$="WA"
200 IFW=7THENST$="TAS"
210 IFW=9THENST$="STOPPED"
220 RETURN
230 IFX$=""THEN RUN"ACS3:3"
240 T$=LEFT$(X$,1):IFT$<"G" THEN
 IB$="A":IFIB$=IA$THEN 30 ELSE 2
90
250 IFT$>"F" AND T$=<"J" THEN IB
$="F":IFIB$=IA$THEN 30 ELSE 320
260 IFT$=>"K" AND T$<="N" THENIB
$="K":IFIB$=IA$THEN 30 ELSE 300
270 IFT$=>"O" AND T$<="V" THENIB
$="O":IFIB$=IA$ THEN 30 ELSE310
280 IB$="W":IFIB$=IA$THEN30 ELSE
 OPEN"I",#2,"SURNW/DAT:1":GOTO33
0
290 OPEN"I",#2,"SURNA/DAT:1":GOT
0330
300 OPEN"I",#2,"SURNK/DAT:1":GOT
0330
310 OPEN"I",#2,"SURNO/DAT:1":GOT
0330
320 OPEN"I",#2,"SURNF/DAT:1"
330 FORI=1TO900:INPUT#2,A(I),A$(
I):IFEOF(2)=-1THEN340 ELSE NEXTI
340 CLOSE:GOTO30
```

Listing 2:

```
1 '***SURNM3***UPDATES SURNAME F
ILES***6/12/84---LATEST UP DATE
OF FILES 12/1/85 ***************
**BY GREG WILSON & G. MORPHETT**
2 CLEAR1500:DIM A$(700),A(700):G
OTO20
3 SAVE"SURNM3/BAS:3":DIR3:PRINTF
REE(3)
4 I$=INKEY$:IFI$=""THEN4
20 T=5
30 GOSUB1500
40 ON T GOSUB 1000,1100,1150,120
0,1400
50 FORI=1TO1450
60 GET#1,I:J=17:K=2:NA$=N$
63 IFLEFT$(N$,1)=CHR$(255)THENCL
OSE:GOTO200
70 IF LEFT$(N$,1)=" "THENNA$="":
GOTO130
80 IF RIGHT$(NA$,1)=CHR$(32)THEN
J=J-1:NA$=LEFT$(NA$,J):GOTO80
90 IFMID$(NA$,K,1)=CHR$(32)THENN
A$=RIGHT$(NA$,J-K):GOTO110
100 K=K+1:GOTO90
110 IFNA$=""THEN140 ELSE Z=Z+1
115 B$=LEFT$(NA$,1):ON T GOTO 16
00,1700,1750,1800,1820
120 L(T)=L(T)+1:PRINTL(T);T;Z;NA
$
125 WRITE#2,Z,NA$
140 NEXTI:PRINTZ:SOUND20,5
150 CLOSE#1:GOSUB1510:GOTO50
200 CLOSE:Z=0:NEXTT
210 FORT=1TO3
220 PRINTL(T)
230 NEXTT:END
1000 OPEN"O",#2,"SURNA:1":RETURN
1100 OPEN"O",#2,"SURNF:1":RETURN
1150 OPEN"O",#2,"SURNK:1":RETURN
1200 OPEN"O",#2,"SURNO:1":RETURN
1400 OPEN"O",#2,"SURNW:1":RETURN
1500 OPEN"D",#1,"ACS2/DAT",103:G
OTO1520
1510 OPEN"D",#1,"ACS2/DAT:2",103
1520 FIELD#1,25AS E$,17AS N$,26A
S S$,17AS T$,4AS P$,14AS Z$
1530 RETURN
1600 IFB$>"F" THEN GOTO 140
1610 GOTO120
1700 IFB$<"F" OR B$>"J" THEN GOT
O140
1710 GOTO120
1750 IFB$<"K" OR B$>"N" THEN GOT
O140
1760 GOTO120
1800 IFB$<"O" OR B$>"V"THEN GOTO
140
1810 GOTO120
1820 IFB$<"W" THEN 140
1830 GOTO120
```

from p 3

these chips possible.

5. The slip of the Aussie dollar against the US$ places us in the unenviable position of having to raise the price of this magazine again if the Aussie dollar doesn't start to behave. I am going to hold this month and see what happens, but will have to go with it if things don't improve this month. Expect Tandy prices, and other imported goods to be effected simularly.

Despite Aussie dollars, power strikes, and lots of fine weather to woo us away from our new desks, the magazine is as crammed as last time! Thanks for the nice comments you've been making - we were pleased with February too!

[illegible]

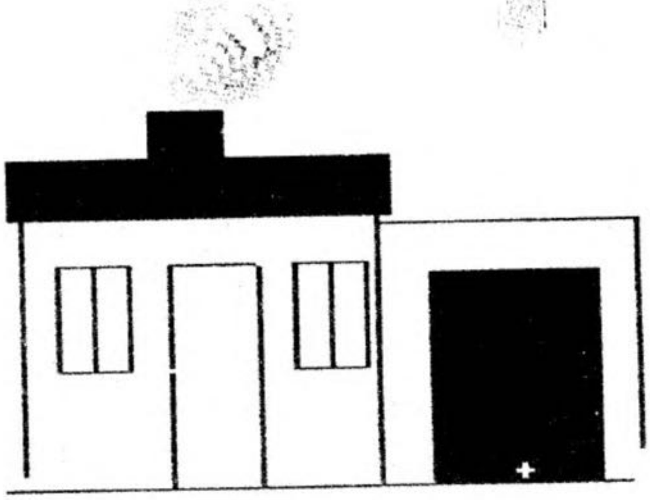
Home Sweet Home

By Marlene Fearing

This program draws a house, a sun and a garage. It opens and closes the garage door, the front door opens and a figure appears and waves. Afterward, the door closes, the grass grows, and smoke comes from the chimney.

This is the first computer program I wrote after getting my computer. I hope it will encourage others to experiment with graphics and animation; it was a lot of fun to create. This program will work with 16K Extended BASIC with tape, or with a disk drive system. Just type it in and watch it draw.

(Marlene Fearing is a student at Pima Community College in Tucson, Ariz., where she is studying for her A.A.S. as a small business computer specialist.)



25	54
52	1
90	230
END	241

The listing:

```
1 '
2 '******************************
3 '*                            *
4 '*     EXECUTIVE HOUSE        *
5 '*     MARLENE FEARING        *
6 '*     812 S. PLUMER          *
7 '*     TUCSON, ARIZ. 85719    *
8 '*                            *
9 '******************************
1Ø PMODE 3,1
11 PCLS (3)
12 SCREEN 1,1
13 '    DRAW MAIN HOUSE
14 LINE (32,18Ø)-(152,88),PSET,B
15 LINE (28,68)-(156,88),PSET,BF
16 PAINT (32,72),2,4
17 LINE (152,91)-(24Ø,18Ø),PSET,
B
18 LINE (17Ø,1Ø8)-(226,18Ø),PSET
,B
19 '          DRAW THE SUN
2Ø CIRCLE (2Ø4,22),1Ø,2
21 LINE (44,1Ø4)-(68,14Ø),PSET,B
22 LINE (124,1Ø4)-(148,14Ø),PSET
,B
23 LINE (82,1Ø4)-(112,18Ø),PSET,
B
24 LINE (76,52)-(1ØØ,68),PSET,BF
25 CIRCLE (84,14Ø),2,2
26 PAINT (44,16Ø),1,4
27 PAINT (169,176),1,4
28 CIRCLE (2ØØ,176),3,2
29 PAINT (2Ø4,22),1,2
3Ø PAINT(56,12Ø),2,4:PAINT (133,
12Ø),2,4
31 LINE (56,1Ø4)-(56,14Ø),PSET
32 LINE (136,1Ø4)-(136,14Ø),PSET
33 '        TO OPEN AND CLOSE GAR
AGE DOOR
34 FOR X=1 TO 5ØØ:NEXT X
35 PAINT (19Ø,179),2,4
36 FOR X=1 TO 15ØØ:NEXT X
37 PAINT (176,11Ø),4,4
38 CIRCLE (2ØØ,176),3,2
39 LINE (Ø,18Ø)-(255,191),PSET,B
4Ø '       TO OPEN AND CLOSE FRONT
 DOOR AND FIGURE TO WAVE AND GO
BACK INSIDE
41 LINE (92,112)-(92,19Ø),PSET
42 LINE (92,19Ø)-(112,18Ø),PSET
43 PAINT (185,19Ø),1,4
44 PAINT (1Ø,185),2,4
45 LINE (92,112)-(112,1Ø4),PRESE
```

```
T
46 PAINT (185,19Ø),2,4
47 LINE (92,112)-(92,19Ø),PRESET
48 LINE (92,19Ø)-(112,18Ø),PRESE
T
49 LINE (Ø,255)-(255,18Ø),PSET,B
5Ø FOR X=1 TO 12Ø:NEXT X
51 LINE (1Ø2,112)-(112,1Ø4),PSET
52 LINE (1Ø2,112)-(1Ø2,19Ø),PSET
53 LINE (1Ø2,19Ø)-(112,18Ø),PSET
54 PAINT (96,124),1,4
55 CIRCLE (96,124),7,Ø
56 LINE (96,13Ø)-(96,164),PSET
57 LINE (96,164)-(84,179),PSET
58 LINE (96,14Ø)-(84,14Ø),PSET
59 LINE (96,164)-(1Ø2,179),PSET
6Ø LINE (96,14Ø)-(1Ø2,14Ø),PSET
61 LINE (86,14Ø)-(86,13Ø),PSET
62 FOR X=1 TO 3ØØ:NEXT X
63 LINE (86,14Ø)-(86,12Ø),PRESET
64 FOR X=1 TO 5ØØ:NEXT X
65 LINE (86,14Ø)-(86,13Ø),PSET
66 FOR X=1 TO 15Ø:NEXT X
67 LINE (86,14Ø)-(86,13Ø),PRESET
68 FOR X=1 TO 15Ø:NEXT X
69 LINE (86,14Ø)-(86,13Ø),PSET
7Ø CIRCLE (96,124),7,1
71 LINE (96,13Ø)-(96,164),PRESET
72 LINE (96,164)-(84,179),PRESET
73 LINE(96,14Ø)-(84,14Ø),PRESET
74 LINE (Ø,18Ø)-(255,18Ø),PSET
75 LINE (96,164)-(1Ø2,179),PRESE
T
76 LINE (96,14Ø)-(1Ø2,14Ø),PRESE
T
77 LINE (86,14Ø)-(86,13Ø),PRESET
78 LINE (1Ø2,112)-(112,1Ø4),PRES
ET
79 LINE (1Ø2,112)-(1Ø2,19Ø),PRES
ET
8Ø LINE (1Ø2,19Ø)-(112,18Ø),PRES
ET
81 LINE (Ø,18Ø)-(255,191),PSET,B
82 CIRCLE (86,14Ø),3,3
83 COLOR 2,1
84 PAINT (3Ø,188),2,4
85 PAINT (232,188),2,4
86 '
87 '     TO DRAW GRASS
88 '
89 POKE 65495,Ø
9Ø DRAW "BMØ,18Ø;R1;U8;R2;D8;R2;
U1Ø;R2;D1Ø;R3;U12;R2;D12;R3;U5;R
2;D5;R3;U5;R2;D5;R2;U3;R2;D2;R2;
U4;R2;D4;R2;U3;R2;D3"
91 DRAW "BM238,18Ø;U1Ø;R2;D1Ø;R3
;U8;R2;D8;R4;U6;R2;D6;R2;U8;R2;D
8;R1"
92 '
93 'SMOKE STARTS HERE
94 '
95 X=82:Y=52: 'CIRCLE CENTERPOIN
T
96 SP=Ø:EP=Ø  'CIRCLE RADIUS
97 FOR R=1 TO 35 STEP .Ø5  'CIRC
LE RADIUS
98 EP=EP+.Ø2: IF EP>1 THEN EP=Ø
99 CIRCLE (X+R,Y-R),R,1,1,SP,EP
1ØØ NEXT R
1Ø1 '
1Ø2 '       TO TURN BACKGROUND TO
 NIGHT
1Ø3 '
1Ø4 PMODE 4,1
1Ø5 SCREEN 1,Ø
1Ø6 CIRCLE (2Ø4,22),1Ø,5
1Ø7 PAINT (2Ø8,22),5.5
1Ø8 '         REDRAWN SMOKE STARTS
 HERE
1Ø9 X=82:Y=52:   'CIRCLE CENTERPO
INT
11Ø SP=Ø:EP=Ø:   'CIRCLE RADUIS
111 FOR R=1 TO 35 STEP.Ø5 'CIRCL
E RADUIS
112 EP=EP+.Ø2:IF EP>1 THEN EP=Ø
113 CIRCLE (X+R,Y-R),R,1,1,SP,EP
114 NEXT R
115 POKE 65494,Ø
116 GOTO 1Ø
117 END
```

Hint . . .

A common practice in programming is to use a REM to head a subroutine or *GOTO* line. This helps make programs easier to read and follow. However, the REM/title should never be the line referenced by the *GOTO* or *GOSUB*. If you start compacting a program by stripping REMs, you'll have nowhere to *GOTO!* Instead of:

```
10 GOSUB 4000
 .
 .
4000 REM SUBROUTINE TO INCREMENT
SCORE
```

put the REM one line number back:

```
10 GOSUB 4000
 .
 .
 .
3999 REM INCREMENT SCORE
4000 IF K> . . . . .
```

With this format, removing the REM will leave the program untouched.

T. Gray
Sunnybrook, Alberta

A Simple Text Processor

By Ashok Basargekar

One of my favorite hobbies is to improve the Color Computer software written by others in my favorite RAINBOW magazine, give it a personal touch and enjoy the results. I remember Mr. Lewandowski's series of articles on the simple text handling program. I used to read the articles, enhance them to my satisfaction and wait for his next installment. After waiting for several months for him to give me some hints on the *EDIT* feature of his text handling program, I decided to take on this task myself.

Before going into the *EDIT* feature, I would like to present a complete face lift that I have given to the other subroutines of the text handler.

The first six lines of my assembly language source code define the ROM routines I will be using. The next 14 lines are the direct page addresses that I will be using to store my constants and variables. I may use a portion of the direct page; that's what the *Getting Started with Color BASIC* manual says! The START of my program uses the auto key repeat feature, published by Roger Schrag in his article on "Super Patched EDTASM". At START1 I release the alpha lock so I start my text processor with lowercase letters. In WIPE, I clear all the text buffer and then branch to FIN1 for my new menu. I beg your pardon, Mr. Lewandowski, I have used my name instead of yours, in the MES1. Instead of using LINPUT routine for text handling, I have made it character-oriented in CONT for continue. I thought that the original PAPER routine was very primitive, so I changed it to give me the top of the form, left margin, line width and line spacing selections. First I take the characters up to the line width and go back to the nearest place where I can break a word before going to the next line. The *CLOAD, CSAVE, LOAD* and *SAVE* routines are the gifts of Roger Schrag from his disk and tape I/O routines. Before I go to the LINPUT for filename, I lock the alpha lock, so that the filename is always in capital letters. The EXIT routine also does the same thing. Finally I come to my *EDIT* routine for some comments.

Here I have used the same memory locations that I used to store the constants of PAPER routine in the direct page. SCL is used to store the text buffer address that will equate to the top left corner of the video screen. MARGIN stores the text buffer address that equates to the bottom right corner of the video screen. These addresses are revised as soon as the Y register (cursor pointer) goes beyond $400-$5FF range. Before bringing the next portion of the text for editing, all the previous buffer area is revised to match the screen buffer. The COPY routine brings a copy of a portion of text in video screen for editing and the REVISE routine sends the edited text from screen to the text buffer. The NXTPGE and PRVPGE routines change the SCL and MARGIN addresses of next page or previous page depending upon the cursor movement. The DELETE routine moves all the text one to the left when the CLEAR key is pressed. The INSERT routine moves all the text one to the right for making room for a character in the middle.

I have used Spectral Associates' *ULTRA 80C* for editing and assembling this program. Of course, you may use any other assembler you wish. Since I have installed the *Lower-Kit*, by Green Mountain Micro, in my CoCo, the entire text is very beautiful on the screen.

The entire machine language code resides from $E00 through $16D4 and for a 32K computer, you will have plenty of text buffer area from $16D5 through $7FFF. The program is completely position independent except the address table for the menu subroutines. The control keys and procedure in using my *Text Processor* are as follows:

Initialization

LOADM"TEXT PRO" and *EXEC* will access this program. You will get a complete menu of selection as follows:

1) COMPOSE

The Compose mode allows you to compose a new text, or to append a typed or loaded text from a tape or disk. Words will not wrap around to the next line while typing, but they will be properly moved to the next line at the time of printing on a paper. Any immediate mistakes can be corrected by moving the cursor backward, with the left arrow key. Once you exit this Composing mode, and return back for continuing the text, you will not be able to correct the previously typed text with the left arrow key. You will need to go to the Edit mode for this purpose. While composing the text, do not press the ENTER key unless you want to go to the next line for a new paragraph. Pressing ENTER will provide a hard carriage return when printing the text on a printer. To exit the Composing mode, simply hit the BREAK key. You will return back to the main menu of selections.

2) EDIT

The text in the Edit mode appears slightly different from that in the Composing mode. You will see a red block at the places you have pressed the ENTER key, for providing a hard carriage return for a new paragraph. The up, down, right and left arrow keys will move the cursor anywhere in the text, while in the Edit mode. The CLEAR key will delete one character at a time. The SHIFT-CLEAR keys will allow you to insert any text in the middle. The flashing cursor will disappear when you are in the Insert mode. You will return back to the Edit mode by pressing the BREAK key. You will exit the Edit mode by pressing the BREAK key again. The text can also be appended at the end while you are in the Insert mode. To revise the text in the Edit mode, simply write new text over the existing text.

3) CLOAD

This selection will allow you to load a text from a cassette tape. The text can be loaded at the end of any typed or other-loaded text, allowing you to merge two or more texts.

4) DLOAD

This selection will allow you to load any text from a disk. You will be asked to enter a filename. The filename must be the entire name including the extension. If the filename is not found, or if the file is on a bad disk, you will receive an error message number. If so, simply press any key to go back to the main menu. Refer to Table 1 for the type of error.

5) PRINT

The underlining codes are presently set for the Brother Correctronic 50 typewriter. The Baud rate is set at 1200. Simply enter the desired printing specifications for total line width, left margin and line spacing. Your text will be printed on the paper according to your specifications. The paper will advance to the new page after printing 60 lines. Therefore, adjust the paper so that three blank lines are left at the top. This will provide three blank lines at the bottom. To change the printer Baud rate and printable lines per page or to change the underlining codes, you will need the following corrections to the software before executing the program.

POKE&HF74, msb : POKE&HF75, lsb of Baud rate constants.

POKE &H100D, n where n = printable lines per page.

POKE &H1016, m where m = blank lines at top and bottom of page.

POKE&H102C, 27 : POKE&H1031, 45 for start of underlining codes for Brother.

POKE&H1037, 27 : POKE&H103C, 82 for end of underlining codes for Brother.

POKE&H102C, 32 : POKE&H1031, 15 for start of underlining codes of LP VIII

POKE&H1037, 14 : POKE&H103C, 32 for end of underlining codes of LP VIII

6) CSAVE

This routine will allow you to save the text on a cassette tape.

7) DSAVE

This subroutine will allow you to save the text on a disk. You will be asked for a filename. It must be up to eight characters in length with an extension up to three characters. If an extension is not specified, none will be assumed. Therefore, give a filename like: *TEXT/DAT* or *TEXT.TXT*, etc.

The codes for the error messages while reading or writing text from or to the disk are as follows:

TABLE 1

CODE	TYPE OF ERROR
19	File already open
20	Bad device or drive number
21	I/O error
22	FM error
23	File not open
24	Input past end of line
27	File not found
29	Disk full
30	Out of buffer space
31	Disk write protected
32	Bad filename
33	Bad file structure
37	Verification error

8) EXIT

This will exit to BASIC. You will lose all the text with this selection. Therefore, make sure that the text is saved on the tape or disk prior to selecting EXIT.

Happy text processing! If you have any questions or suggestions regarding my text processor please drop a line with a SASE to Ashok Basargekar, 1423 North Cleveland Street, Orange, CA 92667, (714) 639-3996.

The listing:

```
                 *********************************
                 * A SIMPLE TEXT PROCESSOR       *
                 * BY ASHOK BASARGEKAR.          *
                 * 1423 NORTH CLEVELAND STREET,  *
                 * ORANGE, CA. 92667.            *
                 *********************************
                 *
                 * MAJOR ROM ROUTINES USED BY THIS PROGRAM.
                 *
     A928        CLS      EQU $A928 Clear screen.
     A30A        SCREEN   EQU $A30A Print on screen.
     A393        LINPUT   EQU $A393 Line input.
     A2BF        PRNTR    EQU $A2BF Print on printer.
     A1C1        INKYS    EQU $A1C1 INKEY$
     A027        QUIT     EQU $A027 Back to Basic.
     A7D3        DELAY    EQU $A7D3 Delay until X=0
                 *
                 * Constants & variables stored in Direct Page.
                 *
     0000        KCLEAR   EQU $0 Auto key repeat
     0001        KHOLD    EQU $1 constants.
     0002        BUFST    EQU $2 Start of text buffer addtess.
     0004        BUFEN    EQU $4 End of text buffer address.
     0006        SCL      EQU $6 Start of current line.
     0008        MARGIN   EQU $8 Left margin.
     0009        LW       EQU $9 Line width.
     000A        CLW      EQU $0A Current line width.
     000B        SPACE    EQU $0B Line spacing.
     000C        LCP      EQU $0C Line counter of page.
     000D        LENGTH   EQU $0D Length of filename.
     000E        DSAVE    EQU $0E Tape/disk error vector.
     0011        STACK    EQU $11 Tape/disk stack pointer.
                 *
0E00                      ORG $E00
                 *
                 * Following interrupt service routine is similar to one
                 * in Rainbow Sept 83, page 77
                 *
0E00 30   8C 6B  START    LEAX <NMI.PCR
0E03 BF   010A            STX $10A
0E06 30   8C 13           LEAX <IRQ,PCR
0E09 BF   010D            STX $10D
0E0C 20   62              BRA START1
0E0E B6   0982   NMI      LDA $982
0E11 27   5C              BEQ REPOUT
0E13 BE   0983            LDX $983
0E16 AF   6A              STX $0A,S
0E18 7F   0982            CLR $982
0E1B 3B                   RTI
0E1C B6   FF03   IRQ      LDA $FF03
0E1F 2A   4E              BPL REPOUT
0E21 B6   FF02            LDA $FF02
0E24 B6   0985            LDA $985
0E27 27   10              BEQ REPEAT
0E29 7A   0985            DEC $985
0E2C 26   0B              BNE REPEAT
0E2E B6   0986            LDA $986
0E31 84   B0              ANDA #$B0
0E33 B7   0986            STA $986
0E36 B7   FF40            STA $FF40
0E39 BE   0152   REPEAT   LDX #$152
0E3C A6   80     RP1      LDA ,X+
0E3E 81   FF              CMPA #$FF
0E40 26   13              BNE RP2
0E42 8C   015A            CMPX #$15A
0E45 26   F5              BNE RP1
0E47 0C   00              INC <KCLEAR
0E49 96   00              LDA <KCLEAR
0E4B 81   06              CMPA #$6
0E4D 26   20              BNE REPOUT
0E4F 0F   00              CLR <KCLEAR
0E51 0F   01              CLR <KHOLD
0E53 20   1A              BRA REPOUT
0E55 0C   01     RP2      INC <KHOLD
0E57 96   01              LDA <KHOLD
0E59 81   1E              CMPA #$1E
0E5B 26   12              BNE REPOUT
0E5D 80   03              SUBA #$3
0E5F 97   01              STA <KHOLD
0E61 8E   0152            LDX #$152
0E64 A6   84     RP3      LDA ,X
0E66 8A   3F              ORA #$3F
0E68 A7   80              STA ,X+
0E6A 8C   015A            CMPX #$15A
0E6D 26   F5              BNE RP3
0E6F 3B          REPOUT   RTI
                 *
                 * Entry to the main program with alpha lock released and
                 * all text buffer cleared.
                 *
0E70 7F   011A   START1   CLR $11A
0E73 31   8D 0857         LEAY BUFF,PCR
0E77 109F 02              STY <BUFST
0E7A 109F 04              STY <BUFEN
0E7D 86   00              LDA #0
0E7F A7   A0     WIPE     STA ,Y+
0E81 109C 25              CMPY <$25 Top of RAM reached?
0E84 26   F9              BNE WIPE
0E86 20   70              BRA FIN1
                 *
                 * Print on screen routine.
                 * Printing continues until a zero byte is reached.
                 *
0E88 A6   80     PRINT    LDA ,X+
0E8A 27   05              BEQ DONE
0E8C BD   A30A            JSR SCREEN
0E8F 20   F7              BRA PRINT
0E91 39          DONE     RTS
```

```
                01040 *
                01050 * Routine to continue with the text one character at a time
                01060 * at the end of previous text.
                01070 *
0E92 109E 04    01080 CONT    LDY <BUFEN
0E95 34   20    01090         PSHS Y
                01100 *
                01110 * Make sure that the flashing cursor does not go below
                01111 * $400 the top left corner of video screen.
                01120 *
0E97 9E   88    01140 FLASH   LDX <$88
0E99 8C   0400  01150         CMPX #$0400
0E9C 24   04    01160         BHS J1
0E9E 0C   89    01170         INC <$89
0EA0 20   F5    01180         BRA FLASH
                01181 *
                01182 * Alternately place a black ($80) and green ($8F) cursor
                01183 * until a key is pressed.
                01184 *
0EA2 86   80    01190 J1      LDA #$80 Get a black cursor.
0EA4 8D   45    01200         BSR KBSCAN
0EA6 26   06    01210         BNE J2  Go to J2 if key pressed.
0EA8 86   8F    01220         LDA #$8F Wipe cursor with green.
0EAA 8D   3F    01230         BSR KBSCAN
0EAC 27   E9    01240         BEQ FLASH Zero means no key pressed.
                01241 *
                01242 * Place a character on screen until BREAK is presed.
                01243 *
0EAE 81   03    01250 J2      CMPA #$03 BREAK?
0EB0 26   0B    01260         BNE J3
0EB2 86   00    01270         LDA #0
0EB4 35   20    01280         PULS Y
0EB6 A7   A4    01290         STA ,Y
0EB8 109F 04    01300         STY <BUFEN
0EBB 20   3B    01310         BRA FIN1 Go to main menu routine.
                01311 * If Back Space key is pressed, J4 makes it sure that
                01312 * Y reg. is >=BUFEN of previously typed or loaded text.
                01313 * J5 revises the text buffer address pointer and echoes
                01314 * back space to screen. J6 ignores CLEAR key.
0EBD 81   08    01320 J3      CMPA #8
0EBF 26   1B    01330         BNE J6
0EC1 86   8F    01340         LDA #$8F
0EC3 A7   9F 0088 01350       STA [$88]
0EC7 35   20    01360         PULS Y
0EC9 31   3F    01370         LEAY -1,Y
0ECB 109C 04    01380 J4      CMPY <BUFEN
0ECE 24   04    01390         BHS J5
0ED0 31   21    01400         LEAY 1,Y
0ED2 20   F7    01410         BRA J4
0ED4 86   00    01420 J5      LDA #0
0ED6 A7   A4    01430         STA ,Y
0ED8 34   20    01440         PSHS Y
0EDA 86   08    01450         LDA #8
0EDC BD   A30A  01460 J6      JSR SCREEN
0EDF 81   0C    01470         CMPA #$0C
0EE1 23   B4    01480         BLS FLASH
0EE3 35   20    01490         PULS Y
0EE5 A7   A0    01500         STA ,Y+
0EE7 34   20    01510         PSHS Y
0EE9 20   AC    01520         BRA FLASH
                01530 *
                01540 * This routine scans key board for a press. Returns zero
                01550 * if none pressed.
                01551 *
0EEB A7   9F 0088 01560 KBSCAN STA [$88]
0EEF BD   A1C1  01570 J7      JSR INKYS
0EF2 26   03    01580         BNE J8
0EF4 5A         01590         DECB
0EF5 26   F8    01600         BNE J7
0EF7 39         01610 J8      RTS
                01620 *
                01630 * Main menu selection routine.
                01640 *
0EF8 BD   A928  01650 FIN1    JSR CLS
0EFB 30   8D 067E 01660       LEAX MES1,PCR
0EFF 8D   B7    01670         BSR PRINT
0F01 BD   A1C1  01680 WAIT    JSR INKYS
0F04 27   FB    01690         BEQ WAIT
0F06 80   31    01700         SUBA #$31
0F08 25   F7    01710         BLO WAIT
0F0A 81   08    01720         CMPA #8
0F0C 24   F3    01730         BHS WAIT
0F0E 48         01740         ASLA
0F0F 8E   0F19  01750         LDX #MENU
0F12 AE   86    01760         LDX A,X
                01770 * X now points to the absolute address of jump
0F14 BF   0F2A  01780         STX BRANCH
0F17 20   10    01790         BRA JUMP
                01800 *
                01810 * Table of address of different routines.
                01820 *
0F19      0F2C  01830 MENU    FDB REST
0F1B      130D  01840         FDB EDIT
0F1D      104F  01850         FDB CLOAD
0F1F      11BD  01860         FDB LOAD
0F21      0F3B  01870         FDB PAPER
0F23      1173  01880         FDB SAVE
0F25      1173  01890         FDB SAVE
0F27      116E  01900         FDB EXIT
0F29      7E    01910 JUMP    FCB $7E
0F2A      0000  01920 BRANCH  FDB 0
                01930 *
                01940 * This routine prints all the text until end and goes
                01950 * for continuation.
                01960 *
0F2C BD   A928  01970 REST    JSR CLS
0F2F 9E   02    01980         LDX <BUFST
0F31 17   FF54  01990         LBSR PRINT
0F34 30   1F    02000         LEAX -1,X
0F36 9F   04    02010         STX <BUFEN
0F38 16   FF57  02020         LBRA CONT
                02030 *
                02040 * This routine gets the user specifications for printing
                02050 * on printer and stores in the direct page.
                02060 * The location SCL is used for temporary storage of each
                02070 * user input.
                02080 *
0F3B BD   A928  02090 PAPER   JSR CLS
0F3E 0F   0C    02100         CLR <LCP
0F40 30   8D 0708 02110       LEAX MES4,PCR
0F44 17   FF41  02120         LBSR PRINT
0F47 8D   31    02130         BSR SPECS
0F49 96   06    02140         LDA <SCL
0F4B 97   09    02150         STA <LW
0F4D 17   FF38  02160         LBSR PRINT
0F50 8D   28    02170         BSR SPECS
0F52 96   06    02180         LDA <SCL
0F54 97   0B    02190         STA <MARGIN
0F56 96   09    02200         LDA <LW
0F58 90   0B    02210         SUBA <MARGIN
0F5A 97   09    02220         STA <LW
0F5C 17   FF29  02230         LBSR PRINT
0F5F 8D   19    02240         BSR SPECS
0F61 96   06    02250         LDA <SCL
0F63 97   0B    02260         STA <SPACE
0F65 17   FF20  02270         LBSR PRINT
0F68 BD   A1C1  02280 L1      JSR INKYS
0F6B 81   03    02290         CMPA #$03
0F6D 27   89    02300         BEQ FIN1
0F6F 81   0D    02310         CMPA #$0D
0F71 26   F5    02320         BNE L1
0F73 8E   0029  02330         LDX #$0029  Baud Rate = 1200
0F76 9F   95    02340         STX <$95
0F78 20   33    02350         BRA GO
                02360 *
                02370 * This subroutine gets the user input of specifications,
                02380 * converts from decimal to Hex number and returns in SCL
                02390 *
0F7A 0F   06    02400 SPECS   CLR <SCL
0F7C 86   80    02410 L0      LDA #$80
0F7E 17   FE6A  02420         LBSR KBSCAN
0F81 26   07    02430         BNE L2
0F83 86   8F    02440         LDA #$8F
0F85 17   FF63  02450         LBSR KBSCAN
0F88 27   F2    02460         BEQ L0
0F8A 81   0D    02470 L2      CMPA #$0D
0F8C 26   01    02480         BNE L3
0F8E 39         02490         RTS
0F8F 81   30    02500 L3      CMPA #$30
0F91 25   E9    02510         BLO L0
0F93 81   39    02520         CMPA #$39
0F95 22   E5    02530         BHI L0
0F97 BD   A30A  02540         JSR SCREEN
0F9A 80   30    02550         SUBA #$30
0F9C 0D   06    02560         TST <SCL
0F9E 26   04    02570         BNE L4
0FA0 97   06    02580 L6      STA <SCL
0FA2 20   D8    02590         BRA L0
0FA4 C6   0A    02600 L4      LDB #$0A
0FA6 9B   06    02610 L5      ADDA <SCL
0FA8 5A         02620         DECB
0FA9 26   FB    02630         BNE L5
0FAB 20   F3    02640         BRA L6
                02650 *
                02660 * This is the main entr, for printing text on printer.
                02670 *
0FAD 9E   02    02700 GO      LDX <BUFST
                02701 *
                02702 * Start address of current line to be printed is stored
                02703 * at SCL, no. of characters that can be printed within
                02704 * selected line width and margin is determined and is
                02705 * stored at CLW.
                02706 *
0FAF 9F   06    02710 LP99    STX <SCL
0FB1 5F         02720         CLRB
0FB2 A6   80    02730 LP1     LDA ,X+
0FB4 27    28       02740           BEQ STORE
0FB6 81    0D       02750           CMPA #$0D      CR?
0FB8 27    24       02760           BEQ STORE
0FBA 5C             02770           INCB
0FBB D1    09       02780           CMPB <LW
0FBD 26    F3       02790           BNE LP1
0FBF 30    1F       02800           LEAX -1,X
0FC1 81    20       02810 LP2       CMPA #$20    SPACE?
0FC3 27    19       02820           BEQ STORE
0FC5 81    2E       02830           CMPA #$2E  PERIOD?
0FC7 27    15       02840           BEQ STORE
0FC9 81    21       02850           CMPA #$21
0FCB 27    11       02860           BEQ STORE
0FCD 81    3B       02870           CMPA #$3B
0FCF 27    0D       02880           BEQ STORE
0FD1 81    2D       02890           CMPA #$2D
0FD3 27    09       02900           BEQ STORE
0FD5 81    3F       02910           CMPA #$3F
0FD7 27    05       02920           BEQ STORE
0FD9 A6    82       02930           LDA ,-X
0FDB 5A             02940           DECB
0FDC 20    E3       02950           BRA LP2
0FDE D7    0A       02960 STORE     STB <CLW
                    02970 *
                    02980 * Main routine for printing a line on printer.
                    02990 *
0FE0 C6    FE       03000           LDB #$FE Device #-2
0FE2 D7    6F       03010           STB <$6F
0FE4 9E    06       03020           LDX <SCL
                    03021 * Print specified left margin if any.
0FE6 D6    08       03030           LDB <MARGIN
0FE8 27    08       03040           BEQ LP4
0FEA 86    20       03050           LDA #$20
0FEC BD    A2BF     03060 LP3       JSR PRNTR
0FEF 5A             03070           DECB
0FF0 26    FA       03080           BNE LP3
0FF2 D6    0A       03090 LP4       LDB <CLW
0FF4 A6    80       03100 LP13      LDA ,X+
0FF6 81    00       03110           CMPA #$00
0FF8 27    4B       03120           BEQ LP5
0FFA 81    0D       03130           CMPA #$0D
0FFC 26    25       03140           BNE LP6
                    03150 *
                    03160 * This routine sends line feeds equal to spacing selected,
                    03170 * after printing each line.
                    03180 *
0FFE D6    0B       03190 LP14      LDB <SPACE
1000 26    01       03200           BNE LP7
1002 5C             03210           INCB
1003 86    0D       03220 LP7       LDA #$0D
1005 BD    A2BF     03230           JSR PRNTR
1008 0C    0C       03240           INC <LCP
100A 96    0C       03250           LDA <LCP
100C 81    3C       03260           CMPA #$3C         60 LINES?
100E 27    05       03270           BEQ LP8
1010 5A             03280           DECB
1011 26    F0       03290           BNE LP7
1013 20    9A       03300           BRA LP99
                    03310 *
                    03320 * This routine skips six lines after printing sixty lines
                    03330 * on each page and goes to new page.
                    03340 *
1015 C6    06       03350 LP8       LDB #6            6 BLANK LINES.
1017 86    0D       03360 LP10      LDA #$0D
1019 BD    A2BF     03370           JSR PRNTR
101C 5A             03380           DECB
101D 26    FB       03390           BNE LP10
101F 0F    0C       03400           CLR <LCP
1021 20    8C       03410           BRA LP99
                    03420 * This routine prints one character at a time on printer.
                    03430 * Check is made for special printer commands for underlining.
                    03440 *
1023 81    20       03450 LP6       CMPA #$20
1025 27    16       03460           BEQ LP11
1027 81    3C       03470           CMPA #$3C         <?
1029 26    07       03480           BNE LP12
102B 86    1B       03490           LDA #$1B
102D BD    A2BF     03500           JSR PRNTR
1030 86    45       03510           LDA #$45
1032 81    3E       03520 LP12      CMPA #$3E         >?
1034 26    07       03530           BNE LP11
1036 86    1B       03540           LDA #$1B
1038 BD    A2BF     03550           JSR PRNTR
103B 86    52       03560           LDA #$52
103D BD    A2BF     03570 LP11      JSR PRNTR
1040 5A             03580           DECB
1041 26    B1       03590           BNE LP13
1043 20    B9       03600           BRA LP14
                    03610 *
                    03620 * This routine sends final carriage return, changes device
                    03630 * code to screen and returns to main menu.
                    03640 *
```

```
1045 86    0D       03650 LP5       LDA #$0D
1047 BD    A2BF     03660           JSR PRNTR
104A 0F    6F       03670           CLR <$6F
104C 16    FEA9     03680           LBRA FIN1
                    03690 *
                    03700 * Load from cassette tape routine.
                    03710 * BREAK key will abort routine and will return to main menu.
                    03720 *
104F BD    A928     03730 CLOAD     JSR CLS
1052 30    8D 0667  03740           LEAX MESS,PCR
1056 17    FE2F     03750           LBSR PRINT
1059 BD    A1C1     03760 WAIT2     JSR INKYS
105C 27    FB       03770           BEQ WAIT2
105E 81    03       03780           CMPA #$03
1060 1027 FE94      03790           LBEQ FIN1
                    03800 *
                    03810 * Tape load routine is similar to that in Oct.83 Rainbow
                    03820 * page 84
                    03830 *
1064 C6    FF       03840           LDB #$FF  Select motor on.
1066 17    00BB     03850           LBSR MOTOR
1069 1026 0286      03860           LBNE ERROR
106D AE    8D 049E  03870           LDX NONAME,PCR
1071 86    49       03880           LDA #$49   Select input from tape.
1073 C6    FF       03890           LDB #$FF   Select on screen.
1075 17    01C9     03900           LBSR COPEN
1078 1026 0277      03910           LBNE ERROR
107C 9E    04       03920           LDX <BUFEN
107E 17    0210     03930 LOOP4     LBSR CINPUT
1081 1026 026E      03940           LBNE ERROR
1085 A7    80       03950           STA ,X+
1087 4D             03960           TSTA
1088 26    F4       03970           BNE LOOP4
108A 30    1F       03980           LEAX -1,X
108C 9F    04       03990           STX <BUFEN
108E 17    01E0     04000           LBSR CCLOSE
1091 1026 025E      04010           LBNE ERROR
1095 C6    00       04020           LDB #$0   Select motor off.
1097 17    008A     04030           LBSR MOTOR
109A 1026 0255      04040           LBNE ERROR
109E 16    FE57     04050           LBRA FIN1
                    04060 *
                    04070 * Routine for user input of tape/disk filename.
                    04080 *
10A1 BD    A928     04090 NAME      JSR CLS
10A4 8E    02DD     04100           LDX #$2DD
10A7 CC    2055     04110           LDD #$2055
10AA A7    80       04120 LOOP2     STA ,X+
10AC 5A             04130           DECB
10AD 26    FB       04140           BNE LOOP2
10AF B6    011A     04150           LDA $11A
10B2 34    02       04160           PSHS A
10B4 86    FF       04170           LDA #$FF Set the alpha lock for
10B6 B7    011A     04180           STA $11A Capital letter filename.
10B9 30    8D 046E  04190 REDO      LEAX MES,PCR
10BD 17    FDCB     04200           LBSR PRINT
10C0 BD    A393     04210           JSR $A393   Get name.
10C3 D1    0D       04220           CMPB <LENGTH Valid length?
10C5 2E    F2       04230           BGT REDO Do it again if invalid.
10C7 35    04       04240           PULS B Reset the
10C9 F7    011A     04250           STB $11A alpha lock.
10CC 39             04260           RTS
                    04270 *
                    04280 * Routine to save text on cassette tape.
                    04290 * See Oct 83 Rainbow page 84
                    04300 *
10CD C6    09       04310 CSAVE     LDB #9
10CF D7    0D       04320           STB <LENGTH
10D1 8D    CE       04330           BSR NAME
10D3 30    8D 05E6  04340           LEAX MESS,PCR
10D7 17    FDAE     04350           LBSR PRINT
10DA BD    A1C1     04360 WAIT3     JSR $A1C1
10DD 27    FB       04370           BEQ WAIT3
10DF 81    03       04380           CMPA #$03
10E1 1027 FE13      04390           LBEQ FIN1
                    04400 * Main CSAVE routine.
10E5 C6    FF       04410           LDB #$FF   Select motor on.
10E7 8D    3B       04420           BSR MOTOR
10E9 1026 0206      04430           LBNE ERROR
10ED 8E    02DD     04440           LDX #$2DD   Point at name.
10F0 86    4F       04450           LDA #$4F   Select output to tape.
10F2 C6    FF       04460           LDB #$FF   Select on screen.
10F4 17    014A     04470           LBSR COPEN
10F7 1026 01F8      04480           LBNE ERROR
10FB 9E    02       04490           LDX <BUFST
10FD A6    80       04500 CLOOP     LDA ,X+   Read a character.
10FF 4D             04510           TSTA
1100 27    09       04520           BEQ SOUT
1102 17    017C     04530           LBSR CSTPRT
1105 1026 01EA      04540           LBNE ERROR
1109 20    F2       04550           BRA CLOOP
110B 17    0173     04560 SOUT      LBSR CSTPRT
110E 1026 01E1      04570           LBNE ERROR
1112 17    015C     04580           LBSR CCLOSE
1115 1026 010A      04590           LBNE ERROR
```

```
1119 C6   00       04600          LDB #$00   Select motor off.
111B 8D   07       04610          BSR MOTOR
111D 1026 01D2     04620          LBNE ERROR
1121 16   FDD4     04630          LBRA FINI
                   04640 *
                   04650 * This routine turns cassette motor on or off (B=0 : off)
                   04660 *
1124 17   01A7     04670 MOTOR    LBSR BEGIN
1127 5D            04680          TSTB
1128 26   06       04690          BNE MOTORN
112A BD   A7EB     04700          JSR $A7EB   Motor off.
112D 16   01C7     04710          LBRA L21
1130 BD   A7CA     04720 MOTORN   JSR $A7CA   Motor on.
1133 16   01C1     04730          LBRA L21
                   04740 *
                   04750 * Routine to process cassette file name.
                   04760 *
1136 D7   6B       04770 CNAME    STB $6B
1138 CE   01D1     04780          LDU #$01D1
113B 6F   C0       04790          CLR ,U+
113D C6   20       04800          LDB #$20
113F E7   C0       04810 CLEAR    STB ,U+
1141 1183 01DA     04820          CMPU #$01DA
1145 25   F8       04830          BLO CLEAR
1147 CE   01D2     04840          LDU #$01D2
114A E6   80       04850 PNAME    LDB ,X+
114C C1   20       04860          CMPB #$20
114E 25   0B       04870          BLO RETURN
1150 E7   C0       04880          STB ,U+
1152 7C   01D1     04890          INC $1D1
1155 1183 01DA     04900          CMPU #$01DA
1159 25   EF       04910          BLO PNAME
115B 39            04920 RETURN   RTS
                   04930 *
                   04940 * Abort save on tape/disk routines if text buffer is empty.
                   04950 *
115C BD   A928     04960 NOTIT    JSR CLS
115F 30   8D 040C  04970          LEAX ERMES,PCR
1163 17   FD22     04980          LBSR PRINT
1166 BD   A1C1     04990 WAIT4    JSR $A1C1
1169 27   FB       05000          BEQ WAIT4
116B 16   FDBA     05010          LBRA FINI
                   05020 *
                   05030 * Exit to basic with a cold start restoring interrupts and
                   05040 * alpha lock.
                   05050 *
116E 0F   71       05060 EXIT     CLR <$71
1170 7E   A027     05070          JMP QUIT
                   05080 *
                   05090 * Save on tape/disk routines.
                   05100 *
1173 34   02       05110 SAVE     PSHS A
1175 DC   04       05120          LDD <BUFEN
1177 93   02       05130          SUBD <BUFST
1179 27   E1       05140          BEQ NOTIT
117B 35   02       05150          PULS A
117D 81   0A       05160          CMPA #10
117F 1027 FF4A     05170          LBEQ CSAVE
                   05180 *
                   05190 * Disk save routines. Refer to July 83 Rainbow page 71
                   05200 *
1183 8D   2E       05210          BSR SETUP
1185 17   008D     05220          LBSR LABEL
1188 8E   02DD     05230          LDX #$2DD
118B 108E 01FF     05240          LDY #$1FF
118F 86   4F       05250          LDA #$4F
1191 C6   01       05260          LDB #$1
1193 17   00C9     05270          LBSR OPEN
1196 26   56       05280          BNE GOOFED
1198 30   8D 0532  05290          LEAX BUFF,PCR
119C A6   80       05300 WRITE    LDA ,X+
119E C6   01       05310          LDB #$1
11A0 17   00E5     05320          LBSR DSKPRT
11A3 26   49       05330          BNE GOOFED
11A5 81   00       05340          CMPA #$0
11A7 26   F3       05350          BNE WRITE
11A9 C6   01       05360          LDB #$1
11AB 17   00CA     05370          LBSR CLOSE
11AE 26   3E       05380          BNE GOOFED
11B0 16   FD45     05390          LBRA FINI
                   05400 *
                   05410 * Routine to setup one buffer and verify on.
                   05420 *
11B3 C6   01       05430 SETUP    LDB #$1
11B5 8D   69       05440          BSR FILES
11B7 C6   01       05450          LDB #$1
11B9 17   00EB     05460          LBSR VERIFY
11BC 39            05470          RTS
                   05480 *
                   05490 * Routine to load a disk data file.
                   05500 *
11BD 8D   F4       05510 LOAD     BSR SETUP
11BF 8D   54       05520          BSR LABEL
11C1 8E   02DD     05530          LDX #$2DD
11C4 108E 01FF     05540          LDY #$1FF
11C8 86   49       05550          LDA #$49
11CA C6   01       05560          LDB #$1
11CC 17   0090     05570          LBSR OPEN
11CF 26   1D       05580          BNE GOOFED
```

```
11D1 9E   04       05590          LDX <BUFEN
11D3 C6   01       05600 READ     LDB #$1
11D5 17   00C4     05610          LBSR INPUT
11D8 26   14       05620          BNE GOOFED
11DA A7   80       05630          STA ,X+
11DC 81   00       05640          CMPA #$0
11DE 26   F3       05650          BNE READ
11E0 30   1F       05660          LEAX -1,X
11E2 9F   04       05670          STX <BUFEN
11E4 C6   01       05680          LDB #$1
11E6 17   008F     05690          LBSR CLOSE
11E9 26   03       05700          BNE GOOFED
11EB 16   FD0A     05710          LBRA FINI
11EE 34   04       05720 GOOFED   PSHS B
11F0 BD   A928     05730          JSR CLS
11F3 35   04       05740          PULS B
11F5 30   8D 0318  05750          LEAX ERRMSG,PCR
11F9 17   FCBC     05760          LBSR PRINT
11FC 86   2F       05770          LDA #$2F
11FE 4C            05780 ERR      INCA
11FF C0   0A       05790          SUBB #$0A
1201 24   FB       05800          BCC ERR
1203 CB   3A       05810          ADDB #$3A
1205 BD   A30A     05820          JSR $A30A
1208 1F   98       05830          TFR B,A
120A BD   A30A     05840          JSR $A30A
120D BD   A1C1     05850 LL1      JSR INKYS
1210 27   FB       05860          BEQ LL1
1212 16   FCE3     05870          LBRA FINI
                   05880 *
                   05890 * Routine to process disk filename.
                   05900 *
1215 34   34       05910 LABEL    PSHS B,X,Y
1217 C6   0D       05920          LDB #$0D
1219 D7   0D       05930          STB <LENGTH
121B 17   FE83     05940          LBSR NAME
121E 35   B4       05950          PULS PC,B,X,Y
                   05960 * Routine to setup disk system memory.
1220 17   00AB     05970 FILES    LBSR BEGIN
1223 34   04       05980          PSHS B
1225 BD   CA3B     05990          JSR $CA3B
1228 35   04       06000          PULS B
122A F7   095B     06010          STB $95B
122D CE   0928     06020          LDU #$0928
1230 8E   0989     06030          LDX #$0989
1233 6F   84       06040 DOBUF    CLR ,X
1235 AF   C1       06050          STX ,U++
1237 30   89 0119  06060          LEAX $119,X
123B 5A            06070          DECB
123C 22   F5       06080          BHI DOBUF
123E 16   00B6     06090          LBRA L21
                   06100 * Routine to open cassete file.
1241 17   00BA     06110 COPEN    LBSR BEGIN
1244 17   FEEF     06120          LBSR CNAME
1247 81   49       06130          CMPA #$49
1249 27   07       06140          BEQ OPENI
124B 81   4F       06150          CMPA #$4F
124D 27   09       06160          BEQ OPENO
124F 7E   A616     06170          JMP $A616
1252 BD   A629     06180 OPENI    JSR $A629
1255 16   009F     06190          LBRA L21
1258 4F            06200 OPENO    CLRA
1259 BD   A65B     06210          JSR $A65B
125C 16   0098     06220          LBRA L21
                   06230 * Routine to open disk file.
125F 8D   6D       06240 OPEN     BSR BEGIN
1261 10BF 0957     06250          STY $957
1265 34   06       06260          PSHS D
1267 8D   45       06270          BSR FNAME
1269 35   06       06280          PULS D
126B BD   C468     06290          JSR $C468
126E 16   0086     06300          LBRA L21
                   06310 * Routine to close cassete tape file.
1271 8D   58       06320 CCLOSE   BSR BEGIN
1273 BD   A437     06330          JSR $A437
1276 20   7F       06340          BRA L21
                   06350 * Routine to close disk file.
1278 8D   54       06360 CLOSE    BSR BEGIN
127A D7   6F       06370          STB $6F
127C BD   CA53     06380          JSR $CA53
127F 20   76       06390          BRA L21
                   06400 * Routine to write on tape.
1281 8D   4B       06410 CSTPRT   BSR BEGIN
1283 BD   A290     06420          JSR $A290
1286 20   6F       06430          BRA L21
                   06440 * Routine to write on disk.
1288 8D   44       06450 DSKPRT   BSR BEGIN
128A D7   6F       06460          STB $6F
128C BD   A282     06470          JSR $A282
128F 20   66       06480          BRA L21
                   06490 * Routine to read tape file.
1291 8D   3B       06500 CINPUT   BSR BEGIN
1293 0F   70       06510          CLR $70
1295 BD   A17F     06520          JSR $A17F
1298 A7   60       06530          STA 0,S
129A 20   5B       06540          BRA L21
                   06550 * Routine to read disk file.
129C 8D   30       06560 INPUT    BSR BEGIN
129E D7   6F       06570          STB $6F
12A0 BD   A176     06580          JSR $A176
```

```
12A3 A7   E4     06590          STA ,S
12A5 20   50     06600          BRA L21
                 06610 * Routine to set verify on.
12A7 8D   25     06620 VERIFY   BSR BEGIN
12A9 F7   0987   06630          STB $987
12AC 20   49     06640          BRA L21
                 06650 * Routine to process disk filename.
12AE C6   FF     06660 FNAME    LDB #$FF
12B0 5C          06670 GETLEN   INCB
12B1 A6   85     06680          LDA B,X
12B3 81   20     06690          CMPA #$20
12B5 24   F9     06700          BCC GETLEN
12B7 6F   E2     06710          CLR ,-S
12B9 B6   095A   06720          LDA $95A
12BC 97   ED     06730          STA $ED
12BE CE   094C   06740          LDU #$94C
12C1 86   20     06750          LDA #$20
12C3 A7   C0     06760 L22      STA ,U+
12C5 1183 0957   06770          CMPU #$957
12C9 26   F8     06780          BNE L22
12CB 7E   CBA4   06790          JMP $CBA4
                 06800 * Routine to prepare everything.(Registers,Error traps etc)
12CE 34   7A     06810 BEGIN    PSHS X,Y,U,DP,A
12D0 4F          06820          CLRA
12D1 1F   8B     06830          TFR A,DP
12D3 B6   018E   06840          LDA $18E
12D6 FE   018F   06850          LDU $18F
12D9 97   0E     06860          STA <DSAVE
12DB DF   11     06870          STU <$11
12DD 86   7E     06880          LDA #$7E
12DF 33   8D 0010 06890         LEAU ERROR,PCR
12E3 B7   018E   06900          STA $18E
12E6 FF   018F   06910          STU $18F
12E9 A6   E4     06920          LDA ,S
12EB 10DF 11     06930          STS <STACK
12EE EE   66     06940          LDU 6,S
12F0 6E   F8 08  06950          JMP [8,S]
                 06960 * Pass through this routine if error.
12F3 54          06970 ERROR    LSRB
12F4 5C          06980          INCB
12F5 20   03     06990          BRA L25
                 07000 * Pass through this if no error.
12F7 5F          07010 L21      CLRB
12F8 20   00     07020          BRA L25
                 07030 * Restore registers and return to caller.
12FA 96   0E     07040 L25      LDA <DSAVE
12FC DE   11     07050          LDU <$11
12FE B7   018E   07060          STA $18E
1301 FF   018F   07070          STU $18F
1304 10DE 11     07080          LDS <STACK
1307 35   7A     07090          PULS A,DP,U,Y,X
1309 32   62     07100          LEAS 2,S
130B 5D          07110          TSTB
130C 39          07120          RTS
                 07130 * Edit routine uses video screen display area ($400-$5FF)
                 07140 * to display portions of text buffer for editing.
                 07150 * Editor uses direct page addresses as follows:
                 07160 * CLN : Cursor address upon entry to new screen page.
                 07170 * SCL : Start of current text buffer address corresponding
                 07180 * to top left corner of video screen.
                 07190 * MARGIN : End of current text buffer address corresponding
                 07200 * to bottom right corner of video screen.
130D 8E   0400   07210 EDIT     LDX #$400 First, the cursor pointer at top left corner.
1310 9F   0A     07220          STX <CLN
1312 DC   02     07230 EDIT1    LDD <BUFST
1314 DD   06     07240 NEWPGE   STD <SCL
1316 C3   01FF   07250          ADDD #$1FF
1319 1093 04     07260          CMPD <BUFEN
131C 25   05     07270          BLO SKIP
131E DC   04     07280          LDD <BUFEN
1320 83   0001   07290          SUBD #1
1323 DD   08     07300 SKIP     STD <MARGIN
1325 17   00B3   07310          LBSR COPY
1328 109E 0A     07320          LDY <CLN
                 07330 * This routine waits for user to press a key. Y reg. points
                 07340 * to the screen address of cursor location. The character
                 07350 * and a black cursor ($80) are flashed alternately until
                 07360 * a key is pressed.
132B E6   A4     07370 EDWAIT   LDB ,Y
132D 34   04     07380          PSHS B Save character on stack.
132F 8E   0400   07390          LDX #$400
1332 BD   A7D3   07400          JSR DELAY
1335 BD   A1C1   07410          JSR INKYS
1338 C6   80     07420          LDB #$80 Get a black cursor.
133A E7   A4     07430          STB ,Y Place it at cursor pointer.
133C 8E   0400   07440          LDX #$400
133F BD   A7D3   07450          JSR DELAY
1342 35   04     07460          PULS B Get the character from stack.
1344 E7   A4     07470          STB ,Y Place it again at cursor pointer.
1346 81   00     07480          CMPA #0
1348 27   E1     07490          BEQ EDWAIT
134A 81   03     07500          CMPA #3 BREAK?
134C 26   06     07510          BNE SKIP0
                 07520 * Always revise the text buffer to match screen before
                 07530 * exiting routine.
134E 17   00A8   07540          LBSR REVISE
1351 16   FBA4   07550          LBRA FINI
                 07560 * Check if any of the arrow keys is pressed.
                 07570 * Revise cursor pointer if arrow key pressed.
                 07580 * If cursor pointer goes beyond screen display area,
                 07590 * go to next page or previous page.
1354 81   0A     07600 SKIP0    CMPA #$0A
1356 26   0D     07610          BNE SKIP1
1358 31   A8 20  07620          LEAY 32,Y
135B 108C 05FF   07630 EDCHK    CMPY #$5FF
135F 1022 00A7   07640          LBHI NXTPGE
1363 20   C6     07650          BRA EDWAIT
1365 81   09     07660 SKIP1    CMPA #$09
1367 26   04     07670          BNE SKIP2
1369 31   21     07680          LEAY 1,Y
136B 20   EE     07690          BRA EDCHK
136D 81   5E     07700 SKIP2    CMPA #$5E
136F 26   0D     07710          BNE SKIP3
1371 31   A8 E0  07720          LEAY -32,Y
1374 108C 0400   07730 EDCK     CMPY #$400
1378 1025 00A6   07740          LBLO PRVPGE
137C 20   AD     07750          BRA EDWAIT
137E 81   08     07760 SKIP3    CMPA #$08
1380 26   04     07770          BNE SKIP4
1382 31   3F     07780          LEAY -1,Y
1384 20   EE     07790          BRA EDCK
                 07800 * CLEAR key will branch to DELETE routine and SHIFT CLEAR
                 07810 * key will branch to INSERT routine.
1386 81   0C     07820 SKIP4    CMPA #$0C
1388 1027 00AE   07830          LBEQ DELETE
138C 81   5C     07840          CMPA #$5C
138E 1027 00F6   07850          LBEQ INSERT
                 07851 * Place the edited character at cursor pointer.
1392 8D   0A     07860          BSR CHANGE
1394 A7   A0     07870          STA ,Y+
1396 108C 05FF   07880          CMPY #$5FF
139A 22   6E     07890          BHI NXTPGE
139C 20   8D     07900          BRA EDWAIT
                 07910 * This routine changes the ASCII of character for
                 07920 * screen printing.
139E 81   00     07930 CHANGE   CMPA #0
13A0 26   02     07940          BNE SK5
13A2 86   FF     07950          LDA #$FF
13A4 81   0D     07960 SK5      CMPA #$0D
13A6 26   02     07970          BNE SKIP5
13A8 86   BF     07980          LDA #$BF
13AA 81   60     07990 SKIP5    CMPA #$60
13AC 25   08     08000          BLO SKIP6
13AE 81   80     08010          CMPA #$80
13B0 24   04     08020          BHS SKIP6
13B2 80   60     08030          SUBA #$60
13B4 20   06     08040          BRA CHNGOK
13B6 81   40     08050 SKIP6    CMPA #$40
13B8 24   02     08060          BHS CHNGOK
13BA 8B   40     08070          ADDA #$40
13BC 39          08080 CHNGOK   RTS
                 08090 * This routine converts back the screen byte into ASCII
                 08100 * character for placing it in the text buffer.
13BD 81   FF     08110 UNCHNG   CMPA #$FF
13BF 26   03     08120          BNE NONULL
13C1 4F          08130          CLRA
13C2 20   16     08140          BRA UNOK
13C4 81   BF     08150 NONULL   CMPA #$BF
13C6 26   04     08160          BNE SKIP7
13C8 86   0D     08170          LDA #$0D
13CA 20   0E     08180          BRA UNOK
13CC 81   40     08190 SKIP7    CMPA #$40
13CE 24   04     08200          BHS SKIP8
13D0 8B   60     08210          ADDA #$60
13D2 20   06     08220          BRA UNOK
13D4 81   60     08230 SKIP8    CMPA #$60
13D6 25   02     08240          BLO UNOK
13D8 80   40     08250          SUBA #$40
13DA 39          08260 UNOK     RTS
                 08270 * This routine brings a copy of portion of the text
                 08280 * buffer area to the video screen.
13DB BD   A928   08290 COPY     JSR CLS
13DE 108E 0400   08300          LDY #$400
13E2 9E   06     08310          LDX <SCL
13E4 A6   80     08320 COP1     LDA ,X+
13E6 8D   B6     08330          BSR CHANGE
13E8 A7   A0     08340 COP2         STA ,Y+
13EA 9C   08     08350          CMPX <MARGIN
13EC 23   F6     08360          BLS COP1
13EE 108C 0600   08370          CMPY #$600
13F2 27   04     08380          BEQ COPOUT
13F4 86   FF     08390          LDA #$FF
13F6 20   F0     08400          BRA COP2
13F8 39          08410 COPOUT       RTS
                 08420 * This subroutine takes the edited text from the screen
                 08430 * area and places it back at the proper location in the
                 08440 * text buffer area.
13F9 108E 0400   08450 REVISE   LDY #$400
13FD 9E   06     08460          LDX <SCL
13FF A6   A0     08470 REV1     LDA ,Y+
1401 8D   BA     08480          BSR UNCHNG
1403 A7   80     08490          STA ,X+
1405 9C   08     08500          CMPX <MARGIN
1407 23   F6     08510          BLS REV1
1409 39          08520          RTS
                 08530 * If Y > $5FF cursor goes to top of next page in this
                 08540 * routine. The text buffer is always revised to match
                 08550 * screen buffer before going to next page.
```

```
140A 31   A9 FE00  00560 NXTPGE  LEAY -512,Y
140E 109F 0A       00570         STY <CLW
1411 8D   E6       00580         BSR REVISE
1413 DC   08       00590         LDD <MARGIN
1415 C3   0001     00600         ADDD #1
1418 1093 04       00610         CMPD <BUFEN
141B 1027 FEF3     00620         LBEQ EDIT1
141F 16   FEF2     00630         LBRA NEWPGE
                   00640 * If Y < $400 the cursor goes to bottom of previous page.
                   00650 * The text buffer is always revised to match screen buffer
                   00660 * before going to previous page.
1422 31   A9 0200  00670 PRVPGE  LEAY $200,Y
1426 109F 0A       00680         STY <CLW
1429 8D   CE       00690         BSR REVISE
142B DC   06       00700         LDD <SCL
142D 83   0200     00710         SUBD #$200
1430 1093 02       00720         CMPD <BUFST
1433 1025 FEDB     00730         LBLO EDIT1
1437 16   FEDA     00740         LBRA NEWPGE
                   00750 * Delete a character routine.
                   00760 * One character at a time is deleted and the text on
                   00770 * screen is moved one to the left upto the bottom right
                   00780 * corner of screen. The next character from the text buffer
                   00790 * area is brought to screen. All the characters in the
                   00800 * text buffer are also moved one address down.
143A 1F   21       00810 DELETE  TFR Y,X Get cursor pointer in X reg.
143C 30   01       00820         LEAX 1,X        X=X+1
143E 8C   0600     00830         CMPX #$0600 Is it beyond screen buffer?
1441 27   26       00840         BEQ DEL2 Go to DEL" if yes.
1443 A6   84       00850         LDA ,X Get the character.
1445 A7   82       00860         STA ,-X Place it to the left.
1447 81   FF       00870         CMPA #$FF End of the text?
1449 26   11       00880         BNE DEL1 Goto DEL1 if not.
                   00881 * End of text means time to revise text buffer.
144B 9E   08       00890         LDX <MARGIN
144D 9F   04       00900         STX <BUFEN
144F 30   1F       00910         LEAX -1,X
1451 9F   08       00920         STX <MARGIN
1453 34   20       00930         PSHS Y Save Video screen cursor pointer.
1455 8D   A2       00940         BSR REVISE
1457 35   20       00950         PULS Y Get back the cursor pointer.
1459 16   FECF     00960         LBRA EDWAIT
                   00961 * Keep on shifting characters to the left until end of
                   00962 * video screen buffer.
145C 30   02       00970 DEL1    LEAX 2,X
145E 8C   0600     00980         CMPX #$600
1461 27   06       00990         BEQ DEL2
1463 A6   84       01000         LDA ,X
1465 A7   82       01010         STA ,-X
1467 20   F3       01020         BRA DEL1
                   01021 * Time to place first character from next text buffer
                   01022 * area into bottom right corner of video screen buffer.
1469 9E   08       01030 DEL2    LDX <MARGIN
146B 30   01       01040         LEAX 1,X
146D A6   84       01050         LDA ,X
146F 27   12       01060         BEQ DEL4 Goto DEL4 if end of text.
1471 17   FF2A     01070         LBSR CHANGE
1474 B7   05FF     01080         STA $5FF
1477 30   1F       01090         LEAX -1,X
                   01091 * Move all the text buffer characters one address down.
1479 30   02       01100 DEL3    LEAX 2,X
147B A6   84       01110         LDA ,X
147D 27   04       01120         BEQ DEL4
147F A7   82       01130         STA ,-X
1481 20   F6       01140         BRA DEL3
                   01141 * shift last character one address down.
1483 A7   82       01150 DEL4    STA ,-X
1485 9F   04       01160         STX <BUFEN
1487 16   FEA1     01170         LBRA EDWAIT
                   01180 * Inserting characters, one at a time in the middle.
                   01190 * All the text after cursor location is moved one to the
                   01200 * right to make room for new user input. The text in the
                   01210 * text buffer area after the character at bottom right of
                   01220 * screen is also moved up one address at the same time.
148A 1F   21       01230 INSERT  TFR Y,X Get cursor pointer in X reg.
                   01231 * Wait for insert.
148C BD   A1C1     01240 INWAIT  JSR INKYS
148F 27   FB       01250         BEQ INWAIT
1491 81   03       01260         CMPA #$03 BREAK?
1493 1027 FE94     01270         LBEQ EDWAIT Exit insert mode.
1497 17   FF04     01280         LBSR CHANGE
149A 34   02       01290         PSHS A Save character to be inserted.
149C A6   80       01300         LDA ,X+ Get character at cursor pointer : X=X+1
149E 8C   0600     01310         CMPX #$0600 Out of screen buffer?
14A1 27   2A       01320         BEQ INS2 Go to INS2 if yes.
14A3 E6   84       01330 INS0    LDB ,X Get next character in B reg.
14A5 A7   80       01340         STA ,X+ Place previous character to the right.
14A7 8C   0600     01350         CMPX #$0600
14AA 27   1F       01360         BEQ INS1
14AC 1E   89       01370         EXG A,B Switch character from B to A
14AE 81   FF       01380         CMPA #$FF End of text?
14B0 26   F1       01390         BNE INS0 Keep on moving characters to the right.
14B2 8C   05FF     01400         CMPX #$05FF End of screen buffer?
14B5 27   EC       01410         BEQ INS0
                   01411 * When text buffer is smaller than screen buffer following
                   01412 * routine is required.
14B7 34   10       01420         PSHS X Save cursor pointer
14B9 9E   04       01430         LDX <BUFEN X=text buffer end pointer.
14BB 9F   08       01440         STX <MARGIN Store it to correspond bottom of screen buffer.
14BD 30   01       01450         LEAX 1,X X=X+1
14BF 9F   04       01460         STX <BUFEN Increase BUFEN
14C1 35   10       01470         PULS X Get cursor pointer again.
14C3 A7   84       01480         STA ,X Store character at cursor pointer.
14C5 35   02       01490         PULS A Get character to be inserted.
14C7 A7   A0       01500         STA ,Y+ Place it at cursor pointer. : Y=Y+1
14C9 20   BF       01510         BRA INSERT Branch to insert.
                   01511 * Here when text buffer needs to be moved up one address.
14CB 1E   89       01520 INS1    EXG A,B Switch characters from B to A.
                   01521 * Character from Bottom right corner of video screen
14CD 17   FEED     01530 INS2    LBSR UNCHNG
14D0 9E   08       01540         LDX <MARGIN
14D2 30   01       01550         LEAX 1,X
                   01551 * This section moves the remaining text buffer one address up.
14D4 E6   84       01560 INS3    LDB ,X
14D6 A7   80       01570         STA ,X+
14D8 1E   89       01580         EXG A,B
14DA 81   00       01590         CMPA #0
14DC 26   F6       01600         BNE INS3
14DE A7   84       01610         STA ,X
14E0 9F   04       01620         STX <BUFEN
14E2 35   02       01630         PULS A Get the charater to be inserted.
14E4 A7   A0       01640         STA ,Y+ Place it at cursor pointer. : Y=Y+1
14E6 108C 05FF     01650         CMPY #$5FF Within screen buffer?
14EA 23   9E       01660         BLS INSERT Go insert more characters.
                   01661 * Make sure to revise text buffer corresponding to the
                   01662 * screen buffer
14EC 17   FF0A     01670         LBSR REVISE
                   01671 * Insert to continue at top of video screen as new page.
14EF DC   08       01680         LDD <MARGIN
14F1 C3   0001     01690         ADDD #1
14F4 DD   06       01700         STD <SCL
14F6 C3   01FF     01710         ADDD #$1FF
14F9 1093 04       01720         CMPD <BUFEN
14FC 25   05       01730         BLO INS4
14FE DC   04       01740         LDD <BUFEN
1500 83   0001     01750         SUBD #1
                   01751 * Insert continues here when next text buffer is smaller
                   01752 * than video screen buffer.
1503 DD   08       01760 INS4    STD <MARGIN
1505 17   FED3     01770         LBSR COPY
1508 108E 0400     01780         LDY #$400
150C 16   FF7B     01790         LBRA INSERT
150F      0000     01800 NONAME  FDB 0
1511      3C       01810 ERRMSG  FCC "(Break) TO EXIT. ERROR 0"
1529      0000     01820         FDB $0
152B      20       01830 MES     FCC "     LOAD/SAVE ROUTINE"
1541      0D       01840         FCB $0D
1542      20       01850         FCC "     PRESS break TO EXIT"
155A      0D0D     01860         FDB $0D0D
155C      20       01870         FCC " enter FILE NAME: "
156E      00       01880         FCB $0
156F      0D       01890 ERMES   FCB $0D
1570      42       01900         FCC "BUFFER EMPTY"
157C      00       01910         FCB $00
157D      20       01920 MES1    FCC +     A SIMPLE TEXT PROCESSOR+
1598      0D       01930         FCB $0D
1599      20       01940         FCC +       BY A.K. BASARGEKAR+
15B1      0D0D     01950         FDB $0D0D
15B3      20       01960     FCC + TEXT IN < > WILL BE UNDERLINED+
15D2      0D       01970         FCB $0D
15D3      20       01980         FCC + SELECT 1-8. HIT break FOR MENU+
15F2      0D0D     01990         FDB $0D0D
15F4      20       10000         FCC + 1 - COMPOSE+
1600      0D       10010         FCB $0D
1601      20       10020         FCC + 2 - EDIT+
160A      0D       10030         FCB $0D
160B      20       10040         FCC " 3 - CLOAD"
1615      0D       10050         FCB $0D
1616      20       10060         FCC " 4 - DLOAD"
1620      0D       10070         FCB $0D
1621      20       10080         FCC + 5 - PRINT+
162B      0D       10090         FCB $0D
162C      20       10100         FCC + 6 - CSAVE+
1636      0D       10110         FCB $0D
1637      20       10120         FCC " 7 - DSAVE"
1641      0D       10130         FCB $0D
1642      20       10140         FCC + 8 - EXIT+
164B      00       10150         FCB $0
164C      20       10160 MES4    FCC " TOTAL LINE WIDTH = "
1660      000D     10170         FDB $000D
1662      0D       10180         FCB $0D
1663      20       10190         FCC " LEFT MARGIN = "
1672      000D     10200         FDB $000D
1674      0D       10210         FCB $0D
1675      20       10220         FCC " LINE SPACING = "
1685      000D     10230         FDB $000D
1687      0D       10240         FCB $0D
1688      20       10250         FCC " TURN ON PRINTER AND enter"
16A2      0D0D     10260         FDB $0D0D
16A4      20       10270         FCC " OR PRESS break TO EXIT"
16BB      00       10280         FCB 0
16BC      00       10290         FCB 0
16BD      52       10300 MES5    FCC "READY CASSETTE?"
16CC      0D00     10310         FDB $0D00
16CE               10320 BUFF    RMB 0
          0E00     10330         END START
```

Designing Your Own Adventure

By George Firedrake and Art Canfil

If you have never played a role playing game and want to begin playing, try a play-by-mail (PBM) game. Flying Buffalo Inc. created the play-by-mail industry. Anyone can learn to play these games. No previous gaming experience is required.

Begin by getting the rules for the game you play from Flying Buffalo Inc., Dept. GMA, P.O. Box 1467, Scottsdale, AZ 85252-1467. Below are names of PBM games and the prices for the rules.

— STARWEB	$2.00
— HEROIC FANTASY	1.00
— BATTLE PLAN	0.50
— NUCLEAR DESTRUCTION	0.25
— GALACTIC CONFLICT	1.00
— STARLORD	1.00
— BOARD OF DIRECTORS	0.25
— FEUDAL LORDS	1.00

Last time we suggested you sign up for *HEROIC FANTASY* and make a move every two weeks or once a month. First get the rules, then design a party of Adventurers and send them in as described in the rules, of course.

Your characters can be human or otherwise. Each character is a fighter or magic-user, but not both. The strength (STR) of a character is used to attack other characters or monsters, to defend oneself and others, cast magic spells, and numerous other things. The constitution (CON) of a character determines the amount of damage a character can withstand and continue living. Each character type has a price (COST). Here are all possible character types.

CODE	KINDRED	CLASS	STR	CON	COST
F	Fairy	Fighter	1	1	1
F	Fairy	Magic-user	1	1	2
G	Gremlin	Fighter	3	4	3
L	Leprechaun	Magic-user	3	4	4
H	Hobbit	Fighter	5	15	5
H	Hobbit	Magic-user	4	15	7
K	Goblin	Fighter	7	20	6
P	Human	Fighter	15	30	9
P	Human	Magic-user	10	30	11
E	Elf	Fighter	25	25	15
E	Elf	Magic-user	20	25	18
D	Dwarf	Fighter	30	40	23
D	Dwarf	Magic-user	30	40	36
O	Ogre	Fighter	35	40	29
O	Ogre	Magic-user	35	40	46
T	Troll	Fighter	50	50	57
X	Giant	Fighter	60	60	72

For any character, you may choose the name and whether the character is male or female.

You assemble a party of Adventurers by "buying" up to 15 characters. You have 100 points to spend in acquiring characters.

Let's try it. For our first group, how about a big guy and 14 tiny helpers? Our group consists of a giaht and 14 fairy magic-users.

QTY	KINDRED	CLASS	STR	CON	POINTS
1	Giant	Fighter	60	60	72
14	Fairy	Magic-user	14	14	28
		TOTALS	74	74	100

Or, instead of 14 fairies, let's try seven leprechauns.

QTY	KINDRED	CLASS	STR	CON	POINTS
1	Giant	Fighter	60	60	72
7	Leprechaun	Magic-user	21	28	28
		TOTALS	81	88	100

The second bunch is higher in both STR and CON than the first group.

When Frodo, et al. set forth to return the ring to Orodruin, his group included hobbits, humans, elves, and dwarves (plus Gollum, of course). Let's put together our own *Fellowship of the Ring.*

QTY	KINDRED	CLASS	STR	CON	POINTS
2	Hobbit	Fighter	10	30	10
2	Hobbit	Magic-user	8	30	14
1	Human	Fighter	15	30	9
1	Human	Magic-user	10	30	11
1	Dwarf	Fighter	30	40	23
1	Elf	Fighter	25	25	15
1	Elf	Magic-user	20	25	18
			118	210	100

This Adventuring party has much more total CON and STR than either previous group. Of course, we really don't know what is important until we send one of our groups into the labyrinth and find out what happens.

YOUR TURN. Design your own bunch of Adventurers. Remember, you have 100 points to spend and you can select, at the most, 15 Adventurers. Choose a name for each character and decide who is male and who is female.

CoCo Can Help Design A Group

The ratios of STR to COST and CON to COST might be

continued on page 31

CoCo LINK

075 - 326370

Telecom have got us again!
It appears that they have Rainbow listed in the Gold Coast phone book only under the CoCoLink phone number. So if you forget our phone number - you're in trouble, because the computer answers 32-6370 automatically.
Save a label from your magazine sometime - it has your sub number, and renewal details on it, as well as both phone numbers, voice and modem.
And speaking of the bulletin board, whilst the power stike is on in Queensland, unless I buy a generator, (which is a distinct possibility), we are not in a position to run CoCoLink.

The new program to run CoColink is about ready - in fact by the time you read this, it will probably be a reality. We have been concerned that there is a lot of time wasted by the computer doing unnecessary checking routines. This will be allieviated, and there will be also be a greater range of options available.

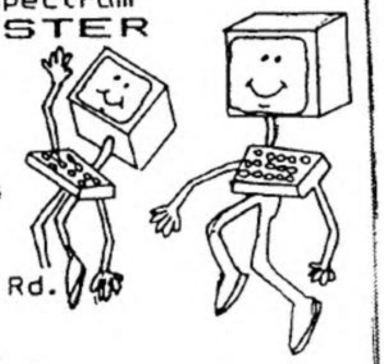
Ken Wagnitz very kindly supplied details of the modifications being carried out by the Perth club to make their CoCos run US terminal programs.
Ken is a pioneer of modeming and I have learnt to respect his advice and abilities. Ken was one of Greg's original 'good guys'.
He took to his CoCo with an oscilloscope, because using Colorcom/E, which he knew worked in the states, he got garbage galore.
Ken says, "I found that the horizontal sync pulses which are used as a source of interupts, and hence as timing in the terminal programs, had missing pulses. 100 pulses are added to the 525 pulses used on NTSC TV, for PAL TV." (This results in 100 additional blank lines, which is why computers designed for PAL TV in the first place, are better.)
The trouble is that dumb Tandy didn't move the interupt line (via a PIA) to the 625 pulse source. So the software was getting 525 out of 625 pulses. The fix is to bend up pin 40 of the appropriate PIA (the interupt input point) and connect it to pin 7 of the AMI brand custom video chip (the source of continuous sync pulses). That custom chip is unique in the CoCo, in that it has 18 pins. The exact proceedure is:

1. Remove the appropriate PIA from its socket.
2. Bend up pin 40 on it.
3. Replace it in it's socket with pin 40 sticking up in the air.
4. Solder a wire to that pin 40.
5. Solder the other end of the wire to pin 7 of the AMI chip. Leave that chip in it's socket, is unmolested on the board, so that pin 7 still contacts the board as well.

Now as to where the hell 'the appropriate PIA' is!
In the first CoCos in Australia, both PIAs are a 6821, the AMI chip is socketed, and it's pin 7 is already wired away to an added small board. The PIA to fiddle is on the left (ie furtherest from the cartridge port). In later CoCos (label in center of case), the PIA scanning the keyboard is a 6822, while the other PIA remains a 6821. The 6822 is the one to fiddle.
The story is that the mod is not necessary for the short case, and neither Ken nor myself have heard of anyone needing to modify the short case.

Software Spectrum
DONCASTER

-Adventures
-Arcades
-Business
-Word Processors
-Utilities



1/234 Blackburn Rd.
East Doncaster
Melb. Vic. 3109.
Phone (03) 842-1205

Want to Connect Your CO-CO to a PARALLEL PRINTER?
Want to RUN your DMP 100/200 up to 30% FASTER in serial mode?
THEN INSTALL A :
MK1 PARALLEL PRINTER INTERFACE

- Locally designed and manufactured
- Compatible with any Standard Centronics Parallel Printer
- Plugs into CO-CO Serial Port and Includes all cables and connectors.
- Increases Printing Speed by up to 30% on Tandy DMP 100/200 Printers when using 4800 or 9600 Baud Rate on your CO-CO.
- Features Six switch selectable baud rates (300-9600)
- Power Pack is required for Printers not supplying power at pin 18 on the Parallel Connector

ONLY: $75 (plus P&P $4)
Add $10 for Power Pack if required.

Available from: Geoff Fiala
18 RUSSELL CRES,
WESTLEIGH. NSW.2120

or phone Sydney (02)-84 3172

RAINBOW WARE

This range of clothing is made here on the Gold Coast.

DESCRIPTION	SIZES	PRICE
V-Neck Crew Neck	12. 14. S. M. L. XL	14.95
Tank Tops	10. 12. 14. S. M. L. XL	13.85
Tennis Dress	S. M. L.	26.20
Hooded Zip-up Jacket	S. M. L.	31.95
Ladies Hooded Pullover Top	S. M. L.	28.50
Ladies Shorts	S. M. L.	17.95
Children's Tank Tops	6. 8. 10. 12. 14	12.80
Children's T-Shirts	8. 10. 12. 14	14.20
COLOURS AVAILABLE	white, pink, aqua, sky blue, yellow, hot pink, black	

PLEASE SEND — ORDER FORM

______________________ $ ______

______________________ $ ______

______________________ $ ______

______________________ $ ______ AMOUNT ______

Complete the section below with one letter, figure or space per square

Mr. ☐ Mrs. ☐ Miss ☐ Initials ☐☐ Surname ☐☐☐☐☐☐☐☐☐☐☐☐ Subscription No ☐☐☐☐☐☐☐

Address ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐ PC

☐ CASH
☐ CHEQUE
☐ POSTAL ORDER

☐☐☐☐☐☐☐☐☐☐☐☐☐

☐ VISA BANKCARD ☐

______________ Cardholder

I UNDERSTAND THAT I CAN RETURN THESE GOODS WITHIN 7 DAYS FOR A FULL REFUND FOR ANY REASON

Of Back Issues, Tapes and Things.

With the exception of Nov 82, we have copies of all back issues available, and in fact need to reduce our stocks of many of them. The early copies of Rainbow are a source of excellent information for the new CoCo owner. The later copies reflect the growing knowledge of the average user of the time. There are games, utilities, hints, and programs for educational, business and club use. There are also many tutorials and articles of interest.

We also have considerable numbers of GoCo Magazine. If you don't have a full set of GoCo's, give me a call!

For those who want to complete their Rainbow collections, we are offering a one for three deal. Buy any three pre August 1984 Rainbows during February, and we'll give you one more of your choice free!

CoCoOz and MiCoOz this Month.

The programs in CoCoOz this month include a screen dump, a program whichdraws then describes the Planets of the Solar System, Matcher an educational game for 3 to 5 yearolds, 2 Maze programs and more.

MiCoOz includes a very clever Educational program by Grahame Pollack on States of Australia, a piece of Irish stupidity, Palindromic Numbers and more.

ANNOUNCING The BEST of CoCoOz!!

To assist teachers and others who are involved with children in learning situations, we haye compiled a 14 program tape (or disk) which refelcts some of the better Educational programming.

Programs include Quizes on Flags and Rivers, the classic "Fractut", a fractions tutor, and "Taxman", a program which teaches Factors. Many of our best Writers are represented and we fully recommend this tape to Educators with CoCos who can't decide what to do with them!

'The Best of CoCoOz' is available for $10.00 on tape, or $21.95 on disk, postage paid.

Now is the time to subscribe to Australian Rainbow

Copies of back issues can be obtained, subject to the availability of stocks, by using this order form and marking clearly which issues you require to be sent to you. Each issue costs $3.95 including postage and packing. Please enclose your cheque/postal order made payable to: Graham Morphett, PO Box 1742, Southport, 4215.

Send off the slip below to ensure that you get on the mailing list for MARCH

CUT OUT AND MAIL or PHONE BANKCARD NO. (075 51 0015)

Subscription Rates

AUSTRALIAN CoCo/MiCo/softgold		AUSTRALIAN RAINBOW
☐ $3.45	Latest per copy	$3.95 ☐
☐ $19	6 months	$21.50 ☐
☐ $31	12 months	$35 ☐

Back Issues

☐ $3 per copy
1st Issue '83

Feb '85 onwards	☐	$3.95
Aug '84-Jan '85	☐	$3.25
to July '84	☐	$3.00

☐ MiCo — First Issue Oct. '83
☐ CoCo/MiCo First Issue Aug. '84

DISKS & TAPES

Rainbow on Tape (program listings) $12 for month of ☐
Please note that RAINBOW on TAPE is issued irregularly

Tape Monthly

CoCoOz			MiCoOz		
☐	Latest	$8	☐	Blank tapes 12 for $18 or $1.70 ea.	☐
☐	6 months	$42	☐	Cassette Cases 10 for $5	☐
☐	12 months	$75	☐	Disks — $3.50 ea. ☐ 10 for $28.99	☐

BOOKS

Byte Elementary	$5.95	☐
Help Medium	SORRY OUT OF STOCK	☐
Facts Advanced	$11.95	☐
MiCoHelp Medium	$9.95	☐
MiCo Exposed V. Advanced	$11.50	☐

BULLETIN BOARD

CoCo Link — Annual Sub. $29 ☐

☐☐☐☐☐☐☐☐☐☐☐☐☐☐

☐ VISA BANKCARD ☐

________________ Cardholder

☐ CASH
☐ CHEQUE
☐ POSTAL ORDER

VISA

MasterCard

New Subscription ☐ Renewal ☐

____

BLOCK CAPITALS PLEASE

If you already subscribe to either Australian Rainbow or Australian CoCo please place Subscription No

Complete the section below with one letter, figure or space per square.

FIRST NAME SECOND NAME

Address

PC

Telephone Number STD Code Local Number

CUT OUT AND MAIL or PHONE BANKCARD NO. (075 51 0015)

The CoCoConnection

Connect your CoCo to the outside world.

Control Robots, Models, Alarms, Lighting Systems, Solar Panels for water or electrical generation, or create your own special use.

There are two models.

Mark I is available now and has eight reporting lines, and eight outputs.

Mark II to be released later in the year, has 4 inputs, and 12 outputs, and can accept the ADDA pack. ADDA packs provide 4 additional inputs, and 12 additional outputs. CoCoConnection will accept a maximum of 16 ADDA packs.

CoCoConection comes complete with a driver program which you activate from your own programs.

PRICES:

MARK I	$180.00
MARK II	T.B.A.
ADDA PACK	T.B.A.

AVAILABLE FROM:
AUSTRALIAN RAINBOW
BLAXLAND COMPUTER CENTRE
CASULA HOBBIES

Please allow 3 weeks for delivery.

Sonya,
do an ad to tell everyone ~~x~~ that the Tandy voice pack Speller program is now available for $39.95 from us.
Thanks
[illegible]

ARE YOUR WALKING FINGERS GETTING FOOTSORE ?

Tired of typing in those long, but wonderful, programs from each issue of the RAINBOW? Now, you can get RAINBOW ON TAPE and give those tired fingers a rest. With RAINBOW ON TAPE, you'll be able to spend your time enjoying programs instead of just typing . typing . typing them! All you need to do ever again is pop a RAINBOW ON TAPE cassette into your recorder. CLOAD and RUN any one you want.

RAINBOW ON TAPE is available as a single issue. It is the perfect complement for the RAINBOW itself.

LATEST & BACK ISSUES $12 each

The Best Color Computer Magazine
Offers The Best Tape Service

Average is 17 Programs
AVAILABLE NOW

Commenced April 82

ORDER RAINBOW ON TAPE TODAY!

Please note that subscriptions to this Tape Monthly are filled spasmodically due to constant delay in receipt of American Master

continued from page 26

useful indexes to help design a group of Adventurers. Here are some examples.

Fairy fighter:	STR/POINTS=1	CON/POINTS=1
Fairy magic-user:	STR/POINTS=.5	CON/POINTS=1
Goblin fighter:	STR/POINTS=1.17	CON/POINTS=3.67

Goblins are durable, compared to their cost, while fairies are fragile, relative to their cost. You can buy a lot of CON for your money by stocking up on goblins!

We have in mind several programs to help design Adventuring teams and play *HEROIC FANTASY*. First, we need a database of information about character types. Here it is:

```
32ØØØ REM**HEROIC FANTASY GMA 21
32ØØ2 REM**CHARACTER TYPES
32ØØ4 REM**CODE$,KIN$,CLASS$,STR
,CON,PTS
32Ø1Ø DATA F,FAIRY,F,1,1,1
32Ø2Ø DATA F,FAIRY,M,1,1,2
32Ø3Ø DATA G,GREMLIN,F,3,4,3
32Ø4Ø DATA L,LEPRECHAUN,M,3,4,4
32Ø5Ø DATA H,HOBBIT,F,5,15,5
32Ø6Ø DATA H,HOBBIT,M,4,15,7
32Ø7Ø DATA K,GOBLIN,F,7,2Ø,6
32Ø8Ø DATA P,HUMAN,F,15,3Ø,9
32Ø9Ø DATA P,HUMAN,M,1Ø,3Ø,11
321ØØ DATA E,ELF,F,25,25,15
3211Ø DATA E,ELF,M,2Ø,25,18
3212Ø DATA D,DWARF,F,3Ø,4Ø,23
3213Ø DATA D,DWARF,M,3Ø,4Ø,36
3214Ø DATA O,OGRE,F,35,4Ø,29
3215Ø DATA O,OGRE,M,35,4Ø,46
3216Ø DATA T,TROLL,F,5Ø,5Ø,57
3217Ø DATA X,GIANT,F,6Ø,6Ø,72
3218Ø DATA Z,ENDFILE,Z,Ø,Ø,Ø
```

This is a small data file consisting of 18 records. Each record contains information about one character type. For instance:

32010 DATA F, FAIRY, F, 1, 1, 1

CODE KINDRED CLASS STR CON POINTS

Line 32004 tells you the names of the variables that we will use to store information from a *DATA* statment.

32004 REM**CODE$, KIN$, CLASS$, STR, CON, PTS

32060 DATA H, HOBBIT, M, 4, 15, 7

The last record, called ENDFILE, with CODE$ = "Z", is not a character type. It is the End-of-File (EOF) record.

$\underbrace{\text{32180 DATA Z,ENDFILE, Z, 0, 0, 0}}_{\text{End-of-File record (No more records in the file.)}}$

End-of-File record
(No more records in the file.)

We have written two programs that use the data file of *HEROIC FANTASY* character types.

The *SCAN CHARACTER TYPES* program begins at Line 1000. It lets you scan the entire file. To run it, type *RUN* or *RUN 1000*. It begins like this.

```
F FAIRY        F  1  1  1
TO DO AGAIN, PRESS SPACE BAR
```

SPACE BAR is in reverse color.

Press the space bar and you get the next record.

```
F FAIRY        F  1  1  1
F FAIRY        M  1  1  1
TO DO AGAIN, PRESS SPACE BAR
```

Keep pressing the space bar until you see 15 records on the screen. Press the space bar again to get the 16th record — the top record is "pushed off the top of the screen" and disappears.

Keep pressing the space bar until ENDFILE appears at the bottom of the screen. Press the space bar again and the CoCo starts over with the first record.

The *COMPUTE COST RATIOS* program begins at Line 2000. Type *RUN 2000* to run this program. First you see:

```
CODE$      STR      CON      PTS      CON/PTS
   CLASS$                        STR/PTS

 F    F     1        1    1     1      1
TO DO AGAIN, PRESS SPACE BAR
```

reverse color

This program works the same way as the *SCAN CHARACTER TYPES* program. Each time you press the space bar, you see another line of information near the bottom of the screen. If you see ENDFILE and press the space bar, the CoCo starts over at the top of the data file.

Here are both programs and the subroutines they use.

The listing:

```
10000 ....195
END .....147
```

```
1 REM**HEROIC FANTASY GMA 21-1
1ØØØ REM**SCAN CHARACTER TYPES
1Ø1Ø CLS
1Ø2Ø RESTORE        'START AT TOP
1Ø3Ø GOSUB 11Ø1Ø 'READ RECORD
1Ø4Ø GOSUB 12Ø1Ø 'SHOW RECORD
1Ø5Ø GOSUB 1ØØ1Ø 'TELL HOW AGAIN
1Ø99 '
11ØØ REM**START OVER IF ENDFILE
111Ø IF KIN$="ENDFILE" THEN 1Ø2Ø
 ELSE 1Ø3Ø
1199 '
2ØØØ REM**COMPUTE COST RATIOS
2Ø1Ø CLS
2Ø2Ø RESTORE        'START AT TOP
2Ø3Ø GOSUB 11Ø1Ø 'READ RECORD
2Ø4Ø GOSUB 13Ø1Ø 'COST RATIOS
2Ø5Ø GOSUB 14Ø1Ø 'SHOW RATIOS
2Ø6Ø GOSUB 1ØØ1Ø 'TELL HOW AGAIN
```

```
2099 '
2100 REM**START OVER IF ENDFILE
2110 IF KIN$="ENDFILE" THEN 2020
 ELSE 2030
2199 '
10000 REM**DO AGAIN SUBROUTINE
10010 PRINT @480, "TO DO AGAIN,
PRESS space bar";
10020 IF INKEY$="" THEN 10020
ELSE RETURN
10099 '
11000 REM**READ RECORD SUBR.
11010 READ CODE$,KIN$,CLASS$,STR
,CON,PTS
11020 RETURN
11099 '
12000 REM**SHOW RECORD SUBR.
12010 PRINT @480, CODE$ TAB(2)
KIN$ TAB(16) CLASS$ TAB(19) STR
TAB(23) CON TAB(27) PTS
12020 RETURN
12099 '
13000 REM**COST RATIOS SUBR.
13010 IF KIN$="ENDFILE" THEN
SC=0: CC=0: RETURN
13020 SC = STR/PTS
13030 SC = INT(100*SC+.5)/100
13040 CC = CON/PTS
13050 CC = INT(100*CC+.5)/100
13060 RETURN
13099 '
14000 REM**SHOW COST RATIOS SUBR
14010 PRINT @480, CODE$ TAB(2)
CLASS$ TAB(5) STR TAB(9) CON
TAB(13) PTS TAB(17) SC TAB(24)CC
14020 RETURN
14099 '
```

Of course, remember to add the data file (lines 32000 through 32180) to the programs.

-- To run *SCAN CHARACTER TYPES*, type *RUN* or *RUN 1000* and press ENTER.

-- To run *COMPUTE COST RATIOS*, type *RUN 2000* and press ENTER.

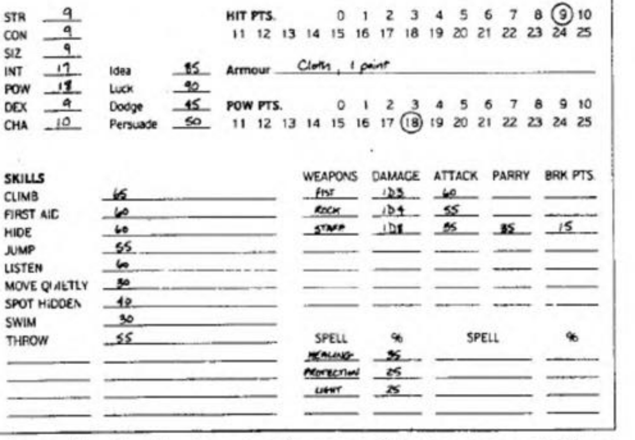
These are "bare bones" programs. Try your hand at improving them. Also think about other programs to help you design an Adventuring team and guide them as they explore the labyrinth.

Who Is A Character?

A character is any imaginary person or other creature created according to the rules of a game system. The characters in *HEROIC FANTASY* are quite simple. The characters in *Dungeons & Dragons* or *RuneQuest* are much more detailed and complex. Characters in *Adventurer's Handbook* are simplified versions of characters found in the very elegant *RuneQuest* system.

We need a way of recording a character's characteristics, abilities, knowledge, possessions, and anything else we want to remember. Below is a blank character record. You may copy it for your own use.

Last time we showed you the character record for Aloysious Anonymous, a very average character. Now meet Rokana.



Character Name Rokana Age 16 Sex F Race Human
Background Villager

STR	9			HIT PTS. 0 1 2 3 4 5 6 7 8 [✓ 9] 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
CON	9			
SIZ	9			
INT	17	Idea	85	Armour Cloth, 1 point
POW	18	Luck	90	
DEX	9	Dodge	45	POW PTS. 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 [✓ 18] 19 20 21 22 23 24 25
CHA	10	Persuade	50	

SKILLS		WEAPONS	DAMAGE	ATTACK	PARRY	BRK PTS.
CLIMB	65	Fist	1D3	60		
FIRST AID	60	Rock	1D4	55		
HIDE	60	Staff	1D8	35	35	15
JUMP	55					
LISTEN	60					
MOVE QUIETLY	30					
SPOT HIDDEN	40					
SWIM	30					

		SPELL	%	SPELL	%
THROW	55				
		Healing	35		
		Protection	25		
		Light	25		

Rokana is a beginning magic-user. She has learned three magic spells called HEALING, PROTECTION, and LIGHT. However, she has not yet mastered these spells. She has a 35 percent chance of successfully casting a HEALING spell and a 25 percent chance with PROTECTION or LIGHT.

Next time we will take Aloysious, Rokana, and perhaps some other characters to a county fair where they can have a wonderful time exercising their skills. In the meantime, we suggest you do some homework. Dig out the following back issues of THE RAINBOW and read "GameMaster's Apprentice."

August 1983 — pages 74-78
October 1983 — pages 170-174
November 1983 — pages 140, 144, 146, 148

Do any of you want us to run a small play-by-mail game? In this game, you would run one character like Aloysious or Rokana. You take your character to a county fair. Today they are called "Renaissance Faires," but in the world of Aloysious and Rokana they were contemporary fairs.

No previous experience is needed to play our play-by-mail game. Your only costs will be a copy of *Adventurer's Handbook* and some self-addressed, stamped envelopes. If you want to play, send a self-addressed, stamped envelope to DragonFun, P.O. Box 310, Menlo Park, CA 94026.

ROLE PLAYING GAMES

Millions of people play fantasy role playing games. A role playing game is a game in which one or more players create and play characters (adventurers) who live their imaginary lives in a specially made game world. The game world is created, managed, and operated by a GameMaster (GM), referee, or dungeon master (DM).

Most people who play role playing games use a formal rule system. Some of the best known are shown below.

***Champions**. Hero Games, 92A 21st Avenue, San Mateo, CA 94402.*

***Dungeons & Dragons (D&D)**. TSR, P.O. Box 756, Lake Geneva, WI 53147.*

***RuneQuest (RQ)**. Chaosium, P.O. Box 6302, Albany, CA 94706.*

***Traveller**. Game Designer's Workshop, P.O. Box 1646, Bloomington, IL 61701.*

***Tunnels & Trolls (T&T)**. Blade, P.O. Box 1210, Scottsdale, AZ 85252.*

Beginners beware! The rule books are formidable. If you are a beginner, we suggest you start with one of the following books, both from Reston Publishing Company, 11480 Sunset Hills Road, Reston, VA 22090.

***Adventurer's Handbook: A Guide to Role Playing Games** by Bob Albrecht & Greg Stafford.*

***Through Dungeons Deep** by Robert Plamondon.*

In "GameMaster's Apprentice," we include how-to-play information for all beginners.

ASSEMBLY FILE

by Kevin

Welcome to the first of what I hope will become an on-going column devoted to newcomers to assembly language programming. I can't help feeling that in establishing this column I am biting off more than I can chew, but we do need a forum to present the answers to those questions that I am asked repeatedly and perhaps generate some interest in the power of simple machine language routines.

In time I hope too that anyone who feels they can contribute an article or two to this column will feel free to do so. In fact they will be more than welcome.

This month I would like begin with a few fundamentals. Before we get too involved I would like to stress what I feel is the best way to approach your learning of assembly programming. Tandy have on their shelves a superb book titled TRS-80 COLOR COMPUTER ASSEMBLY LANGUAGE PROGRAMMING by William Barden, Jr. (Cat. No. 62-2077). At $9.95 it is excellent value for money.

Take this or another text of your preference devoted to programming the 6809 CPU. Sit down by the pool or fireplace and read the book from cover to cover. Don't worry too much about detail, just get a general feel for the subject. Follow this up by re-reading the entire text trying to understand what they are talking about and perhaps working through a few examples. You will then be ready to start applying, in ever increasing detail, what you have learnt using the text book more as a reference concentrating on those areas that interest you (remember you're doing this for pleasure).

Wow! What a colossal task that seems to be. Don't worry, I know you're not going to pay attention to all that I have just said. If everybody did then there would be no need for this column.

I'm going to neatly sidestep my way around the question of an ASSEMBLER (now there's a new word) by saying that you should look long and hard before you dismiss Tandy's EDTASM+ cartridge. You will need an assembler to take your SOURCE CODE (assembly language program which is easy for you and I to understand) and assemble it into MACHINE LANGUAGE (heaps of numbers which the computer finds easy to understand). Hand assembling is a real pain and I would'nt wish the task on even my closest friends. EDTASM+ can be bought from Tandy for $49.95 as a rompack or $79.95 on disk. I will be using the EDTASM+ rompack.

Let's now take a look at an assembly language program and try to make some sense of it all.

```
00100          ORG   $E00
00110  START   LDA   #35     CHR$(35)
00120          STA   $400    TOP LEFT OF SCREEN
00130  LOOP    JMP   LOOP    DO NOTHING FOREVER
00140          END   START
```

Forget about trying to understand what it does for the moment (the program simply puts a # -CHR$(35)- at the top left corner of your screen). Starting at the left, the first column of numbers are the individual line numbers, just like in BASIC.

The next column is the LABEL column. The label equates to a memory location and is used by the assembler as the destination for a branch or loop instruction or even to describe the location of a variable. Look at line number 130. In BASIC we write:

```
130 GOTO 130
```

We have accomplished the same thing with:

```
LOOP   JMP   LOOP
```

The centre column contains the OPCODE. The OPCODE is the one part of the instruction which must be present in the line and is a description of the command which the instruction will carry out. The OPCODE together with the next OPERAND column form the components of an instruction, the OPERAND providing the data or variable acted upon by the OPCODE. Only one instruction may be entered per line.

Finally we have the COMMENTS column and just as in BASIC is there simply to make the program easier to read.

Now that we have an assembly language program that should work we are ready to assemble it into MACHINE CODE. During the assembly process the program may be listed to the printer and three new columns will be seen on the listing. Let's see how our listing now looks:

```
0E00              00100         ORG   $E00
0E00  86  23      00110 START   LDA   #35
0E02  B7  0400    00120         STA   $400
0E05  7E  0E05    00130 LOOP    JMP   LOOP
          0E00    00140         END   START
```

I have left out the COMMENTS column for the sake of clarity. It would normally be included.

The new numbers displayed are pairs of HEXADECIMAL (base 16) numbers. The first column relates to the memory location where the instruction for that line begins. The second column contains the number which the computer understands as the equivelent of our OPCODE and finally

continued on page 42

EZ-Graphics — '85 Style

By Fred B. Scerbo

Each of us often spends a great deal of time making New Year's resolutions which are quite often forgotten within several days of the pledge. Although I have some resolutions which I will try to work into my daily routine, one resolution which I really wanted to make was a pledge to those of you who still have only 16K Extended Color BASIC and have been left along the roadside during our last few "Wishing Well" articles. (The last three have all been for 32K.) While this does not mean that you won't be seeing the most requested sequels, such as "Rockfest II" and "Baseball Fever II," I figured that this would be a good time to offer all of you some shorter listings that will equally satisfy everyone from 16K to 64K. This will be a great relief to all of you who will want to key these listings in but may be too pressed to hammer in the really long ones during the hectic rush of the holiday season. Also, some of you may have just gotten a CoCo for the first time during the holiday season and odds are that more of you got 16K than 64K.

So, as a little New Year's gift for all of you CoCo enthusiasts, here are two short listings which I dare any friends you may have to equal in as few lines on an Apple II or Commodore 64. These are strictly for the fun of creating sharp graphics. Next month we will get back to some more serious wishes.

Who You Gonna Call?

Probably one of the most successful motion pictures of the past summer movie season was the comedy, *Ghostbusters.* Besides being a funny movie, this cinematic effort has as one of its offsprings a symbol which is quickly becoming as common as "Rubik's Cube," Michael Jackson and "Cabbage Patch" dolls. Recently, I have seen this particularly catchy no-ghosts logo showing up on everything from T-shirts to bumper stickers. Well, here's one more place you can look to see this omnipresent poltergeist: on the screen of your CoCo.

Why even do this? Well, as I have said before, young CoCo programmers often take great pride in being able, with just a few program lines, to create graphics which are easily recognized. This eventually will serve to stimulate even the most inexperienced programmer to learn more, and eventually create a program which others can benefit from as well.

Therefore, Listing 1 is an extremely short program which will recreate the *Ghostbusters* logo in rather dramatic detail before your eyes. The actual body of the listing which draws the ghost and the slashed circle is only about 13 lines long. As I just mentioned, I doubt any Apple or Commodore programmers will be able to match this graphic on their machine in as few lines. This just gives you one more weapon to use in convincing others that you made the best choice when you chose a Color Computer.

The actual graphic uses *PMODE 4* with an overlay of *PMODE 3* (without using the *SCREEN* command). The ghost and slash are formed by a combination of semi-circle and *DRAW* commands, accounting for how efficiently this BASIC code can be written. You would need a little trigonometry to get similar results on the other computers.

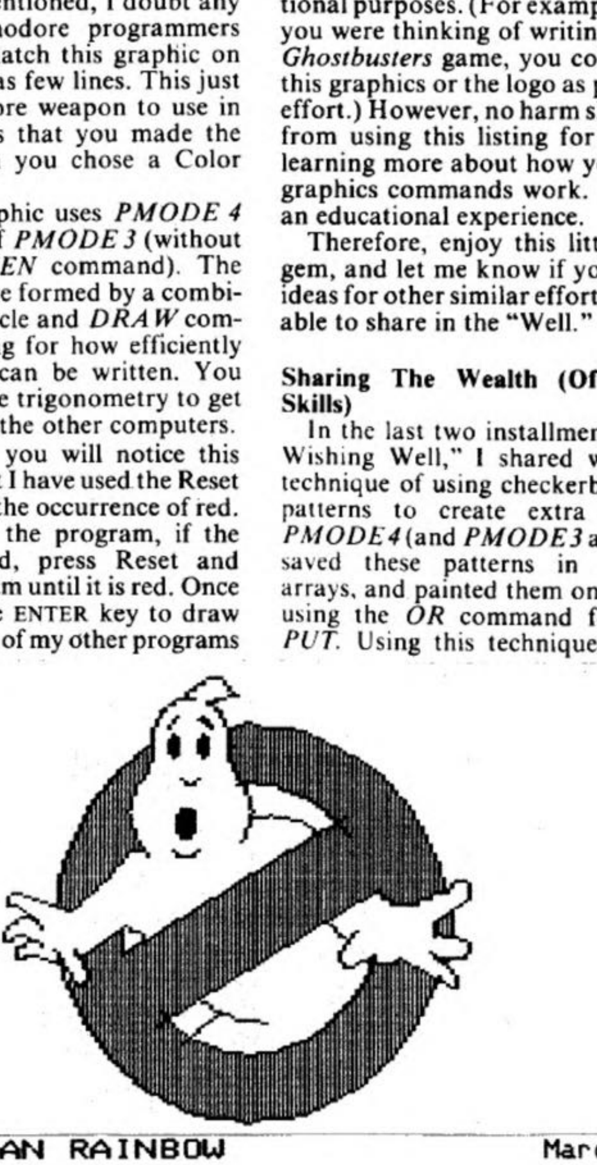
One difference you will notice this time around is that I have used the Reset button to control the occurrence of red. When you *RUN* the program, if the screen is not red, press Reset and *reRUN* the program until it is red. Once it is red, press the ENTER key to draw the graphics. Most of my other programs usually offset a pixel to control the colors without using Reset, but since this was such a short listing, I figured that the Reset was the quickest route to follow.

A final word should be mentioned here before we move on to our second listing. The actual *Ghostbusters* symbol is the property of Paramount Pictures which holds all rights for its commercial use. Therefore, this listing is for your own personal home use for the fun of it, and may not be used for any promotional purposes. (For example, if any of you were thinking of writing your own *Ghostbusters* game, you could not use this graphics or the logo as part of your effort.) However, no harm should come from using this listing for the fun of learning more about how your CoCo's graphics commands work. Consider it an educational experience.

Therefore, enjoy this little graphics gem, and let me know if you have any ideas for other similar efforts I might be able to share in the "Well."

Sharing The Wealth (Of Graphics Skills)

In the last two installments of "The Wishing Well," I shared with you a technique of using checkerboard pixel patterns to create extra colors in *PMODE 4* (and *PMODE 3* as well). We saved these patterns in GET-PUT arrays, and painted them on the screen using the *OR* command found with *PUT.* Using this technique, any area



which has previously been painted black will be filled in with the color found in our array. For a more detailed explanation of how these colors are generated, refer to last month's article.

There was one small drawback with the method used in last month's issue. While the technique was completely effective for what we wanted to accomplish with those football helmet graphics, the routines were not designed for you to easily use if you wanted to use the extra colors in your own drawings. As I promised, I have come up with a way that you could use them easily without having to do a great deal of graphics gymnastics. The method I have listed here will be a piece of cake to anyone who knows how to use the *LINE* command found in your Color Extended language.

Another limitation found in last month's version was the fact that the array covered nearly the full width of the screen. This would mean that your graphics would have to be drawn and colored in a fashion that would not allow you to have a yellow object next to a purple object, since the arrays would overlap each other on the same level. Therefore, these new routines have set up arrays which are only 16 pixels wide and two pixels deep. This allows greater flexibility in this type of painting when more than one color is desired on the same left to right level. It also takes up less than 10 percent of the memory required to do it the original way. The routines used for "Football I and 2" are much faster than this technique. Since speed was more important than flexibility, those were written with speed in mind. As always, you have the classic trade-off. Speed and flexibility are inversely proportional. This month we will emphasize flexibility instead, while sacrificing speed.

Therefore, the BASIC code required to efficiently and easily use these colors has been written to be part of the first 25 lines of your program. If you wish to draw graphics using them, you would simply start your own program lines at Line 100. There is a special syntax which I have developed to handle the colors which I will explain in just a few lines. Simply put, it is a new way of coloring, but at the same time will be very familiar to you.

Actually, the most difficult part of writing this program was thinking of what to draw as a graphics to demonstrate the routines. I didn't want to do a rock logo because those will be showing up in a couple of months, and there was no reason to let the wind out of my sails for that one yet. Secondly, the colors would be of no use for the *Ghostbusters* graphics listed here. Besides, that would make it longer, defeating the whole purpose of doing the logo in the first place.

When the idea for what to draw finally came to me, I wanted to kick myself for not having thought of it in the first place. What is one of the first paintings or drawings that an aspiring young artist starts with? Why, of course, the answer was a bowl of fruit! Sure, it may not be as dramatic as the car from the ZZ Top logo of a few months ago, but it would give me the possibility of drawing and using more than one of these colors side by side.

With this in mind, I developed a set of seven additional color patterns to be included in the arrays. These patterns are set in lines 11-14 and put into the arrays in Line 15. The colors and their corresponding Syntax letters are listed below. Remember, the actual color may depend on how accurately your TV set tint is adjusted to red and blue.

Y = Yellow
B = Light blue
G = Gold or orange
S = Silver or gray
P = Purple (dark)
L = Lime or dark blue
V = Violet

The lime color is not really so much of a green, but depending on your tint, it is about as close as we can get. You will notice that I mentioned that the letters are part of the syntax. You will actually use these letters to call the colors as you need them.

Remember how I mentioned that the syntax would be familiar to anyone who knew how to use the *LINE* command? As you may recall, the syntax for *LINE* is:

```
LINE (x1,y1)—(x2,y2),PSET
```

The variables x1,x2,y1 and y2 are used to define the starting and end points of a line (or box) using x and y coordinates on a field of 256x192 pixels. Therefore, if you wanted to draw a box with opposite coordinates of 10,10 by 20,20 you would write:

```
LINE(10,10)—(20,20),PRESET,BF
```

which would give you a box 10 pixels square painted in black (because of BF which means box filled). To paint this box with our new colors, you would use the same coordinates. These coordinates are placed in a STRING I call PAINT STRING which is identified as PT$ in these routines. The syntax for these coordinates would thus be:

```
PT$="Y010,010—020,020"
```

The Y stands for the color yellow. The next three digits are the coordinate for our x1 coordinate followed by a comma, with the next three digits being the y1 coordinate. We then use a dash and use three digits for x2, a comma, and three digits for y2. The coordinates for a box such as:

```
LINE(10,20)—(30,40),PSET,BF
```

would be:

```
PT$="Y010,020—030,040"
```

In each case, we use this PAINT STRING by following it with the following command:

```
GOSUB 17
```

which takes care of the painting. You will notice that even though the numbers we are using are only two digit numbers, we must use three digits such as 010 for 10, or 006 for the number 6. This is necessary because the PAINT STRING is analyzed in lines 17 and 18 to determine the coordinates and colors. This was much easier than to have you type in the values for five separate variables. If you accidentally use two digits rather than three, the painting will not take place. There will not be an error message. There will just be no painting. Thus, our syntax for PAINT STRING is:

```
PT$="Color,Left Corner,Top
Corner—Right Corner,Bottom
Corner"
```

followed by *GOSUB 17*. All corner coordinates must be three digits. As you can see, if you know how to use *LINE*, you will have no trouble using PAINT STRING.

If you *RUN* the second listing called *Seven More PMODE4 Colors*, you will have a very nice, framed painting of a bowl of fruit with drapes in the background. You will be pleased to see that this does make a very nice graphics to use for showing the colors available on your CoCo.

I did not use all seven colors here. Rather, I used just a few so you would get the idea. To get a nice curtain or draped effect, I used *POKE 178,x* to give a little added realism. Remember, the technique used for this is to use a value between zero and 255 with the *POKE* and the *PAINT* using:

```
PAINT(x,y),,1
```

to get your striped colors. This was described many months ago in THE RAINBOW.

Lines 310 to 380 are designed to let you change *PMODE*s and *SCREEN*s to see how these patterns look under different combinations. Hitting the ENTER key will flip through the various combinations. I have also used Reset to control red in this program. If the screen is not red when you *RUN* it, press Reset until it is and then hit ENTER to continue. Again, I felt that this would be preferable to my other method since the Reset route is very popular, and you might find it easier since you may want to use these routines yourself.

Let's say you have *RUN* this program and now want to use these colors for your own graphics. Load in the program and type:

DEL 110—

and hit ENTER. This will delete all following lines keeping the routines intact. It will also set your screen for *PMODE 4* with *PMODE3* colors. You may wish to alter Line 100 to suit you needs. Since I have already used a number of variables in the routines, you will want to take care not to use these same variables! Here is a list of the variables which you should avoid:

R,B,X,Y,G,S,P,L,V,LC,RC,TC, BC,YY,ZZ

and the string variables:

XX$ and PT$

The variables R and B stand for red and blue, and you may substitute them in the program to suit your needs. I have chosen instead to use the values of 3 and 2 in the program so as to not bury you in variables.

If you wish to use these routines, you may renumber them, but you must leave the *REM* statements intact since this program is under copyright. Feel free to create using these techniques, but remember to give credit where credit is due! That's what makes it possible to share these techniques with you.

Let's try a little experimenting so you can see how this really works. Delete the first lines as I mentioned and type in the following new lines:

```
110 CIRCLE(128,96),60,1,.9
120 PAINT(128,46),1,1
130 PT$="Y068,042—190,150":
GOSUB17
140 LINE(68,42)—(190,150)
,PRESET,B
1000 GOTO1000
```

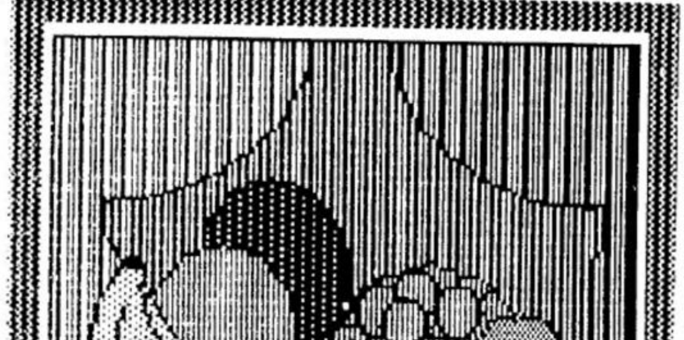
This will draw a circle, paint it black, PAINT STRING it yellow, and surround it with a box that shows the area actually covered by the array. Thus, if you have an irregular shaped object and paint it black, you can fill it in with these colors just as you would with PAINT because we are using *OR* which checks to see if a pixel is set, and if it is not, it sets it to the pattern. You may also need to redraw around the object since the color will fill in any area it overlaps, which is black.

Although these new smaller arrays do mean you can have adjacent colors, be careful not to make the items too close together, as I tried to be sure of with the fruit. With a little experimentation, you will be painting with ease in no time at all. (In fact, you will most likely be seeing the routines and variations on them in upcoming graphics wishes. I mean, why shouldn't I take advantage of this easier method as well?)

In playing with the sample I have just given you, change the letter for the color in Line 130. This will give you a better idea of how to control the colors. Probably the hardest thing to color this way would be concentric circles. If you can handle that one, you can handle most any graphics. Yes, it can be done, but I won't show you how now. Let's see you try it yourself.

Conclusion

When I think of the types of graphics people originally got excited about when Color Extended came to the CoCo, and I see how far we have come with the very same machine, I can only imagine that things will continue to get more exciting. I started with Color BASIC with a $499 16K machine and thought that block graphics were great back in 1981. Who would ever think we would be milking such detail out of this machine without really changing the original language? I'll keep searching for ways to make these things better. You just keep feeding me ideas.



```
60 ....... 231
END .... 118
```

Listing 1:

```
1 '*****************************
2 '*    GHOSTBUSTERS LOGO      *
3 '*    BY FRED B. SCERBO      *
4 '* 149 BARBOUR ST.N.ADAMS.MA*
5 '*      COPYRIGHT (C)  1984   *
6 '* LOGO IS THE PROPERTY OF   *
7 '* OF PARAMOUNT PICTURES &   *
8 '* IS FOR YOUR HOME USE ONLY*
9 '*****************************
10 PMODE4,1:PCLS1:SCREEN1,1:PMOD
E3:R=3:PCLS3
20 IFINKEY$<>CHR$(13)THEN20
30 PCLS4
40 CIRCLE(100,40),20,1,.9,.46,.0
5:DRAW"C4BM100,24NU2R6DR2C1R4ER2
M+6,-1R2L3H6L6G2L2G2LG2DR3"
50 CIRCLE(100,62),30,1,.66,.41,.
63:CIRCLE(100,62),30,1,.66,.9,.1
:DRAW"BM-22,+24M+2,-4BR36M+2,+4F
2BM-16,-20F4M-6,-3R2BL12BUG4BD4B
RD4F2U8F2D5BR9BUNU4F2U8F2D5BD8BL
4G2L4H2BD12BL2D6F2U10R2D10R2U10F
2D6BD6BR2G2L8H2BL16BU2F4H2L6"
60 DRAW"M-12,+10M-16,+10M-24,-6L
4G2D2R4M+10,+6NF2H2L6G2L2G4D2R4E
2R4F2R4F2H2L4G2L4D4R4ER4M+10,+2R
4NH4M+6,+8R4E2U2M-6,-8D2R4M+20,-
10F2R4F2D"
70 CIRCLE(138,80),30,1,.6,.69,.9
:CIRCLE(136,112),42,1,.5,.25,.4
80 DRAW"BM114,130M-18,+10"
90 DRAW"BM166,100F4R2F2NE4G4D2F2
R4M+9,-3F2R9E2M+24,+10R2U4M-12,-
8M+3,+2R4NDR2NDR8E2U2H2M-14,-2ND
U4M+8,-4ND8M+8,-4U4H2L4G4L2NU4L4
```

```
G2L4G2L12H2L2H2"
100 CIRCLE(128,98),92,1,.85,.52,
.675:CIRCLE(128,98),92,1,.85,.74
,.98:CIRCLE(128,98),92,1,.85,.Ø5
,.48
11Ø CIRCLE(128,98),62,1,.85,.74,
.9:CIRCLE(128,98),62,1,.85,.Ø5,.
39
12Ø DRAW"BM68,11ØNU8M+1Ø4,-46BF2
ØBD4BL4M-1Ø4,+46R2"
13Ø PAINT(78,36),R,1:PAINT(134,2
4),R,1:PAINT(218,1ØØ),R,1
14Ø PAINT(2,2),1,1:PAINT(134,5Ø)
,1,1:PAINT(72,1Ø6),1,1:PAINT(92,
136),1,1
15Ø GOTO15Ø
```

```
15........221
25.........18
210.......157
END.......89
```

Listing 2:

```
1 '****************************
2 '* SEVEN MORE PMODE4 COLORS *
3 '*      BY FRED B. SCERBO    *
4 '* 149 BARBOUR ST.N.ADAMS.MA*
5 '*    COPYRIGHT (C)  1984    *
6 '****************************
7 CLEAR1ØØØ:R=3:B=2
8 PMODE4,1:PCLS1:SCREEN1,1:PMODE
3:PCLS3
9 IFINKEY$=CHR$(13)THEN11ELSE9
1Ø 'START COLOR SET
11 CLSØ:PMODE4,1:PCLSØ:SCREENØ,Ø
:DIM Y(3),B(3),G(3),S(3),P(3),L(
3),V(3):LINE(32,Ø)-(48,5),PSET,B
F
12 FORX=31TO47STEP4:PSET(X,Ø,Ø):
PSET(X+2,1,Ø):PSET(X+1,4,Ø):PSET
(X+3,5,Ø):NEXT
13 FORX=32TO47STEP8:PSET(X,8):PS
ET(X+4,9):LINE(X,12)-(X+1,12),PS
ET:LINE(X+4,12)-(X+5,12),PSET:LI
NE(X+2,13)-(X+3,13),PSET:LINE(X+
6,13)-(X+7,13),PSET
14 PSET(X,16):PSET(X+1,17):PSET(
X+4,16):PSET(X+5,17):PSET(X+1,2Ø
):PSET(X+5,21):NEXTX:PMODE3:COLO
R2,3:LINE(32,24)-(48,24),PSET:LI
NE(32,25)-(48,25),PRESET
15 PMODE4:GET(32,Ø)-(47,1),Y,G:G
ET(32,4)-(47,5),B,G:GET(32,8)-(4
7,9),G,G:GET(32,12)-(47,13),S,G:
GET(32,16)-(47,17),P,G:GET(32,2Ø
)-(47,21),L,G:GET(32,24)-(47,25)
,V,G
16 GOTO1ØØ:'PAINTING ROUTINES
17 LC=VAL(MID$(PT$,2,3)):TC=VAL(
MID$(PT$,6,3)):RC=VAL(MID$(PT$,1
Ø,3)):BC=VAL(MID$(PT$,14,3))
18 XX$=LEFT$(PT$,1):IFXX$="Y"THE
N19ELSEIFXX$="B"THEN2ØELSEIFXX$=
"G"THEN21ELSEIFXX$="S"THEN22ELSE
IFXX$="P"THEN23ELSEIFXX$="L"THEN
24ELSEIFXX$="V"THEN25ELSERETURN
19 FORYY=TC TO BC STEP2:FORZZ=LC
 TO RC STEP16:PUT(ZZ,YY)-(ZZ+15,
YY+1),Y,OR:NEXTZZ,YY:RETURN
2Ø FORYY=TC TO BC STEP2:FORZZ=LC
 TO RC STEP16:PUT(ZZ,YY)-(ZZ+15,
YY+1),B,OR:NEXTZZ,YY:RETURN
21 FORYY=TC TO BC STEP2:FORZZ=LC
 TO RC STEP16:PUT(ZZ,YY)-(ZZ+15,
YY+1),G,OR:NEXTZZ,YY:RETURN
22 FORYY=TC TO BC STEP2:FORZZ=LC
 TO RC STEP16:PUT(ZZ,YY)-(ZZ+15,
YY+1),S,OR:NEXTZZ,YY:RETURN
23 FORYY=TC TO BC STEP2:FORZZ=LC
 TO RC STEP16:PUT(ZZ,YY)-(ZZ+15,
YY+1),P,OR:NEXTZZ,YY:RETURN
24 FORYY=TC TO BC STEP2:FORZZ=LC
 TO RC STEP16:PUT(ZZ,YY)-(ZZ+15,
YY+1),L,OR:NEXTZZ,YY:RETURN
25 FORYY=TC TO BC STEP2:FORZZ=LC
 TO RC STEP16:PUT(ZZ,YY)-(ZZ+15,
YY+1),V,OR:NEXTZZ,YY:RETURN
9Ø 'START YOUR PROGRAM HERE
1ØØ PMODE4:PCLS1:SCREEN1,1:PMODE
3
11Ø PMODE4:COLORØ,Ø:LINE(Ø,Ø)-(2
54,192),PSET,B:LINE(12,8)-(243,1
83),PSET,B:PAINT(2,2),Ø,Ø:PMODE3
12Ø PT$="PØØØ,ØØØ-255,ØØ8":GOSUB
17:PT$="PØØØ,182-255,192":GOSUB1
7:PT$="PØØØ,ØØØ-ØØ8,192":GOSUB17
:PT$="P24Ø,ØØØ-252,192":GOSUB17
13Ø GOSUB14Ø:GOTO15Ø
14Ø DRAW"BM4Ø,12ØC1ND4R17ØD4L4G4
D2G2D2G4L2G4L2G4L2G4D6F2R2F2D4L1
16U4E2R2E2U6H4L2H4L2H4L2H2L2H2U2
H2U2H4L4U4":RETURN
15Ø PAINT(5Ø,122),1,1:PT$="SØ36,
12Ø-21Ø,146":GOSUB17:PT$="GØ62,1
48-214,156":GOSUB17:PT$="SØ48,15
8-21Ø,17Ø":GOSUB17:GOSUB14Ø
16Ø GOSUB17Ø:GOTO18Ø
17Ø CIRCLE(6Ø,1ØØ),3Ø,1,1.1,.4,.
7:CIRCLE(13Ø,72),8Ø,1,.5,.2,.5:D
RAW"BM16Ø,1Ø8C1D4G2D2G4":RETURN
18Ø PAINT(158,112),1,1:PT$="YØ3Ø
,Ø76-16Ø,118":GOSUB17:GOSUB17Ø
19Ø CIRCLE(12Ø,8Ø),7Ø,1,.5,.2,.5
:CIRCLE(114,72),8Ø,1,.6,.3,.4:CI
RCLE(66,1ØØ),26,1,1.1,.4,.6
2ØØ CIRCLE(1ØØ,76),3Ø,1,.9:PAINT
(1ØØ,58),1,1:PT$="GØ69,Ø48-12Ø,Ø
```

continued on page 52

16K ECB

LINK

By H. Allen Curtis

Can you do the following with a single cassette load command?

1) Load the text screen to display an introductory message or low resolution picture;
2) Load a BASIC program;
3) Load the graphics screen with a high resolution picture for subsequent display;
4) Load automatically memory protected high RAM with assembly language routines to be called by *USR* functions;
5) Automatically start the BASIC program; and
6) Provide some piracy protection for your program.

If you cannot, then you are missing *Link*. No, I did not mean that you are *the* missing link, but that you are missing out by not using the *Link* program to be presented in this article.

Link is not a pre-loader. That is, *Link* does not have to be loaded into RAM before you issue the command to load your program, screens and subroutines. *Link* concatenates (links) as many as 10 non-contiguous RAM records and writes them on tape. A record is defined here as any program (BASIC or assembly language), any contiguous assembly language routines, or any set of stored data. The linked records written on tape are simply loaded by means of BASIC's *CLOADM* command.

If you want to employ *Link* to record and auto-start an assembly language program instead of a BASIC program, you can readily do so. In fact, *Link* is an assembly language program which will be used to record itself.

Link has the ability to write a record from one RAM location and load it into another specified RAM location without recourse to the offset feature of the *CLOADM* command. Thus, for instance, you may design several text screens, transfer them to new locations in RAM and then use *Link* to write them on tape for future sequential loading and display on the text screen. The fact that you relocate one or more records such as screens does not require you to relocate the other records to be linked and written.

The order in which records are concatenated is left to your discretion. If you, for example, have more than one text screen to be loaded and displayed, you would probably load one or two records between screen records to allow the screen to be displayed for a sufficiently long time.

Link can be employed as part of a protection process for your programs. How *Link* can be used in this way will be discussed in detail at the close of the article.

The program of Listing 1 generates *Link* and stores it in RAM. The strings in lines 20 through 120 of Listing 1 are messages used by *Link* to prompt you in the process of concatenating records and writing them on tape. The values in the *DATA* statements of lines 210 through 520 comprise *Link* routines that actually do the linking and writing of the records on tape. The values in the remaining *DATA* statements form the major portion of the first record to be written on every *Link* produced tape. You do not specify this record. The record is *Link's* means of altering the usual *CLOADM* sequence of instructions to permit the proper loading of concatenated records.

Incorporated into Listing 1 is a check on the accuracy of your typing of the *DATA* statements. Thus, with the use of Rainbow Check Plus you are doubly aided in the correct typing of Listing 1. When you have correctly typed Listing 1 and run the program without the occurrence of any error messages, save it on tape: Type *CSAVE"GENLINK"* and press ENTER.

After saving the program of Listing 1, run it again. Then type *EXEC* and press ENTER. This action will produce the first prompt of *Link*. *Link* requires you to provide a filename for the concatenated records to be put on tape. Usually the filename will be that of the main program whether in BASIC or assembly language. Rather than having a BASIC program that generates *Link*, it is more convenient to have *Link* recorded on tape directly as an assembly language program. Therefore, type the filename *Link* and ENTER it. This will initiate the process of using *Link* to record itself on tape.

The second prompt requests the entry address of the main program, which in this case is *Link*. All the required *Link* addresses have been provided in the REM statement of Line 10 of Listing 1. In accordance with that REM, type 1100 and press ENTER. You do not need to type &H in answering the prompt. The hexadecimal address 1100 is the address at which *Link* starts executing. If any of the characters of the ENTERed address are not a valid hexadecimal digit, a beep alarm will be sounded and the prompt will be repeated.

The next prompt asks for the first source address of the first record that you want on tape. Associated with each record are two sets of addresses — source addresses and destination addresses. The source addresses are the lowest (first) address and the highest (last) address of the record as it is presently located in RAM. The destination addresses are the corresponding RAM addresses into which you want the record to be loaded.

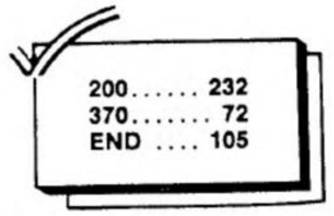
In the case of *Link*, only one record is involved; hence, you should type the first source address of *Link*. That is, type 1000 and press ENTER.

The third prompt is similarly answered by typing and entering the last source address 132B of *Link*.

You will probably want at least two versions of *Link*, one to be loaded in its present RAM location and one destined for high RAM. Therefore, for the former version answer the fourth prompt by typing 1000 and pressing ENTER.

Since you only need to specify one record for *Link*, answer the next prompt by pressing the 'Y' key to indicate yes.

Instead of recording *Link* immediately following *GENLINK* on your cas-


200...... 232
370....... 72
END 105

Listing 1:

```
10 'ENTRY ADDRESS IS &H1100; FIR
ST ADDRESS IS &H1000; LAST ADDRE
SS IS &H132B
20 A$="TYPE & ENTER
30 B$="FILENAME:
40 C$="TYPE (IN HEX)
50 D$="ENTRY ADDRESS:
60 E$="POSITION TAPE
70 F$="FIRST SOURCE
80 G$="LAST
90 H$="FIRST DESTINATION
100 I$="ALL RECORDS SPECIFIED?
 (Y/N)
110 J$="READY CASSETTE TO RECORD
120 K$="THEN PRESS ENTER
130 X=256*PEEK(VARPTR(A$)+2)+PEE
K(VARPTR(A$)+3)
140 FORI=ØTO 174
150 POKEI+&H1ØØØ,PEEK(I+X)
160 IFPEEK(I+X)=Ø THENX=X+8
170 NEXT:IFPEEK(398)=57THENPOKE3
99,174:POKE4ØØ,64:POKE398,126
180 FORI=ØTO596:READL$:L=VAL("&H
"+L$):E=E+L:POKEI+&H1ØAF,L:NEXT
190 FORI=ØTO76:READL$:L=VAL("&H"
+L$):E=E+L:POKEI+&H1E2,L:NEXT
200 IFE<>78Ø82 THENCLS:PRINT"DAT
A ERROR"ELSEPOKE&H9D,17:POKE&H9E
,Ø
210 DATA 5F,3Ø,1,A6,84,26,FA,8D,
1F,84,F,97,7D,8D,12,9A,7D,97,7D
220 DATA 8D,13,84,F,97,7C,8D,6,9
A,7C,97,7C,5D,39,8D,5,48,48,48
230 DATA 48,39,8C,2,DD,27,15,A6,
82,81,3Ø,25,11,81,3A,25,C,81,46
240 DATA 22,9,81,41,25,5,8B,9,81
,4F,39,C6,8,D7,8C,7E,A9,51,7E,A9
250 DATA 28,7E,B9,9C,8D,F8,8D,26
,31,8D,FF,21,3Ø,A8,D6,8D,FØ,8D
260 DATA 1E,8D,12,C6,8,3Ø,1,CE,1
,DA,A6,8Ø,27,36,A7,CØ,5A,26,F7
270 DATA 2Ø,36,8D,D8,C,89,7E,A3,
9Ø,86,A3,8C,86,C3,97,89,39,8D,C6
280 DATA 3Ø,A8,ED,8D,F1,8D,C2,3Ø
,88,DF,8D,BD,2Ø,EB,8D,B9,1F,21
290 DATA 2Ø,DB,8D,E8,96,44,BD,A2
,85,C,89,39,86,2Ø,A7,CØ,5A,26,FB
300 DATA 8D,D7,3Ø,88,18,8D,C3,A6
,1,81,58,26,5,8E,2,2Ø,2Ø,7,17,FF
310 DATA 41,26,E9,9E,7C;BF,1,E5,
BF,2,1E,CE,2,2F,DF,45,C6,31,D7
320 DATA 44,33,8D,1,7F,DF,42,8D,
BF,3Ø,A8,18,8D,B4,A6,1,81,58,26
330 DATA 39,DE,42,CC,Ø,19,ED,C4,
9E,45,ED,84,C6,2Ø,ED,42,DC,19,ED
340 DATA 44,ED,2,DC,1B,ED,46,8D,
A,8D,8,C,44,8D,11,25,72,2Ø,5E,DE
350 DATA 42,33,44,DF,42,DE,45,33
,42,DF,45,39,C,44,86,39;91,44,39
360 DATA 17,FE,DF,26,B5,DE,42,DC
,7C,ED,C4,17,FF,6D,3Ø,A8,25,8D
370 DATA 79,3Ø,13,17,FF,5D,17,FE
,C7,26,EE,DE,42,DC,7C,ED,42,17
380 DATA FF,55,3Ø,A8,2A,8D,61,1F
,21,8D,5D,86,E3,97,89,BD,A3,9Ø
390 DATA 17,FE,A9,26,E8,DE,45,DC
,7C,ED,C4,8D,B6,25,13,8D,A5,BD
400 DATA A9,28,3Ø,A8,3C,8D,3D,BD
,A1,B1,81,59,1Ø,26,FF,62,9E,45
410 DATA 6F,84,6F,1,FE,1,8F,FF,1
,FF,CE,1,E9,FF,1,8F,DC,74,7F,2
420 DATA 45,DE,42,83,Ø,E8,DD,7C,
3Ø,1E,8C,2,2D,27,2E,33,5C,EC,42
430 DATA A3,C4,E3,84,1Ø,93,7C,22
,17,2Ø,EA,7E,B9,9C,8D,FB,17,FE
440 DATA CC,3Ø,A8,74,8D,F3,BD,A1
,B1,81,D,26,F9,39,AE,84,3Ø,1F,BF
450 DATA 2,46,7A,2,45,A,44,BD,A9
,28,17,FE,AA,BD,A7,CA,3Ø,2A,8D
460 DATA D5,BD,A7,E9,BD,A9,28,17
,FE,9A,3Ø,A8,5B,8D,C7,3Ø,8C,6C
470 DATA 9F,42,8E,Ø,F,9F,7C,8E,1
,DA,9F,7E,BD,A7,E5,BD,A7,DB,8E,1
480 DATA 8E,9F,7E,8E,1,BA,9F,7C,
8D,2F,D6,44,CØ,3Ø,D7,44,8E,Ø,1
490 DATA 9F,7C,8D,22,9E,42,AE,84
,9F,7E,CE,1,FF,DF,7C,DE,42,EC,42
500 DATA C3,Ø,1,93,7E,27,F,1Ø,83
,Ø,FF,24,2,D7,7D,8D,2,2Ø,E2,7E
510 DATA A7,F4,Ø,7C,F,7D,8D,F7,9
E,42,3Ø,4,9F,42,A,44,26,CD,BD,A7
520 DATA E9,BE,1,FF,BF,1,8F,16,F
D,FC
530 DATA 2,Ø,Ø,Ø,Ø,1,8E,35,1Ø,B6
,2,45,27,E,FC,2,46,DD,27,DD,23
540 DATA 83,Ø,C8,DD,21,1F,4,8E,Ø
,Ø,BF,1,8F,CE,2,2F,AE,C4,9F,7E
550 DATA BD,A7,F,26,1C,D,7C,2A,F
5,33,42,AE,C4,26,EF,BD,A7.E9,7E
560 DATA Ø,Ø,BD,AD,21,9E,A6,3Ø,4
,9F,A6,7E,AD,CØ,7E,A6,19
```

sette tape, it would be more convenient to record *Link* at the beginning of the reverse side of the tape. Therefore, flip the cassette over, rewind the tape and position it. Then answer the positioning prompt by pressing ENTER.

In accordance with the next prompt, depress the Play and Record buttons of your recorder and then press ENTER. When the recording is finished, the recorder will stop and the initial prompt of *Link* will return.

Now, you can repeat the process to produce a high RAM version of *Link*. Therefore, type the filename *HILINK* and press ENTER. The requested entry address of *HILINK* is the destination entry address which is 3DD4 or 7DD4 depending on whether you have a 16K or 32K RAM, respectively. The first and last source addresses that you must

Listing 2:

```
10 CLS:K=255
20 FORI=ØTO31:POKEI+J+&H4ØØ,K:NE
XT
3Ø K=K-16:J=J+32:IFK>142THEN2Ø
4Ø PRINT@238,"LINK";:PRINT@268,"
EXAMPLE";:K=K+32
5Ø FORI=ØTO31:POKEI+J+&H42Ø,K:NE
XT
6Ø K=K+16:J=J+32:IFK<256THEN5Ø
7Ø FORI=ØTO511:POKEI+&H2AØØ,PEEK
(I+&H4ØØ):NEXT
8Ø PMODE4:PCLS:SCREEN1,1
9Ø CIRCLE(128,96),85
1ØØ PAINT(128,96),1
11Ø FORI=ØTO23:READA$:A=VAL("&H"
+A$):POKEI+&H2DØØ,A:B=B+A:NEXT
12Ø IFB<>3116THENCLS:PRINT@267,"
DATA ERROR":STOP
13Ø DATA BD,B3,ED,DD,44,9E,BA,33
,89,18,Ø,DF,42,A6,84,98,45,A7,8Ø
,9C,42,26,F6,39
```

type are the same as previously, 1000 and 132B. The first destination address is 3CD4 or 7CD4 for a 16K or 32K system, respectively. When you later load *HILINK,* you will not have to use the *CLEAR* command to memory protect it. *HILINK* will be automatically memory protected.

A detailed example will be presented to illustrate how to use *Link*. However, before that presentation, it would be well to determine whether or not you have good recordings of *Link* and *HILINK.* Do *not* use *SKIPF* to make that determination. Use of *SKIPF* on any *Link* produced recording will always yield an I/O Error message. *Link* purposely forces an I/O Error to occur as a means of altering the *CLOADM* command routine. *Link* changes the "hook" that links the ROM and RAM when errors occur. The new hook causes entry to be made to the first loaded record which controls the loading of all succeeding records. The original hook is restored before loading the subsequent records. Hence, those records are checked for I/O Errors as they are loaded.

To test the recordings of *Link* and *HILINK* do the following: turn off your computer and then turn it on again. Type *CLOADM* and press ENTER. Rewind the tape and position it. Finally, depress the Play button. While *Link* is loading, note that the letter 'F' at the top leftmost position of the screen stops blinking. The blinking of 'F' on all *Link* produced recordings will be suspended. The purpose of suspending the blinking of 'F' is to guarantee the unmarred loading of the text screen when you desire to precede the running of the main program with one or more screen messages or pictures. If the recording is good, no I/O Error message will occur. Furthermore, upon the completion of loading, *Link* or *HILINK* will automatically start and the initial prompt will appear on the screen. To exit from *Link* for the *CLOADMing* of *HILINK* press the Reset button. Before you load *HILINK,* note the recorder counter setting for later reference.

If you should happen to have a bad recording of either *Link* or *HILINK, CLOAD* the program *GENLINK* and run it. Then type *EXEC,* press ENTER and repeat the process of recording *Link* and *HILINK* on a new tape.

The programs of listings 2 and 3 are integral parts of the example to illustrate how to use *Link*. Lines 10 through 70 of Listing 2 construct a text screen and transfer its contents to another area of RAM. Lines 80 through 100 produce a simple, high resolution graphics display. The remaining lines of Listing 2 generate a machine language routine and store it in RAM. *Link* will be employed in concatenating and recording the text screen, graphics screen, machine language routine and the BASIC program of Listing 3.

When you have typed the program of Listing 2 correctly, run it. You may wish to save it as a precautionary measure. After running the program of Listing 2, erase it via the *NEW* command. Then type Listing 3.

Line 10 of Listing 3 turns on the previously loaded graphics display. The remaining lines "paint" the display in a variety of colors. The color changes are achieved primarily through the machine language routine called by the *USR* functions of lines 40 and 50. This routine is assumed by the program to have been loaded into the high RAM and automatically memory protected there. The example would be more realistic if the graphics screen had contained an intricate drawing requiring considerable program memory to produce it. In such a case the loading of the completed drawing would result in a significant savings in program memory. Frequently, the saved memory could be put to profitable use in program expansion and improvement.

Do not run the program of Listing 3 when you have finished typing it correctly. Instead refer to the previously noted recorder counter setting in positioning the tape for *CLOADMing HILINK. HILINK* rather than *Link* is used here because *Link* loads into the graphics screen memory area and would therefore ruin the display generated by the program of Listing 2.

The completion of the loading of *HILINK* is signalled by the appearance on the screen of the first prompt. Answer it by typing and entering the filename *EXAMPLE.* Usually the next prompt requires the typing of a hexadecimal address. There is one exception. That occurs when the main program is in BASIC, which is the present situation. In such a case, just press the 'X' key and then ENTER.

The text screen was stored by the program of Listing 2 in the RAM area from 2A00 through 2BFF. Hence, answer the next prompt by typing and entering 2A00. Similarly, type and ENTER 2BFF in response to the last source address prompt. Because you will want the text screen to reside in the usual location, answer the destination address prompt by typing 400 and pressing ENTER.

In order to specify the second record, press the 'N' key in response to the next prompt. The second record is the graphics screen. If you have a cassette-based system, the screen resides at addresses 600 through 1DFF. However, if you have the Disk BASIC ROM connected, the graphics screen is located at addresses E00 through 25FF. Thus, your response to the first source address

Listing 3:

```
1Ø PMODE4:SCREEN1,1
2Ø A=256*PEEK(116)+&HE8:DEFUSR=A
3Ø FORJ=ØTO1
4Ø A=USR(85):GOSUB8Ø
5Ø A=USR(17Ø):GOSUB8Ø:NEXT
6Ø PMODE3:SCREEN1,L:IFL=ØTHENL=1
ELSEL=Ø
7Ø GOTO3Ø
8Ø FORI=ØTO3ØØ:NEXT:RETURN
```

prompt should be the typing and entering of 600 or E00 depending on your system. Likewise, for the last source address prompt, type either 1DFF or 25FF and ENTER. In response to the destination address prompt type and ENTER 600 or E00 for cassette or disk-based systems, respectively.

Press 'N' to permit the specification of the third record. This record is the BASIC program of Listing 3. Typing and entering X will automatically take care of all address specification for you. Actually, an additional record will also be automatically specified. The additional record is only eight bytes long and consists of the vital BASIC program pointers at hexadecimal addresses 19 through 20 (corresponding to decimal addresses 25 through 32).

There is one more record to specify, so once again press 'N' in response to the record's specified prompt. Even though the previous record was numbered three, the present record has been given the number five. The number four record was the eight-byte record automatically specified along with the BASIC program. Record five is the machine language routine generated by the program of Listing 2. It was stored at RAM addresses 2D00 through 2D17. However, it is to be loaded into high RAM at addresses 3FE8 through 3FFF or at 7FE8 through 7FFF depending on whether you have a 16K or 32K RAM, respectively. Therefore, each of the next three prompts should be answered by typing and entering, in order, one of the addresses: 2D00, 2D17 and 3FE8 or 7FE8.

Complete the process by pressing 'Y' and appropriately carrying out the instructions of the final two prompts. In positioning the tape make a note of the counter setting of the recorder for later loading of *EXAMPLE*. The signal that recording is finished is the return of the initial prompt to the screen. You will have a rather long wait for the prompt because of the 6K length of the graphics screen record.

In general, you may specify a maximum of nine records. If one of the specified records is a BASIC program, the most that you may specify is eight records unless the BASIC record is the ninth one specified.

Back to the example, load *EXAM PLE* by means of the *CLOADM* command. You should be quickly greeted with the text screen generated by the program of Listing 2. This screen will remain on display for the time needed to load the other records including the rather lengthy graphics screen. When loading is complete, the BASIC program will automatically start and the graphics screen will replace the text screen. The *USR* called machine language routine will keep changing the colors in the display. To end the program press the BREAK key.

For those with disk systems it is worthwhile interjecting a short note of caution. If you record a tape using *Link* with the disk ROM connected, always load the tape with a connected disk ROM. Likewise, if the tape is recorded with the disk ROM disconnected, always load it with the disk ROM disconnected; otherwise, problems would be likely to occur in the execution of the associated programs.

As was previously mentioned, *Link* can be used as part of a scheme to protect your programs against piracy. There is a simple, yet fairly effective scheme for piracy protecting assembly or machine language programs. The scheme will be illustrated by adding protection to *Link* itself.

With the present unprotected version of *Link*, the Reset button can be pressed to return to the CoCo's command mode in which an *EXEC* command can be employed to gain entry to a preloaded program to analyze *Link*. In the proposed protected version of *Link* every BASIC command will be disabled and will result in an immediate error message when issued.

To add this protection to *Link*, turn your computer off and on again and load *Link*. Then give this version of *Link* the filename *PROLINK*. As you did previously, type and ENTER 1100 for the *Link* entry address. However, before specifying the *Link* program record, you must specify the protection record. It consists of six consecutive zero bytes. Locations 250 through 255 contain such bytes. Therefore, the first and last source addresses are 250 and 255, respectively, in the first record specification. For the first destination address, type 120 and ENTER it. Addresses 120 and 125 are usually stored the number of statements and functions, respectively, in the Color BASIC repertoire of commands. Making those quantities zero tricks the BASIC interpreter into "thinking" that it has an empty vocabulary.

Type 'N' to allow the specification of the *Link* program record. Carry out the remainder of the procedure exactly as you did in the production of the unprotected *Link*.

Some of you who are well versed in assembly language programming and are familiar with the CoCo's memory map may already see a way around this protection scheme. One of the hooks that link the ROM and RAM could be the means of gaining entry to a preloaded program for analyzing *Link*. The occurrence of an error could be made to cause such an entry. Therefore, to make protection more effective you should speeify a second protection record before the *Link* program record. This second record consists of the hooks located at RAM addresses 15E through 18D. There are other hooks but they have already been accounted for in the loader record which is always written on tape without your specifying it. Hence, when you are further protecting a program, 15E, 18D and 15E should be the first source, last source and destination addresses of the second specified record.

BASIC programs cannot be protected in the same manner as assembly or machine language programs. A BASIC program clearly could not run if its commands were disabled. The scheme

BASIC programs cannot be protected in the same manner as assembly or machine language programs. A BASIC program clearly could not run if its commands were disabled. The scheme
for protecting any BASIC program does not disable the BASIC commands during program execution but does so when the program has been stopped by any means.

As in the more effective scheme for piracy protecting assembly language programs, the hooks at addresses 15E through 18D must comprise one protection record. However, one hook address in the record must be changed to point to a short machine language subroutine which forms a second protection record. The subroutine is what controls whether or not BASIC commands are disabled.

The objective of the program of Listing 4 is to generate the two protection records and store them in a convenient place in RAM. For purposes of illustrating the scheme, the area chosen to store the two records ws located at addresses 3000 through 303C. The first address 3000 was assigned in Line 10.

When you protect your own BASIC programs, you should (by appropriately editing Line 10) make the assignment consistent with the memory available to accommodate 61 consecutive RAM locations. Line 20 stores at addresses 3000 through 302F an image of the hooks at 15E through 18D. Lines 30 through 50 along with Line 100 are concerned with generating and storing the short subroutine. The subroutine is stored at

Listing 4:

```
10 CLS:A=&H3000
20 FORI=0TO47:POKEI+A,PEEK(I+&H1
5E):NEXT
30 FORI=0TO10:READD$:D=VAL("&H"+
D$):B=B+D:POKEI+A+48,D:NEXT
40 IFB<>977THENPRINT"DATA ERROR"
:STOP
50 FORI=0TO1:POKEI+A+59,PEEK(I+&
H168):NEXT
60 C=INT(A/256):POKEA+10,C:POKEA
+11,&H30+A-256*C
100 DATA 34,2,96,A6,81,6,25,FE,3
5,2,7E
```

Listing 5:

```
10 CLS:PRINT@226,"TYPE YOUR NAME
 & PRESS ENTER":PRINT@260,"";
20 LINEINPUTA$
30 PRINT@358,"PRESS ENTER TO STO
P"
40 K$=INKEY$:IFK$<>CHR$(13)THEN4
0ELSESTOP
```

addresses 3030 through 303C immediately following the hook record. Line 60 appropriately alters the hook record to provide entry to the short subroutine.

When you have correctly typed the program of Listing 4, save it for future use in protecting BASIC programs.

The protection scheme will be illustrated by applying it to the short example program of Listing 5. Therefore, after running the program of Listing 4, erase it by means of the *NEW* command. Then type Listing 5.

After typing the latter program, *CLOADM Link*. The responses to the *Link* prompts should be consecutively as follows:

```
PROBASIC
X
3000
302F
15E
N
3030
303C
3030
N
X
Y
```

Then appropriately follow the tape positioning and recording prompts. In the positioning process note the recorder counter setting for *PROBASIC*.

To test the protection scheme turn your computer off and then on again. Then load *PROBASIC* using *CLOADM*. When *PROBASIC* is loaded, it should request the typing of your name. The program will then go into a loop. You can stop it by pressing ENTER, BREAK or Reset. Regardless of how you stop *PROBASIC*, typing and entering any BASIC command of your choice will cause the computer to hang up.

Link and the protection schemes were developed for your personal use. If you should wish to employ them commercially, please get in touch with me via THE RAINBOW to discuss mutually agreeable royalty terms.

---

continued from page 33

the number which the Assembler (EDTASM+) has decided is the equivalent of our OPERAND. It is these latter two columns of numbers you are POKEing into the memory locations of the first column when you enter a MACHINE LANGUAGE program from BASIC.

I will not try to explain HEXADECIMAL numbers at this point, but there are a couple of things you need to know. Firstly A number prefixed with a $ symbol indicates a HEXADECIMAL number as opposed to a DECIMAL number when used in your SOURCE CODE listing. Have a look at line 120 and you will see $400. This is the hexadecimal value for the memory location of the top left corner of the screen. Now just to confuse the issue when you enter a hexadecimal value from BASIC you must prefix the number with the characters &H (e.g. &H400 in BASIC is the same as our $400 in ASSEMBLY).

The # symbol you saw in line 110 of our program means that the number following (35) is actual data as opposed to a memory location (address). Note also that I have not included the $ symbol so this value is is a decimal (base 10) value. If you have a look in the table of ASCII CHARACTER CODES on page 280 of GETTING STARTED WITH COLOR BASIC you will see that the symbol generated by the decimal code 35 is a # So I guess we can expect that this is the symbol that will appear in the top corner of our screen (memory location $400). If I had felt so inclined I could have quite happily written line 110 to read:

```
00110 START   LDA  #$23
```

Because I have placed the $ symbol in front of the value the assembler will interpret the 23 as a hexadecimal number which as we can see from our table of ASCII CODES is the same as decimal 35.

Now for those who don't yet have an assembler let's enter our program from BASIC. The only way we can do this is to POKE the MACHINE code values that our assembler has given us into the correct memory locations.

```
10 PCLEAR4:Z=&HE00
20 FOR X=1TO8
30 READ Y
40 POKE Z,Y
50 Z=Z+1:NEXTX
60 DATA &H86,&H23,&HB7,&H04,&H00,
&H7E,&HE0,&H05
70 EXEC &HE00
```

Run this program and lo-and-behold we have our lon awaited # on the screen. You will need to reset th computer to break out of the endless loop in line 130 o our ASSEMBLY program. Now study the BASIC program and th MACHINE LANGUAGE program and try to understand what i happening. Compare the values in the DATA statements i particular. Finally type in this one line program:

```
10 POKE1024,35:GOTO10
```

Have fun with all of this. I hope we are beginning t open your door to the world of ASSEMBLY LANGUAGE. See yo next month.

32K
Disk

PART V

By Colin J. Stearman

It's time we got down to some BASIC cooking and add the code for many of the new commands.

New BASIC Commands

When you add the assembly language in Listing 1 to last month's listing (I will tell you how to do this shortly), it will add the following commands and functions:

COLD

This is a Reset command from the keyboard. When you issue it, any program in memory will be lost and BASIC will be "cold" started. This is useful if you have corrupted BASIC somehow and it performs exactly the same as entering the BASIC command *POKE &H71,0:EXEC&HA027*. The start-up banner will be displayed and the *AUTOEXEC.BAS* file will be run.

WPOKE

This is like *POKE*, but is WORD oriented instead of byte. The syntax is the same as *POKE*, but the value can be anything from zero to 65535. This number is poked into the given address and the next address location.

FAST

Issuing this command puts CoCo into high gear and is exactly the same as *POKE65495,0*. You can run the disk system in the FAST mode if you remove capacitor C85 from the mother board. This is a 220pF capacitor on the "Cartridge Select Signal" at pin 32 socket and ground. A word of warning though: do not attempt any disk input/output while in the FAST mode, because it will surely fail!

SLOW

No prizes for guessing what this one does: it issues the equivalent of *POKE 65494,0* and should be performed whenever a FAST has been issued and disk input/output is required.

XEQ(M)

If you type in *XEQ"GAME"*, it is exactly the same as entering *RUN "GAME"*; in other words the BASIC program *"GAME.BAS"* is retrieved from the disk and run. However, if you enter *XEQM"GAME"*, then the machine code program *"GAME.BIN"* will be loaded from disk and started up. It's equivalent to entering *LOADM"GAME":EXEC*.

AUTO

This "direct only" command automatically generates BASIC program line numbers. If you just enter AUTO then the first line will be 10 and the increment will be 10. If you enter *AUTO 100*, for example, the first line number generated will be 100, with an increment of 10. If you enter *AUTO 4,2* the first line number will be four with an increment of two. To exit the AUTO mode, either press BREAK or ENTER immediately after the line number.

SCAN$

SCAN$ is a function similar to *INKEY$*. Its syntax is the same. However, SCAN$ will wait for a key to be pressed rather than continuing on like *INKEY$*. So, if you have a program Line *100 A$=SCAN$*, the program will wait at Line 100 until a key is pressed, and the key value will be assigned to A$.

DATE$

This string function will return the current date stored in the computer. The format of the date is mm/dd/yy, for example 06/12/84. It is always eight characters long. You can use DATE$ like any other string variable, including assigning it to another string variable with an "equals" statement, or manipulating it with MID$, LEFT$, etc. However, you cannot assign a new string value to it by having it on the left side of an equals sign.

Once this code has been added we can "uncomment" some lines from last month (details below), and the *DIR* command will now pause after the screen fills, awaiting any key to continue. Also, the creation date of each file will be displayed in the directory.

Listing 2 is a BASIC program called *"DATESET.BAS"* which sets the date and also dates any undated files on the disk. Files created before you patched BASIC can be dated this way and also any files created by machine language programs which do not use BASIC to open them. Files will be dated if their date fields in the directory contain $0000 or $FFFF. Files with legitimate dates will not be changed. I have this file on my main editor disk and renamed it *"AUTOEXEC.BAS"* so it runs everytime I start up.

WPEEK

This is the complement of WPOKE and will return the WORD stored at the given address and the next consecutive address. The value returned is in the range zero to 65535. The syntax is the same as for *PEEK*.

Adding The New Functions

Call in last month's listing and make the following changes using the [REF #] given as a locating guide. Remove the commenting asterisk from reference Lines 3 and 5. Then delete reference Lines 12 through 17, 23, 24 and 28. Also, delete the last four lines of last month's listing starting with the line *"ZZLAST EQU *-1"*, as these are in this month's listing.

Now type in the new assembly language code found in Listing 1. Finally, reassemble the result and try it as you did last month's listing. The commands and functions should all work as advertised. If not, double check all your typing or subscribe to RAINBOW ON TAPE!

Coming Next Month

The next installment will be devoted entirely to the construction of the parallel interface and the software to integrate it into BASIC. So clean up the CoCo kitchen and we'll go to it next month.

If you would like the entire *DOS PATCH* program source, along with binary files with and without the parallel port driver for DECB 1.0 and DECB 1.1, just send me a disk (no cassettes please) along with $6 and a stamped, addressed disk mailer. I will load the disk and return it to you promptly. Address this request or any questions to: Colin Stearman, 143 Ash Street, Hopkinton, MA 01748.

Listing 1:

```
>>>UNKNOWN MNEMO---

                0917        OPT   LIS
                0918 ********************************************
                0919 *  PATCH #3 to RSDOS (C)1984 Colin Stearman *
                0920 ********************************************
                0921 *
                0922 ******************************************
                0923 *  "COLD" performs a cold restart
DB56 0F71       0924 COLD   CLR'  $71         RESET COLD FLAG
DB58 7EA027     0925        JMP   $A027       RESTART BASIC
                0926 ***********************
                0927 * "WPOKE" COMMAND
DB5B BDB73D     0928 WPOKE  JSR   $B73D       GET 1ST ARGUMENT # TO FFFF
DB5E 9F2B       0929        STX   $2B         & SAVE TEMPORARILY
DB60 BDB26D     0930        JSR   $B26D       PARSE OVER REQUIRED COMMA
DB63 BDB73D     0931        JSR   $B73D       GET SECOND ARGUMENT
DB66 AF9F002B   0932        STX   [$2B]       DO DOUBLE POKE
DB6A 39         0933        RTS               RETURN TO BASIC
                0934 ************************
                0935 *  "FAST"
                0936 *
DB6B B7FFD7     0937 FAST   STA   65495       SPEED UP PROCESSOR
DB6E 39         0938        RTS
                0939 ********************
                0940 *        "SLOW"
                0941 *
DB6F B7FFD6     0942 SLOW   STA   65494       SLOW DOWN PROCESSOR
DB72 39         0943        RTS
                0944 *******************
                0945 * "XEQ" COMMAND
DB73 814D       0946 XEQ    CMPA  #'M         XEQM?
DB75 2703       0947        BEQ   XEQM        YES
DB77 7EAE75     0948        JMP   $AE75       NO - SAME AS RUN
DB7A BDCEE5     0949 XEQM   JSR   A0021       DO LOADM
DB7D 7FFF40     0950        CLR   $FF40       STOP DRIVE MOTOR
DB80 6E9F009D   0951        JMP   [$9D]       EXEC
                0952 ****************************************
                0953 * "AUTO n,i"
                0954 *
DB84 BDD81B     0955 AUTO   JSR   DIRECT      CURRENT BASIC LINE #
DB87 2668       0956        BNE . SYNERR      SYNTAX ERROR
DB89 CC000A     0957        LDD   #$0A        DEFAULT LINE #
DB8C FD01D1     0958        STD   LINNUM      SAVE IT
DB8F FD01D3     0959        STD   INCNUM      SAVE IT FOR INCREMENT TOO
DB92 9DA5       0960        JSR   <$A5        ANY MORE ON LINE?
DB94 271D       0961        BEQ   NOMORE
DB96 BDB73D     0962        JSR   $B73D       EVALUATE ARGUMENT
DB99 DC52       0963        LDD   <$52        GET IT IN D
DB9B FD01D1     0964        STD   LINNUM      OVERRIDE DEFAULT LINE #
DB9E 9DA5       0965        JSR   <$A5        ANY MORE VALUES?
DBA0 2711       0966        BEQ   NOMORE
DBA2 BDB26D     0967        JSR   $B26D       PARSE COMMA
DBA5 BDB73D     0968        JSR   $B73D       EVALUATE IT
DBA8 DC52       0969        LDD   <$52        GET IT IN D
DBAA 2745       0970        BEQ   SYNERR      CANNOT BE ZERO
DBAC FD01D3     0971        STD   INCNUM      OVERRIDE DEFAULT
DBAF 9DA5       0972        JSR   <$A5        ANY MORE ON LINE?
DBB1 263E       0973        BNE   SYNERR      ERROR IF SO
DBB3 86FF       0974 NOMORE LDA   #$FF        SET UP AUTO FLAG
DBB5 B70149     0975        STA   AUTOFG
DBB8 39         0976        RTS               ALL DONE
                0977 ***********************
                0978 * This is the trap routine to see if in
                0979 * AUTO mode
                0980 *
DBB9 7D0149     0981 INPUT  TST   AUTOFG      AUTO MODE?
DBBC 270C       0982        BEQ   INEXIT
                0983 *******
DBBE FC01D1     0984 DOAUTO LDD   LINNUM      GET LAST LINE NUMBER
DBC1 1083F9FF   0985        CMPD  #$F9FF      TOO HIGH?
DBC5 2304       0986        BLS   NOTHI
DBC7 7F0149     0987        CLR   AUTOFG      RESET FLAG
DBCA 39         0988 INEXIT RTS               RETURN
                0989 *
                0990 *******
DBCB 0F87       0991 NOTHI  CLR   $87         INKEY STORE
DBCD 0F70       0992        CLR   $70         FLAG BUFFER FLUSHED
DBCF EDE4       0993        STD   ,S          D SAVE CURRENT VALUE OVER RETURN
DBD1 F301D3     0994        ADDD  INCNUM      INCREMENT IT
DBD4 FD01D1     0995        STD   LINNUM      AND SAVE IT
DBD7 3506       0996        PULS  D           GET OLD VALUE OFF STACK
DBD9 BDBDCC     0997        JSR   $BDCC       DISPLAY NUMBER
DBDC 8620       0998        LDA   #$20        SPACE
DBDE BDA282     0999        JSR   CHROUT      DISPLAY IT
DBE1 CE03DA     1000        LDU   #$3DA       WHERE CONVERTED # IS
DBE4 BE02DD     1001        LDX   #BASBFR     POINT TO BASIC BUFFER
DBE7 5F         1002        CLRB              SET UP CHARACTER COUNTER
DBE8 A6C0       1003 ILOOP  LDA   ,U+         GET FIRST CHAR
DBEA 2708       1004        BEQ   GOTNUM      GET ALL NUMBERS
DBEC A780       1005        STA   ,X+         MOVE TO BUFFER
DBEE 5C         1006        INCB              COUNTER UP
DBEF 20F7       1007        BRA   ILOOP       CONTINUE
                1008 * JUMP IS HERE SO EVERYONE CAN GET IT WITHOUT
                1009 * LONG BRANCHING
DBF1 7EDA2F     1010 SYNERR JMP   SNERR
                1011 *
DBF4 8620       1012 GOTNUM LDA   #$20        SPACE
DBF6 A780       1013        STA   ,X+         SAVE IT AT BUFFER END
DBF8 5C         1014        INCB              COUNT IT
DBF9 BDA171     1015        JSR   $A171       READ A CHARACTER
DBFC 810D       1016        CMPA  #$0D        RETURN?
DBFE 2704       1017        BEQ   ENDAUT      END AUTO FUNCTION
DC00 8103       1018        CMPA  #$03        BREAK?
DC02 2609       1019        BNE   INDONE      NOT SPECIAL SO EXIT
DC04 7F0149     1020 ENDAUT CLR   AUTOFG      RESET FLAG
DC07 CC0D01     1021        LDD   #$0D01      GET A RETURN IN A, 1 CHR IN B
DC0A BE02DD     1022        LDX   #BASBFR     POINT TO BUFFER START
DC0D 7EA39D     1023 INDONE JMP   $A39D       CONTINUE BASIC LOOP
                1024 *********************
                1025 *       "SCAN"
                1026 *
DC10 9687       1027 SCAN   LDA   $87         HAS A KEY BEEN PRESSED?
DC12 2605       1028        BNE   GOTKEY      YES, RETURN WITH CODE
DC14 BDA1C1     1029 KSCAN  JSR   $A1C1       NO CALL KEY SCAN
DC17 27FB       1030        BEQ   KSCAN       KEEP LOOKING
DC19 7EA56B     1031 GOTKEY JMP   $A56B       RETURN A 1 CHAR. STRING
                1032 ***********************
                1033 *
                1034 *       "DATE$"
                1035 *
DC1C C608       1036 DATE   LDB   #8          CHARACTERS IN MM/DD/YY
DC1E BDB50F     1037        JSR   $B50F       VERIFY SPACE AVLBLE, ALLOCATE
                1038 * X IS RETURNED WITH ADDRESS OF STRING START
DC21 BD03       1039        BSR   DATGET      PUT CURRENT DATE AT 8
```

```
DC23 7EB69B             JMP   $B69B       EXIT VIA STRING$ CODE
          *********
          * DATGET PUTS MM/DD/YY AT ADDRESS IN X BASED UPON
          * VALUE AT DATUM. DATE IS STORED AS FOLLOWS:
          * 15 - 9          8 - 5   4 - Ø
          * YEAR (MOD19ØØ)  MONTH    DAY
DC26 FCØ14E    DATGET LDD   DATUM       GET DATA FOR MONTH
          * ENTER BELOW WITH DATE ALREADY IN D
DC29 34Ø6      DATOUT PSHS  D           SAVE ON STACK
DC2B 44               LSRA              GET UPPER BIT IN CARRY
DC2C 56               RORB              MOVE DOWN
DC2D 54               LSRB              MOVE DOWN
DC2E 54               LSRB              MOVE DOWN
DC2F 54               LSRB              MOVE DOWN
DC3Ø 54               LSRB              MOVE DOWN
DC31 8D16             BSR   DECODE      PUT CHARACTERS IN BUFFER
DC33 862F             LDA   #'/
DC35 A78Ø             STA   ,X+
DC37 E661             LDB   1,S         GET DAY
DC39 C41F             ANDB  #%ØØØ11111  MASK OFF MONTH
DC3B 8DØC             BSR   DECODE
DC3D 862F             LDA   #'/
DC3F A78Ø             STA   ,X+
DC41 E6E4             LDB   ,S          GET UPPER BYTE
DC43 54               LSRB              POSITION YEAR DATA
DC44 8DØ3             BSR   DECODE      GET CHARACTERS IN A,B
DC46 3262             LEAS  2,S         REMOVE DATE FROM STACK
DC48 39               RTS
          *
DC49 4F        DECODE CLRA              SET UP TENS COUNTER
DC4A CØØA      SUBTEN SUBB  #1Ø         REDUCE BY TEN
DC4C 25Ø3             BLO   GOTTEN      EXIT AS WENT NEG
DC4E 4C               INCA              INCREMENT TENS
DC4F 2ØF9             BRA   SUBTEN      CONTINUE SUBTRACTING
          *
DC51 CB3A      GOTTEN ADDB  #1Ø+'Ø      RESTORE UNITS AND
DC53 8B3Ø             ADDA  #'Ø         TENS TO ASCII
DC55 EDB1             STD   ,X++        SAVE IN BUFFER
DC57 39               RTS
          ********************************************
          *        "WPEEK"
          *
          *WPEEK RETURNS 2 BYTES
DC58 BDB74Ø    WPEEK  JSR   $B74Ø       INTEGERIZE PARSED VALUE
DC5B EC84             LDD   ,X          DO DOUBLE PEEK
DC5D DD52      UNSIGN STD   $52
DC5F 7E88ØE           JMP   $88ØE       SEND INSIGNED # TO VARIABLE
          ***************************



DC61           ZZLAST EQU   *-1         last used address value
          *
          * ZZLAST must not be greater than $DFFF for
          * DOS 1.Ø and $DEFF for DOS 1.1.   The latter
          * has the OS-9 Boot program and SWI set routines
          * from $DFØØ to $DF4C
          *
          *
                      OPT   LIS
D994                  END   ADDCOM
     NO ERROR(S) DETECTED
```

```
230 ........ 83
END ...... 200
```

Listing 2:

```
5 '"DATESET.BAS"  LISTING #2 COO
KING WITH COCO- PART 5
1Ø CLEAR 1ØØØ
2Ø 'DATE LOADER
3Ø DIM DAYS(12)
4Ø DATA 31,28,31,3Ø,31,3Ø,31,31,
3Ø,31,3Ø,31
5Ø FOR I=1 TO 12
6Ø READ DAYS(I)
7Ø NEXT
8Ø IF WPEEK(&H14E)<>Ø AND WPEEK(
&H14E)<>&HFFFF THEN 21Ø
9Ø INPUT"DATE(MM,DD,YY)";M,D,Y
1ØØ IF M<Ø OR M>12 THEN 24Ø
11Ø IF Y<Ø THEN 24Ø
12Ø IF D<1 THEN 24Ø
13Ø IF M=2 THEN 16Ø
14Ø IF D>DAYS(M) THEN 24Ø ELSE 1
9Ø
15Ø ' DO FEBRUARY
16Ø IF(INT(Y/4)<>Y/4)AND(D>DAYS(
M))THEN 24Ø
17Ø ' LEAP YEAR
18Ø IF D>29 THEN 24Ø
19Ø DATE =(Y*INT(2^9))+(M*INT(2^
5))+D
2ØØ WPOKE &H14E,DATE
21Ø INPUT"DATE FILES";A$
22Ø IF LEFT$(A$,1)="Y" OR LEFT$(
A$,1)="y"  GOSUB 25Ø
23Ø NEW
24Ø PRINT"ERROR":GOTO9Ø
25Ø ' FILE REDATER
26Ø ' DATES ANY FILES WITH ZERO
OR 255
27Ø ' IN THE DATE FIELD WITH TOD
AYS DATE
28Ø INPUT"DRIVE NO";DR
29Ø PRINT"THESE FILES REDATED WI
TH          ";DATE$
3ØØ IF DR<Ø OR DR>1 THEN 28Ø
31Ø FOR X= 3 TO 11
32Ø DSKI$ DR,17,X,A$,B$
33Ø A$=A$+LEFT$(B$,127)
34Ø FOR N=Ø TO 7
35Ø FILE$=MID$(A$,N*32+1,8)
36Ø EXT$=MID$(A$,N*32+9,3)
37Ø IF ASC(FILE$)=Ø THEN 45Ø
38Ø IF FILE$=STRING$(8,255) THEN
FLAG=1:GOTO46Ø
39Ø MSB=ASC(MID$(A$,N*32+17,1))
4ØØ LSB=ASC(MID$(A$,N*32+18,1))
41Ø IF MSB=Ø AND LSB =Ø THEN 43Ø
42Ø IF MSB<>255 OR LSB<>255 THEN
 45Ø
43Ø MID$(A$,N*32+17,2)=CHR$(PEEK
(&H14E))+CHR$(PEEK(&H14F))
44Ø PRINTFILE$+"."+EXT$
45Ø NEXT N
46Ø B$=RIGHT$(A$,127)
47Ø A$=LEFT$(A$,128)
48Ø DSKO$ DR,17,X,A$,B$
49Ø IF FLAG=1 THEN 51Ø
5ØØ NEXT X
51Ø RETURN
```

GAME 32K ECB

IT'S A MYSTERY

Tony Hallen

Mystery is a learning game designed to test (and exercise) the user's general knowledge of various countries from around the world. The program randomly selects a set of clues relating to the size, major products, demography, topography, etc. of one of five countries. The user must guess the name of the country, and the fewer clues needed, the higher the user scores on that round. After 10 written clues have been presented the map of the country is shown as the final clue.

The program features a partial high - resolution character generator and Hi - Res (PMODE4) maps. The user may try PMODE3 and the high - speed poke (65495,0) to modify the graphics display.

To use Mystery, just run it. The directions are part of the program start-up. Once the clues are displayed on the screen, enter 'G' to make a guess at the country's name or 'N' for the next clue. At the end of each round the start-up prompts are recycled to allow every new player to read the directions. The original version of this program has five additional countries/clue sets. This version is available from: Tony Hallen, 316 S. Jackson St., Rushville, IL 62681.

		2515	236
22	145	3025	221
39	12	3060	226
1040	27	3085	178
2010	145	4010	72
2230	176	END	123

The listing:

```
1 CLS: PRINT@235,"WORKING...
3 PCLEAR4:PMODE4,1:CLEAR5ØØØ
4 GOSUB 4ØØØ
5 DIM CLUE$(4,9), MAP$(4,2), MAR
KER(4), ANS$(4)
7 NOISE$="L1ØØAEFDCGEBAFEGDAO2":
 N1$="L1ØØO+DEFADECCADEGDAECFF":
 N2$="L1ØØO-AGCEDAGFEADGCDEGFO2"
1Ø MAP$(Ø,Ø)="BM174,84M+2,ØM-14,
+18M+6,8D4M-6,3M-6,12M-6,2M-6,9L
24M-14,7M-6,3M-4,-8L2E5L4M-6,-4L
4U6E8H4U3E4H8R4U6M+6.-14E4U3L2U3
L16U4L8
12 MAP$(Ø,1)="U2M+2,-7L4U2H3E6R4
E5R6D2M+26,4R12D1R8M+12,2D2M+22,
6U2M+6,1F3M+14,2M+4,-4M+4,7M-2,+
5M-12,8M-1Ø,5
14 MAP$(Ø,2)="BM84,133M-4,3D1L2M
+2,-3L1ØM+2,-9U9H2U2R2U2L6U2M+1Ø
,-22U8M74,66
2Ø MAP$(1,Ø)="BM138,27M+8,11G5M1
44,42F5M-2,6M+6,-5M+12,6D3M172,5
6M+2Ø,8F9R6D5M21Ø,82M-4,8M-8,6G8
M186,112D6M-6,1ØM-6,11M-24,6G8D6
M-8,9M-4,7M-6,2M118,18ØM-2,-5M-1
Ø,-9L4E13U12M-4,-1
22 MAP$(1,1)="M1Ø2,132U7M+4,-9M-
4,-9L8M-2,-12L6H2M72,9ØU1ØM-12,4
L8M-2,-4L6H5M34,7ØM+6,-9M+1Ø,-5U
6M52,44U4R6U2M-4,2U4R6M+4,-1M+1Ø
,5M+8,-4M78,24M+6,2M+12,-4R2M+2,
4L2D8F2R14M+4,-3M126,35M138,27
29 'AUSTRALIA
3Ø MAP$(2,Ø)="BM162,4ØR2M+2,6D6F
M+2,2R4D1M+6,17R4M+6,3D2M188,86E
3M+4,9F8D12M-4.8D3M-1Ø.12D2G2D6M
-1Ø,3D2L8U1H2G2M152,148U3L4M-2,-
9G2U6G3H2R2U7G8M134,128H4U4L6U2L
1ØM-1Ø,4H1M94,126L1ØM-1Ø,5M-1Ø,-
3H2E4U8M-4,-12H4R2L2M52,96
32 MAP$(2,1)="R4M-2,-7U6M+8,-7M+
16,-2E4M+4,-9D2R4U3F2M94,53M+1Ø,
-5D2F4R4D2R2D2R4H4M116,45R6U2L2U
2M+18,6E2D2M-4,7F2M+12,6D2R6M162
,4Ø
35 'CHINA
36 MAP$(3,Ø)="BM236,32D9M-2,6G3D
1ØG6L4D3L2M-8,1ØL4G4H2U7M188,84L
4D3M+6,3D1R4U2R8D2F1M-6,2G7M2Ø4,
119D4L4D2R1F2D4M-8,16F2L2D1M-18,
14D4H4G2H2D4M156,173D2G2H4U2L8H1
ØM-16.3M112.171H2L2U1H1U1L2U2E2H
3U4H2G2L2M+6,-8U11L4U2L4E1U3M92,
128
38 MAP$(3,1)="G2M-14,4M-12,3G4U4
M-6,-4M-1Ø,-3M24,1Ø3U3E2U6R1E4U4
H2L2M-2,-7M-4,-11L2U2E3M+8,2E2R1
ØM+4,-2U3R2U1ØM+1Ø,2H2U2R2E4M+1Ø
,1U1H2E3M78,3ØM+8,9D11M+14,8M+4,
5F2M+24,6R4M156,67E6U5R8M+12,-8R
6E1H4L4G2H3E1U8M+4,2M+6,-3U3E2U4
E1U1L2U3M194,15
39 MAP$(3,2)="M+1Ø,2M+1Ø,13R4F6M
236,33
4Ø 'CHILE
41 MAP$(4,Ø)="BM12Ø,7U1E2U2D1F2D
5F2DD2R2G4D3R2M+4,11R4D5G6M13Ø,5
1M-6,13D14R2D8G2D4F2D3G2D11G2D11
F2D12F4D1L2D5F2M124,156M-2,4M+8,
1ØE2F2E2M+4,2M144,181R2F2D1L1G2L
4M-12,-7H6U5H2U3E2U2G4U5G2U6E2U3
M112,144U2E2U1E2U6D8R2U19L4D9M11
4,1Ø6
42 MAP$(4,1)="H2U3H2E2M118,79M-2
,-13R2U5L2M+2,-17E2U3H2U28M12Ø,7
1ØØØ 'BEGIN LOADING ARRAY
1Ø2Ø HEADER$="**clues**":TITLE$=
"mystery country":FOOTER$="(N=NE
XT CLUE, G=READY TO GUESS)
1025 FOR T=ØTO4:FORS=ØTO9:READCL
UE$(T,S):NEXTS:READ ANS$(T):NEXT
 T
1030 IF CNTER=5 THEN RUN ELSE CL
S:PRINT@64,"DO YOU WANT"TAB(64)"
INSTRUCTIONS (Y/N)?
1035 A$=INKEY$: IF A$<>"Y" AND A
$<>"N" THEN 1035 ELSE IF A$="N"
THEN 1100
1040 CLS:PRINT@8,TITLE$;TAB(64);
"THIS PROGRAM WILL PRESENT"TAB(3
2)"YOU WITH FACTS OR 'CLUES'"TAB
(32)"CONCERNING A 'MYSTERY COUNT
RY.'
1045 PRINT:PRINT"YOUR JOB IS TO
GUESS THE NAME"TAB(32)"OF THIS C
OUNTRY.  YOUR SCORE"TAB(32)"WILL
 BE LOWER FOR EACH CLUE"TAB(32)"
THAT YOU NEED TO SOLVE THE"TAB(3
2)"'MYSTERY.'
1050 GOSUB2400
1055 PRINT@8,TITLE$:PRINT:PRINT"
AFTER EACH CLUE YOU MAY ASK"TAB(
32)"FOR ANOTHER CLUE BY PRESSING
"TAB(32)"'N' FOR 'NEXT CLUE,' OR
 YOU"TAB(32)"MAY TRY TO GUESS TH
E COUNTRY'S"TAB(32)"NAME BY PRES
SING 'G' FOR"TAB(32)"'GUESS.'
1060 PRINT:PRINT"THE FINAL CLUE
WILL BE AN OUT-"TAB(32)"LINE MAP
 OF THE COUNTRY."TAB(64)"GOOD LU
CK...": GOSUB 2400
1100 FLAG=Ø: CNTER=CNTER+1 'KEEP
 TRACK OF # OF GAMES
1110 CLS:PRINT@10,HEADER$:PRINT@
32,STRING$(32,45);:PRINT@480,FOO
TER$;
1115 COUNTRY=RND(5)-1:IF MARKER(
CO)=1 THEN 1115 ELSE MARKER(CO)=
1
1120 FOR CT=Ø TO 9:PLAY "O"+STR$
(RND(4)+1):PLAY NOISE$:PRINT@(CT
+3)*32,CLUE$(CO,CT)
1125 A$=INKEY$:IFA$<>"N"ANDA$<>"
G"THEN1125
1130 IF A$="N"THEN NEXTCT:GOSUB2
ØØØ
1140 GOSUB2200:GOTO1125
2000 'BEGIN MAP, WRITE MESSAGES
BRANCH  TO GUESS INPUT
2010 COLORØ,1:PCLS:SCREEN 1,1
2015 AA$="THE LAST":DRAW "BM4,10
": GOSUB 4100:AA$="CLUE: ":DRAW"B
M4,20":GOSUB4100
2020 FOR T=Ø TO 2:DRAW MAP$(CO,T
):NEXT T
2025 GOSUB 2500 'PAINT MAP
2026 IF FLAG=2 THEN AA$="THAT'S"
:DRAW"S4BM4,10":GOSUB4100:AA$="I
T!!":DRAW"BM8,20":GOSUB4100:RETU
RN
2030 FLAG=1:FORT=1TO3000:NEXTT '
FLAG=LAST CLUE INDICATOR
2050 GOSUB 2200:RETURN 'INPUT GU
ESS, RETURN
2100 'GIVE ANSWER
2110 CLS:PRINT@64, "SORRY--THE A
NSWER IS";TAB(64);ANS$(CO);".":
PRINT:PRINT
2120 GOTO 2320
2200 'INPUT GUESS
2205 CLS: PRINT@32,"CAREFULLY TY
PE COUNTRY'S NAME";TAB(32);"(SPE
LLING MUST BE EXACT).
2210 PRINT:PRINT:INPUT GUESS$
2215 IF GUESS$=ANS$(CO) THEN 230
Ø
2217 PLAY "O1L30ECDEEDCCDEDDECCE
DDO2
2220 IF FLAG=1 THEN 2100
2225 PRINT:PRINT"NOPE. TRY AGAIN
.";TAB(64);"(PRESS ANY KEY TO RE
TURN)
2230 IF INKEY$=""THEN2230
2235 CLS:PRINT@10,HEADER$:PRINT@
32,STRING$(32,45);:PRINT@480,FOO
TER$;
2245 FOR T=ØTOCT:PRINT@32*(T+3),
CLUE$(CO,T):NEXTT
2250 RETURN
2300 'SCOREBOARD FOR CORRECT ANS
WER
2302 FLAG=2:COLORØ,1:PCLS:SCREEN
1,1:GOSUB 2020
2304 FOR T=1 TO 2: SCREEN 1,Ø:PL
AY N1SE$:SCREEN1,1:PLAY N2SE$:NE
XTT
2310 CLS:PRINT:PRINT:PRINT"YOU G
UESSED IN"CT+1"CLUES";TAB(64);"F
OR A SCORE OF";100-CT*3;"....";T
AB(64);"GOOD JOB!
2320 PRINT:PRINT "TRY ANOTHER GA
ME (Y/N)?"
2330 Z$=INKEY$: IF Z$="Y" THEN 1
030 ELSE IF Z$="N" THEN PRINT:PR
INT"BYE-BYE.":PRINT:END ELSE GOT
O 2330
2400 'PROMPT FOR TURNING PAGE
2410 PRINT:PRINT "PRESS <ENTER>
TO GO ON...":LINEINPUT Z$
2420 CLS
2430 RETURN
2500 'PAINT ROUTINE
2510 IF CO<>5 THENPAINT(122,92),
Ø,Ø
2515 IF CO=4 THEN PAINT(132,176)
,Ø,Ø
2520 IF CO=5 THEN PAINT(152,72),
Ø,Ø:PAINT(128,100),Ø,Ø:PAINT(132
,35),Ø,Ø
```

```
2525 IF CO=7 THEN PAINT(144,87),
0,0:PAINT(148,90),0,0:PAINT(47,9
2),0,0
2530 RETURN
3000 DATA"SIZE OF COLORADO + WYO
MING","1,300 MILES OF COASTLINE"
,"AVG. RAINFALL LESS THAN 20 IN.
","41% OF LAND USED FOR FARMING"
,"     MUCH IRRIGATION USED
3005 DATA"PRINCIPAL PRODUCTS: WI
NE, OLIVES","     VEGETABLES, CIT
RUS FRUIT","     TEXTILES,   FOOT
WEAR"
3010 DATA"RELIGION: MOSTLY ROMAN
 CATHOLIC","3RD LARGEST EUROPEAN
 COUNTRY",SPAIN
3015 DATA "POPULATION: 124,700,0
00","    63% LIVE IN CITIES","ETH
NICS: PORUGUESE, AFRICAN","RELIG
ION: 90% ROMAN CATHOLIC","LARGER
 THAN CONTINENTAL U.S","4,603 MI
LES OF COASTLINE","CLIMATE: TROP
ICAL/SEMI-TROPICAL"
3020 DATA "PORTUGUESE IS OFFICIA
L LANGUAGE","WORLD LEADER IN COF
FEE EXPORTS","LARGEST COUNTRY IN
 S. AMERICA",BRAZIL
3025 DATA "POPULATION: 14,926,80
0","    60% LIVE IN CITIES","ABOU
T THE SIZE OF CONT. U.S.","MUCH
DESERT AND ARID LAND","OFFICIAL
LANGUAGE: ENGLISH","95% OF POP.
IS ENGLISH
3030 DATA "IS A STRONG U.S. ALLY
","YOUNGER THAN U.S. AS A NATION
","PRODUCES MUCH WOOL & MUTTON",
"LOCATED IN SOUTHERN HEMISPHERE"
,AUSTRALIA
3035 DATA "POPULATION: 1,004,000
,000","MOST LIVE ON FARMS","RELI
GION: BUDDHISM, CONFUCIANISM","
  1/10 OF LAND IS CULTIVATED","2
/3 OF LAND DESERT OR MOUNTAINS",
"HAS HIGHEST SPOT IN WORLD","70%
 LITERACY RATE"
3040 DATA "COMMUNIST GOVT.","KNO
WN FOR TEA & SILK PROD.","2ND LA
RGEST COUNTRY IN WORLD",CHINA
3045 DATA "POPULATION: 11,100,00
0","    80% LIVE IN CITIES","SLIG
HTLY LARGER THAN TEXAS","2,650 M
ILES OF COASTLINE","VERY MOUNTAI
NOUS","OFFICIAL LANGUAGE: SPANIS
H","RELIGION: ROMAN CATHOLIC","P
RESIDENT IS HEAD OF GOVT."
3050 DATA "EXPORTS 10% OF WORLD'
S COPPER","LOCATED IN WESTERN HE
MISPHERE",CHILE
3055 DATA "POPULATION: 3,100,000
","    83% LIVE IN CITIES","SIZE
OF COLORADO","HILLY AND MOUNTAIN
OUS","OFFICIAL LANGUAGE: ENGLISH
","84% OF POPULATION IS ENGLISH"
,"99% LITERACY RATE","CHIEF PROD
UCTS: GRAIN, TEXTILES","QUEEN IS
 TITULAR HEAD OF STATE"
3060 DATA "LOCATED IN SOUTHERN H
EMISPHERE",NEW ZEALAND
3065 DATA "POPULATION: 6,343,000
","LANGUAGES: GERMAN, FRENCH","R
ELIGION: ROM. CATH., PROTESTANT"
,"99% LITERACY RATE","2 TIMES TH
E SIZE OF MASS.","MOUNTAINS COVE
R 70% OF LAND","PRESIDENT IS HEA
D OF STATE","PRODUCTS: INSTRUMEN
TS, WATCHES"
3070 DATA"           CHOCOLATE,
CHEESE","            BANKING",SWIT
ZERLAND
3075 DATA "POPULATION 10,000,000
","    34% WORK ON FARMS","OFFICI
AL LANGUAGE: SPANISH","ETHNICS:
NEGRO, SPANISH","96% LITERACY","
SLIGHTLY SMALLER THAN PENN.","2,
500 MILES OF COASTLINE","COMMUNI
ST DICTATORSHIP","PRODUCTS: SUGA
R, TOBACCO"
3080 DATA "A CARIBBEAN COUNTRY",
CUBA
3085 DATA "POPULATION: 700,000,0
00","    22% LIVE IN CITIES","36%
 LITERACY RATE","1/3 THE SIZE OF
 TOTAL U.S.","HAS HIGHEST MOUNT.
 RANGE","VERY DENSELY POPULATED"
,"PRESIDENT IS HEAD OF STATE","P
ARLIAMENTARY GOVERNMENT"
3090 DATA "PRODUCTS: TEXTILES, S
TEEL","          RICE,      GRAIN
S",INDIA
3095 DATA "POPULATION: 69,400,00
0","    65% LIVE IN CITIES","74%
LITERACY RATE","OFFICIAL LANGUAG
E: SPANISH","3 TIMES THE SIZE OF
 TEXAS","45% OF LAND IS ARID","A
VERAGE ALTITUDE: 3,000 FT."
3100 DATA "PRESIDENT IS HEAD OF
GOVT.","PRODUCTS: COTTON, SUGAR
CANE","          COFFEE, RUBBER"
,MEXICO
4000 'CHARACTER DATA
4001 DIM CC$(12)
4002 CC$(0)="U4;E2;F2;D2;NL4;D2;
BM+3,0" 'A
4003 CC$(1)="BM+1,-0;H1;U4;E1;R2
;F1;BM+0,4;G1;L2;BM+6,0" 'C
4004 CC$(2)="NR4;U3;NR2;U3;R4;BM
+3.+6" 'E
4005 CC$(3)="U3;NU3;R4;NU3;D3;BM
+3,0" 'H
```

continued on page 54

THE HOME HURRICANE TRACKING STATION

Wayne Davis Ed Jones
Gene Clifton

Now you can throw away thosr tracking charts you got at the supermarket the other day, your computer has just become an electronic tracking chart.

This program offers two options: projection and position plotting.

By entering the reported latitude and longitude of the hurricane, then inputting the direction of travel, the program will plot the projected course and display it graphically.

In option two, position plotting, it is possible to enter the reported positions (accumilated daily), so that an overall picture can be developed as to the path the hurricane has taken.

This program allows the plots to be saved to disk or tape. These plots can later be reloaded and additional plots can then be added. Just remember to rerecord the new plots on disk or tape.

If this program is being typed by hand, line 60 should be entered as showen, including spaces. Altering the spacing will affect the sound.

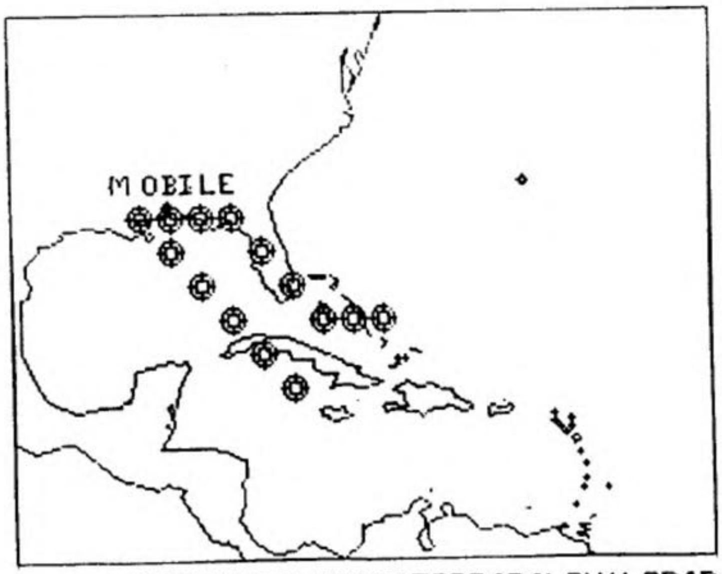
In line 120, option three will reset your computer to a cold start. To prevent the cold start, change POKE113,0:EXEC40999 to END.

As always, save the program to disk or tape before running.

50	75	380	116
110	40	430	197
160	163	530	32
250	97	590	128
310	247	670	122
		END	115

The listing:

```
10 CLEAR1000:PMODE3,1:SCREEN1,1:
PCLS:DRAW"BM45,50C2U14BR8D14BL7B
U7R7BR6BU7D13F1R4E2U12BR7D14U14R
5F2D3G2L5F7BR7U14R5F2D3G2L5F7BR1
4L8R4U14L4R8BR15BD12G2L5H2U10E2R
5F2BR6BU1D13U12E2R3F2D12BL6BU4R4
BD4BR9U14D2F10BD2U14BD14BR5R6L6U
7R4BL4U7R6BD17L120"
20 DRAW"BM123,80U14L4R8BR6D14U14
R5F2D3G2L5F7BR9U12E2R3F2D12BL6BU
4R4BD4BR17BU2G2L4H2U10E2R4F2BR4D
12U14D8E8G7D1F6BR12L6U7R4L4U7R6B
R7D14U14R5F2D3G2L5F7BD3L90":DRAW
"BM30,140C3D6R1E2U1D1F2R1U6D6BR4
```



```
U6R3F1D1G1L2R1D1F2BR4R4L2U6L2R4B
R4R4L2D6"
30 DRAW"BM64,140R4L2D6BR10L4U3R3
L3U3R4BR4D6U6R1D1F4D1R1U6BR15BD6
U6BL1R3F1D1G1L1R1F1D1G1L3BR8BU6D
1F2D3U3E2U1BD6BR5BU1U1BU2U1":DRA
W"BM130,140C4D6R1E2U1D1F2R1U6D6B
R5U1BD1BR7U4E2R1F2D4BL3BU2R2BD2B
R5U1BD1BR11U6R2F2D2G2L2BR8U4E2R1
F2D4BL3BU2R2BD2"
40 DRAW"BM184,140D4F2E2U4BR4R4L2
D6L2R4BR4R4U3L4U3R4BD14BL69H1L2G
2D3F1R3E1BD1BR3U1D1BR14L4U3R3L3U
3R4BD6BR4U1BD1BR17BU5H1L2G2D3F1R
3E1BD1BR4BU6D6R4BR4R4L2U6L2R4BR8
L4D3R3L3D3BR9U6L2R4BR8L4D6R4U6BR
5D6U6R1D1F4D1R1U6D6BR3"
50 DRAW"BM135,173C4L4U6R4BD3BL2L
1BD3BR6U1BD1BR10U6D6R1E2U1D1F2R1
U6D6BR4U1BD1BR10U3D3R4U6BR4D6R4U
6L4BR8D6U6R2D1F4D1U6BR4R4L4D3R3L
3D3R4BR4R4U3L4U3R4":FORX=1TO200:
PMODE3:SCREEN1,0:PMODE4:SCREEN1,
1:NEXT:CLS0:BX=1.8:SO=65312:POKE
65315,63:ST=8:EN=240
60 FORX=ST TO EN STEPBX:UU=UU+1:
IFUU=325THEN70ELSEPOKESO,X:POKES
O,EN-X:NEXT:GOTO60
70 PCLS:FORX=1TO500:NEXT:V=1:DIM
H(100),I(100)
80 V=1:CLS:PRINTSTRING$(32,252);
:PRINTSTRING$(8,128);"path proje
ction";:POKE1068,128:PRINTSTRING
$(9,128);:PRINT"ENTER THE STARTI
NG POINT IN LAT":PRINT"AND LON.
EX: LAT=15.3 LON=75.3. THEN ENTE
R DIRECTION OF TRAVEL. EX: NW OR
 NWW. PRESS ANY KEY TO"
90 PRINT"STOP PROJECTION, THEN A
NY KEY TOSEE THE LOCATION.":PRIN
TSTRING$(32,252);:PRINTSTRING$(7
,128);"location plotting";:POKE1
```

```
327,128:PRINTSTRING$(8,128);:PRI
NT"INDIVIDUAL PLOT LOCATIONS MAY
 BELOADED FROM TAPE OR DISK, OR
MAYBE ENTER AND THEN SAVED."
100 PRINTSTRING$(32,252);:PRINTS
TRING$(9,128);"select 1 or 2 ";:
PRINTSTRING$(11,128);"press 3 to
 end this program";:POKE1516,128
:POKE1520,128:POKE1525,128:POKE1
533,128:POKE1534,128:POKE1535,12
8:POKE1511,128:POKE1512,51:POKE1
513,128:POKE1494,128
110 POKE1493,50:POKE1492,128:POK
E1489,128:POKE1488,49:POKE1487,1
28:SCREEN0,1
120 QQ$=INKEY$:IFQQ$=""THEN120EL
SESOUND180,1:SOUND220,1:IFQQ$<"1
"ORQQ$>"3"THEN120ELSEIFQQ$="2"TH
EN190ELSEIFQQ$="3"THENPOKE113,0:
EXEC40999
130 CLS7:J=1:PRINT@96,STRING$(32
,236);" ENTER LATITUDE (FROM 11
TO 39)":PRINTSTRING$(32,143);:PR
INT@192,STRING$(32,227);:PRINT@1
74,"";:INPUTA:SOUND180,1:SOUND22
0,1:IFA<11ORA>39THEN130
140 PRINT@224,STRING$(32,236);"
ENTER LONGITUDE (FROM 54 TO 95)"
;:PRINTSTRING$(32,143);:PRINTSTR
ING$(32,227);:PRINT@302,"";:INPU
TC:SOUND180,1:SOUND220,1:IFC<54O
RC>95THEN140
150 PRINT@352,STRING$(32,236);"
WHAT IS THE DIRECTION.EX: WNW.":
PRINTSTRING$(32,143);:PRINT@448,
STRING$(32,227);:PRINT@430,"";:I
NPUTB$:SOUND180,1:SOUND220,1:B=(
(42-A)*5.96875):D=((98.5-C)*5.54
37826)
160 IFB$="N"THENX=0:Y=-1:ELSEIFB
$="S"THENX=0:Y=1:ELSEIFB$="W"THE
NX=-1:Y=0:ELSEIFB$="E"THENX=1:Y=
0:ELSEIFB$="NE"THENX=1:Y=-1:ELSE
IFB$="NW"THENX=-1:Y=-1:ELSEIFB$=
"SE"THENX=1:Y=1:ELSEIFB$="SW"THE
NX=-1:Y=1
170 IFB$="ENE"THENX=2:Y=-1:ELSEI
FB$="ESE"THENX=2:Y=1:ELSEIFB$="W
NW"THENX=-2:Y=-1:ELSEIFB$="WSW"T
HENX=-2:Y=1:ELSEIFB$="NNW"THENX=
-1:Y=-2:ELSEIFB$="NNE"THENX=1:Y=
-2:ELSEIFB$="SSW"THENX=-1:Y=2:EL
SEIFB$="SSE"THENX=1:Y=2
180 GOTO270
190 SOUND5,1:CLS8:PRINT@32,STRIN
G$(32,147);" ARE YOU ENTERING NE
W PLOT DATA  OR DO YOU WISH TO A
DD PLOTS TO  OLD RECORDS <ENTER
nEW OR oLD>":PRINTSTRING$(32,156
);
200 NO$=INKEY$:IFNO$=""THEN200E
SEIFNO$<>"N"ANDNO$<>"O"THEN200E
SEIFNO$="O"THEN680
210 CLS:IFV>1THENV=V+1
220 PRINT"    PRESS <ENTER> WHEN
FINSHED":PRINTSTRING$(32,34);:S
UND180,1:SOUND220,1:TA=48
230 TA=TA+16:IFTA>448THENTA=448
240 PRINT@TA+1,V;".  ";:INPUT"LA
";H(V):IFH(V)=0THEN270ELSEIFH(V
<11ORH(V)>41THENPRINT@TA,"     ":
OTO240
250 TA=TA+16:IFTA>458THENTA=464
260 PRINT@TA+3,"LON";:INPUTI(V):
IFI(V)<54ORI(V)>98THENPRINT@TA,
   ":GOTO260:ELSEV=V+1:GOTO230
270 PRINT@0,"     do you want gr
d overlay?":PRINTSTRING$(32,236
;:SOUND5,5:PRINT@0,"     DO YOU
ANT GRID OVERLAY?":SOUND50,5:R$=
INKEY$:IFR$="N"THEN330ELSEIFR$="
Y"THEN280ELSEIFR$<>"N"ORR$<>"Y"
RR$=""THEN270
280 PMODE4,1:SCREEN1,1:COLOR0,1:
PCLS:LN=18:FORLL=1TO9:LINE(LN,0)
-(LN,192),PSET:LN=LN+28:NEXTLL:L
N=11:FORLL=1TO7:LINE(0,LN)-(256,
LN),PSET:LN=LN+30:NEXTLL
290 DRAW"BM11,10R3E1U4H1L2G1D1F1
R2BD3BR7R3E1U1H1L3U3R4BD6BR14R3E
1U4H1L2G1D1F1R2BR10BD3E1U4H1L2G1
D4F1R2BR16H1U1E1R2E1U1H1L2G1D1F1
R2F1D1G1L2BR9R3E1U1H1L3U3R4BD6BR
15H1U1E1R2E1U1H1L2G1D1F1R2F1D1G1
L2BR12E1U4H1L2G1D4F1R2BR15"
300 DRAW "BM123,10E4U2L5BR11BD6R
3E1U1H1L3U3R4BD6BR14E4U2L5BD6BR1
4E1U4H1L2G1D4F1R2BR16R2E1U1H1L2G
1D1F1H1U4E1R2BR7BD6R3E1U1H1L3U3R
4BD6BR15R2E1U1H1L2G1D1F1H1U4E1R2
BR10BD6E1U4H1L2G1D4F1R2BR15R3E1U
1H1L3U3R4BD6BR6R3E1U1H1L3U3R4"
310 DRAW"BM3,20U6L1G2D1R5BD3BR7E
1U4H1L2G1D4F1R2BD20BL10R2E1U1H1E
1U1H1L2G1BD5BR7R3E1U1H1L3U3R4BD3
6BL11R3E1U1H1E1U1H1L2G1BR10BD5E1
U4H1L2G1D4F1R2BD30BL6L4U1E3R1U1H
1L2G1BR7BD5R3E1U1H1L3U3R4"
320 DRAW "BM6,130L4U1E3R1UBH1L2G
1BR11BD4U4H1L2G1D4F1R2BD30BL8U6G
2BD4BR7R3E1U1H1L3U3R4BD36BL8U6G2
BD4BR9E1U4H1L2G1D4F1R2":GOTO340
330 PMODE4,1:SCREEN1,1:COLOR0,1:
PCLS:DRAW"BM0,0R255D191L255U191"
340 DRAW"BM37,62U6R1F2D1U1E2R1D6
BR9E1U4H1L2G1D4F1R2BR5U6L1R3F1D1
G1L1R1F1D1G1L3BR10L4R2U6L2R4BR4D
6R4BR7L4U6R4BD3BL2L1"
350 LINE(136,3)-(135,6),PSET:FOR
LN=1TO180:READLA,LB:LINE-(LA,LB)
```

```
,PSET:NEXT:GOTO42Ø
36Ø DATA136,Ø,13Ø,16,128,13,127,
15,129,2Ø,124,28,126,22,123,18,1
24,12,122,16,122,28,125,34,122,3
5,126,36,124,4Ø,121,4Ø,121,42,11
4,47,113,48,11Ø,49,1ØØ,57,95,64,
94,67,95,71,96,75,1ØØ,84,1ØØ,87,
1Ø1,88,1Ø2,92,1Ø1,97,1ØØ,1ØØ,98,
1Ø1,96,1ØØ,95,97,92,96,91,93
37Ø DATA9Ø,9Ø,87,88,89,85,88,84,
87,85,87,78,84,76,82,73,8Ø,72,77
,73,74,75,72,74,69,71,64,7Ø,62,7
1,58,71,58,67,57,7Ø,48,71,45,71,
48,73,51,72,49,75,52,77,5Ø,79,48
,76,44,78,36,74,34,75,28,74,22,7
5,2Ø,75,18,76,19,77,14,8Ø,1Ø,8Ø,
1Ø,82,5,87,5,91,7,94,6,98
38Ø DATA3,99,3,118,8,128,12,134,
17,137,19,137,22,139,24,139,28,1
38,34,137,39,137,4Ø,135,42,135,4
4,128,44,126,45,124,5Ø,123,53,12
2,58,121,6Ø,122,63,121,64,124,61
,128,58,14Ø,57,136,56,137,57,14Ø
,58,144,56,148,53,152,71,152,77,
153,82,156,84,156,84,162
39Ø DATA82,164,83,168,83,171,81,
177,84,181,86,184,91,188,94,189,
1Ø1,188,1Ø4,185,1Ø6,186,1Ø8,185,
11Ø,187,112,187,119,191,126,185,
126,179,127,18Ø,13Ø,178,133,178,
134,176,139,176,143,175,144,172,
148,17Ø,15Ø,171,151,173,147,175,
149,177,149,18Ø,146,184
4ØØ DATA148,188,151,188,152,187,
15Ø,184,15Ø,18Ø,151,178,157,176,
155,174,156,172,158,172,159,174,
159,176,164,176,169,181,179,18Ø,
185,184,2ØØ,18Ø,2Ø1,181,197,182,
2Ø1,185,2Ø8,187,2Ø8,19Ø,214,191,
82,192,8Ø,186,71,184,6Ø,168,62,1
66,55,164,52,166,38,162
41Ø DATA22,15Ø,18,15Ø,12,152,8,1
53,4,151,Ø,149
42Ø LINE(75,119)-(78,12Ø),PSET:F
ORLN=1TO28:READMA,MB:LINE-(MA,MB
),PSET:NEXT:GOTO44Ø
43Ø DATA81,118,84,118,87,115,92,
115,9Ø,117,95,119,96,118,1ØØ,118
,1Ø1,121,1Ø9,121,1Ø9,124,113,126
,118,126,115,131,119,13Ø,128,131
,134,129,125,126,126,124,121,123
,119,121,99,113,94,113,9Ø,112,83
,113,78,116,78,118,75,119
44Ø LINE(123,141)-(117,143),PSET
:FORLN=1TO9:READNA,NB:LINE-(NA,N
B),PSET:NEXT:GOTO46Ø
45Ø DATA114,142,114,141,112,141,
111,139,114,138,118,139,12Ø,138,
12Ø,14Ø,123,141
46Ø LINE(164,14Ø)-(16Ø,138),PSET
:FORLN=1TO21:READOA,OB:LINE-(OA,
OB),PSET:NEXT:GOTO48Ø
47Ø DATA155,139,153,139,149,142,
147,14Ø,138,139,135,14Ø,134,138,
135,137,144,138,142,136,142,133,
138,132,143,129,148,131,153,13Ø,
156,132,16Ø,132,161,135,166,137,
167,139,164,14Ø
48Ø LINE(18Ø,141)-(176,141),PSET
:FORLN=1TO7:READQA,QB:LINE-(QA,Q
B),PSET:NEXT:GOTO5ØØ
49Ø DATA173,142,173,139,174,138,
178,139,181,138,182,139,18Ø,141
5ØØ LINE(115,1Ø9)-(114,1Ø6),PSET
:FORLN=1TO7:READRA,RB:LINE-(RA,R
B),PSET:NEXT:GOTO52Ø
51Ø DATA111,1Ø4,112,1Ø1,114,1Ø2,
115,1Ø4,115,1Ø6,116,1Ø8,115,1Ø9
52Ø LINE(1Ø8,92)-(114,92),PSET:L
INE-(114,93),PSET:LINE-(1Ø9,93),
PSET:LINE-(1Ø8,92),PSET:LINE(116
,92)-(119,94),PSET:LINE-(117,97)
,PSET:LINE-(116,96),PSET:LINE-(1
18,94),PSET:LINE-(116,92),PSET:L
INE(12Ø,99)-(123,1Ø1),PSET:LINE-
(123,1Ø3),PSET
53Ø LINE(126,1Ø4)-(128,1Ø7),PSET
:LINE(128,11Ø)-(13Ø,113),PSET:LI
NE(134,114)-(136,115),PSET:LINE-
(134,117),PSET:LINE(145,118)-(14
7,118),PSET:LINE-(149,119),PSET:
LINE(137,124)-(141,123),PSET:LIN
E-(139,124),PSET:LINE-(137,124),
PSET
54Ø LINE(2Ø5,149)-(2Ø6,151),PSET
:LINE-(2Ø5,151),PSET:LINE-(2Ø4,1
52),PSET:LINE-(2Ø3,15Ø),PSET:LIN
E-(2Ø5,149),PSET:LINE(2Ø9,178)-(
211,177),PSET
55Ø LINE(2Ø9,18Ø)-(2Ø5,18Ø),PSET
:FORLN=1TO7:READSA,SB:LINE-(SA,S
B),PSET:NEXT:GOTO57Ø
56Ø DATA2Ø9,182,2Ø4,183,2Ø5,184,
2Ø7,184,2Ø8,185,2Ø7,182,2Ø8,18Ø
57Ø FORLN=1TO14:READCR,CS:CIRCLE
(CR,CS),1:NEXT:CIRCLE(187,59),2:
CIRCLE(57,67),2:GOTO59Ø
58Ø DATA143,121,14Ø,121,197,141,
197,144,199,146,2Ø3,143,2Ø3,146,
2Ø1,148,2Ø6,155,2Ø8,159,2Ø8,164,
2Ø7,167,216,167,2Ø4,173
59Ø RESTORE:IFQQ$="1"THENCIRCLE(
D,B),6:SOUND5,1:CIRCLE(D,B),3:CI
RCLE(D,B),6,5:CIRCLE(D,B),3,5:CI
RCLE(D-(5*X),B-(5*Y)),2:CIRCLE(D
-(7*X),B-(7*Y)),1,5:D=D+X:B=B+Y:
A$=INKEY$:IFA$>""THEN63ØELSEIFIN
T(D)<2ORINT(D)>254ORINT(B)<2ORIN
T(B)>19ØTHEN63ØELSE59Ø
6ØØ P=1:IFH(P)=ØTHEN63Ø
```

```
61Ø IFQQ$="2"THENA=H(P):C=I(P):B
=((42-A)*5.96875):D=((98.5-C)*5.
5437826):SOUND5,1:FORPP=1TO4:CIR
CLE(D,B),3,8:CIRCLE(D,B),5,8:CIR
CLE(D,B),3,5:CIRCLE(D,B),5,5:NEX
TPP:CIRCLE(D,B),3,8:CIRCLE(D,B),
5,8
62Ø P=P+1:IFH(P)>ØTHEN61Ø
63Ø A$=INKEY$:IFA$=""THEN63ØELSE
CLS:SOUND18Ø,1:SOUND22Ø,1:PRINT"
   THE HURRICANE LOCATION IS":PR
INTSTRING$(32,156);:F=((B/5.9687
5)-42)*-1:G=((D/5.5437826)-98.5)
*-1:FORX=1TO5ØØ:NEXT:PRINT:IFA=Ø
THENF=Ø:IFC=ØTHENG=Ø
64Ø PRINTSTRING$(32,34);"      LA
TITUDE=";:PRINTF:PRINTSTRING$(32
,34);:SOUND5Ø,1:FORX=1TO5ØØ:NEXT
:FORX=1TO5ØØ:NEXT:PRINT"      LON
GITUDE =";:PRINTG:SOUND5Ø,1:PRIN
TSTRING$(32,34);:FORX=1TO5ØØ:NEX
T
65Ø IFQQ$="1"THENA$=INKEY$:PRINT
@389,"press any key for menu":IF
A$=""THEN65ØELSE8Ø
66Ø FORX=1TO2ØØ:NEXT:SOUND5Ø,1:I
FQQ$="2"THENPRINT@256," DO YOU W
ISH TO SAVE THE PLOTS?
  <yES OR nO>":PRINTSTRING$(32,3
4);:A$=INKEY$:IFA$=""THEN66ØELSE
IFA$<>"Y"ANDA$<>"N"THEN66ØELSEIF
A$="N"THENV=1:GOTO8Ø
67Ø IFV>1THENNO$="N"
68Ø IFNO$="O"THENPRINT@224,STRIN
G$(32,147);"          <tAPE OR  dI
SK?>":PRINTSTRING$(32,156);:TD$=
INKEY$:IFTD$=""THEN68ØELSEIFTD$<
>"D"ANDTD$<>"T"THEN68Ø
69Ø IFNO$="N"THENPRINT@356,"
  <tAPE OR dISK>":PRINTSTRING$(3
2,34);:TD$=INKEY$:IFTD$=""THEN69
ØELSEIFTD$<>"D"ANDTD$<>"T"THEN69
Ø
7ØØ IFNO$="O"THENPRINT@352,STRIN
G$(32,147);:PRINTSTRING$(32,143)
;:PRINTSTRING$(32,156);:SOUND5Ø,
1:PRINT@392,"FILENAME:";:LINEINP
UTFZ$:SOUND5Ø,1
71Ø IFNO$="N"THENPRINT@448,STRIN
G$(32,34);:SOUND5Ø,1:PRINT@424,"
FILENAME:";:LINEINPUTFZ$:SOUND5Ø
,1
72Ø IFTD$="T"THENTD=-1ELSEIFTD$=
"D"THENTD=1
73Ø IFNO$="N"THEN76ØELSECLS7:PRI
NT@192,STRING$(32,147);:PRINT"
     LOADING '";:PRINTFZ$;:PRIN
T"'":PRINTSTRING$(32,156);:OPEN"
I",#TD,FZ$+"/DAT":X=1
74Ø IFEOF(TD)THEN75ØELSEINPUT#TD
,H(X),I(X):X=X+1:GOTO74Ø
75Ø CLOSE:V=X-1:CLS8:PRINT" THES
E '";:PRINTFZ$;:PRINT"' PLOTS LO
ADED":PRINTSTRING$(32,147);:FORQ
X=1TOV:PRINT" ";:PRINTQX;:PRINT"
LAT -";:PRINTH(QX);:PRINT"
LONG -";:PRINTI(QX):FORQZ=1TO4Ø:
NEXTQZ:SOUND22Ø,1:NEXTQX:FORQV=1
TO1ØØØ:NEXT:GOTO21Ø
76Ø V=V-1:CLS8:PRINT@192,STRING$
(32,147);:PRINT" SAVING '";:PRIN
TFZ$;:PRINT"' --";:PRINTV;:PRINT
"PLOTS":PRINTSTRING$(32,156);:OP
EN"O",#TD,FZ$:FORX=1TOV:PRINT#TD
,H(X),I(X):NEXT:CLOSE#TD:GOTO8Ø
```

continued from page 37

```
98":GOSUB17:CIRCLE(1ØØ,76),3Ø,1,
.9,.6,.2
21Ø CIRCLE(82,94),3Ø,3,.9,.58,.1
:CIRCLE(13Ø,72),8Ø,3,.5,.3,.47:P
AINT(82,86),3,3:CIRCLE(82,94),3Ø
,1,.9,.58,.1:CIRCLE(13Ø,72),8Ø,1
,.5,.3,.47
22Ø CIRCLE(194,1Ø4),22,1,.9,.33,
.18:PAINT(194,98),1,1:PT$="V168,
Ø84-21Ø,118":GOSUB17:CIRCLE(194,
1Ø4),22,1,.9,.33,.18:CIRCLE(16Ø,
12Ø),48,1,1,.6,.88:PAINT(16Ø,1ØØ
),2,1
23Ø CIRCLE(16Ø,12Ø),1Ø,1,1,.75,1
:CIRCLE(16Ø,1ØØ),1Ø,1,.9:CIRCLE(
15Ø,9Ø),1Ø,1,.9:CIRCLE(17Ø,86),1
Ø,1,.9:CIRCLE(14Ø,11Ø),1Ø,1,1,.5
,1:CIRCLE(12Ø,11Ø),1Ø,1,.9,.5,1:
CIRCLE(13Ø,1Ø2),1Ø,1,.9,.5,1:CIR
CLE(14Ø,86),1Ø,1,.9,.28,.92
24Ø CIRCLE(154,82),1Ø,1,.9,.53,.
96:CIRCLE(172,11Ø),8,1,1,.15,.75
:DRAW"BM126,9ØC1R6F4D2BF8BR4R4BE
2ØBR6E4BL12BU2U2"
25Ø COLOR1,1:LINE(16,12)-(239,17
9),PSET,B:LINE(16,146)-(66,146),
PSET:LINE(239,146)-(184,146),PSE
T:PAINT(2Ø,148),3,1
26Ø CIRCLE(36,1Ø),8Ø,1,.6,.Ø2,.2
5:DRAW"BM36,56C1D1ØF6"
27Ø CIRCLE(222,1Ø),8Ø,1,.6,.25,.
49:DRAW"BM222,56C1R6D1ØG6D74"
28Ø POKE178,14:PAINT(18,2Ø),,1
29Ø POKE178,26:PAINT(2ØØ,2Ø),,1
3ØØ POKE178,34:PAINT(13Ø,2Ø),,1
31Ø IFINKEY$<>CHR$(13)THEN31Ø
32Ø PMODE3:SCREEN1,1
33Ø IFINKEY$<>CHR$(13)THEN33Ø
34Ø PMODE3:SCREEN1,Ø
35Ø IFINKEY$<>CHR$(13)THEN35Ø
36Ø PMODE4:SCREEN1,Ø
37Ø IFINKEY$<>CHR$(13)THEN37Ø
38Ø PMODE4:SCREEN1,1:GOTO31Ø
```

OS9

by
Bob Thomson

What is available, and which one do we buy? These seem to be the most asked questions, and the hardest to answer. As for what is available, here is a short list:

OS9	BASIC09
RMS	INVENTORY CONTROL
ACCOUNTS PAYABLE	UTILIX HACKERS KIT
SEARCH & RECOVERY	STY(SPELLER)
STY(MAIL MERGE)	DYNASTAR
DYNAFORM	C-COMPILER
PASCAL	ACCOUNTS RECEIVABLE
DYNACALC	O-PACK
DYNASPELL	FILTER KIT
STYLOGRAPH	

and there is more coming through all the time.

So much for what is available, but which do we buy? Well, the best we can do is write a few reviews and hope that they help.

A) OS9

This is an easy one, since we all should have it, anyway. However, there are some short-comings in the manuals. After obtaining the Microware Manuals, I have become more aware of this problem in Tandy manuals. Some firms are now offering manuals only for sale (non-Tandy Manuals).

I noticed in the Microware Manual, there is no mention of the DSAVE command; does this mean that we got something which the big boys pay extra for?

B) BASIC09

Basic09 is an extremely fast enhanced Basic Language system for the 6809. The language is an upward compatable superset of the Standard Basic Language.

Basic09 can also accept and execute most Pascal programs with only minor modifications. It is well suited for a wide range of applications such as Business, Industrial Control, Computer Science, Education, etc.

Basic09 is not just another Basic, but a programming system that has a powerful Text Editor, multi-pass Complier, and run-time interpreter.

Also included is an interactive debugger and system executive. Another feature of Basic09 is that once you PACK a program, it is extremely hard to read, modify or change it.

C) RMS (Record Management System)

Some uses for RMS are: Accounting, Business Record Keeping, Management Information System, Customer or Personel Records, Customised Data Entry, Immediate Data Retrieval, and many situations which require data entry, on-line data retrieval and update, and printed reports.

It is easily customised to fit various other stations or requirements without a large amount of programming knowledge. The data stored under RMS is accessible to user written programs eg, Basic, Pascal, 'C'.

RMS allows the user to determine the format under which the data is to be stored. This format consists of deciding which data items are to be kept, and what the characteristics of each item are to be. The characteristics include, type of data (numeric, alphanumeric, string, date, money-values), the maximum length of the data item, and if necessary, limits or restrictions on the possible values.

Another Nice Feature of OS9

Once again I expound yet another virtue of OS9. With the aid of a very nifty utilty program called SDISK, you can do some very smart things with disk drives. The Tandy drives run very much quieter when you use SDISK to change the step-rate from 30 to 20 ms. It is nice to remove the clatter and grind which we all have been accustomed to when the drive is accessing a disk. The Tandy drives on my system have been running under control of SDISK for four months, at the 20 ms stepping-rate, without any problems.

SDISK can be a lot smarter if you happen to have purchased a TEAC 40 track or 80 track double-sided drive. If your drives are running under SDISK, you can only have a maximum of three drives. However, suppose you have 1 x 40 track, and 2 x 80 track, all double sided, then this gives a magnetic storage capacity of a beautiful 1638K (1.6 megabytes). Compare this with the maximum of 4 Tandy drives, giving a maximum magnetic storage capacity of 717K (0.7 megabytes). Some difference!

Doubling the capacity of your magnetic storage media may not be of any advantage to you. It certainly won't be unless you have unlimited funds to experiment with your favorite hobby. Or more seriously, if you are running serious business software on the CoCo, as I am, then the extra magnetic storage media capacity is a most valuable asset.

The above mentioned combination of TEAC drives was nominated for a special reason. Tandy DOS will not read an 80 track disk (96 tracks per inch), hence your first drive must contain a disk with 48 tracks per inch, and must be formatted with the DSKINI command, so Tandy DOS can read it. However, once your CoCo is fired up, and in the control of OS9, it becomes a lot smarter, and with the help of SDISK software, can read an 80 track disk.

SDISK is incorporated into your Boot-file, the manual which comes with the software gives full easy-to-do instructions.

Stylograph III

This is not a complete review of Stylograph III, but we do promise you one next month.

If you have Stylograpth III, or better still, if you have any version of Stylograph, please, please, please be kind enough to drop me a short scribbled note and let me have comments about your experience with it, giving the particular version number.

I am using Stylograph and VIP Writer as my word processors. While my preference is for the former, I am still using mostly the latter. I think there is a problem with the way I use Stylograph, because I get a lot of "out of memory" problems. There is a high probability that I am doing something wrong, and I find the Manual is difficult to follow.

Stylograph is a very powerful word processor, more so than VIP Writer (which will not run under control of OS9), and it is used on the big machines with a 6809 CPU. That is, Stylograph is used with OS9 level-2, and a special version of it is available for the CoCo.

I am not going to say any more about Stylograph III just now, except that it is a real delight to use. If you have had some experiences, then please drop me a line as soon as possible.

PJB Word-Pack

If you are a serious programmer then Word-Pack is for you. The setting up cost is rather significant, however, if you shop around for your monitor keenly, then you can save about $200, leaving somewhere between $300 and $400 to pay for the monitor and Word-Pack.

You need a monitor, or a high resolution display device, which will clearly display 80 characters per line. The television is just not good enough. I am using an Amber Monochrome Monitor, that is, my display is amber characters on a black background. However, choose carefully because most businesses use a Green Monitor Display, and the reason is that it is supposed to be easier on the eyes. I prefer the amber monitor because I reckon it is easier to read. The choice is yours.

Monitors come with and without sound. The price is not much for either, if you shop carefully. I chose a monitor with swivelling, and height adjustable screen without sound. I could have purchased a monitor with equal resolution with sound, but no swivelling screen, for virtually the same price. In retrospect, I recommend the monitor with sound for that odd moment of idleness when you are tempted to play a game. Games with sound are much more exciting!!

Now the Word-pack. You need a Y-Cable, or a multipack, so you can connect both your Disk Drive Controller and the Word-Pack simultaneously to your CoCo ROM port. The Y-Cable is of course, a lot cheaper, fairly readily available, and is good enough.

The Word-Pack is simular to a ROM-Pack cartridge, and similar size to your Disk Drive Controller. The cable to the monitor plugs into the Word-Pack and not into your CoCo, as does the television. Your CoCo will not drive a monitor, hence the need for the Word-Pack.

There are two Diskettes with the Word-Pack, one is a driver routine for Tandy DOS, the other is a driver routine for OS9 DOS. There are comprehensive and clear instructions and how these routines are to be used. For OS9, the driver routine is incorporated into the boot file, so on booting up, the monitor will burst into action. It is very simple to use, and on my 'cheap' monitor, the display is crystal clear.

Disappointingly, there is a problem with graphics software. For example, I cannot use my VIP Writer, or any other VIP Software which incorporates a high resolution (graphics) screen. However with Basic programs, or with Basic09, Pascal or 'C', the Word-Pack is great, providing there is no graphics.

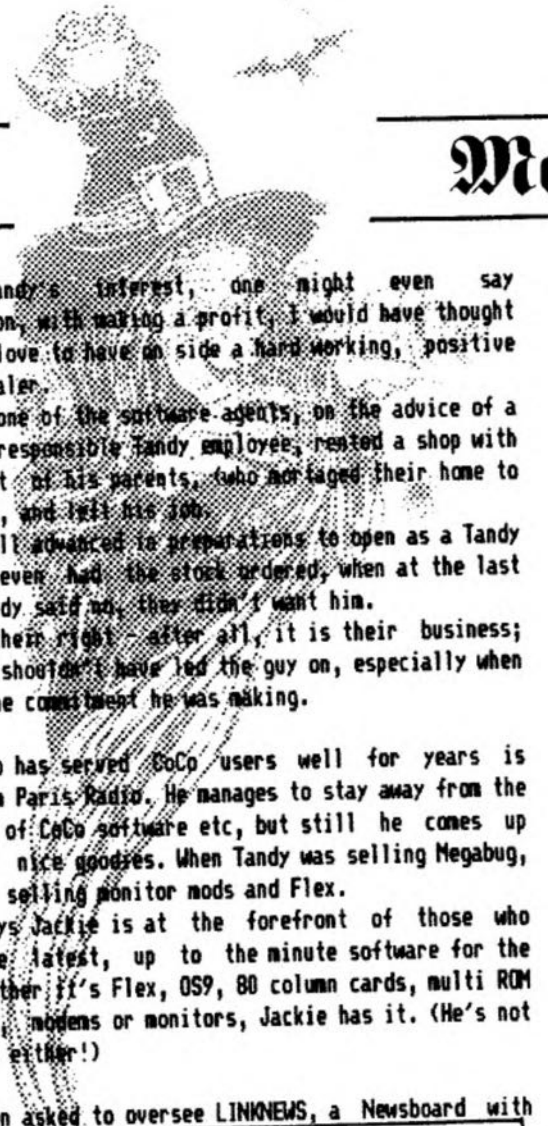
For serious programming, one of the really nice things about the Word-Pack is that you can use the same format for screen display as you use for your 80 column printer. Also, it looks professional.

In finishing I would like to remind you that if you are having trouble with OS9 send a letter to Rainbow or The OS9 User Group (address shown elsewhere). We will try to publish an answer for you. This and future articles will be kept to an easy level for those starting out.

Yours BOB T (get a byte on yourself)

continued from page 48

```
4ØØ6 CC$(4)="BM+1,Ø;R1;NR1;U6;NL
1;R1;BM+4,+6" 'I
4ØØ7 CC$(5)="NU6;R4;U1;BM+3,+1"
'L
4ØØ8 CC$(6)="BM+Ø,-1;F1;R2;E1;U1
;H1;L2;H1;U1;E1;R2;F1;BM+3,+5" '
S
4ØØ9 CC$(7)="BM+2,+Ø;U6;NL2;R2;B
M+3,+6" 'T
4Ø1Ø CC$(8)="BM+Ø, -1;NU5;F1;R2;
E1;U5;BM+3, 6" 'U
4Ø11 CC$(9)="BM+2,+1;U1;BM+Ø,-2;
U5;BM+5,7" '!
4Ø12 CC$(1Ø)="BM+2,-1;U1;BM+Ø,-2
;U1;BM+5,+5" ':
4Ø13 CC$(11)="BM+1,-5;E2;BM+4,+7
" '
4Ø14 CC$(12)="BM+6,Ø" '" "
4Ø15 RETURN
41ØØ 'WRITE 'EM
411Ø FOR XX=1 TO LEN(AA$)
412Ø X$=MID$(AA$,XX,1)
413Ø CC=INSTR(1,"ACEHILSTU!:'",X
$)-1: IF CC<Ø THEN CC=12 'MAKES
BLANKS FROM UNKNOWN CHARS
414Ø DRAW CC$(CC)
415Ø NEXTXX:RETURN
```



Martha Says....

Given Tandy's interest, one might even say preoccupation, with making a profit, I would have thought they would love to have on side a hard working, positive thinking dealer.

Recently one of the software agents, on the advice of a supposedly responsible Tandy employee, rented a shop with the support of his parents, (who mortaged their home to pay for it), and left his job.

He was well advanced in preparations to open as a Tandy dealer - even had the stock ordered, when at the last moment, Tandy said no, they didn't want him.

That is their right - after all, it is their business; but they shouldn't have led the guy on, especially when they saw the commitment he was making.

A guy who has served CoCo users well for years is Jackie from Paris Radio. He manages to stay away from the mainstream of CoCo software etc, but still he comes up with some nice goodies. When Tandy was selling Megabug, Jackie was selling monitor mods and Flex.

These days Jackie is at the forefront of those who supply the latest, up to the minute software for the CoCo - whether it's Flex, OS9, 80 column cards, multi ROM interfaces, modems or monitors, Jackie has it. (He's not a bad sort either!)

I've been asked to oversee LINKNEWS, a Newsboard with (I hope) a difference, for CoCoLink. Although I certainly am interested in any gossip / lies eminating from your group, I also want to get information from you on subjects of national interest. I don't want the sort of stuff Graham was going to put in there - he had (would you believe) a story about the Gold Coast's poor water supply! Who wants to hear about that! No-one! (So it seems! G.)

Maybe you have a good local computer shop; perhaps your meet contact had a baby recently; you could have developed a way to monitor the movement of Melbourne's trams; or you may just wish to embarrass someone - tell me about it and we'll tell everyone.

I believe that there is a letter in the mail to me from Tandy, so I'll wait before showing you the mail that's been going backwards and forwards since I took my CoCo on holidays. In the meantime, the computer has rusted some more, but I've picked out most of the sand, so they can't say that I didn't look after it well.

Blaxland Computer Services supplied us with a really neat amber monitor with a mod for one of the CoCos. I dare say you'll hear more of this from one of the others. You guessed it - they wont let me touch it! So I'll steal their thunder by saying that it looks good, will reduce eyestrain - even with Telewriter, and is just the thing for me. So I'm going to buy my own!

D & L WILSONS

serving melbourne's east

SOFTWARE - Hundreds of titles

HARDWARE - Drives - Printers

SERVICE - Upgrades

bankcard welcome here

No Obligation Demonstrations

doug & louise wilson - 6 stafford st - blackburn -

phone 898-4521

RETURN OF THE **JET-1** **$25**

Just like the movie chase!
Speed bikes through the forrest!
Available only through us.

OS-9
...SOFTWARE ...

S DISK / BOOTFIX	$50
DYNASTAR / FORM	$120
O PAK	$60
SUPER SLEUTH	$70
UTILIX	$70
64K UPGRADES (FITTED)	$90
128K UPGRADES	$260

Blank ROM cases available for ROM if you don't have a disk drive controller.

Also available MR DIS, TIME BANDIT and LUNAR ROVER plus others.

GIVE US A CALL OR WRITE TO
QUEENSLAND COLOUR SOFTWARE SUPPLIES
P.O. BOX 306, CLAYFIELD Q4011

Blaxland Computer Services

(THE COLOR COMPUTER SHOP)

HARDWARE　　OS -- 9　　SOFTWARE

64K RAM UP GRADE KIT....................Posted $89.00
128K RAM UP GRADE KIT..................　$ P.O.A.
FULL INSTRUCTIONS INCLUDED,PLEASE STATE OLD,NEW (GREY) OR OLD,NEW CC--2 WHEN ORDERING

10 x C-10's..............$10　　10 x C-30's.............$15
DISKETTES S.S.D.D 40T per 10.......from $28.50
VIDEO MONITOR DRIVER (with sound output) $27.50
DUAL DOS CARDS........................ $P.O.A.
80/40 DRIVE SWITCH (run your 80T as a 40T) $P.O.A.
MAILING LABELS per 1000 from............$16.00
CASSETTE LABELS per 100..................$4.00
PRINTER PAPER 70 g.s.m. 9½"x 11" per 1000 $22.00
"PRINTER PACKAGE" : BMC -- 80 PRINTER,SERIAL ADAPTOR,
SCREEN DUMP PROGRAMME...................$430.00
(adaptor plugs directly to CO-CO and 9600 Baud selectable)
DISK DRIVES
BARE DRIVES 40/80T D.S.D.D. SLIM LINE.....$P.O.A.
COMPLETE UNITS (ready to run)
1 x 40T D.S.D.D. $700.00
2 x 40T D.S.D.D. $1100.00

bankcard
welcome here

STOCKISTS OF OS--9 and CO-CO SOFTWARE

POSTAGE AND FREIGHT EXTRA (where applicable)

76A MURPHY ST, BLAXLAND. P.O.BOX 125 B/ LAND
PHONE : (047) 39 3903

Casula Hobbies

AUSTRALIAN
MODEL RAILWAY
SPECIALISTS

We stock:

* Australian Rainbow and Australian CoCo magazines.
* CoCoConnection - connect your train layout, burgular alarm, robot or model cars to CoCo.
* Model Railways of Australian, American and European Prototype.

CASULA HOBBIES
P.O. Box 72,
LIVERPOOL. 2179.

SHOP
245 NORTHUMBERLAND ST.,
LIVERPOOL. 2170.

WORLD RIVERS

FEATURES INCLUDE:

* Hi Res Graphics Displays;
* Easy to use - Menu Driven
* Selected Rivers studied in detail (Murray / Darling inc)
* Graphic Quiz / Game with 20 rivers
* Multiple choice test too!

Sysym requirements:
32K + ECB

Phone 02-858-1414 or
write David McNeill
23 Perry St
Dundas. NSW. 2117.

Tape or disk $25.00

TANDY ELECTRONICS DEALER (No. 9320)

1oToADVRSEPT

TANDY COMPUTERS & ACCESSORIES

best prices!

FREE DELIVERY THROUGHOUT AUSTRALIA

90 DAYS WARRANTY

Bankcard & Cheque Orders accepted

BAYNE & TREMBATH

3 Boneo Rd., Rosebud, Victoria 3940

Ph. (059) 86 8288. A/h (059) 85 4947

* For the best deals for CoCo Software and Hardware Give us a call!!!

RAINBOW BITS

supplier of the latest hardware add-ons for CoCo

CoCo MODEM

AUTOMATIC $295 STANDARD $250

(P&P $5) (Reg. Mail $10)

1 Telecom approved to 300 baud bi-directional modem
2 Plugs into Rom port ex expansion pack to work with disk
(Specify Tandy controller or other)
3 Inbuilt terminal programme on Eprom (no loading)
4 Automatic model allows CoCo to dial-up remote computer/B B S
5 Requires no power pack Works with all models of CoCo
6 Leaves printer port free for continuous down load to printer

DISK CONTROLLER (Tandy Compatable)
Improved DOS, Gold Plated Contacts. Has Auto Num. etc.
$200

4-SLOT EXPANSION PACK
Enables disk controller, modem, 80 column card eprom programmer, rom packs etc to be run/selected at one time
$125

CASE CHANGER
LOWER CASE BOARD made and tested $70 (simple installation)
gives true lower case (with descenders) on screen not reverse video

VIDEO AMP CARD $25 (simple installation) for driving video monitor

PHONE (07) 302072, (07) 3690860

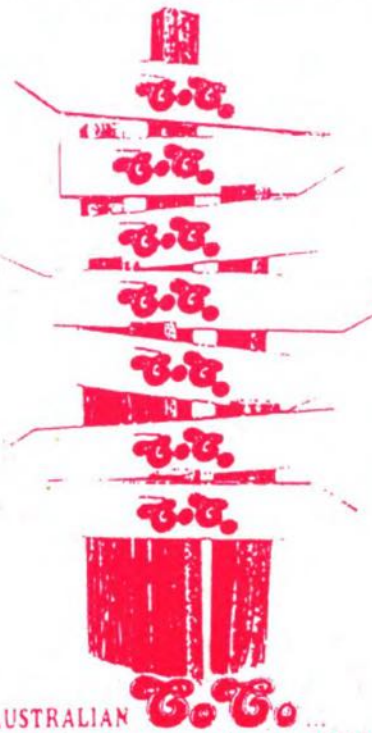
Write to 17 Penley St, The Gap. Q. 4061

RAINBOW BITS

user group CONTACTS

			GOLD COAST	SHERYL BENTICK	075-39-2003	PORT NOARLUNGA	ROB DALZELL	08 386 1647
			GOSFORD	PETER SEIFERT	043 32 7874	PORT PIRIE	KEVIN GOWAN	086 32 1368
' (Stop between numbers = b.h. else			GOULBURN VALLEY	TONY HILLIS	058 59 2251	RINGWOOD	ANDREW RAWLINGS	03 726 6521
' a.h.; but, hyphen between = both.)			GRAFTON	DAVID HULME	066.42.0627	ROCKHAMPTON	KEIRAN SIMPSON	079 28 6162
ADELAIDE	JOHN HAINES	08 278 3560	GREENACRES	BETTY LITTLE	08 261 4003	ROSEVILLE	KEN UZZELL	02 467 1619
ADELAIDE NTH	STAN EISENBERG	08 258 6214	HERVEY BAY	LESLEY HORWOOD	071 22 4989	SALE	BRYAN McHUGH	051 44 4792
ALBURY	RON DUNCAN	060 43 1031	HOBART	BOB DELBOURGO	002 25 3896	SANDGATE	MARK MIGHELL	07 269 5090
ARMIDALE	TOM STUART		IPSWICH	MILTON ROWE	07 281 4059	SEACOMBE HTS	GLENN DAVIS	08 296 7477
BAIRNSDALE	COLIN LEHMANN	051 57 1545	JUNEE	PAUL MALONEY	069 24 1860	SPRINGWOOD	DAVID SEAMONS	047 51 2107
BALLARAT	MARK BEVELANDER	053 32 6733	KENMORE	GRAHAM BUTCHER	07 376 3400	STURT	MARY DAVIS	08 296 7477
BANKSTOWN	KEN HAYWARD	02 759 2227	LEETON	CHRIS NAGLE	069 53 2969	SUNBURY	JACK SMIT	03.744.1355
BLACKTOWN	KEITH GALLAGHER	02-627-4627	LITHGOW	STUART RAYNER	063 51 4214	SUTHERLAND	IAN ANNABEL	02 528 3391
BLAXLAND	BRUCE SULLIVAN	047 39 3903	LIVERPOOL	LEONIE DUGGAN	02-607-3791	SWAN HILL	BARRIE GERRAND	050.32.3838
BOWEN	TONY EVANS	077 86 2220	MACKAY	LEN MALONEY	079511333x782	SYDNEY EAST	BOB JONES	02-331-4621
BRASSALL	BOB UNSWORTH	07 201 8659	MACLEOD	ROBIN ZIUKELIS	03 450211x465	SYDNEY TEENS	ROD HOSKINSON	02 48 5948
BRIGHTON	GLENN DAVIES	08 296 7477	MacQUARIEFIELDS	KIETH ROACH	02 618 2858	TAMWORTH	ROBERT WEBB	067 65 7256
BRISBANE EAST	ROB THOMPSON	07 848 5512	MAFFRA	MAX HUCKERBY	051 45 4315	TAMMOOR	GARY SYLVESTER	046 81 9318
BRISBANE SN	PATRICK SIMONIS	07 209 3177	MAITLAND	LYN DAWSON	049 49 8144	TONGALLA	TONY HILLIS	058 59 2251
BRISBANE SW	GRAHAM BUTCHER	07 376 3400	MARYBOROUGH	NORM WINN	071 21 6638	TOONGABBIE	ARTHUR SLADE	02 622 8940
BRISBANE WEST	BRIAN DOUGAN	07 30 2072	MELBOURNE	JEFF SHEEN	03 528 3724	TOOWOOMBA	MIKE CARTER	076 35 6911
BUNDABERG	JIM McPHERSON	071 72 8329	MELTON	MARIO GERADA	03 743 1323	' BEGIN NTH	DAVID PROUT	076.32.7533
CAMBERWELL	TONY BALDWIN	03 728 3676	MILDURA	SCOTT HEWISON	050 23 6016	' BEGIN STH	LEW GERSEKOWSKI	076 35 8264
CAMPBELLTOWN	LEO GINLEY	02 605 4572	MOE	STEPHEN SEMPLE	051 27 4841	' ADVANCED	GRAHAM BURGESS	076 30 4259
CANBERRA	SHAUN WILSON	062 51 2339	MORPHETTVALE	KEN RICHARDS	08 384 4503	TOWNSVILLE	JOHN O'CALLAGHAN	077 73 2064
CAULFIELD	JEFF SHEEN	03 528 3724	MOREE	ALF BATE	067 52 2465	TRARALGON	MORRIS GRADY	051 66 1331
CHATSWOOD	BILL O'DONNELL	02 411 3336	MORWELL	GEORGE FRANCIS	051 34 5175	UPPER HUNTER	TERRY GRAVOLIN	065 45 1698
CHURCHILL	GEOFF SPOWART	051 22 1389	MT ISA	PAUL BOUCKLEY-SIMONS	077 43 6280	WAGGA WAGGA	BRUCE KING	069 25 3091
COLYTON TEENS	DWAYNE MANSON	02 623 5805	MUDGEE	BRIAN STONE	063-72-1958	WESTLEIGH	ATHALIE SMART	02 848 8830
COOMA	ROSS PRATT	0648 23 065	MURGON	PETER ANGEL	071 68 1628	WHYALLA NORRIE	CHRIS HUNTER	086 45 3395
DANDENONG	DAVID HORROCKS	03 793 5157	NAMBUCCA HDS	WENDY PETERSON	065 68 6723	WOLLONGONG	BRIAN McCAULEY	042 71 4265
DARWIN	BRENTON PRIOR	089.81.7766	NEWCASTLE	LYN DAWSON	049 49 8144	WONTHAGGI	PAT KERMODE	056 74 4583
DENILIQUIN	WAYNE PATTERSON	058 81 3014	NOARLUNGA	ROBBIE DALZELL	08 386 1647	SPECIAL INTEREST GROUPS		
DUBBO	GRAEME CLARKE	068 89 2095	NOWRA	ROY LOPEZ	044 48 7031	BRIZBIZ	BRIAN BERE-STREETER	07 349 4696
EMERALD	LEIGH GAMES	059 68 3392	PARKES	DAVID SMALL	068 62 2682	BRISBANE OS9	JACK FRICKER	07 262 8869
FORSTER	GARY BAILEY	065 54 5029	PENRITH	TOM LENANE	047-31-5303	CARLISLE MiCo	STUART HALL	08 361 1922
FRANKSTON	BOB HAYTER	03.783.9748	PERTH	IAN MACLEOD	09 448 2136	BLAXLAND OS9	BOB THOMSON	047 39 3903
GIPPSLAND STH	PAT KERMODE	056 74 4583	PYMBLE	MARTIN WELLS	02 449 2077	BLAXLAND 128K	BOB THOMSON	047 39 3903
GLADSTONE	ALBERT VAN GORKUM	079 72 2353	PORT MacQUARIE	RON LALOR	065 83 8223	ROCKHAMPTON MiCo	TIM SHANK	079 28 1846

Whichever way you look at it ...



AUSTRALIAN C.C. ...
The only other magazine you need for your TRS 80 Color Computer

THE RAINBOW
Registered by Australia Post-
Registration No. QBG 4009
Graham Morphett.
P.O.Box 1742, Southport. QLD. 4215.

POSTAGE
PAID
AUSTRALIA