For your TANDY Color Computer

Registered by Australia Post: Reg. No. QBG 4009

$3.95

AUSTRALIAN RAINBOW

February, 1985

No.44

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 Telewri:er-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.

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.

G.C.U. GRAPHICOM UTILITY $29.95

- **MULTI DRIVE** - Copy pictures from one disk to another.
- **KILL** - Blank out individual pictures on a pix disk.
- **TRANSFER** - Copy pictures between Graphicom and binary formats.
- **DISPLAY** - View individual pictures.

PICTURE DISKS $12.95

Available from COMPUTIZE

4C	- Artifact color palette
5C	- Large character sets drawn with master design from Derringer Software)
6C	- Same as 5C but set up as stamp set
7	- Miscellaneous Art Set #1
8	- Miscellaneous Art Set #2
9	- Miscellaneous Ads and Examples
10	- Miscellaneous Fonts
11C	- Artifact color palette type fonts
12C	- Art demo from WHITESMITH
13C	- GRAPHICOM PART II function demo

Computer Sofware Shop, 'Tandy Electronics Dealer", Kimberly Park Shopping Village, SHAILER PARK, QLD. Phone (07) 209-7299.
Nrth Qld Colour Software, 9 Durham Court, KIRWAN, TOWNSVILLE, QLD. Phone (077) 73-2064.
Rainbow Valley Comuters, RMB 6680, MAFFRA, VIC. Phone (051) 743-1323.
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.

RAINBOW

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

DEADLINES

Mar	7th Feb '85
April	7th Mar '85
May	7th April '85
June	7th May '85
July	7th June '85

OS-9

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

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

Reg'd Publication QBG 4007

PRINT #-2,

A new year starts.

Thank you to all the folk who sent cards over the Christmas period. Your friendship is valued.

We had a very busy time prior to Christmas. Although we didn't produce a magazine, we used the time to catch up on the Rainbow on Tapes, CoCoOzs, and a myriad of little jobs that had been left undone.

We have since completed MiCoOz, have CoCoLink going, and have started on furnishing the old house we work from with desks that we have designed and built to suit our particular needs.

CoCoLink started with a cough and a flutter, and then finally got underway on 8th January, 1985. Which is not to say that we were slacking. (You may recall that the original intent was to be operational on 1st Dec, 1984.) CoCoLink is the result of 5 months of continued and frantic work by Brian Dougan, Bob Thompson, Warren Warne, Paul Humphries, a friend of Brian's who wants to remain coy, and lastly Kevin, who has worked day and night, learning to cope with the new beast. I thank them all. You will find more on CoCoLink inside.

CoCoConf is also organised ...... finally!

Yes folks, its amazing what a little holiday can do! The venue is not as flash as we would have liked, but it will do admirably.

CoCoConf is attracting attention from a lot of people. You can expect to meet all your favourite suppliers at CoCoConf, perhaps learn something new at one of the tutorials, see the other computers at the Computer Expo, (with your entry to CoCoConf you get a ticket to the Computer Expo), and you will also get to see many of the faces you otherwise only read about. If we have to have Martha there, we'll ensure that there is a good quantity of wide surgical plaster on hand. (If any of the doctors are coming, could you perhaps bring a reserve supply?)

To the new subscribers we say a big welcome. Our magazine exists because others want to share their knowledge with you. We hope you will learn quickly just how

INDEX

friendly and useful the user groups are. If you have a problem, there are people who are willing to help you in your local area. Call your local contact and get to know him. His number is in this magazine.

The 'old' (in this case read 'previous') subscribers and in particular, the advertisers, will want to thank Peggy Annabel for her efforts on their behalf. Peggy was Greg's right hand in advertising matters, and continued to help us when Greg passed away. However Peggy has decided against continuing her agency this year, and I for one will certainly miss her help.

I would appreciate it if Meet Contacts could check to see that their local Tandy store has received a poster from us. There is a square in the bottom right corner of the poster for you to insert your name and phone number.

With the Tandy posters went a set of subscription forms. Thanks go to the Tandy stores for using these forms to introduce our magazine to your customers. Tandy have done a great job this Christmas, they have sold more CoCo's than ever before, and the future of the computer has never looked better.

Notice I said future! If you had bought a Commodore computer this year, (and as usual, the price was enticing enough,) you'd only have a few months before the only ones who want to know about you, Commodore themselves, dropped your computer in favour of the latest incompatible version.

And now the bad news.

You may have noticed the changed price of this magazine. It appears that one or two software resellers in this country have been very slow in paying the yanks their royalties. The problem has gotten worse, because they've been talking amoungst themselves over there, and discovered that payments to them from Australia have been slow across the board. So now all Australian companies dealing in CoCo Software are regarded as pirates by the Americans.

When American companies take the risk to deal with Australian Software companies in the future, they are being very careful to dot their tees etc. In short, because of this, the yanks are making it very hard to deal with them.

This effected our relationship with American Rainbow too.

In our case, they hit us with a retrospective price increase, payable from day 1, and asked for a large sum of money never mentioned before. In fact, if they lived here they would be liable. If we want to go to the international courts, they probably still are liable, but the reality of law is that it only protects the rich, and in any case, we, as CoCo users need their input. At Rainbow, we will.pay their price, and I hope, show them that they have over reacted.

However, as a result, the price of Rainbow has had to rise, even after we faced up to the other hard decision of the Christmas break.

That decision was that considering the number of people not subscribing to GoCo, and considering·the financial discomfort it causes us, GoCo had to go. Effective this issue, the current GoCo subscriptions that we know we have, have been transfered to AUSTRALIAN COCO, where there will be space for input from Model 100 users. If an individual Model 100 user finds that unpalatable, then the sub can be transfered to CoCoLink, a medium more suited to the Model 100.

Plans we had for a bigger more Australian, more informative RAINBOW, have not been shelved altogether. Till we pay the additional moneys to the yanks, we will stay with the current format which is a melding of the Nov size with the Dec/Jan format. In April or May, expect a few pleasant changes.

I'm not telling all about April, but I will tell you that it is likely, due to the kindness of our advertisers in the Blue Mountains of NSW, Blaxland, (and also Bayne & Trembath), that we will be able to supply subscribers only with a set of OS8 chips, free with that issue!

So you can see that the world of the CoCo owner could certainly be a busy one this year. I hope that this year, we at Rainbow can bring you a lot of fun, a goodly slice of new knowledge, and above all, more than a little friendship.

LETTERS

Dear Graham,

I love your magazines, and I have only one small complaint about Australian Rainbow. It is:-

Why don't you put the system requirements at the top of each program, eg 16K ECB or 64K CB etc.

The reason being, I have a CoCo that has been upgraded to 64K, but not extended. I've only had it for four months and am still very green as to what I can do with it. But with the help of Australian CoCo and the Rainbow magazines, plus Pat Kermode I am slowly getting the hang of it all. I would also like to enter my daughter's scores in your scoreboard.

L. O'Meara
Wonthaggi. Vic.

That Kermode name keeps poping up doesn't it!

We try, we really try to put in the system requirements, but I know too that we usually forget at the last moment!

I must say too, that I haven't come across too many 64K non-extended computers or programs, so you are probably the owner of a rare piece of equipment!

Graham.

Dear Graham,

I have only just begun to find out about such words as CoCo, GoCo and MiCo.

It seems that I have a GoCo machine, which, come to think of it is rather apt. Like any owner of a new machine I am keen to find out about it. I bought the Model 100 a few days after Greg Wilson met his untimely death. In my search for information I was told about the GoCo publication and loaned some back issues. As I read I became fond of the man keen to convey information in a friendly style. Yes, Greg was a man with a sincere desire to help newcomers get to know their computer. Being a Christian I have often prayed to asked God to console Greg's dear wife Helga. For if the loss of Greg can be experienced by a total stranger such as myself how much it must be for his dear wife.

To you Graham. I wish the best in your efforts to hold things together. Previously I did'nt know about Coco's and Mico's. I suppose that provided the combined publication can knit together common elements such as general programming practices, review of support hardware such as printers and the like, it could work. Anyway all the best.

John Pollard
Gymea. NSW.

Dear Sir,

I would first like to thank you for a real good magazine I have had many hours of fun playing the games and reading the articles. Some of the things I have read about in the first few magazines were over my head but after a few magazines I look back and find I understand them. I hope you will keep up the good work.

I am starting up a users group in Bunbury which we hope will be under way over Christmas, Jan or Feb.

I have also updated my computer from the MC-10 to the 64K ECB. Thank you again for a great book.

Gordon Giles
Bunbury. Vic.

Dear Gordon,

You didn't give me your phone number for the back of the magazine!

Graham.

Dear Australian Rainbow,

I have found a dangerous command in the CoCo. It is EXECUTE. If it is typed in, the computer hangs up, (no syntax error) so I thought I had better warn the other readers.

Trent McDonald
Aspley. QLD.

Dear Trent,

We'll ask Dr CoCo to tell you why this happens. If he doesn't know, we'll ask Martha.

Graham.

.... Continued on P 42

EDUCATION PAGE

December is the time when the rest of us get jealous of the teachers of this world, as they take off on their beaut long holidays.

The folk at Tandy were busy none the less, as were we, preparing for the new school year.

Tandy are about to release the Model 1000 computer, an IBM PC workalike, which has additional features that make it particularly suited to the school environment, (like joystick ports, light pen ports, and DeskMate, a multi function program which includes a word processor, spreadsheet, filing system, telecomunications, a calendar and electronic mail).

We were working on several projects.

First was the 'Best of CoCoOz' tape. This tape includes in it the best of the Education programs from the early days. Several programs have since become standards in their subject. "Best Of" is available from us or our agents for $10.00.

Second project - the Speech Pack Speller. This program works with the Tandy Speech Pack, and currently holds spelling data from the Queensland Year Two Curriculum. As time allows, we will input data for other years, or you can do it yourself.

Priced at $39.95, this package will provide the motivation for many kids to learn to spell.

Our third project was CoCoLink, our Bulletin Board.

CoCoLink will have an area set aside specifically for discussion of educational topics, and for the down loading of educational programs. We're biased I know, but we would encourage you to add a modem to your school's equipment - it opens a whole new world of enquiry!

We introduced the CoCoConnection this month too! The CoCoConnection allows you to connect CoCo to robots, experimental apparatus, and to models. (It was originally conceived to operate Model Railways.

CoCoConnection has 8 external outputs, and 8 external inputs, to enable what ever you are controling to 'sense' it's environment.

The CoCoConnection Mark I is $180.00, and further details can be found in the centre pages.

Lastly, of course, we have spent a great deal of time preparing for this month's issue of AUSTRALIAN CoCo, which is an Educational issue.

Others have been busy for you too! Silicon Software have introduced several new programs this year. The first to be ready, is the CoCo Grade Book, which quite simply is a data base for teacher. Look for a review of this excellent program in AUSTRALIAN RAINBOW next month. They have also just completed a program which will set up your school's timetables for the year. This program is a real work saver, accomplishing in short time what usually takes a busy Principal a week or more to do.

Holbrook Primary School in NSW did well recently thanks to student Scott Bowler. Scott won a COCO for himself and six for his school in a competition run in conjunction with the Technology Month Celebrations held during November in NSW.

Neville Stone, a chalkie from Jesmond High in Newcastle found himself accepting the Woman's Day Teacher of the Year award. The prize happened to be a Model 4p for Neville, and a Model 4 for the school. Also included was Software to the value of $2600. I hear his students really think he's something! (Wonder what the magic formula is!)

Whilst the Atari computer is presently on contract to the NSW Education Dept, it no longer meets the specification, and so effectively, a space exists for a bright, up and coming computer to fill its place.

If any of you in that state want to get together to try and apply a little pressure on them wot decide about these things, I'd be happy to coordinate your efforts.

I mean, after all, there are any number of reasons why the CoCo should have been on the contract in the first place ... start with the state wide, on the spot back up, the users groups in most major centres, the unrivalled ability of CoCo, and the growing range of syllabus related software.

I suppose one does have to give bureaucrats time to think about these things, but surely CoCo's time MUST be near!

COMPUTERS IN CLASSROOMS

Ken Stewart

(It is especially for teachers like Ken, that we have started this column. Teaching in one teacher schools is often like getting the mushroom treatment, and you remember the mushroom treatment surely. I think the place for a lot of this sort of discussion is the bulletin board, but which ever, if you have experience with computers in the classroom, tell us about it!)

G'day. I teach at a one teacher school (year 1 to 7) in Central Qld and have used my own CoCo in school throughout 1984. The Education page is the best thing Graham has done to Rainbow, and I want to support his work by contributing this article.

Firstly I'll put in a plug for Tandy - the Color Computer is highly under-rated as an educational computer. All we hear about is Apple and BBC, good for sure, but very expensive. In small schools that expense can not be justified. So let's move down to Commodore, Microbee, and Tandy. And I'm sure I'd have fun getting parts, service, or advice for the other two. Microbee is good, but we have Rainbow, (and AUST COCO, G.), User

Groups, Manuals, standard large screen color, and Tandy stores within easy reach. We also have software from several sources for a fraction of the cost you can expect on the expensive machines. So come on Tandy, ADVERTISE! (His words Mike Murray, I didn't put him up to it! G.)

18 months ago there was considerable discussion and debate about whether and how computers should be used in primary schools. Now everyone is on the bandwagon, and for many schools it was a blind leap aboard. Many schools will soon be cluttered with expensive teaching aids that few teachers understand or use effectively. Computers are valuable learning aids which can enrich and enhance learning, so we need teachers who can use them intelligently.

Many people at first thought computers would revolutionise learning and in fact replace the teacher. Many teachers hoped their teaching load would be lightened, and a lot of software is advertised as not needing any computer knowledge on the part of the teacher. Well, sorry to be a party pooper, but that is not what I have found. The teacher is still the most important feature in the classroom. Computers are all about communication, planning, decision making, interaction, using information wisely - so choose software carefully and be prepared to be involved closely.

Here's what I do:

1. Computer awareness - best learnt by doing. The first and really important thing I teach is that a computer is a dumb machine. The most important part of a computer system is the person useing it.

2. Drill and practice - still very useful for revising material in a different way.

3. Simulation and problem solving. This is one of the most important reasons for using a computer in a school as it helps the children learn to think. I use Logo a bit but wish I knew more about it. I also use adventures and simulations a lot, ranging from Lemonade, Oregon, Everest, to Greymoon and Seaquest.

4. Word processing. This is the real thing if you're talking about awareness. The only efficient way for me is to have the children write and proof read rough drafts of stories, which are then typed in by the teacher aide on Scriptsit, saved on tape and later the children load and edit their stories and then print them on the printer. They are always very proud of these finished products.

5. Electronic Blackboard. Years 1 and 2 use the computer for flashcard practice of new words from reading books.

I am still very much a learner and no doubt there are plenty of others who have different ways of doing things, so how about writing in and spreading your knowledge, eg, does anyone use a data base like First Fleet, and how useful is it?

Well, so long for now, and happy computing.

EDUCATION NOTES

16K
ECB

A program to help students seek information

Gathering Information From The CoCo Encyclopedia

By Steve Blyn

This month's article continues the thoughts begun last month about information gathering. Today's students are required to gather more information than in the past but are fortunate to have many additional places to gather this information.

One of the services of CompuServe is Grolier's Encyclopedia. Entering *GO AAE* will get you to this CompuServe feature. The students may call this service and let CompuServe search the encyclopedia database for the topic needed. The various places in the encyclopedia that have information on the topic needed will be displayed and the student may choose to view any or all of these sections.

This type of electronic search is fast and efficient. It should not, however, replace completely the ability to look up in a standard encyclopedia one's own information. It is similar to the advent of inexpensive calculators. Even though everyone can afford to own a calculator, it is still necessary to know how to do most of the computations on your own.

Looking up information in an encyclopedia would be a good deal easier if they all consisted of 26 volumes, one for each letter. Of course, this is impractical. Encyclopedias have fewer than 26 volumes and combine information on several letters. There are fewer topics that begin with the letter X than with the letter A. Letter A topics may cover an entire volume on their own. The information that begins with the letter X, however, is almost always combined with the other letters surrounding it. Volume 12, for example, may contain information that begins with the letters W, X, Y, and Z.

Volume 12 might, therefore, appear as 12 W-Z. It is sometimes confusing to newcomers as to where the information for X and Y is. It must be clearly explained to students that they may have to search for where their initial letter is contained.

It's often even more confusing to students to decide which letter to look up in the first place. This requires both thought and practice. The Color Computer, for example , might be listed in a computer encyclopedia under C for computer, T for Tandy, M for microcomputers, or even M for Motorola — the developer of the 6809 chip. A student must learn to think of the various possibilities where the information sought might be contained and then narrow down the choices to the most logical few.

Names are located by the first letter of the last name. Lonnie Falk, for example, would be found in the volume containing F's. The gold rush, however, would be found under the G's rather than the R's. Cars present an even greater challenge as they are usually found under the A's for automobiles.

Fast sports cars would also be found under A's for automobiles. This is not apparent to many students. They might very well waste a lot of time searching in the F, S and C volumes for this information. They should be encouraged to first reason out the most logical choices of where to find their information.

This month's program draws an encyclopedia set and then presents a topic to be looked up. The student's task is to press the number of the volume that would most likely contain information on that topic.

We have included 10 topics for the children to consider. Please consider this a starter set of questions rather than a finished set. Either delete and replace our questions or add to the existing questions. A thinking skill such as what is being considered here cannot possibly be mastered by a student with any given set of 10 or 20 questions. It should rather be a skill that builds up to more and more difficult questions to be answered.

Lines 60-430 contain the strings to draw the letters and numbers needed. Lines 470-720 draw the encyclopedia set. Lines 730-830 ask the question "Which book contains information about . . . " Line 810 sends the program to 950 to select from one of the 10 given questions. Lines 880 and 890 decide if the answer is right or wrong and print the appropriate message.

Press ENTER and the program returns to Line 440 to check the counter and give the next question. If the counter indicates that five questions have been done, then a report card is shown on Lines 1080-1110. You may continue or end the program at this point.

The lines that are user modifiable are Lines 970-1060. They contain the questions and answers. There are three parts to each line. QQ$ represents the questions. Each letter of the alphabet stands for its picture. Thus, SAM is represented by S$+A$+M$. AN is the correct volume number. RA$ represents the picture for the correct number. The numbers go from N1$-N9$. These are the lines that we encourage you to modify or add to for your own purposes. Of course, the R value on Line 950 should correspond to the number of questions used.

We encourage you to alter our programs in any way that may help your children or students. As a side benefit to altering programs written by others, you also increase your own programming skills.

Before leaving this month, we would like to pass on some educational news. The College Board Association has decided to include a programming exam in its Advanced Placement Test Schedule. These are tests that high school seniors can take for advanced placement in college. The programming language that they have chosen for the test is PASCAL, rather than BASIC.

This means that to receive advanced college credit, our high school students will need to become proficient in PASCAL. At Computer Island, we have been using the PASCAL version for the Color Computer offered by Deft Systems Inc . We find it easy to get started and use. The manuals are extensive, clear and complete. We feel that high school students would obtain a very good background in PASCAL by using this product.

260	197
550	46
780	206
970	216
END	54

The listing:

```
10 REM"ENCYCLOPEDIA"
20 REM"STEVE BLYN,COMPUTER ISALN
D,NY,1984
30 RS=RND(-TIMER)
40 CLEAR2000
50 PCLS:PMODE3,1:SCREEN1,1
60 REM"THE LETTERS AND NUMBERS"
70 A$="BEHUNU2R4NU2DGL2BGBL6"
80 B$="BEHENR3HER3D4L3BGBL6"
90 C$="BU4ER2FD2GL2HBG2BL4"
100 D$="BEHU2ER3D4L3BGBL6"
110 E$="BER3U2NL2U2L4BG5BL2"
120 F$="BUR4U2NL3U2BG5BL5"
130 G$="BUR4U3HL2GDRBG3BL4"
140 H$="BUU2NU2R4NU2D2BGBL9"
150 I$="BR2BUU4BU2BD7BL8"
160 J$="BUU3ER2FDBG3BL7"
170 K$="BUE2NH2R2ND2U2BG5BL5"
180 L$="BU5R4D4BGBL9"
190 M$="BUNU4E2F2U4BG5BL5"
200 N$="BUU4F4U4BG5BL5"
210 O$="BEHU2ER2FD2GL2BGBL6"
220 P$="BER3U2NU2L3GNFBG2BL4"
230 Q$="BEHU2ERNDNURFD2GL2BGBL6"
240 R$="BEHERNH2R2NU2D2L3BGBL6"
250 S$="BU2FR2EHL2HER2FBG4BL6"
260 T$="BUR2NU4R2BDBL10"
270 U$="BUU3ER2FD3BGBL9"
280 V$="BUU2E2F2D2BGBL9"
290 W$="BUU4F2E2D4BGBL9"
300 X$="BUE2NH2NE2F2BGBL9"
310 Y$="BUE2NU2F2BGBL9"
320 Z$="BUNR4E4L4BG4BDBL2"
330 N1$="BE2NU3DEBFBGBL9"
340 N2$="BENR3HER3U2L4BG5BL"
350 N3$="BENR3HENR2HER3BG5BL5"
360 N4$="BENU4E3L4BG4BL2"
370 N5$="BER4U2L3HER3BG5BL5"
380 N6$="BU2FR2EU2NHGL2HER2BG5BL
4"
390 N7$="BUNR4UE3BG5BL4"
400 N8$="BER2EHEHL2GFNR2GFBGBL6"
410 N9$="BER2EHL2GNFU2ER2FBG4BL6
"
```

```
420 SP$="BE4BUBG5BL3":REM"SPACE"
430 LN$="L4":REM"DASH"
440 PCLS:PMODE3,1:SCREEN1,1
450 CT=CT+1:REM"THE COUNTER"
460 IF CT>5THEN 1080
470 COLOR6:FOR T=10 TO 230 STEP
30
480 LINE(T,20)-(T+20,60),PSET,B
490 NEXT T
500 COLOR7:FOR T= 10 TO 230 STEP
 30
510 LINE(T+4,25)-(T+16,30),PSET,
BF
520 NEXTT
530 LINE(0,70)-(255,73),PSET,BF
540 DRAW"C6A2S8BM75,4"+R$+E$+F$+
E$+R$+E$+N$+C$+E$
550 REM"DRAW THE LETTERS ON THE
BOOKS"
560 DRAW"A2C6S4BM16,52"+A$+SP$+B
$
570 DRAW"BM46,52"+C$+SP$+E$
580 DRAW"BM76,52"+F$+SP$+H$
590 DRAW"BM106,52"+I$+SP$+L$
600 DRAW"BM136,52"+M$+SP$+P$
610 DRAW"BM166,52"+Q$+SP$+S$
620 DRAW"BM196,52"+T$+SP$+V$
630 DRAW"BM226,52"+W$+SP$+Z$
640 REM"DRAW VOLUME NUMBERS"
650 DRAW"C8S8BM25,35"+N1$
660 DRAW"BM55,35"+N2$
670 DRAW"BM85,35"+N3$
680 DRAW"BM115,35"+N4$
690 DRAW"BM145,35"+N5$
700 DRAW"BM175,35"+N6$
710 DRAW"BM205,35"+N7$
720 DRAW"BM235,35"+N8$
730 REM"DRAW-WHICH BOOK CONTAINS
 INFORMATION ABOUT"
740 A1$=W$+H$+I$+C$+H$+SP$+SP$
750 A2$=B$+O$+O$+K$+SP$+SP$
760 A3$=C$+O$+N$+T$+A$+I$+N$+S$+
SP$+SP$
770 A4$=I$+N$+F$+O$+R$+M$+A$+T$+
I$+O$+N$+SP$+SP$
780 A5$=A$+B$+O$+U$+T$
790 DRAW"S8C6BM20,80"+A1$:DRAW+A
2$:DRAW+A3$
800 DRAW"BM30,100"+A4$:DRAW+A5$
810 GOSUB 950
820 DRAW "BM40,120"+QQ$+SP$+SP$
830 DRAW LN$
840 AN$=INKEY$
850 IF AN$="1" THENDRAW N1$ ELSE
 IF AN$="2" THENDRAW N2$ ELSE IF
 AN$="3" THENDRAW N3$ ELSE IF AN
$="4" THENDRAW N4$ ELSE IF AN$="
5" THENDRAW N5$ ELSE IF AN$="6"
THENDRAW N6$ ELSE IF AN$="7" THE
NDRAW N7$ ELSE IF AN$="8" THENDR
AW N8$ ELSE 840
860 REM" REACT TO THE STUDENT'S
ANSWER"
870 DRAW"BM50,140"+SP$
880 IF VAL(AN$)=AN THEN DRAW+C$+
O$+R$+R$+E$+C$+T$:SOUND230,3:RT=
RT+1
890 IF VAL(AN$)<>AN THEN DRAW+A$
+N$+S$+W$+E$+R$+SP$+I$+S$:DRAW+S
P$+SP$+RA$:SOUND75,3
900 FORT=1 TO 200:NEXT T
910 DRAW"S4BM65,162"+P$+R$+E$+S$
+S$+SP$:DRAW+E$+N$+T$+E$+R$+SP$:
DRAW+T$+O$+SP$+G$+O$+SP$+O$+N$
920 LINE(50,160)-(180,170),PSET,
B
930 AN$=INKEY$
940 IF AN$="" THEN 930 ELSE 440
950 R=RND(10)
960 REM"QQ$ IS THE QUESTION AND
AN AND RA$ REPRESENT THE CORRECT
 ANSWER"
970 IF R=1 THEN QQ$=R$+U$+S$+S$+
I$+A$:AN=7:RA$=N7$:REM"UNION OF
SOVIET SOCIALIST REPUBLICS"
980 IF R=2 THEN QQ$=C$+A$+R$+S$:
AN=1:RA$=N1$:REM"AUTOMOBILE"
990 IF R=3 THEN QQ$=S$+A$+M$+SP$
+F$+I$+N$+K$:AN=3:RA$=N3$
1000 IF R=4 THEN QQ$=R$+E$+D$+SP
$+T$+R$+A$+I$+N$+S$:AN=7:RA$=N7$
1010 IF R=5 THEN QQ$=P$+O$+O$+D$
+L$+E$+S$:AN=2:RA$=N2$:REM"DOGS"
1020 IF R=6 THEN QQ$=B$+O$+B$+SP
$+L$+E$+E$+SP$+J$+R$:AN=4:RA$=N4
$:REM"LEE"
1030 IF R=7 THEN QQ$=R$+O$+C$+K$
+S$:AN=3:RA$=N3$:REM"GEOLOGY"
1040 IF R=8 THEN QQ$=P$+E$+R$+S$
+I$+A$+N$+SP$+C$+A$+T$+S$:AN=2:R
A$=N2$
1050 IF R=9 THEN QQ$=G$+O$+L$+D$
+SP$+R$+U$+S$+H$:AN=3:RA$=N3$
1060 IF R=10 THEN QQ$=M$+I$+A$+M
$+I$:AN=3:RA$=N3$:REM"FLORIDA"
1070 RETURN
1080 CLS:PRINT@10,"REPORT  CARD"
;
1090 PRINT@128,"YOU DID ";RT;"QU
ESTIONS CORRECTLY."
1100 IF RT=5 THEN PRINT@170,"EXC
ELLENT";
1110 PRINT@324,"PRESS <ENTER> TO
 GO ON";:PRINT@360,"OR 'Q' TO QU
IT.";
1120 EN$=INKEY$
1130 IF EN$=CHR$(13) THEN RUN EL
SE IF EN$="Q" THEN 1140 ELSE 112
0
1140 CLS:END
```

Are Computers Producing Unrealistic Expectations?

By Michael Plog, Ph.D

I recently saw an advertisement on television that bothered me a great deal. A concerned mother was discussing her child with a teacher. The child was not doing well at all in school. For the first half of the advertisement, I thought it might have been produced by the Mormons. (One of those extremely well done "get in touch with your child" spots.) Then, out of nowhere, the teacher suggests to the mother that a computer might help the child with her work. The next scene shows a young girl waiting at home. She is very unhappy, waiting for the results of the parent/teacher conference to determine her future. Mother comes in and says to the girl that they are going to buy a computer. A change comes over the girl; she smiles and hugs her mother, content that she will now be a success in school.

The ad bothered me a lot. It is true that the Pennsylvania state education department conducted a study and concluded that using computers in the classroom improves student learning and contributes to teacher efficiency. Also, a study from New York University found that a significant number of home computers were bought with a primary purpose of education. The interest in educational uses of computers is growing as the evidence of effectiveness mounts. What bothered me about the advertisement is the development of unrealistic expectations. A computer at home is not a guarantee of school success.

Return for a moment to the study conducted by New York University. The principal researcher of this study was Joseph Giacquinta, professor of educational sociology. For three months, doctoral students observed 20 families in the New York area. They compiled about 2,000 pages of log reports. That is an impressive amount of information for a case study, and this study may be the first in-depth look at use and effects of microcomputers in the home.

The primary activity for the families using microcomputers for education was programming, or learning how to program. A "distant second" activity was word processing — school papers or class notes. Only a few families used educational software prepared by professionals to learn school subjects and skills.

The researchers of this study found that parents believed programming would make a person more logical or rational. Also, programming was being stressed in the schools their children attended.

Unfortunately, learning to program a computer does not make people more logical. It simple makes them better programmers. So far as I know, there is no evidence that programming skills transfer to other (i.e., logical) skills; any more than learning Latin makes a person more disciplined and logical. (Are you old enough to remember that assumption by educators?)

If students are learning programming in their schools, then practicing on a home computer can do nothing but help their school work — but only that portion of their school work related to programming computers. Learning to program a computer will not improve a student's knowledge of history (or biology, or geography, or literature).

Please do not misunderstand me. I personally favor students learning computer programming. It is a skill that can be useful for many purposes throughout life. I know teachers who reduce their homework (yes, teachers have homework also) by putting student records on computers. Many service clubs and associations now have members who keep records on a microcomputer. Untold churches keep massive amounts of member information on a microcomputer. An uncountable number of professionals use their home computers for office work. And with all this, we have not even touched on the self-improvement possibilities of computer uses in hobbies and personal interests. But, learning to program a computer will not make a person a better reader — unless, of course, the programming results in software for reading.

Well, why are these parents in New York not buying more professionally prepared educational software? The researchers found the parents thought the materials were inadequate. Other factors were also noted by the researchers. Parents may not be aware of existing programs; they (parents) may lack the skill to evaluate programs; the cost of educational software may be too high; and parents may not know how the programs could be used by their children.

The preliminary findings from this study indicate that parents see the computer as an important educational tool for their children. The families, however, need additional help in understanding how the computer can be used for learning.

The parents in this study (as parents everywhere, probably) wanted to improve achievement at school and help

.... Continued on P 12

We mean business...

If you think of CoCo as a games machine, it will come as something of a shock to you to learn that many use CoCo in their daily business lives. It will surprise you further to learn that CoCo can be optioned up, to the point where it becomes a real contender as a business machine.

If you have the inclination, you can install an 80 column card, change your keyboard, add two double sided 80 track drives, install either a monochrome or RGB monitor, add memory, buy OS-9, Stylograph 3, RMS, and several other business packages. You will then have a system equal to, if not better than, most of the current small business machines. Think about it. Could be a lot of fun learning how to set up a business system like that!

Since we started this column, we have heard from many of you who have CoCo as a hobby computer, but also use it in some way in your business.

It is apparent that there are several types of program that you want, but of them all, the invoicing program, and a good data base are the ones you most avidly seek.

Although we gave you a data base last month, we intend to describe yet another this month. Firstly, we do this because the disk version of my invoice program is incomplete - so you can't have it yet!

Secondly, we think you may find it of interest to see the program which runs the Rainbow data base.

Thirdly, if this program is compared with last month's, the changes can be seen readily, and perhaps an understanding of program development can be gained.

ACS3 is designed to work with four or five other programs. It is linked specifically to one called GT3, which will appear next month.

ACS3 resides along with all other working daily programs at Rainbow, on disk 3, the last disk on a four disk system.

The files that ACS3 creates reside on disk 0 and 2, the opposite sides of the same disk in our system.

Six fields are available, we use E$ to store your renew dates, (Greg used only 5 characters for this job, we have to use 25!), N$, S$, T$, and P$ are used for Name, Street, Town, and Postcode, respectively.

The length of these is determined by experience, and as Greg was the master of data handling, we accept his formula as being the best compromise between attempting to get as many details as possible on the disk, and saving as much disk space as possible.

Note that no State indicator is necessary, the program analyzes the postcode to determine the State, thus saving disc space.

Z$ holds your charge card information, or in the absence of that, your phone number, or other notes that may be needed from time to time.

The big advantage of having your own data base, as against one of the 'package' data bases, is that generally, your own will execute faster, (as it doesn't have to be 'all things to all men'), and of course you can have it do exactly what you want it to do, when you are ready to do it.

Ultimately, the reason that we like this data base so much is that it does things our way, and with few exceptions, without a lot of fuss.

There is little point in walking you through the various processes of this data base, unless you already have it typed in ... and then you won't need to be walked through! So we'll leave that discussion there and turn to a couple of points of interest.

The first is the inclusion of a 'help' page to assist when entering subscription data.

The second is the occasional ie errors and similar. We (touch wood) have not had a crashed disk with this program, but we do have problems at times when we have to switch around between disks a lot.

The biggest problem occurs after the program is resaved to disk 3. When we RUN after such a save, the whole system hangs up.

It's not something that worries me greatly, after all, all that is required is a cold start, but I've no doubt that if the boffins in a group like the Chatswood group, or the Telecom group in Melbourne, or the Perth boys put their heads together, they'd probably solve it in one!

One final note. As this is a working program, under constant development, and at times, deletion, there are several references to systems that no longer work or are required. This could be called sloppy programming, and Bob Thomson probably will do just that. I prefer to think that as a compromise between getting a job done, and keeping the thing tidy so that later changes can be made with all speed (time being money), that it is realistic to have things in menus that are either under development, or are in the process of total deletion.

In this case, I specifically refer to the 'NEXT' command in the edit mode, which I have not been happy with. I want to increment the page by using this command, but haven't finished the job!

The second major item that does not work is the Alphabetical Sort in the Print Menu. As I require space for other things I am slowly deleting this function. The sort is very slow, and when one considers that we don't need it in the first place, it makes sense to be rid of it!

For 64k system with 4 disk drives.

THE LISTING:

```
1 '**ACS3***MAINTAINS ACCOUNTS**
   ******************FILES********
   **COPYRIGHT BY G. MORPHETT***
   **********7/01/85************
2 CLEAR1500:GOTO20
3 SAVE"ACS3/BAS:3":DIR3:PRINTFRE
E(3):PRINT"PRESS ANY KEY";
4 I$=INKEY$:IFI$=""THEN4
20 CLS0
75 CLS0:PRINT@40,"ACS3";
80 PRINT@128,"P";:PRINT@131," PR
INT A/CS FILE ";:PRINT@192,"E";:
PRINT@195,"  EDIT A/CS FILE ";:P
RINT@320,"S";:PRINT@323,"  SPECI
AL LABEL  ";:PRINT@256,"A";:PRIN
T@259," ADD TO ACS FILE ";
81 PRINT@384,"F";:PRINT@387,"
F I N I S H   ";:PRINT@458,"INKE
Y CODE";
90 I$=INKEY$:IFI$=""THENPRINT@45
8,"              ";:PRINT@458,"INKEY
 CODE";:GOTO90
100 CLS0:IFI$="P"THEN800 ELSE IF
 I$="E"THEN200 ELSE IF I$="A"THE
N 300 ELSE IF I$="F"THEN885 ELSE
 IF I$="S"THEN330
110 GOTO75
200 PRINT@19,"LINE #";:INPUT L:G
OSUB1000:GET#2,L
205 CLS0:PRINT@19,"LINE #";LL;:P
RINT@32,"F";:PRINT@34,"  F I N I
 S H  ";:PRINT@96,"X";:PRINT@98,
"     NEXT      ";
210 PRINT@128,"L";:PRINT@130,"
 L A B E L   ";:PRINT@160,"E";:P
RINT@162,E$;:PRINT@192,"N";:PRIN
T@194,N$;:PRINT@224,"S";:PRINT@2
26,S$;:PRINT@256,"T";:PRINT@258,
T$;:PRINT@288,"P";:PRINT@290,P$;
215 PRINT@320,"B";:IFLEFT$(Z$,3)
="496"THENPRINT@322,LEFT$(Z$,3);
"-";MID$(Z$,4,2);"-";MID$(Z$,6,3
);"-";RIGHT$(Z$,6);:GOTO217
216 PRINT@322,Z$;
217 PRINT@384,"H";:PRINT@386,"
 HELP    ";:PRINT@460,"INKEY CODE
";
220 I$=INKEY$:IFI$=""THEN220 ELS
E IFI$="F"THEN GOTO1050 ELSE IFI
$="L"THEN890
223 IFI$="H"THENGOSUB5000:GOTO20
5
225 'IFI$="X"THENPUT#2,L:GET#2,L
+1:LL=LL+1:GOTO205
230 PRINT@352,I$;:INPUTX$
235 IFX$=""THEN205
240 IFI$="E"THEN LSETE$=X$
241 IFI$="N"THENX=2:GOSUB950:LSE
TN$=X$
242 IFI$="S"THENX=2:GOSUB960:LSE
TS$=X$
243 IFI$="T"THEN LSETT$=X$
244 IFI$="B"THEN RSETZ$=X$
245 IFI$="P"THEN LSETP$=X$
250 GOTO205
300 CLS0:GOSUB3000:IFKK=1 GOTO 2
05 ELSE L=1451:GOSUB1000:L=LOF(2
)+1451:PRINT@32,"LINE #";L;
320 PRINT@77," ";:PRINT@64,"EXPI
RY";:INPUTA1$:IFLEN(A1$)>25 THEN
320
325 IFLEFT$(A1$,3)="///"THENL=LO
F(2):GOTO75
330 PRINT@121," ";:PRINT@96," NA
ME ";:INPUTA2$:IFLEN(A2$)>17 THE
N330
340 PRINT@128,"STREET";:PRINT@18
9," ";:PRINT@159," ";:INPUTA3$:I
FLEN(A3$)>27THEN340
350 PRINT@217," ";:PRINT@192," T
OWN ";:INPUTA4$:IFLEN(A4$)>17THE
N350

360 PRINT@235," ";:PRINT@224,"PC
ODE ";:INPUTA5$:IFLEN(A5$)>4THEN
360
365 IFI$="S"THEN375
370 PRINT@268," ";:PRINT@256,"BC
ARD ";:INPUTA6$:IFLEN(A6$)>14THE
N370
375 IFI$="S"THENN$=A2$:S$=A3$:T$
=A4$:P$=A5$:GOSUB894:GOTO75
380 LSETE$=A1$:LSETN$=A2$:LSETS$
=A3$:LSETT$=A4$:RSETP$=A5$:LSETZ
$=A6$:PUT#2,L
390 GOTO75
790 GOTO885
800 CLS0:PRINT@64,"N";:PRINT@67,
"    NUMERICAL ORDER ";:PRINT@12
8,"A";:PRINT@131," ALPHABETICAL
ORDER ";:PRINT@192,"P";:PRINT@19
5,"    POST CODE ORDER ";:PRINT@
426,"INPUT CODE";
810 I$=INKEY$:IFI$=""THEN810 ELS
E IFI$="A"THEN860 ELSE IFI$="P"T
HEN841 ELSE IFI$<>"N"THEN800
820 CLS0:INPUT"FIRST #";B:PRINT#
-2,CHR$(27)"Q";" AC/S ADDITIONS"
:L=B:LL=B:IFB>1450THENB=B-1450
825 GOSUB1000:L=LL-1:FORI=B TO14
50:L=L+1:GET#2,I:IFLEFT$(N$,1)=C
HR$(255) THEN830 ELSE GOSUB1100:
PRINT#-2,L;TAB(5);E$" ";N$;S$;T$
;ST$;" ";P$;" ";Z$:PP=PP+1:IFPP=
60THENFORY=1TO6:PRINT#-2:NEXTY:P
P=0
830 NEXTI:CLOSE:IFL=1450THENB=1:
L=1451:GOTO825
835 GOTO75
```

```
841 CLS0:LL=0:L=1:PRINT@8,"POST
CODE SEARCH";:PRINT@64,;:INPUT"L
OW NUMBER";LN:INPUT"HIGH NUMBER"
;HN:GOSUB1000
842 FORT=1TO2:FORL=1TO1450:LL=LL
+1
843 GET#2,LL:IFVAL(P$)=>LN AND V
AL(P$)<=HN THEN GOSUB850
844 NEXTL:L=1451:CLOSE:GOSUB1000
:NEXTT
845 I$=INKEY$:IFI$=""THEN845 ELS
E 80
850 PRINTLL;E$:PRINTN$:PRINTS$:P
RINTT$:GOSUB1100:PRINTST$;" ";P$
:PRINT:RETURN
860 FORI=1TO LOF(2):READL:GET#2,
L
870 PRINT#-2,E$;" ";N$;L;TAB(29)
;S$;T$;Z$;" ";P$;CHR$(10)
880 NEXTI:GOTO75
885 CLOSE#2:RUN"GT3:3":END
890 PUT#2,L:PRINT
894 GOSUB1100:LINEINPUT"ENCLOSE
";OS$
895 A$=E$
900 PRINT#-2,CHR$(27)"P";LL;A$
905 PRINT#-2,OS$:PRINT#-2
910 PRINT#-2,N$
920 PRINT#-2,S$
930 PRINT#-2,CHR$(27)"X";T$;:IFI
$="S"THENPRINT#-2,"           ";
931 PRINT#-2,ST$;" ";P$;CHR$(27)
"Y"
935 E$=A$:CLS0:FORTT=1TO3:PRINT#
-2:NEXT:IFI$="S"THENRETURN ELSE
GOTO205
940 MID$(X$,X,1)=CHR$(ASC(MID$(X
$,X,1))+32):RETURN
950 IFMID$(X$,X,1)="^"THEN962 EL
SE IFMID$(X$,X,1)=" "THEN RETURN
 ELSE IFMID$(X$,X,1)>"."THENGOSU
B940 ELSE X=X+1
951 X=X+1:IFX>LEN(X$)THEN230 ELS
E GOTO950
960 IFMID$(X$,X,1)="^"THEN962 EL
SE IFMID$(X$,X,1)=" "THENX=X+1:G
OTO961 ELSE IF ASC(MID$(X$,X,1))
<65 THEN961 ELSE GOSUB940
961 X=X+1:IFX>LEN(X$)THENRETURN
ELSE GOTO960
962 PRINT@351," ";:INPUTX$:RETUR
N
1000 LL=L:IFL>1450THENOPEN"D",#2
,"ACS2/DAT:2",103:L=L-1450:GOTO1
020
1010 OPEN"D",#2,"ACS2/DAT",103
1020 FIELD#2,25AS E$,17AS N$,26A
S S$,17AS T$,4AS P$,14AS Z$
1030 RETURN
1050 PUT#2,L:CLOSE#2:GOTO75
1100 ST$="":IFLEFT$(P$,1)="2"THE
NST$="NSW"
1105 IFLEFT$(P$,2)="26"THENST$="
ACT"
1110 IFLEFT$(P$,1)="3"THENST$="V
IC"
1120 IFLEFT$(P$,1)="4"THENST$="Q
LD"
1130 IFLEFT$(P$,1)="5"THENST$="
SA"
1140 IFLEFT$(P$,1)="6"THENST$="
WA"
1150 IFLEFT$(P$,1)="7"THENST$="T
AS"
1160 RETURN
3000 L=1451:GOSUB1000:LL=1450
3001 'L=1:GOSUB1000:LL=0
3002 FORL=1TO1450:LL=LL+1:GET#2,
L:IFLEFT$(P$,1)="9" OR LEFT$(N$,
1)=CHR$(255)THEN3020
3010 NEXT:CLOSE#2:RETURN
3020 KK=1:RETURN
5000 CLS0:PRINT@13,"CODES";
5010 PRINT@32,"8: RAINBOW":PRINT
"A: R/BOW, C/M & COCOOZ":PRINT"M
: MEET CONTACT - RAINBOW":PRINT"
S: MEET CONTACT - R/G, C/M & OZ"
:PRINT"I: COCO":PRINT"B: BOTH (R
/BOW & C/M)":PRINT"C: COCOOZ"
5015 PRINT"T: ROT - CHARGE EACH
MONTH":PRINT"R: ROT - PAID":PRIN
T"X: AMERICAN ROT":PRINT"Z: MICO
OZ":PRINT"L: COCOLINK"
5020 PRINT@460,"ANY KEY";
5030 I$=INKEY$:IFI$=""THEN5030 E
LSE CLS0:RETURN
```

From P 9 ...

their children compete against others at school, college, and at work. Parents feared, however, that their children would get "hooked on computers" and neglect other important interests.

Mr. Giacquinta and his graduate students will continue working with the families in the study, and hope to expand the research to a regional study, involving about 45 schools and hundreds of families. I hope they obtain the necessary funds for this research. The educational community needs the type of information uncovered by this research. We need the information to plan and develop computer curriculum, to understand what is happening when students interact with machines, and to help build the educational experiences that will be required in the next century.

Let's go back for a bit now, to the advertisement noted at the beginning of this article. The problem is unrealistic expectations. Computer salespeople hold up hardware and software as the cure for everything from stupidity to warts. When these claims are discounted by the general public, the danger is that real benefits of computers for education will also be discounted. We need to know what the computer will do, and what it will not do. The process of finding out what can and cannot be done by computers is education. Let's keep on getting educated.

RESCUE ON ALPHA II

Program by Steven C. Mitchell

With a flick of your wrist,you snap the button on the video and the bureau chief of the Galactic Secret Service's image fades abruptly from the screen. This new assignment from the Service is a troubling one. It seems that every sleazy character in the galaxy is trying to make a grab for power these days, but this latest attempt appears more serious than most. The bureaucrats at the government palace on Centra are certainly in a state of panic over it. But then, those people think they're under attack every time a solar flare extends farther than normal toward their planet.

It seems that some character named Zarcon, one of the last of the cosmic wizards,is masterminding this new threat to the security and peace of the galazy. And if memory serves, those wizards were a particulary nasty bunch.Their sect was wiped out by a government attack on their home planet during the last consolidation wars, and the few wizards who happened to be off-planet at the time have been vowing revenge ever since.

The story is that several months ago, a scientist doing research in controlled animal mutations, Professor Ion Smartguy, disappeared without a trace from his laboratory on Outpost VI. Not much importance was placed on the matter at the time; after all, it's a big universe, and people disappear all the time for one reason or another. But the intelligence division of GSS has now learned that Professor Smartguy was kidnapped by Zarkon and is being held in a research bunker on a small planetoid known as Alpha II. According to the intelligence reports, Smartguy is being forced to bfeed an army of mutated, killer beasts — a destructive force more powerful than any before it — which Zarkon intends to unleash upon the galaxy.

In fact, if the reports can be taken seriously, he has already assembled a sizable force of these giant mutants which he uses to augment the legion of robot troops that guard his research bunker. If the reports on the strength of these beasts, called Gormas, are true, it's easy to see why those government vapor-heads back on Centra are so shaken. And the information that they're conducting experiments on the sand snake, indigenous to Alpha II, are especially disquieting. Even without controlled mutation, those creatures are widely regarded as the most deadly ever encountered in the universe. After all, Alpha II is where the term "vicious as a snake" first came into use among the early explorers.

Your mission, as related by the GSS bureau chief, is to gain entrance to Zarkon's stronghold, survive any encounters with the mutants and robot guards, rescue Professor Smartguy and his equipment, defeat the evil wizard, overload the nuclear reactor, and then steal a spacecraft and escape before it explodes. Now, they certainly aren't asking for too much, are they? But after all, they *did* put their best agent on the job. With courage and a lot of luck, you just might be able to pull this one off.

Having finally reached a stable orbit around Alpha II, you glance at the small chunk of rock and ice floating just outside the helm's viewport. It's hard to imagine that this tiny, unimposing speck of debris could harbor any tremendous threat to the galaxy. You learned long ago, though, that judging a situation by appearances is fatal, all too often, in this line of work.

With this pleasant thought swirling through your mind, you energize the transporter and step into the shimmering beam of light, remembering too late that it has been malfunctioning lately, and has acquired the nasty habit of separating you from your weapons and equipment during transport. As the light flares and then dwindles to a glow, the cabin is left in silence.

And the Adventure begins. . . .

Loading and Playing Instructions

Rescue On Alpha II is a graphics Adventure requiring at least 32K of RAM and Extended Color BASIC. If you are entering the program from the listing, it should be saved with *CSAVE"ALPHA II"* before *RUNning*. After that, the program can be loaded with the *CLOAD* command; no other commands are needed prior to loading. If you have a disk drive, the program will not run with the drive controller plugged in.

Upon initialization, you will be asked whether your machine can handle the speed-up *POKE*; if it will, type 'Y' and if not, type 'N'. If the answer is no, the program will run a bit slower, but nothing else is changed. If you answer yes and have trouble with keyboard response, rerun the program and change your answer.

The program uses standard two-word commands: a verb followed by a noun. A multipurpose USE command takes the place of most verbs when using an object. For a list of verbs that the program understands, you can take a peek at Line 140 of the listing. Only one abbreviated command is supported: INV for an "inventory" of items that you are carrying. Directional commands must be entered in full, such as GO SOUTH.

To win the game, you must locate the professor and his equipment: a white case, an instrument box, a chest and a machine known as a Biotron. Then, after pushing the self-destruct button in the nuclear reactor room, you must escape with the professor and his equipment in the awaiting spaceship before the reactor blows.

Sound easy enough? In this program by Steven Mitchell, saving the galaxy from evil is never easy.

— Kevin Nickols

		660	104
88	124	800	22
110	122	855	62
130	58	890	170
150	66	952	134
188	138	1102	16
253	190	1115	180
295	188	1124	222
361	95	1145	255
387	89	1171	48
422	145	1210	116
470	161	1229	243
500	63	1251	71
570	237	END	85

The listing:

```
1 X=RND(TIMER):GOTO5Ø
2 GOSUB12Ø1:GOSUB12Ø5
3 RETURN
4 GOSUB12Ø1:GOSUB12Ø6:GOTO3
5 GOTO17Ø
6 GOSUB5:GOTO57Ø
8 GOTO1275
9 GOTO1245
1Ø FORX=1TO12:GOTO13
12 FORX=12TO1STEP-1
13 PLAY STR$(X):NEXT:GOTO3
15 FOR X=1TO2Ø:PLAY"L4ØO3CO4C":N
EXT:GOTO3
2Ø END
5Ø CLEAR95Ø:PMODE4,1:PCLS1:SCREE
N1,1
8Ø DIM L$(71),LO$(25),O$(25),O(2
5),T(4,71),C(11),C$(11),W$(26),E
(6),E$(6)
82 FORX=1TO71:READL$(X):NEXT
84 FORX=1TO25:READLO$(X),O$(X),O
(X):IF LO$(X)=""THEN LO$(X)="THE
 BUNKER TO THE EAST":O$(X)="BUNK
ER"ELSE IF O(X)<1THEN GOSUB76Ø
85 NEXT
86 T$(1)="NORTH":T$(2)="EAST":T$
(3)="SOUTH":T$(4)="WEST"
88 FORX=1TO71:READT(1,X),T(2,X),
T(3,X),T(4,X):NEXT
9Ø L=1:WN=6:NU$="NOTHING UNUSUAL
"
92 FORX=1TO11:READC$(X),C(X):NEX
T
94 FORX=1TO26:READW$(X):W$(X)=W$
(X)+"BR4":NEXT
95 N$="          RESCUE ON ALPHA II
":GOSUB5:N$="         BY STEVEN C. M
ITCHELL":GOSUB5:N$=" ":GOSUB5:N$
="DO YOU WANT THE SPEED UP POKE?
":GOSUB5
96 X$=INKEY$:IF X$="Y"THEN POKE6
5495,ØELSE IFX$="N"THEN POKE6549
4,ØELSE96
99 FORX=1TO6:READE$(X),E(X):NEXT
:GOTO2ØØ
1ØØ DATATHE DESERT,S,THE LIVING
QUARTERS,A LEAD LINED ROOM,THE C
ELL BLOCK,,AN OUTDOOR WALKWAY,,,
A SMALL ROOM,,,AN OUTDOOR WALKWA
Y,THE ENTRY HALL,,
1Ø2 DATAA SMALL SQUARE ROOM,EVIL
 ZARKON'S QUARTERS,THE LANDING B
AY,,A MEETING ROOM,THE WEST GYM,
THE EAST GYM,EVIL ZARKONS BEDROO
M,THE ESCAPE SHIP,,,,,S
1Ø4 DATATHE NORTH LAB,,THE CHEMI
STRY LAB,THE BREEDING PIT,C,C,TH
E LAIR OF THE GORMA,THE SOUTH LA
B,B,B,B,C,C,C,
1Ø6 DATAB,B,B,C,C,C,,,,C,C,C,,TH
E REACTOR ROOM,,C,C,C,C,,,S,C,C,
C,THE LAIR OF THE GIANT SAND SNA
KE
11Ø DATA,,1,,,7,,,13,A SIGN ON T
HE WALL,SIGN,1Ø,ATHLETIC EQUIPME
NT,EQUIPMENT,22,A SLOT ON THE NO
RTH WALL,SLOT,11,BUTTON ON THE W
EST WALL,BUTTON,17
112 DATATHE SELF DESTRUCT BUTTON
,BUTTON,59,THE KEY HOLE,HOLE,25,
A POOL OF WATER,POOL,56
114 DATAA CAPE,CAPE,,A KEY,KEY,,
A CHAIN ,CHAIN,,A SILVER DISK,DI
SK,,A LASER GUN,GUN,-1,A PACK OF
 POISON PELLETS,PELLETS,-1,THE I
NSTRUMENT BOX,BOX,,THE WHITE CAS
E,CASE,37,THE CHEST,CHEST,,THE B
```

```
IOTRON,BIOTRON,,A RING,RING,24
116 DATAA PAIR OF GLOVES,GLOVES,
-1,A LEAD JAR,JAR,2,RADIOACTIVE
MATTER,MATTER,4,THE PROFESSOR,PR
OFESSOR,5
12Ø DATA,,7,,,,-1,,,,9,,,,-1,,,,
11,,,,,
121 DATA1,,13,,-1,9,14,,3,,,8,-1
,11,16,,-2,12,,1Ø,,,18,11
122 DATA7,-1,19,,8,15,2Ø,-1,,16,
,14,1Ø,17,-1,15,,,,16,12,,24,
124 DATA13,,25,,14,,26,,,,-1,,-1
,23,,,,,-1,22,18,,,
126 DATA19,,,,2Ø,27,,,-1,28,,26,
,29,,27,-1,3Ø,,28,,,,29
128 DATA,32,38,,,33,,31,,34,4Ø,3
2,,-1,,33,69,36,42,-1,7Ø,,43,35,
,,44,
13Ø DATA31,,-1,,46,4Ø,46,41,33,4
1,47,39,48,39,48,4Ø,35,43,49,44,
36,44,5Ø,42,37,42,51,43

132 DATA-1,,52,,39,47,39,48,4Ø,4
8,54,46,41,46,41,47,42,5Ø,55,51,
43,51,56,49,44,49,57,5Ø
134 DATA45,53,58,,,54,,52,47,17,
6Ø,53,49,56,62,57,5Ø,57,63,55,51
,55,64,56
136 DATA52,,65,,,,-1,,54,-1,,,68
,62,68,-1,55,63,69,61,56,64,7Ø,6
2,57,68,71,63
138 DATA58,66,,,-1,67,,65,,,,66,
61,69,61,64,62,7Ø,35,68,63,71,36
,69,64,,,7Ø
14Ø DATAGO,1,USE,2,PUSH,3,GET,4,
TAKE,4,DROP,5,INVENTORY,6,OPEN,7
,LOOK,8,LEAVE,5,INV,6
144 DATAU4E2F2D2NL4D2,NR3U6R3FDG
NL3FDGBR,BUFNR3HU4ER3BD6,NR3U6R3
FD4GBR
145 DATANR4U3NR3U3R4BD6,U3NR3U3R
4BD6,BUFR2EU2L2BL2D3U5ER2FBD5,U6
D3R4U3D6
146 DATAR4L2U6L2R4BD6,BUFR2EU5BD
6,U6D3RNE3F3,NU6R4
147 DATAU6F2E2D6,U6DF4DNU6,BUU4E
R2FD4GL2HFBR3,U6R3FDGL2BD3BR3
148 DATABUU4ER2FD4GL2HFR2EHF2L,U
6R3FDGL3R2F2D,BUFR2EUHL2HEUR2FBD
5,BU6R4L2D6BR2
149 DATABU6D5FR2EU5D6,BU6D2FD2FE
U2EU2BD6,NU6E2F2NU6
15Ø DATAUE4UBL4DF4D,BU5UDF2E2UDG
2D3BR2,BU6R4DG4D1R4
152 DATAA ROBOT GUARD,8,A ROBOT
GUARD,26,A KILLER-ROBOT,11,EVIL
ZARKON,31,THE GIANT SAND SNAKE,7
1,THE GORMA,37
17Ø PMODE4:IFV>1ØTHEN GOSUB415
172 COLOR4:X$=STR$(1ØØ+(V*8)):DR
AW"BMØ."+X$
```

```
174 IFLEN(N$)<34THEN18Ø
175 FORZ=34TO1STEP-1:X$=MID$(N$,
Z,1):IF X$=" "THEN177
176 NEXT:Z=34
177 NA$=MID$(N$,Z+1):N$=LEFT$(N$
,Z)
18Ø FORZ=1TO LEN(N$):N1$=MID$(N$
,Z,1):N1=ASC(N1$)
182 IFN1>64AND N1<91THEN N1=N1-6
4:DRAW W$(N1):GOTO19Ø
183 IFN1=63THEN DRAW"BU5ER2FDGLD
BDDBR6"
184 IFN1=58THEN DRAW"BUUBU2UBD5B
R3"
185 IFN1=46THEN DRAW"RBR3"
186 IFN1=44THEN DRAW"R2DGEUBR4":
GOTO19Ø
187 IFN1=33THEN DRAW"UBU2U3BD6BR
3"
188 IFN1=45THEN DRAW"BU3R3BD3BR3
":GOTO19Ø
189 IFN1=39THEN DRAW"BU5URG2BD4B
R4"ELSE DRAW"BR7"
19Ø NEXT
192 V=V+1:IFNA$=""THEN3
194 N$=NA$:NA$="":GOTO5
2ØØ GOSUB5ØØ:GOSUB29Ø:N$="YOU AR
E IN "+L$(L):GOSUB1ØØØ:V=1:GOSUB
5
21Ø VB=Ø:N$="YOU SEE: ":GOSUB77Ø
224 IF VB=ØTHEN N$=N$+NU$
226 VB=Ø:GOSUB5:IF WN<3THEN N$="
YOU ARE GRAVELY WOUNDED!":GOSUB5
:GOTO23Ø
227 IF WN<6THEN N$="YOU ARE WOUN
DED!":GOSUB5
23Ø N$="OBVIOUS EXITS:"
24Ø FOR X=1TO4:IF T(X,L)>ØTHEN N
$=N$+T$(X)+", "
241 NEXTX:GOSUB5:FORX=1TO4
242 IF T(X,L)=-1 THEN N$="A CLOS
ED DOOR IS TO THE "+T$(X)+".":GO
SUB5
244 IF T(X,L)=-2 THEN N$="A LASE
R BARRED DOORWAY IS TO THE "+T$(
X)+".":GOSUB5
245 NEXT:GOSUB78Ø
248 GOSUB7ØØ:N$="WHAT NOW?":GOSU
B5:GOSUB249:GOTO26Ø
249 A$="":TIMER=Ø:BB=Ø
25Ø I$=INKEY$:GOTO256
251 SOUND1,1:IF I$=CHR$(13)THEN3
252 N1=ASC(I$):IF N1=8 AND LEN(A
$)>Ø THEN A$=LEFT$(A$,LEN(A$)-1)
:IF II$=" "THEN DRAW"BM-7,Ø":GOT
O25ØELSE DRAW"C1L8UR7UL7UR7UL7UR
7UL7D6":COLORØ:GOTO25Ø
253 IF N1=32THEN DRAW"BR8":A$=A$
+" ":GOTO25Ø
254 IF N1>64AND N1<91THEN A$=A$+
I$:II$=I$:N1=N1-64:DRAW W$(N1)
255 IF LEN(A$)>24THEN3ELSE25Ø
256 IF TIMER>599THEN TIMER=Ø:GOS
UB5ØØ:FOR Z=1TO6:IF BB=1AND E(Z)
=L THEN FOR X=1TO2:GOTO245ELSE I
F BB=ØAND E(Z)=L THEN57ØELSE NEX
T
257 IF I$=""THEN25ØELSE251
26Ø FOR X=1TO LEN(A$):IF MID$(A$
,X,1)=" "THEN A1$=LEFT$(A$,X-1):
B$=MID$(A$,X+1,LEN(A$)-X+1):GOTO
27Ø ELSE NEXT
265 A1$=A$
27Ø FOR X=1TO11
272 IF C$(X)=A1$ THEN A=C(X):GOT
O3ØØ
274 NEXT
28Ø N$="I DON'T KNOW WHAT "+A1$+
" MEANS.":GOTO426
29Ø IF L$(L)=""THEN L$(L)="A HAL
LWAY"
292 IF L$(L)="C"THEN L$(L)="THE
CAVERNS"
294 IF L$(L)="B"THEN L$(L)="THE
BOTANICAL GARDENS"
295 IF L$(L)="S"THEN L$(L)="A ST
ORAGE ROOM"
296 GOTO3
3ØØ ON A GOTO 32Ø,34Ø,36Ø,375,4Ø
Ø,42Ø,44Ø,48Ø
315 IF VA>ØTHEN T(VA,VC)=-1:VA=Ø
316 GOTO3
32Ø FOR X=1TO4:IFB$=T$(X)THEN DR
=X:GOTO325
322 NEXT:N$="I DON'T KNOW WHERE
"+B$+" IS.":GOTO426
325 X=T(DR,L):IFX>ØTHEN GOSUB58Ø
:L=T(DR,L):GOSUB315:GOTO2ØØ
327 IFX=-2THEN N$="THE LASERS CU
T YOU TO PIECES.":GOSUB5:GOTO3ØØ
Ø
328 IFX=-1THEN N$="THE DOOR BLOC
KS YOUR WAY":GOTO6
329 N$="YOU CAN'T GO THAT WAY!":
GOTO426
34Ø FOR X=11TO25:IF B$=O$(X)AND
O(X)=1ØØØTHEN342
341 NEXT:N$="YOU DONT HAVE A "+B
$:GOTO426
342 IF X=12 AND L=25 THEN2ØØØ
343 IF X=24 AND E(6)=L THEN55Ø
344 IF X=14 AND L=11 THEN T(1,11
)=5:N$="THE LASER BARS ARE GONE.
":LO$(14)="A SHINY DISK IN THE S
LOT":O(14)=11:DRAW"CØ":GOSUB1157
:GOTO6
345 IF X=15THEN52Ø
346 IF X=16 THEN56Ø
348 IF X=22 THEN VE=1:N$="THE GL
OVES ARE ON.":GOTO6
354 N$="NO EFFECT":GOTO6
36Ø IF B$<>"BUTTON"THEN354
361 IF L=17THEN365
362 IF L=59 THEN N$="THE SELF-DE
STRUCT TIMER IS ON!!":F=1:FF=Ø:G
OTO6
364 N$="I SEE NO "+B$:GOTO6
365 IF T(4,17)=16THEN N$="THE RO
OM DROPS LIKE AN ELEVATOR.":X=54
:GOTO367
366 N$="THE ROOM RISES.":X=16
367 T(4,17)=X:GOTO6
```

```
375 IF B$="PROFESSOR"THEN38ØELSE
 Z=Ø:FOR X=1TO24:IF O(X)=1ØØØTHE
N Z=Z+1
376 IF Z>5THEN N$="YOU CAN'T TAK
E THAT. YOU'RE CARRYING TOO MUCH
!":GOTO426
378 NEXT
38Ø FOR X=1TO25:IF B$=O$(X)AND O
(X)=L THEN 39Ø
382 NEXT
384 N$="I DON'T SEE A "+B$+"!":G
OTO426
385 IF VE=ØTHEN N$="THE RADIOACT
IVITY ATE INTO YOUR HANDS,ENTERE
D YOUR SYSTEM, AND  KILLED YOU!"
:GOSUB5:GOTO3ØØØ
386 N$="THE MATTER IS JELLY-LIKE
. WHAT WILL YOU PUT IT IN?":GOSU
B5:GOSUB249:IF O(23)<>1ØØØTHEN38
8ELSE IF A$="JAR"OR A$="THE JAR"
THEN N$="YOU GOT IT!":LO$(24)=LO
$(24)+" IN LEAD JAR":GOTO396
387 FOR X=11TO25:IF A$=O$(X)AND
O(X)=1ØØØTHEN N$="THAT WON'T HOL
D IT!":GOTO6
388 N$="YOU DON'T HAVE THAT!":GO
TO6
39Ø IF X<11 THEN N$="YOU CAN'T T
AKE THAT!":GOTO6
391 IF X=24THEN385
392 IF X<25 THEN N$=LO$(X)+" HAS
 BEEN TAKEN"
393 IF X=14AND RIGHT$(LO$(14),1)
="T" THEN GOSUB17Ø:LO$(14)="A SH
```

```
INY DISK":T(1,11)=-2:N$="THE LAS
ER BARS RETURN!":GOSUB1156
394 IF X=25 THEN N$="THE PROFESS
OR COMES WITH YOU!"
395 IF X=23AND RIGHT$(LO$(24),3)
="JAR"THEN O(24)=1000
396 O(X)=1000:GOTO410
400 FOR X=11TO25:IF B$=O$(X)AND
O(X)=1000THEN O(X)=L:N$="OK":GOT
O405
402 NEXT:N$="YOU DON'T HAVE THAT
.":GOTO6
405 IF X=22THEN VE=0
406 IF X=23AND O(24)=1000THEN O(
24)=L
407 IF X=24THEN LO$(24)="RADIOAC
TIVE MATTER"
410 GOSUB5:GOSUB750:GOTO570
415 COLOR1:LINE(0,102)-(255,191)
,PSET,BF:V=1:GOTO3
420 GOSUB415:N$="YOU ARE CARRYIN
G:"
422 FOR X=11TO24:IF O(X)=1000THE
N N$=N$+LO$(X)+", "
424 NEXT:IF O(25)=1000THEN GOSUB
170:N$="THE PROFESSOR IS WITH YO
U."
426 GOSUB5:GOTO248
440 FOR X=1TO4:IF T(X,L)=-1AND B
$="DOOR"AND L<>61THEN GOSUB450:N
$="THE DOOR IS OPEN TO THE "+T$(
X):T(VA,VC)=VD:GOTO6
441 IF L=61 THEN N$="IT'S LOCKED
 FROM THIS SIDE.":GOTO6
442 NEXT:IF B$="DOOR"THEN N$="I
SEE NO DOOR!":GOTO6
448 N$="OK":GOTO6
450 VC=L:IFL=13THEN VA=2:VD=14
451 IFL=14THEN VA=4:VD=13
452 IFL=27THEN VA=1:VD=21
453 IFL=21THEN VA=3:VD=27
454 IFL=8THEN VA=1:VD=2
455 IFL=2THEN VA=3:VD=8
456 IFL=4THEN VA=3:VD=10
457 IFL=10THEN VA=1:VD=4
458 IFL=38THEN VA=3:VD=45
459 IFL=45THEN VA=1:VD=38
460 IFL=23THEN VA=3:VD=29
461 IFL=29THEN VA=1:VD=23
462 IFL=60THEN VA=2:VD=61
463 IFL=59THEN VA=3:VD=66
464 IFL=66THEN VA=1:VD=59
465 IFL=22THEN VA=1:VD=16
466 IFL=16THEN VA=3:VD=22
467 IFL=34THEN VA=2:VD=35
468 IFL=35THEN VA=4:VD=34
469 IFL=61THEN VD=61
470 GOTO3
480 IF A$=A1$THEN GOSUB580:GOTO2
00
```

```
481 IF B$=O$(4)AND L=10THEN N$="
IT SAYS:":GOSUB170:N$="DANGER-RA
DIATION BEHIND DOOR!":GOTO6
482 IF B$=O$(6)AND O(14)=1000AND
 L=11THEN N$="IT'S ABOUT THE SIZ
E OF THE DISK.":GOTO6
483 FOR X=1TO25:IF B$=O$(X)THEN4
85ELSE NEXT
484 N$="I DON'T SEE A "+B$:GOTO4
26
485 IF X<14THEN487ELSE Z=X-13:ON
 Z GOTO488,487,489,487,487,487,4
87,490,491,492,494
487 N$=NU$:GOTO6
488 N$="IT'S SILVER, AND HAS GRO
VES LIKE A RECORD.":GOTO6
489 N$="THEY'RE IN A SMALL BOX A
ND SMELL BAD.":GOTO6
490 N$="IMPRINTED ON THE INSIDE
IT SAYS: I WILL PROTECT YOU FROM
 MAGIC.":GOTO6
491 N$="THEY ARE WELL LINED AND
HEAVY.":GOTO6
492 IF O(24)=1000THEN N$="IT CON
TAINS RADIOACTIVE MATTER"ELSE N$
="IT'S EMPTY"
493 GOTO6
494 N$="IT GLOWS!":GOTO6
500 FOR X=1TO6:IF E(X)=L THEN3EL
SE NEXT:FOR X=1TO6:DR=RND(4):IF
T(DR,E(X))>0THEN E(X)=T(DR,E(X))
:BB=1:GOTO510
501 IF T(DR,E(X))=-1THEN LL=L:L=
E(X):XC=VC:XA=VA:XD=VD:GOSUB450:
E(X)=VD:L=LL:VC=XC:VA=XA:VD=XD:B
B=1
510 NEXT:GOTO3
514 Y=RND(7):IF Y=1THEN E(X)=17
515 IF Y=2THEN E(X)=52
516 IF Y=3THEN E(X)=10
517 IF Y>3THEN E(X)=15
518 IF E(X)=L THEN514ELSE3
520 FOR X=1TO4:IF E(X)=L THEN525
521 NEXT:GOTO354
525 IF X<4THEN GOSUB514:N$="THE
ROBOT DISINTIGRATES!":GOSUB530:G
OTO248
526 IF O(21)=1000THEN X=4:N$="YO
U HAVE DESTROYED EVIL ZARKON!":G
OSUB529:GOTO248
527 GOTO354
529 E(X)=6
530 PMODE3:FOR Y=2TO50STEP2:CIRC
LE(130,50),Y,RND(4):SOUND Y,1:NE
XTY:GOSUB5:GOTO750
550 X=6:N$="YOU HAVE DESTROYED T
HE GORMA!":GOSUB529:GOTO248
560 X=5:N$="YOU HAVE KILLED THE
SAND SNAKE!":GOSUB529:GOTO248
570 GOSUB580:GOTO248
```

```
58Ø FOR X=1TO6:IF E(X)=L THEN59Ø
ELSE51Ø
59Ø ON X GOTO6ØØ,6ØØ,62Ø,64Ø,66Ø
,68Ø
6ØØ X=RND(17)+1Ø:N$="THE ROBOT F
IRES A RAY":GOSUB5:IF X<23THEN61
Ø
604 IF X>23AND O(24)=1ØØØTHEN N$
="THE LEAD JAR AND RADIOACTIVE M
ATTER ARE GONE!":O(23)=4:O(24)=4
:GOTO5
6Ø5 FOR X=11TO22:IF O(X)=1ØØØTHE
N61Ø
6Ø6 NEXT:N$="NOTHING HAPPENS!":G
OTO5
61Ø IF O(X)<>1ØØØTHEN6Ø4ELSE N$=
"YOU NO LONGER HAVE THE "+O$(X)+
"!":GOSUB5:O(X)=Ø:GOTO76Ø
62Ø N$="THE ROBOT FIRES A LASER
RAY!":GOSUB5:X=RND(6)
621 IF X=1THEN N$="IT MISSES!":G
OTO5
622 WN=WN-1:N$="IT HITS YOU!!":G
OSUB5:IF WN<1THEN3ØØØ
623 GOTO3
64Ø X=RND(3):IF X=1THEN65Ø
644 N$="ZARKON ATTACKS WITH MAGI
C!":GOSUB5:IF O(21)=1ØØØTHEN648
645 N$="YOU ARE HIT!":GOSUB5:WN=
WN-2:IF WN<1THEN3ØØØ
646 GOTO3
648 N$="THE RING PROTECTS YOU!":
GOSUB5:GOTO3
65Ø N$="ZARKON FIRES A LASER GUN
!":GOSUB5:X=RND(4):IF X=1THEN621
ELSE645
66Ø N$="THE SAND SNAKE ATTACKS":
GOSUB5:X=RND(7)
664 IF X=1THEN621ELSE645
68Ø N$="THE GORMA ATTACKS":GOSUB
5:X=RND(8)
684 IF X=1THEN621ELSE WN=WN-1:GO
TO645
7ØØ FF=FF+1:IF F=ØTHEN3
7Ø2 IF FF=1ØTHEN N$="YOU HEAR AN
 EXPLOSION!":GOTO5
7Ø4 IF FF=15THEN N$="YOU HEAR A
LARGE EXPLOSION!":GOTO5
7Ø6 IF FF=22THEN N$="THE PLANETO
ID EXPLODES AROUND YOU!":GOSUB5:
GOTO3ØØØELSE3
75Ø GOSUB1ØØ2
752 GOSUB77Ø:GOTO78Ø
76Ø IF O(X)=ØTHEN O(X)=RND(71)EL
SE O(X)=RND(3Ø)
761 Y=O(X):IF Y=1OR Y=7OR Y=13OR
 Y=19OR Y=25OR Y=5OR Y=6THEN76ØE
LSE3
77Ø FOR X=1TO25:IF O(X)=L THEN N
$=N$+LO$(X)+",  ":VB=1:GOSUB8ØØ
772 NEXT:GOTO3
78Ø FORX=1TO6:IF E(X)=L THEN N$=
E$(X)+" IS HERE!":GOSUB9ØØ:GOSUB
5
782 GOTO772
8ØØ IF X<11THEN3
8Ø1 Z3=3:Z3$="3":PMODE3:Y=X-1Ø:O
N Y GOTO81Ø,815,82Ø,825,83Ø,835,
84Ø,845,85Ø,855,86Ø,865,87Ø,875,
88Ø
81Ø Z1=9Ø:Z2=64:Z$="94,64":Z3=1:
Y$="U2H4L2G4D2F4D2G12D2F4L2E2NE8
F4R2E2U2NU4F4R4E4U2H4U6H2U4H2E4"
:GOTO89Ø
815 Z1=1ØØ:Z2=84:Z$="1Ø2,84":Z3=
1:Y$="H4G4F4E4R6ND2R4ND4R2":GOTO
89Ø
82Ø Z1=5Ø:Z2=82:Z$="52,82":Z3=1:
Y$="G2H2E2F4E4F4E4H2G4H4E4H4E2F2
G4F4G4H4":GOTO89Ø
825 IF L=11AND RIGHT$(LO$(14),1)
="T"THEN826ELSE Z1=74:Z2=86:Z$="
7Ø,86":Z3=4:Y$="U2E4R4F4D2G4L4H4
BR4BE2EFGH":GOTO89Ø
826 DRAW"BM162,58C3E4R4F4L12E1R2
C4R6H2R4":GOTO3
83Ø Z1=24:Z2=8Ø:Z$="22,82":Z3$="
1":Y$="U22L2U2R6D8F2D6F4BL2NU2BR
4R8U2D8L8H2L4G4H4":GOTO89Ø
835 Z1=1Ø4:Z2=86:Z$="1Ø2,88":Z3$
="2":Y$="U1ØR8BD2G2F2E2H2BD4NE4N
F4NG4NH4BU6R8D12L16U2":GOTO89Ø
84Ø Z1=1Ø:Z2=86:Z$="8,82":Z3=2:Y
$="D14R2ØU14L2ØE6F4BG2NU6E8D8L4U
2E2U2E6F4D4":GOTO89Ø
845 Z1=24:Z2=96:Z$="22,98":Z3=4:
Y$="U6E4R6U2R4D2L4R2ND2R6F4D6L2Ø
":GOTO89Ø
85Ø Z1=48:Z2=94:Z$="46,96":Z3$="
2":Y$="U12NR12E6R22F6D12NL32U12L
12BL2D4G2H2U4BL8BU2U4BD6D4BD2D6R
1ØNU2R1ØU6BU2U4BU2U4":GOTO89Ø
855 Z1=66:Z2=96:Z$="64,98":Z3=4:
Y$="U1ØE4H4E4H4E4R8F4G4F4G4F4R1Ø
D1ØL24":GOSUB89Ø:DRAW"BU1C2U8R24
D8L24BE4U2R6BU7L6BU8R6BD15D2BR4E
2F2G2H2":GOTO3
86Ø Z1=11Ø:Z2=96:Z$="1Ø8,96":Z3=
2:Y$="NL2E2F2NR2G2H2":GOTO89Ø
865 Z1=92:Z2=96:Z$="9Ø,98":Z3$="
1":Y$="U6H4U2E2F6H2U6E2F2ND4E2F2
ND4E2F2ND4E2F2D12G4L12":GOTO89Ø
87Ø Z1=12Ø:Z2=96:Z$="118,98":Z3=
2:Y$="U12NR13U2E4R4E2H2G2F2R4F4D
2D12L16":GOTO89Ø
875 IF RIGHT$(LO$(24),3)="JAR"TH
EN3
876 DRAW"BM16Ø,59C3R1ØU1L1ØU1R9U
1L7U1R4U1L2BU4U4BR1ØBD2G4BD4BR2R
4BL2ØL4BR6BU4H4":GOTO3
```

```
880 Z1=12Ø:Z2=94:Z$="118,96":Z3=
4:Z3$="2":Y$="U2E2R6U12L4U16F4U8
D2H4U8BD16F2H8U8E4R8U2H2U6E4R4F4
D6G2D2R8F4D8G8E2BU16D8G4U2D8E4D1
6L4D12R6F2D2L14NU14L14":GOSUB89Ø
882 DRAW"C2BE14BU4U22BR4R4BL8U4N
H4E4U4G2L4H2BE2R3BRBU3L2BDL2BUL2
BU3R6F2D2
884 PMODE3:DRAW"BM126,8ØC2R4E2F2
R4":PAINT(12Ø,94),1,2:GOTO3
89Ø FOR Z=1TO4:Z1$="C"+STR$(Z):D
RAW"BM"+Z$+Z1$+Y$:PAINT(Z1,Z2),Z
,Z:NEXTZ
892 DRAW"BM"+Z$+"C"+Z3$+Y$:PAINT
(Z1,Z2),Z3,VAL(Z3$):PMODE4:GOTO3
9ØØ PMODE3:Z1=126:Z2=98:Z$="1ØØ,
1ØØ":ON X GOTO91Ø,91Ø,91Ø,93Ø,94
Ø,95Ø
91Ø Z3=4:Z3$="1":Y$="E16H16U16E1
6R8U4H1ØU1ØE1ØR1ØF1ØD1ØG1ØD4R8F1
6D16G16F16L6Ø":IF X=3THEN Z3=3
911 X$="L4D4R8U4L4"
912 GOSUB89Ø:DRAW"CØR3ØBU3Ø"+X$+
"BL1Ø"+X$+"BR2Ø"+X$+"BH1Ø"+X$+"B
H1Ø"+X$+"BR1Ø"+X$:DRAW"BR1Ø"+X$+
"BH1ØBU4NL8NR8BU1ØL6E2R1ØF2L8BU1
4L4"+X$+"ND2BR12ND2"+X$:GOTO3
93Ø Z3=1:Z3$="2":Y$="U3ØH4U4E2U2
ØE4R4U4H2U2E8R2E2R4F2R2F8D6F6D26
G4D3ØL34
932 GOSUB89Ø:PMODE3:DRAW"C2R4E4U
12E2U34E4R2U6C3L2U4H4U4E4R6F4D4G
4D4L4U2R4U2L4BU6BL2L2BR6R2BL12C2
D6F6R4E6U4E2U4D1ØF6BL16D4F4D2ØC3
L8R8C2D14F2D12F4R6U6ØBG3ØE4U4H4L
2G4BR4C3D4L2R4U2R2ND2NU2NG8E8C4E
2":GOTO3
94Ø Z3=2:Z3$="3":Y$="H4U4E4H4U4E
4H4U4E4R1ØH6U8H6U6E6R14F6D6G6D8F
6R8F4D4G4F4D4G4R2ØE4R8E2R6G6L4G8
L2ØG4L3Ø
942 GOSUB89Ø:PMODE3:DRAW"C3R3ØE4
U4H4G2L24H2BU12F2R24E2BU12L2ØBU2
6C4L6DR4BR6R4UL6UC1L8R2ND2R8ND2R
2BD8F2H4L6NG4DC4D2BR6U2":GOTO3
95Ø Z3=3:Z3$="1":Y$="E8U8H12E4G1
2H4F8R6F6G2L2H4L6G4D8G2H2U8H6L6G
6R2H2U4E4R12L2H6U2E26R18NF6U4H8U
12F6R6NF2L6E4R12F4L6NG2R6E6D12G8
D4NG6R18E12F2H2ØL6G6L2H2U2E6R8E6
U8E2F2D8F6R1ØF8D2G2R2H8L6F18D2G2
6L1ØNH2R2D6G12D8F8L42
952 GOSUB89Ø:DRAW"CØR8BU24F4R4BR
1ØR4E4BU4G4L4BL1ØL4H4BU4F4R4BR1Ø
R4E4BU2ØBL6C1U4RND4CØNE2L9NH2DC1
D3RU3R2CØR4":PMODE3:DRAW"BU1ØBR4
C1U2C4F2G2H2E2BL12F2G2H2E2D2C1U
96Ø GOTO3
1ØØØ PCLS1
1ØØ2 PMODE3:COLOR1:LINE(Ø,Ø)-(25
5,1ØØ),PSET,BF
1ØØ5 IF L>4ØTHEN1Ø1Ø
1ØØ6 ON L GOTO11Ø1,11Ø2,11Ø3,11Ø
4,11Ø6,3,11Ø7,11Ø8,1126,111Ø,111
1,1112,1113,1114,1115,1116,1117,
1118,1119,112Ø,1121,1122,1122,11
24,1125,1126,1127,1128,1129,11Ø2
,1131,1132,1133,1134,9,9,9,1138,
8,8
1Ø1Ø X=L-4Ø:ON X GOTO8,9,9,9,114
5,8,8,8,9,9,9,1152,1153,1154,9,9
,9,112Ø,1159,116Ø,9,9,9,9,1126,1
129,11Ø2,9,9,9,9
11Ø1 DRAW"BMØ,2ØC3R4E2D16F8D8F6D
6F4D8E4D6F8R6F1ØD6R172U4ØNL2Ø2U3
ØL1ØE2ØR3ØD2ØL3Ø":PAINT(8Ø,84),3
,3:CIRCLE(88,64),32,3,.5,.5:PAIN
T(88,5Ø),4,3:PAINT(222,28),2,3:Y
=12:GOTO1225
11Ø2 GOSUB12Ø1:DRAW"BM15Ø,5ØR5ØL
1ØU1ØL1ØNF1ØD1ØNE1ØL4U8L6D8L6U16
H4L4G4NR12D16BL8ØC3R4L2C4U1ØD2ØG
4D4R8U4H4":GOTO12Ø8
11Ø3 GOSUB12Ø1:GOSUB1192:DRAW"BM
15Ø,9Ø"+X$+"BL136U3ØL3ØD4ØR3ØU1Ø
":GOTO119Ø
11Ø4 GOSUB12Ø1:DRAW"BM8Ø,9Ø":GOT
O1196
11Ø6 GOSUB12Ø1:GOSUB1192:DRAW"BM
4Ø,84"+X$+"BL58U4"+X$:GOTO3
11Ø7 DRAW"BMØ,ØC4F2ØD6ØNG2ØU6ØF1
ØD4ØNG1ØE8U4NR216NF44U2ØNH8F2U4H
2U4H2U4H2U4H2Ø":PAINT(2,Ø),2,4:G
OSUB1212:PAINT(98,96),3,4:Y=4Ø:G
OTO122Ø
11Ø8 GOSUB1112:GOTO1214
111Ø GOSUB11Ø8:DRAW"BM14Ø,5ØC3R2
ØDL2ØDR2ØDL2ØDR2ØDL2ØC1U6R2ØD14L
2ØU1ØBR4R12UL12BD6R12BD2L4BL2L6"
:GOTO3
1111 GOSUB2:DRAW"BM2Ø,2Ø":GOSUB1
2Ø6:DRAW"BM1ØØ,8ØU5ØR5ØD5ØR4U24B
R4D4R2ØU4L2ØBL4U3ØL58D54":GOSUB1
155:GOTO12Ø8
1112 GOSUB4:GOTO12Ø8
1113 DRAW"BMØ,ØC4F2ØD6ØNG2ØE1ØU4
ØNH1ØF4U4H2U4H2U4H2U4H12":PAINT(
2,Ø),2,4:DRAW"BM255,6ØC4L222F4Ø"
:GOSUB121Ø:PAINT(98,96),3,4:DRAW
"S6":Y=4Ø:GOTO122Ø
1114 GOSUB12Ø3:GOSUB12Ø5:GOSUB12
Ø6:DRAW"BM36,36D1ØC4H4D26BR19ØU2
6G2D22BL1ØØBU2U2ØR18D2ØL2C1L14":
GOTO12Ø8
1115 GOSUB12Ø2:GOSUB1213:GOSUB12
15:DRAW"BR8ØR2ØC4U3ØL2ØD3Ø":GOTO
12Ø8
1116 GOSUB2:GOSUB1211:GOSUB123Ø:
GOTO12Ø8
1117 GOSUB12Ø1:GOSUB123Ø:DRAW"U2
```

```
6BR36U8R4D8L4U4R2C3U2":GOTO12Ø8
1118 GOSUB11Ø4:DRAW"BM2Ø,2ØC4":G
OSUB12Ø5:DRAW"BM2Ø,2Ø":GOSUB12Ø6
:GOSUB1194:DRAW"BM4Ø,8Ø"+X$:GOTO
3
1119 DRAW"BMØ,2ØC4F2ØL4NL16D28NR
238D12ND2ØR2ØF2Ø":PAINT(Ø,26),2,
4:PAINT(9Ø,9Ø),3,4:DRAW"S8":Y=3Ø
:GOTO122Ø
112Ø GOSUB12Ø2:GOSUB1215:GOTO12Ø
8
1121 GOSUB12Ø1:GOSUB1211:DRAW"BM
5Ø,9ØC2":GOSUB1228:DRAW"BM7Ø,92"
:GOSUB1229:PAINT(56,82),4,2:PAIN
T(7Ø,74),4,2:PAINT(52,74),4,2:GO
TO3
1122 GOSUB12Ø1:GOSUB1211:DRAW"BM
Ø,7ØR4E2R4E2R4E2R216F2R4F2":PAIN
T(2,72),2,4:PAINT(5Ø,66),2,4:PAI
NT(238,7Ø),2,4:DRAW"BM94,8ØD16F6
R12E6U4NL24U12L3BU4ØE2U4H4L12G4D
4F2R2E2R6F2NR2BH2C2L6GFGFGFGBU7R
4GFGFGFGBU7R3FGFGFGFBU7R2DBDDBDD
BDD":IF L=22THEN124ØELSE3
1124 GOSUB11Ø3:GOTO12Ø8
1125 DRAW"BMØ,1ØC4F6R17ØF2R6F2ØN
D6ØR6ØD4ØG2ØL25ØBU6ØBR2Ø4L6ØU2ØR
34":PAINT(Ø,14),2,4:PAINT(25Ø,5Ø
),4,4:DRAW"BD79C1R2ØU5ØL1ØG8L1ØU
4D8U4R1ØG8D6R8D22G6R6BU28L5ØU6NR
4ØU16NR56D5ØR5ØL2ØC3L2ØR1ØU24R1Ø
L2ØU2ØR2D6BR8U1ØR14D1ØL14BU6ØBL6
ØC1D1ØØL8ØU88":GOTO1195
1126 GOSUB2:GOTO12Ø8
1127 GOSUB12Ø3:GOSUB121Ø:DRAW"BM
36,36D6H2D24G2U28D28E2R8ØU2ØR2ØD
2ØL2C1L16":GOTO12Ø8
1128 GOSUB12Ø2:GOSUB1212:DRAW"BM
116,7ØC4U3ØR3ØD3ØL2C1L26":GOTO12
Ø8
1129 GOSUB1132:GOTO121Ø
1131 GOSUB4:DRAW"BM4Ø,9Ø":GOSUB1
196:X=196:Z=4:DRAW"BM188,86":GOT
O1197
1132 GOSUB12Ø1:GOSUB123Ø:GOTO12Ø
8
1133 GOSUB2:GOSUB12Ø6:DRAW"BM6Ø,
84":GOSUB1196:X$="U1ØL2D1ØBL4":D
RAW"BM18Ø,52C3U1ØH2L2G2D2BL2ØC2D
9L4U9L1R5D7L2C3BL2ØBD2C3"+X$+X$+
"C2"+X$+"C3"+X$+X$+"C4U6R4ØD6":G
OTO3
1134 GOSUB12Ø1:GOSUB1214:X=3:GOS
UB1231:X=3Ø:Z=4:DRAW"BM22,84":GO
SUB1197:X=2ØØ:DRAW"BM192,84":GOT
O1197
1138 GOSUB12Ø1:GOSUB1214:DRAW"BM
2Ø,9ØC2":GOSUB1228:DRAW"BM4Ø,92"
:GOSUB1229:PAINT(24,82),4,2:PAIN
T(38,74),4,2:PAINT(22,74),4,2:X=
196:Z=2:DRAW"BM188,86":GOTO1197
1145 GOSUB12Ø1:GOSUB1214:GOTO12Ø
8
1152 GOSUB2:GOSUB12Ø6:GOTO12Ø8
1153 GOSUB12Ø2:GOSUB1215:GOSUB12
16:GOTO12Ø8
1154 GOSUB12Ø3:GOSUB12Ø5:GOSUB12
Ø6:DRAW"BM36,46C4H4D26BR19ØU26G4

":GOTO12Ø8
1155 IF RIGHT$(LO$(14),1)="T"THE
N3
1156 DRAW"C3"
1157 DRAW"BM1Ø6,79U48BR8D48BR8U4
8BR6D48BR8U48BR8D48":GOTO3
1159 GOSUB12Ø1:DRAW"C1NR8ØNH1ØD6
ØNR8ØG1ØC3U8ØR8ØD8ØNL8ØE1ØU6ØH1Ø
C2G2ØL4ØNH2ØD2ØNG2ØR4ØNU2ØF2ØNL8
ØR2E2R2E2R2":PAINT(4Ø,4Ø),4,2:GO
TO117Ø
116Ø GOSUB12Ø1:GOSUB121Ø:GOTO12Ø
8
117Ø DRAW"BM14Ø,86C4U2ØL2R64L2D2
ØL4U4L26U2C1NR28C4U1ØL26D1ØC1NL2
C4D6L4":PAINT(142,84),2,4:DRAW"C
1BE18BR16R12BD2L4BU2R12D4BD2D6L2
ØU6R12BD2L4BU2R12BU2L2ØU4BH2C4L4
H2L18G2E4R14F2H2U1ØL14D1ØBR56U1Ø
R8D1ØL8"
1171 PAINT(21Ø,6Ø),2,4:DRAW"U8R2
C3R4":PSET(21Ø,6Ø,3):GOTO3
119Ø IF L=9THEN X=2ELSE X=4
1191 PAINT(6Ø,8Ø),X,2:X$="C1U8L1
ØD2NL2U2L12D8R22":DRAW"BM68,6Ø"+
X$:DRAW"BM68,7Ø"+X$:DRAW"BM68,8Ø
"+X$:GOTO3
1192 X$="C2U16D8R6ØND8U2L6ØUR6ØU
L6ØUR6ØNU4H4L6G4R12":GOTO3
1194 X$="C2U1ØE2R2F2D2ND8G2L2H2D
1ØR2U2L2R36D2L2U2R2L6U1ØE2R2F2D2
ND1ØG2L2H2D2L24BU6BL2U14E4R2ØF4D
14BL4BD4H2L16G2":GOTO3
1195 PAINT(176,64),1,1:PAINT(16Ø
,54),1,3:DRAW"BM16Ø,68C4U4R4D4L4
BU16L8BD2R5":GOTO3
1196 DRAW"C2U3ØL2R124L2D3ØL12Ø":
PAINT(82,8Ø),4,2:DRAW"C1BE2":FOR
 X=1TO6:DRAW"U14BU4U6R8D2R2U2L2R
8D6NL16BD4NL16D6L2ND2R2D8NL16BR4
":GOTO51Ø
1197 DRAW"C2U6ØR4ØD6ØL4U4L32D4L4
":PAINT(X,76),3,2:PAINT(X,76),Z,
2:DRAW"U8BR6C1R28U48L28D48R14U24
NU24BR4R2BL1ØL2":GOTO3
12Ø1 DRAW"BMØ,ØC4F2ØD6ØNG2ØR214N
F2ØU6ØNE2ØL214":GOTO3
12Ø2 DRAW"BMØ,ØC4F3ØD4ØNG3ØR192N
F3ØU4ØNE3ØL192":GOTO3
12Ø3 DRAW"BMØ,ØC4F36D28NG36R182N
F36U28NE36L18Ø":GOTO3
12Ø5 DRAW"BMØ,1ØØC4E4U8ØF1ØD56NL
```

```
1ØD4C1G8C4":GOTO3
1206 DRAW"BM254,100C4H4U80G10D56
NR1ØD4C1F8C4":GOTO3
1208 PAINT(Ø,4),2,4:PAINT(5Ø,5Ø)
,2,4:PAINT(255,4),2,4:GOTO3
121Ø DRAW"BMØ,2ØF1ØD3ØBL2H2G2F2E
2BR2D3Ø":PAINT(2,26),2,4:GOTO3
1211 DRAW"BM255,2ØG1ØD3ØBR2F2E2H
2G2BL2D3Ø":PAINT(252,26),2,4:GOT
O3
1212 DRAW"BM2Ø,4ØND4ØF6D14L2D2R2
U2D14":PAINT(22,44),2,4:GOTO3
1213 DRAW"BM235,4ØND4ØG6D14R2D2L
2U2D14":PAINT(232,44),2,4:GOTO3
1214 DRAW"BM1ØØ,8ØC4U4ØR3ØD2ØBL4
H2G2F2E2BR4D2ØL3Ø":PAINT(1Ø2,79)
,2,4:GOTO3
1215 DRAW"BM3Ø,4ØC4H6D42E2C1E4":
GOTO3
1216 DRAW"BM222,4ØC4E6D42H2C1H4"
:GOTO3
122Ø DRAW"BM15Ø,5ØC2R1ØF4R4L4G4L
1ØD4R6NE2L1ØR4U4L4U6L4D2R2L2D4R4
L2ØG4R6NE2L1ØR4U4R4L16U8L4D6R4U2
NR2ØU6E2H2E2H2R4F2R2F2R2F2R2F2R2
F2R2ØH4L1ØG4S4":PAINT(15Ø,52),3,
2:PAINT(15Ø,52),4,2:PAINT(2Ø,96)
,4,4
1225 FOR X=1TO6Ø:PSET(RND(2ØØ)+Y
,RND(5Ø),RND(3)+1):NEXTX:GOTO3
1228 DRAW"U2ØD1ØR12L2D1ØU6L6ND2L
4E4R1ØNG4D6U1ØR2L12D4U14G4":GOTO
3
1229 DRAW"U16L8E6R6ØG6L4ND8L48D4
R4ND8R4ØE4D12BG4U16":GOTO3
123Ø DRAW"BM11Ø,8ØC1R3ØC4U4ØL3ØD
4Ø":GOTO3
1231 CIRCLE(13Ø,1ØØ),8Ø,4,.14,.5
:PAINT(13Ø,98),X,4:GOTO3
124Ø DRAW"BM4Ø,12D4C3D38C4L2G2D2
F2R2E2U2H2BR6F2D2G2L2H2U2E2R2C3U
4ØC4U2BD7ØC3L4G4D3F4R4E4U3H4":PA
INT(46,86),3,3:DRAW"C1BD2G6BU8F8
":GOTO3
1245 DRAW"BMØ,3ØC3E2D2E2F2U2E4F6
D3ØF2D3ØR2ØNU4ØE4R6E4R6ØU3ØE4U3Ø
E6R2F4E6F4D4F2E2F4E2U4E6R2F4D2ØF
4D44R2ØF4R1ØF4R3ØU3ØE4U3ØE4F2D4F
2E2U2E2F4E4BU3ØG8D4G2H2U6H4G6D4G
4H2U2H4U2G8D4G8U4H6G8D2G4H6U4G4D
2H8D6H8BL4ØG1ØD2G4H4U6H6L2G4D6G2
D4G2ND4H2U6H4U2H6
1246 DRAW"G4D4G2D6G2H2U4H6R2D4G6
D4G4ND4H8U4H6G4H8G4H6U4H8G8R2
1247 DRAW"BR4BD6D4G4BR2ØF6D4G6D8
BR2ØG6H4BR3ØD6F4R6F8BU2ØU6H4BR24
D6G4BD16NH8D6G4BR9ØU4H8BR1ØBU2ØE
4R6F1ØBF2ØD6G4BH1ØU4H6BE16D4E8BG
46BR2U8E8BF8BD14U6E8U4E6BL162ND1
8NE6BH4D24U3ØH4"
```

```
1248 IF L=36OR L=37 OR L=71THEND
RAW"BM236,3ØC1D3ØG4D28C3F8R16":I
F L<>36 THEN DRAW"BM112,84C1NU4Ø
C3E6R4ØF6U2C1NU4Ø":IF L=37 THEN
DRAW"BMØ,1ØØC3R1ØE8U6NE8U2C1U26H
2U2Ø
1249 PAINT(2,2),3,3:PAINT(2,14),
2,3:PSET(2,24,2):PAINT(2ØØ,24),2
,3
125Ø IF L=61OR L=35THEN DRAW"BM1
12,84R2E8U4ØR32D2ØBL2H2G2F2E2BR2
D2ØNF8L32":PAINT(13Ø,7Ø),2,3
1251 IF L=56THEN X=2:GOTO1231ELS
E3
1275 GOSUB2:GOSUB12Ø6:GOSUB123Ø:
DRAW"BM2Ø,ØF12R8ØU12BR2ØD12R84E1
2":PAINT(4Ø,Ø),4,4:PAINT(2ØØ,Ø),
4,4
1276 DRAW"BM14,86C1NE8C3R86E6C1N
L88C3U1ØL86C1ND8C3G6ND1ØR86ND1ØE
6BR4ØD1ØC1NR88C3F6R88C1NH8C3U1ØH
6C1ND8C3L88F6ND1ØR88"
1277 FOR X=62TO196STEP43:PAINT(X
,78),2,3:NEXTX
1278 DRAW"BM2Ø,74C2":GOSUB129Ø:D
RAW"BR54":GOTO129Ø
129Ø FOR X=1TO2Ø:X$=STR$((RND(8)
*2)+2):Y$=STR$((RND(4)*2)+2):DRA
W"U"+X$+"NH"+Y$+"NU"+Y$+"NE"+Y$+
"D"+X$+"BR4":GOTO51Ø
2ØØØ GOSUB415:PLAY"L99":GOSUB1Ø:
N$="THE SHIP TAKES OFF FOR HOME!
":GOSUB5
2Ø18 IF O(25)=1ØØØOR O(25)=25THE
N N$="YOU SAVED "+LO$(25):GOSUB1
Ø:F1=1ELSE N$="YOU DID NOT SAVE
"+LO$(25):GOSUB12
2Ø2Ø GOSUB5:FOR X=17TO2Ø:IF O(X)
<>25AND O(X)<>1ØØØTHEN2Ø4ØELSE N
EXT:N$="YOU SAVED ALL "+LO$(25)+
"'S EQUIPMENT!":F2=1:GOSUB1Ø
2Ø22 GOSUB5:IF F=1THEN X$="YOU D
ESTROYED":GOSUB1ØELSE X$="YOU DI
D NOT DESTROY":GOSUB12
2Ø23 N$=X$+" THE PLANETOID!"
2Ø24 GOSUB5:IF F=1AND F1=1AND F2
=1THEN N$="        PERFECT MISSION
!":GOSUB15ELSE N$="BETTER LUCK N
EXT TIME!":GOSUB12
2Ø3Ø GOTO3ØØ2
2Ø4Ø N$="YOU MISSED SOME OF "+LO
$(25)+"'S EQUIPMENT!":GOSUB12:GO
TO2Ø22
3ØØØ N$="YOU ARE DEAD!!"
3ØØ1 PLAY"O1GFD#DC"
3ØØ2 GOSUB5:N$="PLAY AGAIN?":GOS
UB5
3ØØ3 X$=INKEY$:IF X$="Y"THEN RUN
3ØØ4 IF X$="N"THEN END
3ØØ5 GOTO3ØØ3
```

QUICKIES

The Line Box

By Kevin Oberberger

```
10 PMODE3,1:PCLS1:SCREEN1,1
20 CIRCLE(128,95),85,4:CIRCLE(12
8,95),60,4:PAINT(128,30),4,4
30 LINE(165,50)-(70,115),PSET:LI
NE(185,70)-(85,137),PSET:PAINT(1
00,95),4,4
40 CIRCLE(128,95),36,2,1,.26,.35
:DRAW"BM116,129C2M-2,+8M-10,+78M
86,105M-6,-5"
50 CIRCLE(53,82),32,2,1,.1,.24:D
RAW"BM51,111F4G2H4L7F2L7G4H2E6L7
G2H2E4R2BL2M-4,-1H2L2U1E2M+10,+3
M+20,+1BL2M+14,-25M+6,-8M+2,-2M+
4,-2"
60 CIRCLE(88,57),18,2,1,.3,.7:CI
RCLE(108,54),18,2,1,.85,.2:CIRCL
E(102,30),20,2,1,.5,.8:DRAW"BM82
,40E2U7":CIRCLE(102,30),20,2,1,.
87,.1
70 CIRCLE(113,25),20,2,1,.65,.85
:DRAW"BM118,12E7R4D6M-8,+8":CIRC
LE(140,88),38,2,1,.7,.83:LINE(15
9,54)-(86,105),PSET:PAINT(90,80)
,1,2:PAINT(90,60),1,2
80 DRAW"BM173,81E7F2M+22,-8R2D2M
-10,+4G3BE2M+6,-1M+6,+1F2D1L9G2M
+10,+4G2M-10,-4L9G5L2H2U3L4H2":P
AINT(180,78),1,2
90 CIRCLE(94,29),2,2,2:CIRCLE(10
4,29),2,2,2:DRAW"BM96,56F2E2U2H2
G2D2":PAINT(98,55),2,2:DRAW"BM96
,66F2E3BU24G3H2BU22BL2G4BR18BU1H
3"
100 GOTO 100
```

Nova

**By Michael Rosenberg
and
Tobin Wonn**

```
10 '      **** N O V A ****
20 '   BY MICHAEL ROSENBERG
30 '            AND
40 '         TOBIN WONN
50 '       MAY 10, 1984
60 PCLEAR8
70 PMODE 4,1:PCLS:SCREEN1,1
80 FORJ=0TO255STEP2
90 LINE(J,191)-(128,96),PSET:NEX
T
100 FORJ=0TO255STEP2
110 LINE(J,0)-(128,96),PSET:NEXT
120 FORJ=0TO191STEP2
130 LINE(128,96)-(255,J),PSET:NE
XT
140 FORJ=0TO191STEP2
150 LINE(128,96)-(0,J),PSET:NEXT
160 PMODE4,5:PCLS:SCREEN1,1
170 FORJ=0TO255STEP4
180 LINE(J,191)-(128,96),PSET:NE
X1
190 FORJ=0TO255STEP4
200 LINE(J,0)-(128,96),PSET:NEXT
210 FORJ=0TO191STEP4
220 LINE(128,96)-(255,J),PSET:NE
XT
230 FORJ=0TO191STEP4
240 LINE(128,96)-(0,J),PSET:NEXT
250 FURX=1105STEP4
260 PMODE3,X:SCREEN1,1:FORY=1TO1
00:NEXTY,X
270 GOTO250
```

No Ghosts

By Balinda Fortman

```
1 POKE65495,0
2 PMODE1,1
3 PCLS3
4 SCREEN1,0:POKE65314,248
5 FORX=3TO7
6 FORY=2TO6
7 FORZ=0TO3
8 COLORZ
9 A=0:B=255:C=0:D=191
10 LINE(A,C)-(B,D),PSET,B
11 A=A+Y:B=B-Y:C=C+X:D=D-X
12 IFA<255ANDC<191THEN10
13 NEXTZ
14 NEXTY,X
15 RUN
```

An Optical Illusion

By Bryan Kerr

```
1 ' AN OPTICAL ILLUSION
2 ' BRYAN KERR
```

```
3 ' 1195 BRIARWOOD DR
4 ' JACKSON,MS   39211
5 ' 8/17/83
1Ø PMODE3,1:PCLS:SCREEN1,Ø
2Ø A$="C4D3ØM-3Ø,-15U3ØNM+3Ø,+15
M+3Ø,-15M+3Ø,+15NM-3Ø,+15D3ØM-3Ø
,+15U3Ø"
3Ø DRAW"S3XA$;BM-8Ø,-4ØXA$;BM+8Ø
,-4ØXA$;BM+8Ø,+4ØXA$;BD8ØXA$;BM-
8Ø,+4ØXA$;BM-8Ø,-4ØXA$;"
4Ø B$="C1BM-3Ø,+15C1NU8D2NM+8,+4
U2BM+3Ø,-15":C$="C1BM+3Ø,+15NU8D
2NM-8,+4U2BM-3Ø,-15":D$="C1BU29N
M-8,+4NM+8,+4BD29"
5Ø E$="C4BM-3Ø,-15M-4Ø,-2ØD1ØM+4
Ø,+2ØU1ØM+1Ø,-5M-4Ø,-2ØM-1Ø,+5M+
4Ø,+2ØBM+3Ø,+15"
6Ø F$="C4BD31D4ØNM-8,-4M+8,-4U4Ø
BM-8,+4BM-8,-4ND4ØBM+8,+4BU31"
7Ø G$="C4BD1M+3Ø,-15M+4Ø,-2ØNM-8
,-4D1ØM-4Ø,+2ØU1ØM-1Ø,-5NM+4Ø,-2
ØM+1Ø,+5M-3Ø,+15"
8Ø DRAW"BM128,96C4XA$;BU2XB$;XC$
;XD$;BD8ØXD$;BM-8Ø,-4ØXC$;XD$;BU
8ØXC$;BM+8Ø,-4ØXB$;XC$BM+8Ø,+4ØX
B$;BD8ØBM+8Ø,+4ØXD$;XB$;XE$;BU8Ø
XB$;BD8ØBM-8Ø,+4ØXG$;XE$;BM-8Ø,-
4ØXG$;BU8ØXF$;XG$;BM+8Ø,-4ØXF$;B
M+8Ø,+4ØXE$;XF$BM-8Ø,+4ØXE$;XF$;
BM-8Ø,+4ØXG$;XE$;XF$;
9Ø FOR C=2TO3:FORT=1TO15:READX,Y
:PAINT(X,Y),C,4:NEXTT,C
1ØØ FOR T=1TO7:READX,Y:PAINT(X,Y
),4,4:NEXTT
11Ø DATA 128,94,13Ø,153,128,3Ø,1
6Ø,74,188,6Ø,68,6Ø,96,74,96,49,1
58,49,68,12Ø,188,12Ø,116,1ØØ,14Ø
,1ØØ,1Ø5,14Ø,152,14Ø
12Ø DATA 135,95,13Ø,16Ø,13Ø,37,1
9Ø,75,19Ø,135,7Ø,135,7Ø,75,13Ø,8
5,13Ø,135,118,1Ø5,95,56,165,8Ø,1
58,145,19Ø,95,7Ø,95
13Ø DATA 1Ø3,65,151,65,158,95,98
,95,1Ø3,125,158,125,128,Ø,
14Ø GOTO14Ø
```

The Hole

By Gregg A. Ford

```
1 'CIRCLE2
2 'GREGG FORD
3 '95 ELDRED BEDFORD,OHIO 44146
1Ø PMODE 4,1:PCLS:SCREEN 1,1:FOR
 X=1Ø TO 17Ø STEP 2
2Ø CIRCLE(128,96),X,7
3Ø NEXT
4Ø CLS:FOR X=1Ø TO 17Ø STEP 5
5Ø CIRCLE(128,96),X,7
6Ø NEXT
7Ø FOR X=1Ø TO 17Ø STEP 1Ø
8Ø CIRCLE(128,96),X,2
9Ø NEXT
1ØØ FOR X=Ø TO 17Ø STEP 4
11Ø CIRCLE(128,96),X,3
12Ø NEXT
13Ø FOR X=Ø TO 17Ø STEP 15
14Ø CIRCLE(128,96),X,Ø
15Ø NEXT
16Ø FOR X=1Ø TO 17Ø STEP 9
17Ø CIRCLE(128,96),X,4
18Ø NEXT
19Ø FOR X=1Ø TO 17Ø STEP 3
2ØØ CIRCLE(128,96),X,5
21Ø NEXT
22Ø FOR X=1Ø TO 17Ø STEP 1
23Ø CIRCLE(128,96),X,6
24Ø NEXT
25Ø FOR X=1Ø TO 17Ø STEP 2
26Ø CIRCLE(128,96),X,7,2
27Ø NEXT
28Ø FOR X=1Ø TO 17Ø STEP 3
29Ø CIRCLE(128,96),X,8,2
3ØØ NEXT
31Ø FOR X=Ø TO 17Ø STEP 5
32Ø CIRCLE(128,96),X,Ø,2
33Ø NEXT
34Ø FOR X= 1Ø TO 17Ø STEP 7
35Ø CIRCLE(128,96),X,1,2
36Ø NEXT
37Ø FOR X=Ø TO 17Ø STEP 1
38Ø CIRCLE(128,96),X,6
39Ø NEXT X
42Ø CLS:PRINT@232,"THAT'S ALL FO
LKS!"
43Ø GOTO 43Ø
```

The Impossible Cube

By Stephen Lai

```
5 '      IMPOSSIBLE CUBE
1Ø PMODE4,1:PCLS:SCREEN1,Ø
2Ø DRAW"S44BM34,3NFR12NM-1,2F5NM
-2,1D12NH2L12NEH5NU12URNM+2,1ER3
D2L2F2NU8M+1,2ENR8U3NR3U2NR3U3NR
3HNELH3ND7M-1,-2M+3,1NF3R7F4D9UH
3U4NR3U2NR2U2HLD3NL3D5ENR8F3"
3Ø FORE=1TO8:READA,B:PAINT(A,B):
NEXT:DATAØ,Ø,1ØØ,3Ø,6Ø,7Ø,14Ø,9Ø
,19Ø,9Ø,17Ø,5Ø,14Ø,16Ø,76,14Ø
4Ø GOTO4Ø
```

Pascal's Triangle

By Stephen Lai

```
5 '      PASCAL'S TRIANGLE
1Ø DIM P(256)
2Ø PMODE 4,1:PCLS:SCREEN 1,1
3Ø P(128)=1
4Ø FOR S=1 TO 127
5Ø FOR F=128-S TO 128+S STEP 2
6Ø P(F)=P(F-1)+P(F+1)-SGN(P(F-1)
+P(F+1))*2
7Ø PSET(F,S,P(F)+5)
8Ø NEXT F,S
9Ø GOTO 9Ø
```

Everything You Always Wanted To Know

About The Color Computer

But Radio Shack Didn't Tell You

This is part five in a series concerning various aspects of the Color Computer

By Andy Kluck

In response to a lack of information from Radio Shack on the newest revisions of the CoCo's ROMs and the bugs in the old ROMs they replace, I have compiled this article to explain the major differences between them. Also included is a utility program for use in 64K systems for installing any set of BASICs in RAM from files. Numbers in parentheses are approximate addresses of some of the ROM changes.

Color BASIC 1.1

The most important difference between Color BASIC 1.0 and 1.1 is that the Reset routine ($A027-$A06D or -$A073 in 1.1) of 1.1 has the added ability to detect the 64K RAM jumper and properly initialize the SAM to use 64K RAMs. In order to make room for the extra code required to make this test, the initialization routines ($A06E, or $A074 in 1.1 through $A0CA) have been largely rewritten. The keyboard scan routine ($A1C1-$A26D) has also been rewritten to prevent the joystick buttons from causing a string of false characters to be detected. The printer driver ($A2C5-$A2FA) has been modified to send eight bits per character instead of seven. A bug ($A440) in the CLOSE routine for output files that prevents writing of the end of file mark for files with lengths that are integer multiples of 255 has been fixed. Because of this problem, reading such a sequential file (of length 0, 255, 510, etc. bytes) written by Color BASIC 1.0 causes the computer to keep searching for the next record past the end of the file. However, Extended BASIC 1.0 and 1.1 both fix this bug using a RAM hook, so only BASIC 1.0 without Extended BASIC causes this problem. Finally, an error ($A6EB) has been fixed to allow for visible blinking of the corner of the screen during *SKIPFing* of ungapped files, such as those made by *CSAVE* and *CSAVEM*.

The major incompatibility problem caused by these changes is that the few programs using the keyboard scan routine to detect the joystick buttons may now require certain keys on the keyboard to be used instead.

Color Basic 1.2

The major change made in the 1.2 BASIC ROM is that the routine that tests for the BREAK and Shift @ keys before each BASIC statement ($A1C1-A26D) is executed now scans the keyboard only if at least one key is held down. This results in faster execution of BASIC programs whenever no keys or joystick buttons are down. The speed increase may range from a few percent in programs with lots of slow instructions to 200 percent in programs like:

```
10 FOR A=0 TO 1000::::::::::NEXT
```

In order to implement this, the keyboard scanner has been compressed, and its entry point referenced by the vector at $A000 has moved from $A1C1 to $A1CB. There are also some minor changes. The 1.2 printer output routine ($A2C3-$A2FA) waits for a ready signal from the printer before and after sending each character instead of only making this test after sending like the earlier ROMs. This prevents the loss of one character that may otherwise have been outputted before the printer was ready. Also, the initial Baud rate constant ($A114) has been changed from 87 to 88, apparently in an attempt to more closely approximate the correct Baud rate.

A new patch ($B23F) prevents a syntax error when a Hex or Octal constant is preceded by a plus sign with Extended BASIC installed *PRINT + &H20* incorrectly gives an error in the "get operand" routine has been patched ($B3ED-$B426) to give a TM Error when a string is used as the argument for some functions that should require numeric arguments. For example, *CLEAR: PRINT EOF(A$),CHR$(A$)* gives no error in Color BASIC 1.0 and 1.1. Finally, the floating point bug ($B9D6) has been fixed so that CoCo no longer thinks that *PRINT 1000 + 1E-38'Correct answer is 1000* is 1E-38. Watch out for programs that call the keyboard scanner directly at $A1C1 instead of through the vector at $A000. In revision 1.2, $A1C1 is the address of the routine which quickly tests the keyboard and then scans the keyboard normally only if at least one key is held down or otherwise clears the A register and returns. Therefore, machine language programs that call $A1C1 to scan the keyboard may work normally except that they do not accept any keyboard character twice in a row, because the rollover table is not updated between key presses. For the same rea-

(Andy Kluck is an electrical engineering student at the University of Texas at Austin.)

son, BASIC programs, which test the keyboard rollover table at $152-159 to detect which keys are pressed (a technique which has been referred to as "auto repeat"), may receive a false indication that a key is being held down after it has been released, until another key or joystick button is pressed. For example, this routine tests if the 'Q' key when held down keeps printing the message after the 'Q' is released until another key is pressed in Color BASIC 1.2:

```
10 IF (PEEK(&H153) AND4)=0
THEN ?"Q DOWN"ELSE CLS
20 GOTO 10
```

Extended Basic 1.1

The most important change in Extended BASIC 1.1 is that *PCLEAR* has been patched ($96A3-$96B3, $80D0-$80DD) to work properly inside a program. Also, *PRINT USING* has been patched ($9179, $917D) so that numbers printed in exponential format with exponents greater than nine are not botched up as in Extended BASIC 1.0. Another patch ($8C1B-$8C22, $8C51) fixes *DLOADM;* under Extended BASIC 1.0 without Disk BASIC, it didn't work at all.

Finally, *PMODE*($962C-$962D) now tests location $BC to find the correct start of graphics page one instead of assuming it is $600; this was not a major problem, since $600 is always the start of page one except with Disk BASIC, which fixes *PMODE* by itself through a RAM hook. The most likely problem caused by the changes is with programs that use a *PCLEAR 0* routine that calls the ROM's *PCLEAR* routine in the area $96A5-$96B3, which has been rewritten; such routines need to be replaced with an equivalent designed to work with either ROM.

Disk Basic 1.1

First of all, Disk BASIC 1.1 adds a new interpret loop ($C8B0-$C90B in new ROM) through one of the RAM hooks, which has the effect of speeding up execution in the same way as Color BASIC 1.2, so that a system with either one or both of new ROMs will run at the increased speed. *GET, PUT*, and *LOC* have been modified to work with record numbers up to 65535 instead of just 32767. Extra error checking has been implemented in some places, and most of the bugs in the close routine for random files have been fixed. Closing random files in any order other than last-opened, first-closed caused a system crash in Disk BASIC 1.1, and this has been fixed ($CB93-$CBB1 in new ROM); however, a similar crash in the close routine after an I/O Error while the file is open during COPY has not. This last problem will probably require major patches in COPY, CLOSE, or both to fix. In Disk BASIC 1.1, closing a random file no longer ($CACB in old, $CB80 in new) deletes strings fielded in buffers other than the one being closed.

Also, in the old ROM, string array elements fielded in any random buffer while any random file is being closed may have their descriptor addresses messed up, causing general confusion, and this has been fixed ($CA8C-$CAD5 in old, $CB3D-$CB88 in new) in revision 1.1. The bug in FILES that was analogous to the *PCLEAR* bug has been fixed (new code at $D1E2-$D1EE), along with the one (new code at $D1A8-$D1AE) that sometimes causes FILES to allocate the beginning of graphics area to an odd page where the SAM could not display it. Also, in Disk BASIC 1.0 the FILES statement reserved one more byte for random file buffers than was requested; *FILES 2,200* would allow a random file to be opened with a record length of 201; this is corrected (instruction at $D0AB-$D0AC in old ROM removed) in revision 1.1. A bug that could bite (byte?) multi-drive systems causing wrecked file allocation tables after files are written on one drive while files are opened on the next higher drive (wrong index in instruction at $C70C-$C70D in old ROM) has been fixed. Also, in Disk BASIC 1.1, the prompting string in the statement *INPUT #DN,"INPUT NUMBER"; N$* is ignored (by new code $C860-$C887) unless DN is 0, in which case it is printed to the screen. The old Disk BASIC would have printed "INPUT NUMBER" into the random file buffer, if DN were an open random file. A new command, DOS (main routine , $DF00-$DF4B) has been added, which loads all of track $22 (34) from drive 0 starting at address $2600 and jumps to $2602 if the characters in $2600-2601 are "OS." DOS appears to have been implemented by Radio Shack instead of Microsoft, and is not well-written; depending on various circumstances, DOS may erase part of the BASIC program, variables, stack (causing a crash), or strings before testing for "OS" to see if the disk is even bootable.

For what it's worth, DOS may be called from machine language by *JMP [$C00A]* using a vector new to Disk BASIC 1.1. There is also a new vector at $C008, called during initialization, which points to a routine ($DF4C-$DF58) that sets the RAM vectors for SW12 and SW13 to RTIs; for some unknown reason a SW13 is used at the beginning of the DOS routine. The SW12 and SW13 to RTIs; for some unknown reason a SW13 is used at the beginning of the DOS routine. The warned users to press Reset after each *DSKINI* command when doing more than one disk at a time to ensure proper formatting. There is an error in the Disk 1.0 *DSKINI* routine which causes the write-precompensation flag of the disk controller, which is supposed to be set only for track numbers greater than 21, to be set while formatting all tracks if *DSKCON's* track register is set greater than 21 when *DSKINI* is executed, possibly causing I/O Errors. This which is fixed (by an instruction at $D5F1-the new ROM, and it appears to be what Radio Shack was referring to, although the description doesn't exactly match the problem. The error does not only occur when *DSKINI* is used more than once; it can happen any time when the last sector read was on a track greater than 21; and besides, pressing Reset doesn't prevent the problem, because it doesn't change the track register. However, doing a *DIR* of any drive, or a *POKE &HEC,0* before each *DSKINI*, will. There are also changes in other parts of the ROM that I wasn't able to analyze which may suggest other problems in the old ROM. Because the ROM has been completely reassembled, all useful routines have been moved, including *DSKCON*. Therefore, most programs that call any Disk ROM routines except *DSKCON* through the vector at $C004 won't work with Disk BASIC 1.1. This does not necessarily mean that such programs are obsolete with the new ROM if they will run in the 64K RAM mode, since this allows the user to install the old Disk BASIC in RAM.

What They Didn't Fix

The most annoying problem that should have been fixed but wasn't, is Disk BASIC's lack of a suitable method of synchronizing the disk head of each drive when it is first used. The current software just assumes that each one is at track zero, and if this isn't correct, *DSKCON* attempts to read the wrong track before finding the correct position. If the head position is initially inside of track 17, it results in ramming the mechanism against the inside stops, often knocking the drive so far out of alignment that the head must be syn-

.... Continued on P 32

CRYSTAL BLADE Software

QUALITY SOFTWARE FOR THE TRS-80 COLOR COMPUTER

P.O. BOX 256, ROSEVILLE. NSW. 2069

LARGEST RANGE OF SOFTWARE IN AUSTRALIA - PROMPT MAIL ORDER SERVICE

SEND $2 FOR COMPLETE CATALOGUE (OVER 30 PAGES)

ROSEVILLE USER'S GROUP MEETING. ENQUIRIES PHONE: (02) 467-1619

..... PROGRAMMING PROBLEMS - I/Output Errors on loading; Software Operation queries - there is sure to be someone there with years of experience with the COCO who would be happy to assist you and give advice.

Arcade 16K

ASTRO BLAST	31.00	LAS VEGAS	22.00
ASTRA LANDER	22.00	LEMANS	32.00
BEAM RIDER	31.00	LIGHT RUNNER	31.00
BLOCHEAD	31.00	MAD BOMBER	22.00
BUMPERS	31.00	MOON SHUTTLE	36.00
BUTTERFLY BOMBER	32.00	MS NIBBLER	31.00
CATERPILLAR	31.00	TRAVELIN TOAD	31.00
CAVE HUNTER	31.00	NOVA PINBALL	25.00
CESSNA HUNTER	22.00	PENGON	32.00
CHOPPER STRIKE	34.00	PLANET INVASION	31.00
COLOR FURY	34.00	RAIL RUNNER	31.00
COLORPEDE	37.00	ROBOT BATTLE	31.00
COSMIC CLONES	31.00	SEAWOLFE	24.00
DEFENSE	27.00	SHARK TREASURE	31.00
DEMON ASSAULT	31.00	SPACE INVADERS	31.00
DOODLE BUG	31.00	SPACE RACE	31.00
EL BANDITO	31.00	SPACE RAIDERS	31.00
FLIPPER	22.00	SPACE SENTRY	27.00
FLYING TIGERS	31.00	SPACE TRADER	22.00
FOOTBALL	31.00	STRIP TEASE	18.00
GALACTIC HANGMAN	20.00	SUB HUNT	24.00
GHOST GOBBLER	31.00	TRAPFALL	34.00
GLAXXONS	31.00	WHIRLYBIRD RUN	31.00
HAYWIRE	31.00	XYGOID	31.00
INTERGALACTIC FORCE	27.00	YAHTCC	16.00
KATERPILLAR ATTACK	31.00	ZEUS	24.00
KOMET KAZE	31.00	ZONE SIX	31.00

Arcade 32K

CASHMAN	34.00
CHOPPER STRIKE	34.00
COLOR FURY	34.00
CUBIX	31.00
DEMON SEED	34.00
DESERT PATROL	31.00
DEVIOUS	31.00
DONKEY KING	32.00
EIGHT BALL	31.00
FLIGHT	27.00
FOOD WAR	31.00
ICEMASTER	31.00
INTERCEPT 4	34.00
LANCER	31.00
LUNAR ROVER PATROL	31.00
MS GOBBLER	31.00
MUDPIES	34.00
OUTHOUSE	34.00
POOYAN	36.00
ROBOT BATTLE	31.00
TIME BANDIT	34.00
TUT	31.00
WACKY FOOD	31.00
ZACKSUND	32.00
ZAXXON	43.00

Adventures

ADVENTURES IN WONDERLAND	31.00
BLACK SANCTUM	22.00
CALIXTO ISLAND	22.00
ERLAND	31.00
GANGBUSTER	31.00
GREYMOON	22.00
INCA TREASURE	22.00
KEYS OF WIZARD	25.00
LOTHAR'S LABRYNTH	16.00
MONSTERS & MAGIC	31.00
PARANOIDS ANONYMOUS	22.00
PIRATES AHOY	16.00
PYRAMID	27.00
QUEST	27.00
SQUIRE	16.00
VIKING	31.00

Graphic Adv.

ACROSS THE RUBICON	22.00
ADVENTURE TRILOGY	27.00
BEYOND CIMEEON MOON	27.00
CALIXTO ISLAND	31.00
CONQUEST OF KZIRGLA	27.00
DUNGEONS OF DEATH	27.00
FEMBOTS REVENGE	27.00
INSPECTOR CLUESEAU	30.00
LABRINTH	27.00
SCEPTRE OF KZIRGLA	22.00
SEAQUEST	31.00
SHENANIGANS	31.00
TREK 16	25.00
WIZARD 64	27.00
WIZARDS TOWER	27.00

Future

SUPER ASTROLOGY	31.00
I CHING	22.00
NUMEROLOGY	25.00
TARROT	27.00

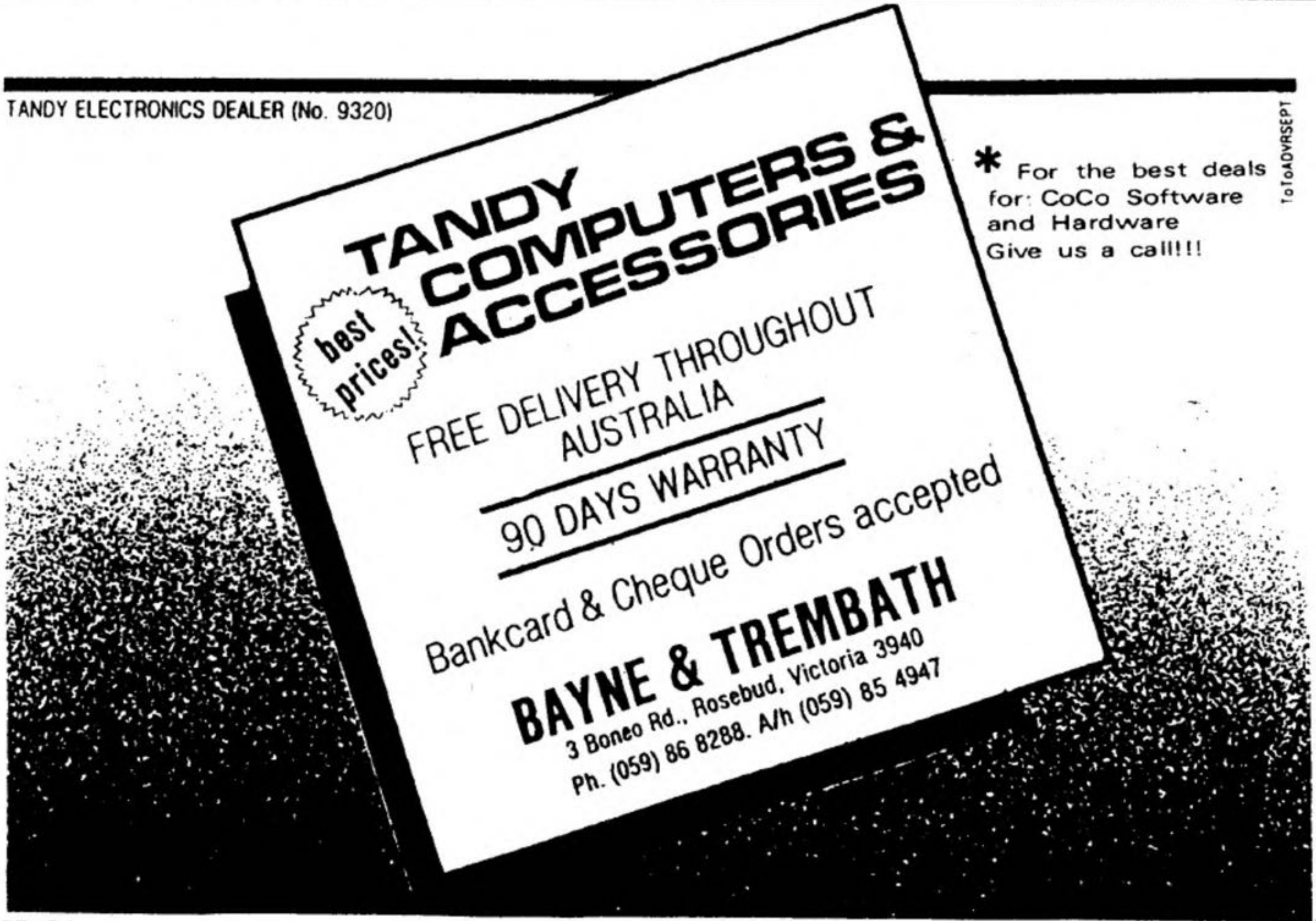
Utilities

64 COL MOD I/III EMULATOR	20.00
AUTO RUN	21.00
BUGOUT MONITOR	27.00
COCO CALLIGRAPHER	31.00
DISASSEMBLER	27.00
DISK MANAGER	36.00
DISK UTIL	25.00
GRAPHIC ANIMATOR	16.00
MAGIC BOX	28.00
PLUS 32	20.00
PRITTY PRINTER	25.00
QUICK SORT	16.00
SCREEN MACHINE	37.00
SOUND SOURCE	27.00
SUPER SCREEN	37.00
SUPER ZAP	42.00
TAPE UTIL	42.00
THE COMPOSER	31.00
VIP DISK ZAP	70.00
VDOS / VDUMP	34.00

Applications

AUTOTERM	48.00
CC DATABASE / LETTERWRITER	54.00
CC WRITER DISK	42.00
CC WRITER TAPE	25.00
COLOR DFT	32.00
DISK ZAPPER	48.00
ELITE CALC	65.00
ELITE WORD	70.00
FILMASTR	42.00
GRAPHICS EDITOR	22.00
MAILING LIST	59.00
MAIL / MERGE	31.00
STATGRAF	25.00
SUPER SCROLL	31.00
TELEWRITER-64	59.00
TERMTALK	47.00
VIP CALC	65.00
VIP DATABASE	70.00
VIP SPELLER	70.00
VIP TERMINAL	54.00
VIP WRITER	65.00
TIMS	31.00

TANDY ELECTRONICS DEALER (No. 9320)


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!!!

ToToADVRSEPT

CoCo LINK

As mentioned in the editorial this month, CoCoLink is off and running. The phone number is 075-32-6370, in case you didn't read your mailing label last month!

Understandably, we are still ironing out little problems, but none of these has affected CoCoLink's availability to callers.

Calls to CoCoLink are free until 28th February, by which time we will hopefully at least know about most of the bugs!

After that time, visitors will be given only limited access to the system.

The major difficulty that we face is that the system is really only set up for 100 users, a fact we did not know when we purchased the software. Because there are a heap of very clever folk working on the software, I don't believe that this factor will be a problem in the longer term. In the short term, it means that we will have to establish a waiting list for those who wish to subscribe, but miss getting on initially. GoCo'ers who want to take advantage of the transfer offer have placement priority until 15th Feb.

Apart from expanding the membership files, CoCoLink is to have added to it in the coming weeks, a department for Model 100 users, one for MC-10 users, one for those interested in Educational topics, and the ability, (which it doesn't currently possess), to upload programs from you.

Other computer users have also shown interest in CoCoLink. At present there is no plan to make space available for programs for other brands, although discussion regarding other brands is welcome.

I am very excited by the arrival of CoCoLink. It's not just that it adds a new dimension to your Tandy computer; I believe that we have the embryo of a new medium, for a new age.

Just around the corner, I can see computers becoming a real tool to be used in the home like a TV, Video, or Newspaper, but having greater consequence. We are moving away from the days of the computer being just a hobbyist's toy, or a thing for the kids to play games on. The modem is the gadget which will put a computer in every home.

There are further plans for CoCoLink, some of which I am not prepared to talk about at this stage, and some which are the product of my scattered brain and therefore have yet to be checked for validity. But I can tell you, that we will be expanding CoColink this year, and that we seek new and innovative ways to utilize this powerful medium.

CoCo LINK

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

VISA MasterCard

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..................

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

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		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		□ $3.95/copy
		□ $3.25/copy to Dec.
	1st issue '81	□ $3.00/copy to July

BOOKS

Byte Elementary	$5.95	□
Help Medium	$9.95	□
Facts Advanced	$11.95	□
MiCoHelp Medium	$9.95	□
MiCo Exposed V. Advanced	$11.50	□

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	$6	□	Blank tapes 12 for $18 or $1.70 ea.	□
□	6 months	$32	□	Cassette Cases 10 for $5	□
□	12 months	$58	□	Disks — $3.50 ea. □ 10 for $28.99	□

BULLETIN BOARD

CoCo Link — Annual Sub. $29 □

□□□□□□□□□□□□□□

□ VISA BANKCARD □

____________________ Cardholder

VISA

□ CASH

□ CHEQUE

□ POSTAL ORDER

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)

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.

As usual you will find the programs on the two tapes detailed in this month's Australian CoCo.

The programs reflect the Educational flavor of the issue, however there is also an excellent game called FIREFOX. Both tapes are have more programs than usual, reflecting the increasing quantity of programs being received.

ANNOUNCING The BEST of CoCoOz!!

To assist teachers and others who are involved with children in learning situations, we have compiled a 12 program tape which reflects some of the best educational programs for CoCo.

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, postage paid.

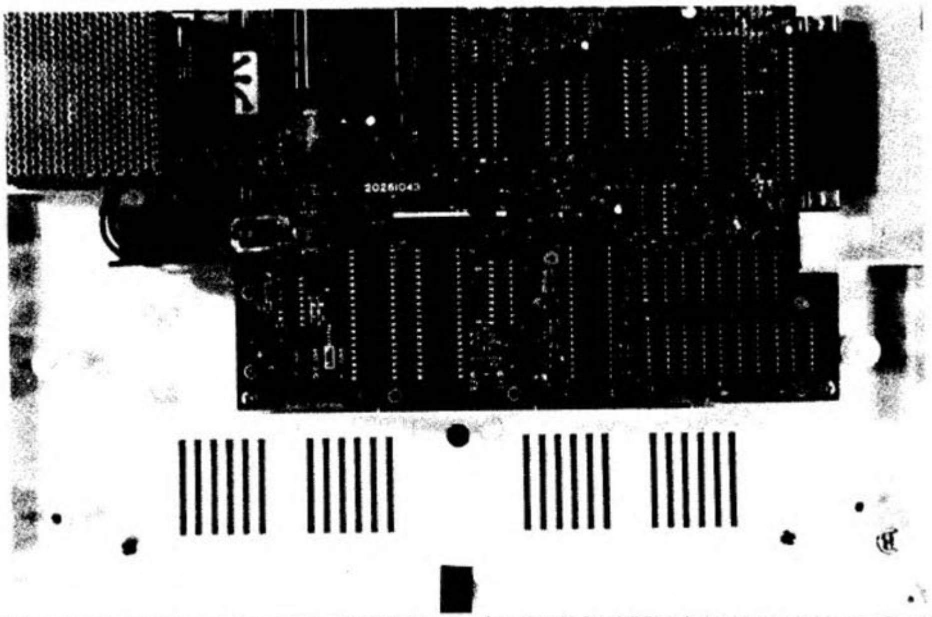
TURN OF THE SCREW

An Introduction To The Inside Of The CoCo 2

By Tony DiStefano

My, doesn't time go fast? I can't believe I've been writing for RAINBOW for two years now. January being THE RAINBOW's Beginners issue, I decided to introduce the novice to the inside world of the Color Computer. The latest CoCo 2 is the newest Color Computer to be introduced by Tandy. It is different inside from the old CoCo 2. You can tell the difference by the shield covering the power transformer. Though it functions the same, the insides of this CoCo are very different (again!). More on that later.

Before we get on our way, let me mention that I just came back from my second RAINBOWfest. I must say that these shows are great. I found THE RAINBOW staff to be very friendly and helpful. It is amazing to see that much enthusiasm generated about the Color Computer. Chances are I'll see some of you at the next RAINBOWfest, too, in California. Stop in and say hello. Look for me at the R.G.S. Micro booth.



Now, let's look into this little thing, but remember, opening your computer might void your warranty. Radio Shack only warranties the computer for three months, so after that you are on your own, anyway. First of all, *never* open the computer with the power on. Now that that's said, let us continue.

To open your CoCo, use the following procedure. Place the computer upside down on a towel (or other soft surface) on a clean work table. Remove the four screws (one in each corner) with a medium-sized Phillips screwdriver. There is one more screw to remove; it is behind the little sticker that says "Opening case will void warranty. See owner's manual for warranty information." You must break this seal to remove the last screw. That is how Radio Shack can tell if you have opened it. Just push the screwdriver through the center of the sticker; it will give way to a hole. Some of the CoCo 2s may have a sixth screw on the other side. Remove the last screw. Turn the computer back right side up, and gather up the screws that drop out. Grab the top cover of the computer and pull it off. Wow! Look at all those things. The components marked with the letter 'U' (or 'IC' in the case of the newest CoCo) are known as ICs (Integrated Circuits). Table 1 labels all the ICs used in the computer and gives a short description of each.

TABLE 1

IC #	Name	Description
1	SC77527	SALT Power supply and RS-232 CHIP
2	MC6821P	PIA Peripheral interface adapter
3	SC77526	DAC Digital to analog converter
4	NE555D	Timer for color burst in *PMODE* 4
5	74LS273	Octal D-Type Flip-flop
6	74LS244	Octal Buffer Driver
7	SC67331P	IIA Industrial interface adapter
8	MC6847P	VDG Video Display Generator
9	MC6809EP	CPU Central Processing Unit.
10	74LS02	Quad 2-input Nor Gate
11	74LS138	3 to 8 Decoder Chip
12	8040364B	ROM BASIC 1.2
13	8042364A	ROM EX BASIC 1.1
14-21	8040517	16K DRAM Dynamic Random Access Memory
22	MC6883P	SAM Synchronous Address Multiplexer

Beginners Project Parts List

Quantity	Description	Radio Shack #
1	LED	276-068 or 276-069 or 276-073
1	RESISTOR 1k ohms	271-8023

Some of the components that make up the CoCo are very sensitive to static electricity. You must be careful not to zap (permanent damage caused by static discharge) a chip by touching the pins with your fingers. If you must touch a chip, always touch a ground point with your fingers first. This will discharge any static your body might be carrying to ground. A good ground

.... Continued on P 42

. From P 25 ...

chronized to track zero two times before it finds the directory. The right way would be to initially set the track counter of each drive to some out of range value, and have *DSKCON* upon finding this value either synchronize the drive to track zero, or even better, attempt to read an address field and use its track number.

The program listed below can be used to load any available combination of the three BASICs into RAM, either to substitute Disk BASIC 1.0 for a newer version ROM to run an incompatible program, or to upgrade to the revised versions. It requires 64K of RAM, and copies of whichever BASICs are to be replaced. To make these, you need to temporarily acquire a CoCo with the ROM versions you need, and make the necessary files,

(C)SAVEM "CBASIC12",&HA000, &HBFFF,&HB44A
(C)SAVEM "XBASIC11",&H8000, &H9FFF,&HB44A
(C)SAVEM "DBASIC11",&HC000, &HDFFF,&HB44A

changing the filenames according to which version you're getting; i.e., "DBASIC10" for Disk 1.0 ROM, etc. To determine the revision numbers of the Color BASIC and Extended ROMs in each system, these statements may be used:

PRINT PEEK(41301)-48 ' Revision of Color BASIC
PRINT PEEK(33023)-48 ' Revision of Extended BASIC, if applicable.

Of course, if you don't feel like pirating the BASICs you need, you can always ask Radio Shack for help in getting your ROMs replaced. To use the *INSTALL* program, place either the word "INTERNAL" or a filename to be loaded for one of the three BASICs in each of the *DATA* statements in Lines 280-300, in the order of Color BASIC, Extended BASIC, Disk BASIC. If all three are "INTERNAL," *INSTALL* simply copies the BASICs unchanged from ROM to RAM. To make the cassette version, change the *LOADM* in Line 130 to *CLOADM*, and remove Line 170; also make *DATA* statement in Line 300 "INTERNAL" unless you want to add Disk BASIC. Either save the program on a disk with copies of the ROM files, or on tape, preferably followed by the ROM files, and run it. *INSTALL* operates in two different modes depending on whether Disk BASIC is loaded or not.

If Disk Basic is not loaded, no adjustments are necessary to BASIC's internal variables; in this case the user may insert a line like: *70 RUN "PROGRAM"* to *RUN* an application program after loading the set of BASICs. Afterwards the user may *POKE &HFFDE,0* or press Reset to switch back to ROM BASIC, and then *POKE &HFFDF,0* to bring back the RAM version.

If Disk BASIC is loaded, *INSTALL* assumes it is of a different revision than the one in ROM. Because different revisions of Disk BASIC have their routines moved around, it is necessary to set the hooks and other variables according to the new BASIC. This is done by calling the Extended BASIC cold start routine at $8002, and results in an automatic *NEW* and printing of the sign-on message. The practice of using *POKEs* to switch between RAM and ROM causes a crash if a different version of Disk BASIC is in RAM because of incompatible hook addresses. Reset may be used to bring back the ROMs, at least with Disk BASIC 1.0 and 1.1, but BASIC will be cold started because the Reset vector at $71 no longer points to a NOP instruction.

130 201
END 211

The listing:

```
10 REM INSTALL PROGRAM FOR BASIC
S
20 REM ANDY KLUCK 8-16-83
30 X=PEEK(&H8000):POKE &H8000,25
5-X
40 IF PEEK(&H8000)<>X THEN POKE
&H8000,X:PRINT"ERROR- PROGRAM MU
ST BE STARTED  IN ROM MODE":END
50 GOTO 310
60 POKE &H9D,&HB4:POKE &H9E,&H4A
:PCLEAR 4:POKE&HFFDF,0 ' SET EXE
C POINTER TO FC ERROR; SWITCH TO
 RAM
70 END
80 CLEAR 200,&H1EFF
90 FOR I=&H1F00 TO &H1F3D:READ I
$:X=VAL("&H"+I$):S=S+X:POKE I,X:
NEXT
100 IF S<>6475 THEN PRINT"DATA E
RROR":STOP
110 EXEC &H1F00:GOSUB 190 ' COPY
 ROMS TO $2000-$7FFF
120 FOR I=1TO3
130 READFI$:IF FI$<>"INTERNAL"TH
EN LOADMFI$,&HA000
140 NEXT
150 EXEC &H1F02:GOSUB 190 ' COPY
 $2000-$7FFF TO $8000-$DFFF IN R
AM; SWITCH BACK TO ROM MODE.
160 IF FI$="INTERNAL"THEN CLEAR
200,&H7FFF:GOTO60 ' IF DBASIC NO
T LOADED
170 POKE &HEA,0:POKE &HEB,0:EXEC
PEEK(&HC004)*&H100+PEEK(&HC005)
' MOVE DRIVE 0 HEAD TO TRACK 0
180 CLEAR200,&H7FFF:EXEC &H1F05
' SWITCH TO RAM MODE AND JMP $80
02
190 IF PEEK (&H1F04) THENPRINT"B
AD MEMORY ERROR":END ELSE RETURN
200 DATA 20,09,20,0F,00,B7,FF,DF
210 DATA 7E,80,02,8E,80,00,CE,20
220 DATA 00,20,0B,8E,20,00,CE,80
230 DATA 00,1A,50,B7,FF,DF,7F,1F
240 DATA 04,31,89,60,00,34,20,EC
250 DATA 81,ED,C4,10,A3,C1,27,05
260 DATA 86,01,B7,1F,04,AC,E4,26
270 DATA EE,B7,FF,DE,35,A0
280 DATA CBASIC12
290 DATA XBASIC11
300 DATA DBASIC11
310 PMODE0,1:PCLEAR1:GOTO80
```

It Is Better To PUT Than To GET

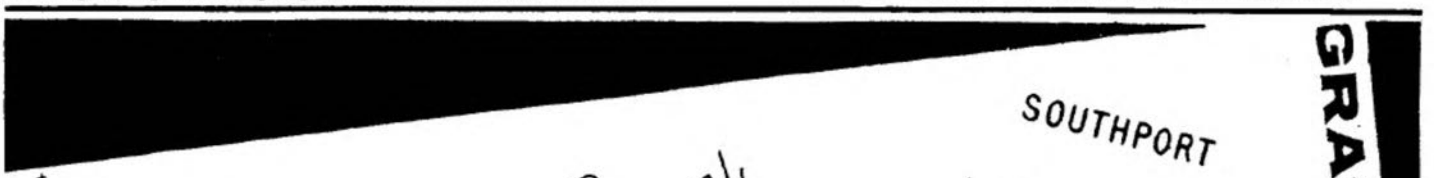
By Alexander B. Trevor

A little experience with Extended Color BASIC graphics will make it obvious that by far the fastest way to place graphics elements on the screen is with the *PUT* statement. In fact, it is almost the only way to implement any kind of animation on the Color Computer in BASIC. Most descriptions of *PUT* tie its use to the *GET* statement, although this does not have to be the case, as we shall see. (Radio Shack's *Going Ahead With Extended Color BASIC*, pages 67-71, and *Color Computer Graphics,"* by William Barden, Jr., pages 143-154 are two examples). This is not too surprising, since the two statements are intended to be complementary. The idea is the *GET* will copy graphics data out of a specified rectangular area on the graphics screen, and save it in an array. The *PUT* statement allows you to place the data back on the screen at the same or another location.

In a typical game application, a number of "sprites" (such as rockets, robots, lunar landers, etc.) will be defined at the beginning of the program and then used with *PUT* statements throughout the game. The usual method of defining the sprites is to use the *DRAW* command to (slowly) draw each sprite on the graphics screen; then, each sprite is stored in a separate array with the *GET* command.

There are several problems with this method, though none of them may be serious in many applications: first, the *DRAW* command must be done on a separate graphics page, or it will deface anything that is already on the page. This is an important consideration for graphics editors, which may allow you to work on an image already in

(Alexander Trevor, who holds a master's degree in electrical engineering, is executive vice president of Computer Resources at CompuServe, and is a member of the IEEE and the ACM. In his "spare time" he writes software for CompuServe's DEC-10 mainframes and for a variety of microcomputers.)



graphics memory, but not important in any program that clears the graphics page upon start up. Second, if there are many items to be drawn, the setup process can cause a noticeably long delay. Third, after *GETting* graphics into an array the contents of the array become "invisible," and cannot be printed to the screen, tape or disk. The *GET* command purposely trashes part of the array header to cause this undesirable side effect.

The method described here avoids these problems through an alternate way to load arrays with graphics data that can be used in a *PUT* statement. To use this technique, it is necessary to understand the format that data is stored on the screen, and also the nature of arrays. Rather than attempt to describe all the possibilities, I will limit this discussion to *PMODE 4*, the high resolution mode consisting of 256 horizontal elements (pixels) and 192 pixels vertically. The technique is easily extended to other graphics modes.

In *PMODE 4*, each byte in the graphics page contains eight pixels along the horizontal. Since there are 256 pixels on each line, it takes 32 bytes to hold the first line. There are 192 lines in the entire picture, or 6,144 bytes. Data for *GET* and *PUT* is stored in a similar scheme. If you *GET* a 5 x 4 area into an array, you are storing 5 x 4, or 20 pixels. In this case one pixel requires one bit of storage. The pixels are stored tightly packed in the array, with the upper left hand pixel stored as the most significant bit of the zero byte of the array. Since each byte can contain eight bits, this graphic will require only three bytes. For example a lowercase 'v' might appear on the screen like this:

Graphics Screen Area (5 horiz x 4 vert)

	Column				
	1	2	3	4	5
Row 1:	1	.	.	.	1
Row 2:	1	.	.	.	1
Row 3:	.	1	.	1	.
Row 4:	.	.	1	.	.

As stored in the array;

Byte 1: 1 0 0 0 1 1 0 0
(row 1) (row 2, 1-3)

Byte 2: 0 1 0 1 0 1 0 0
(row 2, 4-5) (row 3) (row 4, 1)

Byte 3: 0 1 0 0 0 0 0 0
(row 4, 2-5) (fill)

Arrays in Extended Color BASIC consist of elements that are five bytes each. Five bytes are necessary to hold a floating point number with the precision used in ECB, but the five bytes have no bearing on the graphics use of arrays except to make dimensioning and loading more confusing. In the above example, the three bytes will fit easily into the five bytes that will be allocated to a single array element.

How do we get three bytes of data into the first three bytes of a five byte integer array element? First, we must find out the address of the array element with the *VARPTR* statement — one of the less frequently used BASIC commands. Then, we *POKE* the data into the memory locations reserved for the array. That's all there is to it! The array can now be used in a *PUT*.

To see how this works in practice, let's follow through the example of a lunar lander sprite as shown below. The sprite is drawn in a rectangular area on the graphics screen eight columns wide and seven rows high:

	Column								Hex
	0	1	2	3	4	5	6	7	Value
Row 1.	.	.	1	1	1	1	.	.	2C
Row 2:	.	1	1	1	1	1	1	.	7E
Row 3.	.	1	1	1	1	1	1	.	7E
Row 4	.	1	1	1	1	1	1	.	7E
Row 5	.	.	1	.	.	1	.	.	24
Row 6	.	.	1	.	.	1	.	.	24
Row 7	.	1	1	.	.	1	1	.	66

In this example, it is particularly easy to determine the value to be poked into the array because the graphics area is exactly eight pixels wide. Thus, an entire row of pixels fits exactly into one byte of the array. In cases where the rectangle width is not a multiple of eight (such as the example of the lowercased 'v' above), you may wish to draw the graphic on the screen using any method (*DRAW* command, or graphics editor), then use the program given in Listing 1 to print out the appropriate values.

Listing 2 is a complete example for the "*PUT* without *GET*" technique. In Line 20, four graphics pages are reserved. Line 30 sets the horizontal and vertical size of the array. These values (HSIZE and VSIZE) are then used to *DIM*ension the array LL. HSIZE*VSIZE is the number of pixels; this is divided by eight bits per byte and five bytes per array element (i.e., 40), and rounded to the next integral byte. A subroutine to load the graphics array LL is called from Line 50, while lines 60-90 simply *PUT* the element on the graphics screen. The symbol setup routine (lines 100-170) is the key to the technique. In Line 110, all variables to be used within the subroutine are referenced *This is absolutely necessary* in order to prevent the location of the array from changing after the *VARPTR* statement. If an undefined variable is encountered by BASIC, all variables are relocated in memory, invalidating the address returned by the *VARPTR* statement in Line 120. With an accurate address for the zero element of LL in variable P, the data is *POKEd* into each byte of the array in Line 150. In a variation of this method, the graphics data can be read in from disk or tape rather than from the *DATA* statements.

I have found this technique particularly useful for programs that use a number of sprites, and in which it was desirable to reduce the setup time to a minimum. Next time you want to speed up a graphics program, *PUT* something you didn't *GET*. You'll be rewarded with a faster program.

Listing 1:

```
10 ' PRINT DATA FOR A "PUT"
20 ' USE AFTER DRAWING GRAPHIC
```

```
30 X=100: Y=100 'UPPER L CORNER
40 HSIZE=8 'SET HORIZONTAL SIZE
45 VSIZE=7 'SET VERTICAL SIZE
50 DIM LL(HSIZE*VSIZE/40+1)
55 'VARIABLES MUST BE REFERENCED
56 'BEFORE CALLING VARPTR
60 I=0: P=0: J=0
70 GET(X,Y)-(X-1+HSIZE,Y-1+VSIZE
),LL,G
80 P=VARPTR(LL)+12
90 FOR I=0 TO HSIZE*VSIZE/8-1
100 IF J=0 THEN PRINT:PRINT"DATA
 ";
110 PRINT HEX$(PEEK(P+I));
120 IF J>6 THEN J=0 ELSE J=J+1:P
RINT",";
130 NEXT
140 PRINT"0"
```

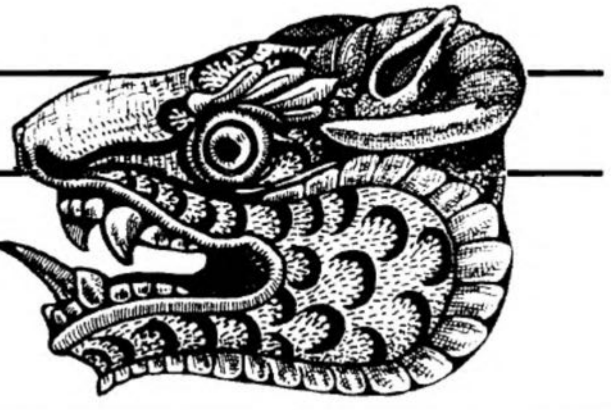
Listing 2:

```
10 ' GRAPHICS PUT WITHOUT GET
20 PCLEAR 4: PMODE 4
30 HSIZE=8:  VSIZE=7
40 DIM LL(HSIZE*VSIZE/40+1)
50 GOSUB 100
60 PCLS:      SCREEN 1,1
70 X=100:     Y=100
80 PUT(X,Y)-(X-1+HSIZE,Y-1+VSIZE
),LL,PSET
90 GOTO90
100 'SYMBOL SETUP SUBROUTINE
110 I=0: T$="      ": P=0   'MUST
USE VARIABLES BEFORE VARPTR!
120 P=VARPTR(LL(0))
130 FOR I=0 TO HSIZE*VSIZE/8-1
140 READ T$
150 POKE P+I,VAL("&H"+T$)
160 NEXT
170 RETURN
180 'LUNAR LANDER SYMBOL
190 DATA 3C,7E,7E,7E,24,24,66
```

ADVENTURE CONTEST WINNER

16K ECB | 32K ECB

THE HEAD of the BEAST



Program By Mark Nelson

Deep in thought, Jon stepped through the doorway of the small, moss-covered cabin. The cold, damp fog fled before him as the warm air inside burst out into the night. He could barely distinguish the edge of the forest that surrounded him, but he didn't need to see it clearly to know where every tree stood, where every vine hung. Wilmouth Forest was the only world he had ever known, and he knew it well.

The cool night air and the familiar sight of the forest brought a torrent of memories rushing to the surface of his troubled mind - memories that he had been able to put aside for the past two years. Had it been that long? Had it been two whole years since he had last hunted game in the forest at his father's side - since his mother and father had walked happily into the forest and never returnerd? What unspeakable fate had occured to them? It was difficult to examine these questions without tears welling up in the young man's eyes.

But it was time, and time past for such examination. The world that Jon once knew in the peaceful confines of Wilmouth Forest existed on longer. It had become foul and obscene. Where once roamed brave and noble knights along the byways, now skulked only base criminals pretending to be knights they had once cowered before. Where once young maidens had gathered mushrooms from the dawn-streaked, dew-stippled forest floor, hags now collected dung and rotting matter for their vile concoctions. And where once could be found hart and boar to rival in the land, only the most unearthly creatures were now reported by those fortunate enough to see them in time to escape unscathed.

Evil begets evil; so it is and so it has always been. And all of the evil now ensconced in the boughs of the woods could be traced to a single cause: The coming of The Beast to Wilmouth Forest. From that day hence, joy and peace were known no more in the Kingdom of Daethon.

Jon knew of The Beast only by the terrified tales of those who happened to stumble into the clearing surrounding his hut, clawed and mangled and beyond wit's bounds. He himself had scarcely gone beyond the outlying trees since his parents had fallen victim to the shadows beyond, and not once had he ventured outside the hut after nightfall. But this fact alone spoke no ill of the young man. The tales and the sight of those unfortunate souls were enough to paralyze even the stoutest heart.

Tonight, though, a new resolve had come over Jon in his solitude. Shaking off the bonds of fear, he had thrown open the door of the hovel and taken the first brave steps into the night. No more would he cower before the fire, held prisoner by an unseen and nameless terror. And no more would he allow the death of his parents to go unavenged.

As Jon turned to go inside, he heard something come crashing through the undergrowth just beyond the first stand of trees. Reaching for his knife, he saw a horse and rider bound into the clearing and twirl once before the mount could be brought to rein.

Mustering his courage, Jon stepped into the shaft of light emanating from the doorway. With knife drawn, he stood before the lathering steed. "Hail, fellow," he challenged. "If thee come for solace, then well met. But if thee have mischief in mind, prepare thyself."

"I seek no mischief," the rider responded in shaken tones. "Sheathe thy blade and bid welcome to a messenger of the king."

"What herald could be of such import as to bring thee through this evil forest at night?" Jon queried as he grabbed the reins and cautiously helped the rider dismount. "Does the King of Daethnon care no more for thee and thy fellows than to spend thy lives for a message?"

"Well met, indeed," replied the rider, "and well spoken. Were that I were in thy charge instead. But mine sire is not so callous. These evil times bode ill for all men, whether they ride or hide. Dangerous times warrant dangerous acts."

"Verily, thou dost echo mine own thoughts," said Jon. "Speak then thy message, herald, that I may judge its import for mineself."

Reaching into the leather pouch swung over his shoulder, the rider pulled forth a rolled parchment, tied in the center with a golden ribbon that caught the dim starlight, flashing brightly. Unrolling the document, he rose to his full stature and read: "Know ye people throughout the realm that by these presents, His Majesty, the King of Daethnon, doth hereby proclaim that whosoever slayeth the beast that dwelleth in the Forest of Wilmouth, and further, that bringeth the head of this beast before His Royal Highness, shalt be rewarded for his service to the crown the hand of the Princess Shera in marriage and one quarter of all the lands of the Kingdom of Daethnon."

Awestruck, Jon stumbled against the side of the horse before clutching the stirrup and catching his balance. It was an omen, he thought — an omen of fortune in a time where the word no longer held meaning.

"I accept this charge," he bellowed, "for surely, it doth suit my purpose. Verily, I shalt seek out this demon and send it back to the netherworld from which it was spawned."

Whirling around, Jon strode into the cabin, grabbed his pack and bow, and extinguished the tiny lamp on the table. Without a glance, he walked back through the doorway for the last time and headed for the clearing's edge.

As the morning light began filtering over the treetops from the east, the messenger called after him as he disappeared into the brush. "Well met, indeed, sire, and may we soon meet again. Know ye that the prayers of all the realm go with thee."

Loading and Playing Instructions

The Head of the Beast is a fairly unusual Adventure in that a joystick is used to enter all commands. Originally intended for 32K Extended BASIC, it is easily adapted to 16K ECB, with the only sacrifice being the game-save routine.

If you are entering the program from the listing into a 32K or 64K ECB machine, no changes need to be made. Simply save the game with *CSAVE "BEAST"*. Afterward, the program can be loaded with *CLOAD* and no other commands are necessary.

To enter the program from the listing into a 16K ECB machine, first enter *POKE 25,6*. Then omit Lines 3000-3070 as you type in the listing and add the following lines:

```
5 CLEAR 500
3000 RETURN
```

The program can then be saved in the usual manner. Thereafter, when loading the game, enter *POKE 25,6* before *CLOADing*.

If you subscribe to RAINBOW ON TAPE, the program can be loaded into 16K ECB by first entering *POKE 25,6* and then *CLOAD"BEAST"*. After the program has loaded, type:

```
DEL 3000-3070
3000 RETURN
5 CLEAR 500
```

The program is now modified to run in 16K of RAM and should be saved on tape before playing. As before, *POKE 25,6* must be entered each time thereafter before loading.

To play *The Head of the Beast*, you must have a joystick plugged into the right joystick port. The commands are selected by moving the cursor with the joystick until the desired command is highlighted and then pressing the fire button. When a verb is selected, the words on the screen are replaced by a group of nouns. Movement is accomplished in the same manner, with permissible directions surrounded by black bars.

If you are using the 32K version with the game-save routine, you may save your game position by moving the cursor to the word *TAPE* and pressing the joystick fire button. Then move to the word *SAVE*, press the fire button, and press the Play and Record buttons on the cassette recorder. When prompted, enter a filename of up to eight letters and hit ENTER. The game can then be loaded in the same manner, returning to the position in the game where the save was made.

You're now ready to engage The Beast. And remember, in this Adventure by Mark Nelson, a princess awaits your triumph.

— Kevin Nickols

130	 12	3110	 165
1002	 189	3270	 245
1550	 230	8000	 43
2510	 171	19000	... 240
2720	 94	50035	... 237
2950	 248	END	 141

The listing:

```
5 CLEAR 15ØØ
1Ø CLS6:C$=CHR$(223):PRINT@1Ø4,"
head"C$"of"C$"the"C$"beast";:GOS
UB18ØØØ:PRINT@392,"(C)COPYRIGHT
1984";:PRINT@427,"MARK NELSON";
1ØØ DIM V$(25),VR$(25),D$(44),SE
$(44),OT$(1Ø),L$(5,5),N$(5,5),NR
$(5,5),NP(5,5),CA$(13),HY(14),HX
(14)
1Ø5 BL$=STRING$(32," "):BT=6528Ø
:B1=126:B2=254:L=2:Z=1:CA$(1)="P
ACK":CA$(2)="BOW":CA$(3)="ARROW"
:CA=3:SL$=STRING$(32,223):LH=18:
VR$="light"
11Ø FOR X=1TO25:READ V$(X):NEXT:
GOSUB18ØØØ:FORX=1TO25:READVR$(X)
:NEXT:FORX=1TO25:READP:P$=P$+CHR
$(P):NEXT:GOSUB18ØØØ:FORY=1TO5:F
ORX=1TO5:READNP(X,Y):NEXTX,Y:FOR
X=1TO44:READD$(X):NEXT:FORX=1TO4
4:READSE$(X):NEXT:FORX=1TO1Ø:REA
DOT$(X):NEXT
12Ø GOSUB18ØØØ:FORX=1TO44:READN,
S,E,W:NO$=NO$+CHR$(N):S$=S$+CHR$
(S):E$=E$+CHR$(E):W$=W$+CHR$(W):
NEXT:GOSUB18ØØØ:FORY=1TO5:FORX=1
TO5:READL$(X,Y):NEXTX,Y:GOSUB18Ø
ØØ:FORY=1TO5:FORX=1TO5:READN$(X,
Y):NEXTX,Y:FORY=1TO5:FORX=1TO5:R
EADNR$(X,Y):NEXTX,Y
13Ø C$=CHR$(128):CLS
2ØØ GOSUB1ØØØ:GOSUB3Ø5:GOSUB9ØØØ
:GOSUB4ØØ:GOSUB9ØØØ:GOSUB15ØØØ:G
OTO2ØØ
3Ø5 GOSUB5ØØØ:PRINT@32Ø,SL$;
31Ø FOR X=1TO25:PRINT @ASC(MID$(
P$,X))+3ØØ,V$(X);:NEXT:IF ASC(MI
D$(NO$,L)) THENPRINT@ASC(MID$(P$
,3))+299,C$"N"C$;
313 IF ASC(MID$(S$,L)) THENPRINT
@ASC(MID$(P$,13))+299,C$"S"C$;
315 IF ASC(MID$(E$,L)) THENPRINT
@ASC(MID$(P$,9))+299,C$"E"C$;
317 IF ASC(MID$(W$,L)) THENPRINT
@ASC(MID$(P$,7))+299,C$"W"C$;
32Ø PRINT@ASC(MID$(P$,LH))+3ØØ,V
R$;
35Ø GOSUB7ØØØ:HL=(JY-1)*5+JX:IFH
L=2ORHL=4ORHL=8ORHL=12ORHL=14ORH
L=LH THEN36ØELSEPRINT@ASC(MID$(P
$,HL))+3ØØ,VR$(HL);:PRINT@ASC(MI
D$(P$,LH))+3ØØ,V$(LH);:LH=HL
36Ø X=PEEK(BT):IF X=B1 OR X=B2 T
HEN VR$=VR$(LH):RETURN ELSE35Ø
4ØØ PRINT@336-LEN(V$(LH))/2,VR$;
41Ø ON LH GOTO 111Ø,,131Ø,,151Ø,
1610,1710,,1910,2020,2110,,2310,
,2510,2610,2710,2810,2910,3000,3
110,3210,3310,3400,3510
1000 PRINT @0,"I AM ";:IFD$(L)="
C"THEN PRINT"IN A CAVE"ELSEIFD$(
L)="T"THENPRINT"ON A TRAIL"ELSE
IFD$(L)="W"THEN PRINT"ON THE WES
T SHORE"ELSE IFD$(L)="E"THEN PRI
NT"ON THE EAST SHORE"ELSE PRINTD
$(L)
1002 IFD$(L)="C"THEN1040ELSECV=0
1005 PRINT@32,"I SEE: ";:IF SE$(
L)="" THEN PRINT"NOTHING OF INTE
REST" ELSE PRINT SE$(L)
1030 PRINTSL$;:RETURN
1040 IFMA>1ORCN>2THENCV=0:GOTO10
05ELSEPRINT"IT'S TOO DARK TO SEE
.":CV=1:GOTO1030
1110 GOSUB10000:FORX=1TO10:IF N$
=OT$(X)THEN 1115 ELSE NEXT:GOTO1
2000
1115 FORX=1TOCA:IFCA$(X)=N$THENP
RINT"YOU ALREADY HAVE THE "N$:RE
TURNELSENEXT
1120 CA=CA+1:CA$(CA)=N$:PRINT N$
" TAKEN.":SE$(L)="":RETURN
1310 GOSUB5050:N=ASC(MID$(NO$,L)
):IF N THENL=N:RETURN ELSE2060
1510 GOSUB10000:PRINTL$(HX(HX),H
Y(HX)):IFN$="PACK"THEN5110ELSEIF
N$="INN"THENSE$(L)="DOOR":RETURN
ELSE IFN$="TABLE"THENSE$(L)="CAN
DLE":RETURN ELSEIFPEEK(1120)=96
THENPRINT@96,"I SEE NOTHING SPEC
IAL."
1520 RETURN
1550 PRINT@NP(X,Y),N$(X,Y);:C=C+
1:HY(C)=Y:HX(C)=X:RETURN
1610 GOSUB10000:FORX=1TOCA:IF N$
=CA$(X) THEN1620ELSENEXT
1618 PRINT"YOU DON'T HAVE THE "N
$".":RETURN
1620 IFN$="PACK"THEN1640 ELSE DR
$=N$:PRINT DR$;" DROPPED.":GOSUB
6000:IF SE$(L)<>DR$ THEN PRINT"A
 THIEF SUDDENLY APPEARS AND     S
TEALS THE "N$"."
1630 RETURN
1640 DR$=N$:GOSUB6000:RETURN
1710 GOSUB5050:W=ASC(MID$(W$,L))
:IF W THENGOSUB14000 ELSE2060
1720 L=W:RETURN
1910 GOSUB5050:E=ASC(MID$(E$,L))
:IF E THENGOSUB14000 ELSE2060
1920 L=E:RETURN
2020 GOSUB10000:IF N$="INN"THEN
2030 ELSE IF N$="SHED"THEN2040 E
LSE IF N$="BOAT"THENL=13:RETURN
ELSEIFN$="CAVE"THEN 2055ELSE2060
2030 IF B=1 THEN L=16:RETURN:ELS
E11000
2040 IFL=5THENL=4:RETURNELSE2060
2055 GOSUB5050:PRINT"THE BOAT DR
IFTS AWAY.":IFL=40THENL=39
2056 RETURN
2060 PRINT"YOU CAN'T GO THERE.":
RETURN
2110 GOSUB10000:IF N$="KNIGHT"TH
EN2120 ELSE IF N$="SNAKE"THEN213
0 ELSE IFN$="BEAST"THEN 2150ELSE
12000
2120 PRINT"YOU ATTACK THE KNIGHT
. HE PUSHESYOU OFF OF THE BRIDGE
 AND YOU   FALL 1000 FEET TO YOU
R DEATH.":GOTO2155
2130 PRINT"YOU ATTACK!  THE SNAK
E BITES BE-FORE IT DIES.":S=1:SB
=3:SE$(43)="DEAD SNAKE":L$(5,5)=
"THE 'GULLET SNAKE' LIES DEAD.":
RETURN
2150 PRINT"YOU ATTACK THE BEAST!
  THE BEASTSEEMS TO BE SMILING A
S HE PICKS YOU UP AND EATS YOU A
LIVE!"
2155 PRINT@320,SL$;:GOSUB5000:PR
INT@293,"YOUR ADVENTURE IS OVER"
;
2158 JX=JOYSTK(0):IFJX>33 THENPR
INT@424,"PLAY";:PRINT@434,"quit"
;:GOTO2160
2159 PRINT@424,"play";:PRINT@434
,"QUIT";
2160 X=PEEK(BT):IFX=B1 ORX=B2 TH
EN2165ELSE2158
2165 IF JX<33 THEN RUN ELSE CLS:
END
2310 GOSUB5050:SO=ASC(MID$(S$,L)
):IF SO THENL=SO:RETURN ELSE2060
2510 IFL=3ORL=5 THEN CA=CA+1:CA$
(CA)="ROCKS"
2512 GOSUB10000:IFL=3ORL=5 THENC
A$(CA)="":CA=CA-1
2515 IFN$="ROCKS"THEN2530ELSEIFN
$="ROPE"THEN2550ELSE12000
2530 IFL=8 THENL=3 ELSEIFL=10 TH
ENL=5ELSE IFL=3 THENL=8ELSE IFL=
5 THENL=10
2540 RETURN
2550 IFRP=0THEN11000ELSEIFKN=1TH
EN2570
2560 PRINT"AS YOU CLIMB OUT ON T
HE ROPE,   THE KNIGHT GRABS HIS
SWORD AND  SLICES THE ROPE.  YOU
 FALL 1000 FEET TO YOUR DEATH.":
GOTO2155
2570 PRINT"YOU CLIMB THE ROPE OV
ER THE CAN-YON.":DR$="ROPE":GOSU
B6000:IF L=23 THEN L=22 ELSE L=2
3
2580 RETURN
```

```
2610 GOSUB19000:GOSUB13000:GOSUB
19010:IFN1$="DOOR"OR N1$="BRIDGE
"THEN 2630
2620 PRINT"THAT DOESN'T WORK.":R
ETURN
2630 IFN$="AXE"THEN2640ELSE2620
2640 IFN1$="DOOR"THEN PRINT"THE
DOOR SWINGS OPEN.":MID$(W$,L,1)=
CHR$(16):L$(2,5)="IT'S OPEN.":SE
$(L)="OPEN DOOR":RETURN
2650 PRINT"THE KNIGHT GRABS HOLD
 OF THE     BROKEN BRIDGE AS IT S
WIN3S DOWN AND SMASHES AGAINST T
HE CANYON  CLIFF!  HE FALLS TO H
IS DEATH.":KN=1:SE$(L)="BROKEN B
RIDGE":RETURN
2710 GOSUB10000:IF N$="ARROW" TH
EN FOR X=1TOCA:IF CA$(X)="BOW"TH
EN2770 ELSE NEXT:GOTO11000
2720 IF N$="BOW"THEN FORX=1TOCA:
IF CA$(X)="ARROW"THEN 2770 ELSE
NEXT:GOTO11000:RETURNELSE12000
2770 IF L=23 THEN 2775ELSE2780
2775 PRINT"THE ARROW WHIZZES THR
OUGH THE    AIR AND HITS A TREE O
N THE OTHERSIDE OF THE CANYON.
";:IF TI=1 THEN PRINT"THE ROPE I
SSTRETCHED ACROSS THE CANYON TIE
DTO THE ARROW.":AR=1:SE$(L)="BRI
DGE, ROPE, KNIGHT"
2777 GOTO2783
2780 PRINT"YOU HIT NOTHING."
2783 DR$="ARROW":GOSUB6000:RETUR
N
2810 GOSUB10000:IF N$="MATCH"THE
N2830 ELSE IF N$="CANDLE"THEN284
ØELSE12000
2820 IFN$="MATCH"THENPRINT"YOU C
AN'T LIGHT A MATCH TWICE.":RETUR
NELSE12000
2830 IF MA>1 THEN PRINT"THE MATC
H IS ALREADY LIT.":RETURN:ELSEIF
ML=1 THEN2820 ELSE MA=5:PRINT"TH
E MATCH IS NOW LIT.":ML=1:RETURN
2840 IF MA>1 THEN CN=50:PRINT"TH
E CANDLE IS NOW LIT.":CL=1:RETUR
N:ELSE 11000
2910 GOSUB10000:IF N$="SNAKE"THE
N2950 ELSEPRINT"YUUUUCK!":PRINT"
I REFUSE TO EAT THE "N$".":RETUR
N
2950 IF S=1 THEN PRINT"YOU FEEL
BETTER.  THE SNAKE MUSTHAVE BEEN
 AN ANTIDOTE FOR ITS    OWN VENOM
.":SB=-1:SE$(L)="":RETURN
2960 PRINT"YOU PICK UP THE SNAKE
 AND TRY TODEVOUR IT ALIVE!  THE
 SNAKE       STRIKES.  YOU FEEL DI
ZZY!  YOU  ARE DEAD.":GOTO 2155
3000 GOSUB5000:PRINT@424."LOAD":
PRINT@434,"SAVE"
3010 JX=JOYSTK(Ø):IFJX>33THENPRI
NT@424,"LOAD";:PRINT@434,"save";
:GOTO3030
3020 PRINT@424,"load";:PRINT@434
,"SAVE"
3030 X=PEEK(BT):IFX=B1 ORX=B2 TH
EN3040 ELSEIFINKEY$=" "THEN200EL
SE3010
3040 GOSUB5000:IFJX>33THEN3060
3045 PRINT@334,"load";:GOSUB1600
Ø
3050 OPEN"I",#-1,A$:FORX=1TO44:I
NPUT#-1,SE$(X):NEXT:FORX=1TO10:I
NPUT#-1,CA$(X):NEXT:INPUT#-1,L,C
A,MA,CN,CO,B,S,CL,ML,BS,S,HT,W,L
$(2,5),SB,TI,TB,AR:CLOSE#-1:MID$
(W$,17)=CHR$(W):GOTO200
3060 PRINT@334,"save";:GOSUB1600
Ø
3065 OPEN"O",#-1,A$:FORX=1TO44:P
RINT#-1,SE$(X):NEXT:FORX=1TO10:P
RINT#-1,CA$(X):NEXT:PRINT#-1,L,C
A,MA,CN,CO,B,S,CL,ML,BS,S,HT,ASC
(MID$(W$,17)),L$(2,5),SB,TI,TB,A
R:CLOSE#-1:GOTO200
3110 GOSUB10000:IF N$="ROPE"THEN
3120ELSE12000
3120 PRINT@320,SL$;:PRINT@332,"t
o"CHR$(223)"what";:QU=Ø:GOSUB100
40:IFN$="BRIDGE"THEN3140ELSE IFN
$="ARROW"THEN3130 ELSE PRINT"YOU
 CAN'T TIE THE ROPE TO THE":PRIN
T N$".":RETURN
3130 PRINT"THE ROPE IS TIED TO T
HE ARROW.":TI=1:RETURN
3140 PRINT"THE ROPE IS TIED TO T
HE BRIDGE.":TB=1:RETURN
3210 GOSUB10000:FORX=1TO10:IF OT
$(X)=N$THEN3230 ELSENEXT:GOTO120
ØØ
3230 FOR X=1TOCA:IF CA$(X)=N$ TH
EN 3250 ELSE NEXT
3240 PRINT"YOU DON'T HAVE THE ";
N$;".":RETURN
3250 DR$=N$:GOSUB6000:IF N$="SPE
AR"THEN 3260 ELSE PRINT"YOU THRO
W THE "N$".":RETURN
3260 IF L=15 THEN 3270 ELSE PRIN
T"YOU THROW THE SPEAR.  IT FLIES
  SWIFTLY AND SMOOTHLY THROUGH T
HEAIR.":RETURN
3270 PRINT"YOU THROW THE SPEAR A
T THE BEASTWITH DEADLY ACCURACY!
  THE SPEARLODGES HIGH IN THE BE
ASTS CHEST.THE BEAST FALLS TO TH
E GROUND    NEAR DEATH!":BS=1:RET
URN
3310 GOSUB10000:IF N$="DOOR"THEN
11000ELSE IFN$="PACK"THENPRINT"T
```

```
HE PACK IS OPEN."
3330 RETURN
3400 GOSUB10000:IFN$="BOAT"ORN$=
"OAR"THEN3410ELSE12000
3410 FORX=1TOCA:IFCA$(X)="OAR"TH
EN3420ELSENEXT:GOTO11000
3420 IFLEFT$(D$(L),9)="IN A BOAT
"THEN PRINT"THE OAR SLIPS OUT OF
 YOUR HANDS AND DRIFTS DOWN RIVE
R.":DR$="OAR":GOTO6000ELSEPRINT"
YOU FEEL LIKE AN IDIOT AS YOU
START TO ROW ON DRY LAND."
3499 RETURN
3510 GOSUB13000:IFN$="SWORD"ANDN
1$="BEAST"THEN3520 ELSE2620
3520 IFBS=1THEN PRINT"YOU CUT OF
F THE HEAD OF THE      BEAST!  YO
U'VE DEFEATED HIM!":GOTO20000ELS
EPRINT"YOU LAND A BLOW WITH YOUR
 SWORD.":HT=HT+1:IFHT>3THENPRINT
"THE BEAST PICKS YOU UP AND IM-
 PALES YOU ON A STALAGTITE. OR I
S";
3530 PRINT"IT STALAGMITE?  YOU N
EVER COULD REMEMBER.":GOTO2155
5000 FORAZ=352TO448STEP32:PRINT@
AZ,BL$;:NEXT:PRINT@479,BL$;:RETU
RN
5050 FOR X=96TO288STEP32:PRINT@X
,BL$;:NEXT:PRINT@96,"";:RETURN
5110 GOSUB5050:PRINT"I HAVE IN M
Y PACK:"
5120 FOR X=2 TO CA:PRINT CA$(X),
:NEXT:PRINT:RETURN
6000 IFDR$="PACK"THENGOSUB12000:
PRINT"THE PACK IS STRAPPED ON.":
GOTO200
6005 FOR X=2TO CA:IF CA$(X)=DR$
THEN CA$(X)="":FOR A=X TO CA:CA$
(A)=CA$(A+1):NEXTA:CA$(CA)="":CA
=CA-1:IFSE$(L)=""THENSE$(L)=DR$:
RETURN
6010 NEXTX:RETURN
7000 JX=INT(JOYSTK(0)/12):JY=INT
(JOYSTK(1)/12):IF JX=0 THENJX=1
7005 IFJY=0THENJY=1
7006 RETURN
7010 LO=25:FOR X=1TOC:P=PEEK(BT)
:IFP=126 ORP=254 THEN10070 ELSEI
F INKEY$=" "THEN200 ELSE TX=ABS(
HX(X)-JX):TY=ABS(HY(X)-JY):IF TX
+TY<LO THEN LO=TX+TY:T2=X
7020 NEXT:T1=T2:RETURN
8000 PRINT@NP(HX(HX),HY(HX)),N$(
HX(HX),HY(HX));:PRINT@NP(HX(T1),
HY(T1)),NR$(HX(T1),HY(T1));:HX=T
1:RETURN
9000 X=PEEK(BT):IFX=127ORX=255TH
ENRETURNELSE9000
10000 QU=0:C=0:HX=0:GOSUB5000
10005 IFCV=1THENPRINT@96,"IT'S T
OO DARK TO "V$".":GOTO200
10010 FORX=1TO5:FORY=1TO5:FORA=1
TOCA:IFN$(X,Y)=CA$(A)THENGOSUB15
50
10020 NEXTA:IFN$(X,Y)=RIGHT$(SE$
(L),LEN(N$(X,Y))) ORN$(X,Y)=LEFT
$(SE$(L),LEN(N$(X,Y))) THENGOSUB
1550
10030 NEXT Y,X
10040 IFQU=1 THEN GOSUB5050:RETU
RN ELSEGOSUB7000:GOSUB7010
10050 IF T1=HX THEN10040ELSEGOSU
B8000
10055 GOTO 10040
10070 GOSUB5050:N$=N$(HX(HX),HY(
HX)):QU=1:RETURN
11000 PRINT"YOU CAN'T DO THAT NO
W.":RETURN
12000 PRINT"YOU CAN'T "V$(LH)" T
HE "N$".":RETURN
13000 GOSUB10000:N1$=N$:PRINT@33
1,"with"CHR$(223)"what";:QU=0:GO
SUB10040:RETURN
14000 IFL=43ANDS=0 THENPRINT"THE
 SNAKE BITES AND WON'T LET   YOU
 PASS.":SB=3:RETURN
15000 CN=CN-1:MA=MA-1:IFMA=1THEN
PRINT"YOUR MATCH WENT OUT."
15010 IFCN=1THENPRINT"YOUR CANDL
E WENT OUT."
15015 IF TB=1 AND TI=1 AND AR=1
THEN RP=1
15020 SB=SB-1:IFSB=0THENPRINT"YO
U'VE DIED FROM THE SNAKE BITE.":
GOTO2155
15030 IFL=13 THENL=19ELSE IFL=19
 THENL=33 ELSE IFL=33 THENL=40:G
OTO15050ELSE RETURN
15040 PRINT"YOU ARE DRIFTING.":R
ETURN
15050 IFB=0THENPRINT"YOUR BOAT H
AS HIT GROUND.":B=1
15060 RETURN
16000 PRINT@361,"READY CASSETTE"
:PRINT@393,"PRESS ANY KEY":IFINK
EY$=""THEN16000ELSEGOSUB5000:PRI
NT@352,"";:INPUT"ENTER FILE NAME
";A$:RETURN
18000 R=R+1:PRINT@199,"JOYSTICK"
C$"CONTROLLED";:PRINT@268,"ADVEN
TURE";:IFR>3 THENR=1
18005 ON R GOTO 18010,18020,1803
0
18010 PRINT@199,"joystick";:RETU
RN
18020 PRINT@208,"controlled";:RE
TURN
18030 PRINT@268,"adventure";:RET
URN
```

```
19000 IFL=23THEN CA=CA+1:CA$(CA)
="BRIDGE"
19001 RETURN
19010 IFL=23THEN CA$(CA)="":CA=C
A-1
19011 RETURN
20000 GOSUB5000:PRINT@160,"YOU B
RING THE HEAD TO PRESENT TOTHE K
ING.  THE KING SPEAKS .... 'YOU
HAVE DEFEATED THE BEAST.   YOU S
HALL MARRY THE PRINCESS     SHERA
 AND SHALL BE CALLED MY SONTO ON
E DAY BE KING.'  YOU LIVE  HAPPI
LY EVER AFTER."
20005 PRINT@39,"DEAD, HEADLESS B
EAST";
20010 PLAY"O2L2FL3B-L8B-L1B-P8L2
FL3O3CL8O2AL1B-P8L2FL3B-L8O3E-L2
E-L3DL8CO2L3B-O3L16CO2L16B-L3AL8
B-L2O3C":FORX=1TO500:NEXT
20020 FORX=1TO2:FORA=1TO1000:NEX
T:PLAY"O4L2CO3L3BL8F#L4AGFDL64CD
CDCDCDCDCDCDCDCDCDCL32O2BO3CL4DL
6O2GL16O3DL6EP8O2L8CEGO3CEGO4L2C
L3O3BL8F#L4AGFDL64CDCDCDCDCDCDCD
CDCDCDCDCDCBCL8EP8L8DL16EL2DO3L8
C":NEXT
20030 GOTO20030
50000 DATA GET,"",N,"",LOOK,DROP
,W,*,E,GO,KILL,"",S,"",CLIMB,BRE
AK,SHOOT,LIGHT,EAT,TAPE,TIE,THRO
W,OPEN,ROW,CUT
50005 DATA get,"",n,"",look,drop
,w,*,e,go,kill,"",s,"",climb,bre
ak,shoot,light,eat,tape,tie,thro
w,open,row,cut
50020 DATA 53,59,67,71,77,85,95,
99,103,109,117,123,131,135,141,1
49,155,161,167,173,181,187,193,1
99,205
50025 DATA 352,359,365,371,377,3
84,391,397,403,409,416,423,429,4
35,441,448,455,461,467,473,480,4
87,493,499,505
50030 DATA IN A FOREST,T,AT THE
TOP OF THE FALLS,IN AN OLD STORA
GE SHED,AT THE TOP OF THE FALLS,
IN A FOREST,T,AT THE FOOT OF THE
 FALLS,"",AT THE FOOT OF THE FAL
LS,T,W,IN A BOAT ON A RIVER
50032 DATA E,C,IN A SMALL INN,EA
ST OF AN INN,W,IN A BOAT ON A RI
VER
50035 DATA E,C,IN A DARK FOREST,
ON A BRIDGE OVER A CANYON,T,T,W,
ON A BRIDGE,E,C,IN A DARK FOREST
,IN THE MOUTH OF A CAVE
50040 DATA  C,IN A BOAT IN A CAN
YON,C,C,C,C,C,IN THE MOUTH OF A
CAVE,IN A BOAT ON A LAKE,C,C,C,C
50050 DATA MATCH,"",SWORD,AXE,SH
ED,"","",ROCKS,"",ROCKS,"",OAR,B
OAT,BOAT,THE BEAST,TABLE,INN,RIV
ER,BOAT,RIVER,"","",BRIDGE AND K
NIGHT
50060 DATA "",ROPE,BRIDGE,"",BRI
DGE,"",SHIELD,"","",BOAT,"","","
","","","",CAVE,"","",SNAKE,SPEA
R
50070 DATA MATCH,SHIELD,ROPE,OAR
,CANDLE,AXE,SPEAR,BOW,ARROW,SWOR
D
50080 DATA ,6,,,,7,,,,,,,,,5,,,,
,,1,,7,,2,11,,6,,12,,,,,,,,14,,,
7,17,12,,8,18,,11,,,,,10,20,,,,2
1,,,,,17,,11,25,,,12,26,,,,,,,14
,28,,,15,29,,,,30,,,,,24,,,31,,2
3,17,,,,18,,27,,,,28,26,20,,,27,
21,34,,,22,35,,,24,,32,,,38,,31,
0
50090 DATA 0,,,29,,35,,30,,36,34
,,,,35,,,38,,32,43,39,37,,,,38,,
,,,34,,,,,,43,,38,,44,42,,,,43
50100 DATA "","","",IT'S VERY ST
RAIGHT AND SHARP,SALIVA DRIPS OU
T OF ITS MOUTH   AND DOWN ITS CH
IN AS IF IT SEES A DELICIOUS MEA
L!
50110 DATA "",IT'S A SMALL ONE M
AN BOAT,"",THEY LEAD UP TO THE T
OP OF THE  FALLS,THE ENTRANCE LE
ADS INTO DARKNESS,"","",IT'S VER
Y HEAVY AND STRONG BUT  DULL
50120 DATA "","","","",IT'S A FI
NELY CRAFTED SPEAR      DESIGNED
FOR HUNTING
50130 DATA ON THE TABLE IS A CAN
DLE,THERE IS A DOOR.,HE WON'T LE
T YOU GET ACROSS THE BRIDGE.  HE
 LOOKS SEVEN FEET     TALL IN HIS
 FULL ARMOR.,THE DOOR IS BOLTED
SHUT. A SIGN ON THE DOOR READS '
CONDEMNED BY ORDER OF THE KING'
50135 DATA IT IS THE BOW YOUR FA
THER GAVE  YOU WHEN YOU WERE A Y
OUNG MAN.  YOU FONDLY REMEMBER H
UNTING      TRIPS IN WILMOUTH FOR
EST.,""
50136 DATA IT IS THE RARE 'GULLE
T SNAKE'
50140 DATA SHIELD,ROPE,MATCH,ARR
OW,BEAST,CANDLE,BOAT,OAR,ROCKS,C
AVE,BRIDGE,PACK,AXE,RIVER,SWORD,
!,SHED,SPEAR,TABLE,INN,KNIGHT,DO
OR,BOW,TRACK,SNAKE
50150 DATA shield,rope,match,arr
ow,beast,candle,boat,oar,rocks,c
ave,bridge,pack,axe,river,sword,
!,shed,spear,table,inn,knight,do
or,bow,track,snake
```

From P 4 ...

Dear Graham,

A few weeks ago I took the plunge and bought myself a Color Computer II. I also managed to get hold of a pile of back-dated Australian Rainbows from a friend who used to own an 80C.

What I would like to know, is:

1 - Does Australian Rainbow still exist (I hope so - the American one is mainly repeated advertisments)?

2 - Who in Australia st.:ks Telewriter-64?

3 - Who can tell me how to do a 64K upgrade on a CoCo II - all the articles are for a Model I?

I am very impressed by the enthusiasm showed in your encouragement and involvement of and in the 80C in Australia. Congratulations.

Darrell Berry
Exeter. TAS.

Darrell,

Telewriter 64 is distributed in Australia by Computerware for Micros (see ad this magazine).

Your user group is the key to all the local assistance you need. Briefly however, if you have the new short case CoCo, the easiest way to have it upgraded at present is to talk to your Tandy man. If you have the long case, then the up grade is not too difficult, and someone local there will almost certainly be able to assist you.

Thanks for the nice words,
Graham.

Dear Graham,

While I was looking at the letters in the Dec/Jan Rainbow I noticed a letter mentioning problems with the Shack's 'Screen Print' utility and it so happens we had the same problem. Luckily my dad spotted a column in the American Rainbow (Oct.'83 P 290) telling you how to upgrade the Shack's screen print utlity to 32K. At the end of my letter is complete instructions and the program to upgrade.

Step 1: You must have a copy of Radio Shack's screen print program.

Step 2: Enter the following program:

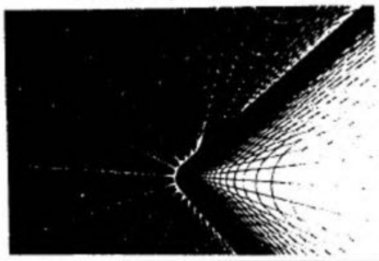
```
10 CLS
20 FOR I=15744 TO 16376
30 P=PEEK(I)
40 IF P=64 OR P=62 OR P=63THEN P=P+63
50 POKE I+16128,P
60 NEXT I
70 PRINT "DONE"
```

Step 3: With this program in your computer load the Radio Shack "Screen Print" program. Do not CLEAR 200,15743 prior to loading and do not EXEC the program.

Step 4: Run the basic program above. It will move the Screen Print program to high RAM while correcting it. Do not EXEC at this time.

Step 5: Make a recording, or two, of your new Screen Print program. To do this, set up your recorder to record and key in the following command:

```
CSAVEM"SCPRT",31872,32504,31872
```



Step 6: Test this new program. Turn off your computer and wait 15 seconds before turning it on. Rewind your tape to the beginning of the new version. Type in the following:

```
CLEAR 200,31871
CLOADM "SCPRT"
EXEC
```

And now enter this short program:

```
10 PMODE 4,1:PCLS:SCREEN 1,0
20 FOR I=1 TO 10
30 CIRCLE (RND (256)-1,RND (192)-1,RND (40))
40 NEXT I
50 GOTO 50
```

Turn on your printer. Press the BREAK key and press SHIFT (up arrow), the screen print command. The printer should print the same picture which is on your screen. If it does not, you will have to redo the entire procedure.

USING THE NEW VERSION
######################

You may use this new version of the Screen Print program just as you would use the old version except use the following memory locations:

```
Loading:           CLEAR 200,31871
                   CLOADM "SCPRT"
                   EXEC
Reverse printing:  POKE 32431,255
Normal printing:   POKE 32431,0
Subroutine:        DEFUSR0 = 31913
Saving:            CSAVEM"SCPRT",31872,32504,31872
```

I hope this set of procedures will help anyone with this problem.

James Butler. (No Address)

From P 31 ...

point to touch is the RF adapter. That is the big metal can sitting to the left, where you plug in the TV wire. Another point is one of the metal clips that hold the bottom shield to the main PCB (Printed Circuit Board). You will find these clips all around the edge of the PCB.

Now that we have seen the insides of the CoCo and are a bit more familar with its parts, let's do something to it. About the simplest thing we can do is add a pilot light. It is not hard, and if you take it one step at a time, anyone will be able to do it, and the good thing about it is that it costs less than $1. By the way, this pilot light will work on any version, not just the CoCo 2. Before you plunge into this though, if you do not have any soldering experience, practice on something else first. To do this, you will need a soldering iron. A low power, medium or fine tip soldering iron will do. The solder to use must be a rosin core and not too thick. Radio Shack sells both at a reasonable price. If you have never handled a soldering iron before, get Radio Shack's proto-board and practice on it first.

Here are the step-by-step instructions on how to solder:

1) Make sure that your soldering tip is clean and hot. A wet sponge is great to clean the tip.
2) Secure the component to the PCB.
3) Heat the component and the PCB with the iron.
4) Touch the end of the solder to the component. My personal habit is to position the solder so that it will touch the iron, component and PCB at the same time.
5) When enough solder flows, remove the solder.
6) Remove the iron from the joint.
7) Wait until it cools before moving the component or the PCB.

Figure 1

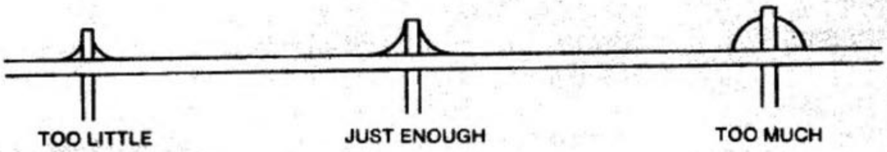
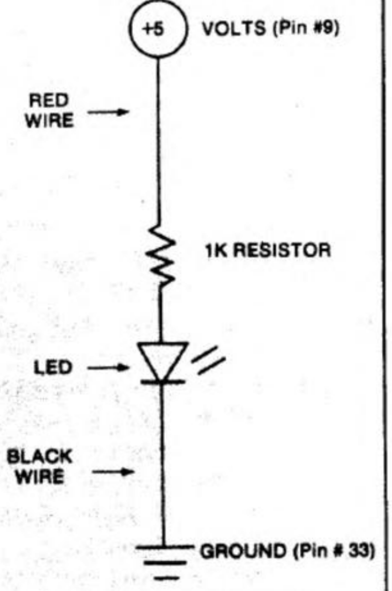


Figure 2

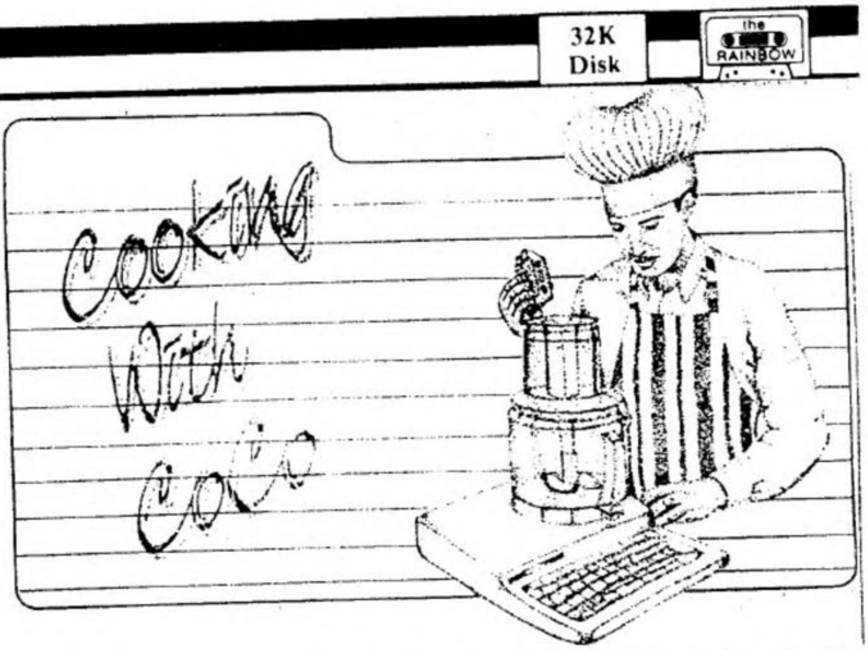


To make a good joint takes practice. To put the right amount of solder also takes practice. Too much or too little could result in a bad connection. Examine Figure 1, and notice the difference between too little and too much solder. Sometimes a bad connection can be turned into a good con-

.... Continued on P 55

EXPANDING BASIC

32K
Disk



PART IV

By Colin J. Stearman

If you were paying close attention last month, you might have noticed I included a couple of items in the patch listing which were not mentioned in the text of the article. These were put in at the last minute due to the overwhelming number of reader requests for them. Before we get started on this month's feature, I will describe what they were.

DECB 1.1

It seems more of you have the new revision of Disk BASIC than I imagined, and were frustrated by this series being based on the 1.0 revision. Well fret no more, as the part three listing contains patch addresses for both revisions. I have used MAC's conditional assembly to select which revision to assemble. If the label *REV* is zero then the 1.0 version is built and if it's one then 1.1 is built. The listing each month will be assembled for 1.0, but all information will be included regarding what to change for 1.1.

DECB 1.1 takes up more room in the ROM than does 1.0, so I have had to leave some features out. First to go is the fix to the *FILES* command. I haven't checked, but would like to think that 1.1 fixed that bug itself. Second, the fully spelled out error messages and return of the error message name in *ENAME$* had to go. These seemed like the least important, but if you disagree, leave something else out and include them. But whatever you do, don't let the additions go beyond *$DEFF*. The OS-9 boot routine resides at *$DF00* through *$DF4C*.

Finally, each month RAINBOW ON TAPE will have the machine code file for both revisions of BASIC. The name of the file will be built from the initials of the article, the part number and then V10 for *DECB 1.0* and V11 for *DECB* 1.1. So this month the files will be *CWC4V10* and *CWC4V11*.

Drive Step Rate

Many of you have disk drives that can step from track to track at a rate faster than the 30 ms (milliseconds) set by BASIC. Even my old RS drives can step at 20 ms.

If you look at last month's listing Lines 225 through 232, you will see that I adjusted the rate to 20 ms. That's why your drives sounded a little strange. If you had problems maybe you should set this back to 30 ms.

There are four possible settings; 30 ms, 20 ms, 12 ms and 6 ms. This patch will affect all your drives equally, so set the value to that of the slowest drive, if you have a mix. I have patched both the RESTORE rate and SEEK rate. The first sets the rate at which the drive is restored to track zero; the second, the rate at which each track is sought. I toyed with making a command to allow BASIC to change the rate "on the fly." But that takes up precious ROM space and you would always want the fastest rate your drives can handle. If you don't know how fast your drives are, keep reducing the rate until a *LOAD* command fails, then go back a notch.

Back to Business

Last month we ended the assembly code listing with a series of dummy functions. Next month we will add the code to make some of them functional. But this month we introduce FLEXIKEY.

Hands up all of you CoCo keyboard-pounders who have just entered a long direct command to BASIC, only to notice a "typo" in the second character. I guess I'm not alone! With FLEXIKEY you can instantly save the bad line, recall it for editing and re-execute it. You never have to type in the same thing twice. I must confess, the idea came from my IBM PC at work, which has similar functions.

FLEXIKEY

The FLEXIKEY routine completely replaces BASIC's normal keyboard entry routine and places each entered BASIC line into a buffer when you press the ENTER key. This entry is then recallable for re-execution or modification by a set of simple commands.

The best way to describe how it works is by example. Let's say you have just typed in the command

```
COPY"OLP.PGM" TO "NEW.PGM"
```

and ENTERed it. It returned an ?NE error because you meant to type *OLD.PGM*. Instead of retyping the whole line, use the right arrow key to recall each letter from the buffer. Pressing it seven times will recall

```
COPY"OL
```

with the cursor just after the 'L'. Now type in the 'D'. This replaces the incorrect 'P'. You could get the rest of the line out by repeatedly pressing the right arrow, but if you press SHIFT/right arrow the remainder of the line appears, with the cursor at the end. If you were to press ENTER, then this line would be put into the holding buffer and executed also.

But let's say that just as you were about to press ENTER you realized that the proper program name was *VERY OLD.PGM.* You could press ENTER anyway and get another error and then edit again, but if you press SHIFT/@ the command line will be stored in the buffer without execution, ready for further editing. When you do this a '@' is displayed at the end of the line to remind you that the command was just stored and not executed.

So you do this and then press the right arrow five times to recall *COPY"*. To insert the *VERY*, press the SHIFT/up arrow. This puts you into the insert mode and each character typed will be inserted in the command line, with the remaining characters in the buffer not overwritten. The overtype mode is returned whenever you press a left, right or down arrow key. Once *VERY* is typed, the SHIFT/right arrow key will recall the remainder of the line for entry.

But once again you get an ?NE error because the name of the file was really *VERY.PGM* (will you ever get it right?). Press the right arrow key nine times until *COPY"VERY* is displayed. Now press the down arrow key three times, once for each letter in *OLD*. SHIFT/right arrow will then spit out the rest of the line which now reads

```
COPY"VERY.PGM" TO "NEW.PGM"
```

If you are editing a line and things get really scrambled, don't worry, just hit left arrow to delete the character to the left of the cursor. The original character at that position is still in the buffer and could be pulled out with right arrow. If the whole line is messed up, press SHIFT/left arrow and the whole thing will disappear. But the original line is still in the buffer so you can start all over.

Some of the arrow keys now used by FLEXIKEY previously created printable characters (square brackets, left arrow and the like). To get these now, press SHIFT/CLEAR and then the arrow key you want. The normal character will appear. To get the back slash which SHIFT/CLEAR normally produces, press SHIFT/CLEAR twice.

FLEXIKEY does not interfere with the normal operation of BASIC's *EDIT* command. It works in the command mode and also within BASIC programs when entry is via an *INPUT* command. Also, some machine language programs use BASIC's entry routine, and therefore FLEXIKEY is available for use within them also. (Computerware's MACRO assembler MAC falls into this category, for one.)

The buffer used by FLEXIKEY is the cassette buffer, so correct operation will not occur immediately after cassette input/output operations. It does not interfere with this I/O, it's just that they share a common buffer area.

As I said earlier, once you get used to remembering FLEXIKEY is there, you'll wonder how you ever managed without it.

Adding The New Functions

This is a simple process using your editor. Call in last month's listing and make the following changes using the [*REF#*] given as a locating guide. 'Uncomment' (remove the initial asterisk from) reference Line 1 and delete all lines *after* reference Line 29, as these are in this month's listing.

Type in the additional code in Listing 1 at the end of the existing code. Then reassemble the result and try it as you did last month's listing. You should find that FLEXIKEY works as described. If not then it's "hunt the typo" time, until it does.

EDTASM+ Bug

A bug in *EDTASM+* can cause you problems. If your assembly creates *Multiply Defined Symbol* errors when you know there aren't any, then the bug bit you! It manifests itself when you use arithmetic in the operand field, and the math references a label.

For example, in the program *SYSTEM* from part one, *EDTASM+* does not like the line *CMPU #BUFFER+256*, but if you change it to *CMPU #256+*BUFFER it likes it just fine. So look for lines like this before tearing all your hair out!

A Gentle Reminder

When you have transferred BASIC (unmodified or otherwise) to a disk or an EPROM using information in this series, the result is *still copyrighted* by RS and Microsoft. Giving the disk or EPROM away or selling it to others infringes on this and is illegal.

None of my patch code contains original RS BASIC code and is itself copyrighted. However, it may be freely distributed as long as my copyright notice remains intact, both in the source code and in the start-up banner. My revisions may not be sold for profit without my written consent.

Coming Next Month

We will add the code to make many of the new BASIC commands fully functional, including *COLD* and *AUTO* and *DATE$*. So let's make it a date$!

If you would like the entire DOSPATCH *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.

The listing:

```
0708        OPT   LIS
0709 ********************************************
0710 *  PATCH #2 to RSDOS (C)1984 Colin Stearman *
0711 ********************************************
0712 *
0713 ******************************************
0714 *               FLEXIKEY
0715 **        BASIC LAST LINE RE-ENTRY AND EDIT ROUTINE
0716 * This is not a callable command, but a set of
0717 * direct commands from the keyboard, to allow access
0718 * to the last command entered. It is designed to
0719 * work only when called from BASIC and does not
0720 * interfere with the EDIT commands.
0721
0722 *        COMMANDS ARE:
0723 *
0724 * LEFT ARROW -        output next character of old line
0725 * SHIFT/LEFT ARROW- output rest of old line
0726 * SHIFT/UP ARROW -  insert, no old line increment
0727 * DOWN ARROW -        delete next character in buffer
0728 * SHIFT/@ -           store line input so far.
0729 *                     No interpretation
0730 *
```

```
            0731 ******************************************
            0732 * GENERAL PRINCIPLE OF OPERATION:
            0733 *
            0734 * To allow access to special keyboard entries the
            0735 * RAM hook at $16A is modified to go to this routine.
            0736 * If the device is 0, the keyboard,
            0737 * the key and cursor are obtained and output from
            0738 * here. The special keys interpreted  and characters
            0739 * are drawn from this as required. One permanent RAM
            0740 * location is used to indicate the need to initialize
            0741 * pointer.
            0742 *
            0743 * At the end the old return is removed from the stack
            0744 * so it is not taken.  This allows the input
            0745 * handling routine to handle the character as normal.
            0746 *
            0747 * Because SHIFT/UP ARROW & SHIFT/RIGHT ARROW are also
            0748 * used to create the left arrow and ], these are
            0749 * now obtained by pressing SHIFT/CLEAR first.
            0750 * As this is the backslash this can be obtained by
            0751 * pressing SHIFT/CLEAR twice.
            0752 *
            0753 * FLAGS:
            0754 * INTFLG  0 = line in BASIC buffer just stored
            0755 *         FF = line in hold buffer in use
            0756 * HLDPTR zero-based pointer into hold buffer
            0757 * INSERT  0 = Insert mode off
            0758 *         FF = Insert mode on
            0759 * WHLINE  0 = SHIFT/RIGHT ARROW not previously pressed
            0760 *         FF = SHIFT/RIGHT ARROW previously pressed
            0761 *
            0762 ******************************************
            0763 *
DA60 966F   0764 KEYBRD LDA   DEVNUM
DA62 270C   0765        BEQ   KEY         DEVICE IS KEYBOARD

---

            0766 *      SEE IF CASSETTE I/O GOING ON
DA64 81FF   0767        CMPA  #-1         CASSETTE DEVICE CODE
DA66 2605   0768        BNE   JMPOUT      NOT CASETTE SO DO NOTHING
DA68 8601   0769        LDA   #1
DA6A B7014A 0770        STA   INTFLG      MAKE FLAG POSITIVE
DA6D 7EC58F 0771 JMPOUT JMP   CHRVCT      CONTINUE OLD CODE
            0772 ******************************************
            0773 *
DA70 3414   0774 KEY    PSHS  B,X         PRESERVE REG VALUES
DA72 AE67   0775        LDX   7,S         SEE IF CALLED FROM IDLE LOOP
DA74 8CA39D 0776        CMPX  #$A39D      IDLE LOOP CALL RETURN ADDRESS
DA77 2704   0777        BEQ   INIDLE      IN THE IDLE LOOP
DA79 3514   0778        PULS  B,X         FLAGS NOT AFFECTED
DA7B 20F0   0779        BRA   JMPOUT      IS NOT IDLE LOOP
            0780 * THIS ENTRY LINE RECALL WILL ONLY FUNCTION
            0781 * WHEN IN THE BASIC IDLE LOOP
            0782 *
DA7D 0F70   0783 INIDLE CLR   $70         FLAG BUFFER FLUSHED
DA7F 7D014A 0784        TST   INTFLG      HAVE WE BEEN HERE SINCE
            0785 *                    LAST <CR>?
DA82 270A   0786        BEQ   GETTKN      NO CLEAR THE FLAGS
            0787 * YES SEE IF CASSETTE I/O JUST DONE
DA84 2B2B   0788        BMI   TESTWH      NO SO CONTINUE
DA86 7F01DA 0789        CLR   HLDBFR      SET FIRST BYTE IN HOLD=0
DA89 7F014A 0790        CLR   INTFLG      READY FOR COMPLEMENTING
DA8C 2000   0791        BRA   GETTKN      GO CLEAR FLAGS
            0792 *
            0793 *
            0794 *      FIRST TIME THROUGH SINCE <CR> SO SET UP
DA8E 73014A 0795 GETTKN COM   INTFLG      SET FLAG TO $FF
            0796 *      CLEAR FLAGS
DA91 7F01D7 0797 RENTER CLR   HLDPTR
DA94 7F01DB 0798        CLR   INSERT
DP97 7F01D9 0799        CLR   WHLINE
            0800 *
            0801 *      READ CHARACTER FROM KEYBOARD
            0802 *
DA9A BDA1B1 0803 KYREAD JSR   GETKEY      RETURNS KEY IN A
            0804 *
```

```
            0805 *      NOW SEE WHAT WE GOT
            0806 *
DA9D 8109   0807        CMPA  #$09        RIGHT ARROW next character
DA9F 2715   0808        BEQ   GETCHR      GO DO IT
DAA1 815D   0809        CMPA  #$5D        SHIFT/RT ARROW rest of line
DAA3 2605   0810        BNE   J1          NOT THIS
DAA5 7301D9 0811        COM   WHLINE      SET WHOLE LINE FLAG
DAA8 200C   0812        BRA   GETCHR      GET NEXT BUFFER CHARACTER
DAAA 815F   0813 J1     CMPA  #$5F        SHIFT/UP ARROW insert toggle
DAAC 261F   0814        BNE   J2          NOT THIS
DAAE 7301DB 0815        COM   INSERT      TOGGLE INSERT FLAG
            0816 *
            0817 *      SEE IF SHIFT/RT ARROW PREVIOUSLY PRESSED
DAB1 7D01D9 0818 TESTWH TST   WHLINE      OUTPUT WHOLE LINE IF SET
DAB4 27E4   0819        BEQ   KYREAD      NO SO READ KEYBOARD
            0820 ************************************
            0821 *      GET CHARACTER FROM HOLDING BUFFER
DAB6 7F01DB 0822 GETCHR CLR   INSERT      RESET INSERT FLAG
DAB9 F601D7 0823        LDB   HLDPTR      GET POINTER
DABC 8E01DA 0824        LDX   #HLDBFR     POINT X TO HOLDING BUFFER

---
DABF A685   0825        LDA   B,X         GET CHARACTER
DAC1 2605   0826        BNE   GOODCH
            0827 *      ALL BUFFER IS OUT
DAC3 7F01D9 0828        CLR   WHLINE      RESET POINTER
DAC6 20D2   0829        BRA   KYREAD      IGNORE
            0830 *      GOT GOOD CHARACTER
DAC8 7C01D7 0831 GOODCH INC   HLDPTR      MOVE PAST CHARACTER
DACB 204A   0832        BRA   EXIT        AND RETURN WITH IT
            0833 *************************
DACD 8113   0834 J2     CMPA  #$13        SHIFT/@ close line
DACF 2759   0835        BEQ   LINCLS      GO TO LINE CLOSE
DAD1 810D   0836        CMPA  #$0D        RETURN enter
DAD3 276E   0837        BEQ   ENTER
DAD5 8108   0838        CMPA  #$08        BACKSPACE delete last char
DAD7 2709   0839        BEQ   J4
DAD9 810A   0840        CMPA  #$0A        DOWN ARROW delete next char
DADB 2617   0841        BNE   J3
DADD BDDB1C 0842        JSR   INCPTR      INCREASE HOLD POINTER
DAE0 20B8   0843        BRA   KYREAD      JUMP BACK TO KEY READING
            0844 *
            0845 *      HANDLE BACKSPACE IF INSERT OFF
            0846 *      DECREASE HLDPTR
DAE2 7D01DB 0847 J4     TST   INSERT
DAE5 2625   0848        BNE   CONXIT      ON SO DON'T DECREMENT
DAE7 8D02   0849        BSR   DECPNT      CONDITIONAL DECREMENT HLDPTR
DAE9 2021   0850        BRA   CONXIT      GO TO CONDITIONAL EXIT
            0851 ******
DAEB 7D01D7 0852 DECPNT TST   HLDPTR
DAEE 2703   0853        BEQ   ATZERO      ALREADY ZERO
DAF0 7A01D7 0854        DEC   HLDPTR      REDUCE HLDPTR BY ONE
DAF3 39     0855 ATZERO RTS
            0856 ******
DAF4 8115   0857 J3     CMPA  #$15        SHIFT/BCKSP clear to start
DAF6 2711   0858        BEQ   CLRPNT      GO CLEAR HLDPTR
DAF8 810C   0859        CMPA  #$0C        CLEAR
DAFA 270D   0860        BEQ   CLRPNT      DITTO
DAFC 8103   0861        CMPA  #$03        BREAK
DAFE 2709   0862        BEQ   CLRPNT      YES SO RESET HLDPTR AND EXIT
DB00 815C   0863        CMPA  #$5C        SHIFT/CLEAR special insert
DB02 2608   0864        BNE   CONXIT      NO SO CONDITIONALLY EXIT
DB04 BDA1B1 0865        JSR   GETKEY      GET ANOTHER KEY
DB07 2003   0866        BRA   CONXIT      AND CONDITIONALLY EXIT
            0867 *********
DB09 7F01D7 0868 CLRPNT CLR   HLDPTR      CLEAR HLDPTR
            0869 *********
DB0C 8120   0870 CONXIT CMPA  #$20        CHECK FOR CONTROL CHARACTER
DB0E 2507   0871        BLD   EXIT        EXIT FROM ROUTINE
            0872 *      PRINTABLE CHARACTER SO SEE IF INSERT ON
DB10 7D01DB 0873        TST   INSERT
DB13 2602   0874        BNE   EXIT
DB15 8D05   0875        BSR   INCPTR      INCREMENT HLDPTR
DB17 3514   0876 EXIT   PULS  B,X         RECOVER INCOMING VALUES
DB19 3262   0877        LEAS  2,S         CLEAN OLD RETURN OFF
DB1B 39     0878        RTS               RETURN TO BASIC CALL
            0879 **********
```

```
DB1C 8E01DA     0880 INCPTR LDX   #HLDBFR     POINT TO HOLDING BUFFER
DB1F F601D7     0881        LDB   HLDPTR
DB22 6D85       0882        TST   B,X         GET CHARACTER IN HOLD
DB24 2703       0883        BEQ   ZEROBT      ZERO BYTE SO AT AT END
---

DB26 7C01D7     0884        INC   HLDPTR
DB29 39         0885 ZEROBT RTS
                0886 ***********************************
                0887 *       DO SHIFT/@ LINE CLOSE
DB2A 6FF801     0888 LINCLS CLR   [1,S]       ZERO OUT LAST BYTE
                0889 *       1,S IS X, THE PNTR IN THE BASIC INPUT BFR
                0890 *
DB2D 8640       0891        LDA   #'@         LOAD @ SIGN
DB2F BDA282     0892        JSR   CHROUT      OUTPUT IT
DB32 BDB958     0893        JSR   RETURN      OUTPUT CARRIAGE RETURN
DB35 C601       0894        LDB   #1          RESET BASICS CHARACTER COUNT
DB37 E7E4       0895        STB   ,S          ON STACK
DB39 8E02DD     0896        LDX   #BASBFR     ALSO BUFFER POINTER
DB3C AF61       0897        STX   1,S         ALSO ON STACK
DB3E 8D0E       0898        BSR   MOVBLK      TRANSFER INPUT BUFFER TO HOLD
DB40 7EDA91     0899        JMP   RENTER      RESET AND START OVER
                0900 ***********************************
                0901 *       DO ENTER
DB43 7F014A     0902 ENTER  CLR   INTFLG      INDICATE BASIC BUFFER CHANGED
                0903 *
                0904 *       CLEAR LAST BYTE IN BASIC INPUT BUFFER
                0905 *       FOR MOVE CODE TO DETECT IT
DB46 6FF801     0906        CLR   [1,S]
DB49 8D03       0907        BSR   MOVBLK      TRANSFER INPUT BUFFER TO HOLD
DB4B 7EDB17     0908        JMP   EXIT        AND LEAVE
                0909 ***********************************
                0910 *       COPY BASIC INPUT BUFFER TO HOLD UNTOKENIZED
DB4E 8E02DD     0911 MOVBLK LDX   #BASBFR     GET START OF BASIC BUFFER
DB51 108E01DA   0912        LDY   #HLDBFR     GET START OF HOLD BUFFER
DB55 E680       0913 DOMORE LDB   ,X+
DB57 E7A0       0914        STB   ,Y+
DB59 26FA       0915        BNE   DOMORE      NOT A ZERO BYTE YET
DB5B 39         0916        RTS
                0917 *************************************
                0918
                0919
DB5B            0920 ZZLAST EQU   *-1         last used address value
                0921 *
                0922 * ZZLAST must not be greater than $DFFF for
                0923 * DOS 1.0 and $DEFF for DOS 1.1.   The latter
                0924 * has the OS-9 Boot program and SWI set routines
                0925 * from $DF00 to $DF4C
                0926 *
                0927 *
                0936        OPT   LIS
D991            0937        END   ADDCOM
        NO ERROR(S) DETECTED
```

GAME

16K

The Hi-Res lowercase punctuation Interplanetary Fruit Fly Baby

By Martin Kaste

I'm sure most of you ambitious amateur programmers are familiar with the profound message of KISS, "Keep It Simple, Stupid!" Yet some people never seem to learn and continue slaving over thousand-line programs for months, only to watch them die slowly before their eyes on the screen, wondering where they went wrong. To me, the beauty of programming is that the most successful programs are usually the shortest, supported by a good idea, written in a few hours time and gradually perfected over a period of weeks.

This is true in the case of *The Interplanetary Fruit Fly*. It's short, simple and demonstrates a nifty little technique for displaying pseudo high-resolution graphics on the text screen: "lowercase punctuation."

You're probably asking, "A which?!" Let me explain. "Lowercase punctuation" is what I call all the periods, commas, question marks, brackets, etc., which your trusty old CoCo uses, with a special twist. Using *POKEs*, we can display these punctuation marks and other characters the same way the computer displays lowercase letters: the white symbol on a black rectangular background. With these characters, using a little ingenuity, we can create just about any figure we please.

These symbols can't be *PRINTed*, but, as I said before, the *POKEs* come to our rescue. I know some of you beginning programmers may shy away a little from *POKEs*. But don't worry, these *POKEs* are all addressed to the video RAM part of the computer, and can't interfere with its normal functioning, so bear with me.

The *POKE* addresses we are going to use range from 1024 to 1535, one location for every one of the 512 characters on the text screen. Except for the symbols we're interested in, most of the 256 characters available with *POKE* can be *PRINTed*, some only with the use of *CHR$* codes. To save time, I have compiled a convenient list of the symbols we want and the value of each:

0 = @	38 = &	51 = 3
1-26 = alphabet	39 = '	52 = 4
27 = [	40 = (	53 = 5
28 = \	41 =)	54 = 6
29 =]	42 = *	55 = 7
30 = ↑	43 = +	56 = 8
31 = ←	44 = ,	57 = 9
32 = black rectangle	45 = -	58 = :
33 = !	46 = .	59 = ;
34 = "	47 = /	60 = <
35 = #	48 = 0	61 = =
36 = $	49 = 1	62 = >
37 = %	50 = 2	63 = ?

Now, let's get to the point. Type in the program listing, but be careful! There is one of those useful but occasionally dastardly speed up *POKEs* in Line 5. Before you run the program, save it on tape. If you want to run it before that, delete Line 5 and add it again when you're ready to save.

After the opening title, a green stripe appears at the top of the otherwise black screen displaying the current score, high score and number of shots remaining, respectively. The game starts immediately, but can be stopped using the pause feature on the CoCo.

The Interplanetary Fruit Fly has already started to make irritating kamikaze dives at you, and all you have to defend yourself with are fifty shots of space age fruit fly repellent! You are controlling a cannon loaded with the stuff at the bottom of the screen, which you move with the left and right arrow keys. With the help of two *PEEKs,* your cannon has continuous smooth movement, meaning that it keeps traveling until you release the key. The fire button, which is the up-arrow key, also has this feature.

The action in the game is not difficult to explain. The movements of the cannon and the projectiles it fires are simple: *POKE* the figure into the new location, cover up its trail with black *POKE 32s*. The cannon travels at intervals of two spaces at a time, the projectile zips upward on a *FOR/NEXT* loop, jumping 32 spaces at a time for vertical movement. The Fly is a little more complicated, but not much. It flies down much the same way the projectile goes up, only the 32 spaces added each time are varied by a SIN(X) function, (Line 17). The result is a fly that weaves around and appears to dodge shots. As you probably guessed the flapping of its wings is an illusion caused simply by switching the parentheses back and forth, open and closed, in each step.

The player is awarded 500 points when he or she "wings" the Fly, 1,000 when the player manages to kill it, blasting the center of its body with the deadly repellent, and a new one takes its place almost instantly. The game ends if and when the Fly touches your cannon or when you run out of shots, whichever comes first. I suggest you ration your shots, because they go quickly, so try to shoot only when the Fly is in range. Another tip: try not to let yourself be fenced into the corners; they're perfect for the Fly to nail you.

One last note for those of you with Extended Color BASIC. You may want to liven up the game a bit by substituting the *SOUND* commands with faster, more exotic *PLAYs*. I use *PLAY"T 4505D04E03A#02DDD01AADA"* in Line 10, *PLAY"T50D"* in Line 19, *PLAY"T50;01ADDE#D#AACA"* in Line 24, and *PLAY"T250;01A A05D03CCCCC"* in Line 28. You can think up something much better, I'm sure. Also, feel free to change any characters that I used in the game. The program layout is simple enough to allow you to make all kinds of modifications.

Enjoy and happy spritzing!

17 223
END 166

The listing:

```
Ø '***THE*****INTERPLANETARY***
1 '******FRUIT****FLY***********
2 'BY***MARTIN*OLAF*KASTE*******
3 '***********MAY*1984***********
4 '******************************
5 POKE65495,Ø
6 CLSØ
7 FORA=1186TO1213
8 FORC=C TO C+5:POKEA,RND(26):NE
XTC
9 PRINT@162,MID$("the"+CHR$(128)
+"interplanetary"+CHR$(128)+"fru
it"+CHR$(128)+"fly",1,A-1186);:S
OUNDC,1:NEXTA
1Ø PRINT@189,"y";:FORA=1TO4:SOUN
D5Ø,1:SOUND75,3:SOUND15Ø,3:SOUND
2,1:SOUND3,1:PRINT@2ØØ,"by"+CHR$
(128)+"martin"+CHR$(128)+"kaste"
;:POKE148Ø,4Ø:POKE1481,3:POKE148
2,41:POKE1483,32:PRINT@462,"smar
tsoft";:NEXT
11 FORN=1Ø56TO1535:POKEN,32:NEXT
N
12 PRINT@Ø," "
13 PRINT@1,"ØØØØ";:PRINT@22,"SHO
TS:";:PRINT@29,"5Ø";:IFHS=ØTHEN1
4ELSEPRINT@12,HS;
14 SH=5Ø:XY=4Ø:YX=41
15 A=1516
16 Z=1Ø56+RND(32):IFSH=<ØTHEN24
17 POKEZ,32:POKEZ+1,32:POKEZ+2,3
2:IFZ>15Ø3THEN16ELSEZ=Z+SIN(M)*R
ND(3):Z=Z+32:M=M+1:POKEZ,XY:POKE
Z+1,43:POKEZ+2,YX:CC=XY:XY=YX:YX
=CC
18 IFZ<A+3ANDZ>A-2THEN24
19 IFPEEK(341)=247THENFORC=A-31T
O1Ø56STEP-32:POKEC,42:POKEC+32,3
2:NEXTC:POKEC+32,32:SH=SH-1:PRIN
T@28,SH;:IFPEEK(Z+1)=32THENS=S+1
ØØØ:GOSUB28:PRINT@1,S;:POKEZ,32:
POKEZ+1,32:POKEZ+2,32:GOTO16:ELS
EIFPEEK(Z)=32ORPEEK(Z+2)=32THENS
OUND5Ø,1:S=S+5ØØ:PRINT@1,S;
2Ø IFSH=ØTHEN24
21 IF PEEK(343)=247THENB=-2:IFA=
15Ø4THENB=Ø
22 IF PEEK(344)=247THENB=2:IFA=1
532THENB=Ø
23 POKEA,32:POKEA+1,32:POKEA+2,3
2:A=A+B:B=Ø:POKEA,47:POKEA+1,42:
POKEA+2,28:GOTO17
24 SOUND1,2:SOUND13,1:SOUND1,3:S
OUND2,4:SOUND1,2:FORA=1TO3ØØ:NEX
T
25 PRINT@33,"TO PLAY AGAIN, PRES
S THE A KEY";:A$=INKEY$:IFA$<>"A
"THEN25
26 IF HS<=S THEN HS=S
27 S=Ø:GOTO11
28 POKEZ,62:POKEZ+2,6Ø:POKEZ-31,
22:POKEZ+33,1:SOUND5Ø,1:SOUND15Ø
,2:SOUND123,1:SOUND14Ø,3:POKEZ,3
2:POKEZ+1,32:POKEZ+2,32:POKEZ-31
,32:POKEZ+33,32:RETURN
```

THE EASY WAY

By Dennis Lewandowski

Radio Shack released a Color Computer with 4K, expandable to 16K. That was it; 16K was to be the maximum amount of memory that could be obtained from it. However a rather intelligent gentleman wrote an article in the March 1981 *BYTE* magazine, telling how to piggyback 16K chips and fool the SAM and CPU into seeing 32K.

Rumors flew fast and furious about a 32K memory kit from Radio Shack. It involved the use of partially bad 64K chips. Now the minds at Frank Hogg Laboratories went to work. Using an extra gate already available on the board, they could toggle the upper bank of the 64K chips in and out, thus, making the current 64K Color Computer we have today. Everyone went to work to develop a 128K modification, but the same stumbling block kept getting in the way; the ROM version of BASIC will only support 32K. With prices dropping on computers faster then pig bellies on Wall Street, most research went the way of the horse. Sure, there are a couple of 128K modifications currently available, however the hows and whys are being kept guarded secrets, making software support almost impossible.

Now a little background on this 128K modification. The main objectives are basically common sense. The modification must be usable by BASIC, and relatively inexpensive to upgrade the current CoCo. It also has to like FLEX, and OS-9. For these objectives to be reached we chose to use two sets of 64K chips. There are six other chips involved in the modification, which take care of selecting the banks. Actually five of the chips take care of the bank selection, but due to timing considerations, the sixth chip makes certain that the computer operates with RAM chips of all speeds. The method was limited to bank selection since the CPU can only address 64K at any one given time. We chose to exchange the lower banks of 32K, address between zero and 32767 (0000-7FFF). Now with three lower banks of RAM, BASIC can have three programs resident in memory. Also there is another 32K bank of RAM, addresses 32768 to 65535 (8000-FFFF), where a control program can be placed to operate the lower three banks, similar to 128K operation of an Apple II. Realistically speaking, with ROM included, there is a total of 160K available. Refer to Figure 1 for a block diagram of how the 32K pages are configured.

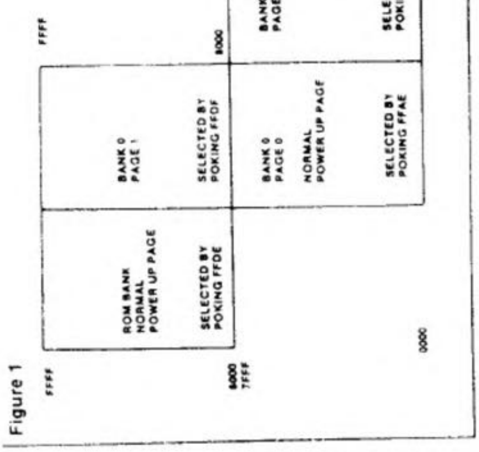
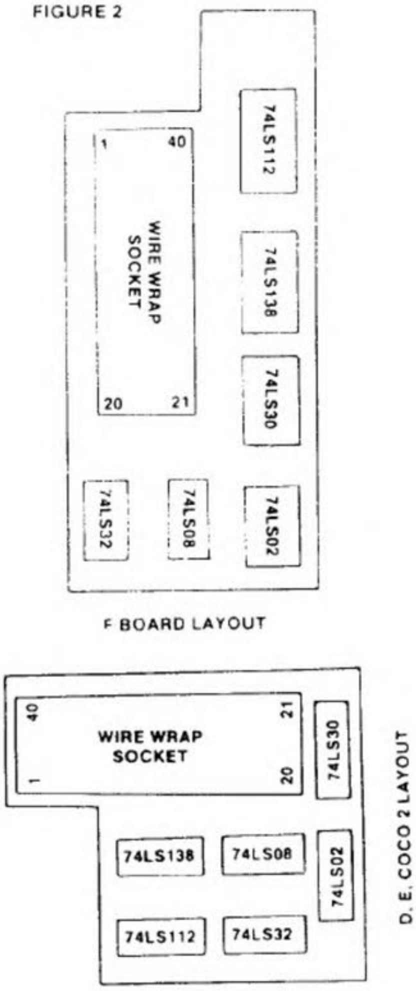


Figure 1

FIGURE 2



How To Do It

The simplest way of inserting memory, since there are only eight RAM sockets, is to piggyback the chips. There are data lines running through the 74LS244 (U19), that exist only at that chip. To place the additional RAM anywhere else would require the removal of the 74LS244 and installing a series of jumpers from its position. The drawback of this is since late version "E" boards, the 74LS244 has been soldered onto the board. Also the more wire that runs around inside the computer, the more noise (RF) the computer will generate. By piggybacking the 64K chips, these problems disappear. Most of the signals required to operate more memory will come from the SAM chip; to do this the SAM is elevated by means of a wire wrap socket. The additional chips are then placed on a board attached to that socket. Depending on your own level of soldering ability, there are a couple of ways to proceed. Using the suggested board layout (Figure 2), cut a sheet of perf board to dimension. Then follow the wiring diagram (Figure 3), and hand wire the board. The parts necessary to make 64K CoCo into a 128K CoCo are as follows:

8 - 64K RAM Chips (4164 or equiv.)
1 - 74LS02 Quad NOR gate
1 - 74LS08 Quad AND gate
1 - 74LS30 8 Input gate
1 - 74LS32 Quad OR gate
1 - 74LS112 Dual JK Flip Flop
1 - 74LS138 3 to 8 Line Decoder
1 - 270 ohm Resistor
1 - 4.7 uf Capacitor
6 - .1 uf Capacitors
1 - Spool of Wire (R/S # 278-501)

One important item is that pin 11 of the SAM chip does not go through to the original socket as all the other pins do. It should be cut off right below its connection to pin 4 of the 74LS32.

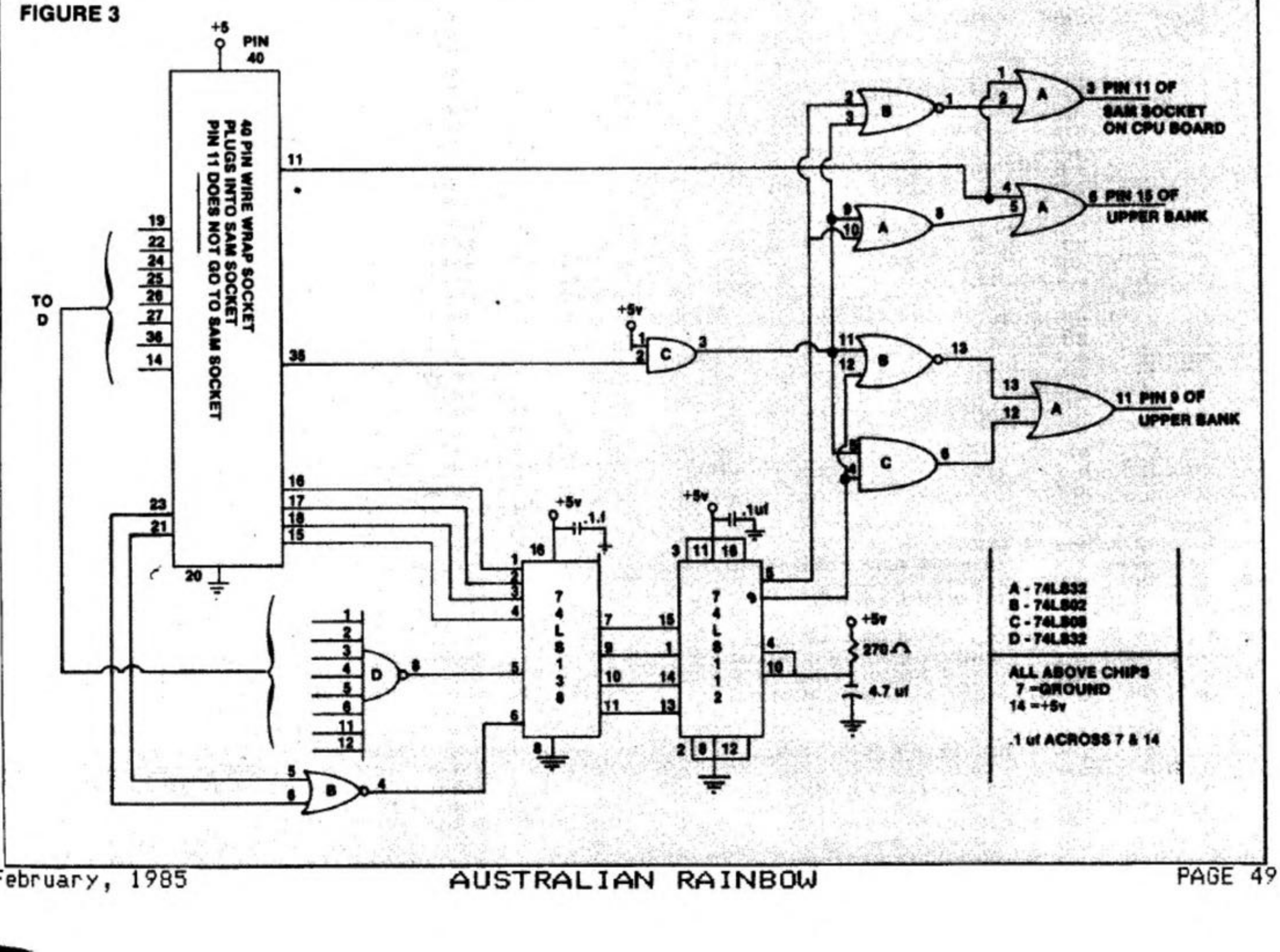
If you prefer the board with the six chips soldered and tested, it is available from DSL Computer Products for $34.95. If you want to just plug in and go, the complete mod including an additional 64K of RAM is $99. The installation of the complete mod is solderless.

How To Use It

Once the 128K modification is done; what can you do with it? When you power up the computer everything should be the same as normal including the familiar answer to the *?MEM* question. So how do you have 128K? Type in the command *POKE &HFFAE,0* and press ENTER. Now if you see garbage on the screen this is normal, press Reset, and the computer powers up again. If you don't have a disk system the computer will reset automatically, usually. You are now in bank zero, page zero of RAM. This is the normal bank in which a 64K Color Computer will power up. To enter the upper bank of 64K type *POKE&HFFAC,0:POKE&HFFAF,0.* Again, if you see garbage just press Reset. You are now in bank one, page zero. (One way to be sure you are switching banks is that the screen will change with the bank. If the poke has no effect you may have a wiring problem.) Finally, type *POKE&HFFAD,0.* This is bank one, page one. Now all three lower pages of 128K mod have been initialized.

Referring to Figure 1 again, the normal memory map has not been changed, but rather modified. Two 32K pages have been added along side the present 32K page used by BASIC. All current software will run as it normally does. The only way a different bank can be accessed is by poking (writing) to a memory location between 65452 and 65455. The value poked into that address can be zero to 255. By switching banks, three programs can reside in the computer at the same time. The CPU can only run one program at a time, unless another operating system is controlling it, such as FLEX or OS-9. However, all three programs can be run in such a way that it will seem that they are running all the time. There is one consideration using BASIC; that is the location of the Stack Pointer. If one of the programs clears space for strings or arrays, the Stack Pointer would be adjusted. Then when that bank is switched out the Stack Pointer would be pointing to

FIGURE 3



nowhere. The results could cause the computer to lock up. For this reason there is a program listing included called *STAKSTAT*. This will initalize all three lower banks, transfer ROM to the upper bank of RAM, and add a new command to BASIC. By entering the command *PAGE x*, where x is 0, 1, or 2, that page will be selected and the correct stack value will be placed into the Stack Pointer. Please note that *STAKSTAT* can be used freely with any 128K software that you develop. However, if you wish to send it to a magazine to have it published, remember where you saw it first!

Listing 1:

```
                00100 ********************************
                00110 **                            **
                00120 **     PAGER VERSION 9.10.84  **
                00130 **                            **
                00140 **     COPYRIGHT (c) 1984     **
                00150 **     DSL COMPUTER PRODUCTS  **
                00160 **     WRITTEN BY             **
                00170 **     GERALD S. EGGART       **
                00180 **                            **
                00190 ********************************
                00200
                00210 *
                00220 *THIS PROGRAM IS FOR USE WITH THE DSL 128K UPGRADE
                00230 *UPON ENTRY (EXEC) ALL REGISTERS ARE MODIFIED
                00240 *UPON CALL (PAGE) STACK IS PRESERVED & ALL REGISTERS MODIFIED
                00250 *
7000            00260       ORG   $7000   LOAD ADDRESS
                00270
                00280 *
                00290 *INITIALIZE ZERO PAGE ROUTINE FOR NEW COMMAND
                00300 *
                00310
7000 CE   013E  00320 INIT1 LDU #$13E     THE ADDRESS WERE THE INTERPRETER
                00330 *                   VECTOR TABLE IS
                00340
7003 86   01    00350 HRE   LDA #1        ONE NEW COMMAND
7005 A7   C0    00360       STA ,U+
7007 30   8D 0024 00370     LEAX TABLE,PCR  ADDRESS OF NEW COMMAND TABLE
700B AF   C1    00380       STX ,U++
700D 30   8D 0023 00390     LEAX SUBRO,PCR NEW COMMAND HANDLING ROUTINE ADRS
7011 AF   C1    00400       STX  ,U++
7013 6F   C4    00410       CLR ,U          NO SECONDARY FUNCTIONS ADDED
7015 8E   B277  00420       LDX #$B277      SECONDARY FUNCTION HANDLEING
                00430 *                     ROUTINE (?SN ERROR)
7018 AF   43    00440       STX 3,U         CLEAROUT NEXT TABLE ENTRY...
                00450 *                     NO MORE COMMANDS ADDED
701A 6F   45    00460       CLR 5,U
701C AF   42    00470       STX 2,U
                00480
701E 30   8D 0022 00490     LEAX ENTRY,PCR ADDRESS OF ROUTINE TO DO PAGEING
7022 AF   8D 0001 00500     STX ADDRE,PCR  STORE IT IN THE JUMP TABLE
7026 39         00510       RTS
                00520

                00530 *
                00540 *      NEW COMMAND TABLE
                00550 *
                00560
7027      0000  00570 ADDRE FDB 00
7029      0000  00580 ADD1  FDB 00
702B      0000  00590 ADD2  FDB 00
702D      0000  00600 ADD3  FDB 00
                00610
                00620 *
                00630 *      NEW WORD TABLE
                00640 *
                00650
702F      50    00660 TABLE FCC /PAG/
7032      C5    00670       FCB 'E+$80
7033      00    00680       FCB 0
                00690
                00700 *
                00710 *      NEW   INTERPRETER ROUTINE FOR THE ADDED WORD(S)
                00720 *
                00730
7034 81   E4    00740 SUBRO CMPA #$E4  HIGEST TOKEN VALUE USED
                00750 *                (#$E5 FOR DISK 1.1)
7036 25   03    00760       BLO LO     IF ITS LOWER ITS A VALID TOKEN
7038 7E   B277  00770       JMP $B277  NOT A VALID TOKEN SO ?SN ERROR
                00780
703B 30   8D FFE8 00790 LO  LEAX ADDRE,PCR  GET ADDRES OF WERE TO GO TO
                00800 *                     EXEC COMMAND
703F 80   E1    00810 SUBR1 SUBA #$E1       SET OFFSET BYTE TO WHICH COMMAND
7041 7E   ADD4  00820       JMP $ADD4       GO LET ROM EXEC COMMAND (ENTRY)
                00830
                00840 *
                00850 *      THIS IS THE TRUE EXECUTION OF THE PAGE COMMAND
                00860 *
                00870
7044            00880 ENTRY
7044 BD   B70B  00890       JSR   $B70B   EVALUATE AN EXPRESSION
                00900 *                   >0 /<256 AND RETURN IT IN B
7047 C1   02    00910       CMPB  #2      IS THE PAGE VALUE GREATER THAN 2
7049 22   29    00920       BHI   ERROR   IF IT IS THEN 'FC ERROR'
704B 5D         00930       TSTB          IS IT PAGE 0?
704C 26   07    00940       BNE   CONTST  NOPE THEN GOTO CONTST
                00950
                00960 *
                00970 *     PAGE 0

                00980 *
704E AE   E4    00990       LDX   ,S    GET RTS ADDRESS OFF THE STACK
7050 7F   FFAE  01000       CLR   $FFAE FLIP TO PAGE ZERO
7053 20   16    01010       BRA   ENDIT AND EXIT
                01020
                01030 *
                01040 *     PAGE 1 AND PAGE 2 TEST
                01050 *
7055 C1   01    01060 CONTST CMPB #1    IS IT PAGE ONE?
7057 26   0A    01070       BNE   PG2   NO GOTO PG2
                01080 *
                01090 *     PAGE 1
                01100 *
                01110
7059 AE   E4    01120       LDX   ,S    GET RETURN ADDRESS
705B 7F   FFAC  01130       CLR   $FFAC GO TO PAGE 1
705E 7F   FFAF  01140       CLR   $FFAF
7061 20   08    01150       BRA   ENDIT AND EXIT
                01160 *
                01170 *     PAGE 2
                01180 *
                01190
7063 AE   E4    01200 PG2   LDX   ,S   GET RETURN ADDRESS OFF STACK
7065 7F   FFAD  01210       CLR   $FFAD GOTO PAGE 2
7068 7F   FFAF  01220       CLR   $FFAF
                01230
706B 10BE 00F3  01240 ENDIT LDY   >$F3  GET THE LOC OF THIS PAGES STACK
706F 1F   24    01250       TFR   Y,S   PUT STACK BACK
7071 AF   E4    01260       STX   ,S    SAVE RETURN ADDRESS BACK ON STACK
7073 39         01270       RTS
                01280 *
                01290 *     ERROR HANDLER IF PARAMETER IS BAD IN PAGE COMMAND
                01300 *
                01310

7074            01320 ERROR
7074 7E   B706  01330       JMP   $B706  GOTO 'FC ERROR'
                01340
                01350 *
                01360 *     THE FOLLOWING CODE IS USED ONCE TO
                01370 *     SET UP THE COMPUTER UPON EXEC
                01380 * THIS CODES MOVES ROM TO RAM
                01390 *             RELOCATES THE MAIN PRGORAM
                01400 *             AND INITAILIZES ALL THREE BANKS
                01410 *             TO A POWER UP STATUS
                01420 *
                01430
7077 B6   0134  01440 START LDA   $134
707A 81   00    01450       CMPA  #$0
707C 26   06    01460       BNE   STCONT
707E CE   0134  01470       LDU   #$134
7081 FF   7001  01480       STU   INIT1+1
                01490
7084 81   14    01500 STCONT CMPA #20
7086 26   0A    01510       BNE   HERE
7088 86   E5    01520       LDA   #$E5
708A B7   7035  01530       STA   SUBRO+1
708D 86   E2    01540       LDA   #$E2
708F B7   7040  01550       STA   SUBR1+1
7092 1A   50    01560 HERE  ORCC  #$50     DISABLE INTERUPTS
7094 CE   8000  01570       LDU   #$8000   START OF ROM
7097 8E   FFDE  01580       LDX   #$FFDE   ROM ON LOCATION
709A 108E FFDF  01590       LDY   #$FFDF   ROM OFF LOCATION
                01600 *
                01610 *     COPY ROM TO RAM
                01620 *
709E EC   C4    01630 LOOP  LDD   ,U       GET TWO BYTES OF ROM
70A0 6F   A4    01640       CLR   ,Y       GOTO ALL RAM MODE
70A2 ED   C1    01650       STD   ,U++     SAVE THE BYTES IN RAM
70A4 6F   84    01660       CLR   ,X       BACK TO ROM/RAM MODE
70A6 11B3 FE00  01670       CMPU  #$FE00   AT THE END OF ROM?????
70AA 22   02    01680       BHI   EXIT     YEP... GOTO EXIT
70AC 20   F0    01690       BRA   LOOP     GO DO SOME MORE
                01700
70AE CE   F000  01710 EXIT  LDU   #$F000    DONE MOVING ROM TO RAM SO NOW
                01720 *                     LETS MOVE THE PROGRAM ABOVE ROM
70B1 30   8D FF4B 01730     LEAX  INIT1,PCR START OF PROGRAM
70B5 108E F000  01740       LDY   #$F000    DESTINATION ADDRESS
70B9 7F   FFDF  01750       CLR   $FFDF     GOTO ALL RAM MODE
                01760
```

```
7ØØC EC   81      Ø177Ø MLOOP  LDD   ,X++       GRAB 2 BYTES OF PROGRAM
7ØØE ED   A1      Ø178Ø        STD   ,Y++       SEND IT ABOVE ROM
7ØCØ BC   7141    Ø179Ø        CMPX  #DUMY      END OF PROGRAM???
7ØC3 25   F7      Ø18ØØ        BLO   MLOOP      NO THEN GO GET SOME MORE
                  Ø181Ø
                  Ø182Ø *
                  Ø183Ø *      NOW GO TO THE ROUTINE INTPGE TO INITILAIZE MEMORY.
                  Ø184Ø *      FIGURE OFFSET
                  Ø185Ø *
                  Ø186Ø
7ØC5 3Ø   8D ØØØB Ø187Ø        LEAX  INTPGE,PCR ADDRESS OF WEAR THE
                  Ø188Ø *                       ROUTINE NOW IS
7ØC9 1F   1Ø      Ø189Ø        TFR   X,D
7ØCB 83   7ØØØ    Ø19ØØ        SUBD  #INIT1      SUBTRACT THE START OF THE PROGRAM
7ØCE C3   FØØØ    Ø191Ø        ADDD  #$FØØØ      ADD OFFSET
7ØD1 34   Ø6      Ø192Ø        PSHS  D           FAKE RETURN ADDRESS
7ØD3 39           Ø193Ø        RTS               GOTO INTPGE ROUTINE THAT HAS
                  Ø194Ø *                        BEEN TRANSFERED INTO UPPER MEMORY
                  Ø195Ø *
                  Ø196Ø *      THIS ROUTINE INITAILIZES ALL LOWER 32K PAGES
                  Ø197Ø *
                  Ø198Ø
7ØD4 1Ø8E FFAC    Ø199Ø INTPGE LDY   #$FFAC  PAGE SELECT ADDRESS
7ØD8 86   F2      Ø2ØØØ        LDA   #$F2
7ØDA B7   ØØF3    Ø2Ø1Ø        STA   >$F3
7ØDD 7F   FFAE    Ø2Ø2Ø        CLR   $FFAE
7ØEØ B1   ØØF3    Ø2Ø3Ø        CMPA  >$F3
7ØE3 27   13      Ø2Ø4Ø        BEQ   CNTINT
7ØE5 CE   ØØØØ    Ø2Ø5Ø        LDU   #$Ø
7ØE8 7F   FFAF    Ø2Ø6Ø INITLP CLR   $FFAF
7ØEB EC   C4      Ø2Ø7Ø        LDD   ,U
7ØED 7F   FFAE    Ø2Ø8Ø        CLR   $FFAE
7ØFØ ED   C1      Ø2Ø9Ø        STD   ,U++
7ØF2 1183 7FFF    Ø21ØØ        CMPU  #$7FFF
7ØF6 25   FØ      Ø211Ø        BLO   INITCP
                  Ø212Ø
7ØF8 CE   ØØØØ    Ø213Ø CNTINT LDU   #$Ø     START COPY AT ZERO
7ØFB 8D   29      Ø214Ø        BSR   COPY    GO COPY A PAGE
7ØFD 1Ø8C FFAD    Ø215Ø        CMPY  #$FFAD  ARE WE AT THE LAST 32K PAGE?
71Ø1 27   Ø4      Ø216Ø        BEQ   OUT     YEP.... TIME TO GO!
71Ø3 31   21      Ø217Ø        LEAY  1,Y     NO SO POINT Y TO NEXT MAGIC MEMORY LOCATION
71Ø5 2Ø   F1      Ø218Ø        BRA   CNTINT  GO DO IT ALL AGAIN
                  Ø219Ø
71Ø7 7F   FFAF    Ø22ØØ OUT    CLR   $FFAF   NOW TO COPY THE STACK ADDRESS
                  Ø221Ø *                    TO ALL PAGES
71ØA 7F   FFAC    Ø222Ø        CLR   $FFAC
71ØD 1F   41      Ø223Ø        TFR   S,X     GRAB THE STACK ADDRESS
71ØF BF   ØØF3    Ø224Ø        STX   >$F3    SAVE IT IN PAGE 1
7112 7F   FFAD    Ø225Ø        CLR   $FFAD
7115 BF   ØØF3    Ø226Ø        STX   >$F3    ALSO PAGE 2
7118 7F   FFAE    Ø227Ø        CLR   $FFAE
711B BF   ØØF3    Ø228Ø        STX   >$F3    AND PAGE 3
711E 1C   AF      Ø229Ø        ANDCC #$AF    RENABLE ITTERUPTS
712Ø 7F   FFAE    Ø23ØØ        CLR   $FFAE
7123 7E   FØØØ    Ø231Ø        JMP   $FØØØ   GO INITAILIZE THE NEW WORD TABLE
                  Ø232Ø *                    IN PAGE Ø AND RETURN TO BASIC
                  Ø233Ø *
                  Ø234Ø *      THIS ROUTINE COPIES ONE 32K PAGE TO ANOTHER
                  Ø235Ø *
7126 7F   FFAE    Ø236Ø COPY   CLR   $FFAE     GOTO PAGE ZERO
7129 EC   C4      Ø237Ø        LDD   ,U        YANK A COUPLE OF BYTES
712B 7F   FFAF    Ø238Ø        CLR   $FFAF     SELECT OTHER 64K BANK
712E 6F   A4      Ø239Ø        CLR   ,Y        SELECT WHICH 32K PAGE
713Ø ED   C1      Ø24ØØ        STD   ,U++      SAVE THE BYTES
7132 1183 7FFF    Ø241Ø        CMPU  #$7FFF    END OF 32K PAGE???
7136 25   EE      Ø242Ø        BLO   COPY      NO THEN DO SOME MORE!
7138 17   FEC5    Ø243Ø        LBSR  INIT1     YEP SO INITAILIZE THE NEW COMMAND
                  Ø244Ø *                      TABLE IN THIS PAGE
713B 7F   FFAE    Ø245Ø        CLR   $FFAE     BACK TO PAGE ZERC FOR THE RTS
713E 39           Ø246Ø        RTS
                  Ø247Ø
                  Ø248Ø *
                  Ø249Ø *      DUMY LABEL SO I KNOW WHERE THIS PRGRAM ENDS!!!
713F      ØØØØ    Ø25ØØ STACK  FDB   $Ø
7141      ØØØØ    Ø251Ø DUMY   FDB   $Ø
          7Ø77    Ø252Ø        END   START
ØØØØØ TOTAL ERRORS
```

22........ 34
END 55

Listing 2:

```
Ø '*******************************
1 '**                           **
2 '** PAGER VERSION 9.1Ø.84     **
3 '**                           **
4 '**      COPYRIGHT 1984       **
5 '** DSL COMPUTER PRODUCTS     **
6 '**                           **
7 '*****************************
8 CLEAR2ØØ,&H7ØØØ
9 FORI = 28672 TO 28976
1Ø READ A
11 POKE I,A
12 NEXT I
13 EXEC 28767
14 CLEAR2ØØ ,&H7FFF
15 NEW
16 DATA 191,224,Ø,16,191,224,2,2
55,224,4,253,224,6,189,183,11,19
3,2,34,59
17 DATA 31,65,191,Ø,243,93,38,7,
174,228,127,255,174,32,22,193,1,
38,1Ø,174
18 DATA 228,127,255,172,127,255,
175,32,8,174,228,127,255,173,127
,255,175,16,19Ø,Ø
19 DATA 243,31,36,175,228,19Ø,22
4,Ø,16,19Ø,224,2,254,224,4,252,2
24,6,57,19Ø
2Ø DATA 224,Ø,16,19Ø,224,2,254,2
24,4,252,224,6,126,183,6,23,Ø,36
,127,255
21 DATA 223,48,141,Ø,23,16,142,1
7Ø,241,141,8,142,24Ø,Ø,191,171,1
73,32,44,166
22 DATA 128,39,4,167,16Ø,32,248,
57,8Ø,65,71,69,16Ø,Ø,26,8Ø,2Ø6,1
28,Ø,142
23 DATA 255,222,16,142,255,223,2
36,196,111,164,237,193,111,132,1
7,131,254,Ø,34,2
24 DATA 32,24Ø,57,2Ø6,24Ø,Ø,48,1
41,255,86,16,142,24Ø,Ø,127,255,2
23,236,129,237
25 DATA 161,14Ø,113,47,37,247,48
,141,Ø,11,31,16,131,112,Ø,195,24
Ø,Ø,52,6
26 DATA 57,16,142,255,172,134,24
2,183,Ø,243,127,255,174,177,Ø,24
3,39,19,2Ø6,Ø
27 DATA Ø,127,255,175,236,196,12
7,255,174,237,193,17,131,127,255
,37,24Ø,2Ø6,Ø,Ø
28 DATA 141,39,16,14Ø,255,173,39
,4,49,33,32,241,127,255,175,127,
255,172,31,65
29 DATA 191,Ø,243,127,255,173,19
1,Ø,243,127,255,174,191,Ø,243,28
,175,127,255,174
3Ø DATA 57,127,255,174,236,196,1
27,255,175,111,164,237,193,17,13
1,127,255,37,238,127
31 DATA 255,174.57.Ø.Ø
```

hogg_wash

128K And FLEX

By Frank Hogg

In a companion article, Dennis Lewandowski shows how you can add 64K to your CoCo to bring the memory up to 128K. Now the question is what to do with it. Of course, it would be nice to be able to use this with FLEX and OS-9, but there is a problem with OS-9 that is too complex to go into here. However, it is easy to use it with FLEX. In this column, I am going to discuss some thoughts on how this could be done, and next time I will provide a program to do it. I understand that Dennis' company, DSL, is doing a RAM Disk for FLEX.

Figure 1

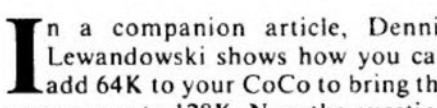


Figure 1 shows how the 128K is mapped into the system. Keeping in mind that the 6809 can only work with 64K at any one moment, we see that we can have either ROM or RAM in the upper 32K, which is nothing new, but we can now have any one of three 32K banks in the lower 32K. Both FLEX and OS-9 work by running the CoCo in the all-RAM mode where the upper 32K is RAM and the lower 32K is RAM. OS-9 uses memory in a very complex and rather slick way. While it is possible for OS-9 Level II to address more than 64K, it is done in a much more elegant way than we have here. The main shortcoming is that systems (Gimix, etc.) that support OS-9 Level II have hardware that allows the memory to be broken into 2K or 4K segments. The system puts these chunks together in various different physical pieces that look to the 6809 as a contiguous block of memory. OS-9 Level II has been designed to work with this type of fancy hardware. About the only thing that could be done that would be useful would be a memory disk for OS-9. This would also be handy for FLEX and is what Dennis is working on, but there is a simple way to implement the other two 32K banks or RAM with FLEX.

First the limitations: FLEX has a total of 46K user memory that is addressed from $0000 to $B7FF. We can switch the lower 32K, $0000 to $7FFF between the three different banks, but the memory from $8000 to $B7FF is part of the upper 32K of RAM that cannot be switched.

Doing It In XBASIC

It would be nice to have several Extended BASIC programs running that could switch from one to the other. The problem is that TSC's XBASIC uses all of the user memory and puts its stack at the top of user memory. A way around this would be to move MEMEND to $7FFF, which would allow switching without creaming XBASIC's stack. However, there are complexities involved with this. First, consider the state of Extended BASIC when the switch is made. If you did it by using *POKE*s, then you would jump from one running BASIC program into who knows where in another BASIC program. If you did it with an *EXEC* command from BASIC, you would have to preserve the return address stored in FLEX for the bank you were in and set up the return address for the bank you were jumping into. This whole thing is fraught with peril. However, it would provide an almost endless variety of ways to crash the system. A considerable amount of thought needs to go into this aspect of using the extra memory. This will be part of next month's article.

Another way that would be very easy to implement, and would be fairly safe, would be to simply *LOAD* programs that you are going to use into the different banks, and then just switch to the bank and jump to the warm start address of the program to use it. This could be called the poor man's virtual disk, and it does fit into a magazine article very nicely.

Let's look at the problems of doing this. Loading a program into memory is simple: you just *GET* it, as *GET 0.ED.CMD*. Now, you can't just jump to the program's start address, because it will tell you it needs a filename to edit. Most FLEX programs get their filenames from the command line. The sequence ED TEST first loads ED, and then ED runs, and the first thing it does is to look at the command line to get the filename of TEST. If we had ED in memory, and we knew its start address was $0100, we could just type *JUMP 100 TEST*. The JUMP command resides at $C100 and, therefore, will not crash ED. In order to use this with the 128K, we need to switch to the bank with ED in memory and then jump to the start address with a filename on the command line. We need a command that we will call BANK, which takes two arguments. The first is the bank to jump to and the second is the address to transfer. BANK with no arguments should return the bank we are currently in, in case we get lost in memory. The *piece de resistance* would be to make BANK memory resident so we would not have to access the disk every time we used it.

There is a block of memory at $FE00 that is not used by FLEX. It is 256 bytes long. We can write a program that will patch itself into the user command table and reside at $FE00 and do all the things we want to do. We could even try to preserve the warm start return address for each bank for possible use by BASIC.

While you are waiting for the next issue of THE RAINBOW for this program, you can play with the concept by using the MON command to switch banks and the GET and JUMP commands to try it out.

This would be very useful to the assembly language programmer. Writing assembly language is fun, but it can be a pain. After you test your program, you usually have to make a small correction, then reassemble it and test it again. This series of ED, ASM, TEST, ED, ASM, TEST etc., is a nuisance. With three banks to work with, we could load one with ED, one with ASM, and use the third for testing. Even doing BASIC programming, we could load one bank with BASIC and the other with ED and use the powerful editor instead of the one in BASIC. This technique could be used for any language development.

The one key thing you need to know is the cold start address of the programs you wish to use this way. You can find this by using the MAP command that is available with Ed/Asm or some other utilities, or you could refer to Figure 2 which has a listing of addresses for various popular programs. Caution: I have not had time to test any of these because of the magazine's deadline, so I will leave the fun part to you.

Figure 2

ED	$0000
ASM	$0003
XBASIC	$0000
DYNA-C	$0100
ABASIC	$0100
TSC EDIT	$0000
TSC ASMB	$0000

128K Another Way

After I started this article, I received a 128K board from RGS Micro. Their 128K board for the CoCo is similar to the DSL board but appears to have some additional capabilities. The two approaches are different. The DSL approach is one that you can do yourself for the least amount of money. The RGS board is more expensive but may be easier for the faint of heart. I would like to go into a good comparative review, but I didn't have the time to do it before this deadline. I can say that from what I've seen so far, I don't think you would be unhappy with either choice. Both techniques offer 128K, both will work with FLEX, and both will work with OS-9 as a RAM disk. I don't think that either will work like OS-9 Level II, but that is just my opinion. There are some pretty clever programmers that, have surprised me in the past. I can say with some confidence that it won't be easy to get it to run like Level II. After all, it took Microware one year to get from Level I to II, and they had the source! However, a 64K RAM disk would be very useful in OS-9, FLEX or Disk BASIC.

Problems

The 128K boards remind me of when you had to modify your computer to get to 64K. In order to use 128K you will have to open your computer and break Radio Shack's precious seal. Well, if your computer is out of warranty, then there is nothing to lose. Some people think that they will have trouble trying to get their computer fixed at Radio Shack if they have modified it. This is probably true if the modification interferes with repairing the computer. If you fall into this category, then you should be prepared to remove any modifications before you return your computer to Radio Shack for repair. With this in mind, consider how you will accomplish the upgrade to 128K in a way that is reversible.

I modified an old 'F' board with the DSL mod, and it was easy and went off without a hitch. The mod is reversible. When I got the RGS board, it was for a CoCo 2. My CoCo 2 had the 64K RAMs soldered in! This would have made doing the DSL modification *very* difficult. You would have to remove the 64K RAMs and put in sockets. If you have a 16K CoCo 2 that was upgraded to 64K, then you should have sockets for the RAM chips. The only ones that didn't were the 64K CoCo 2s.

I heard from Bob Rosen at Spectrum Projects that there was a new board in the CoCo 2 that was different from my board. He said that it was smaller and that the chips were in different places.

I would suggest that you open your CoCo case and find out what revision board you have. Also note if the SAM (74LS83) and the 74LS244 are soldered in or in sockets. In my system the 74LS244 was soldered and I cut the pins from the chip and soldered a socket to those pins for the upgrade. (This was for the RGS upgrade.)

After you have this done, and with the open computer in front of you, call the company you wish to do business with and tell them what you have. This way, you will save yourself and them a lot of trouble.

The keyboard on my CoCo 2 had a small metal ledge that interfered with the RGS board. They are working on the problem, so check with them or DSL about your computer. As time goes by, these little difficulties will be worked out as they have been in the past.

It is not difficult to do this upgrade and the results are well worth it. Tune in next month for part two.

New Fix For DynaForm

We came up with a new fix for using *DynaForm* with printers that produce their own line feeds when they receive a carriage return.

Problem: *DynaForm* was designed to create boldface, underline, and double strike by controlling the carriage returns and line feeds of the printer. To do boldface for example, *DynaForm* will print the line normally, then print a carriage return, then the word or words that are to be boldfaced. This is repeated four times for boldface, two times for double strike, and for underline it just prints an underline. Because of this, *DynaForm* requires a printer that does *not* produce a line feed when it gets a carriage return from the computer. This flies in the face of Radio Shack OS-9's standard of doing just the opposite. We printed a quick fix that filtered the output of *DynaForm* to strip any line feeds, but this was a pain to use and did not work on the CoCo for some reason.

New Solution — while you are waiting for the new release of *DynaForm* that will really fix this and, by the way, make several improvements, we have discovered a better temporary fix that works . . . almost.

While talking to a user on the phone, it dawned on me that we could probably just null out the line feed in *DynaForm* to cure the problem. This had the advantage of being user-fixable with debug. Hot on the trail, I dug out the listing of *DynaForm* and started to look for the line feed variable. It turned up at offset $0FE1. Just change the $8A to a $80 and we're set. Well, it didn't quite work, so back to the listing, where I found *DynaForm* was sending a bunch of line feeds to bring it to the bottom of the page. Change this to a carriage return and we are all set — almost. Changed offset $0530 from a $27 to a $28 and tried it again. Here is where the 'almost' came in. Everything worked fine, but the pages were short by two lines. Well, after going blind and getting sleepy, I decided to be lazy and just tell you to put '.PL 68' in your file to accommodate the lost two lines, and it would work. It is not as elegant as I would like, but we should have the new *DynaForm* in a month or so, and this was just going to be a quick fix anyway, and I was getting tired, and on and on. Anyway, here is the procedure for the fix.

```
OS9:load df
OS9:debug
Interactive Debugger
DB:l df
    0000 87
DB:. .+530
    0530 27
DB:=28
    0531 78
DB:. .+FE1-530
    0FE1 8A
DB:=80
DB:. .+1327-FE1
    1327 28
DB:=EE
    1328 BE
DB:=13
    1329 E8
DB:=1C
    132A
DB:Q
OS9:ident df -m
OS9:del -x df
OS9:save /d0/cmds/df df
```

The ident will just confirm that the CRC is good. If not, then you did something wrong. Below is a comparison of the "before" (#1) and the "after" (#2) files. The last three numbers are the

CRC values. I had you change them to avoid having to use verify after you save this file. If the ident showed the CRC as good then save this to your CMDS directory with a new name, or first delete the original and use the old name. I'll leave that up to you.

You will not be able to use the built-in boldface, underline and double strike until we get the new version to you. This fix is limited.

Now all you have to do is put '.PL 68' in your files and everything will probably work. Let me know how you made out.

By the way, we will be sending out free updates to all U.S. customers who bought *DynaForm*. Just send in your disk with proof of purchase (invoice etc.), and we'll ship it as soon as it is ready. Try not to call, because the ladies who answer the phone get mad at me when I leak information on a product before it is ready to ship.

Differences

```
byte         #1  #2
---------    --  --

00000530     27  28
00000FE1     8A  80
00001327     28  EE
00001328     BE  13
00001329     E8  1C

Bytes compared: 0000132A
Bytes different:  00000005
```

Printer Control Characters

DynaStar and *DynaForm* support embedded control codes, but we did not do a good enough job of telling you how to use them in the manual, so let me go over it here.

DynaStar has a special feature that is invoked by typing a control P. When you do this, *DynaStar* waits for a control character to be typed by you. If you don't type a control character, then *DynaStar* thinks you want to cancel the operation and goes back to whatever it was doing. Suppose you wanted to send the control code $1D to your printer. $1D is a control] (that is, control key and a closing bracket), so in *DynaStar* type a control P and a control]. You will see a funny triangle-shaped character followed by a ']'. This is the indicator that you have embedded a control] in the text file.

When *DynaForm* processes the file, it looks for that funny triangle (a $80 by the way), and when it sees it, it knows that the next character is a control character that is to be sent to the printer, which it does. You have to precede each control character with a control P. and if you need to send regular characters after the control character, then just type them in. For instance, if you wanted to send a ESC then a 'p' then a control Q to your printer, you would type control P, ESC, p, control P, control Q. It would look like this on the screen (Substitute '^' for the triangle):

```
^[p^Q
```

Get the picture? Play with it for awhile, and you will see that you can do just about anything with this feature.

Disk Drive Advice

At the Princeton RAINBOWfest, we were swamped with questions about what type of disk drives to buy. The choices are getting very complicated, and the prices are getting very low. I was talking to Bob Phillips at Gimix, and he told me that there were 35 Japanese disk drive manufacturers in the market. The competition is fierce, and this means that prices have dropped to the point that anyone can afford to buy any kind of drive they might want. As an example of this, we are buying 80-track, double-sided, half-height drives, guaranteed by the manufacturer for one year, for less than we were buying single-sided, 40-track, full-size drives six months ago.

When you look at how the prices have dropped, you might decide to wait and see if they are going to drop any more. I have been told by people in the know that even the Japanese cannot make drives at these prices for long, and that as soon as inventories get back to normal, prices will either go up or stay at this level. However, nobody really knows for sure just what is going to happen. The best thing to do is buy a name brand drive from a company that will be around for awhile. You will want to get service on the drive someday, although at these prices they are almost disposable.

What size? How many tracks? Single-sided or double? Let me simplify this for you. There are only a few differences between the various drives available. They are: Tracks per inch are either 48 or 96. The standard RS drive is 48tpi. This also holds true for 40-track drives, whether single-sided or double. The number of tracks per inch is the same. The 96tpi drives are usually referred to as 80-track drives. Some companies confuse the issue by calling double-sided 40s, 80-track drives. But they also call double-sided 80s, 160-track drives. All 48tpi drives are compatible. You can put a single-sided disk in a double-sided drive, and it will read it. The other way won't work, unless you formattéd the disk as single-sided. What this all means is that double-sided, 40-track drives are completely compatible with the standard Radio Shack drives and operating system. You only get complicated when you get to the 96tpi.

The differences between single and double-sided drives are as follows. Disk BASIC reads and writes to one side of the disk, 35 tracks. It really doesn't matter what type of drive you have in the system; it will treat it as a single-sided, 35-track disk. This means that all drives will work, but that anything over 35-track, single-sided will be of little use to you if all you use is Disk BASIC. (RAINBOW printed patches to Disk BASIC to use the other side and 40 tracks in a past issue.) If you use FLEX, then it will use any drive currently on the market to its fullest. If you use OS-9, then you will need either SDisk from D.P. Johnson or a similar program from Computerware or other vendors.

My personal favorite is two half-height, double-sided, 40-track drives, in a single vertical case. The price drop on the 80s would make me consider them, but if I did, I would have to keep a 35 or 40 around for copying files to and from standard Radio Shack disks.

Here are some storage comparisons with OS-9 disks:

SS RS 35 track	630 sectors
SS 40 track	720 sectors
DS 40 track	1,440 sectors
DS 80 track	2,880 sectors

This shows that a double-sided 80 has over four-and-a-half times the storage as the standard RS disk. Three of these can be put on the system for a total of 8,640 sectors, or over 2 megabytes of storage! By comparison, four Radio Shack drives have only 2,520 sectors, or 6/10 of a megabyte. If you compare the cost per byte of storage, the 80s come out the best, but the inconvenience may not be worth it. That is why I like the DS 40s. Plenty of room plus compatibility.

What about hard disks? RGS is supposed to be getting a hard disk for us to play with; Dale Puckett is doing a review and then we get to try it. I guess how soon we get it will depend on how much Dale likes it. I will let you know.

That's it for this column. I have been spending my summer sailing and driving a bulldozer over land that will have our new log home on it next year if the banks cooperate. Because of this, I have been lax in writing my column. The next few columns on the 128K upgrades should be a lot of fun to do and read, so I hope that I am forgiven for playing in the sun instead of with my CoCo.

Till next month . . .

Martha Says....

As I told you last month anything they can do around here to reduce the value of my work, they seem to do with great glee. Even to the point of wrecking my nice new header and placing it on the wrong side of the page! I suppose it is the role of the junior reporter to come last, but I don't accept it well I can tell you!

I've been looking at word processors, because I don't like this Telewriter thing they have here.

Graham has always said that he prefers Scriptsit, and I agree that for the novice, such as myself, Scriptsit is good. But it doesn't right hand justify, and they don't like staggered right hand margins around here.

When you have a lot of text in Scriptsit, it can also be very slow, so for this reason, I can understand why one might look elsewhere.

Alternatively, I've seen CC Writer which is almost as vile as OS-9's editor; Telewriter of course, VIP Writer and Stylograph for OS-9.

They say that Elite Writer isn't bad, but I haven't, as yet, seen it. But I wouldn't give you 10 cents for VIP Writer, so that only leaves Telewriter, if you don't have OS-9, or Stylo.

Stylo is really quite good, except that our version appears to be the OS8 version and is written for XMODE/12, so of course, because Graham has the micro dot with the instructions, we never seem to be able to get to see the whole program work.

So I use Telewriter 64. Why do I use Telewriter? well what else is there?

I had a letter from that lady that keeps writing to me, again this month. I too, took my CoCo on holidays with me. And like her I was able to set myself up on the beach with the computer, monitor, drives, etc. I had some problems with people pulling the power cord at first, but I fixed that by baring a 300mm length of wire (both the live & the other one - whatever they call it). In salt encrusted sand, I found that this was sufficient to keep folk back to a distance of about six or seven feet. After the first couple of kids, people started to learn.

Anyway, we are back at work now and wouldn't you want to know, bits of the computer are starting to rust.

I phoned Tandy about this, but as yet, we have notarrived at any amicable resolution. If they don't respond soon, I'm going to reprint my letters to them AND any replies they deign to make, in this column.

Jim tells me that Graphicom II is about to hit the streets, and it promises to be better than ever. I like drawing pictures with Graphicom, but when I drew a picture of Peter Collison's wife and sent it to him he withdrew the double page ad that he was going to put in AUST COCO. I'll show you next month!

From P 42 ...

nection just by heating up the joint again. After it cools, the joint should be shiny and smooth. Practice several times until you get the hang of it. There is one more thing to remember; after all the soldering is done, clean the PCB with Radio Shack rosin cleaner-remover.

Now that you feel more at ease with soldering, it is time to put your newly acquired talent to work. Yes, the pilot light. There are only four parts to this project. The LED, a 1K (K=1000) ohm ½ watt resistor, and two short lengths of colored wire (preferably red and black). That is it. Examine the schematic in Figure 2. This is a diagram on how the components connect together and to the computer. The first thing to do is mount the LED. You must decide where to put it. After that, you must check that when mounted, it does not interfere with the normal operation of the computer, i.e., short out or lean on other components and does not prevent the cover from fitting properly.

Mount the LED by drilling a ¼-inch hole where the LED is to be mounted. Cut both sides of the resistor leads to about ¼ inch. Solder one side of the resistor to the long end of the LED. Solder one end of the red wire to the other end of the resistor. Solder one end of the black wire to the other (short) end of the LED. Twist the two wires together lightly and cut them about 18 inches long. This should be long enough to have the cover out of your way if ever you want to open the computer again.

Now, solder the other end of the red wire to inside of pin 9 of the edge connector. That is the five volts side. How do you get to pin 9? Simple, just start counting from the end closest to the back of the computer. All the top pins are odd numbered, so count 1, 3, 5, 7, 9. Make sure that you don't short out two pins with the solder. Finally, solder the black wire to pin 33, count that one the same way. Pin 33 is the ground return pin.

Place the cover on top of the computer (without the screws for now) and turn the computer on. The LED should turn on. If not, chances are that you got the wires to the LED reversed. In that case, unsolder the resistor and the black wire to the LED and resolder them the other way. Otherwise, you should not have any problems. Tuck the wire in the cover and place the cover back on. Make sure that the wire does not stick out and that the keyboard is sitting on the pegs properly. Turn the computer over and replace the screws. There you are, your first modification to your computer. Now doesn't that make your day?

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

Casula Hobbies

AUSTRALIAN
MODEL RAILWAY
SPECIALISTS

We stock Train sets of
American,
English,
and Australian Prototype.

DUE IN FEBRUARY:

the CoCo Connection.

Connect your computer to your model railway, burgular alarm, model cars, etc, for the ultimate in control.

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

SHOP
245 NORTHUMBERLAND ST.,
LIVERPOOL. 2170

Now, for all versions of COCO, on disk or on cassette, the definitive EXPANDED COLOR BASIC and the BASSEMBLER

EXPANDED BASIC is designed to be added on to Extended Basic and includes the following new features:

Printing text in all modes and colors
A special 51x24 screen for PMODE4
Scrolling of any screen section any way
Extra colors in a new mode
Borders for the text screen
Extra graphics pages
User definable sound effects
REPEAT...UNTIL loops
Multiline IF...THEN...ELSE statements
Procedures in addition to subroutines
Local and lower case variables
Full ON-ERROR implementation
Breakdisabling, auto-line numbering
On-screen editing, 10 function keys
User-definable characters /printer widths
Single key entry of most BASIC words
Ability to execute strings as commands.

The BASSEMBLER is a UNIQUE macro-assembler that allows you to inter-mix assembly language and BASIC in the same program - you can literally have one line assembly and the next line BASIC.
The BASSEMBLER understands the complete 6809 assembly language, macros and other pseudo-ops, labels and LINE NUMBERS too.
A disassembler/moniter Program is included with the BASSEMBLER.

Easy to follow instructions and demonstration programs come with the BASSEMBLER and EXPANDED BASIC.
64K is required. Specify disk or tape (giving model of recorder) and specify COCO model (1 or 2).

The BASSEMBLER	$15
EXPANDED COLOR BASIC	$30
EXPANDED BASIC + BASSEMBLER	$40
DISK USERS add an extra	$ 3

Buy now, direct from the author:
TINO DELBOURGO, c/- NIGEL MANN,
145 Nelson Road, Sandy Bay, Hobart, Tasmania
7005. Phone (002) 25 1424

64k Upgrade
Software & Hardware TO ORDER
Education Software a specialty
KEITH GALLAGHER
02 627 4627

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

user group CONTACTS

(Stop between numbers = b.h. else a.h.; but, hyphen between = both.)

ADELAIDE	JOHN HAINES	08 278 3560
ADELAIDE NTH	STVN EISENBERG	08 250 6214
ALBURY	RON DUNCAN	060 43 1031
ARMIDALE	TOM STUART	
BAIRNSDALE	COLIN LEHMANN	051 57 1545
BALLARAT	MARK BEVELANDER	053 32
BALLARAT	MARK BEVELANDER	053 32 6733
BANKSTOWN	KEN HAYWARD	02 759 2227
BLACKTOWN	KEITH GALLAGHER	02-627-4627
BLAXLAND	BRUCE SULLIVAN	047 39 3903
BOWEN	TONY EVANS	077 86 2220
BRASSALL	BOB UNSWORTH	07 201 8659
BRIGHTON	GLENN DAVIES	08 296 7477
BRISBANE EAST	ROB THOMPSON	07 848 5512
BRISBANE STH	PATRIC SIMONIS	07 209 3177
BRISBANE SW	GRAHAM BUTCHER	07 376 3400
BRISBANE WEST	BRIAN DOUGAN	07 30 2072
BUNDABERG	JIM McPHERSON	071 72 8329
CAMBERWELL	TONY BALDWIN	03 728 3676
CAMPBELLTOWN	LEO GINLEY	02 605 4572
CANBERRA	SHAUN WILSON	062 51 2339
CAULFIELD	JEFF SHEEN	03 528 3724
CHATSWOOD	BILL O'DONNELL	02 411 3336
CHURCHILL	GEOFF SPOWART	051 22 1389
COLYTON TEENS	DWAYNE MANSON	02 623 5805
COOMA	ROSS PRATT	0648 23 065
DANDENONG	BRETT CRUICKSHANK	03 547 6604
DARWIN	BRENTON PRIOR	089.81.7766
DENILIQUIN	WAYNE PATTERSON	058 81 3014
DUBBO	GRAEME CLARKE	068 89 2095
EMERALD	LEIGH EAMES	059 68 3392
FORSTER	GARY BAILEY	065 54 5029
FRANKSTON	BOB HAYTER	03.783.9748
GIPPSLAND STH	PAT KERMODE	056 74 4583
GOLD COAST	SHERYL BENTICK	075-39-2003
GOSFORD	PETER SEIFERT	043 32 7874
GRAFTON	DAVID HULME	066.42.0627
GREENACRES	BETTY LITTLE	08 261 4083
HOBART	BOB DELBOURGO	002 25 3896
IPSWICH	MILTON ROWE	07 281 4059
KENMORE	GRAHAM BUTCHER	07 376 3400
LEETON	CHRIS NAGLE	069 53 2969
LITHGOW	STUART RAYNER	063 51 4214
LIVERPOOL	LEONIE DUGGAN	02-607-3791
MACKAY	LEN MALONEY	079511333x782
MACLEOD	ROBIN ZIUKELIS	03 450211x465
MacQUARIEFIELDS	KIETH ROACH	02 618 2858
MAFFRA	MAX HUCKERBY	051 45 4315
MAITLAND	LYN DAWSON	049 49 8144
MELBOURNE	JEFF SHEEN	03 528 3724
MELTON	MARIO GERADA	03 743 1323
MILDURA	SCOTT HEWISON	050 23 6016
MOE	STEPHEN SEMPLE	051 27 6841
MORPHETTVALE	KEN RICHARDS	08 384 4503
MOREE	ALF BATE	067 52 2465
MORWELL	GEORGE FRANCIS	051 34 5175
MT ISA	PAUL BOUCKLEY-SIMONS	077 43 6280
MUDGEE	BRIAN STONE	063-72-1958
MURGON	PETER ANGEL	071 68 1628
NAMBUCCA HDS	WENDY PETERSON	065 68 6723
NEWCASTLE	LYN DAWSON	049 49 8144
NOARLUNGA	ROBBIE DALZELL	08 386 1647
NOWRA	ROY LOPEZ	044 48 7031
PARKES	DAVID SMALL	068 62 2682
PENRITH	TOM LEHANE	047-31-5303
PERTH	IAN MACLEOD	09 448 2136
PYMBLE	MARTIN WELLS	02 449 2077
PORT MacQUARIE	RON LALOR	065 83 8223
PORT NOARLUNGA	ROB DALZELL	08 386 1647
PORT PIRIE	KEVIN GOWAN	086 32 1368
RINGWOOD	ANDREW RAWLINGS	03 726 6521
ROCKHAMPTON	KEIRAN SIMPSON	079 28 6162
ROCKHAMPTON MiCo	TIM SHANK	079 28 1846
ROSEVILLE	KEN UZZELL	02 467 1619
SALE	BRYAN McHUGH	051 44 4792
SANDGATE	MARK MIGHELL	07 269 5090
SEACOMBE HTS	GLENN DAVIS	08 296 7477
SPRINGWOOD	DAVID SEAMONS	047 51 2107
STURT	MARY DAVIS	08 296 7477
SUNBURY	JACK SMIT	03.744.1355
SUTHERLAND	IAN ANNABEL	02 528 3391
SWAN HILL	BARRIE GERRAND	050.32.3838
SYDNEY EAST	BOB JONES	02-331-4621
SYDNEY TEENS	ROD HOSKINSON	02 48 5948
TAMWORTH	ROBERT WEBB	067 65 7256
TAHMOOR	GARY SYLVESTER	046 81 9318
TONGALLA	TONY HILLIS	058 59 2251
TOOWOOMBA	MIKE CARTER	076 35 6911
" BEGIN NTH	DAVID PROUT	076.32.7533
" BEGIN STH	LEW GERSEKOWSKI	076 35 8264
" ADVANCED	GRAHAM BURGESS	076 30 4259
TOWNSVILLE	JOHN O'CALLAGHAN	077 73 2064
TRARALGON	MORRIS GRADY	051 66 1331
UPPER HUNTER	TERRY GRAVOLIN	065 45 1698
WAGGA WAGGA	BRUCE KING	069 25 3091
WESTLEIGH	ATHALIE SMART	02 848 8830
WHYALLA NORRIE	CHRIS HUNTER	086 45 3395
WOLLONGONG	BRIAN McCAULEY	042 71 4265
WONTHAGGI	PAT KERMODE	056 74 4583
SPECIAL INTEREST GROUPS		
BRIZBIZ	BRIAN BERE-STREETER	07 349 4696
BRISBANE OS9	JACK FRICKER	07 262 8869
CARLISLE MiCo	STUART HALL	08 361 1922
BLAXLAND OS9	BOB THOMSON	047 39 3903
BLAXLAND 128K	BOB THOMSON	047 39 3903