For your TANDY Color Computer

Registered by Australian Post — Publication No. QBG 4009

PNG K5.95 NZ $3.95
$4.50

AUSTRALIAN RAINBOW

December, 1985

No.54

NOW AVAILABLE ON THE **QUICKSHIP PROGRAM** from your TANDY STORE
R90-9715

CoCo Max®

COLORWARE

File Edit Goodies Font Style

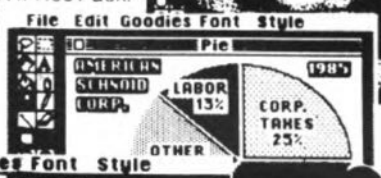
untitled

What is CoCo Max? Simply the most incredible graphic and text creation "system" you've ever seen. You will be generating images like these in minutes.

Who is CoCo Max for? Anyone who has ever held a pencil for fun, school or business will love it. A 6-year old will have fun doodling, a 15-year-old will do class projects, and adults will play with it for hours before starting useful applications *(illustrations, artwork, business graphics, flyers; charts, memos, etc.)* This is one of the rare programs that will be enjoyed by the whole family.

Just point and click to activate *CoCo Max*'s powerful features, including: mirror images, rubber banding, edge tracing, zoom, lasso, sixteen colors, thirty patterns, thirty-two paint brush shapes and textures, undo, rubber stamping, icons, pull down menus, pencil, spray can, eraser, toolbox, and so on, and so on...

The Hi-Res Input Pack is the key to *CoCo Max*'s unmatched power. It gives you direct access to the 49,152 pixels on your screen. That's **12 times** the regular joystick input. It looks like a ROM pack, and plugs into your CoCo or Multipak. Your regular joystick, mouse, or Koala Pad simply plugs into the Hi-Res Pack. Whether you are an artist or an accountant, even if you are the "I can't draw" type, you will be amazed by what you can do with *CoCo Max*.



AMAZING!! AMAZING!! AMAZING!!

Price Includes
- Cocomax disk
- Hi-RES Input Pack
- Detailed Manual

$99.95

Disk System: a Multipak or 'Y' Adaptor' is needed to plug the disk pack and the Hi-RES pack

SYSTEM REQUIRED:
- Any 64K COCO
- A standard Joystick, Mouse or Koala Pad

All these pictures are **unretouched** screen photos or printouts (on an Epson RX-80)

CFM

COMPUTERWARE FOR MICROS.

bankcard welcome here

VISA

Peter & Jillian Collison
11 Grantley Avenue,
Rostrevor S.A. 5073
Phone: (08) 336 6588

CFM COMPUTERWARE FOR MICROS.

VISA

bankcard welcome here

Peter Collison
11 Grantley Avenue,
Rostrevor S.A. 5073
Phone: (08) 336 6588

DRIVE 0

Panasonic 40-track single-sided disk drive, slimline case and power supply. Plus CFM controller complete with B-DOS.

Only $399.00
(Plus $8.00 Shipping)

DRIVE 1

Panasonic 40-track single-sided disk drive, slimline case and power supply.

Only $170.00
(Plus $8.00 Shipping)

B-DOS ©

USER FRIENDLY DISK OPERATING SYSTEM NOW AVAILABLE ON DISK FOR YOUR CO-CO

AUTO (line number) : 35-40 TRACKS
ERROR TRAPPING : BAUD (value)
COLD (cold start) : PDIR (print dir)
PCLEAR (16 pages) : SWAP (var1, var2)
OS9/DOS (included) : UNNEW.....

A COMPREHENSIVE MANUAL + MUCH MORE
*** ONLY $44.95 ***

NOW DRIVE ZERO! $399.

UPGRADE-KITS

Up-grade your short case 16K Coco II to 64K with our up-grade kit. Complete with instructions.

Only $85.00

• LOWER CASE KIT •

TRUE LOWER CASE PLUS REVERSE VIDEO

Now with dual 5:7 and 7:9 characters. Use your COLOR BURNER to put in your own special character sets. (optional)

For visual comfort and Pro programming send for LOWERKIT II-C . . . **$89.95**

• COLOR BURNER •

An EPROM programmer is the perfect tool for creating your own program packs. The COLOR BURNER programs the most popular erasable, programmable eproms: 2716(2K), 2732(4K), 2764(8K), 27128(16K) and 68764/66(8K).

EASY TO USE - STEP BY STEP INSTRUCTIONS.

** ONLY $99.95 **

SUPER BACK UP UTILITY ©

. . . WITH S.B.U. FROM COMPUTIZE YOU'LL NEVER NEED ANOTHER BACK-UP UTILITY FOR YOUR CO-CO!!!

1. **Tape to Tape**
2. **Tape to Disk**
3. **Auto Relocate**
4. **Disk to Tape**
5. **Disk to Disk**

- **Menu Driven!**
- **Requires 32K Extended Co-Co**
- **Requires 1 or 2 Drives**
- **All Machine Language!!!**

*** ONLY $49.95 ***
(SUPPLIED ON DISK)

** COLOR QUAVER **

THE ULTIMATE ALL-SOFTWARE MUSIC SYNTHESIZER! COLOR QUAVER, an amazing Music experience from your Color Computer.

FEATURES:

- Real electronic music synthesis that is more than bleeps.
- Full four part harmony, all in precise tempered tuning.
- Five usable octaves, variable tempos, rhythms from 32nd note to whole note.
- Fast compiler provides finished music in five seconds . . . up to 250 notes per voice line, 1000 notes in all.
- Over 40 pages of instructions, explanations, hints, listings and samples.

** INCREDIBLE VALUE AT ONLY . . . $39.95 ** (tape only)

AUSTRALIAN RAINBOW

INDEX

UPPERCASE = ARTICLE + PROGRAM
lowercase = article only

AUSTRALIAN RAINBOW Publisher and Editor Graham Morphett Co editor Kevin Mischewski Assistant Editor Sonya Young With grateful assistance from Brian Dougan, Richard and Judy Bob Thomson Paul Humphries, Alex Hartmann, Michael Horn Jim and Sheryl Bentick Annette Morphett Cover Art Jim Bentick
ADVERTISING DEADLINES. The 7th of the proceeding month of publication All advertising is arranged through TeToAdvertising, PO Box 8730, Gold Coast Mail Centre, Qld 4217
OS 9 Kevin Holmes is the contact for os 9 information He also has access to OS 9 Software from the US His address is: 30 Pearson St, Morors, NSW, 2350
All programmes in this issue of AUSTRALIAN RAINBOW are available on Rainbow on Tape

Telephone. 075 51 0015 Voice, 075 32 8370 CoCoLink
Printed by: Australian Rainbow Magazine PO Box 1742 Southport Qld 4215
Founder GREG WILSON

Welcome to this, our last issue of Australian Rainbow for 1985.

At the end of 1985, it is apparent that CoCo, despite what Tandy's opposition continually predicts, is very much alive and well. And is still coming up with the goods. Our magazine this month illustrates the point very ably.

So if you have just bought a CoCo, or are contemplating such a purchase, never fear, you are in good company, your computer is better than most, and as capable as the latest from other companies, and you are about to have a lot of fun!

We are going to bow to your wishes - as conveyed in the surveys, and slash our zeros in print-outs of programs from now on.

There will be a carry over period, where some programs are so treated, and some are not. But by February, most programs will be listed this way.

See! Surveys do work!

Our office will be closed from 20th December to 7th January.

We've had an eventful year and we're all looking forward to the break.

A pal of ours, Peter Kennedy, has started his own Tandy Computer Dealership in Brisbane's City Plaza.

Tandy have made some very brave decisions this year. In fact history may well record 1985 as Tandy's year of change.

Tandy have gone from a company which wanted to do it all by themselves, to one which enjoys working with external organisations - such as ourselves.

But placing a dealer in the heart of what would be considered 'home territory' by the Tandy establishment, took a great deal of fortitude.

We look forward to similar moves in other states, and we wish the ventures every success.

What is the difference to the man in the street?

Well apart from the ability to see Tandy computers actually sitting beside their competitors, you can also obtain a broader range of soft and hardware from such a store.

If you are in Brisbane sometime, call in and tell Peter I said he'd make you a cup of coffee!

The man behind these changes has been Mike Murray, (General Manager, Tandy Australia), whose term in Australia is complete, and so Tandy have seen fit to repatriate him to his native Fort Worth.

We wish Mike all the best in the coming years, and we hope that he has many pleasant memories of Australia. We appreciate very much his interest in our enterprise and in the User Groups of Australia.

In his place comes Joe Tanner, who has also received an undetermined sentence.

These guys must be really bad to be sent here without any idea of how long their banishment is going to be! At least our convict fathers were told if it was to be for life!

Of course with such changes, there is always a little uncertainty about what changes a new man is going to make - especially with the staff.

I honestly have been told nothing, but I wouldn't be feeling too secure if I were in a store that is not producing!

I have long encouraged you to get to know your local Tandy person better. You don't HAVE to buy him / her a drink, but I hear it doesn't hurt!

I'd like to take that one a little further.

If you need further info on a product, or if you have a problem your local Tandy store seems to be experiencing trouble with, or if you get a brilliant idea, why not call one of the heavies at Tandy Head Office.

The phone number is 02-675-1222, and we will be introducing you to a broader cross-section of the people and their functions over the coming months. But here are three or four to start with:

* Ken Allen - Computer Buyer. Ken likes to hear of anything special you are doing with your computer. Things that are away from the mainstream of usage. He also needs to know about equipment that breaks down too often.

* Karel Davey - Education Co-ordinator for all of Australia minus Queensland. Having been a School Principal herself, Karel understands the administrative and the educational problems associated with getting a school's involvement with computers under way. A discussion with her along these lines is never a waste of time.

* Joe Tanner - Ring him up and welcome him to Australia. Nothing like being able to say that you've had a little word with the GM if things get difficult!

* Lisle Jones - Agriculture Co-ordinator. I never speak with Lisle without getting a mental picture of him in a dairy - one hand busily milking a cow, the other entering some statistic about the cow's performance in DynaCalc!

Seriously though, if you are on the land, then keep in touch with Lisle, he knows about the latest programs, and is always interested in hearing about the programs you use on your farm.

In fact right now, if you have a program you use on the farm with your CoCo, Lisle is especially interested to hear about it.

These are important and busy people, and you might have to call once or twice to talk with them, but they will

appreciate your call, if you have something to say or discuss which is concise and which will benefit either yourself or Tandy. Don't be too scared to call them - they are there if you need them.

Speaking of Tandy, their new 1200/75 baud modem (Cat. No 269404) is now available. At $399.95, it is priced midway between the cheapies and some of the expensive units.

Tandy's Bulletin Board is up and running too.

Called Tandy Access, you can phone it on 02-625-8071. You need a modem and a terminal program to do this, but you do not need a Tandy computer. Tandy Access operates at 300 baud.

We are on the board, and you can leave messages for us there, although we would prefer you leave them on Minerva, OTC's Data Base, of which further details in a moment.

CoCoTex, Ron Wright's excellent program for CoCo which allows access to Viatel is available from Tandy for $79.95.

We are on Viatel too, and you are invited to leave messages there for us if you wish.

We have also joined Minerva, and following a trial period, may move CoCoLink to it.

In the meantime, we invite you to call us on Minerva. The cost of the telephone call is that of a local call, you are charged $15.00 per hour usage, and you are charged $0.40 / month / K storage.

All in all, not a bad deal.

Call Andrew Maston at OTC in Sydney to obtain the appropriate form.

We have been working on the CoCoConnection for a year now. We had planned that it would be available quite some time ago, but we changed the prototype a couple of times and this held up production.

But at last the CoCoConnection is available, and we are very pleased with it.

If you have a job for the computer which requires that it be interfaced with the real world - then this is the unit for you!

Christmas is a time when one can become caught up in the razzle of commerce and very nearly forget the more important things of life.

Our hope for you this Christmas is that you will find time to enjoy your family, and that you will find tranquility in an increasingly spinning world.

On behalf of Jim and Sheryl and their kids Gemma and Katie; Jim Rogers; Brian and Bev, and their kids Jason and Natasha; Alex Hartmann; Rainer and Michael Horn; Paul Humphreys; Kevin and Glen Mischewsky; Barry and Ziggy Cawley and their kids Claudia and Clifford; Janet Stott; Rodney and Sonya Young; and Annette, Katie and myself - have a happy christmas, and please stay safe on the roads!

One-Liner Contest Winner . . .

This program figures monthly loan payments. Just enter the principal, interest rate and number of years.

J.C. Longo
Oshawa, Ontario

The listing:

```
1Ø CLS:PRINT:INPUT" AMOUNT BORRO
WED";A:INPUT" INT. RATE (%)";R:I
NPUT" AMORTZN. PERIOD (YRS)";Y:K
=1+R/2ØØ:F=K^(1/6)-1:D=1-K^(-2*Y
):PRINT@161,"THE MONTHLY PAYMENT
 IS";INT(1ØØ*A*F/D+.5)/1ØØ:PRINT
@193,"MOLY. INT. FACTOR: ";:PRIN
TUSING".##########";F
```

One-Liner Contest Winner . . .

This one-liner animates a running stick figure across the screen. It demonstrates some principles of flicker-free animation, the use of a variable as a horizontal coordinate in a DRAW string (=H;) and the use of a short PLAY pause as a delay.

The listing:

```
1 PCLEAR8:A$(1)="BR1ØU8NU6G4R3":
A$(3)="E8NU6D5E4":A$(2)="BU4E4R4
NU6F4R3":PMODE4,1:PCLS1:SCREEN1,
1:DRAW"BMØ,96CØR255":FORX=24ØTO1
Ø STEP-3Ø:H=X:FORY=1TO3:PMODEØ,6
:PCLS1:PMODE4,5:DRAW"CØBM=H;,95"
+A$(Y):PLAY"P12Ø":PCOPY6TO2:H=H-
1Ø:NEXT:NEXT:GOTO1
```

Stan Osterbauer
Sequim, WA

EDUCATION PAGE

When I said last month that this column would be on holidays this month, I didn't figure on having the news and articles that have turned up!

The NSW Education Dept contract had still not been let as I wrote this, so poor old Karel Davey (Tandy's everywhere but Queensland Education person) dosen't know whether she is alive or dead - but, she has broad shoulders so she'll handle the wait OK I think!

Don't forget that the February edition of Australian CoCo is an Education issue. We still need programs and articles for that magazine too - so if you have something good - get it in quick!

We have a very interesting program of software development lined up for next year. In conjunction with the Australian Software Development Council and Tandy, we will be initiating the development of a number of programs for release to schools which are curriculum related. The main thrust of this work will get underway in February.

Finally, and further to my words in the editorial this month, if as an Educator, you have not made yourself known to either Karel Davey or Leo Wilson, Tandy's Education people, then do so.

It really does pay to know the right people, and as these two are so approachable, in this case it is also easy! They can't help you, if they don't know of your needs.

Traveling for CoCo

This month I have been to two distinctly different areas of Australia in our never ending quest to get the schools to recognise the value of having a CoCo around!

The first trip was to St George in the central west of Queensland.

There I attended the Annual General Meeting of the Isolated Children Parents' Association.

The main topic of interest was the integration of computer and satellite technology to provide classroom-like instruction for isolated kids.

I attended with Tandy's Leo Wilson and Mike Raasch, and Allan Williams from Pencom - the premere Education software supplier.

Currently evaluations of various systems are being considered, so it is unfortunate to report that no system will become operative in the near future; but anytime after mid year 1986, the Queensland Education Department may well have teachers who will be working on TV, the replies of their class coming in on a sophisticated bulletin board.

I took a heap of software with me, but spent most time on the CoCoConnection and Ears.

We had hundreds of kids through, and they were rapt in these two programs!

In fact I was very proud to be an Australian and to meet these children. They are a special breed of super nice people. To Scrunge, Glen, Sarah, and specially all the kids from Bollon, who traveled over one hundred miles on bulldusted roads to see us - Hi Ya!!

The CoCoConnection is a new phase in the use of computers in schools, and teachers and kids voted it a big success. The reason for this is that it gives one the opportunity to explore the physical world, whilst taking measurements which can be stored for later processing and evaluation.

We only used the Quizmaster (Nov '85 AUST RAINBOW) program this time, but temperature measurement, Ph measurement, colour change and a range of other simple experiments are possible. AND THEN, experiments in control - Robotics, Burgular Alarms, control of say, water reticulation units, can also be achieved.

All in all, the trip was very worthwhile, and once more proved that you don't have to pay a lot of money to get a worthwhile computer which can do virtually all the jobs one might require of a computer.

The trip to Kempsey was different again. Kempsey is on the coastal plain of NSW, so for a start, the topography alone was very different.

I was to attend a meeting of the Nulla Nulla Creek school for Aboriginies, which is some 60 K west of Kempsey.

This school has been built by hand by the people there. They even made their own bricks from the nearby creek mud.

But don't think that this school is some gunya out in the sticks, it is a modern well equipped school!

All classes from first class through to the final year of high school are taught; the NSW Department of Education's cirriculum being followed.

Additional to these studies, is a course in Aboriginal Heritage and Culture.

Achademically, the school is very successful with many students achieving very high passes in both the School Certificate and the Higher School Certificate.

The reason is of course, that the kids are given some understanding of who they are. This makes a big difference - the contrast with the achademic and job aquisition skills of the "town" aborigines in Kempsey being the telling point.

I went there, along with Tandy's Karel Davey and the local Tandy Dealer - Rick, the bush basher.

The purpose was to put CoCo through its hoops and demonstrate its suitability for such a school.

We did that, and the CoCo was shown to be a most acceptable machine, but what struck me most was the beauty of the people there.

Not so much physical, but mental. It made me wish I could go to school in Nulla Nulla Creek!

ON BEING A TEACHER

(Like Tom, I also did Ag at Armidale Teachers' College, so this story has special meaning for myself, and a couple of mates, Greg and Bear, who read this magazine.
In fact I thought this was Bear's work at first, but then I remembered that Bear isn't that good at snooker! G.)

by Tom Brittons' son

My father, Thomas William Britton left Yanco Agricultural High School in December, 1951, at the ripe old age of seventeen. He had just completed five years of "study" at said educational establishment, and had been able to come up with a Leaving Certificate somehow. He said they must have been giving them for football or cricket or some other sporting activity.

After deciding that his education was still sadly lacking, he proceeded to Armidale Teachers' College where he trained to be an Agricultural/Science teacher. (He mentioned here that in 30 years with the N.S.W. Department of Education, he is still waiting for his first appointment as a teacher of Agriculture. He wondered if there were any vacancies at his old 'alma mater' - plants still grow the same way don't they?)

How did he decide to be a teacher ? He said that after observing the lives of the pedagogues at Y.A.H.S from 1946-51 he believed it to be the only job. He still believes this to be so, but he still can't get a job as an Ag. teacher,let alone be appointed to Yanco.

He says the training for the job (Ag. teacher), consisted mainly of cricket and football with dai-ly training visits to the colle-ge snooker room.

His other recollection of college life, that he would tell me about, was of an excellent art collection.

I asked him who paid for all this and he said that he is still paying for it - though the Education Department did contribute in a very minor way. My grandmother claims she did all the paying.

Actually he had what was known as a Teachers' College Scholarship - given for his snooker ability, he claims.

For the past 30 years he has commenced work at 9.00 a.m. and has been in the pub by 3.10 p.m. He says there is plenty of opportunity for overtime if you are so inclined - like marking the kid's books, or doing some preparation. However as there was no provision for overtime pay he had never done any overtime. He said that he did mark a book once but that the parent of said lucky child claimed he had missed a spelling mistake - so he hasn't marked one since - he thinks it was 1955 that this happened.

When asked about general working conditions he settled back with a 'tinny' and proceeded thus:

First - he was placed in a room 18 X 18 with 54, mixed sex kids. He was told by his Principal (maybe it was Headmaster in those days) to educate them all. What they were to be taught was in the curriculum and it was his responsibility to make out a plan of the work he was to give them - this was to be planned 5 weeks in advance and placed on the Headmaster's table on the Monday morning of

the first week of the period.

Failure to do so led to being kept in after school to get it done. He said that he had to do playground duty at least twice each day as well as being 'on call' at all other times.

He recalls that at one inspection he was asked why he hadn't filed off the sharp edges of the 'tank-stands'. He said the Inspector wasn't amused by his reply.

In winter the rooms were heated by 'Warmrays' if you could get someone to cut and 'bring in' the wood.

In summer you opened the window he said, and hoped that the dust would not make the 110 degree temperatures too oppressive.

I mentioned 'super' while he was taking a sip from the 'tinny'.

A wonderful thing, he said, when I took it out. Now it's not nearly as attractive - since Bob increased the tax on his 'lump sum'. Mum would do pretty well though if he were unfortunate enough to 'kick the bucket'.

While he thought about this I mentioned 'pay' - his face clouded and became ruddy. I changed the subject immediately. I gather it's not too bad as he has managed to send us to private or boarding schools - three boys.

I mentioned holidays - he said he didn't get any but that the kids got about ten weeks each year.

He had been sick for some time last year so I asked him how he had got on for sick leave - great, he said. The Department sure looks after you if you don't get sick. He said he had over 200 sickies left, plus this year's entitlement, so the provisions must be pretty good. He said that as he hadn't taken much sick leave in the last 30 years that he could probably be off for 12 months without any loss in pay - something to do with his good record and the Director's discretion, though he said some Directors didn't have much of it.

He said that he wasn't a member of a Union as such - he was a member of a professional organisation known as the Teachers' Federation. He says provided you leave everything to the 'cons' things work out O.K. - provided you don't take things too seriously. Provided you remember that you make your own decisions no matter what the membership says, you'll be O.K.

I still don't understand this bit. When I asked him if you had to join a Union he said no preference was given to Union members, whatever that means.

When I asked him how he came to be in Coolamon from 1954 to 1961 he started to laugh. When he had composed himself he said that unlike everyone else looking for their first appointment he applied for the North Coast. Apparently Coolamon was the nearest they had available.

Now he started to reminisce - oh, for those working conditions at Coolamon. There he was - a nineteen year old budding genius about to take these slow moving, wheat farmers' kids by storm. The town owed him the best in accommodation - the pub.

He stayed there for five years - the last two of his seven years there living on a farm with a golfing friend. Why did he leave, I ventured? An Inspector thought it best for the town that he move and so Holbrook, on the Hume Hwy, became his home.

Once again he was to educate the secondary kids of that Central School town. Working conditions were the same - primitive - though class sizes had fallen to 48. Nightlife was great if you enjoyed talking to snooker balls. The boarding conditions had improved greatly too.

The landlord had a daughter. He said he used to stay at the Club each night until everyone, the daughter included, were asleep.

He says he doesn't know whether these conditions were an advantage or a disadvantage.

He became engaged to be married while at Holbrook - and not to the landlord's daughter (- he sighed with relief).

He wasn't sure how much his bed cost him but he thought it was about $10.00 in today's money. At the pub in Coolamon it had cost him $4.00 for a while but had increased to $10.00 by the time he had left.

When he moved back to Sydney, and Liverpool Boys' High, he married and bought his own home. Seven years at Liverpool and then to the bush again.

Here working conditions were really great. It hadn't rained at Binya for four years. The dust in the school yard was at least a foot thick and continually wafted in from the powdery road outside the gate. Thank heavens the cleaner (school) was a lovely woman.

In the first four weeks the lowest temperature was 105 degrees, at 3 a.m. During the day it ranged between 110 and 123, in the waterbag. Even the Kero T.V. played up, he said.

After four weeks the heavens opened and it rained the next month. Now mud was eighteen inches thick and water laped the verandah of the residence. The authorities had built the school and house in a disused creek-bed.

The rain improved life in the district as the 'teacher' was credited with bringing the rain. He was also credited with bringing the mice plague later that year. He was also credited with running the best P&C Meetings - they finished out on the road about 6 a.m. when the last of the flagons had been emptied.

I asked why he had left. He muttered something about school inspectors - I didn't catch it all and thought it best not to pursue the matter any further.

He moved from Binya to Bingara. What a move - through hundreds of miles of floodwaters. Two years later he was back in the city a sadder and wiser man.

He now struggles to his local High School, about a kilometre away where class sizes are between 10 and 15 for his G.A's. He wishes he was back in Coolamon with 60 kids again.

I didn't bother to ask if he had always had the same job. Somehow it seemed a waste of time.

I asked him when he was able to get his first car. He said it came with his wife. Ever since he traded it in on a Kingswood he has had trouble with his wife.

I said, "Dad, what are the pros and cons of your job?" When I came back from school last Monday night, he was still telling me about the hard times he had encountered and not one good thing had I heard about teaching. I think he was a 'bit cheesed off'.

Working With The Electronic Book Part 1

By Steve Blyn

This month we will present the first of a two-part series on Radio Shack's Electronic Learning Book. It is one of the newer peripherals Radio Shack has produced for the Color Computer. We think it can be a very valuable educational tool.

As in the case of several other Radio Shack peripherals, the Electronic Book comes with no documentation. I don't know what the reason behind this is, but it is not the first time. The original X-Pad was equipped with no documentation or software, and the Touch-Pad, similarly, arrived nude.

There are software packages available for the Electronic Book (the ones I've seen are very enjoyable) and these do include documentation. The price of a package is close to the price of the book. I'm sure many potential buyers of the Book will probably like to create programs for it to add to their purchased collection. It is our intent to help RAINBOW readers write programs for the Electronic Book; it is an interesting piece of hardware that should not be overlooked due to lack of user instructions.

Now for the good news. The Book is a wonderful idea and a pleasure to use. This month's column includes two programs. The first demonstrates what the Electronic Book does; the second is a simple number-matching game for preschoolers. Next month, we will go into programs for older children. In order to use the programs, an Electronic Book must be used.

In case you've never seen one, the Electronic Book is a loose-leaf notebook. The inside has a full-size vinyl page with huge numbers from one to 12 imprinted on the vinyl. The idea is for the child to touch a number to give a response. The Book comes with a cable that plugs into the right joystick port. A different joystick value is issued each time one of the numbers in the Book is pressed. Wires underneath the vinyl accomplish this task. The secret of programming with the Book is to be able to read these values and, therefore, determine which number is pressed.

The first program (Listing 1) shows what the Book does. A number is pressed in the Book, and the corresponding number is circled on the replicated screen version of the Book's vinyl page.

Following is a table of the joystick values that are emitted when each number is pressed. JOYSTK(0) tells us what the X-coordinate reading is and JOYSTK(1) is the Y-coordinate value.

JOYSTK(0) Value	JOYSTK(1) Value	Number Pressed
63	0	1
53	0	2
41 or 42	0	3
8	0	4
19	0	5
30	0	6
0	63	7
0	52	8
0	40	9
0	7	10
0	18	11
0	28	12

These values are listed in the program on lines 100-210. We found the '3' was slightly temperamental and needed two numbers to be checked for a proper JOYSTK(0) reading. This presents no problem and may merely be a glitch in our models of the CoCo.

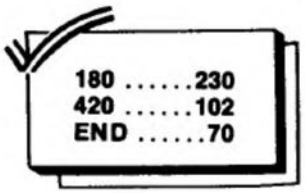
The program continually scans the joystick values. When it makes a match to the table listed, it responds with some music and circles the number pressed.

The second of this month's programs (Listing 2) is the number-matching exercise for preschoolers. A replica of the Book is drawn on the screen and a number is circled. The child presses the corresponding number on the pad. If correct, the number on the screen will be circled again in a different color. If incorrect, both the error and the correct answer will be shown.

We think the educational strength of this device is especially beneficial for preschoolers and physically handicapped students. Both of these populations often have motor dexterity problems and find it difficult to use the computer's keyboard.

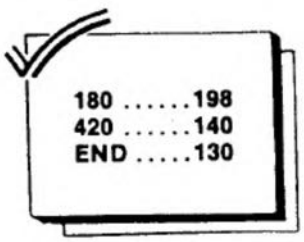
This concept can be extended to many types of educational games, activities and learning situations. Next month, we will present an educational program on a slightly higher grade level to further illustrate the use of this new tool.

In conclusion, we issue a challenge to the more adventurous. We, at Computer Island, have begun to work with the Tandy 1000. Since some CoCo owners are also Tandy 1000 owners, we decided to try the Electronic Book on it. Although not designed for it, we have found that the Electronic Book works quite well on the Tandy 1000. We will leave it to you to determine what the necessary joystick values are for that computer. We will include our findings in next month's article. □



Listing 1: ELECBOOK

```
1Ø REM"ELECTRONIC BOOK UTILITY"
2Ø REM"STEVE BLYN,COMPUTER ISLAN
D,NY,1985
3Ø CLEAR2ØØØ
4Ø GOSUB 28Ø
5Ø GOSUB 49Ø
6Ø N=Ø
7Ø X=A:Y=B
8Ø A=JOYSTK(Ø)
9Ø B=JOYSTK(1)
1ØØ IF A=63 AND B=Ø THEN N=1:CIR
CLE(47,16),16,8,.7:DB$=E1$
11Ø IF A=53 AND B=Ø THEN N=2:CIR
CLE(121,16),16,8,.7:DB$=E2$
12Ø IF A=41OR A=42 AND B=Ø THEN
N=3:CIRCLE(197,16),16,8,.7:DB$=E
3$
13Ø IF A=8 AND B=Ø THEN N=4:CIRC
LE(47,36),16,8,.7:DB$=E4$
14Ø IF A=19 AND B=Ø THEN N=5:CIR
CLE(119,36),16,8,.7:DB$=E5$
15Ø IF A=3Ø AND B=ØTHENN=6:CIRCL
E(197,36),16,8,.7:DB$=E6$
16Ø IF A=Ø AND B=63 THEN N=7:CIR
CLE(47,56),16,8,.7:DB$=E7$
17Ø IF A=Ø AND B=52 THEN N=8:CIR
CLE(12Ø,56),16,8,.7:DB$=E8$
18Ø IF A=Ø AND B=4Ø THEN N=9:CIR
CLE(197,56),16,8,.7:DB$=E9$
19Ø IF A=Ø AND B=7 THEN N=1Ø:CIR
CLE(48,76),16,8,.7:DB$=E1$+EØ$
2ØØ IF A=Ø AND B=18 THEN N=11:CI
RCLE(122,76),16,8,.7:DB$=E1$+E1$
21Ø IF A=Ø AND B=28 THEN N=12:CI
RCLE(198,76),16,8,.7:DB$=E1$+E2$
22Ø IF N<1 THEN 8Ø
23Ø COLOR5:LINE(16Ø,97)-(2ØØ,12Ø
),PSET,BF
24Ø DRAW"C7BM17Ø,1ØØ"+DB$
25Ø COLOR4:LINE(15Ø,12Ø)-(188,11
6),PSET,BF
26Ø PLAY"O4L8CDEFGEC"
27Ø GOTO 5Ø
28Ø REM"DRAWINGS FOR THE NUMBERS
 AND LETTERS NEEDED"
29Ø C$="BU4ER2FD2GL2HBG2BL4"
3ØØ E$="BER3U2NL2U2L4BG5BL2"
31Ø H$="BUU2NU2R4NU2D2BGBL9"
32Ø O$="BEHU2ER2FD2GL2BGBL6"
33Ø S$="BU2FR2EHL2HER2FBG4BL6"
34Ø U$="BUU3ER2FD3BGBL9"
35Ø Y$="BUE2NU2F2BGBL9"
36Ø EØ$=O$
37Ø E1$="BE2NU3DEBFBGBL9"
38Ø E2$="BENR3HER3U2L4BG5BL"
39Ø E3$="BENR3HENR2HER3BG5BL5"
4ØØ E4$="BENU4E3L4BG4BL2"
41Ø E5$="BER4U2L3HER3BG5BL5"
42Ø E6$="BU2FR2EU2NHGL2HER2BG5BL
4"
43Ø E7$="BUNR4UE3BG5BL4"
44Ø E8$="BER2EHEHL2GFNR2GFBGBL6"
45Ø E9$="BER2EHL2GNFU2ER2FBG4BL6
"
46Ø SP$="BE4BUBG5BL5":'*SPACER
47Ø RETURN
48Ø REM"DRAW THE SCREEN"
49Ø PCLS:PMODE3,1:SCREEN1,1:PCLS
5
5ØØ COLOR6:LINE(2Ø,Ø)-(235,92),P
SET,B
51Ø LINE(24,2)-(231,9Ø),PSET,B
52Ø PAINT(22,1),6,6
53Ø DRAW"A2S8C6BM5Ø,1Ø"+E1$
54Ø DRAW"BM125,1Ø"+E2$
55Ø DRAW"BM2ØØ,1Ø"+E3$
56Ø DRAW"BM5Ø,3Ø"+E4$:DRAW"BM125
,3Ø"+E5$:DRAW"BM2ØØ,3Ø"+E6$
57Ø DRAW"BM5Ø,5Ø"+E7$:DRAW"BM125
,5Ø"+E8$:DRAW"BM2ØØ,5Ø"+E9$
58Ø DRAW"BM45,7Ø"+E1$+EØ$
59Ø DRAW"BM12Ø,7Ø"+E1$+E1$:DRAW"
BM195,7Ø"+E1$+E2$
6ØØ DRAW"A2S8C7BM5Ø,1ØØ"+Y$+O$+U
$+SP$+C$+H$+O$+S$+E$+SP$
61Ø RETURN
```



Listing 2: NMBRMTCH

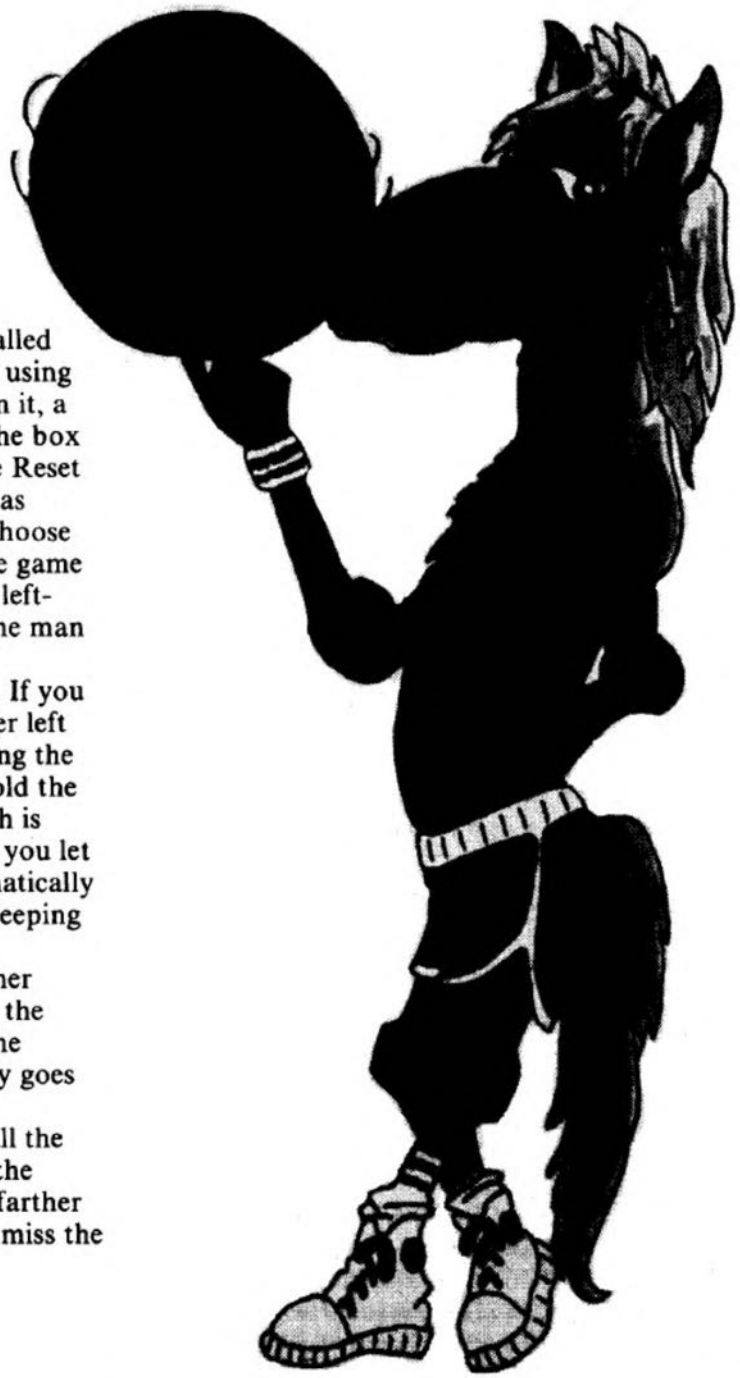
```
1Ø REM"ELECTRONIC BOOK-MATCHING
NUMBERS"
2Ø REM"STEVE BLYN,COMPUTER ISLAN
D,NY,1985
3Ø CLEAR2ØØØ:CLS 4
4Ø XY=RND(-TIMER)
5Ø GOSUB 27Ø
6Ø GOSUB 41Ø
7Ø N=Ø
8Ø X=A:Y=B
9Ø A=JOYSTK(Ø)
1ØØ B=JOYSTK(1)
```

```
11Ø IF A=63 AND B=Ø THEN N=1:CIR
CLE(47,16),16,7,.7
12Ø IF A=53 AND B=Ø THEN N=2:CIR
CLE(121,16),16,7,.7
13Ø IF A=41OR A=42 AND B=Ø THEN
N=3:CIRCLE(197,16),16,7,.7
14Ø IF A=8 AND B=Ø THEN N=4:CIRC
LE(47,36),16,7,.7
15Ø IF A=19 AND B=Ø THEN N=5:CIR
CLE(119,36),16,7,.7
16Ø IF A=3Ø AND B=ØTHENN=6:CIRCL
E(197,36),16,7,.7
17Ø IF A=Ø AND B=63 THEN N=7:CIR
CLE(47,56),16,7,.7
18Ø IF A=Ø AND B=52 THEN N=8:CIR
CLE(12Ø,56),16,7,.7
19Ø IF A=Ø AND B=4Ø THEN N=9:CIR
CLE(197,56),16,7,.7
2ØØ IF A=Ø AND B=7 THEN N=1Ø:CIR
CLE(48,76),16,7,.7
21Ø IF A=Ø AND B=18 THEN N=11:CI
RCLE(122,76),16,7,.7
22Ø IF A=Ø AND B=28 THEN N=12:CI
RCLE(198,76),16,7,.7
23Ø IF N<1 THEN 9Ø
24Ø IF N=G THEN PLAY"O4L8CDEFGFE
DC"
25Ø IF N<>G THEN PLAY"G"
26Ø GOTO 6Ø
27Ø REM"DRAWINGS FOR THE NUMBERS
 AND LETTERS NEEDED"
28Ø E$="BEHU2ER2FD2GL2BGBL6"
29Ø E1$="BE2NU3DEBFBGBL9"
3ØØ E2$="BENR3HER3U2L4BG5BL"
31Ø E3$="BENR3HENR2HER3BG5BL5"
32Ø E4$="BENU4E3L4BG4BL2"
33Ø E5$="BER4U2L3HER3BG5BL5"
34Ø E6$="BU2FR2EU2NHGL2HER2BG5BL
4"
35Ø E7$="BUNR4UE3BG5BL4"
36Ø E8$="BER2EHEHL2GFNR2GFBGBL6"
37Ø E9$="BER2EHL2GNFU2ER2FBG4BL6
"
38Ø SP$="BE4BUBG5BL5":'*SPACER
39Ø RETURN
4ØØ REM"DRAW THE SCREEN"
41Ø FOR T= 1 TO 2ØØØ: NEXT T:PCL
S:PMODE3,1:SCREEN1,1:PCLS5
42Ø COLOR6:LINE(2Ø,Ø)-(235,92),P
SET,B
43Ø LINE(24,2)-(231,9Ø),PSET,B
44Ø PAINT(22,1),6,6
45Ø DRAW"A2S8C6BM5Ø,1Ø"+E1$
46Ø DRAW"BM125,1Ø"+E2$
47Ø DRAW"BM2ØØ,1Ø"+E3$
48Ø DRAW"BM5Ø,3Ø"+E4$:DRAW"BM125
,3Ø"+E5$:DRAW"BM2ØØ,3Ø"+E6$
49Ø DRAW"BM5Ø,5Ø"+E7$:DRAW"BM125
,5Ø"+E8$:DRAW"BM2ØØ,5Ø"+E9$
5ØØ DRAW"BM45,7Ø"+E1$+EØ$
51Ø DRAW"BM12Ø,7Ø"+E1$+E1$:DRAW"
BM195,7Ø"+E1$+E2$
52Ø G=RND(12)
53Ø IF G=1 THEN CIRCLE(47,16),18
,8,.7:DB$=E1$
54Ø IF G=2 THEN CIRCLE(121,16),1
8,8,.7:DB$=E2$
55Ø IF G=3 THEN CIRCLE(197,16),1
8,8,.7:DB$=E3$
56Ø IF G=4 THEN CIRCLE(47,36),18
,8,.7:DB$=E4$
57Ø IF G=5 THEN CIRCLE(119,36),1
8,8,.7:DB$=E5$
58Ø IF G=6 THEN CIRCLE(197,36),1
8,8,.7:DB$=E6$
59Ø IF G=7 THEN CIRCLE(47,56),18
,.8,.7:DB$=E7$
6ØØ IF G=8 THEN CIRCLE(12Ø,56),1
8,8,.7:DB$=E8$
61Ø IF G=9 THEN CIRCLE(197,56),1
8,8,.7:DB$=E9$
62Ø IF G=1Ø THEN CIRCLE(48,76),1
8,8,.7:DB$=E1$+EØ$
63Ø IF G=11 THEN CIRCLE(122,76),
18,8,.7:DB$=E1$+E1$
64Ø IF G=12 THEN CIRCLE(198,76),
18,8,.7:DB$=E1$+E2$
65Ø PLAY"O3L8CEG"
66Ø DRAW"S24C7BM12Ø,12Ø"+DB$:S=8
67Ø RETURN
```

Test your basketball shooting skill, but unlike a day at the races . . .

If Your 'Horse' Comes In First, You Lose!

By Tim Jones

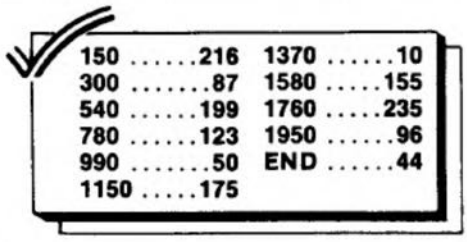


A re-creation of the popular basketball game called "Horse," this two-player game can be played using joysticks or the keyboard. When you first run it, a small box appears in the upper left-hand corner. If the box is red, just press any key. If the box is blue, press the Reset button and run the game again. After the program has finished drawing the graphics, you will be asked to choose either joystick or keyboard play. The controls for the game are pretty much the same, for example: Pressing the left-arrow key or moving the joystick to the left causes the man to move left.

The next prompt asks for beginner or expert play. If you choose beginner, there will be a rectangle in the upper left corner of the screen. This box is to aid you in guessing the correct strength of your shot. To shoot, press and hold the joystick button or the space bar until desired strength is reached; the longer you wait the higher he shoots. If you let the bar reach the top of the box, the man will automatically shoot the ball. If you choose expert play, there are beeping sounds to go by instead of the bar.

Once your shot has been released, the ball will either make it, miss or bounce off the goal. If you make it, the next player gets a chance to make the same shot. If he misses, he gets an 'H'. If he makes the shot, then play goes back to you.

Play continues in this manner until a player gets all the letters in the word "horse." Therefore, the object of the game is to make the more difficult shots by moving farther away from the goal, and hoping your opponent will miss the same shot.



150	216	1370	10
300	87	1580	155
540	199	1760	235
780	123	1950	96
990	50	END	44
1150	175		

The listing: HORSE

```
Ø '****************************
1 '*           HORSE            *
2 '*      BY TIM J. JONES       *
3 '*       P.O. BOX 7938        *
4 '*     CLINTON, LA 7Ø722      *
5 '****************************
1Ø CLEAR1ØØØ
2Ø PCLEAR4
3Ø PMODE4,1:PCLS:SCREEN1,1:FOR X
=1TO41 STEP2:LINE(X,Ø)-(X,4Ø),PS
ET:NEXTX
4Ø A$=INKEY$:IF A$="" THEN 4Ø
5Ø CLS3:O=67:PMODE3,1:PCLS:SCREE
NØ,Ø
6Ø READ IN:IF IN=999 THEN 9Ø
7Ø IF IN=888 THEN O=O+7:GOTO6Ø
8Ø O=O+1:PRINT@O,CHR$(IN+32);:GO
TO6Ø
9Ø PRINT@271,"by";
1ØØ PRINT@396,"tim"+CHR$(175)+"j
ones";
11Ø FOR T=ØTO31:PRINT@T,CHR$(137
+16);:NEXT
12Ø FOR T=31 TO 448 STEP32:PRINT
@T,CHR$(137+16);:NEXT
13Ø FOR T=479 TO 448 STEP-1:PRIN
T@T,CHR$(137+16);:NEXT
14Ø FOR T=448 TO Ø STEP-32:PRINT
@T,CHR$(137+16);:NEXT
15Ø PRINT@487,"COMPLETED JULY 19
84";
16Ø DIM A(2Ø),B(2Ø),M(6Ø),M2(6Ø)
,M3(6Ø),GL(9Ø),GB(9Ø),GH(9Ø),GT(
9Ø),U(6Ø),U2(6Ø),U3(6Ø)
17Ø GET(12Ø,131)-(128,138),A,G
18Ø LINE(Ø,191)-(255,191),PSET
19Ø S$(1)="D8U4L6D4U8"
2ØØ S$(2)=S$(1)+";BM+1Ø,+Ø;D8R6U
8L6R6"
21Ø S$(3)=S$(2)+";BM+4,+Ø;D8U4R4
F2D2U2H2REU2HL4R4;"
22Ø S$(4)=S$(3)+";BM+7,+Ø;BD8LR4
EU2HL3HU2ER4"
23Ø S$(5)=S$(4)+";BM+5,+Ø;D8R6L6
U4R4L4U4R6"
24Ø P$="U6R5FDGL2BM+8,-3;D6R4BM+
4,+ØU5ER2F2DL4R4D3BM+8,+Ø;U3L2H1
U2D2FR5EU2;BM+4,+Ø;R4L4D3R2L2D3R
4;BM+4,+Ø;U6R5FDGL2R2FD2"
25Ø DRAW"C3":LINE(Ø,Ø)-(255,Ø),P
SET
26Ø MS$="S4C2L3D5R2C3U4R4D3L2DR2
L2U3L3D5U5R3C4DC3D5C4L4D8R2U8R2D
8R2U8L4D9C3D5C2D2R4L2U3C3U4"
27Ø BS$="S4C3L3D5R2C2U4R4D3L2DR2
L2U3L3D5U5R3C4DC2D5C4L4D8R2U8R2D
8R2U8L4D9C2D5C3D2R4L2U3C2U4"
28Ø DRAW"BM12Ø,166;XMS$;"
29Ø DRAW"BM6Ø,166;XMS$;"
3ØØ DRAW"BM118,175;C3D2R8UL8"
31Ø DRAW"BM6Ø,176;C3M66,174U8D1Ø
L4"
32Ø DRAW"BM3Ø,166;XMS$;"
33Ø DRAW"BM28,176;C3M38,164R2M28
,177;"
34Ø GL$="S5C3U3ØL2D3ØL2U3ØH1ØUF1
2UH12UF14UH14C2U6D18U4L2C4L8"
35Ø DRAW"BM25Ø,19Ø;XGL$;"
36Ø FOR X=222 TO 23Ø STEP4
37Ø PSET(X,148,2)
38Ø NEXT X
39Ø Y=148
4ØØ FOR T=1TO3
41Ø PSET(226,Y,2)
42Ø FOR X=224 TO 228 STEP4
43Ø PSET(X,Y+1,2)
44Ø NEXT X
45Ø Y=Y+2:NEXTT
46Ø DRAW"C3"
47Ø Y=131
48Ø LINE(222,Y)-(226,Y),PSET
49Ø Y=Y+1
5ØØ LINE(22Ø,Y)-(228,Y+5),PSET,B
F
51Ø Y=Y+6
52Ø LINE(222,Y)-(226,Y),PSET
53Ø DRAW"C2":LINE(2,14)-(16,79),
PSET,B
54Ø DRAW"C4":LINE(16,15)-(2Ø,15)
,PSET
55Ø GET(22Ø,131)-(228,138),B,G
56Ø PUT(126,171)-(134,178),B,OR
57Ø PUT(62,159)-(7Ø,166),B,OR
58Ø PLAY"A"
59Ø GET(11Ø,164)-(14Ø,19Ø),M,G
6ØØ PLAY"A"
61Ø GET(54,152)-(76,19Ø),M2,G
62Ø GET(24,159)-(4Ø,19Ø),M3,G
63Ø DRAW"BM3Ø,166;XBS$;"
64Ø DRAW"BM28,176;C2M38,164R2M28
,177;"
65Ø PSET(32,166,3)
66Ø DRAW"BM12Ø,166;XBS$;"
67Ø DRAW"BM6Ø,166;XBS$;"
68Ø DRAW"BM118,175;C2D2R1ØUL1Ø"
69Ø DRAW"BM6Ø,176;C2M66,174U8HFD
1ØL4":PSET(122,166,3):PSET(62,16
6,3)
```

```
7ØØ PUT(22Ø,131)-(228,138),A,PSE
T
71Ø GET(11Ø,164)-(14Ø,19Ø),U,G
72Ø GET(54,152)-(76,19Ø),U2,G
73Ø GET(24,159)-(4Ø,19Ø),U3,G
74Ø GET(216,12Ø)-(255,191),GL,G
75Ø PUT(222,18Ø)-(23Ø,187),B,PSE
T
76Ø GET(216,12Ø)-(255,191),GB,G
77Ø PUT(222,18Ø)-(23Ø,187),A,PSE
T
78Ø PUT(222,143)-(23Ø,15Ø),B,OR
79Ø FOR X=222 TO 23Ø STEP4
8ØØ PSET(X,148,2)
81Ø NEXT X
82Ø FOR T=149 TO 151 STEP2
83Ø PSET(224,T,2):PSET(228,T,2)
84Ø NEXTT
85Ø PSET(226,15Ø,2)
86Ø GET(216,12Ø)-(255,191),GH,G
87Ø PUT(216,12Ø)-(255,191),GL,PS
ET
88Ø PUT(222,158)-(23Ø,165),B,PSE
T
89Ø LINE(22Ø,154)-(23Ø,152),PRES
ET,BF
9ØØ GET(216,12Ø)-(255,191),GT,G
91Ø LINE(54,152)-(74,19Ø),PRESET
,BF
92Ø LINE(24,16Ø)-(17Ø,19Ø),PRESE
T,BF
93Ø CLS:PRINT:PRINT"(J)OYSTICK O
R (K)EYBOARD"
94Ø A$=INKEY$:IF A$="" THEN 94Ø
95Ø IF A$="J" THEN OP=Ø:GOTO99Ø
96Ø IF A$="K" THEN OP=2:GOTO98Ø
97Ø GOTO94Ø
98Ø PLAY"T255CDEFAB":PRINT"LEFT
ARROW = MOVE LEFT":PRINT"RIGHT A
RROW = MOVE RIGHT":PRINT"SPACE B
AR = SHOOT":FOR O=1TO2ØØØ:NEXTO
99Ø CLS:PRINT:PRINT:PRINT"      1
 BEGINNER":PRINT:PRINT"      2 EX
PERT":PRINT:PRINT"       <1 OR 2>
"
1ØØØ A$=INKEY$:IF A$="" THEN 1ØØ
Ø
1Ø1Ø IF A$="1" THEN BR=1
1Ø2Ø IF A$="2" THEN BR=Ø:LINE(Ø,
2)-(2Ø,8Ø),PRESET,BF
1Ø3Ø IF A$<"1" OR A$>"2" THEN 1Ø
ØØ
1Ø4Ø PL=1:M=1:SC(1)=4:SC(2)=Ø:PL
AY"T255CDEFAB"
1Ø5Ø LINE(2Ø,2)-(1ØØ,6Ø),PRESET,
BF
1Ø6Ø DRAW"BM3Ø,1Ø;C3;XP$;BM+1Ø,-
6;GED6L2R4"
1Ø7Ø DRAW"BM3Ø,4Ø;C2;XP$;BM+9,-4
;UER3FD2M-5,+3;R6"
1Ø8Ø PUT(216,12Ø)-(255,191),GL,P
SET
1Ø9Ø SCREEN1,Ø
11ØØ X=11Ø:CH=2Ø
111Ø DRAW"C3"
112Ø PUT(216,12Ø)-(255,191),GL,P
SET
113Ø LINE(4,15)-(14,78),PRESET,B
F
114Ø G=11:Q=8Ø:IF PL=1 THEN PUT(
X,164)-(X+3Ø,19Ø),M,PSET
115Ø IF PL=2 THEN PUT(X,164)-(X+
3Ø,19Ø),U,PSET
116Ø IF OP=2 THEN 13ØØ
117Ø W=JOYSTK(Ø):E=JOYSTK(2)
118Ø IF M=Ø THEN 125Ø
119Ø IF PL=2 THEN 123Ø
12ØØ IF W=<1Ø THEN X=X-2:CH=CH+1
121Ø IF W=>5Ø THEN X=X+2:CH=CH-1
122Ø GOTO125Ø
123Ø IF E=<1Ø THEN X=X-2:CH=CH+1
124Ø IF E=>5Ø THEN X=X+2:CH=CH-1
125Ø PK=PEEK(6528Ø)
126Ø IF PK=124 OR PK=252 THEN 12
5Ø
127Ø IF PL=1 AND PK=126 OR PL=1
AND PK=254 THEN 136Ø
128Ø IF PL=2 AND PK=125 OR PL=2
AND PK=353 THEN 136Ø
129Ø GOTO 133Ø
13ØØ IF PEEK(343)=247 AND M=1 TH
EN X=X-2:CH=CH+1
131Ø IF PEEK(344)=247 AND M=1 TH
EN X=X+2
132Ø IF PEEK(345)=247 THEN 136Ø
133Ø IF X=>19Ø THEN X=19Ø
134Ø IF X=<46 THEN X=46
135Ø GOTO114Ø
136Ø PLAY"T255FFFFFP1FFFFFP1FFFF
FP1AAAAA":SP=X
137Ø IF OP=2 THEN 143Ø
138Ø PK=PEEK(6528Ø)
139Ø IF PK=124 OR PK=252 THEN 13
8Ø
14ØØ IF PL=1 AND PK=126 OR PL=1
AND PK=254 THEN 144Ø
141Ø IF PL=2 AND PK=125 OR PL=2
AND PK=253 THEN 144Ø
142Ø GOTO147Ø
143Ø IF PEEK(345)=247 THEN 144Ø
ELSE 147Ø
144Ø G=G+1:Q=Q-2:IF BR=1 THEN LI
NE(4,Q)-(14,Q-1),PSET,B ELSE PLA
Y"T255CF"
145Ø IF G=43 THEN 147Ø
146Ø GOTO 137Ø
```

```
147Ø X=X+14:Y=154:G=-G:ST=G
148Ø IF PL=2 THEN 152Ø
149Ø PUT(X-1Ø,152)-(X+12,19Ø),M2
,PSET
15ØØ PUT(X-1Ø,159)-(X+6,19Ø),M3,
PSET
151Ø GOTO 154Ø
152Ø PUT(X-1Ø,152)-(X+12,19Ø),U2
,PSET
153Ø PUT(X-1Ø,159)-(X+6,19Ø),U3,
PSET
154Ø X=X-2
155Ø IF X+12=>244 THEN 164Ø
156Ø G=G+6:X=X+12:Y=Y+G
157Ø IF Y=>218 THEN 164Ø
158Ø PUT(X,Y)-(X+8,Y+7),B,OR
159Ø PUT(X-12,Y-G)-(X-4,Y+7-G),A
,PSET
16ØØ IF X=>216 AND X=<23Ø AND Y=
<148 AND Y=>13Ø THEN 163Ø
161Ø IF G=>-ST+6 THEN 164Ø
162Ø GOTO 155Ø
163Ø GOTO 172Ø
164Ø PLAY"T255AAAAAP1FFFFF":LINE
(Ø,156)-(24Ø,19Ø),PRESET,BF
165Ø LINE(23Ø,1)-(255,12Ø),PRESE
T,BF
166Ø DRAW"C4":LINE(Ø,191)-(255,1
91),PSET
167Ø IF M=Ø THEN SC(PL)=SC(PL)+1
168Ø M=1:PL=PL+1:IF PL=3 THEN PL
=1
169Ø DRAW"BM36,16;XS$(SC(1));BM3
6,46;XS$(SC(2));"
17ØØ IF SC(1)=5OR SC(2)=5THEN 18
4Ø
171Ø GOTO 1Ø9Ø
172Ø IF RND(1ØØ)=<CH THEN 189Ø
173Ø PUT(216,12Ø)-(255,191),GH,P
SET
174Ø PUT(216,12Ø)-(255,191),GT,P
SET
175Ø PUT(216,12Ø)-(255,191),GB,P
SET
176Ø PLAY"T255AABBCCAAFFAACCCC"
177Ø LINE(Ø,156)-(24Ø,19Ø),PRESE
T,BF
178Ø PL=PL+1:IF PL=3 THEN PL=1
179Ø IF M=Ø THEN M=1:X=11Ø:CH=2Ø
:GOTO182Ø
18ØØ IF M=1 THEN M=Ø
181Ø X=SP
182Ø LINE(4,15)-(14,78),PRESET,B
F
183Ø GOTO111Ø
184Ø PLAY"T4O3L4EL4.EL8EL4D#EFL4
.FL8EL2FL4FL4.FL8FL4EFGL4.GL8F#L
2GL4AO4CO3BAGECDEGL8FL4EDL1C"
185Ø FOR O=1TO15ØØ:NEXTO
186Ø CLS:PRINT:PRINT"PRESS <ENTE
R> TO PLAY AGAIN":PLAY"T25P8"
187Ø A$=INKEY$:IF A$<>CHR$(13) T
HEN 187Ø
188Ø GOTO1Ø4Ø
189Ø PUT(216,12Ø)-(255,191),GL,P
SET
19ØØ PUT(222,139)-(23Ø,146),B,PS
ET
191Ø PUT(222,139)-(23Ø,146),A,PS
ET
192Ø PUT(216,139)-(224,146),B,PS
ET
193Ø FOR O=1TO6Ø:NEXTO
194Ø PUT(216,139)-(224,146),A,PS
ET
195Ø PUT(21Ø,142)-(218,149),B,PS
ET
196Ø FOR O=1TO3Ø:NEXTO
197Ø PUT(21Ø,142)-(218,149),A,PS
ET
198Ø PUT(2Ø8,16Ø)-(216,167),B,PS
ET
199Ø FOR O=1TO6Ø:NEXTO
2ØØØ PUT(2Ø8,16Ø)-(216,167),A,PS
ET
2Ø1Ø PUT(2Ø8,182)-(216,19Ø),B,PS
ET
2Ø2Ø FOR O=1TO3Ø:NEXTO
2Ø3Ø GOTO 164Ø
2Ø4Ø DATA 133,143,143,138,143,13
7,131,131,134,143,129,131,131,13
4,143,137,131,131,134,143,129,13
1,131,131,888
2Ø5Ø DATA 143,132,14Ø,14Ø,136,14
3,133,143,143,138,143,132,14Ø,14
Ø,137,143,134,14Ø,14Ø,141,143,13
2,14Ø,14Ø,143,888
2Ø6Ø DATA 143,133,143,143,138,14
3,133,143,143,138,143,133,143,13
Ø,141,143,141,143,143,138,143,13
3,143,143,143,888
2Ø7Ø DATA 143,135,143,143,139,14
3,139,131,131,135,143,135,143,14
3,131,143,139,131,131,135,143,13
1,131,131,131
2Ø8Ø DATA 999
```

Graphics Quickies

IT'S CHARLIE THE CANADIAN DOG, EH!

16K ECB

by Ray Larabie

Charlie's picture can be dumped to a printer for colour or black-and-white printouts using a screen dump program.

The Listing: CHARLIE

```
10 'CHARLIE (C) COPYRIGHT 1985 R
AY LARABIE ALL RIGHTS RESERVED
20 CLS:PRINT"CHARLIE (C) 1985 RA
Y LARABIE":PRINT"  ALL RIGHTS RE
SERVED"
30 INPUT"BLACK & WHITE OR COLOR
(B/C)";A$
40 IF A$="C" THEN PMODE3,1:SCREE
N1,0:PCLS2:GOTO70
50 PMODE4,1:SCREEN1,1:PCLS1:COLO
R0,1
60 'left ear
70 DRAW"BM111,24;M98,8;M89,8;M79
,11;M72,19;M68,36;M65,71;M56,87;
M49,96;M43,102;M28,103;M25,107;M
24,119;M27,128;M36,140;M40,145;M
56,152;M72,155;M87,155;M95,149"
80 'right ear
90 DRAW"BM159,24;M164,12;M172,8;
M175,9;M180,16;M182,24;M180,32;M
177,44;M180,67;M191,86;M208,95;M
220,90;M228,95;M229,99;M227,111;
M222,124;M208,139;M188,151;M176,
153;M165,148"
100 'brow
110 DRAW"BM111,24;M110,15;M125,2
0;M140,19;M144,11;M143,23;M156,1
1;M159,13;M159,24"
120 'snout
130 DRAW"BM122,77;M107,80;M98,86
;M91,92;M83,100;M78,111;M76,124;
M81,117;M83,132;M89,143;M95,149;
M99,141;M94,142;M112,157;M131,16
3;M128,156;M132,152;M135,158;M13
2,168;M142,163;M156,145;M158,157
;M165,145;M167,131;M166,115;M174
,120;M172,104;M164,91
140 'shoulders & eyelashes
150 DRAW"M157,83;M142,79;BM40,14
5;M22,160;M30,158;M0,188;BM193,1
48;M225,164;M216,168;M255,187;BM
111,55;M152,49"
160 'eyes
170 CIRCLE(125,53),9,,2.2:PAINT(
125,49),1,4
180 CIRCLE(141,52),9,,2.2:PAINT(
141,47),1,4
190 CIRCLE(131,54),4:CIRCLE(134,
54),4
200 'mouth
210 CIRCLE(130,113),15,,1.5,0,.5
0
220 'nose
230 CIRCLE(136,90),15,,.3:CIRCLE
(130,92),25,,.3:PAINT(130,95),,0
240 IF A$="C" THEN PAINT(0,0),3,
4 ELSE PAINT(0,0),0,0
250 'end
260 FORX=255TO1STEP-1:Y=Y+1:POKE
140,X:EXEC43345:POKE140,Y:EXEC43
345:NEXTX:Y=0:GOTO260
```

READ MY LIPS!

16K ECB

by Bill Bernico

The following program is a graphics display of a fellow named "KoKo" (sound familiar?) who will "spit out" sentences from the usre's input information. The fascinating part of the program is the way the information is presented.

For ambitious programmers who would like to sustitute different sentences, the programming techniques used are listed below.

Line	Description
80-150	Accepts user input and assigns to variables
170-200	Draws KoKo
210	Allows 20 spaces before the printing begins
220	Holds the framework of the sentence and variables from the user input
240	Creates the sound and prints at 256, the sentences created in Line 220

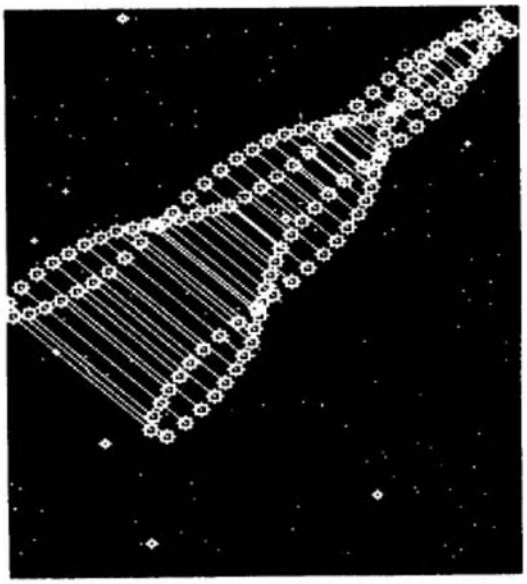
The Listing: SPEAK2ME

```
10 'SPEAK TO ME
20 'BY BILL BERNICO
30 '708 MICHIGAN AVE.
40 'SHEBOYGAN, WI 53081
50 '(414) 459-7350
60 '
70 CLEAR 1000
80 CLS:PRINT"WHAT IS YOUR NAME":
INPUT N$
90 CLS
100 CLS:PRINT"HOW OLD ARE YOU":I
NPUT X$
110 CLS:PRINT"ARE YOU A BOY OR G
IRL":INPUT"(B/G)";BG$
120 IF BG$="B"THEN BG$="BOY"ELSE
IF BG$="G"THEN BG$="GIRL"ELSE BG
$="PERSON"
130 CLS:PRINT"WHAT STREET DO YOU
 LIVE ON":INPUT Z$
140 CLS:PRINT"WHAT CITY DO YOU L
IVE IN":INPUT CT$
150 CLS:PRINT"WHAT STATE IS THAT
 IN":INPUT ST$
160 B$=CHR$(128):Y$=CHR$(159):E$
=CHR$(157)
170 CLS:PRINT@51,STRING$(8,128):
PRINT@82,STRING$(10,128):PRINT@1
13,STRING$(12,128):PRINT@146,Y$;
Y$;Y$;STRING$(8,128)
180 PRINT@178,;Y$;E$;Y$B$;B$;Y$;
STRING$(5,128):PRINT@209,STRING$
(8,159);STRING$(4,128)
190 PRINT@242,STRING$(10,159):PR
INT@275,STRING$(9,159):PRINT@306
,STRING$(9,159):PRINT@340,STRING
$(7,159):PRINT@372,STRING$(7,159
)
200 PRINT@402,STRING$(11,255):PR
INT@433,STRING$(13,175):PRINT@46
5,STRING$(13,255):PRINT@496,STRI
NG$(15,175);
210 FOR U=1 TO 20:Q$=Q$+CHR$(143
):NEXT U
220 S$="HELLO "+N$+", MY NAME IS
 KOKO. I'LL BE VISITING "+Z$+" S
TREET WHEN I'M IN "+CT$+" NEXT W
EEK. I'M LOOKING FOR A "+BG$+" A
BOUT "+X$+" YEARS OLD TO HELP ME
 FIND THE CAPITAL OF "+ST$+"."
230 L$=Q$+S$+Q$
240 FOR P=1 TO LEN(L$)-19:SOUND
RND(9)+180,2:PRINT@256,MID$(L$,P
,19);:NEXT P
250 PRINT@481,"aNOTHER OR eND";
260 W$=INKEY$:IF W$="A"THEN 10 E
LSE IF W$="E"THEN 270 ELSE 260
270 CLS:PRINT"BYE, ";N$
280 PRINT"SEE YA LATER
```

SPACE WAVES

16K ECB

by Joey Goodson



Rcircsin draws sine wave designs on an outerspace background and plays a short sound effect. The design is erased and then starts over with a new picture.

For quicker drawings, POKE65495,0.

The Listing: RCIRCSIN

```
  1 '*RCIRCSIN* - BY JOEY GOODSON
  2 A=RND(-TIMER)
  4 G=10:R=3
```

DRAWING THE LINE

4K

by Jimmy McGill

The following is a Lo-Res line drawing program. Simply use the arrow keys to move up, down, left and right; use the number keys '1' through '8' to select the desired colour ('0' will switch to black).

The Listing: LINEDRAW

```
10 CLS(0)
15 A=31
17 B=15
20 A$=INKEY$
30 IF A$=""THEN GOTO 20
50 REM DIRECTIONS
51 IF A$=CHR$(94)THEN B=B-1:IF B
<0 THEN B=0
52 IF A$=CHR$(10)THEN B=B+1:IF B
>31 THEN B=31
53 IF A$=CHR$(8)THEN A=A-1:IF A<
0 THEN A=0
54 IF A$=CHR$(9)THEN A=A+1:IF A>
63 THEN A=63
60 REM COLORS
61 IF A$="1"THEN C=1
62 IF A$="2"THEN C=2
63 IF A$="3"THEN C=3
64 IF A$="4"THEN C=4
65 IF A$="5"THEN C=5
66 IF A$="6"THEN C=6
67 IF A$="7"THEN C=7
68 IF A$="8"THEN C=8
69 IF A$="0"THEN GOSUB 100
70 SET(A,B,C)
80 GOTO 20
100 RESET(A,B):GOTO 20
```

TITLE PAGE POWER

16K ECB

by Bill Bernico

One thing every program needs is an eye-catching title page. Kromico uses the DRAW and PAINT method that puts the title on the screen, paints the letters, switches to an alternate screen and starts over with another colour, giving you a possibility of 24 different screen presentations. Use the power of your CoCo to spruce up those title pages!

The Listing: KROMICO

```
10 'KROMICO
20 'EXAMPLES OF GRAPHIC TITLES
30 'BY BILL BERNICO
40 '708 MICHIGAN AVE.
50 'SHEBOYGAN, WI 53081
60 '(414) 459-7350
70 '
80 PMODE3,1:SCREEN1,1:PCLS
90 X=1 'START PAINTING WITH COLO
R #1
100 'LINES 110-240 DRAW AND PAIN
T "KROMICO" IN FAT LETTERS
110 DRAW"BM2,1D70R12U25F25R214U1
0L209H25E35L16G25U25L11" 'K
120 PAINT(6,7),X,4
130 DRAW"BM40,57U44R25F6D14G6L10
F18L12H10D10L8BU38BR8R8F4D6G4L8U
14" 'R
140 PAINT(44,55),X,4
150 DRAW"BM78,52U32E6R20F6D32G6L
20H6BR10BU2UF3R8E3U25H3L9G3D25"
'O
160 PAINT(80,52),X,4
170 DRAW"BM118,57U42R10F8E8R10D4
2L10U26G8H8D26L10" 'M
180 PAINT(120,55),X,4
190 DRAW"BM163,57U42R10D42L10" '
I
200 PAINT(165,55),X,4
210 DRAW"BM180,46U23E7R20F7D6L8U
3H3L11G3D22F3R11E3U3R8D6G6L21H6U
4" 'C
220 PAINT(182,44),X,4
230 DRAW"BM221,50U28E6R20F6D28G6
L20H6BR10BU2UF3R8E3U21H3L9G3D21"
'O
240 PAINT(224,52),X,4
250 'LINES 260-330 DRAW "SOFTWAR
E" IN SMALLER LETTERS (NO PAINTI
NG)
260 DRAW"BM75,80U2H2L4G2D4F2R4F2
D4G2L4H2U2" 'S
270 DRAW"BR14D2F2R4E2U12H2L4G2D1
1" 'O
280 DRAW"BR16D3U9NR6U7R9" 'F
290 DRAW"BR7R11L6D16U16BR12" 'T
300 DRAW"D12F4E4NU6F4E4U12BD16BR
7" 'W
310 DRAW"U12E5F6D5NL9D6" 'A
320 DRAW"BR6U16R6F2D4G2L6R2F6D2"
'R
330 DRAW"BR6NR9U8NR6U8R9" 'E
340 GOSUB440 'TIME DELAY
350 'LINES 360-400 TRY DIFFERENT
SCREEN STYLES
360 PMODE4,1:SCREEN1,0:GOSUB440
370 PMODE1,1:SCREEN1,1:GOSUB440
380 PMODE4,1:SCREEN1,1:GOSUB440
390 PMODE3,1:SCREEN1,0:GOSUB440
400 PMODE1,1:SCREEN1,0:GOSUB440
410 X=X+1 'AFTER ALL 6 SCREEN ST
YLES ARE TRIED, PAINT WITH A DIF
FERENT COLOR
420 IF X>4THEN 80 'REPEAT AFTER
ALL FOUR COLORS ARE TRIED
430 PMODE3,1:SCREEN1,1:PCLS:GOTO
110 'ERASE SCREEN AND START OVER
440 FORT=1TO1500:NEXTT:RETURN
```

```
5 PMODE4,1:PCLS:SCREEN1,1
6 FORW=1TO150:PSET(RND(252),RND(
191)):NEXTW:FORW=1TO10:CIRCLE(RN
D(255),RND(191)),RND(2):NEXTW
10 IFB=0 THEN X=X+3
11 IFB=1 THEN X=X-3
12 IFBB=0 THEN XX=XX+3
13 IFBB=1 THENXX=XX-3
14 G=G+GG
20 Y=G+10*(SIN(X))
30 LINE(X,Y)-(Y,XX),PSET
31 CIRCLE(X,Y),R:CIRCLE(Y,XX),R
32 IFX=255 THENB=1
33 IFX=0 THEN B=0
34 IFXX=192 THEN BB=1:GOSUB100:P
CLS:R=RND(10):FORW=1TO150:PSET(R
ND(255),RND(191)):NEXTW:FORW=1TO
10:CIRCLE(RND(255),RND(191)),RND
(2):NEXTW
36 IFXX=0 THEN BB=0
37 IFG=248 THEN GG=-2
38 IFG=10 THEN GG=2
50 GOTO10
100 PLAY"V1"
101 J=RND(5):J$="O"+STR$(J):PLAY
J$
110 FORJ=1TO15
120 PLAY"T255V+ABGABG":NEXTJ
130 FORJ=1TO15
140 PLAY"T255V-ABCDEFGABCDEFG":N
EXTJ
150 FORJ=1TO460:NEXTJ
160 J=RND(4):ONJ GOSUB 200,250,3
00,350
161 RETURN
200 FORJY=0TO191:LINE(0,JY)-(255
,JY),PRESET:NEXTJY:RETURN
250 FORJY=191TO0STEP-1:LINE(0,JY
)-(255,JY),PRESET:NEXTJY:RETURN
300 FORJX=0TO255:LINE(JX,0)-(JX,
191),PRESET:NEXTJX:RETURN
350 FORJX=255TO0STEP-1:LINE(JX,0
)-(JX,191),PRESET:NEXTJX:RETURN
```

SPEED OPTIMIZING

32K ECB

by Doug Farrell

Ball presents a stunning, visual graphics effect. Before loading and running Ball, enter the following pokes:

```
POKE28160,0:POKE25,110:NEW
```

The Listing: BALL

```
 10 CLS6:PRINT@224,"CAN THE COMPU
 TER WORK AT HIGH   SPEED?     (Y
 /N)"
 20 A$=INKEY$:IFA$="Y"THENPOKE654
 95,0:WA=40:GOTO30ELSEIFA$="N"THE
NPOKE65494,0:WA=20:GOTO30:ELSEIF
A$=""THEN20ELSEGOTO20
30 IN=1:S=1.33333333
40 R=-.390625:Z=400:Z1=3600
50 CLS4:PRINT@224,STRING$(32,32)
;
60 PRINT@192,STRING$(32,153);
70 PRINT@256,STRING$(32,153);
80 A$=" 1YOU WON'T BELIEVE THIS!
!! 3PRETTY INCREDIBLE I MIGHT AD
D 50K, OK, I'M WORKING ON IT!!!
7ALLRIGHT, GET READY, HERE IT IS
 9"
90 FORP=1TO13STEP4
100 PMODE3,P:PCLS3:COLOR1,1
110 M1=INSTR(A$,STR$(IN))+2
120 PRINT@224,MID$(A$,M1,INSTR(M
1,A$,STR$(IN+2))-M1)
130 FORL=0TO256STEP64:LINE(L,191
)-(128,96),PSET:NEXT
140 FORL=112TO189STEP8
150 L1=L+IN:L2=L1-96:I2=IN/40:L2
=L1-96
170 LINE(257-L1*S,L1)-(L1*S,L1),
PSET:LINE(257-L1*S,L1)-(257-L1*S
,L1-26+L2*R),PSET:LINE-(0,L1-26+
L2*R),PSET:LINE(L1*S,L1)-(L1*S,L
1-26+L2*R),PSET:LINE-(255,L1-26+
L2*R),PSET
180 NEXT
190 COLOR1,1
200 LINE(0,80)-(255,80),PSET
210 PMODE0,P:PCLS0:PMODE3,P:PAIN
T(10,78),1,1
220 COLOR1,1
230 LINE(0,127)-(128,70),PSET:LI
NE-(255,127),PSET:LINE(0,80)-(25
5,80),PSET
240 FORX=58TO178STEP4
250 Y=SQR(Z*(1-(X-118)^2/Z1))
260 LINE(X-2,159-Y)-(X-2,159+Y),
PSET
270 NEXT
280 CIRCLE(128,96),60,2:PAINT(12
8,38),2,2:CIRCLE(128,96),60,1
290 FORA=.2TO1STEP.2
300 CIRCLE(128,96),60,1,A-I2,0,.
53
310 NEXT
320 FORA=0TO.8STEP.2
330 CIRCLE(128,96),60,1,A+I2,.5,
1
340 NEXTA
350 IN=IN+2
360 NEXTP
370 P=1
380 PMODE4,P:SCREEN1,1
390 FORW=0TO WA:NEXT
400 P=P+4:IFP>13THENP=1
410 GOTO380
```

WHICH WAY DID THAT MANGY MONGREL GO ?

by Brad Nation

Doghouse is a short program to challenge the visual reflexes of everyone from toddlers to adults. The 2,087-byte Color BASIC and machine language program is simple. A dog runs in and out of three doghouses and after it stops you are asked which house you think he is in. After each response the dog's speed is adjusted by approximately 30 percent, faster if your answer was right and slower if it was wrong.

Due to the machine language subroutine, the speed of the dog can be varied from as fast as 0.03 seconds to cross the screen from doghouse #1 to doghouse #3, to as slow as 1.9 seconds to cross the screen.

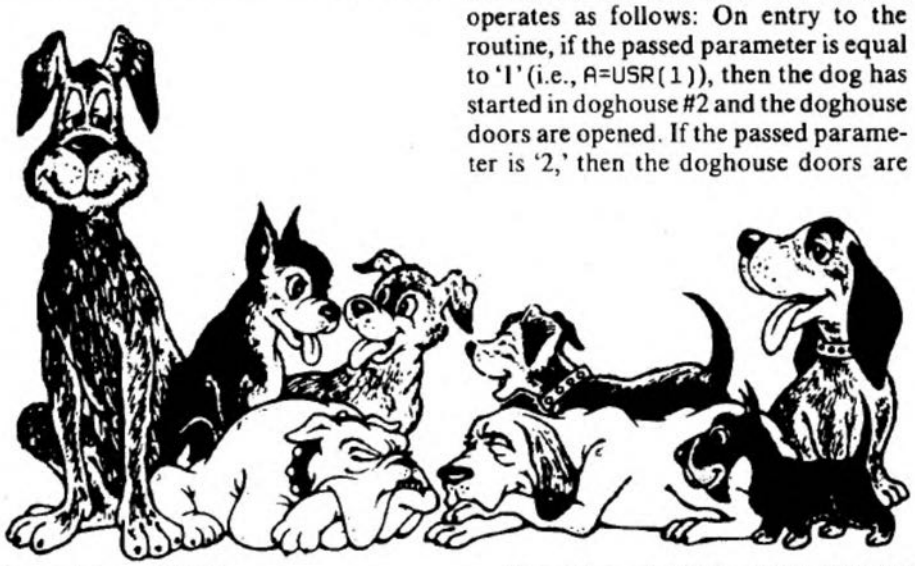
Doghouse was written for any Color Computer, 4K or more, Extended or non-Extended BASIC. Please note, however, that users with Color BASIC should replace Line 2 with 2 POKE 275,ML/256:POKE 276,ML-PEEK (275)*256. This is because the DEFUSR command is not supported by Color BASIC, so the machine language routine's execution address must be poked into locations 275 and 276. As an alternative, users of either system can replace Line 2 with 2 IF(PEEK(32768)=69 AND PEEK(32769)=88) THEN DEFUSR= ML ELSE POKE 275,ML/256: POKE 276, ML-PEEK(275)*256 which allows the program to determine if Extended BASIC is present or not. The program will then be able to run on any system.

The 321-byte machine language routine is position-independent so users with 32K or more may want to put the ML routine into higher memory by changing the CLEAR statement in Line 1 to CLEAR 200,32446, and also the value for ML in Line 1.

Also, 4K BASIC users must change the CLEAR statement to CLEAR 200,3774 and change the value of ML to ML=3775. This is due to the smaller memory size.

Type in *Doghouse* and run the program. Operating instructions are provided on-screen. Pressing 'S' when "PRESS 4 TO GO" is displayed will show the speed the dog will run during the next round (100 = fastest and 0 = slowest).

The assembly listing for the ML routine is not included. The routine operates as follows: On entry to the routine, if the passed parameter is equal to '1' (i.e., A=USR(1)), then the dog has started in doghouse #2 and the doghouse doors are opened. If the passed parameter is '2,' then the doghouse doors are closed so the dog can't be seen; if the passed parameter is '3,' then the doors are opened to show the dog. If the passed parameter is any other value (in Line 7 NP equals 1317, 1327 or 1337), then the dog runs to the new location.



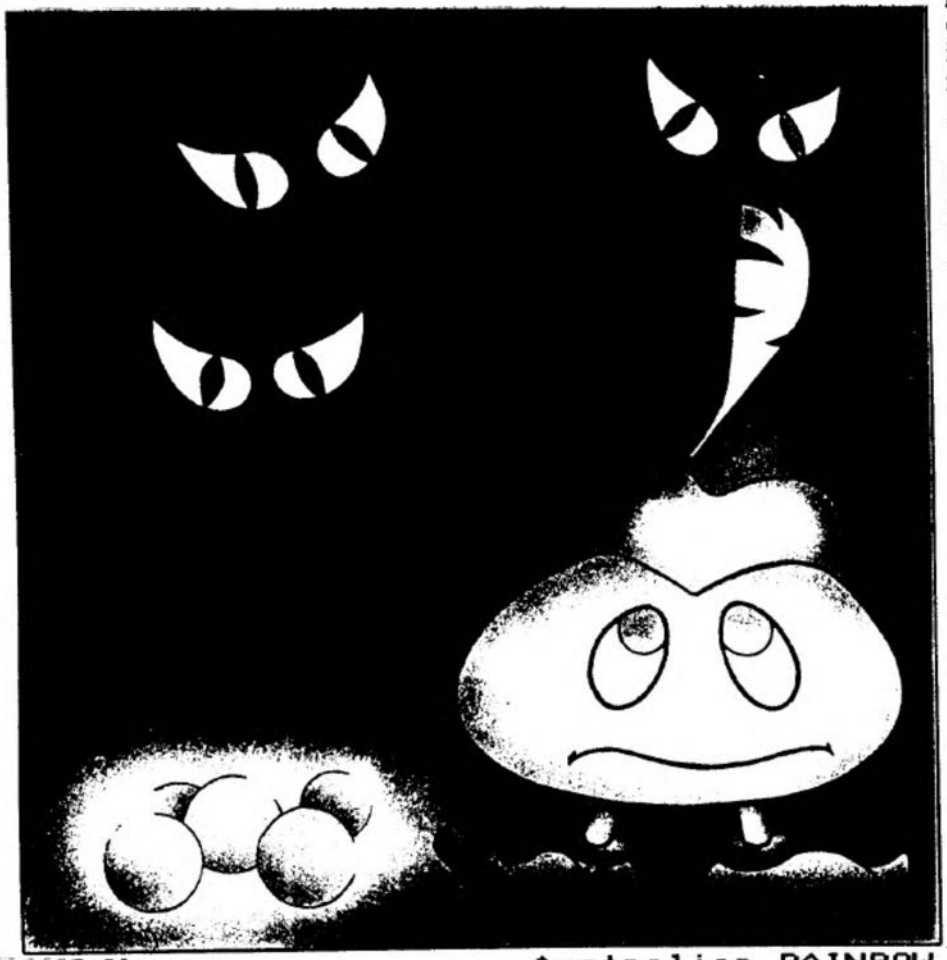
The listing: DOGHOUSE

```
1 CLS3:PRINT@137," doghouse game
 ";:CLEAR 200,16062:PRINT@448:PR
INT:ML=16063
2 DEFUSR=ML' EXTENDED ONLY
3 FOR I=ML TO ML+320:READ J:POKE
 I,J:NEXT I:A=USR(1):R$=" GOOD,
YOU WERE RIGHT! ":C$=CHR$(175):F
OR I=1 TO 5:C$=C$+C$:NEXTI:S=25
4 PRINT@100,C$;:PRINT@104," PRES
S '4' TO GO ";:A$=INKEY$
5 A$=INKEY$:IF A$="S" THEN PRINT
@100,C$;:PRINT@106," speed =";IN
T(100-S/1.27);:FORI=1TO900:NEXTI
:GOTO4 ELSEIF A$="4" THEN PRINT@
100,C$;:PRINT@107,"GET READY";:
ELSE A=RND(6):GOTO 5
6 FOR I=1034 TO 1449 STEP 32:J=R
ND(8)*16+127:POKE I,J:POKE I+10,
J:SOUND(255-(I-1035)/2),1:NEXTI:
POKE ML+232,S:POKE ML+234,S:FOR
I=1034 TO 1449 STEP 32:POKE I,17
5:POKE I+10,175:NEXTI:PRINT@100,
C$;:FOR I=1 TO 15+RND(15)
7 NP=RND(3)*10+1307:IF NP=OP THE
N 7 ELSE A=USR(NP):OP=NP:NEXT I:
A=USR(2):PRINT@103," WHERE IS TH
E DOG?";:A$=INKEY$
8 A$=INKEY$:G=VAL(A$):IF G<1 OR
G>3 THEN 8 ELSE A=USR(3):S=PEEK(
ML+232):G=G*10+1307:PRINT@100,C$
;:IF G<>NP THEN 10
9 PRINT@101,;:FORI=1 TO LEN(R$):
PRINT MID$(R$,I,1);:IF MID$(R$,I
```

```
,1)=" " THEN FOR J=1 TO 50:NEXTJ
,I:FORJ=1TO500:NEXTJ:S=S/1.3:GOT
O 4 ELSE SOUND(10+10*I),1:NEXTI
10 PRINT#101," SORRY, YOU WERE W
RONG ";:FOR I=1 TO 11:SOUND 50,1
:SOUND 75,1:NEXT I:S=S*1.3+1:IF
S<127 THEN 4 ELSE S=127:GOTO 4
11 DATA 189,179,237,193,1,16,39,
0,30,193,2,39,79,193,3,39,67,237
,141,0,233,236,141,0,227,16,163,
141,0,224,39,119,16,44,0,116,22,
0,147,95,231,141,0,212,142,5,47,
175,141,0,201,175,141,0,199,134,
49,183,4
12 DATA 197,134,50,183,4,207,134
,51,183,4,217,142,4,227,49,141,0
,206,141,46,142,5,1,141,41,230,1
41,0,168,39,105,32,70,51,141,0,2
03,32,4,51,141,0,208,142,5,33,14
1,16,142,5,65,141,11,142,5,97,14
1,6,142
13 DATA 5,129,141,1,57,31,50,166
,160,39,22,167,132,167,137,0,10,
167,137,0,20,48,1,32,238,166,160
,39,4,167,128,32,248,57,198,1,23
1,141,0,98,106,141,0,91,141,190,
174,141,0,84,48,30,49,141,0,83,1
41,68,48
14 DATA 136,30,141,218,48,136,27
,32,29,95,231,141,0,65,108,141,0
,58,141,157,174,141,0,51,49,141,
0,65,141,37,48,136,27,141,187,48
,136,28,16,142,202,202,16,175,13
2,48,3,16,142,197,197,16,175,132
15 DATA 134,25,198,25,90,44,253,
74,44,248,22,255,33,141,153,48,1
36,29,141,148,57,5,57,5,57,0,204
,198,198,0,195,194,193,0,128,128
,128,193,0,201,201,204,0,194,193
,195,0,194,128,128,128,0,172,168
,179,164,172,0,168
16 DATA 179,183,191,191,191,187,
179,164,0,191,255,255,255,255,25
5,255,255,191,175,0,191,207,207,
207,207,207,207,207,191,175,0
```


BROTAN the BLUE

by Alan Saporta



Just a few minutes ago, you were the high apprentice of Zygon, a powerful wizard who dwells in the hills of a countryside. It was a routine magic-creating day until the wizard asked you to hand him the maroon potion, but accidentally, you gave him the azure potion. The mixing of his new potion with the wrong ingredient (he hadn't noticed your foolish mistake) resulted in an explosion and destruction of part of his laboratory.

Even though he was close to the blast, he did survive and, in his rage, turned you into a Brotan. Brotans are short, blue creatures (no, not Smurfs! Brotans are shorter) with two green eyes and one mouth. Their only purpose in life is to eat yellow energy dots and (something that goes without saying) avoid red Gremlins who eat anything, including yellow energy dots, themselves and blue Brotans.

If you attain an energy score of 15,000, Zygon will return you to your normal state. But, until then, he has placed you in a strange box where two things are constantly produced: yellow energy dots and red Gremlins.

How to Play

This is a game for those with 32K Extended Color BASIC. You use three of the arrow keys to move the Brotan to the right, left and down. Once you go down an opening, you can't go back up. Because PEEKs are used to poll the keyboard, there is no need to continuously hit the keys; just hold down the arrow key of your choice.

After the title page is displayed, press any key (except BREAK) and the screen ("strange box") will be drawn. The strange box has six levels. At the beginning, the first two levels have three openings, the next three have two openings and the exit level has one.

The Brotan begins at the topmost, inner level at the left. To cross an opening just continue in one direction and a bridge will be formed. If you want to descend an opening, place yourself on top of it (lean a bit to the left) and press the down-arrow key. The bridge will be broken (if there is one) and a blue "slide" will form on which the Brotan will descend.

The energy dots are the yellow plus signs that line the top of each level. They are placed at a random distance from one another and can be worth zero, four, eight or 12 points. That, too, is random. When you score, the screen flashes and sounds, and four points are added to your energy score. If an energy dot is worth eight points the screen will flash twice and eight points are added, etc. Some dots are what I call "duds" — they are worthless. If you encounter one, you will simply consume it with no ill effects.

In the middle of the box is a giant red energy dot. If eaten, the screen flashes and 50 points are added. There is one per screen.

When you advance a skill rank, one of the openings on one of the levels will be blocked, preventing your escape to a lower level and increasing your chances of losing one "man" (you have eight). The skill ranks are as follows: Completing the first screen places you on the first rank, then every 250th point you will advance to the next rank (meaning more openings will be blocked if and when you reach the next screen). You can jump two or three skill ranks in one screen (just count how many openings have been blocked when the next screen is drawn). When you reach the seventh skill rank everything stays the same. There will only be one opening per level through which to escape.

For your convenience, there is a pause feature for those who get that annoying phone call right before they reach the 15,000 mark. By pressing any key (except BREAK) all action will stop. When you lose a man your score and the amount of men you have left are updated.

If your computer can't take the "speed-up" POKE 65495,0, then delete it from lines 10 and 2011. Also, if you reach 15,000 you are given the option to continue (just in case you want to know how high a score you can get). When you reach 2,000 points you get an extra man and at 5,000 you get two more. You also get points when you descend openings. One limitation, though: You can't have more than nine men at once.

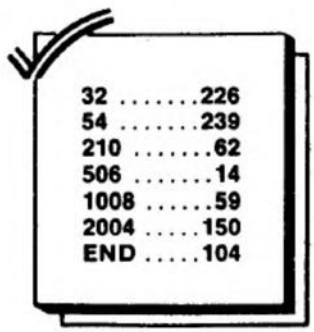
About Those Gremlins

They pop out everywhere (the amount and where they pop out is randomly determined). If one ever blocks your only means of reaching the next level, you can kill him by hitting him, but it will cost you at least one man; many will get two and some can never be killed, making death inevitable. Most *can* be killed, but that's a chance you'll have to take.

Hints

Here are a few hints that will help: 1) The leftmost row of energy dots is duds; 2) Even though the red Gremlins appear as if they are nomads, after some experience you might catch a "pattern" they may follow; and 3) If two red Gremlins are on both sides of you and both block another opening, always kill the one on your left (the Brotan's right).

This game is based more on luck and risk than skill. Will you eat more energy dots, or will you escape down an opening because with one man left you're afraid a red Gremlin will appear and block your path to victory? *Brotan the Blue* is a great escape from the monotonous life of a wizard's apprentice.

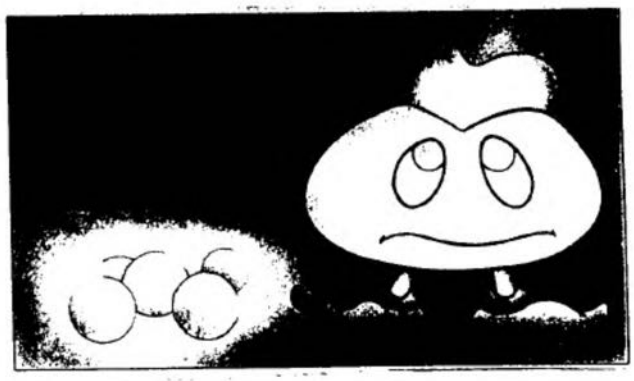


The listing: BROTAN

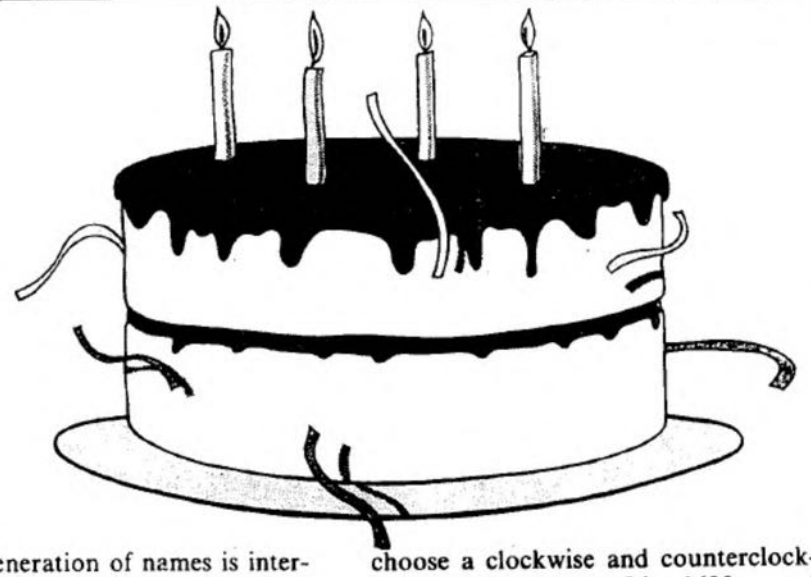
```
0 '*******BROTAN********
2 '(C)1984 BY A.A.M.S.&ASAP CU
4 '*WRITTEN BY ALAN A SAPORTA*
6 '**ALL RIGHTS RESERVED******
8 '*****JULY 1984************
10 PCLEAR8:CLEAR1000:X=32:Y=25:G
Y=8:SC=0:DIM BL(24,16):FR=0:X=RN
D(-TIMER)
12 POKE 65495,0:GOSUB 2000
14 GOSUB400:GOSUB1000:GOSUB1030
16 CL=RND(50):FORCD=32TO224 STEP
 CL:FOR RY=25TO125 STEP 20:CIRCL
E(CD,RY),3,2:NEXT RY,CD:CD=218:X
=32:Y=25:MX=CD:IF RK=5 THEN GOTO
101217 CIRCLE(134,72),5,4:PAINT(
135,72),4,4
18 D=PEEK(342):L=PEEK(343):R=PEE
K(344):S=X:F=Y:RK=0:CD=218
20 IF D=247 AND Y=>25 AND Y<108
THEN GOTO100 ELSE IF Y>108 AND D
=247 THEN GOTO500
22 IF L=247 THEN X=X-2:GOTO28
24 IF R=247 THEN X=X+2:GOTO28
26 GOTO18
28 IF X<32 THEN X=32
30 IF X>218 THEN X=218
32 PUT(S,F)-(S+8,Y+11),SP:IFSC>2
000THENGY=GY+1:IFSC>5000ANDFR=1T
HENGY=GY+2:FR=2:IFSC>10000ANDFR=
2THENGY=GY+3:FR=3
34 PUT(CD,Y)-(CD+8,Y+11),SP:IFSC
<15000 THEN 36 ELSE IFRP=1 THENG
OTO35
35 CLS:PRINT"YOU HAVE BEEN LET O
UT!":PRINT:PRINT"YOU HAVE WON!":
PRINT:PRINT"BUT WOULD YOU LIKE T
O CONTINUE TO SEE HOW HIGH A SCO
RE YOU CAN GET";:INPUTGP$:IFLEFT
$(GP$,1)<>"Y"THEN END ELSE PMODE
3,1:SCREEN1,0:RP=10
36 PUT(X,Y)-(X+8,Y+11),GU
```

```
37 IF Y=65 AND PPOINT(X-1,Y+10)=
4 OR PPOINT(X+10,Y+9)=4 AND X>11
9 AND X<139 THEN SC=SC+50:PMODE4
,1:SCREEN1,1:PMODE3,1:SOUND150,2
:SOUND155,2:SOUND162,2:SCREEN1,0
38 A=PPOINT(X-1,Y):B=PPOINT(X+12
,Y):IFX=32THEN18ELSE IF A=4 OR B
=4 THEN GOTO 200 ELSE IF A=2 OR
B=2 AND X>33 THEN SC=SC+4:PMODE3
,1:SCREEN1,1:PLAY"T20003V31GABBG
CGDAB":SCREEN1,0
40 RR=RND(101):IF RR>50 AND RR<5
5 THEN GOTO 50 ELSE GOTO 18
42 GOTO18
50 MX=CV:CD=RND(188)+32:CR=CD-MX
:IFCR>30 THENCD=MX+30 ELSEIFCR<0
THEN GOTO56
52 CD=MX+CR:PUT(CD,Y)-(CD+8,Y+11
),SP:PUT(CD,Y)-(CD+8,Y+11),MO:GO
TO38
54 GOTO18
56 RF=MX-CR:FOR CV=MX TO RF:PUT(
CV,Y)-(CV+12,Y+11),SP:PUT(CV,Y)-
(CV+12,Y+11),MO:RK=8:GOTO38:NEXT
 CV
58 GOTO18
100 Y(1)=Y:Y(2)=Y+20:IFPPOINT(X+
4,Y+13)<>1THENPOKE342,255:GOTO18
102 FOR YR=Y(1) TO Y(2):PUT(X,YR
)-(X+12,YR+11),GU:NEXT YR:SC=SC+
5:POKE342,255:Y=YR-1:GOTO18
200 RK=1:GY=GY-1:IFGY=0THENGOTO2
08
201 IF GY>10 THEN GY=9
202 COLOR1,1:LINE(150,160)-(220,
190),PSET,BF:DRAW"C3BM152,165D8U
8F8E8D8BR4R6L6U8R6L6D4R3L3D4R6BR
6U8F8U8BR4BD7BR6C2"+N$(GY):IF RK
=3THENGOTO1012
204 SOUND3,4:SOUND2,5:SOUND1,8:G
OSUB300
206 GOTO 18
208 PMODE4,1:SCREEN1,1:PMODE3,1:
PLAY"L80T12EFDCBAG":FORT=1TO900:
NEXTT:PCLS
210 CLS0:PRINT@0,"YOUR SCORE:";S
C
212 PRINT@128,"PLAY AGAIN (Y/N)"
;:INPUTA$:SCREEN0,1:IFLEFT$(A$,1
)="Y"THEN10ELSE CLEAR:END
214 END
300 COLOR1,1:LINE(30,160)-(130,1
80),PSET,BF:SC$=STR$(SC):YS=84:F
ORR=2TOLEN(SC$):S$=N$(VAL(MID$(S
C$,R,1))):DRAW"BM"+STR$(YS)+",17
2C4"+S$:YS=YS+10:NEXTR:DRAW"BM30
,172C4R5U3L5U2R5BR4BD5U5NR5D5R5B
R4NR5U5R5D5BR4U5R6D2L6RF3BR6NR5U
3NR3U2R5"
302 RETURN
400 FORZ=0TO9:READN$(Z):NEXTZ:RE
TURN
402 DATA"U6R4D6NL4BR","BR2BU3E3D
6BR","NR4E5HL3G2BR","R4U3NL2U3L4
BR","BU2U4D4R6L2U4D6BR3","BU6R6L
6D2R4FD2GL4R3BR3","U6D6R5U3L5R5D
3BR","E6L6R6BD6BR","U6R4D3NL4D3N
L4BR","BR4U3L4U3R4D6BR"
500 IFX>48THENGOTO18ELSE IFX=>32
 AND X<48THENFORY=125TO152:PUT(X
,Y)-(X+12,Y+12),GU:PUT(X,Y)-(X+1
2,Y+11),SP:NEXTY:RL=2
502 IF RL<>2THENPOKE342,255:GOTO
18ELSEGOSUB1000:GET(136,48)-(160
,64),BL:SC=SC+15
504 IFPPOINT(72,36)=1THENPUT(60,
28)-(84,44),BL:IFSC>250THENPUT(3
2,48)-(56,64),BL:IFSC>500THENPUT
(200,88)-(224,104),BL:IFSC>750TH
ENPUT(164,28)-(188,44),BL:IFSC>1
000THENPUT(76,68)-(100,84),BL
506 IFSC>1250THENPUT(164,108)-(1
88,124),BL:IFSC>1500THENPUT(164,
28)-(188,44),BL
508 X=32:Y=25:POKE342,255:GOTO16
510 GOTO18
1000 PMODE3,1:PCLS2:COLOR4,2:LIN
E(8,8)-(248,152),PSET,B
1002 DRAW"C4BM8,8F8R20H8L20F8D13
6NH8R16U16NL8H8ND8U4R80U8NH8L80U
8NR72U4R24U8NH8L24U8NR16U4R60U8N
H8L60U8NR52U4R16U8NH8L16U8NR8U4R
44U8NH8L44U8NR36U4R12U8"
1004 DRAW"BM44,8D8F8U8H8R196C2R8
ND7L8C4F8L196D8R180ND4R8D12H8L52
D8F8U8NH8R52BD8L8ND4NL44R8D12H8L
20D8F8U8NH8NR20D8R12ND4R8D12H8L4
8D8F8U8NH8R48BD8L48R40ND4R8D12H8
L16D8F8U8NH8R16BD8L16R8ND4R8D12"
1006 DRAW"H8L52D8F8U8NH8R52BD8L5
2R46ND4R8D12NH8L192H8NR192D16F8U
16D16BRC2L16BM30,8R13BM8,145D7R7
BM76,28C4D8F8U8H8R52F8BM144,28D8
F8U8H8R20F8BM112,108D8F8U8H8R52F
8"
1008 LINE(84,36)-(136,44),PSET,B
:LINE(152,36)-(172,44),PSET,B:LI
NE(56,56)-(116,64),PSET,B:LINE(1
32,56)-(204,64),PSET,B:LINE(100,
76)-(178,84),PSET,B:LINE(64,96)-
(208,104),PSET,B:LINE(120,116)-(
172,124),PSET,B
1010 DRAW"BM48,48D8F8U8H8R60F8BM
124,48D8F8U8H8R72F8BM92,68D8F8U8
H8R76F8BM56,88D8F8U8H8R144F8":FO
R YT=40TO120 STEP20:PAINT(134,YT
),3,4:NEXT YT:PAINT(160,40),3,4:
PAINT(100,20),3,4:PAINT(20,36),3
,4:PAINT(92,60),3,4:PAINT(40,16)
,1,4:RK=3:GOTO202
```

```
1011 RK=5:GOTO16
1012 RETURN
1050 DIM GU(8,11),MO(8,11),SP(8,
11)
1052 PCOPY1TO5
1054 A$="BR2R5L5GD4FR5EU4HL3D4L2
R4DGLD3FR2L2HGL":B$="C4R8L3D3LU3
D3L3D4R8U4L5D3L2R4L2D3G2LRE2F2R2
"
1056 DRAW"BM40,25C3;XA$;"
1058 GET(40,25)-(48,36),GU:PCOPY
5TO1:GET(40,25)-(48,36),SP
1060 DRAW"BM40,25;XB$;":GET(40,2
5)-(48,36),MO:PCOPY5TO1:RETURN
2000 PMODE4,2:SCREEN1,1:PMODE3,2
:PCLS4
2002 A$="ND24R24D24NR8NU24L24D32
R32NU32BR12U56R24D24L24F32BR12U5
6R24D56NL24BR10BU56NR24R12D56BR1
2U12R4U12R4U12R4U12R4U8D8R4D12R4
D12NL20R4D12R4D12BR12U56D8R7D12R
7D12R7D12R7D12U56"
2004 DRAW"BM12,20C1;XA$;BM20,24C
3;XA$;"
2006 DRAW"BD100BL200C3D28R12U12N
L12D12BR8BU4NU16R12NU16D12NL12U1
2BR12BU4R8C2BR12BD8U12NR16U8R16D
20BR6NU20R12BR6U12NR16U8R16D20BR
6U20F20NU20BR20U12NR16U8R16D20BR
4RULD"
2008 DRAW"S5;BD10BL90C3L20D10R20
D10NL20;S4;BR6U12R16D4NL16D8BR6U
12R16D4NL16BD8BR6U12R16D12NL16BR
6U12R16D4L16R8F8BR13U12L7R16BD12
BR6U12R16D4NL16D8"
2010 FORR=1TO1000:NEXTR:PAINT(18
0,37),2,3:PAINT(50,64),2,3:PAINT
(40,25),2,3:PAINT(120,35),2,3:PA
INT(75,35),2,3:PAINT(0,0),2,3:PA
INT(40,148),2,3
2011 POKE65494,0:PLAY"T8V3102BAG
AB;P1;B;P1;BBBBB;P1;A;P1;A;P1;AA
AAA;P1;B;P1;B;P1;BBBBB;P1;BAGA;B
;P1;B;P1;BBBBB;P1;A;P1;A;P1;B;P1
;A;P1;GGGGGGGG":POKE65495,0
2012 EXEC44539:PCLS:PMODE3,1:SCR
EEN1,0:PCLS2:RETURN
```



CoCo Bakes A Cake



By John Plaxton

Birthday is a short program that draws a cake and decorates it to your specifications. You can display a name and an age, and select either red or blue flickering candles. Although designed as a birthday cake, it can be used for other festivities such as anniversaries, etc. And, it is low in calories.

Using this program, like making a cake, will involve a little work. After all, you can't buy a culinary masterpiece in a fast-food restaurant. *Birthday* needs a 16K, or larger, mixing bowl; the decorations are generated by using the ingredients PMODE3, Extended Color BASIC and the DRAW command. To generate any name you will have to be familiar with the subcommands F, R, E, D, L, U, G, H, GM(,), (M+,-,) and perhaps LINE-(,).

DRAW is an extremely powerful command (and I suspect the envy of owners of other computer systems). It is easy to use once you get the idea and try a couple of drawings. The instructions on pages 54-62 in Radio Shack's *Going Ahead With Extended BASIC* are quite clear. However, one point not covered is that sometimes if you use U, D, R, etc., and follow it by a semicolon, you will get a syntax error. I always have a number (including '1') preceding the semicolon.

Another point to consider is the width of the vertical and horizontal lines. Vertical lines tend to be about half the width of horizontal lines, therefore two vertical lines are drawn: one line up, move over one pixel, then another line of the same length down.

The generation of names is interesting because you must try to make them follow the curve of the cake. You will need to use the "M+,+" subcommand because you will have to draw lines at angles other than in 45 increments. (If you were to have a 360 degree navigational protractor, U=000°, E=045°, R=090°, F=135°, D=180°, G=225°, L=270° and H=315°.)

To see what I mean, RUN the program, select 4* LARRY and look at the letter 'L.' On the cake, the bottom of the 'L' is horizontal and does not appear to be resting on the surface of the cake. It should have been drawn tipped slightly downwards. To do this, the last direction in Line 1390 (R8) should be replaced by ;M+8,+3; which DRAWS a line that ends eight pixels to the right of, and three pixels down from, the beginning position. If you know the coordinates where a DRAW command ends you could use the LINE-(,) command to draw the same line. For example, you can get the same result by writing `1390 DRAW "BM84,122; U16RD16":LINE-(93,124),PSET"L.`

Each letter will have to be drawn individually (even if used twice in a name) because of the need to maintain perspective (or curve) of the cake. Look closely at the coding for the 'M's in MOM (lines 1550 and 1570).

The flickering candle effect is generated by continually redrawing the flame using one of two colors. Once all the candles are drawn (JJ=0), random candles and colors are chosen. Unfortunately, the flickering always occurs in a clockwise direction. To randomly choose a clockwise and counterclockwise direction, change Line 1690 to read `1690 IF RND(2)=1 THEN FOR C=3.3 to 9.58 STEP 6.28/AG ELSE FOR C=9.58 to 3.3 STEP-6.28/AG`. However, because of the number of candles and their position change with the selected age, and since there are very small round-off errors in the numerical calculations, extra candles will be drawn.

The rotating plate posed an interesting problem because of the way the PAINT command works. Remove lines 2000-2010 by inserting a REM (or ') at the beginning of each line, then RUN and notice the difference. Drawing a line that allows the paint to "seep through" at the bottom of the plate gives the appearance of continuous motion.

Having described how the DRAW commands are used, I hope you will enjoy DRAWing your own names. Of course, rosettes and other decorations can be easily added. I would appreciate receiving the code for all names generated.

For those of you who have programs that synthesize four-voice music, I strongly encourage replacement of Line 1970 with appropriate code. For the same reasons you bought a CoCo, your masterfully decorated cake deserves the best.

```
200 .....83
1360 .....14
1530 .....5
1730 .....178
1900 .....111
END .....177
```

The listing: BIRTHDAY

```
0 '*** BIRTHDAY CAKE *****
10 CLS(RND(9)-1):PRINT@203," BIR
THDAY ";
20 PRINT@460," COPYRIGHT 1985 ";
30 PRINT@492,"cancoco software";
40 FOR T=0 TO 1500:NEXT T
50 CLS:PRINT:PRINT"WHOSE NAME WI
LL BE ON THE CAKE:  1* RAINBOW"
60 PRINT" 2* MOM":PRINT" 3* DAD"
70 PRINT" 4* LARRY"
80 PRINT" 5* ROBERT"
90 PRINT" 6* "
100 PRINT" 7* "
110 PRINT" 8* "
120 PRINT" 9* "
130 NA$=INKEY$:IF NA$=""THEN130
140 NA=VAL(NA$):IF NA=0THEN130
150 PRINT"HOW OLD IS THE BIRTHDA
Y KID";:INPUT AG
160 IF AG>99 THEN150
170 IF AG<2 THEN RA=0 ELSE IF AG
<11 THEN RA=.6 ELSE RA=1
180 PMODE3,1:PCLS0:PCLS5:SCREEN1
,1:COLOR6,5
190 CIRCLE(128,75),84,6,.4
200 LINE(44,75)-(44,125),PSET
210 LINE(212,75)-(212,125),PSET
220 CIRCLE(128,125),84,6,.4,0,.5
230 CIRCLE(128,100),84,6,.4,0,.5
240 CIRCLE(128,125),120,7,.4,.89
,.64
250 CIRCLE(128,122),94,7,.4,.94,
.57
260 COLOR7,5:LINE(43,92)-(43,112
),PSET
270 LINE(216,93)-(216,113),PSET
280 ON NA GOSUB1460,1550,1610,13
90,1310,600,800,1000,1200
290 GOSUB1650:GOTO1800
500 '***
600 RETURN
700 '***
800 RETURN
900 '***
1000 RETURN
1100 '***
1200 RETURN
1300 '*** ROBERT
1310 DRAW"BM74,120;U14E2;M+6,+2;
F2D5G2;M-8,-2;R2":DRAW"M+8,+9;"'
R
1320 DRAW"BM92,123;BU2U10E2;M+6,
1;F2D10G2;M-6,-1;H2"'O
1330 DRAW"BM114,126;U14R8F2D2G2L
6R6F2D4G2L6"'B
1340 DRAW"BM134,126;U14R8L8D7R8L
8D7R8"'E
1350 DRAW"BM154,125;U14E2;M+6,-1
;F2D5G1;M-8,+1;R3;M+7,+7;"'R
1360 DRAW"BM179,120;U14;BM-5,2;M
+10,-4;"'T
1370 RETURN
1380 '*** LARRY
1390 DRAW"BM84,122;U16RD16R8"'L
1400 DRAW"BM102,126;U12E4R2F4D4L
10R10D8"'A
1410 DRAW"BM124,128;U14E2R4F2D5G
L6R":DRAW"M+7,+7;"'R
1420 DRAW"BM144,126;U14E2R4F2D5G
L6R":DRAW"M+7,+7;"'R
1430 DRAW"BM169,124;U10NH6E8"'Y
1440 RETURN
1450 '*** RAINBOW
1460 DRAW"BM120,128;U14;M+10,+14
;U14;"'N
1470 DRAW"BM109,127;U14;"'I
1480 DRAW"BM141,128;U14;M+8,-1;F
2D2G2;M-8,+2;M+8,-2;F4D2G2;M-8,+
1;"'B
1490 DRAW"BM088,124;U10E4R2F4D4;
M-10,-1;M+10,+1;D7;"'A
1500 DRAW"BM161,125;BU2U10E2;M+6
,-1;F2D10G2;M-6,+1;H2"'O
1510 DRAW"BM071,118;U14;M+7,+3;F
2D4G2;M-7,-4;M+5,+4;M+4,+7;"'R
1520 DRAW"BM178,120;BU12;M+5,+12
;M+3,-10;M+2,+8;M+4,-16;"'W
1530 RETURN
1540 '*** I LOVE YOU
1550 DRAW"C8;BM098,126;U14F7E6D1
4"'M
1560 DRAW"BM122,128;BU2U10E2R6F2
D10G2L6H2"'O
1570 DRAW"BM144,127;U14F6E7D14"'
M
1580 IF AG>29 AND AG<33 THEN AG=
29
1590 RETURN
1600 '*** DAD
1610 DRAW"C6;BM100,126;U14R8F2D1
0G2L8"'D
1620 DRAW"BM122,127;U10E4R2F4D4L
10R10D6"'A
1630 DRAW"BM144,126;U14R8F2D10G2
L8"'D
1640 RETURN
1650 '*** DRAW CANDLES
1660 IF AG=0 THEN1800
1670 IF NA=3THEN C$="C6;U20FR4EH
L4GER2;XC1$;UHHUUEUEFDFDDGGL2D1;
C6;R3FD20GL4H"'CANDLE
1680 IF NA<>3THEN C$="C7;U20FR4E
HL4GER2;XC1$;UHHUUEUEFDFDDGGL2D1
;C7;R3FD20GLUH"
1690 FOR C=3.3TO9.58STEP6.28/AG
1700 IF JJ=3 AND RND(3)=1THEN177
0
1710 X=INT(COS(C)*60*RA)+128
1720 Y=INT(SIN(C)*25*RA)+75
1730 IF RND(4)=2THEN C1$="C7" EL
SE C1$="C8"
1740 DRAW"BM"+STR$(X)+","+STR$(Y
)+";"+C$
1750 IF NA=3THEN PAINT(X+3,Y-15)
,6,6:PAINT(X+3,Y-4),6,6
1760 IF NA<>3THEN PAINT(X+5,Y-15
),7,7:PAINT(X+3,Y-4),8,7
1770 NEXT C
1780 JJ=3
1790 RETURN
1800 '*** NUMERALS
1810 AG$(0)="BU2U10E2R6F2D10G2L6
H2"
1820 AG$(1)="R8L4U14G2"
1830 AG$(2)="BU12E2R6F2D4G2L6G2D
4R10"
1840 AG$(3)="BU12E2R6F2D3G2L4R4F
2D3G2L6H2"
1850 AG$(4)="BU14D7R12L4U7D14"
1860 AG$(5)="BM+10,-14;L10D4F2R6
F2D4G2L6H2"
1870 AG$(6)="BM+10,-12;H2L6G2D10
F2R6E2U4H2L6G2"
1880 AG$(7)="BU14R10DL10R10;M-7,
+14
1890 AG$(8)="BU2U3E2R6L6H2U3E2R6
F2D3G2F2D3G2L6H2"
1900 AG$(9)="BU7U5E2R6F2D4G2L6H2
F2R6E2D6G2L6H2"
1910 '*** DRAW AGE
1920 A1=AG:IF AG>9 THEN A1=INT(A
G/10):A2=AG-A1*10
1930 IF AG<10 THEN DRAW"BM"+STR$
(122)+","+STR$(152)+";"+AG$(A1)
1940 IF AG>9 THEN DRAW"BM"+STR$(
112)+","+STR$(152)+";"+AG$(A1)
1950 IF AG>9 THEN DRAW"BM"+STR$(
132)+","+STR$(152)+";"+AG$(A2)
1960 '*** PLAY SONG
1970 PLAY"O2L8DDL4ED6L2F+L8DDL4E
DAL2GL8DDL4O3DO2BGF+EP8O3L8CCL40
2BGAL2G"
1980 'COLOR7,5:LINE(128,158)-(12
8,170),PSET
1990 FOR JP=6TO8STEP2
2000 IF RND(AG)>2*AG/3 THEN2050
2010 COLOR7,5:LINE(128,158)-(128
,170),PSET
2020 PAINT(40,94),JP,7
2030 COLORJP,5
2040 ON NA GOSUB1460,1550,1610,1
390,1310,600,800,1000,1200
2050 GOSUB1660
2060 NEXT JP
2070 LP=LP+1:IF LP>20THEN CLS:RU
N
2080 IF RND(100)>85THEN1970ELSE1
990
```

A-MAZE-ING MANEUVERING WITH JETPACK CHALLENGE

by Joey Goodson

In *Jetpack Challenge* you are beamed down from your starship to planet Mazon to practice maneuvering skills with your jetpack. You are beamed down to the maze of your choice on the planet and the task is to get out of the maze as quickly as possible without hitting the sides of the maze.

When you run the program, a menu comes up giving you the choice of four mazes and a choice to change the parameters of the program. The mazes are numbered in order from one to four — from easy to hard — and are named accordingly. For example, in the teleporter maze, you fly to the teleporter at the bottom left of the screen and it teleports you to the top part of the maze.

In choosing to change the parameters, you are able to change the rate of acceleration horizontally and vertically. You are also given the option to make the walls lethal (deadly) to touch or not. The program is self-explanatory and will show how to change the parameters.

To control the man, use the right joystick. Push the button to thrust upward and move the joystick right or left to go in any of those directions. If you center the joystick, there is no acceleration taking place to the left or right; therefore, if you are going in a particular direction, move your joystick in the opposite direction to slow down or to stop.

Jetpack Challenge is fun for competing against a friend to see who can get through the mazes the fastest, or just trying to beat your previous time. The program also keeps track of who has the best time on each maze.

```
20 .......181   3111 .....154
90 .......220   3216 ......77
1040 .....170   3320 .....101
2165 ......80   10035 ....244
2195 ......63   END ......80
```

The listing: JETPACK

```
1 '*****************************
  *                           *
  *     JETPACK CHALLENGE     *
  *           BY              *
  *       JOEY GOODSON        *
  *                           *
  *****************************
5 FORX=1TO4:T(X)=0:T$(X)="MR. NO
BODY":NEXTX:M$(1)="BEGINNER'S MA
ZE":M$(2)="REGULAR MAZE":M$(3)="
TELEPORTER MAZE":M$(4)="DEATH MA
ZE"
7 'T(X)  = TIME FOR EACH MAZE***
  T$(X) = PERSON'S NAME     ***
  M$(X) = NAME OF EACH MAZE ***
10 DIM RS(1,1), LS(1,1), SB(1,1)
'DIDN'T HAVE TO DIM A LARGE
 AMOUNT OF SPACE SO (1,1) WORKS
11 'DEFAULT PARAMETERS**********
12 VX=.2:VY=.3:RG=.1:FF=0
15 PMODE4,1:COLOR5,0:PCLS:GOSUB1
000'DRAW SPACEMAN
16 GOSUB2100'CHOOSE MAZE
19 'X,Y = ACTUAL COORDINATES****
    XT,YT = TEMPORARY COOR. ****
    XX,YY = INCR. OR DECR.  ****
    LEFT AND UP DECR.       ****
    XV,YV = ACCELLERATION   ****
    SS = TELLS WHICH SPACEMAN***
   PICT. TO USE (0-R,1-L)  ****
20 '***FT = FINISHED MAZE AND
GETTING MAN TO DISAPPEAR***FM =
WHETHER MAZE IS LETHAL OR NOT***
TG=TOGGLE FOR SOUND EFFECTS AT
MENU21 SS=0:XT=X:YT=Y:XX=0:YY=0:
XV=VX:YV=VY:GR=RG:FT=0:FM=FF:TT=
0:TG=0
37 SCREEN1,1:FORP=1TO460:NEXTP:G
OSUB3265
38 'MAIN PROGRAM****************
39 TIMER=0
40 IFJOYSTK(0)>49THENXX=XX+XV:SS
=0
50 IFJOYSTK(0)<14THENXX=XX-XV:SS
=1
60 IFPEEK(65280)=126ORPEEK(65280
)=254THENYY=YY+(GR-YV) ELSEYY=YY
+GR
70 X=X+XX:Y=Y+YY
75 IFY<0THENTT=TT+TIMER:GOTO100
76 IFX<9ANDMA=3THENGOSUB3250
80 PUT (XT,YT)-(XT+8,YT+12),SB,P
SET
81 IFPPOINT(X,Y)=5ORPPOINT(X+8,Y
+12)=5ORPPOINT(X+8,Y)=5ORPPOINT(
X,Y+12)=5THEN10000
85 IFSS=0THENPUT(X,Y)-(X+8,Y+12)
,RS,PSETELSEPUT(X,Y)-(X+8,Y+12),
LS,PSET
90 XT=X:YT=Y:GOTO40
91 'END MAIN PROGRAM************
100 'FINISHED MAZE**************
105 PUT(XT,YT)-(XT+8,YT+12),SB,P
SET:X=XT:Y=0
106 IFSS=0THENPUT(X,Y)-(X+8,Y+12
),RS,PSETELSEPUT(X,Y)-(X+8,Y+12)
,LS,PSET
110 FT=1:FORP=1TO150:NEXTP:GOSUB
3255:FORP=1TO460:NEXTP:GOTO10100
1000 'DRAWING SPACE MAN   ******
      RS = RIGHT SPACEMAN PICT.*
      LS = LEFT  SPACEMAN PICT.*
      SB = ERASE SPACEMAN PICT.*
```

```
                                ******
1010 LINE(124,84)-(126,90),PSET,
BF:LINE(127,85)-(130,94),PSET,BF
:LINE(127,82)-(127,84),PSET:LINE
(128,82)-(131,85),PSET,B:LINE(13
1,88)-(132,88),PSET:PSET(132,87)
:LINE(131,94)-(132,94),PSET:PSET
(131,86)
1030 LINE(118,84)-(120,90),PSET,
BF:LINE(114,85)-(117,94),PSET,BF
:LINE(117,82)-(117,84),PSET:LINE
(113,82)-(116,85),PSET,B:LINE(11
2,88)-(113,88),PSET:PSET(112,87)
:LINE(112,94)-(113,94),PSET:PSET
(113,86)
1040 GET(124,82)-(132,94),RS,G
1045 GET(112,82)-(120,94),LS,G
1050 GET(124,102)-(132,114),SB,G
1060 PCLS:RETURN
2100 CLS:PRINT@7,"JETPACK CHALLE
NGE":PRINT:FORX=1TO4:PRINT"(";X;
") ";M$(X):NEXTX:PRINT"( 5 ) CHA
NGE PARAMETERS"
2104 IFTG=1THENTG=0:GOTO2110
2105 J=RND(5):J$="O"+STR$(J):PLA
Y"T255V1"+J$
2106 FORJ=1TO15:PLAY"V+ABGABG":N
EXTJ
2107 FORJ=1TO15:PLAY"V-ABCDEFGAB
CDEFG":NEXTJ
2110 PRINT:INPUT"WHICH MAZE (1-4
) OR (5)";MA:MA=INT(MA):IFMA)0AN
DMA<6THEN2115ELSE2100
2115 PRINT:PRINT"WAIT A MINUTE P
LEASE -          BEAMING DOWN TO
 MAZE."
2120 ON MA GOSUB 3000,3100,3200,
3300,2150
2130 GOTO19
2150 'CHANGE PARAMETERS*********
2155 TG=1:CLS:PRINT@7,"CHANGE PA
RAMETERS"
2160 PRINT:PRINT"PARAMETERS ARE
SET AS FOLLOWING ON STARTUP OF T
HE PROGRAM******":PRINT"HORIZONT
AL THRUSTS = ";VX:PRINT"UPWARD T
HRUST = ";VY:PRINT"GRAVITY = ";R
G
2165 PRINT"FOR EASIER CONTROL TR
Y*********":PRINT"HORIZONTAL THR
USTS = .1":PRINT"UPWARD THRUST =
 .2":PRINT"GRAVITY = .1"
2170 PRINT@365,"CHANGE"
2175 INPUT"HORIZONTAL THRUSTS TO
 - ";VX:INPUT"UPWARD THRUST TO -
";VY:INPUT"GRAVITY TO - ";RG
2180 CLS:PRINT@7,"CHANGE PARAMET
ERS"
2185 PRINT:PRINT"NOW, WOULD YOU
LIKE THE WALLS OFTHE MAZE TO BE
LETHAL OR NOT.   IF LETHAL - YOU
 DIE WHEN YOU    TOUCH A WALL, I
F NON-LETHAL -   YOU GET RESET B
ACK TO YOUR      PREVIOUS POSITI
ON BEFORE YOU    TOUCHED THE WAL
L AND HAVE 5"
2186 PRINT"SECONDS ADDED TO YOUR
 TIME."
2187 PRINT"UPON STARTUP OF THE P
ROGRAM,    THE WALLS ARE SET TO
-";FF:PRINT"(0=LETHAL, 1=NON-LET
HAL)"
2190 PRINT:INPUT"WOULD YOU LIKE
THE WALLS LETHAL OR NOT (0=YES,
1=NO)";FF:IFFF<>0ANDFF<>1THEN218
0
2195 CLS:PRINT@7,"PARAMETERS CHA
NGED":PRINT"PARAMETERS ARE NOW C
HANGED TO**":PRINT"HORIZONTAL TH
RUSTS = ";VX:PRINT"UPWARD THRUST
 = ";VY:PRINT"GRAVITY = ";RG:PRI
NT"WALLS = ";FF
2200 PRINT"TO MAKE YOUR CANGES P
ERMANENT,  CHANGE VARIABLES IN (
LINE 12)   TO YOUR CHOICE THEN S
AVE THE    PROGRAM AGAIN. THE VA
RIABLES =  VX = HORIZONTAL THRUS
TS":PRINT"VY = UPWARD THRUST":PR
INT"RG = GRAVITY":PRINT"FF = WAL
LS (LETHAL OR NOT)"
2205 PRINT"PRESS (2) TO GO TO MA
IN MENU"
2210 IFINKEY$="2"THEN2100ELSE221
0
3000 'BEGINNER'S MAZE***********
3010 DRAW"BM255,8;L4D167L87U108L
85D13R68D107L143U47R104U13L68U10
7R163D108R9U117L12U4L20U3L25U4"
3020 POKE178,2:PAINT(0,0),,5
3030 X=28:Y=157
3040 RETURN
3100 'REGULAR MAZE**************
3110 DRAW"BM160,0;D12R8U8R36D8R3
9D64L100U5L28U11L12U12L8D63L32U1
6L24U67L11D84R28D51R32U16R48U31R
68D47R23U55L96D7L27U39R27D7R120D
104L75U47L12D31L52D16L80U51L28U1
32R59D67"
3111 DRAW"R8U48R53D12R12D12R28D4
R55U15L15D7L36U7L25U4L3U5L4U15L8
U2L9U2L7U3L12U1L8U1L4U2L4U1"
3120 POKE178,2:PAINT(0,0),,5
3130 X=116:Y=88
3140 RETURN
3200 'TELEPORTER MAZE***********
3210 DRAW"BM35,51;L19U47R19D24R1
9"
3211 LINE(55,28)-(167,92),PSET:L
INE(35,51)-(147,115),PSET
3212 DRAW"BM167,92;R44D27R32D64L
95U23L17D27L64U24":DRAW"BM147,11
5;R44D27R32D17L55U23L81":DRAW"BM
95,152;R8D12L8U12"
3213 LINE(19,139)-(67,163),PSET:
LINE(39,116)-(87,136),PSET
3214 DRAW"BM19,139;D8L15U39R15D8
R20":DRAW"BM167,47;D8L15U39R15D8
R20"
3215 LINE(187,24)-(219,36),PSET:
LINE(167,47)-(199,63),PSET
3216 DRAW"BM219,36;R12U24L15U4L2
0U4L12U4":DRAW"BM199,63;R52U55R4
"
3220 POKE178,2:PAINT(0,0),,5:PAI
NT(100,156),,5
3230 X=22:Y=12
3240 RETURN
3250 'TELEPORTING***************
3252 XX=0:YY=0:PUT(XT,YT)-(XT+8,
YT+12),SB,PSET
3254 PUT(X,Y)-(X+8,Y+12),LS,PSET
3255 'DISAPPEARING**************
3256 FORP=12TO1STEP-1
3258 PUT(X,Y)-(X+8,Y+12),SB,PSET
:IFSS=0THENPUT(X,Y)-(X+8,Y+P),RS
,PSETELSEPUT(X,Y)-(X+8,Y+P),LS,P
SET
3259 PLAY"T255O4V"+STR$(P*2)+"AB
CG"
3260 NEXTP
3262 PUT(X,Y)-(X+8,Y+12),SB,PSET
3263 IFFT=1THENRETURN
3264 FORPP=1TO150:NEXTPP:X=158:Y
=29:XT=X:YT=Y'FOR TELEPORTING
MAZE
3265 'APPEARING*****************
3266 FORP=1TO12
3268 PUT(X,Y)-(X+8,Y+P),RS,PSET
3269 PLAY"T255O4V"+STR$(P*2)+"AB
CG"
3270 NEXTP
3272 SS=0
3275 RETURN
3300 'DEATH MAZE****************
3310 DRAW"BM255,13;L4D35L23D12R2
3D127L52U56L36U36L8D41R24D31R12D
20L119U68L38U56R23U27L12D20L19D6
8R39D63L59U35R31D15R8U23L39U120R
12D12R12U28R51D48R21D24L5D59R32D
28R36U11L27U32L18U13L12U52R8U31R
77D20L12D63R40D48R8U55L43U44R27U
33"
3320 DRAW"R21U14L73U2L7U3L5U2L3U
2L3U2L5U2"
3330 DRAW"BM70,99;R10U14L10D14;B
M134,95;R8U19R21U28L21D19L8D27"
3340 POKE178,2:PAINT(1,1),,5:PAI
NT(71,98),,5:PAINT(152,60),,5
3350 X=14:Y=160'START COOR.S
3360 RETURN
```

```
10000 'KILLED
10010 IFFN=1THENX=XT:Y=YT:XX=0:Y
Y=0:TT=TT+300:PLAY"T25503V15AAAA
AAAAAA":GOTO85
10015 IFSS=0THENPUT(X,Y)-(X+8,Y+
12),RS,PSETELSEPUT(X,Y)-(X+8,Y+1
2),LS,PSET
10020 PLAY"T25501V30":FORP=1TO30
:PLAY"V-AAA":NEXTP
10025 FT=1:FORP=1TO150:NEXTP:GOS
UB3255
10030 CLS:PRINT"SORRY, YOU DIDN'
T MAKE IT       THROUGH THE ";M$
(MA);".":PRINT"YOUR FUNERAL WILL
 BE HELD       TOMORROW. I HOPE
YOU CAN MAKE   IT."
10035 FORP=1TO460*8:NEXTP:GOTO10
140
10100 CLS:PRINT"CONGRADULATIONS!
 YOU MADE IT OUTOF THE ";M$(MA);
"!"
10110 PRINT"IT TOOK YOU "INT(TT/
3600)"MIN"INT(TT/60)-INT(TT/3600
)*60"SEC"
10130 IFT(MA)=0ORTT<T(MA) THENT(
MA)=TT:GOTO10134ELSEFORPP=1TO460
*5:NEXTPP:GOTO10140
10134 PLAY"T255V1"
10135 FORP=1TO15:PLAY"V"+STR$(P*
2)+"O2CDEFGAB03CDEFGAB04DEFGAB05
CDEFGAB":NEXTP
10137 INPUT"YOU COMPLETED THE MA
ZE IN THE   LEAST TIME. WHAT IS
YOUR NAME   CHAMPION";T$(MA)
10140 CLS:PRINT27,"JETPACK CHAMP
IONS":PRINT
10160 FORP=1TO4
10170 PRINTN$(P);" -"
10180 PRINTT$(P),INT(T(P)/3600);
"MIN";INT(T(P)/60)-INT(T(P)/3600
)*60"SEC"
10190 IFP=4THEN10200ELSEPRINT
10200 NEXTP
10205 PLAY"T25501V30":FORP=1TO30
:PLAY"V-CDEFGAB":NEXTP
10210 PRINT:PRINT"PRESS (2) TO P
LAY AGAIN."
10220 IFINKEY$="2"THENCOLOR5,0:P
CLS:GOTO16ELSE10220
```

GAME | 16K ECB | 32K Disk | the RAINBOW

Computer Trapshooting

By Richard W. Rutter

This Extended Color BASIC game simulates the real life sport of skeet shooting, in which each participant, using a shotgun, attempts to hit flying clay disks thrown from traps. This type of shooting is lots of fun, however, it tends to get fairly expensive if vigorously pursued. Our alternative, *Computer Trapshooting*, is both enjoyable and inexpensive. A 32K Color Computer and a set of joysticks are all that's needed.

Let's first discuss the game's objectives and how to play it. There are three play modes: "Singles," "International Singles" and "Doubles." Singles and International Singles are very similar except that in the latter, targets fly noticeably higher or lower, making it more difficult to hit them. With Doubles, you have to shoot at two targets instead of one.

One shot per target is allowed, and you must shoot before the target begins to fall or it will automatically be scored a miss (the real life game is more forgiving). Your score, the number of shots and the number of actual hits, is displayed and updated after each shot. For Singles and Doubles, you may choose any one of nine difficulty levels. In International Singles, the difficulty level will automatically be set at nine, the highest level.

The goal is fairly obvious: Try to hit as many targets as possible, and be able to do so at difficulty level nine. You will probably find Doubles the most challenging, but practice will be needed in order to master any of the three play modes.

The joysticks are very important. The right joystick is for shooter #1 and the left joystick is for shooter #2, if any. The firebutton fires the shot pattern. The gun sight with the small white bead in the middle is what you use to chase the targets. Using the joystick, the gun sight may be moved up, down, left or right. To launch the target, pull the joystick all the way down.

I recommend positioning the gun sight at the top center of the pill-box shaped trap house before launching the target so you'll be able to catch up with it. Practice using the joystick to manipulate the gun sight before actually attempting to shoot any targets. You may change the gun sight's picture by pressing 'S' — try it out. If you happen to shoot before the target is in motion, don't worry; each result is not scored until a target has been shot and broken, or it hits the ground.

When two shooters are playing, there are two sets of scores. The set at the

bottom is always for shooter #1. The middle set is for shooter #2. Each player's turn can be determined by which "shot at" box is enclosed with an orange rectangle. The proper joystick responds only when it is the appropriate player's turn. You may find it more entertaining when competing with someone else.

You may return to the option menu before completing the current session by pressing SHIFT CLEAR. You are notified with both a sight and sound display when the current shooting session is complete. You then need to press SHIFT CLEAR to return to the option menu, as described.

It should be noted that *Computer Trapshooting* uses the POKE 65495,0 option to double the execution speed. If your computer cannot operate at that speed, you need to remove the POKE commands and raise the octaves of the PLAY commands for proper sound effects.

The Graphics Display

A few more comments are needed about the graphics display. Note the green number in the middle of the trap house. This tells the number of targets you will need to shoot. Note the white number at the middle left. This number tells what difficulty level you are in, from L1 to L9. The difficulty level determines the likelihood of the targets coming out of the trap house at angles. At level one, all single targets come straight out. At level nine, many angled targets are likely. The white number at the middle right tells which play mode you are in, M1, M2 or M3.

Tips to Improve Your Score

Here are a few tips to help improve your scores. You need to react quickly after the target is thrown. To assure a hit, position the gun sight just below and at the center of the target. Press the firebutton and hold it down until the shot is fired. Be sure to continue moving the gun sight with the target or you'll probably shoot behind it.

When shooting Doubles, remember to release the firebutton after the first shot so you don't waste your second shot. Also, shoot the straight target first, then go after the angle target; it is much easier that way.

Program Structure

Finally, let's consider the program's structure. Although it can fit into a 16K cassette system, a 32K system is necessary to assure that it always runs properly. This is due to numerous PPOINT checks, which tend to gobble up memory bytes. If you have a 16K system and would still like to use *Computer Trapshooting*, you need to remove the Doubles play option and shorten the menus. At least 1,500 bytes must be removed in order to adapt the program to 16K. An easier way is to upgrade your computer to 32 or 64K — it is a worthy investment.

Here is a list of the variables used by the program. They are (in alphabetical order): A$, AF, AM, AT, BS, C, D, DL, DO, DS$, DT, G0, G1, G2, G3, G4, GS, H2, HC, HI, I, IN, J, LT, M, N, N$(0)-N$(9), N1, N2, N3, N4, ND, P, PF, R$, RA, S$, SA, SB, SC, SH, SP, SP$, T, T1, T2, T3, T4, T5, TC, TS, UM, X and Z. If you make additions to the program, in order to avoid needless problems, use variable names other than those found in the above list. Before loading the program, PCLEAR4 to assure that your system is set up for graphics. Also, a CLEAR200 will reserve ample space for it.

I now go trapshooting whenever I want, rain or shine, and it doesn't cost the proverbial arm or leg. You can do it, too!

170 127	710 106	1240 118
370 252	880 167	1400 239
500 206	1040 136	END 137
600 39	1140 255	

The listing: TRPSHOOT

```
10 CLS0:CLEAR200:DIMT(1,1):DIMBS
(1,1):DIMLT(1,1):DIMGS(1,1):DIMS
P(2,2):DIMSB(2,2):DIMDT(2,2):UM=
-4:G1=118:G2=114:G3=128:G4=110:D
S$=CHR$(128):POKE65495,0
20 N$(1)="BR2U6G2E2D6L2R4BR2"
30 N$(2)="BR2R4L6U2R4U4L4R4D4BD2
BR4"
40 N$(3)="R4U3L4R4U3L4R4D6BR2"
50 N$(4)="BR4U6D3L4U3D3R4BD3BR2"
60 N$(5)="R4U3L4U3R4BD6BR2"
70 N$(6)="R4U3L4D3U6D6R4BR2"
80 N$(7)="BR4U6L4D1U1R4D6BR2"
90 N$(8)="R4U3L4D3U6R4D6BR2"
100 N$(9)="BR4U6L4D3R4D3BR2"
110 N$(0)="U6R4D6L4BR6"
120 SP$="BR2"
130 PRINT@33,"richard"DS$"rutter
"DS$"welcomes"DS$"you"DS$"to";:S
CREEN0,1
140 PRINT@101,"computer"DS$"trap
shooting";:SCREEN0,1
150 PRINT@197,"your"DS$"selectio
n";:SCREEN0,1:PRINT@261,"1"DS$"s
ingles";:PRINT@325,"2"DS$"intern
ational"DS$"singles";:SCREEN0,1
160 PRINT@389,"3"DS$"doubles";:P
RINT@453,"4"DS$"end"DS$"program"
;:SCREEN0,1:IN=0:DO=0:RA=0:T5=0:
H2=0
170 A$=INKEY$
180 IFA$="1"THEN230
190 IFA$="2"THENIN=1:GOTO230
200 IFA$="3"THENDO=1:RA=2:GOTO23
0
210 IFA$="4"THEN1570
220 GOTO170
230 CLS0:IFDO=1THENPRINT@193,"en
ter"DS$"number"DS$"of"DS$"10"DS$
"pair"DS$"rounds";:GOTO250
240 PRINT@192,"enter"DS$"number"
DS$"of"DS$"25"DS$"target"DS$"rou
nds";:SCREEN0,1
250 PRINT@302,"1-9";:SCREEN0,1
260 A$=INKEY$
270 IFVAL(A$)>0THEN290
280 GOTO260
290 IFND=1THENAF=VAL(A$):GOTO340
```

```
3ØØ DL=VAL(A$)*25
31Ø IFDO=1THENDL=(DL*4)/5
32Ø IFIN=1THENAF=9:GOTO34Ø
33Ø CLSØ:PRINT@228,"enter"DS$"di
fficulty"DS$"level"DS$"1-9";:SCR
EENØ,1:IFND=ØTHENND=1:GOTO26Ø
34Ø CLSØ:PRINT@226,"enter"DS$"nu
mber"DS$"of"DS$"shooters"DS$"1-2
";:SCREENØ,1
35Ø A$=INKEY$:IFA$="1"THEN38Ø
36Ø IFA$="2"THEN38Ø
37Ø GOTO35Ø
38Ø M=VAL(A$):IFPF=1THEN54Ø
39Ø CLSØ:PRINT@224,"ready"DS$"on
"DS$"the"DS$"left";:SCREENØ,1
4ØØ PMODE3,1:PCLS7
41Ø DRAW"C8BM11Ø,151U2ØR5ØD2ØL5Ø
H15U2ØF15R5ØH15L46BD2ØBL2L98BR16
ØR95"
42Ø PAINT(1Ø5,13Ø),6,8:PAINT(13Ø
,145),6,8:PAINT(125,125),6,8
43Ø CIRCLE(13Ø,175),7Ø,8,.15,.Ø1
,.5:CIRCLE(13Ø,165),5Ø,8,.15,.Ø1
,.5
44Ø DRAW"C8":LINE(6Ø,175)-(8Ø,16
5),PSET:LINE(2ØØ,175)-(18Ø,165),
PSET
45Ø PAINT(75,173),5,8
46Ø LINE(85,182)-(1Ø3,17Ø),PSET:
LINE(113,183)-(123,173),PSET:LIN
E(145,184)-(141,172),PSET:LINE(1
77,182)-(163,17Ø),PSET
47Ø PAINT(5,18Ø),6,8
48Ø CLSØ:PRINT@238,"ready"DS$"on
"DS$"the"DS$"right";:SCREENØ,1
49Ø DRAW"C8BM5Ø,5ØU2R3U1R3D1R3D2
L6":PAINT(53,49),8,8:DRAW"C6R6":
GET(5Ø,5Ø)-(56,46),T:PAINT(53,49
),7,7:GET(5Ø,5Ø)-(56,46),BS
5ØØ DRAW"C8BM5Ø,5ØU1R3U1R3D1R3D1
C6L8":GET(5Ø,5Ø)-(56,46),LT:PUT(
5Ø,5Ø)-(56,46),BS
51Ø FORX=1TO1Ø:CIRCLE(5Ø+X,49),3
,6,.9:CIRCLE(5Ø+X,5Ø),2,6,.9:NEX
TX:PSET(55,47,5):PSET(55,46,5):G
ET(5Ø,5Ø)-(6Ø,46),GS:PAINT(55,49
),7,7:PUT(G1,G2)-(G3,G4),GS
52Ø FORX=1TO15:PSET(RND(16),RND(
14),6):NEXTX:GET(Ø,14)-(16,Ø),SP
:DRAW"C7":LINE(Ø,14)-(3Ø,Ø),PSET
,BF:GET(Ø,14)-(16,Ø),SB
53Ø DRAW"C8":FORX=1TO1Ø:PSET(RND
(16),RND(1Ø),8):NEXTX:FORX=1TO5:
PSET(RND(16),RND(1Ø),6):NEXTX:GE
T(Ø,1Ø)-(16,Ø),DT:DRAW"C7":LINE(
Ø,1Ø)-(3Ø,Ø),PSET,BF
54Ø DRAW"C7":LINE(1Ø,15Ø)-(34,14
Ø),PSET,BF
55Ø CLSØ:PRINT@229,"ready"DS$"on
"DS$"the"DS$"firing"DS$"line";:S
CREENØ,1
56Ø PF=1:GOSUB115Ø
57Ø DRAW"C5BM14,148U6D6R4BR4":SC
=AF:GOSUB117Ø
58Ø DRAW"C5BM6,177U8D4R6U4D8BR8U
8L2R4L2D8L2R4BR1ØU8L4R8":DRAW"BM
215,165R4U4L4U4R4BD8BR4U8D4R6U4D
8BR4U8R6D8L4R4BR8U8L4R8BD22BL32U
5E5R1F5D1L9R9D4BR1ØU1ØL5R11"
59Ø DRAW"C5":LINE(12Ø,146)-(15Ø,
136),PSET,BF:DRAW"C6BM12Ø,144":S
C=DL:GOSUB117Ø
6ØØ DRAW"C7":LINE(219,15Ø)-(245,
14Ø),PSET,BF:DRAW"C5BM223,149U6F
4E4D6BR5"
61Ø IFDO=1THENDRAWN$(3):GOTO64Ø
62Ø IFIN=1THENDRAWN$(2):GOTO64Ø
63Ø DRAWN$(1)
64Ø IFM=2THENTC=1:PF=1:GOSUB115Ø
ELSEPAINT(17Ø,157),6,6:PAINT(45,
157),6,6
65Ø N1=Ø:N2=1:N3=126:N4=254:TC=Ø
66Ø PLAY"T1ØL1ØV31O3;12;11;1Ø;9;
8;7;6;5;4;3;2;1T255L255":SCREEN1
,1
67Ø IFTS=DL THENIFM=1THEN1Ø4ØELS
EIFT5=DL THEN1Ø4Ø
68Ø A$=INKEY$:D=JOYSTK(Ø):I=JOYS
TK(N1):J=JOYSTK(N2):P=PEEK(6528Ø
)
69Ø IFA$=CHR$(92)THEN1Ø6Ø
7ØØ IFA$="S"THENDRAW"C7":LINE(G1
+GØ,G4+1)-(G1+GØ,G4),PSET:DRAW"C
5":GØ=GØ+1:IFGØ>3THENGØ=-2:LINE(
G1+GØ,G4+1)-(G1+GØ,G4),PSET:GET(
G1,G2)-(G3,G4),GS ELSELINE(G1+GØ
,G4+1)-(G1+GØ,G4),PSET:GET(G1,G2
)-(G3,G4),GS
71Ø IFI<15THENIFG1>=16THENPUT(G1
,G2)-(G3,G4),BS:G1=G1-16:G3=G3-1
6
72Ø IFI>55THENIFG3<=243THENPUT(G
1,G2)-(G3,G4),BS:G1=G1+16:G3=G3+
16
73Ø IFJ<2ØTHENIFG4>34THENPUT(G1,
G2)-(G3,G4),BS:G2=G2-7:G4=G4-7
74Ø IFJ>4ØTHENIFG2<114THENPUT(G1
,G2)-(G3,G4),BS:G2=G2+7:G4=G4+7
75Ø PUT(G1,G2)-(G3,G4),GS
76Ø IFJ=63THENIFPF=ØANDDO=ØTHENP
F=1:GOSUB9ØØELSEIFPF=ØTHENPF=1:S
A=1:AT=1:GOSUB127Ø
77Ø IFP=N3 ORP=N4 THENIFSH=ØANDD
O=ØTHENPLAY"O1;1;2;3;4;1;2;3;4":
SH=1:GOSUB111ØELSEIFRA>ØTHENPLAY
"O1;1;2;3;4;1;2;3;4":RA=RA-1:GOS
UB138Ø
78Ø IFPF=ØTHEN68Ø
```

```
79Ø IFDO=1THEN84Ø
8ØØ PUT(T1,T2)-(T3,T4),BS
81Ø IFT1<=8ORT3>=244ORT4<34THENP
UT(G1,G2)-(G3,G4),BS:SH=Ø:GOTO1Ø
2Ø
82Ø T1=T1+AM:T3=T3+AM:T2=T2+UM:T
4=T4+UM:PUT(T1,T2)-(T3,T4),T
83Ø GOTO68Ø
84Ø PUT(118,T2)-(124,T4),BS
85Ø PUT(T1,T2)-(T3,T4),BS
86Ø IFT1<=8ORT3>=244ORT4<=34THEN
PUT(G1,G2)-(G3,G4),BS:GOTO146Ø
87Ø T1=T1+AM:T3=T3+AM:T2=T2+UM:T
4=T4+UM:IFAT=1THENPUT(T1,T2)-(T3
,T4),T
88Ø IFSA=1THENPUT(118,T2)-(124,T
4),T
89Ø GOTO68Ø
9ØØ N=RND(AF):ONN GOTO92Ø,91Ø,93
Ø,94Ø,95Ø,96Ø,97Ø,98Ø,99Ø
91Ø T1=1Ø2:T3=1Ø8:AM=-8:GOTO1ØØØ
92Ø T1=118:T3=124:AM=Ø:UM=-4:GOT
O1Ø1Ø
93Ø T1=13Ø:T3=136:AM=8:GOTO1ØØØ
94Ø T1=114:T3=12Ø:AM=8:GOTO1ØØØ
95Ø T1=118:T3=124:AM=-8:GOTO1ØØØ
96Ø T1=11Ø:T3=116:AM=-8:GOTO1ØØØ
97Ø T1=138:T3=144:AM=8:GOTO1ØØØ
98Ø T1=134:T3=14Ø:AM=12:GOTO1ØØØ
99Ø T1=98:T3=1Ø4:AM=-12
1ØØØ IFIN=1THENUM=RND(5)*-1
1Ø1Ø T2=114:T4=11Ø:PLAY"O3;1;2;3
;4":PUT(T1,T2)-(T3,T4),T:RETURN
1Ø2Ø PUT(T1,T2)-(T3,T4),BS:IFPPO
INT(T1,T2+4)=7THENT2=T2+4:T4=T4+
4:PUT(T1,T2)-(T3,T4),LT ELSEPF=Ø
:G1=118:G2=114:G3=128:G4=11Ø:PUT
(G1,G2)-(G3,G4),GS:Z=1:IFM=1ORTC
=ØTHENTS=TS+1:GOTO115ØELSET5=T5+
1:GOTO115Ø
1Ø3Ø GOTO1Ø2Ø
1Ø4Ø SCREEN1,Ø:PLAY"T1ØL1ØO2;1;2
;3;4;5;6;7;8;9;1Ø;11;12O3;1;2;3;
4;5;6":SCREEN1,1
1Ø5Ø A$=INKEY$
1Ø6Ø IFA$=CHR$(92)THENND=Ø:SC=Ø:
TS=Ø:HI=Ø:SH=Ø:CLSØ:HC=Ø:TC=Ø:IF
PF=1THENPUT(118,T2)-(124,T4),BS:
PUT(T1,T2)-(T3,T4),BS:GOTO13ØELS
EPF=1:GOTO13Ø
1Ø7Ø GOTO1Ø5Ø
1Ø8Ø IFT2<134THENT2=T2+2:T4=T4+2
:PUT(T1,T2)-(T3,T4),LT ELSEPF=Ø:
G1=118:G2=114:G3=128:G4=11Ø:PUT(
G1,G2)-(G3,G4),GS:Z=1:TS=TS+1:GO
TO115Ø
1Ø9Ø GOTO1Ø5Ø
11ØØ PUT(G1-5,G2-8)-(G3+1,G4-18)
,SB:PUT(G1,G2)-(G3,G4),BS:G1=118
:G2=114:G3=128:G4=11Ø:PUT(G1,G2)
-(G3,G4),GS:PUT(G1-5,G2-8)-(G3+1
,G4-18),SB:GOTO67Ø
111Ø PAINT(G1-6,G4-9),7,7:PAINT(
G1-6,G4-14),7,7:PAINT(G1-6,G2-8)
,7,7:PAINT(G1-6,G4-13),7,7:PUT(G
1-5,G2-5)-(G3+1,G4-15),SP:PUT(G1
-5,G2-5)-(G3+1,G4-15),SB
112Ø IFPF=1THENIFPPOINT(T1+1,T2-
1)<>7ANDPPOINT(T3-1,T2)<>7ORG4<T
2 THEN11ØØ
113Ø IFPF=1THENSH=Ø:PUT(T1,T2)-(
T3,T4),BS:PUT(G1-3,G2-8)-(G3+3,G
4-14),DT:IFM=2ANDTC=1THENH2=H2+1
:T5=T5+1:GOSUB115ØELSEHI=HI+1:TS
=TS+1:GOSUB115Ø
114Ø SH=Ø:GOTO11ØØ
115Ø DRAW"C7":IFM=2ANDTC=1THENLI
NE(164,161)-(2Ø4,151),PSET,BF:DR
AW"C8":LINE(215,191)-(255,181),P
SET,B:DRAW"C5BM166,159":SC=T5:N1
=Ø:N2=1:N3=126:N4=254:GOTO117Ø
116Ø SC=TS:LINE(215,191)-(255,18
1),PSET,BF:DRAW"C5BM217,189":IFM
=2THENN1=2:N2=3:N3=125:N4=253:DR
AW"C8":LINE(164,161)-(2Ø4,151),P
SET,B:DRAW"C5BM217,189"
117Ø S$=STR$(SC)
118Ø FORC=1TOLEN(S$)
119Ø R$=MID$(S$,C,1)
12ØØ N=ASC(R$):IFN>47THENDRAWN$(
N-48)ELSEDRAWSP$
121Ø DRAWSP$:NEXTC
122Ø IFZ=1THENZ=Ø:IFM=1THEN67ØEL
SEIFTC=ØTHENTC=1:GOTO67ØELSETC=Ø
:GOTO67Ø
123Ø IFM=2THENIFPF=1ANDTC=1THENP
F=Ø:DRAW"C7":LINE(4Ø,161)-(8Ø,15
1),PSET,BF:DRAW"C5BM42,159":SC=H
2:GOTO117Ø
124Ø IFPF=1THENPF=Ø:DRAW"C7":LIN
E(Ø,191)-(4Ø,181),PSET,BF:DRAW"C
5BM2,189":SC=HI:GOTO117Ø
125Ø IFM=2THENTC=TC+1:IFTC>1THEN
TC=Ø
126Ø UM=-4:RETURN
127Ø N=RND(AF):ONN GOTO128Ø,131Ø
,13ØØ,129Ø,132Ø,133Ø,134Ø,135Ø,1
36Ø
128Ø T1=13Ø:T3=136:AM=8:GOTO137Ø
129Ø T1=138:T3=144:AM=8:GOTO137Ø
13ØØ T1=94:T3=1ØØ:AM=-8:GOTO137Ø
131Ø T1=1Ø2:T3=1Ø8:AM=-8:GOTO137
Ø
132Ø T1=11Ø:T3=116:AM=-12:GOTO13
7Ø
133Ø T1=126:T3=132:AM=12:GOTO137
Ø
134Ø T1=11Ø:T3=116:AM=-12:UM=-6:
```

```
GOTO137Ø
135Ø T1=126:T3=132:AM=12:UM=-6:G
OTO137Ø
136Ø T1=138:T3=144:AM=12:UM=-6
137Ø T2=114:T4=11Ø:PLAY"O3;1;2;3
;4":PUT(T1,T2)-(T3,T4),T:PUT(118
,T2)-(124,T4),T:RETURN
138Ø PAINT(G1-6,G4-9),7,7:PAINT(
G1-6,G4-14),7,7:PAINT(G1-6,G2-8)
,7,7:PAINT(G1-6,G4-13),7,7:PUT(G
1-5,G2-5)-(G3+1,G4-15),SP:PUT(G1
-5,G2-5)-(G3+1,G4-15),SB
139Ø IFSA=1ANDG4>T2 THENIFPPOINT
(119,T2-1)=7ORPPOINT(123,T2)=7TH
ENPUT(118,T2)-(124,T4),BS:PUT(G1
-3,G2-8)-(G3+3,G4-14),DT:SA=Ø:HC
=HC+1:PUT(G1-3,G2-8)-(G3+3,G4-14
),SB:IFM=2ANDTC=1THENT5=T5+1:H2=
H2+1ELSEHI=HI+1:TS=TS+1
14ØØ IFAT=1ANDG4>T2 THENIFPPOINT
(T1+1,T2-1)=7ORPPOINT(T3-1,T2)=7
THENPUT(T1,T2)-(T3,T4),BS:PUT(G1
-3,G2-8)-(G3+3,G4-14),DT:AT=Ø:HC
=HC+1:PUT(G1-3,G2-8)-(G3+3,G4-14
),SB:IFM=2ANDTC=1THENH2=H2+1:T5=
T5+1ELSEHI=HI+1:TS=TS+1
141Ø IFRA=ØANDHC=2THENRA=2:HC=Ø:
GOSUB115Ø
142Ø IFPF=ØTHENRA=2
143Ø IFPF=ØTHENPUT(G1,G2)-(G3,G4
),BS:G1=118:G2=114:G3=128:G4=11Ø
:PUT(G1,G2)-(G3,G4),GS
144Ø IFTS=DL THENIFM=1THEN1Ø4ØEL
SEIFT5=DL THEN1Ø4Ø
145Ø RETURN
146Ø PUT(T1,T2)-(T3,T4),BS:IFSA=
1THENPUT(118,T2)-(124,T4),BS
147Ø T2=T2+4:T4=T4+4:IFT2>=136TH
EN152Ø
148Ø IFSA=1THENIFT2>=116THENSA=Ø
:IFM=1ORTC=ØTHENTS=TS+1ELSET5=T5
+1
149Ø IFAT=1THENPUT(T1,T2)-(T3,T4
),LT
15ØØ IFSA=1THENPUT(118,T2)-(124,
T4),LT
151Ø GOTO146Ø
152Ø IFAT=1THENAT=Ø:IFM=1ORTC=ØT
HENTS=TS+1ELSET5=T5+1
153Ø IFHC=ØTHENZ=1:PF=Ø
154Ø G1=118:G2=114:G3=128:G4=11Ø
:PUT(G1,G2)-(G3,G4),GS:HC=Ø:RA=2
155Ø GOSUB115Ø
156Ø GOTO67Ø
157Ø POKE65494,Ø:CLS:PRINT@6,"US
ER ENDED PROGRAM.":END
```

A Diversion A Day Keeps Programming Fey

By Joseph Kolar

Today, the beginner will be subjected to a variety of programming tidbits. In the process, he or she will develop a program though it won't perform any useful function. Any program, useless or not, may be used as a vehicle for sharpening programming skills. It should be interesting and fun, and when finished, you should marvel at your classy accomplishment.

Here is the rationale for the current project: Consider some of the words that evolve into our language with abbreviations, contractions and acronyms — better known as jargon. Our computer hobby is overwhelmed with "jargonese." The beginner can become confused when reading, for instance, a computer ad, and blanches at the plethora of jargon that doesn't give a clue as to what the product claims to do.

Why not devise a program to create artificially produced words, or "pseudo-words," ones that look and sound like actual words? These won't be in any dictionary!

Let's use Latin as our source. Three categories will be required: prefixes, roots and suffixes. We will combine them to originate one to three syllable words. Since the raw material of our project is real components of Latin-derived words, some real words will be generated. Amongst the dross of lackluster real words, we will invent some plausible nonsense words.

Every program line in Listing 1 has a REM marker in front of it. Since it is always more instructive to create a program and watch it take on substance, you should key it in and worry about correcting typing errors as we unveil the lines. On running, you should get nothing on the screen except "OK" if it has no errors. LIST it and check to make sure each program line begins with a REM marker.

Now, we get to work! Every time you remove a REM marker (') from a line, pretend that you have just typed it in and that all the REM marker lines are invisible. Remove the REM marker from Line 100, EDIT100 and position the cursor under the REM marker. Next, press 'D' and ENTER. This is the first line we've prepared. We are going to put a list of prefixes into a DATA statement. There are 20 common prefixes in this array.

Unmask Line 101. We gleaned 50 Latin roots. For the purists, we usurped a bit of literary license with NOMIN reduced to NOM and LIBER shortened to LIB.

Uncover Line 102. We managed to scrounge up 33 suffixes. Now our raw material has been assembled.

We will store these word elements in three arrays.

CoCo automatically allocates 11 slots, from zero to 10. In order to reserve more space, we must tell CoCo in a DIM statement what our requirements are.

Uncover Line 10. The array, A$, contains the 20 prefixes, B$ contains the 50 Latin roots and C$ houses the 33 suffixes.

We began constructing our program at Line 100. Why not Line 10? Lots of empty lines are left to add to and work around program lines that we create as we progress. Remember, you can't always know where or how your idea will develop. It is important that you do not renumber or disturb your program lines. Your data is in lines 100-102 and will remain there until the program is completed. If you keep relocating the program lines, you tend to lose track of them. Rather than mentally recalling the line number, you must seek it out by scrolling through the listing.

Now that we have reserved array space, we must put our data into the proper files. Unmask Line 200. Using a FOR/NEXT loop, we take every item in Line 100, calling each prefix, 'A', and enter it in an assigned location in the prefix A$ array. Thus A$(1) contains AB, A$(2) contains RE, A$(3) holds EX, etc. Each prefix has a specially defined home.

Uncover Line 201: here we place the 50 roots, 'B'.

Unmask Line 202. The suffixes, 'C', are assigned a home. All the word particles are now in CoCo's memory in arrays when the program in RUN. Press BREAK and take a few samplings, for example, type PRINT A$(1);B$(50);C$(2) — you should get ABMOVATE. The three word elements have produced a nonsense word!

We must tell CoCo we want random particles and allow it to select them. Uncover Line 250. The 20 prefixes designated with the variable 'A', the 50 roots assigned as 'B' and the 33 suffixes called by 'C' are placed in three RND statements, one for each group.

We must figure out where to locate our randomly selected word particles. Uncover Line 300. Using PRINTTAB, we decide to start our column of words 10 spaces from the left margin. We will print, starting at that location, a randomly selected prefix. We end the statement with a semicolon (;) because we want the root tacked at the end of the prefix.

Unmask lines 301 and 303. In Line 301 we determine which root is used and in Line 303 we choose the suffix. When we RUN, we get a three-syllable word.

Run the program a few times. It is a pain to generate just one psuedo-word. Line 303 directs CoCo to go to Line 315. We plan to make 10-word lists to study the words we create.

Unmask Line 315. We set up a counting system to keep track of each time a word is printed (E=E+1), and until 10 words have been printed, to go back to Line 250 and keep knocking out words. When the tenth word (E=10) is created, CoCo is sent to Line 316.

Unveil Line 316. We print a pair of blank lines. Uncover Line 317. This command holds the panting CoCo from spewing out more words until you are finished looking at the list. This is done by pressing any key. Since we want to create another list of 10 words, we reset the counter to zero (E=0).

Uncover lines 1 and 319. We clear the screen to black in both cases. Then, in Line 319, we loop back to Line 250 to generate a new sampling of psuedo-words. RUN and keep tapping any key to check out what you have wrought.

This produces a nice variety of words, but they are all three-syllable words. Quite boring! Why not make some with two syllables, the prefix and the root? While we are at it, why not make a sampling of two-syllable words containing the root and suffix?

To set up this system, unmask Line 260. The variable 'D' asks for three random alternatives. If D=1, we set up a three-element word. If D=2, we go to another location to create a word consisting of only the prefix and root. If D=3, we go to a third location to create a word consisting of a root and suffix.

Uncover lines 305, 306 and 308. Line 305 prints the prefix, Line 306 prints the root and Line 308 sends CoCo on its way to count the word. Uncover lines 310 and 314. These two lines print the root and suffix words and move on to tally the number of words.

Now that we have a working program, unmask lines 6-9. Substitute your name and address in the appropriate places and adjust the PRINTTAB locations to center the title. You will be given credit for the program for a brief moment, then the first set of words is created. Line 316, already unveiled, pushes the title up and out of the display. Put the REM marker temporarily back in Line 316 to verify this.

Now that we have personalized the listing, uncover Line 5 and RUN. Note how neatly the text lines are placed on the display. It looks downright professional. Press BREAK and LIST. The lines will be scrolled on slowly and allow you to look at the program lines.

If you must modify, improve or change the program, you can POKE 359,126 to get back to the normal listing. This will eliminate the smooth text placement, but you can't have everything. This completes the listing.

All sorts of work is required to polish this program. It needs to be individualized so it is truly your own work. Note that no real title card is created — no instructions, no ending, no nothing.

Here is the rule. First, create a viable core program. Then and only then, in the following order, insert sound and graphics embellishments, if any. Next create the end panel, the instructions and the front title card, in that order. (Adding sound slows down the execution of a program and is a bother to listen to as you make repeated trial runs.)

Listing 2 is a collection of three program lines you may want to key into your finished program. They were pulled out of the original listing because the RIGHT$ function will be covered and explained in an upcoming article. These lines will add a bit more pizazz to the phony word list.

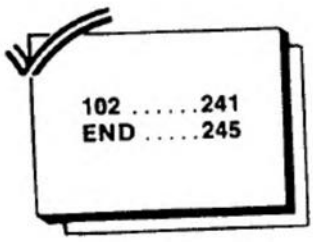
Line 302 determines in a three-syllable word if a root ends in 'T', then C$(1) (TION) becomes ION, eliminating the double TT. For instance, REJECTTION becomes REJECTION. If the root word does not end in 'T', then TION becomes SION.

Line 307 determines in a prefix-root word if the root ends in 'T', then 'S' is added. If not, an 'E' is added.

Line 311 determines in a root-suffix word if the root ends in 'T' then TION becomes SION. If not, TION becomes ION.

You may not like some of these lines, so don't use them. You many want to experiment where the root ends in SS and is followed by TION, expand your word lists or use Greek word elements. See what a neat batch of psuedo-words you can concoct.

Let me leave you with the thought that no matter how foolish an idea may be, it is easy to get caught up in developing it into a fun program. If you enjoyed working on this diversion, you are on your way to programmer's heaven. □



Listing 1: BASTRAN1

```
Ø '<LISTING1>
1 'CLSØ
5 'POKE359,6Ø
6 'PRINT:PRINT:PRINTTAB(1Ø)"JOSE
PH KOLAR"
7 'PRINTTAB(6)"17Ø9 DICKINSON ST
REET"
8 'PRINTTAB(8)"INVERNESS, FLORID
A":PRINT
9 'PRINT:PRINT:FOR Z=1TO5ØØ:NEXT
1Ø 'DIM A$(2Ø),B$(5Ø),C$(33)
1ØØ 'DATA AB,RE,EX,PRO,CON,DE,IN
,SUB,PER,PRE,AD,CIRCUM,DIS,INTER
,OB,TRANS,ANTI,ANTE,AN,UN
1Ø1 'DATA DUCT,FER,JECT,SCRIP,PO
N,MIT,PORT,CEPT,VECT,CUR,CAP,CIP
,CED,CUP,DICT,FACT,FEC,FIC,GRAT,
LOC,MEND,MENT,MULT,NOV,PELL,PED,
PRESS,PUG,SERV,SPECT,SPEC,SPIC,S
TRUCT,STRUD,VEN,VENT,VICT,VIN,TU
RN,PLEX,GRESS,PULS,LUC,OMN,DOC,M
ORT,NAV,NOM,LIB,MOV
1Ø2 'DATA TION,ATE,ENT,ANCE,OUS,
ABLE,AL,ER,IFY,ENCE,ITY,TUDE,OR,
IAC,ACIOUS,ACY,AGE,ANCY,ARY,ITIO
US,IVITY,ERY,ESCENT,FEROUS,IC,IN
E,ISM,IST,IVE,ORY,ULENT,ISE,IZE
2ØØ 'FOR A=1 TO 2Ø:READ A$(A):NE
XT
2Ø1 'FOR B=1 TO5Ø:READ B$(B):NEX
T
2Ø2 'FOR C=1 TO 33:READ C$(C):NE
XT
25Ø 'A=RND(2Ø):B=RND(5Ø):C=RND(3
3)
26Ø 'D=RND(3):ON D GOTO3ØØ,3Ø5,3
1Ø
3ØØ 'PRINTTAB(1Ø)A$(RND(A));
3Ø1 'PRINTB$(RND(B));
3Ø3 'PRINTC$(RND(C)):GOTO315
```

```
305 'PRINTTAB(10)A$(RND(A));
306 'PRINTB$(RND(B))
308 'GOTO315
310 'PRINTTAB(10)B$(RND(B));
314 'PRINTC$(RND(C))
315 'E=E+1:IF E=10 THEN GOTO316
ELSE250
316 'PRINT:PRINT
317 'EXEC44539:E=0
319 'CLS0:PRINT:PRINT:GOTO250
```

Listing 2: BASTRAN2

```
0 'LISTING2  ADDITIONAL LINES FO
R LISTING1
302 IF RIGHT$(B$(B),1)="T" THEN
C$(1)="ION" ELSE C$(1)="SION"
307 IF RIGHT$(B$(B),1)="T" THEN
C$(C)="S"ELSE C$(C)="E":GOTO315
311 IF RIGHT$(B$(B),1)="T" THEN
C$(1)="SION" ELSE C$(1)="ION"
```

Listing 3: BASTRAN3

```
0 '<POKE178>
5 CLEAR500
10 PMODE4,1:PCLS2:SCREEN1,1
20 O$="NR10U20R10D20BR6"'O
21 R$="U20R10D10L10F10BR6"'R
22 J$="R10NU20BR6"'J
23 E$="U10NR8U10R10BD20NL10BR6"
24 K$="U10NU10NE10F10BR6"
25 L$="NU20R10BR6"
26 A$="U20R10D10NL10D10BR6"
27 T$="BR5U20NL5R5BD20BR6"
28 I$="NU20BR6"
29 H$="U10NU10R12NU10D10BR6"
30 W$="NU20R6NU16R6NU20BR6"
31 N$="U15NU5F10NU15D5BR6"
32 B$="U20R8D10NL8R2D10NL10BR6"
40 FOR X=RND(8) TO RND(26)
41 TT$=T$+"BR4"+H$+"BR4"+E$
42 UU$=R$+A$+I$+N$
43 VV$=B$+O$+W$
50 POKE178,X
51 DRAW"S4BM102,65"+TT$
52 DRAW"S8BM27,135"+UU$:DRAWVV$
53 DRAW"S4BM103,66"+TT$
54 DRAW"S8BM28,136"+UU$:DRAWVV$
55 DRAW"S4BM104,67"+TT$
56 DRAW"S8BM29,137"+UU$:DRAWVV$
57 DRAW"S4BM105,67"+TT$
58 DRAW"S8BM30,137"+UU$:DRAWVV$
59 NEXT:GOTO40
```

SOUND TUTORIAL | **16K ECB** | the RAINBOW

An Adventure Into Sound Experimentation

By Bill Bernico

One of the things a lot of programs overlook is also one of the things the CoCo is very good at ... producing sounds. With the help of SOUND, PLAY and even EXEC commands, you can produce some very impressive sounds to help showcase your programs and make them more interesting.

The SOUND statement is the simplest. A combination of SOUND 191,1:SOUND 220,2:SOUND 10,5, for example,will produce three separate notes. A shorter, easier way is to use the PLAY command. With it, simply tell the computer to PLAY "CFG" and you will also get three separate notes.

As for EXEC, I had to experiment with different combinations to come up with a sound. For example, in lines 530 and 550, I used it to create the "helicopter" and "raspberry" sounds. EXEC 43345 by itself will produce a short "click" or "blip" sound. Enclose it in a loop, as in Line 550, and you will get the raspberry sound.

In Line 530 you'll probably wonder why there is a CIRCLE command when there are no graphics. CIRCLE is there only to create a delay between EXEC's so there won't be one continous sound, rather a broken sound such as a helicopter taking off. You can substitute a short FOR/NEXT loop if you don't have Extended BASIC.

Running the Program

Sound Story could have simply been a menu with different sounds to pick from, but careful analysis revealed that the attention span of the program would be about 83 seconds. With that in mind, I decided to integrate these sounds into a short (and I do mean short) Adventure.

There are no treasures to collect, no damsels in distress to save and no dugeons to crawl out of. When you come across the house in the woods, just pick an option from the list provided. Each choice will demonstrate a different sound.

Tandy ELECTRONICS

BUDGET-PRICED COLOR COMPUTERS

- *Create Color Graphics with Sound and Music*
- *Typewriter-Quality Keyboard*
- *Attaches to Any TV*

Save $60

Reg 239.95

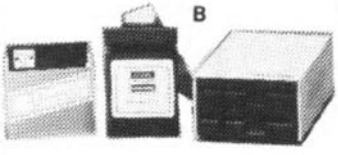
179.95

16K Standard BASIC Color Computer 2. The easiest way to begin learning about computing! Simply plug into your TV, snap in a Program Pak™ cartridge from Tandy's huge range of entertainment and educational software for hours of fun. Expand with printer, disk drive, modem, more. Includes manual. 26-3134

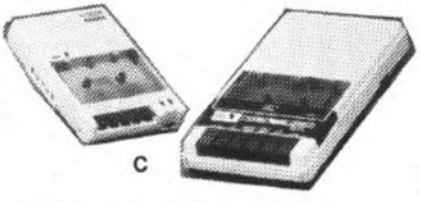
- **16K EXTENDED** Reg 299.95 Sale **229.95**
- **64K EXTENDED** Reg 499.95 Sale **299.95**

Exciting Color Computer Accessories & Programs

A

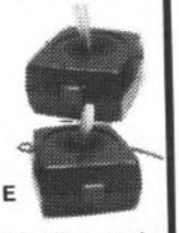


B



C

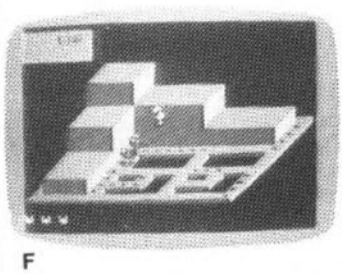
D



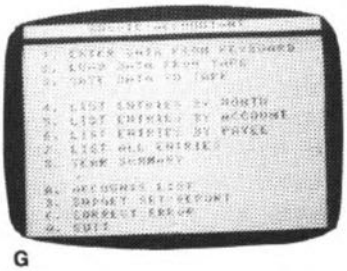
E

A. Electronic Book. Makes learning a lot of fun for kids! Features touch-sensitive "pages" for answering questions. Requires joystick. 26-3141 ..... **29.95**
Maze Master Basic computer use. 26-2541 ..... **39.95**
Shape Maker Identify shapes. 26-2542 ..... **39.95**
The Number Factory Math games. 26-2543 ..... **39.95**
Word Wizard Test your spelling. 26-2544 ..... **39.95**
B. FD-501 Color Thinline Disk Drive #0. 156K memory. 13.3cm double-density 35-track floppy disk drive with DOS Program Pak, cable, blank disk, manual. 26-3129 . **599.00**

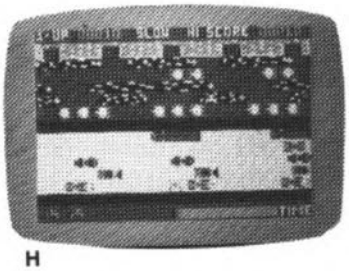
C. Computer Cassette Recorders. Ideal for recording and loading programs on tape. **CCR81.** Auto stop, cue/review. 26-1208 ..... **79.95**
CCR82. Sound monitor, pause. 26-1209 ..... **69.95**
D. Deluxe Joystick. Patented stick mechanism provides fast and accurate cursor control. Features dual-axis fine-tuning controls and choice of "spring-centering" or 'free-floating' operation. 26-3012 ..... **39.95**
E. Joysticks. Quick-response action for games, graphics or business software! Single-shot button. 26-3008 ... **Pr/29.95**



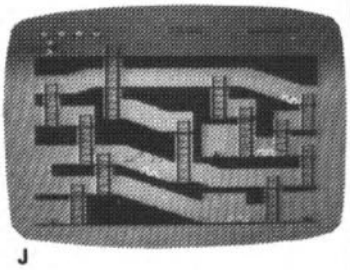
F



G



H



J

F. Ice Castles. Collect gems from castles! Beware of Gem Eaters and Berthilda the Witch. 26-7358 ..... **29.95**
G. Aussie Accountant. Records credits and debits! Keeps accounts up to date, monthly balances. 26-7962 ..... **39.95**

H. Frogger. Frvourite arcade game! Guide frog safely over the busy highway and dangerous river. 26-7304 ..... **29.95**
J. King Tut. Hidden treasure in a tomb! Beware of snakes and the King's ghost. 26-7313 ..... **29.95**

EXCITING!! COLOR COMPUTER PACKS

16K Starter Pack

Save $106.80

Reg **386.75**

279.95

Your pack includes: 16K Standard BASIC Color Computer 2. A low-cost introduction to the exciting world of computers! Do color graphics, play games. 26-3134
Cassette Recorder. Store programs and games on tape! 26-1209
Joysticks. Fast action! 26-3008
Rom Pak™ Game. Arcade fun!
Illustrated Computer Dictionary. Learn the language! 26-9605

16K Extended Home Educator Pack

Save $126.70

Reg **506.65**

379.95

You'll find in this pack: **16K Extended BASIC Color Computer 2.** Learn with education software, do your budget, draw graphics! 26-3136
Cassette Recorder. 26-1208
Joysticks. 360° action! 26-3008
'AustQuiz' Brain teaser! 26-7960
ROM Pak™ Game. Adventure!
'Aussie Accountant'. Money manager! Easy budgeting. 26-7962
Illust. Computer Dictionary. 26-9605

64K Extended BASIC Family Learning LAB

Save $246.65

Reg **696.60**

449.95

Your complete system consists of: **64K Extended BASIC Color Computer 2.** Business or advanced home use! Graphics, animation. 26-3127
Cassette Recorder. 26-1208
Joysticks. High speed fun 26-3008
'AustQuiz' Local knowledge! 26-7960
ROM Pak™ Game. Skill!
'Aussie Accountant'. Keep track of your finances! 26-7962
Illust. Computer Dictionary. 26-9605

WE SERVICE WHAT WE SELL!

Tandy ELECTRONICS

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

Nearly 350 Stores Australia-Wide

Available from 350 Stores Australiawide including Tandy Computer Centres

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

CORRECTION — " Destination: Moon Base Amphibia "

"Destination: Moon Base Amphibia" (November '85, Page 41): The printed listing on pages 43 to 46 of our November issue has a number of incomplete lines. The reason is these lines were packed with additional statements to such an extent that they did not list properly, and the incomplete printout was not discovered in time to correct. We regret any inconvenience and frustration this editorial oversight may have caused those readers who typed in this program.

What happened? BASIC stores each command word (GOTO, PRINT, FOR and the like) as a single-byte "token"; when you list a program to the screen or the printer, edit a line or save the program in ASCII (on tape or disk), a special routine puts the words back in place of the tokens. If an extremely long line has been entered (either by using the EDIT mode to cram them in or by using a special packing program) it's possible for the de-tokenizing routine to run out of buffer space and just drop off everything after the first 248 characters.

If you have the November RAINBOW ON TAPE, please note that the copy of *Amphibia* on the tape *does* run properly, but if you try to LIST or LLIST it you will have the same problem that we did for the same reason. The commands are there, but you just can't see them or print them out.

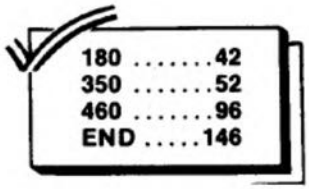
To solve the problems that the packed lines cause in entering the program by hand, follow these instructions. Using the EDIT command, change the ends of the following lines in Listing 1:

20 - Delete : (just the final colon)
910 - Delete : (just the final colon)
1000 - Leave as is (Although commands are missing, they will be inserted in a separate line.)
1016 - Delete :PAINT(214
1100 - Leave as is
1200 - Delete :IFY<72THENY=7
1218 - Delete :PAINT(116,5
1300 - Delete everything from :D1$="U4 (on through end)
3000 - Delete :GET(20,90) - (46,105),A1,
3010 - Delete :PAINT(40,55),,
3028 - Delete :LIN
3052 - Delete :LINE(
3100 - Delete :FORX=120TO130ST
3142 - Delete :PAINT(95
3162 - Delete :PM
3220 - Delete :P
3246 - Delete :PMODE3
3300 - Delete :PAINT(100,
3316 - Delete :DRAW"BM"+
6905 - Delete :GOSU
7905 - Delete :FO
9500 - Leave as is
9900 - Delete :PRINT@227,"PRESS <ENT

Now key in the following new lines:.

```
21 PRINT@388,"PRESS <ENTER> TO B
EGIN";

911 BG=5254:BX=5

1001 PSET(X1,Y+1):X1=X1-1:Y=Y-2+
RND(3):IFY<80THENY=80ELSEIFY>96T
HENY=96

1017 PAINT(214,60),,1:POKE178,1:
PAINT(0,70),,1:RETURN

1101 COLOR1,0:LINE(0,170)-(255,1
92),PRESET,BF:LINE(0,170)-(255,1
92),PSET,B

1201 IFY<72THENY=73ELSEIFY>88THE
NY=87

1219 PAINT(116,52),,1:PAINT(216,
49),,1

1301 D1$="U4E2R1U4E1U2H1L1H1U1H1
U1E1U1E1R1E1U1E1D1F1E1U1F1D1F1R1
F1D1F1D1G1D1G1L1G1D2F1D4R1F2D4L1
2":DRAW"C1S16BM156,160;XD1$;"

3001 GET(20,90)-(46,105),A1,G:CO
LOR1,0:LINE(20,90)-(46,105),PRES
ET,BF

3011 PAINT(40,55),,1:COLOR1,0:CI
RCLE(20,58),4,1:CIRCLE(32,58),4,
1:CIRCLE(52,58),4,1

3029 LINE(102,19)-(106,24),PRESE
T:PMODE4,5:COLOR1,0

3053 LINE(X,Y)-(X+8,Y),PSET:X=X-
2:NEXTY:COLOR2,1:FORX=218TO234ST
EP4:PSET(X,17):NEXTX

3101 FORX=120TO130STEP4:PSET(X,4
7):PSET(X,48):NEXTX:D2$="E2R1F2G
2L1H2"

3143 PAINT(95,34),,1:PMODE3,5:CO
LOR2,1:LINE(94,33)-(256,64),PSET
,B

3163 PMODE3,5:COLOR3,1:LINE(12,1
02)-(64,102),PSET

3221 PMODE4,5:POKE178,1:PAINT(11
0,79),,1:PMODE3,5:DRAW"S16BM102,
90;XD2$;"

3247 PMODE3,5:DRAW"C3S12BM24,120
;XD4$;":PAINT(28,116),3,3

3301 PAINT(100,41),,1:COLOR1,0

3317 DRAW"BM"+STR$(X)+",48;XD1$;
BM"+STR$(X)+",68;XD2$;":NEXTX:LI
NE(129,39)-(255,77),PRESET,BF:RE
TURN

6906 GOSUB6300:GOTO7000

7906 FORI=1TO30:PRESET(RND(255),
161+RND(7)):NEXTI:GOSUB6500

9501 PRINT:PRINT"BONUS 2000!!!":
S1=S1+2000:FORI=1TO800:NEXTI

9901 PRINT@227,"PRESS <ENTER> TO
 CONTINUE"
```

The REM statements are highlighted in the listing so that you might examine how each sound is created. I came across a lot of these sounds by accident, by trial and error, and by experimentation. I keep a one-word reminder taped over my CoCo at all times - Experiment!



The listing: SNDSTORY

```
1Ø 'SOUND STORY
2Ø 'BY BILL BERNICO
3Ø '7Ø8 MICHIGAN AVE.
4Ø 'SHEBOYGAN, WI 53Ø81
5Ø '(414) 459-735Ø
6Ø '
7Ø '*** TITLE PAGE SOUND ***
8Ø CLS3:PRINT@43,"SOUND STORY";
9Ø FORX=1TO255STEP1Ø:SOUNDX,1:NE
XTX
1ØØ FORX=1TO5ØØ:NEXTX
11Ø '*** AUTHOR'S NAME SOUND ***
12Ø PRINT@2Ø1,"BY ";:SOUND22Ø,2:
FORX=1TO15Ø:NEXT:PRINT"BILL ";:S
OUND11Ø,2:FORX=1TO15Ø:NEXT:PRINT
"BERNICO";:SOUND1Ø,2
13Ø GOSUB56Ø
14Ø CLS:PRINT"IMAGINE YOURSELF W
ALKING THROUGHTHE WOODS.  YOU CO
ME ACROSS A    HOUSE SET BACK IN
THE TREES.     AS YOU APPROACH, Y
OU SEE THE     DOOR TO THE HOUSE.
15Ø GOSUB56Ø
16Ø CLS:PRINT"ON THE DOOR YOU NO
TICE THERE IS A KNOB, A DOORBELL
, A KNOCKER    AND A SMALL PANE O
F GLASS IN THECENTER.  THERE DOE
SN'T APPEAR TOBE ANYBODY HOME.
17Ø GOSUB56Ø
18Ø CLS:PRINT@4Ø,"WHAT WILL YOU
DO?                ----------
---
19Ø PRINT@1ØØ,"1.) KNOCK ON THE
DOOR
2ØØ PRINT@132,"2.) RING THE DOOR
BELL
21Ø PRINT@164,"3.) TRY THE DOORK
NOB
22Ø PRINT@196,"4.) LOOK IN THE W
INDOW
23Ø PRINT@228,"5.) GO AROUND THE
 BACK
24Ø PRINT@26Ø,"6.) ESCAPE IN HEL
ICOPTER
25Ø PRINT@292,"7.) GIVE UP
26Ø PRINT@36Ø,"CHOOSE (1-7)
27Ø A$=INKEY$:IFA$=""THEN27Ø
28Ø A=VAL(A$)
29Ø IF A<1 OR A>7 THEN 26Ø
3ØØ ON A GOTO 33Ø,37Ø,41Ø,43Ø,46
Ø,53Ø,55Ø
31Ø GOTO 31Ø
32Ø '*** DOOR KNOCK SOUND ***
33Ø CLS:FORX=1TO3:PLAY"T15ØO1V31
CDCGABV25CV2ØCV15DV1ØDV5EV1E":PR
INT@133,"KNOCK!  KNOCK!  KNOCK!"
:NEXTX
34Ø PRINT@33Ø,"NOBODY HOME":GOSU
B58Ø:GOTO18Ø
35Ø GOSUB 58Ø
36Ø '*** DOORBELL SOUND ***
37Ø CLS:PRINT@135,"RRRRRRRRRRRRI
NG!!":PLAY"L2ØV3ØT5ØO5GBGBGBGBGB
GBGBGBGBGBGBGBGBGBGBGBGBGBGBGBG
38Ø PRINT@33Ø,"NOBODY HOME":GOSU
B58Ø:GOTO18Ø
39Ø GOSUB 58Ø
4ØØ '*** BURGULAR ALARM ***
41Ø CLS:PRINT@162,"WARNING!  WAR
NING!  WARNING!":PRINT@224,"(FOR
GOT ABOUT THE BURGLAR ALARM)":Q$
="1;2;3;4;5;6;7;8;9;1Ø;11;12;":F
ORX=1TO15:PLAY"O5L242V15;":PLAY
Q$:NEXTX
42Ø GOSUB58Ø:GOTO18Ø
43Ø CLS:PRINT@195,"YOU SEE A BIR
D CAGE ON A          STAND.  THERE
'S A SMALL           BIRD IN IT. H
E JUST SAW YOU."
44Ø '*** BIRD CHIRP SOUND ***
45Ø FORX=1TO5:PLAY"L255T255V2Ø05
CGCEC":FORY=1TO2ØØ:NEXTY,X:GOSUB
58Ø:GOTO18Ø
46Ø CLS:PRINT@1ØØ,"YOU'RE IN THE
 BACK YARD.          YOU SEE THE O
WNER OF THE          HOUSE WITH A
MACHINE GUN!               BETTER R
UN....
47Ø GOSUB58Ø
48Ø '***MACHINE GUN SOUND ***
49Ø FORF=1TO3
5ØØ FORX=1TO1Ø:PLAY"L19ØO1V28CV2
9DV2ØEV1ØF#V21G#V16GV31":NEXTX:F
ORD=1TO5ØØ:NEXTD
51Ø NEXTF:GOTO18Ø
52Ø '***HELICOPTER SOUND ***
53Ø CLS:FORF=449TO42STEP-31:PRIN
T@F,"UP, UP AND AWAY":FORY=1TO3:
CIRCLE(1Ø,1Ø),4:EXEC43345:PRINT@
F,"":NEXTY,F:GOSUB58Ø:GOTO 18Ø
54Ø '*** RASPBERRY SOUND ***
55Ø CLS:FORX=1TO1ØØ:EXEC43345:NE
XTX:PRINT"MY, WHAT A SHORT ADVEN
TURE":END
56Ø PRINT@484,"HIT ANY KEY TO CO
NTINUE";:EXEC44539:RETURN
57Ø GOTO 57Ø
58Ø FORX=1TO2ØØØ:NEXTX:RETURN
```

ASSEMBLY FILE

BY KEVIN

In itself this months program is no big deal but it does serve to illustrate how we can put our knowledge of the Color Computers memory map to use.

Firstly lets decide what we want to do with our program. I'm not entirely satisfied with the way Tandy have decided to set up my CoCo's variables on power up so firstly what I want my program to do is to reset these variables to suit my usage. However once these variables have been reset I have no desire to have the ML. routine clogging up valuable memory space after the variables have been set so I must not locate this part of the program where it will lock itself into memory. The second thing I want to be able to do is to install a few of my own routines and again have them consume a minimum ammount of my valuable user RAM.

So lets look at our first requirement. For purposes of explaining the routine, we will only change the printer baud rate variable to enable me to run my printer at 1200 baud instead of 600. From BASIC all that is required to effect this change is the simple statement POKE150,41. In itself hardly worthwhile setting up a pogram for that but I'm lazy and if I can get CoCo to do that for me along with a few other things simply by typing one line:

```
LOADM"0":EXEC
```

then I reckon things are great. Of course, if we need to type a CLEAR command in before our LOADM then thats just so much more work but unless you have 64K and can relocate your permanent routines into the upper 32K of RAM then there is not much we can do about that. Or is there.

As far as I can tell there is a very small block of unused RAM just below the beginning of the text screen so perhaps we can use that, providing our extra routines are short enough.

What shall I use to demonstrate the way in which can hook our own routines into BASIC. Why not somthing a little different again. The ASCII Code loaded into the CPU's A Register when both the shift and up-arrow key are pressed simultaneously is $5F. Once our program has been EXECuted a Cold Start will be performed if you press these keys.

Now, let me explain how the program works. I have chosen to ORG the program at $2000 simply because that is slap bang in the middle of everything and will no doubt be overwritten once you start writing your BASIC program.

I have chosen SETUP as my first label merely to identify the start of the program and to give some indication of the purpose of this block of the program. Lines 110 & 120 simply set the baud rate variable at address 150 to the value 41, which value sets the CoCo's output baud rate to 1200.

If you examine the memory map in last months magazine you will find addresses 167-169 referred to as the character output hook. What happens is that BASIC before outputting a character to an external device, eg.the screen, or printer, it will JMP to address $167. On power up BASIC sets the value of these VECTORS according to the version of ROM fitted. The first byte of the three bytes which make up one of these vectors (or ROM Hooks) always contains the value $7E which translates to the opcode for the JMP instruction. The two bytes following contain the address to which the JMP is to be made, this usually being another address in ROM the actual address being of no real concern to us (unless you want to start following the actual workings of your ROM- HEAVY GOING!!). What is inportant is that we can put our own address into these locations and thus redirect CoCo to our own program. BUT!!! We MUST build in a means of returning control to the ROMS and I will now show you how this is done.

Still within the SETUP portion of the program Line 130 loads the X Register with the contents of address 168 & 169 (remember the X Register is a two byte register.) You should now be able to understand that the X Register now contains the address the ROM character output routine would have been redirected to if we had left it alone.

To get a little ahead of ourselves we now go to line 250 and find we have reserved two bytes for our own use. It would perhaps have been better programing style to use the Pseudo-op RMB 02 but FDB 00 does work and just happened to be the first form of code that came to mind at the time I wrote the program. So now we look at line 140. This tells us that we want to store the contents of the X Register in the two bytes we reserved in line 250. the use of ,PCR addressing merely ensures that we obey one of the first rules of 6809 programming and use Postion Independant Code whenever possible. You will later see that it is important to store the values we found at $168-$169 within the body of the main part of the program. Apart from the fact that we don't want to lose these values when BASIC overwrites the SETUP block of the program.

We can now go about putting the address of the start of our own routine into address 168-169. But first we must know where our program actually starts. The easiest way to do this is shown in line 150. Load the Effective Address of START into the X Register using PCR (position independent) addressing. When the assembler program encounters line 150 it will look for the label START, determine its address (the Effective Address) and cause that address to be loaded into the X Register when the Machine Language program is EXECuted. Now that we have the address of the start of our own routine in the X Register all that remains is to store it at $168-$169 using the STX instruction.

And in the flash of an eye we are returned to BASIC with the Return To Stack instruction.

Well that's all very jolly. We have redirected BASIC to our routine and ensured we know where BASIC originally wanted to go. Not forgetting that we have also increased the baud rate of the printer.

Lets now take a look at the main, or permanent part of our program. Line 180 uses a second ORG statement to enable us to put the main routine out of harms way, away from BASIC's touch. I've chosen $3F0 because, as I stated earlier our routine is short enough consuming only 14 bytes and will fit in the small gap in System RAM just below the beginning of BASIC's Text Screen Memory, which starts at $0400. Failing being able to fit the routine here I would have had to use either $3FF0 in 16K or $7FF0 in a 32K machine. Both of these locations would have demanded that the CLEAR command be used to protect the program from BASIC.

Whenever a key is pressed under BASIC it's appropriate character is displayed on the screen. So if we intercept the character output routine, as we have done above, we can now compare the contents of the A Register, which is used by BASIC to store the character it is outputing. (Incidently, I'm not sure but as far as I can tell BASIC generally uses the A register to store its variables such ASCII Codes from the keyboard etc.) In our case if the ASCII code $5F is found in the A Register CoCo will carry out lines 210-230 (POKE113,0:EXEC40999) and cause a Cold Start. Otherwise it will follow the relative branch to line 240 where it finds the value $7E which just happens to be the opcode for the JMP instruction. Where do we JMP to? Why to the address stored in the locations reserved by line 250 of course. After all they are the values immediately following the JMP instruction are they not.

Just one final caution before I finish. Within this routine of mine there has been no need to preserve the contents of any registers on the stack. The only time we have interfered with the registers contents is in line 210 but that part of the routine is fairly destructive anyway so we do not achieve anything by preserving the value of the A Register. However if you are using a more sane routine and are going to change the contents of any register then be sure to first use the PSHS or PSHU instructions to preserve the contents of those registers, not forgetting of course to PULS or PULU again just prior to handing control back to BASIC.

```
2000                    00100          ORG     $2000
2000 86  29             00110 SETUP    LDA     #41
2002 97  96             00120          STA     150
2004 BE  0168           00130          LDX     $168
2007 AF  8D E3F1        00140          STX     VECTR2,PCR
200B 30  8D E3E1        00150          LEAX    START,PCR
200F BF  0168           00160          STX     $168
2012 39                 00170          RTS
03F0                    00180          ORG     $3F0
03F0 81  5F             00190 START    CMPA    #$5F
03F2 26  07             00200          BNE     VECTR1
03F4 96  00             00210          LDA     0
03F6 97  71             00220          STA     113
03F8 7E  A027           00230          JMP     $A027
03FB     7E             00240 VECTR1   FCB     $7E
03FC     0000           00250 VECTR2   FDB     0
         2000           00260          END     SETUP
00000 TOTAL ERRORS

SETUP   2000
START   03F0
VECTR1  03FB
VECTR2  03FC
```

BYTE MASTER

The Long And Winding Road To Assembly Graphics

By R. Bartly Betts

Sometimes the shortest way home is the longest way around, as my grandmother used to tell me. It was good advice when faced with the prospect of either climbing a perpendicular cliff or walking around the rocks.

This month it might be good advice when faced with the job of setting up your computer for assembly language graphics. It is a long trip to explain the process in detail but, when we get to the other side, the trip is going to seem worth the effort. Also, it is probably easier than trying to scale the cliff and take the short route.

Back in the Old Days

A number of people have asked about accessing graphics screens from assembly language. There has been a lot written about it, but often the information is difficult to understand.

Back in the days when Tandy sold the Non-Extended Color Computer, they also supplied a Color BASIC manual that contained information for accessing graphics pages directly. This was necessary because Color BASIC did not contain PMODE, SCREEN or COLOR commands. There was no way to obtain high resolution graphics except by directly manipulating the registers that set graphics pages and modes.

When Extended Color BASIC became the standard, there seemed no need for this information in a BASIC manual and

it was excluded. However, this is exactly the information you need to access graphics in assembly language.

Graphics Background

As well, many Color Computer owners found it difficult to understand the information in the Color BASIC manual. It is still hard to obtain simple, easy-to-use information about the Color Computer's graphics capabilities. Usually manuals and articles assume readers already have basic graphics knowledge.

Graphics images consist of patterns of pixels (graphics blocks). For each pixel on the screen, you may choose two, four or eight colors, depending on the amount of detail desired. The degree of detail or definition is referred to as "resolution." In a two-color graphics mode, a pixel is either set (on) or reset (off). When it is set, it is the foreground color. When it is reset, it is the background color. However, in four-color modes, the pixels can be in any of four color conditions.

On the graphics screen, pixels are arranged in rows and columns. The lowest resolution screen is the text screen, consisting of 16 rows from top to bottom and 32 columns from side to side. To get a feel for how graphics work, imagine that the 'X' characters below each represent one screen position. The text screen is then arranged in this manner:

```
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

I used the text screen for my examples to save typing thousands of 'X' characters. The principle is the same for both the text and graphics screens. For instance, you can fill the screen with X's to see how this works. Use these BASIC commands to do the job:

```
10 FOR T=0 TO 511
20 PRINT @ T,"X";
30 NEXT T
```

Now, create a cross on the screen by replacing the 'X' characters with a space by adding these lines:

```
40 PRINT @ 80, " "
50 PRINT @ 112, " "
60 PRINT @ 144, " "
70 PRINT @ 174, "     "
80 PRINT @ 208, " "
90 PRINT @ 240, " "
100 PRINT @ 272, " "
```

This simple program starts at the middle of Line 3 (2 x 32 = 64 + 16 = 80) and prints a space. Because a line is exactly 32 columns wide, calculate the position immediately beneath the first by adding 32. The result is 112. Add 32 again for the next position; you get 144. The cross piece begins on the next line, but two spaces short of the middle, and contains five spaces. The next three positions follow immediately below.

```
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXX     XXXXXXXXXXXXXX
XXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXX XXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

On the text screen, this example takes up seven of the 16 screen rows and is quite large. However, using the highest resolution graphics screen, the same pattern could be smaller than a character on this page.

I realize this example seems extremely simple and perhaps somewhat boring, but it demonstrates the principles behind creating graphics with your computer. Perhaps such a simple example helps clear up some of the mystery behind computer graphics.

"Good" Resolution Needs Lots of Memory

Such "test" graphics are not very exciting because of the limitations of their size. However, the Color Computer has a total of eight graphics modes, five of which are accessible through Extended BASIC.

The number of pixels in graphics screen rows and columns depends on which of the graphics modes selected. Higher resolution uses smaller pixels. When using smaller pixels, it takes more of them to "fill" a screen. Because each pixel requires one bit of memory, higher resolution graphics require more memory.

The graphics mode selected also influences the shape of the graphics screen pixels. For instance, if you select a screen that has 64 pixels across and 64 pixels down, each pixel is square. However, if you select a screen that is 256 pixels across by 192 pixels down, the pixels are longer than they are wide. This is something to consider as you create images.

In the lowest resolution graphics, using SET and RESET on the text screen, the screen grid is 64 by 32, or 2,048 pixels. In the highest resolution graphics, the screen grid is 256 by 192, or 49,152 pixels.

Setting Graphics

There are five steps to establishing graphics screens. They are:

1) Determine what graphics mode you require.
2) Select the appropriate amount of RAM.
3) Clear the selected memory.
4) Switch in the Video RAM Page (the memory graphics page).
5) Set the graphics control register.

Choosing a Graphics Mode

The graphics mode used is up to you. The higher resolution graphics provide finer detail. For some applications this might be important, for others it might not. Table 1 shows what graphics modes are available.

The first three modes are not implemented in BASIC, but can be chosen from assembly language if you wish.

Note that in the two-color modes there is only one bit required for every graphics "element" or pixel. For instance, the 128 x 64 two-color mode requires 1,024 bytes of memory

(128 x 64 = 8,196 bits or 1,024 bytes). Four-color modes, however, require two bits for each pixel. For instance, the 128 x 192 four-color mode uses 6,144 bytes (128 x 192 x 2 = 49,152 bits or 6,144 bytes).

Table 1

Resolution	Colors	Memory Required	BASIC PMODE
64x64	4	1Ø24	N/A
128x64	2	1Ø24	N/A
128x64	4	2Ø48	N/A
128x96	2	1536	Ø
128x96	4	2Ø48	1
128x192	2	2Ø48	2
128x192	4	6144	3
256x192	2	6144	4

Only one bit is required in the two-color mode because a set bit represents the foreground color and a reset bit represents the background color. A four-color mode requires two bytes for each pixel to provide the choice of four color combinations, 00, 01, 10 or 11.

To set any of the elements to the foreground color in the two-color mode, all you need to do is set the corresponding bit to one; each bit in a byte can have its own color. To reset the bit, set the appropriate bit to '0'.

In four-color modes, think of a byte consisting of only four elements, rather than eight. You set the color of any of the four elements by setting the two adjacent bits to either 0, 1, 2 or 3 (00, 01, 10, 11 in binary).

Did You Pass Your Binary Lessons?

Using high resolution graphics is where binary lessons really are of value. In a two-color graphics mode of 256 by 192 pixels, bytes have to be set in such a manner that you get the proper bit values. Every eight bits has to be treated as a unit even though they are individual units. In a four-color graphics mode, you have to treat every two bits as individual units within a four-unit environment. It might seem a little tricky now, but if you know your binary and Hex, it becomes easy with practice.

Once you have decided on the graphics mode, you have to select where the graphics "pages" will reside, and tell your computer. The pages can be put anywhere you wish in free RAM, but the top of memory is usually best. At the top of memory, it is easy to protect your graphics pages from being destroyed by a BASIC operation.

A "video page" is considered to be 512 bytes. Thus, when selecting video locations, use 512 byte units. For instance, if you have 32K of memory and wish to use the highest resolution, set aside 6,144 bytes of memory. Calculate the proper starting address for the graphics by subtracting 6,144 from the top of your computer's memory, for instance 32,512 - 6,144 = 32,368. Now, to find out which video page represents your selection, divide by 512 (32,368 divided by 512 = 51.5). Choose 51 as the nearest whole page that provides enough room.

You now need to convert 51 to a seven-bit binary number (0110011) and use that number to set the video page register, located from memory location 65478 to 65491 according to Table 2. The column labeled "Bit No." represents the bit location in a byte, with the right-most bit being '0' and the left most bit being '6'.

Table 2

Bit No.	This Location Resets	This Location Sets
Ø	65478	65479
1	6548Ø	65481
2	65482	65483
3	65484	65485
4	65486	65487
5	65488	65489
6	6549Ø	65491

If a bit is to be set (equals 1), set the first odd-numbered memory location. If the bit is reset (equals 0), set the first even-numbered memory location. The binary number is read from right to left and the procedure follows the example in Table 3.

Table 3

Binary Number Right to Left	Set This Number Even	Odd
1		65479
1		65481
Ø	65482	
Ø	65484	
1		65485
1		65487
Ø	65488	

If this seems confusing, remember that you start at the right of your binary number and move left. Set the next odd memory address if the bit is '1', or set the next even address if the bit is 0. In this case, "set" means to put any value into the address. For instance, from BASIC you could do the following pokes:

```
POKE 65479,Ø
POKE 65481,Ø
POKE 65482,Ø
POKE 65484,Ø
POKE 65485,Ø
POKE 65487,Ø
POKE 65488,Ø
```

As you see, '0' is a valid "value." You could use 1 or 100 or 205, if you wish. From assembly language you could use whatever value happens to be in Register A and store it in memory:

```
STA 65479
STA 65481
STA 65482
    . . . and etc.
```

Clearing Video Memory

Once the video pages have been calculated and set, you might want to clear the memory to prepare it for whatever applications you plan. A simple way is to establish the beginning and end of memory to clear and use a loop to store each byte with '0'. Calculate the beginning address by multiplying 51 by 512 for an answer of 26,112. The lines to do the job might look like this:

```
        LDX     #26112
        LDA     #Ø
CLEAR   CMPX    #32512
        BEQ     DONE
        STA     ,X+
        BRA     CLEAR
DONE    RTS
```

Setting the VDG Register

The first step in setting the graphics mode is to set the VDG register. This is a three-byte memory location that is set using a value from Table 4.

Table 4

Video Control Value

Graphics Mode Mode	VDG Pattern	To Select Color Set ø / 1
SG6	øøø	16 / 24
SG8	ø1ø	ø / ø
SG12	1øø	ø / ø
SG24	11ø	ø / ø
G1C	øø1	128 / 136
G1R	øø1	144 / 152
G2C	ø1ø	16ø / 168
G2R	ø11	176 / 184
G3C	1øø	192 / 2øø
G3R	1ø1	2ø8 / 216
G6C	11ø	224 / 232
G6R	11ø	24ø / 248

Look up the three-bit VDG pattern that selects the graphics mode you want, Column 2. Set the VDG register, locations 65472 to 65477, in the same manner as before, that is, set the register locations according to the values of the three-bit pattern selected using Table 5.

Table 5

Bit No.	This Location Resets	This Location Sets
ø	65472	65473
1	65474	65475
2	65476	65477

For instance, if you select G6C, the bit pattern is 110. Set the register in the following manner:

Table 6

Binary Number Right to Left	Set This Number Even	Odd
ø	65472	
1		65475
1		65477

Again, you can set these register locations by storing any value in them, including '0'.

Setting the Control Value

To complete the task of setting your computer to put graphics to work, select the control value for the graphics mode desired from the third column of Table 4. Then use the value to set the control register located at 65314.

Tandy's manual states that you must store the selected control value without changing bits 0, 1 and 2 at location 65314. However, it doesn't seem that you can change these three least significant bits whether you want to or not. To demonstrate this, type the following:

```
PRINT PEEK (65314) <ENTER>,
```

The screen displays '4'. Then, if you type:

```
POKE 65314,ø : PRINT PEEK (65314) <ENTER>
```

The screen again displays '4'. Poking the location with '0' did not change the value. However, if you stored a value of 240 at location 65314, then peeked the location (as shown below), you find it now holds a value of 244. All but the three least significant bytes are changed:

```
POKE 65314,24ø : PRINT PEEK(65314)
```

But, that's enough discussion. By storing the proper value in memory location 65314, you have completed graphics initialization.

Because of the ease with which you can experiment and make changes, I suggest trying the graphics initialization in BASIC before going to assembly language. Once you have worked out exactly what you want to accomplish in BASIC, transforming it to assembly language is quite easy. In fact, I think writing the assembly language program is easier than the BASIC one.

Until Next Time

I promised myself I wouldn't type an article any longer than could be handled by the 24K available to *Telewriter*. I have about reached my limit. However, you do have all the information needed to get the job done, so dig in and try graphics on your own.

Next month I will complete the assembly language graphics introduction and talk more about creating graphics images. In the meantime, have fun.

CORRECTION

"Scan It And Understand It With Rapid Reading" (Sept 1985, Page 22): T.C. Taulli tells us there have been reports of BS (Bad Subscript) Errors in his program. He recommends changing Line 170 to read:

```
17ø CLS3: FOR XX=1 TO 5øø:NEXT
```

"The Great Rainbow Simulation Package" (Sept 1985, Page 42): Carlos Santiago writes that there is an error on Page 47 in Line 5842 of Listing 2, WORLDWAR. The value -1 actually represents "Defend Current Position," and -2 represents "Remain Neutral." To correct this, change Line 5842 to read as follows:

```
5842 PRINT"INPUT NEW GOAL FOR TH
E          ";Y$(C);" TROOPS":PR
INT" -1 = DEFEND CURRENT POSITIO
N.   -2 = REMAIN NEUTRAL
          ø TO 12 = INVADE THAT REGI
ON"
```

Treasure Island (Rainbow, July 1985) had a bug in it. Line 96 should be deleted and Line 94 should read:

```
94 IF A$()"S"THEN99
```

The difficulty arose because Dean sent one of his backup copies, and when I checked it I couldn't make it play, but thought it was just me - I don't like adventure games!

I hope all those who have been asking will now be able to enjoy Deans game. G.

Screen Pokes Made Easy

By Bill Bernico

One of the things computer enthusiasts like to do is trade original programs with other hackers. Why not? They're proud of their own creations. Every so often, after your program has made the rounds it finds its way back to you, and upon listing it, someone has substituted their name for yours in the credit lines!

That's partly because the forger figured it would never get back to you and partly because you made that line easily accessible! I realize an experienced hacker can modify and crack almost anything; I can't help you there. However, the people who switch credits are doing it because they probably don't know that much about programming their own original material. These are the people I can help you with.

Leaving your name right out in plain sight is tempting, but hiding it in POKE statements discourages anyone from modifying it. The procedure is simple enough if you know the meaning of the numbers being poked.

First, think about how text looks on the screen. Second, think about how it gets there via the PRINT @ statements. Third, familiarize yourself with the text screen in terms of POKE positions, and last, know which POKE values correspond with the characters you want on the screen.

Let's look at step one. Mentally arrange the text screen, or even write it down to see how it will look. Table 1 shows that for every PRINT @ position, there is a corresponding POKE position. Instead of the screen positions starting at zero and ending with 511, as it does with PRINT @, the POKE screen starts with 1024 and ends with 1535 — still 511 positions.

The screen presentation in Table 1 is a result of both methods. You can use the procedure in *SCRNPOKE* or the procedure in Listing 3 to get the same results. Listing 3 is a lot shorter, but is more easily tampered with (decide if your text is important enough to hide before choosing either method).

Step two asks us to think about how the text got there using PRINT @ statements. In this case, Line 20 of Listing 2 says PRINT @ 195, "sCREEN pOKES mADE eASY. To put that same text on the screen using POKE statements, we first have to know the screen positions and what values go there. You can glance quickly at Table 2 or use the method in Listing 1. Either will show the correct POKE value for the character wanted. With this in mind, we can see that instead of starting to PRINT @ 195, we will poke values starting at position 1219 and continue poking up to position 1240, a total of 22 pokes, *including blank spaces*.

Looking at Table 2, we can see how each value corresponds with each character. These values are the numbers that show up in the DATA statements: three lines — three DATA statements. We could have poked each one individually, but that takes up more space.

Another point I want to touch on has to do with special inverse characters. While it's true you can create *some* of these screen pokes using PRINT @ statements, some special inverse characters can only be accessed by poking them into screen locations. For example, you can SHIFT 0 to get inverse letters, but to get inverse numbers, 1985 for example, you have to poke the four values into the screen. Inverse 1 = POKE value 49, 9 = 57, 8 = 56 and 5 = 53. The same is true for characters like the percent sign, quotation marks, colons, etc.

The last point I'd like to make has to do with the last screen position, number 1535. If you PRINT @ 511, the screen will scroll up. It won't if you poke that same character into position number 1535. Keep this in mind if you want to use PRINT @ with a FOR/NEXT loop to put a border around your text screen. PRINT @ every border location *except* number 511. Leave it blank and fill it in with POKE 1535,xx (xx being the character that matches the rest of the border).

Listing 1: TITLPOKE

```
10 CLS
20 FOR X=1219 TO 1240
30 READ A
40 POKE X,A
50 NEXT X
60 FOR Y=1285 TO 1299
70 READ B
80 POKE Y,B
90 NEXT Y
100 FOR Z=1350 TO 1357
110 READ C
120 POKE Z,C
130 NEXT Z
140 DATA 19,67,82,69,69,78,143,
         16,79,75,69,83,143,13,
         65,68,69,143,5,65,83,
         89
150 DATA 2,89,143,2,73,76,76,
         143,2,69,82,78,73,67,79
160 DATA 104,67,105,143,113,
         121,120,117
```

Listing 2: DEMOPOKE

```
10 CLS
12 PRINT@9,"screen pokes
14 FOR X=0 TO 255
16 PRINT@96,"EXAMPLE: POKE 1297,
";X
18 PRINT@192,"POKE VALUE";X;
20 PRINT@224,"INTO SCREEN
22 PRINT@256,"POSITION 1297 ->
24 PRINT@305,"^ TO GET THIS
26 PRINT@483,"HIT ANY KEY FOR NE
XT VALUE";
28 POKE 1297,X
30 EXEC 44539
32 NEXT X
```

Listing 3: TITLPRNT

```
10 CLS
20 PRINT@195,"sCREEN pOKES mADE
eASY
30 PRINT@261,"bY bILL bERNICO
40 PRINT@326,"(C) 1985
```

Table 1

ADD THESE VALUES TO EITHER METHOD

PRINT @	POKE	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
0	1024																															
32	1056																															
64	1088																															
96	1120																															
128	1152																															
160	1184																															
192	1216						S	C	R	E	E	N		P	O	K	E	S		M	A	D	E		E	A	S	Y				
224	1248									B	Y		B	I	L	L		B	E	R	N	I	C	O								
256	1280												(	C	)		1	9	8	5												
288	1312																															
320	1344																															
352	1376																															
384	1408																															
416	1440																															
448	1472																															
480	1504																															

Table 2

PV=POKE Value CHR=Character

INVERSE

PV	CHR	PV	CHR	PV	CHR	PV	CHR
0	@	16	P	32	■	48	0
1	A	17	Q	33	!	49	1
2	B	18	R	34	"	50	2
3	C	19	S	35	#	51	3
4	D	20	T	36	$	52	4
5	E	21	U	37	%	53	5
6	F	22	V	38	&	54	6
7	G	23	W	39	'	55	7
8	H	24	X	40	(	56	8
9	I	25	Y	41	)	57	9
10	J	26	Z	42	*	58	:
11	K	27	[	43	+	59	;
12	L	28	\	44	,	60	<
13	M	29	]	45	-	61	=
14	N	30	↑	46	.	62	>
15	O	31	←	47	/	63	?

NORMAL

PV	CHR	PV	CHR	PV	CHR	PV	CHR
64	@	80	P	96	□	112	0
65	A	81	Q	97	!	113	1
66	B	82	R	98	"	114	2
67	C	83	S	99	#	115	3
68	D	84	T	100	$	116	4
69	E	85	U	101	%	117	5
70	F	86	V	102	&	118	6
71	G	87	W	103	'	119	7
72	H	88	X	104	(	120	8
73	I	89	Y	105	)	121	9
74	J	90	Z	106	*	122	:
75	K	91	[	107	+	123	;
76	L	92	\	108	,	124	<
77	M	93	]	109	-	125	=
78	N	94	↑	110	.	126	>
79	O	95	←	111	/	127	?

Memory On A STRING$ Budget

By Jorge Mir

One of the most powerful string functions available in Extended BASIC is the INSTR (in-string) function. This function searches for the first occurrence of String 1 in String 2 and returns the position at which the match is found. If no match is found, it returns a zero.

This may be difficult to grasp at first, so let's look at some examples in order to illustrate the function:

```
1. A=INSTR ("ABCDEFG","C")
2. A=INSTR ("ABCDEFG","CDE")
3. A=INSTR ("ABCDEFG","X")
4. A=INSTR (2,"ABCDEFG","CDE")
5. A=INSTR (5,"ABCDEFG","CDE")
```

Example 1 — In this first example, the value of 'A' will be set to three since the character 'C' is located in the third position of the string "ABCDEFG"

Example 2 — In this case, the value of 'A' will also be set to three since the string "CDE" starts at the third position of the string "ABCDEFG"

Example 3 — Here is a case where 'X' is not contained in the string "ABCDEFG," so the value of 'A' is set to zero, meaning no match was found.

Example 4 — You can add a numeric value (must be less than the length of the string being tested) at which point you want to start the search. In this case, the computer will start searching at Location 2 (the 'B') of string "ABCDEFG" to see if the string "CDE" is contained within it. The value of 'A' will be set to three in this example since a match will be made and "CDE" starts at Position 3 of the string being tested.

Example 5 — In this example, the computer will start searching at Location 5 (the 'E') to see if "CDE" is contained in the rest of the string. The value of 'A' will be set to zero since "CDE" is not contained in that section of the string being tested.

You can use string values in the program step rather than the actual strings. Here is an illustration:

```
10 X$ = "ABCDEFG"
20 Y$ = "CDE"
30 Z$ = "X"
40 A = INSTR (X$,Y$)
50 B = INSTR (X$,Z$)
60 PRINT A
70 PRINT B
```

If you run this program, the screen will show a three (the value of 'A') since Y$ was found in X$ starting at Location 3, and a zero (the value of 'B') since Z$ is not contained within X$.

This is powerful stuff! But, how do you use it in a program? Here are some examples. Let's assume you have a data file containing names and addresses which have been stored in string values from N$(1) to N$(100) and you want to find out if there is anyone in the file who lives in Atlanta, Georgia. The subroutine might look like this:

```
1000 INPUT "KEYWORD: ";K$
1010 FOR X = 1 TO 100
1020 IF INSTR (N$(X),K$) = 0 THEN 1040
1030 PRINT N$(X)
1040 NEXT X
```

When the computer reaches this subroutine, you will be prompted for a "keyword" (Step 1000). In this case you will enter the word ATLANTA. The subroutine then goes into a loop, testing each of the 100 items in memory. Step 1020 tests to see if the word "Atlanta" (stored in K$) is contained in the string N$(X). If it is not, it will return a zero and the program jumps to Step 1040 where it continues to the next name and address. If it is contained in the string N$(X), it will then continue with Step 1030 and print that name and address record on the screen.

Please note that in the case above all records containing the word "Atlanta" will be printed on the screen, so if there is an Atlanta, Florida in the file, it will also be printed. Likewise, if someone's name is Atlanta, or if there is a street named Atlanta, those will also be printed. If you want to avoid this, you can type the keywords ATLANTA, GEORGIA, thus restricting the search further.

The INSTR function can also be used to conserve space in a program. For example, let's assume your program contains a menu from which the user is to select an item. The subroutine might look like this:

```
2000 CLS
2010 PRINT " A - ITEM ONE"
2020 PRINT " B - ITEM TWO"
2030 PRINT " C - ITEM THREE"
2040 PRINT:PRINT"    YOUR CHOICE?"
2050 I$ = INKEY$ : IF I$ ="" THEN 2050
2060 ON INSTR("ABC",I$) GOTO 2100,2200,2300
2070 GOTO 2050
2100 PRINT " YOU CHOSE ITEM ONE":END
2200 PRINT " YOU CHOSE ITEM TWO":END
2300 PRINT " YOU CHOSE ITEM THREE":END
```

Steps 2000-2040 print the menu on the screen. Step 2050 assigns a value to I$ equal to the key pressed on the keyboard (if no key is pressed, it repeats the step until a key is pressed). Step 2060 determines if the key pressed is an 'A,' 'B' or 'C.' If it is one of these keys, the program will continue with step 2100, 2200 or 2300. If it is not any of these three keys, Step 2070 will return to the start of the loop at Step 2050.

The function can also be used to conserve memory space by compacting data into strings and later being able to separate the data. Let's assume you want to add certain information to the name and address file referred to in a previous example. In addition to variables N$(1-100), you can also create variables I$(1-100) to store the added data, but this will use up a lot of memory. Instead, you can add the data to variables N$(1-100) using a specific code in order to indicate where the name and address data ends and the added data begins.

Let me illustrate how this could be done. Suppose you want to code each name and address file to indicate whether it belongs to a friend (code 1), a relative (code 2) or a business associate (code 3). In addition, you want to indicate whether a Christmas card was received (code 4), or sent (code 5).

Here is the way the data could be compacted:

```
3000 FOR X = 1 TO 100 : PRINT N$(X)
3010 INPUT "TYPE (1-3): ";T$
3020 INPUT "XMAS CARD RECEIVED (Y/N)";I$
3030 IF I$="Y" THEN CR$ = "4" ELSE CR$ = ""
3040 INPUT "XMAS CARD SENT (Y/N)";I$
3050 IF I$="Y" THEN CS$ = "5" ELSE CS$ = ""
3060 N$(X) = N$(X) + "/" +T$ + CR$ + CS$
3070 NEXT X
```

Step 3060 compacts the data by adding the '/' to indicate where the name and address ends and the codes start. If you had indicated that the record was a business associate from whom a Christmas card was received, it would look like this:

```
NAME AND ADDRESS/34
```

To select specific names and addresses through a subroutine, you have created the variable C1$ containing a '3' and the variable C2$ containing a '4' because you want to search the name and address file to find all business associates who have sent you Christmas cards, then print such names and addresses. Here is what the subroutine would look like in order to accomplish this:

```
4000 FOR X = 1 TO 100
4010 A = INSTR (N$(X),"/")
4020 IF A = 0 THEN 4060
4030 IF INSTR (A,N$(X),C1$) = 0 THEN 4060
4040 IF INSTR (A,N$(X),C2$) = 0 THEN 4060
4050 PRINT LEFT$(N$(X),A-1)
4060 NEXT X
```

In steps 4010-4020, we find out if the record has been coded. If the character "/" is not contained in the string, it means it has not been coded and the program goes to the next record.

In steps 4030-4040, we find out if both codes are contained in the record by examining the characters contained in variable N$(X) starting at the spot in the record where "/" was found. If neither of these codes is found, the program advances to the next record.

In step 4050 we print the record, except we exclude the character "/" and the codes that follow, only printing the name and address. In other words, we print the left side of the record up to the character prior to where "/" is located.

Now, let's complicate this matter further. Let's assume the name and address record which is all contained in a single string contains a '$' to indicate separations between name, city, state and ZIP code (since you want the data printed in the usual format rather than in a single line). In this case, N$(X) would look like this:

```
JOE SMITH$1234 MAIN STREET$ANYTOWN, WI 53533/34
```

In order to print the record in the regular name and address format, steps 4045 and 4055 would be added to the subroutine and would look like this:

```
4045 N=INSTR(N$(X),"$"): IF
     N=0 THEN 4050 ELSE
     MID$(N$(X),N,1) =
     CHR$(13):GOTO 4045
4055 N=INSTR(N$(X),CHR$(13)):
     IF N=0 THEN 4060 ELSE
     MID$(N$(X),N,1)="$":GOTO
     4055
```

In Step 4045, we replace each "$" with a character 13 (a carriage return) until all have been replaced. In Step 4055, we restore the "$" in their original spot before continuing to the next record.

On the other hand, if you wanted to print the name and address file in single line format, instead of inserting a CHR$(13) where each "$" is located, just insert a blank space (either a CHR$(143) or " " would do the trick).

There are many other uses of this function which could simplify and compact your programs considerably, thus conserving valuable memory. You should become familiar with it by writing your own subroutines in various formats and for various purposes. You will soon recognize its many uses when you sit down to write that huge program that will do everything for you.

Hint . . .

Cold POKE

If for some reason you want to simulate a cold start up, type POKE 113,0 and ENTER, then press the Reset button. (The usual EXEC 40999 doesn't work properly if your computer is in the 64K mode.)

Hint . . .

Automatic Lowercase

To move automatically to lowercase, type POKE 282,0. To switch back to all caps, type POKE 282,1.

Install a SHIFT-lock key on your CoCo's keyboard

The Permanent Shift

By David Geoffroy and Norman Racine

After owning a TRS-80 Color Computer for some time, I have discovered a way of modifying the computer to make the use of uppercase lettering more practical.

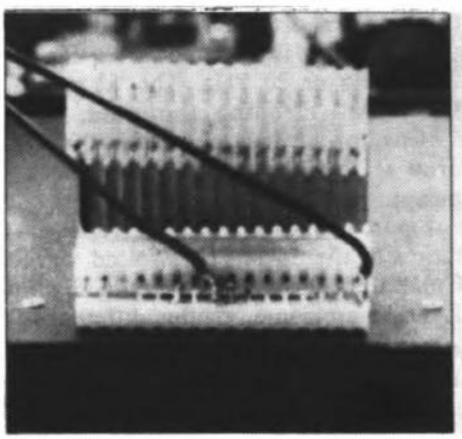
To use the computer in typewriter form, as in word processing, I noticed it did not have a SHIFT-lock key, as does a typewriter. I found it was so simple to install a SHIFT-lock key I wondered why I hadn't done it sooner.

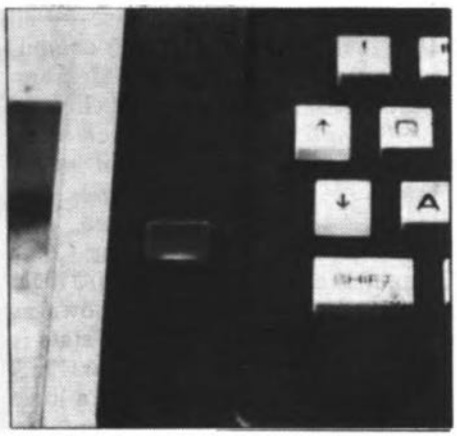
For installation, all that's needed is a push on/push off key switch (Radio Shack #275-1565) and two wires. If desired, connectors (Radio Shack #64-3049) can be installed on the wires to make disconnection possible if the cover is taken off again. (A note of warning: Modification will cancel your warranty.)

First, unplug the computer and remove the screws from the bottom. It is best to turn the computer upside down to remove the screws. Next, turn the computer back over, remove the cover and unplug the keyboard from the main board.

There are 16 pins on the keyboard connector. Pin #3 on the keyboard is missing — it is an unused ground. Now solder one end of the wires to pins #8 and #16 of the keyboard.

After soldering the wires to the pins, it is necessary to drill a hole in the lid





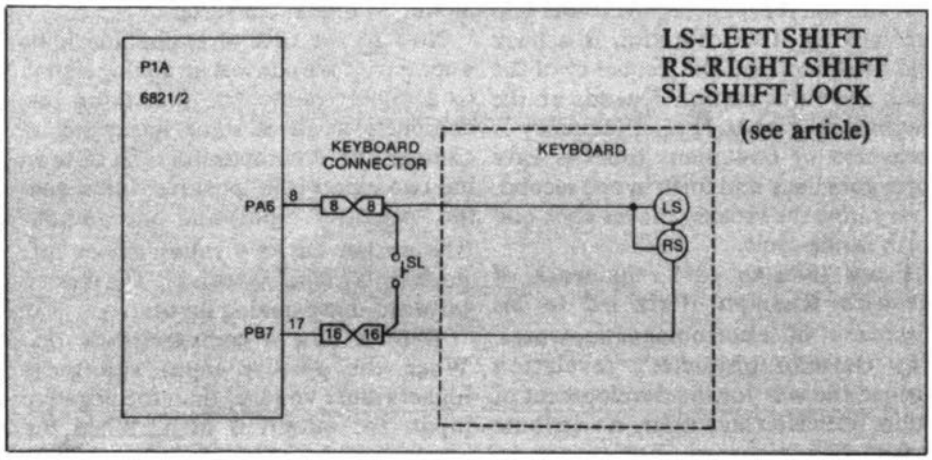
to the left of the left SHIFT key. Due to a post, the hole cannot be straight across from the SHIFT key, but will be just a little up. Also, the keyboard lies under the lid a little, so don't try to locate the switch too near the edge of the opening.

Now that the hole has been drilled, install the switch. Next, solder the wires to the switch. It makes no difference which wire goes where. Now, plug the keyboard in, put the top back on, install the screws and it's all done.

The SHIFT-lock key is useful for upper- and lowercase word processing, and it is great for listing programs. When listing programs, type LIST, push the SHIFT-lock key and then the ENTER key. When the '@' key is pressed, the listing will stop. By pressing any key again, and then the '@' key, there will be a few more lines listed.

A word of caution, though: The SHIFT-lock key is not labeled, but the SHIFT-lock key is red and does extend higher than any of the other keys. With a little effort it's easy to memorize where and what the key does.

(For anyone having questions concerning this project, Mr. Geoffroy may be contacted at 4700 28th Avenue, Sacramento, CA 95820.)



(David Geoffroy is a veteran of the United States Air Force. He lives in Sacramento, Calif., and works for the city repairing traffic signals. He has owned a CoCo for about four years now and enjoys it very much.

Norman Racine is interested in hardware and utility programs; he does assembly language and BASIC programming. He works for National Business Systems and owns a Color Computer 2.)

The Analog-To-Digital Converter

By Tony DiStefano

The world inside your computer consists of zeros and ones — all that goes on inside your computer hinges on two values. Memory, PIAs, CPUs, VDGs and SAM chips all transfer information between each other using only two different states. These states are called *logic states*.

The first logic state is zero, also known as "low" or "lo." In the Color Computer (and most computers) a logic state low is zero volts, also known as *ground* level. The second logic state is one, also known as "high" or "hi." Again, in the Color Computer, a logic state high is five volts. Except for specified tolerances, all other voltages in between are undefined and if encountered can give the computer some unpredictable results. This is the digital universe of computers. Figure 1 shows a typical digital wave form.

The real world, however, deals in ever changing states. Digital ones and zeros are just two of millions of different states that exist. The real world is an analog world. A good example of the analog world is speech. You can speak loud, you can speak low or many levels in between.

Sound waves are ever changing. For example, if you take a microphone and amplifier and hum into it, the speaker will vibrate, reproducing the sound you are making. That vibration is a back and forth motion. The frequency of the back and forth motion depends on the frequency of your hum. Frequency is measured by how many times a wave form goes back and forth in one second. Every time the speaker moves back and forth is one cycle.

From 1886 to 1888, the work of Heinrich Rudolph Hertz led to his discovery of electromagnetic waves. The German physicist's revelation opened the way for the development of radio, television and radar. As a tribute to him, the frequency of any wave, be it digital or analog, is measured in hertz (or Hz, for short). In the audible range, the frequency is from about 20 Hz to 20,000 Hz or 20 kHz. The 'k' stands for "kilo" meaning thousand. Our CoCos, for instance, run at 894,000 Hz or .9 MHz. The 'M' stands for "mega" meaning million.

Figure 2 shows a graphic representation of the output of a sound wave. Compare it to the wave form in Figure 1. There are some obvious differences; it is these differences that make it impossible for a computer to directly and accurately read and duplicate an analog wave form.

Don't despair, there are ways around it. This is the first of a two-part project on how you can use a computer to measure analog signals. This project stems from several letters received from my readers requesting that I build a computerized oscilloscope adapter for the joystick port. I looked into the joystick port as an input, but found it to be inaccurate or not fast enough. By the time you finish reading this, you'll know why.

Anyway, this month we'll cover the theory on how a computer (and a little hardware) can convert an analog signal to a digital value. Next month we'll cover how to build and calibrate the analog to digital converter.

Now to the task of explaining how a computer can convert an analog signal to a digital value. The first thing the computer needs is some hardware, a comparator. A comparator is an IC that has two inputs (the "positive" input and the "negative" input) and one output. The output has two states; on or off, good for a digital computer. The inputs, however, have analog inputs.

Here is how a comparator works. When the positive input voltage is higher (more voltage) than the negative input, the output is high. When the positive input voltage is lower (less voltage) than the negative input, the output is low. Figure 3 shows a block diagram of a computer-controlled comparator.

The way it works is simple. If we had a known voltage at the negative input, by reading the output (high or low) we could tell if the test voltage at the positive input is higher or lower than our reference voltage. Furthermore, if we change our reference voltage and zero into the unknown voltage, we will then know what the unknown voltage is. This technique is known as successive approximation.

The procedure for successive approximation is as follows: Start by putting half of the maximum voltage your device can measure to the reference voltage. If the output of the comparator is high, that means the unknown voltage is higher than the reference voltage. We then increase the reference voltage by half the difference of the present value and the last value and test again. If the output of the comparator is low, that means the unknown voltage is lower than the reference voltage. We would then decrease the reference voltage by half the difference and test again. Continue to do this until we have reached the unknown voltage.

Let's take an example. In this example, I round off the reference voltage to the nearest whole number for ease of calculation. The maximum voltage is 100 and the unknown voltage 47. The first reference value is 50 — too high, so we subtract from the present value using the successive approximation method. New reference is now (100-50)/2 or 50-25; the new reference is 25 — too low, so we add. The new reference is now (50-25)/2 or 25+13; the new reference is 38, still too low. Add again, (25 -38)/2 or 38+7. The new reference is now 45, again too low. Add (38-45)/2 or 45+4 and the new reference is 49. That's too high, so substract (49-45)/

2 or 49-3. The new reference is now 46, which is too low, so add (46-49)/2 or 46+2. The new reference voltage is now 48. Too high, so substract (49-48)/2 or 48-1. We have now reached the point where our reference voltage matches the unknown voltage.

Actually, the rounding off is not limited to integer calculation, but rather to the resolution of the reference voltage. When zeroing into the unknown voltage, you divide until the unit change in voltage is one. You cannot divide and get a more accurate fix on the unknown value. No matter how close you get, the comparator will always give a higher or lower value. The more accurate the reference, the closer you get to the real value of the unknown voltage.

This reference accuracy is one of the reasons why I chose not to use the joystick input. You see, inside the Color Computer there is all of the previously mentioned circuitry: a voltage comparator, a variable voltage reference, an unknown voltage input (joystick) and the interfacing circuit to control it all. A more common name for a variable voltage reference is "Digital-to-Analog Converter" or DAC for short. The DAC inside the CoCo is limited. It has a fixed output of .25 to 4.75 volts and the resolution of about 0.0715 volts. The range is not very good for an analog-to-digital project.

Another reason for not using the joystick input is speed. You see, the successive approximation method talked about earlier is time-consuming. The CPU has to calculate the next reference voltage value, set up the DAC, read the comparator output and make the proper decision.

The speed at which an unknown voltage can be found is very important. When the unknown voltage is stable and not changing, the computer can take all the time in the world to figure out what the voltage is. But, if the unknown voltage is changing, like the humming mentioned earlier, speed is important.

The amount of time it takes the CPU or other device to find an unknown voltage value is called the "conversion time." The faster the conversion time, the more samples can be taken and the more accurate the wave shape reproduction can be. For example, if you have a loop to read the A to D converter which takes 10 ms (ms = milliseconds = 1/1000 seconds), that means you have 100 samples per second. If you are sampling a wave form that is 1,000 Hz, you will miss a lot of information. It is safe to say you need at least 10 times the sample rate to reproduce a particular sine wave with reasonable accuracy.

A BASIC loop using the JOYSTK command will limit you to about 3 Hz — not very fast. In machine language, you can get a lot faster, but it is still slow due to the overhead created by the CPU having to do the conversion. In the case that the CPU has an external A to D converter, a great increase in speed and accuracy can be achieved. With the right software the effective conversion rate for an external A to D could be as high as 800 hertz.

The last thing I must mention this month is that the A to D circuit requires negative voltage. This is no problem with the first CoCos, but it is with the CoCo 2. The CoCo 2 has no negative voltage available at the cartridge connector. There is, however, negative voltage available inside the CoCo 2.

To bring this voltage to the cartridge connector is simple; you just need one piece of wire and a soldering iron. First, unplug the computer, then open it and locate the chip with the number SC77527; this is the SALT chip. You will find -12 volts on pin 15 of this chip (just what the doctor ordered). Solder one end of a piece of wire to that pin.

Figure 1

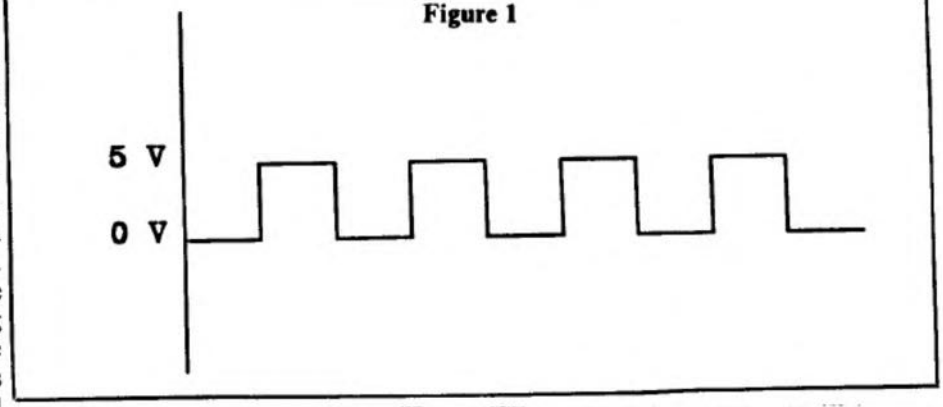


Figure 2

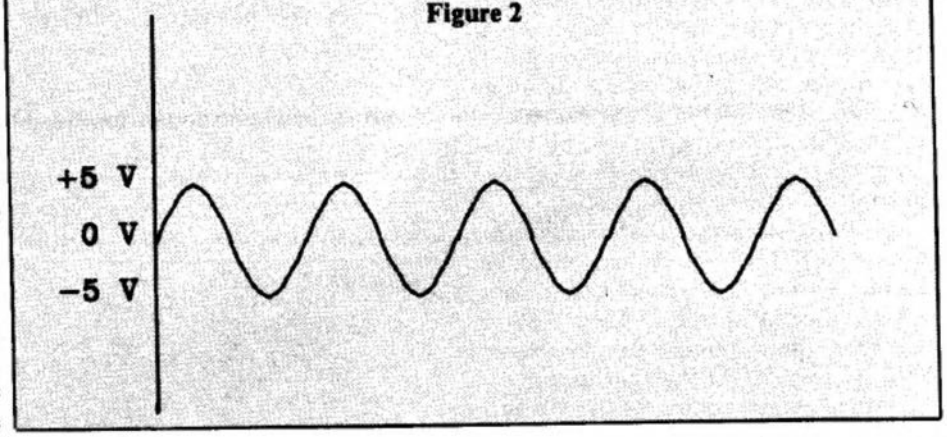
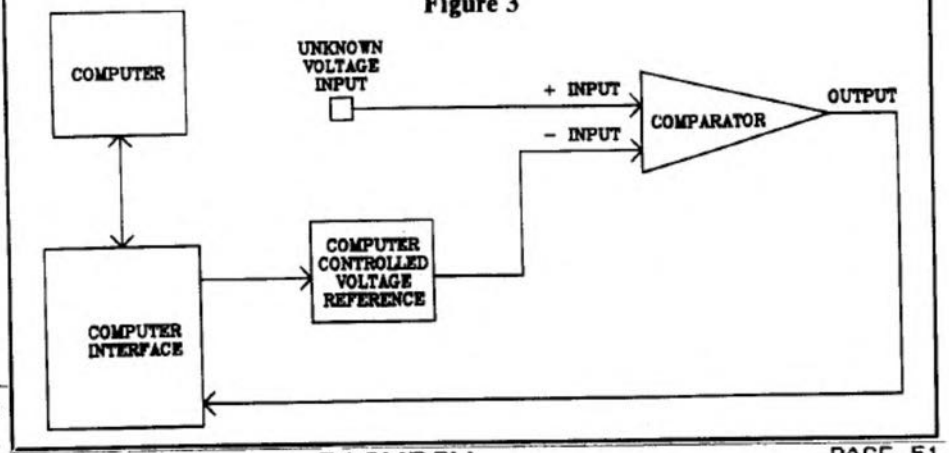


Figure 3



Locate pin #1 of the cartridge connector (it is the top pin closest to the back of the computer) and solder the other end of wire to this pin. Before you plug anything into the computer, measure the voltage to that pin. It should be about -12 volts, give or take two volts.

On the CoCo 2 this pin is normally not connected to anything. On the regular CoCo, this is the regulated -12 volt pin. The -12 volts you just added to that pin are not regulated, but in this and most cases, it will not matter. There will be a negative voltage regulator on the A to D convertor. Of all the peripherals I have seen for the CoCo, only one uses the negative voltage and it doesn't matter that it is not regulated. The most important part of this project is the chip that does all the work. There are many chips on the market today, ranging from very cheap and slow to extremely fast and expensive. My budget (and I am sure I'm not alone) is very tight. I found this chip in a local electronics surplus store and paid a little less than two dollars for it. The chip is the Teledyne Semiconductor number 8700CJ. It is an eight-bit analog-to-digital converter.

This converter is a fully self-contained, single 24-pin, dual in-line package. The circuit requires only passive support components. The conversion technique used in this chip is a bit different than the one I talked about last month, but the net results are quite the same. Conversion is performed by an incremental charge balancing technique that has inherently high accuracy, linearity and noise immunity.

An amplifier integrates the sum of the unknown analog current and pulses of reference current. The number of pulses (charge increments) needed to maintain the amplifier summing junction near zero is counted. At the end of conversion the total count is transferred into the eight digital outputs. Figure 1 shows the pinout of the 8700CJ analog-to-digital converter. The following is a pin-by-pin description of this converter.

Pin #	Description
1 to 4	No connection
5 to 12	Eight data lines — These output-only data lines represent the eight-bit value as a result of the conversion. Pin #5 is the most significant bit, Bit 7. Pin #12 is the least significant bit, Bit 0.
13	Iref — This the reference input current used to compare to the unknown current.
14	Iin — This is the unknown input current to be measured.
15	AMPout — The output of the first comparator. Used to limit high frequency oscillation.
16	ZEROadjust — This input is used to adjust so a zero voltage will be accurate.
17	Ibias — This input current adjusts between the speed of the conversion and the supply current. The faster the conversion, the more current it requires.
18	VSS — This pin requires 5 volts power supply.
19	VDD — This pin requires +5 volts power supply.
20	GND — This pin is the system ground.
21	INIT — This pin is a TTL level input used to start the conversion process. Can also be connected to be free-running.
22	BUSY — This pin is a TTL level output. When it is high (logic 1), the converter is busy calculating the next value.
23	VALID — This pin is a TTL level output that is high when the data at the eight data pins is latched with valid data.
24	Another pin with no connection.

Figure 2 shows the circuit I designed for this project. IC #2 is the converter chip — it is the heart of the project. All the pins described need not be repeated, however, there are a couple of other support chips that could use a little explanation.

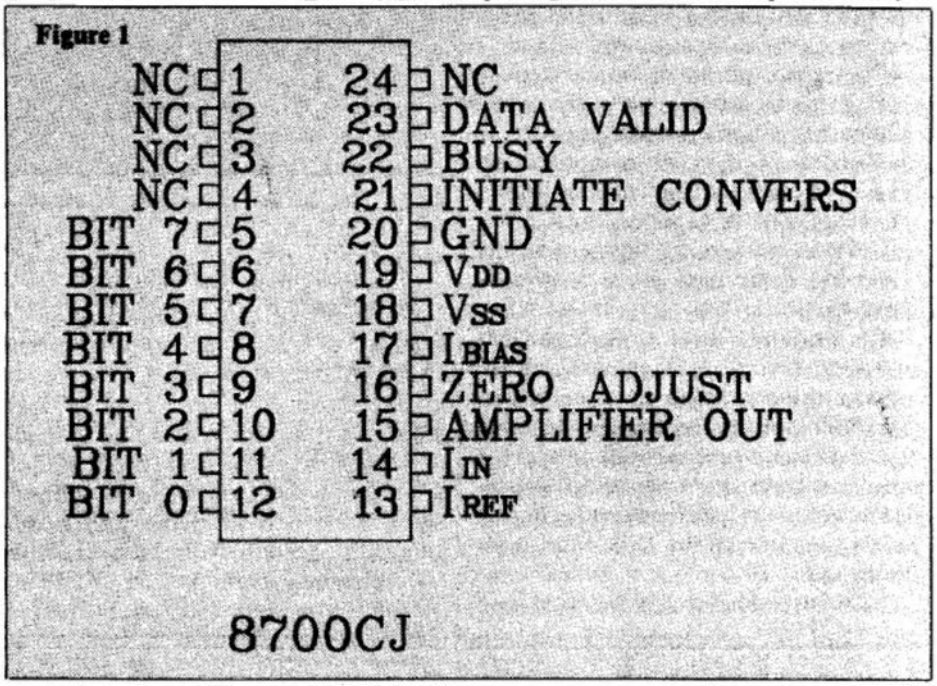
The first (IC #1) is a 74LS374. This is an eight-bit, D-Type flip-flop with tri-state outputs. It is used to store the data produced by the converter and to act as a buffer to the computer. The converter is wired in a free-running mode. That means as soon as it is finished doing a conversion, it immediately starts again as opposed to waiting for a signal from the computer to do another conversion. The *data valid* pin of the converter is connected to the *clock* pin of the 374, therefore transferring valid data from the converter to the flip-flops. Data is transferred from the converter to the flip-flop on the rising edge of the signal only, therefore no data is lost when the converter is busy doing the next conversion.

IC #3 is the other chip needed to make this work. It is a voltage regulator; a negative voltage regulator at that! It can take any negative input voltage from about -8 volts to -30 volts. The output will be a regulated -5 volts.

Why all this negative voltage? Well, the converter is kind of fussy that way. It needs -5 volts to work (something to do with the linearity I am told). If you are using a regular CoCo or a Multi-Pak Interface, there is no problem, but if you have a CoCo 2, you will have to fish out some negative voltage. (See last month's issue on how to do that.)

The 7905 is a three-pin chip that looks more like a power transistor than an IC. The pin numbers and description of this chip are simple. Looking at the chip and legs pointing downward, the left-most pin is ground. The center pin is the input

Figure 1



and the right-most pin is the output. The IC does not need to be mounted on a heatsink; there is not enough power demanded of it. It also does not need a socket.

The rest of the parts are just to make the converter work properly. There are only two adjustments to make; I'll get to that later, but now I would like to focus your attention on the three resistors, R5, R6, R7, and switch S1. You may or may not want to include these in your final circuit. You may want even more than three resistors. It all depends on what you want to use this circuit for.

The input resistor, R5, R6 or R7, depending on which one is in circuit at the time, is a scaling resistor. The value of this resistor will determine what the full-scale voltage value will be. To determine the full scale voltage, you must follow this simple formula: Rin = Vin full scale / 10uA.

Where Rin is the resistor value in question, Vin is the *full-scale* voltage wanted and 10uA (micro amps) is the current required for full scale. This current can be changed but will remain constant for now.

For example: You want a 10-volt full scale. Using the above formula, 10 volts divided by 10 micro amps equals 1,000,000 or one megaohm. I put R5 at 100K ohms to give a 1-volt full-scale reading, R6 is one megaohm for a 10-volt reading and R7 at 10 megaohm to give a 100-volt full-scale reading. These should be metal film resistors with a one percent tolerance or better to ensure high accuracy. I used these because of their thermal stability and low noise generation.

These three scales can be whatever you choose. Just follow the formula and you won't go wrong. You can even add more resistors for more scales, but of course you will have to change the switch S1. If you need only one scale, only one resistor is needed and you won't need the switch either.

The construction is not too complicated, but care has to be taken because the 8700CJ converter is a CMOS chip and it is very sensitive to static electricity. Use proper anti-static handling procedures. Do not insert the chip into its socket until everything is finished, checked and cleaned. It is important to clean the board properly. Leftover solder flux on the board can affect the accuracy of the circuit. It may even cause it to fail altogether.

Use the Parts List and the diagram in Figure 2 to build the circuit. After the circuit is finished and ready to be tested, insert the converter and power up. Like most of my projects, this one is made to work with the cartridge connector on the side of the computer. It will not work with a disk drive controller plugged in because it uses the SCS line and is memory mapped at $FF40 or 65344. If you want to change where it is mapped, read my article, "How to Follow a Memory Map," in the June 1985 RAINBOW. It will, however, work with a Multi-Pak Interface.

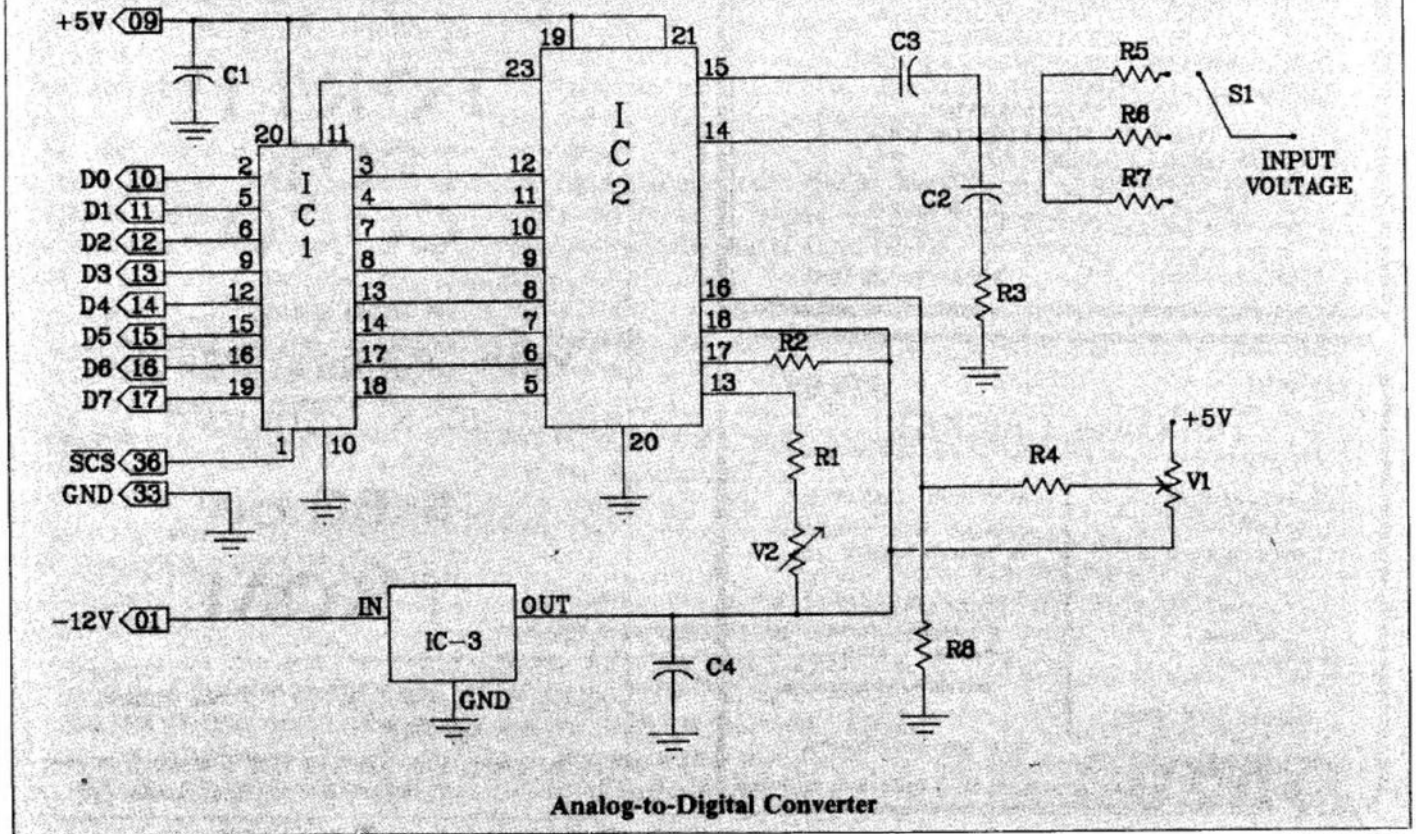
Follow the procedures with the MPI to set it up. In order to see if all is working well, a simple program is necessary.

```
10 CLS
20 PRINT @0, PEEK(65344) : GOTO 20
```

Run the program. Touch and let go the junction of C3 and C2 with your finger. The number on the screen should change value. If it does, all is well and you are ready for the adjustment procedures. If it dosen't, check over the circuit, repair the problem and try again.

The adjustment procedure is simple. The first adjustment is the *zero adjust*. Ground the input, that is, add a jumper from the input pin to ground. Adjust V1 until the value on the screen reads zero. Increase V1 until it just changes to one and then back off until it changes back to zero. Now remove the ground clip and enter a reference voltage. This reference voltage should be the full-scale voltage of the resistor selected

Figure 2



Analog-to-Digital Converter

above. This is the full-scale adjustment. For instance, if R6 is selected and the value is one megaohm, the full-scale voltage is 10 volts. Put a known 10-volt source to the input.

Different resistor values require different full-scale voltages. Adjust V2 until you read 254. Increase V2 until it just turns to 255. Go back to the zero adjust and check it again. Do this until both adjustments are right. If your values for R5, R6 and R7 are accurate all the other scales will follow. The accuracy depends on the accuracy of these resistors. If you are a real stickler, you can add a trim pot on every resistor and adjust each full-scale separately. That is all the adjustments you have to do.

That covers the hardware end of an analog-to-digital converter. There are a few things to remember about the circuit. First of all, it is only good for positive voltages. Negative voltages will register only as zero. It will not, however, hurt the converter. There is a way of biasing the converter to except negative voltages. If enough readers are interested, I'll do another article on how to expand on this converter.

The input impedance depends on the full-scale resistor. It will typically range from 100K ohms to about 10 megaohms. The possible uses for this type of circuit are endless. First, it is a voltage meter, used for measuring voltages of batteries, transformers, adapters, other circuits and many more. But, for most of these items it is simpler to use a $5 Radio Shack volt meter.

So why the fancy-pants converter? Well, there are many purposes. With the proper input device, one could make a long term study of outside temperature patterns. Another would be the slow changes of alpha waves in meditation. With the right software you could use your computer as an oscillascope or even control the temperature of your house. I can think of many things, just use your imagination.

As always, if you have a question or a problem and absolutely can't wait for the post office, call me on Monday nights *only*, and after supper, at (514) 473-4910. If you write and want a response, include a self-addressed, stamped envelope; my address is 4680 18th Street, Laval Quest, Quebec H7R 2P9. Sorry, I don't do windows.

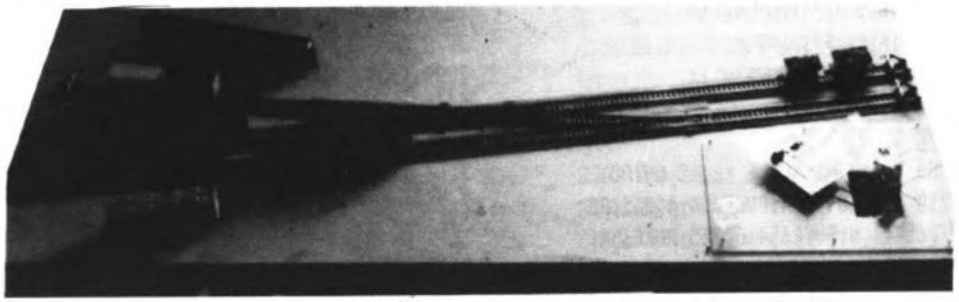
Parts List

IC1	—	74LS374 octal flip-flop
IC2	—	8700CJ eight-bit A-to-D (teledyne semiconductor)
IC3	—	7905 -5 volt regulator
R1	—	320K ohms ¼ watt
R2	—	100K ohms ¼ watt
R3	—	100 ohms ¼ watt
R4	—	100K ohms ¼ watt
R5	—	10K ohms ¼ watt one percent metal film
R6	—	1 meg ohms ¼ watt one percent metal film
R7	—	10 meg ohms ¼ watt one percent metal film
R8	—	1K ohms ¼ watt
C1	—	.1 uf 25 volts
C2	—	270 pf 25 volts
C3	—	68 pf 25 volts
C4	—	.1 uf 25 volts
V1	—	20K trim pot
V2	—	50K trim pot
S1	—	SPTT rotary switch
Misc.	—	24-pin socket, 20-pin socket, CoCo proto board, wire, solder, case, etc.

16K CB

ROBOTICS

by Graham Morphett



The model, as described with controlling relays exposed (right).

Last month, we began to look at what is required to get a small model railway operational using the CoCoConnection and CoCo as the controlling devices.

Since then, the track plan has taken a jump into hyper-space! Instead of the single track with perhaps one siding, we have ended up with a plan which is quite complex, and which presents several shunting problems.

As well as the train layout, the program has been developing too.

Initially, we simply took the Quizmaster program and deleted the sections which specifically related to the job Quizmaster does, retaining the input port checking routines for later use.

Then we looked at how we could fire up the ports as outputs. Until we determine how we will finally utilise the things we learned there, we have left these routines where they are.

So currently, the program can do two things, dependant on line 2. Change line 2 to read GOTO10 and the computer will control Stop/Go and Forward/Reverse functions of the train.

All that is required is that you connect a common wire to pin 8 of the end terminal, and wires from pin 1 of both ports 2a and 2b of the CoCoConnection.

These wires are then sent to relays which control the desired functions.

Lines 100 - 130 are a loop which checks for an INKEY$ of G, S, F, and R. (W and E are for later and you needn't include them yet).

With all connections made, and the program RUNning, assuming the train controller is on too, a press of G will make the train go. F is forward; R is reverse; S is stop.

We have begun a program at line 5000, which allows us to progress to the next stage, that is, from experiment to a real program. Please note that we are not intending necessarily to join these two programs. It is just convenient to have them both in memory at the same time at present.

The program at line 5000 sets on screen, the track diagram we are utilising, and allows us to identify the sidings and points. It also shows the changes that occur as the points are changed.

The POKES that physically change the points will be included in the routines from 5400 onwards at a later date.

In creating this track diagram, we realised that we didn't really need a real train! We could have a lot of fun simulating one on such a track diagram. Perhaps we'll do that after we've made it over the hump of the current project!

Next month, we expect to begin to meld the experimenting we are still doing, with the Track Diagram. There is quite a bit of work involved, because there is the model railway side to prepare as well.

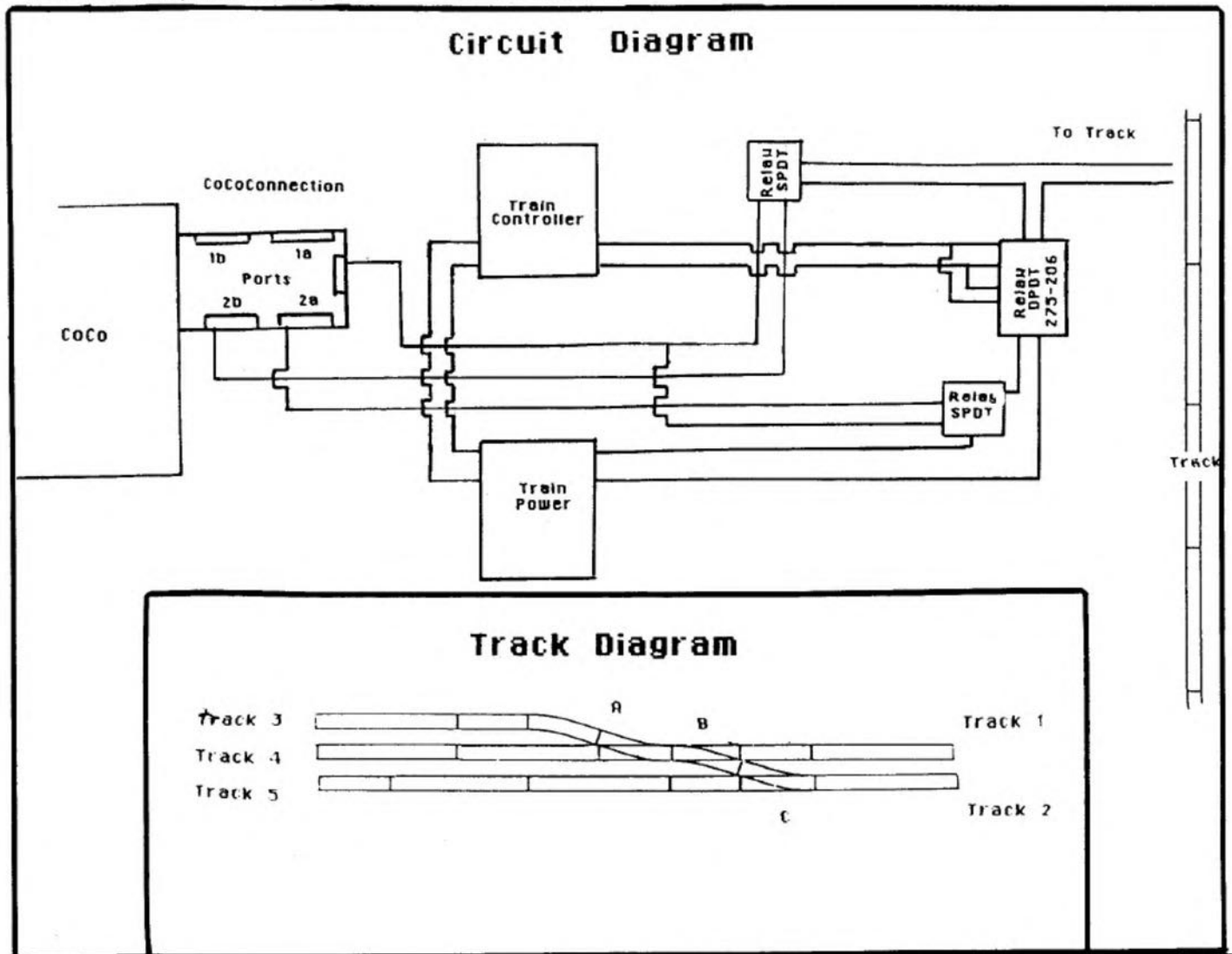
All track wiring is complete, but point motors are not yet fixed and the sensors we are to use have not been selected, neither have the sites for their placement been prepared.

The project is not at all complex. It just requires a little time and patience.

THE LISTING:

```
1 '********TRAIN*****************
  **********G. MORPHETT**********
  ***********NOV 85*************
2 GOTO5000
3 SAVE"TRAIN":DIR:STOP
10 'GOTO100
20 'INITALISES PORT2B AS OUTPUT
PORT
30 POKE65415,4:POKE65414,255:POK
E65415,0:POKE65414,255:POKE65415
,4
50 'POKE65413,0:POKE65412,0:POKE
65413,04
52 GOTO100
55 POKE65414,127
56 I$=INKEY$:IFI$="R"THENPOKE654
14,191 ELSE IFI$="S"THEN POKE654
14,127 ELSE IFI$="G" THEN POKE65
414,255
57 IFI$<>""THENPRINT@448,I$;
58 GOTO56
60 FORI=0TO7
65 IFI<=4 THEN A=INT(255-2^I) EL
SE A=INT(256-2^I)
70 'POKE65414,A
80 FORT=1TO300:NEXTT
85 PRINTA
90 NEXTI
98 STOP
99 'INITALISE 2A AS OUTPUT?
100 POKE65413,4:POKE65412,255:PO
KE65413,0:POKE65412,255:POKE6541
3,4
```

```
110 POKE65412,127
120 I$=INKEY$:IFI$="R"THENPOKE65
414,191
121 IFI$="G"THENPOKE65412,255
122 IFI$="S"THENPOKE65412,127
123 IFI$="W"THENPOKE65412,223
124 IFI$="E"THENPOKE65412,239
125 IFI$="F"THENPOKE65414,127
129 IFI$()""THENPRINTI$
130 GOTO120
190 STOP
200 '****INITALISE 1B AS OUTPUT
210 POKE65411,4:POKE65410,255:PO
KE65411,0:POKE65410,255:POKE6541
1,52
215 POKE65411,0:POKE65410,0:POKE
65411,4
220 POKE65410,127
230 I$=INKEY$:IFI$="R"THENPOKE65
410,191
235 IFI$="G"THENPOKE65410,255
240 IFI$="S"THENPOKE65410,127
245 IFI$()""THENPRINTI$
250 PRINT@230,"PORT 1B":GOTO230
2000 '***INITIALISE 1A AS INPUT
2010 POKE 65409,0:POKE 65408,0:P
OKE 65409,60
2020 POKE 65415,0:POKE 65414,0:P
OKE 65415,60
2030 '
2040 POKE65414,1
2050 STOP
2060 'PROBABLY NOT REQ
2070 READQ$,AN$
2080 IFQ$="Q"THEN700
2090 CLS:PRINTQ$
2100 A=PEEK(65408)
2110 B=255-A
2120 PLAY"O1;L255;ABCDEFG"
2130 IF B=0 THEN 2100
2140 GOSUB2170:GOSUB650:GOTO2070
2150 GOTO2100
2160 '
2170 '**CHECK FOR INPUT
2180 IFB=32THENX=6
2190 IFB=64THENX=7
2200 IFB=128THENX=8
2210 IFB=1THENX=1
2220 IFB=2THENX=2
2230 IFB=4THENX=3
2240 IFB=8THENX=4
2250 IFB=16THENX=5
2260 RETURN
2270 '
5000 '***TRACK DIAGRAM ROUTINE**
5010 CLS0:PRINT@9,"track";CHR$(1
28)"diagram";:PRINT@256,STRING$(
32,220);
5015 PRINT@247,CHR$(217);:PRINT@
214,CHR$(217);
5020 PRINT@160,STRING$(32,211);
5025 PRINT@180,CHR$(219);:PRINT@
147,CHR$(217);
5030 PRINT@114,CHR$(217);:PRINT@
96,STRING$(18,220);
5040 PRINT@256,CHR$(188);:PRINT@
287,CHR$(188);:PRINT@160,CHR$(17
9);:PRINT@191,CHR$(179);:PRINT@9
6,CHR$(188);
5100 PRINT@325,"options";:PRINT@
361,"Input";CHR$(128)"routine";:
PRINT@393,"Show";CHR$(128)"track
";CHR$(128)"numbers";
5110 PRINT@425,"Change";CHR$(128
)"points";
5200 I$=INKEY$:IFI$()""THEN5250
5210 I=I+1:IFI=1THENI$="i" ELSE
```



```
IF I=4THENI$="I"
5211 IFI=1 OR I=4 THEN PRINT@361
,I$;:GOTO5225
5215 IF I=2THENI$="s" ELSE IF I=
5THENI$="S"
5216 IFI=2 OR I=5 THEN PRINT@393
,I$;:GOTO5225
5220 IFI=3 THEN I$="c" ELSE IF I
=6 THEN I$="C"
5221 PRINT@425,I$;
5225 IFI=6 THEN I=0
5227 FORT=1TO50:NEXT
5230 GOTO5200
5250 IFI$="I" OR I$="i" THEN 530
0
5255 IFI$="S" OR I$="s" THEN GOS
UB 5350
5260 IFI$="C" OR I$="c" THEN GOS
UB 5350
5270 GOTO5200
5350 PRINT@187,"1";:PRINT@283,"2
";:PRINT@370,"3";:PRINT@166,"4";:
PRINT@262,"5";
5360 PRINT@146,"a";:PRINT@215,"b
";:PRINT@246,"c";
5365 IFI$="C" OR I$="c" THEN 540
0
5395 RETURN
5400 'CHANGE POINTS
5410 I$="":PRINT@257,STRING$(30,
204);:PRINT@161,STRING$(30,195);
:GOSUB5350
5420 PRINT@325,CHR$(128);CHR$(12
8);"press";CHR$(128)"point";CHR$
(128)"required";:PRINT@352,STRIN
G$(32,128);:PRINTSTRING$(32,128)
;:PRINTSTRING$(32,128);
5450 J$=INKEY$:IFJ$="C"THEN5500
5455 IFJ$="B"THEN5550
5460 IFJ$="A"THEN5600
5470 GOTO5450
5500 IFC=0 THEN C=1:L$=STRING$(2
2,220):L1$=CHR$(201):GOTO5510
5505 IFC=1 THENC=0:L$=STRING$(22
,204):L1$=CHR$(217)
5510 PRINT@257,L$;:PRINT@247,L1$
;
5515 GOSUB5350
5520 GOTO5450
5550 IFB=0THEN B=1:L$=STRING$(8,
211):L1$=CHR$(201):GOTO5570
5560 IFB=1THENB=0:L$=STRING$(8,1
95):L1$=CHR$(217)
5570 PRINT@183,L$;:PRINT@214,L1$
;
5575 GOSUB5350
5580 GOTO5450
5600 IFA=0 THENA=1:L$=STRING$(18
,204):L1$=CHR$(201):L2$=STRING$(
19,211):L3$=CHR$(203):GOTO5620
5610 IFA=1THENA=0:L$=STRING$(18,
220):L1$=CHR$(217):L2$=STRING$(1
9,195):L3$=CHR$(203)
5620 PRINT@97,L$;:PRINT@114,L1$;
:PRINT@147,L1$;:PRINT@180,L3$;:P
RINT@161,L2$;
5640 GOSUB5350
5650 GOTO5450
```

CoBBS:

An Introduction To The System And Its Special Features

By Richard Duncan

In this four-part series, you will learn about a BBS program that gives the SysOp complete control over the operation

Over several years there have been many bulletin boards written for the CoCo, from the simple to the very involved. As the complexity of the system increased, so did the price of the software. The desire to operate a bulletin board starts with most users during their first connection with a modem.

CoBBS, the Color Bulletin Board System, was started by the need to have a BBS that gives the SysOp (the system operator) more power in handling his operation and more users than many other systems, both commercial and public domain. CoBBS has been online and running for over a year and is about to top its 10,000th caller. In this first of a four-part series, we will take an overview of the system and provide the listing for the main program.

CoBBS (the program) supports up to 255 different menus, with up to 21 commands in each menu. The *Error Trapping* (Andy Kluck) and *Clock* (A. Trevor) programs used with the system come from CompuServe. *CoBBS* handles many different read/write message requirements, system controls and ASCII uploading and downloading. It was written on the original CoCo, but the CoCo 2 has been used by other SysOps and there are provisions within the program for the CoCo 2.

Hardware requirements are a 64K Color Computer with at least one drive, Multi-pak, RS-232 Pak and an auto-answer modem. To be able to take full advantage of the software, you need the RS-232 Pak. I started out using the serial port but quickly abandoned it after seeing how much power the RS-232 Pak added to the system. I strongly urge the use of an RS-232 Pak, but for those who would rather use the serial port, we will provide a modification for the RS-232 driver and show how to change CoBBS to operate with it in Part Four of this series.

The system has one bug that I have not been able to figure out. *CoBBS* must be operated in the 64K mode or it may crash. In this mode a few bytes of the first section of BASIC at memory location $8000 are written over, but it does not affect the system's operation.

User access, or the prevention of it, is always a concern. Some of the simpler systems do not have the capability to handle different types of callers. *CoBBS* is set up with each user having a privilege level and 16 "flags" in his user log file. Each menu and command on that menu also has a privilege level and 16 flags. For a user to see a menu or any command within the menu, he must meet or exceed the privilege level required. If the menu or command has a flag "set" (a one instead of a zero), the user must also have that flag "set" to be able to see and use it.

For example, a menu is set up with the following access requirements:

Priv=30 F1=00000000 F2=00000000

Before a user can access that menu he must have a privilege of 30 or higher. Since no flags are set, none are required in the user's flags. If the access requirement is:

Priv=30 F1=00000001 F2=00000000

a user would not only have to be given a privilege of 30 to see the menu, but he must also have the eighth flag (the last flag of group F1) set or he will not be allowed on that menu. The same is true of each command and each menu.

Each menu may have up to 21 commands. Each individual command is specified by the system operator. Without considering the various types at this time, let's look at Table 1 to see how each user is affected.

Table 1

Command	Priv	F1	F2
Read messages	25	00000000	00000000
Leave messages	30	00000000	00000000
Access user log	30	10000000	00000000
Access tracer	30	00000000	10000000
Terminate	0	00000000	00000000

Users who have access to this menu will see the header, menu number and name. But from there on, each of the following sees something different on the *same* menu.

User #1 Priv=20 P1=00000000 P2=00000000

This user only sees the terminate function. His privilege is too low for the system to allow access to any of the other commands.

User #2 Priv=25 P1=10000000 P2=00000000

User #2 may read messages since he meets that command's requirements. He may also terminate, but that is the only command he will see. Even though he has the first flag of P1 (the user's flags) set, he still cannot leave, or access the user log and the tracer routine because he does not have the required privilege.

User #3 Priv=30 P1=00000000 P2=10000000

User #3 may read and leave messages. He may also access the tracer routine since his first P2 flag is set. He may *not* access the user log since he does not have the first P1 flag set. Of course, he may also terminate.

User #4 Priv=40 P1=11111111 P2=11111111

This user has access to everything because all levels of access meet and/or exceed those required.

This type of operation allows the system operator to have control over the system by each individual calling, and can actually be set up so a menu is different to each caller.

When a menu is loading, it first checks to see if the user is allowed access to that menu, then it loads in all the command types for the menu. It checks the privilege and flag requirement for each command and compares it to the user to determine his access.

If accessible, the system keeps the command in memory. It retains the text for that command, the key required to initiate it and the *type* of command that is designated by ASCII character. This character is called the command type and is used by the system to determine what you want to do when the user hits the corresponding key. Table 2 lists the different command types.

Menus are added or changed using a Menu File Editor (a file called *SMF/EDI*). The menu displayed on the screen to the user is in the system in *memory* and not written as BASIC lines, which leads to greater flexibility and ease in changing various menus and commands. You do not have to modify the program to change, add or delete a command on any menu.

Table 2 lists the different commands available to the SysOp for menus. A command is referred to as a *type*, which is an ASCII character '!' through 'Z'. The *data* referred to is required by some types to complete their task. A type 'W' with a '3' in data allows the user to read all public messages on any menu number from '3' down and any private message if it is either to or from him.

The command types can be grouped into 11 categories. Remember that the type is specified by the SysOp with the file *MENU/SYS*, which contains all the information. Each command within the file contains the text for that command, which keystroke (input) to accept for the command, the privilege level, 16 flags, type of command and data (which may not be required for some types).

CoBBS consists of eight main programs. While online, only two files, *USER/SYS* and *COBBS/SYS*, are

needed for normal operations. Other files include *STARTUP/BAS* and *COTERM/BIN*, which are used while booting the system, three editor programs (*SCF/EDI, SUL/EDI* and *SCF/EDI*) and a message base purger (*SMP/EDI*). All files are needed to operate *CoBBS*, but we will be able to boot the program and play with it after Part Two next month.

The listing for the main program, *COBBS/SYS*, is included here. The system is set up in a series of subroutines that are accessed by the menu routine (lines 1000-1167). The menu routine determines the commands that are active, based on the user's privilege and flags. Hooks are used just as BASIC has hooks in memory.

A quick way to check a type's routine is to take the ASCII value of that letter. The ASCII value of 'R' for the standard read type is 82. Line 82 contains the pointer to the subroutine that handles the read function. Some hook lines set up data values before jumping to the proper subroutine.

The system includes a tracer routine so you know what each user did while online. The output of the tracer information can be to tape, disk or printer. The routine starting at Line 9600 is the tracer save routine.

A continual check for carrier detect is made while a user is on the system. If a carrier is lost, the system will reboot and be read for another call in a few seconds. Errors are trapped, and if not, a normal part of the operation is reported on the screen and sent to the tracer file. If the system cannot recover from the error it will shut down until you can determine the problem.

The system has a "blunderbuss" type of protection, if desired. If the system is protected, it will lock up unless the protection is removed before entering BASIC. Hackers cannot have access to your valuable files by causing the system to enter BASIC, if it ever does.

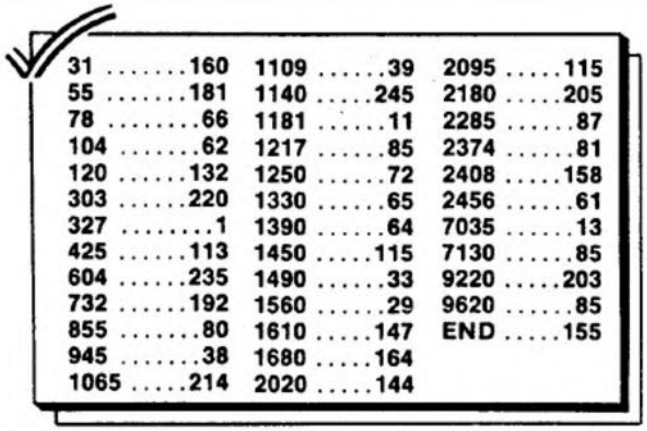
There are several *CoBBSs* operating around the country now. If you would like to check one out, my BBS number is (501) 735-5614 and is online 24 hours a day. If you have any questions about the system, please call my board and leave a message, or page me and if I'm available I can go voice on that line to talk with you.

Next month, we will discuss the various commands available and how they operate along with the rest of the files that make up the main operating system. □

Table 2

Command Types

Type	Name	Data
!	LOAD "SMF/EDI",R	
"	Reserved	
#	LOAD "SUL/EDI",R	
$	LOAD "SCF/EDI",R	
%	Reserved	
&	Reserved	
'	Print messages	0 or 1
(	LOAD "LOOKTRAC/BAS",R	
)	LOAD "LOOKREG/BAS",R	
*	Time on system	
+	SysOp option	
,	SysOp option	
-	SysOp option	
.	SysOp option	
/	SysOp option	
0	List HELP file	BNAME
1	List INFO file	BNAME
2	List TEXT file	BNAME
3	List BULL file	BNAME
4	Reserved	
5	Reserved	
6	Reserved	
7	Scan, 1 menu, all messages	Menu #
8	Scan, menu down, private TO/FROM	Menu #
9	Scan, menu down, all messages	Menu #
:	Reserved	
;	Reserved	
<	Reserved	
=	Reserved	
>	Reserved	
?	Reserved	
@	Menu control editor	
A	Abort out of menu	0,1,2
B	Reserved	
C	Chat. Page system	
D	Download a file	BNAME
E	Exit and run program	program name
F	Load and run file specified	
G	Goto menu #	Menu #
H	Reserved	
I	Reserved	
J	Reserved	
K	Kill messages	
L	Leave messages (stand.)	Menu #, (P)
M	Read E-mail (private)	Menu #
N	Leave E-mail (private)	Menu #
O	List user log	
P	Password enter/change	
Q	Quit, Exit to BASIC	0,1
R	Read messages, standard	Menu #
S	Scan messages, standard	Menu #
T	Terminate	0,1
U	Upload an ASCII file	0,1,2,3,4
V	Read, 1 menu, all messages	Menu #
W	Read, menu down, TO/FROM	Menu #
X	Read, menu down, all	Menu #
Y	Leave SysOp messages	
Z	Restart logon	



31	160	1109	39	2095	115
55	181	1140	245	2180	205
78	66	1181	11	2285	87
104	62	1217	85	2374	81
120	132	1250	72	2408	158
303	220	1330	65	2456	61
327	1	1390	64	7035	13
425	113	1450	115	7130	85
604	235	1490	33	9220	203
732	192	1560	29	9620	85
855	80	1610	147	END	155
945	38	1680	164		
1065	214	2020	144		

The listing: COBBSSYS

```
Ø '-COBBS (COBBS/SYS) COPYRIGHT
<C>1984 BY RICHARD DUNCAN
9 RUN T 3Ø:PRINT
1Ø CLEAR35ØØ:DIM BC$(21),L$(8Ø),
TY$(21),TX$(21),D$(21),KY$(21):N
U$=CHR$(Ø):DR$(Ø)="Ø":DR$(1)="1"
:DR$(2)="2":DR$(3)="3"
11 DEFUSRØ=&HØEDØ:DEFUSR1=&HØED3
:DEFUSR2=&HØE81:RUNT3Ø:GOSUB95ØØ
:GOSUB25:GOTO1ØØØ
25 REM
26 IFPEEK(4658)=ØTHEN28 ELSE CD=
PEEK(65385):CD=CD AND 32:IF CD<>
Ø OR PEEK(4657)<>Ø THEN 27 ELSE2
8
27 CLOSE:FORT=1TO1ØØ:NEXTT:TR$=T
R$+"LOSS OF CARRIER":GOSUB9615:L
OAD"USER/SYS",R
28 II$=INKEY$:IFII$=""THEN29ELSE
CH$=II$:K=INSTR("^_ ]",II$):IFK=
ØTHEN29ELSE ON K GOSUB8ØØ,31,713
Ø,98ØØ
29 RETURN
3Ø PRINT:PRINT"UNABLE TO ACCESS
PROPERLY.":PRINT"PLEASE CALL BAC
K LATER....":POKE65386,1Ø6:LOAD"
USER/SYS",R
31 IFPEEK(4684)<>188THENPOKE&HAC
7D,&HA3:POKE&HAC7E,&H9Ø:GOSUB86Ø
:GOSUB88Ø:ENDELSEGOSUB86Ø:GOSUB8
8Ø:FORX=&H112B TO &H1136:POKE X,
&H12:NEXTX:END
33 LOAD" SMF/EDI",R
34 RETURN
35 LOAD" SUL/EDI",R
36 RETURN
37 RETURN
38 RETURN
39 PRINT:PRINT:D=VAL(D$):IFD=ØTH
ENPRINT"FILE NOT AVAILABLE AT TH
IS TIME.":RETURN ELSEIF D=1 THEN
PRINT"SORRY, YOU ARE NOT AUTHORI
ZED.":RETURN
4Ø LOAD"LOOKTRAC",R
41 LOAD"LOOKREG",R
42 PRINT:PRINT:PRINT:F3=1:GOSUB7
1Ø:F3=Ø:RETURN
43 RETURN
44 RETURN
45 RETURN
46 RETURN
47 RETURN
48 X$="HELP":GOTO7ØØØ
49 X$="INFO":GOTO7ØØØ
5Ø X$="TEXT":GOTO7ØØØ
51 X$="BULL":GOTO7ØØØ
52 RETURN
53 RETURN
54 RETURN
55 BQ=VAL(D$):F6=1:GØ=1:GOTO2ØØØ
56 F6=1:GOTO87
57 F6=1:GOTO88
58 X$="":GOTO7ØØØ
59 RETURN
6Ø RETURN
61 RETURN
62 RETURN
63 RETURN
64 GOTO3ØØ
65 X=VAL(D$):GOTO2ØØ
66 RETURN
67 GOTO71ØØ
68 B=VAL(LEFT$(D$,1)):TD=B:IFB<3
THEN D$=RIGHT$(D$,4):GOTO12ØØ EL
SE ZK=INSTR(D$,"P"):ZD=VAL(MID$(
D$,2,1)):GOTO12ØØ
69 GOTO55Ø
7Ø RUNT111Ø:PRINT:PRINT"RUN FILE
: ";:GOSUB675:IFCH$=""THENRETURN
 ELSE LOAD CH$,R
71 BØ=BC:BC=VAL(D$):RETURN
72 RETURN
73 RETURN
74 RETURN
75 GOTO19ØØ
76 ZK=INSTR(D$,"K"):PF=INSTR(D$,
"P"):BL=VAL(D$):GOTO14ØØ
77 ML=1:BQ=VAL(D$):F6=Ø:GOTO2ØØØ
78 ZK=INSTR(D$,"K"):PF=INSTR(D$,
"P"):ML=1:BL=VAL(D$):F6=Ø:GOTO14
ØØ
79 GOTO9ØØ
8Ø GOTO4ØØ
81 IFVAL(D$)=ØTHEN31ELSE75Ø
82 BQ=VAL(D$):F6=Ø:GOTO2ØØØ
83 BQ=VAL(D$):F6=1:GOTO2ØØØ
84 IFVAL(D$)=ØTHEN98ØØELSEPRINT:
PRINT"WOULD YOU LIKE TO LEAVE":P
RINT"THE SYSTEM OPERATOR A MESSA
GE? ";:GOSUB625:IFCH$="Y"THENPRI
NT"YES":GOSUB95:GOTO98ØØ ELSEPRI
NT"NO":GOTO98ØØ
85 TU=VAL(LEFT$(D$,1)):TD=VAL(MI
```

```
D$(D$,2,1)):GOTO13ØØ
86 MQ=1:BQ=VAL(D$):GØ=1:F6=Ø:GOT
O2ØØØ
87 BQ=VAL(D$):GØ=2:GOTO2ØØØ
88 BQ=VAL(D$):GØ=3:GOTO2ØØØ
89 GOSUB95:RETURN
9Ø TR$=TR$+CHR$(13)+"RESTART LOG
 ON":GOSUB9615:LOAD"USER/SYS",R
95 MF$=NA$:MT$="SYSOP":MS$="SYSO
P MSG":BL=255:ML=2:GOSUB1445:RET
URN
1ØØ REM
1Ø1 DC=VAL(HEX$(PEEK(&HØEFD))):H
R=VAL(HEX$(PEEK(&HØEFE))):MN=VAL
(HEX$(PEEK(&HØEFF))):SS=VAL(HEX$
(PEEK(&HØFØØ))):HR$=RIGHT$(STR$(
HR),2):IFHR<1ØTHEN MID$(HR$,1,1)
="Ø"
1Ø4 MN$=RIGHT$(STR$(MN),2):IFMN<
1ØTHEN MID$(MN$,1,1)="Ø"
1Ø5 TI$=HR$+":"+MN$:IF SS<54ORMN
>58 THEN 1Ø8
1Ø6 MN=MN+1:IF MN>59 THEN MN=Ø:H
R=HR+1:IF HR>23 THEN HR=Ø
1Ø7 X=USR1(HR*256+MN)
1Ø8 X=PEEK(4611):IF X<>DC THEN 1
11
1Ø9 DA$=RIGHT$(STR$(PEEK(46Ø8)),
2)+"/"+RIGHT$(STR$(PEEK(46Ø9)),2
)+"/"+RIGHT$(STR$(461Ø)),2)
11Ø GOTO118
111 MM=PEEK(46Ø8):DA=PEEK(46Ø9):
YY=PEEK(461Ø):DA=DA+1:IFDA>31THE
N DA=1:MM=MM+1:IF MM>12 THEN MM=
1:YY=YY+1
112 POKE46Ø8,MM:POKE46Ø9,DA:POKE
461Ø,YY:POKE4611,DC:GOTO1Ø9
118 AX=PEEK(4615)*6Ø+PEEK(4616):
PT=HR*6Ø+MN:IFF3=1THEN124ELSEUO=
PEEK(4619):IF PT-AX<(UO*5)-5 OR
UO=255 THEN 121
119 TØ=PEEK(4619)*5:T1=TØ-(PT-AX
):IFT1<1THEN T1=Ø:GOTO12Ø ELSE P
RINT"YOU ONLY HAVE";T1;"MINUTES
LEFT!":GOTO121
12Ø PRINT"YOU HAVE NO TIME LEFT!
":TR$=TR$+"TIMED OUT "+TI$:GOSUB
9615:GOTO982Ø
121 '
124 RETURN
15Ø REM
152 F=ASC(F$):E=128:F$=""
154 FOR Q=1 TO 8
156 J=INT(F/E)
158 IF J=Ø THEN F$=F$+"Ø"ELSEF$=
F$+"1"
16Ø F=F-(E*J):E=E/2
162 NEXT Q
166 RETURN
175 REM
177 E=1:F=Ø
179 FOR Q=8 TO 1 STEP -1
181 IFMID$(F$,Q,1)="1"THEN F=F+E
183 E=E*2:NEXTQ:F$=CHR$(F)
185 RETURN
2ØØ REM
2Ø5 '
21Ø '
215 ON X+1 GOTO 22Ø,23Ø,24Ø
22Ø BC=BØ
225 RETURN
23Ø BC=PEEK(4618)
235 RETURN
24Ø BC=Ø
245 RETURN
3ØØ REM
3Ø3 BX=1:M=Ø:B=PEEK(4618):P9=Ø
3Ø6 CLS:PRINT:PRINT:PRINT"   MEN
U CONTROL EDITOR":PRINT:GOSUB339
3Ø9 PRINT"PRESENT LOG-ON MENU:";
B:PRINT:PRINT"<?> FOR LIST, OR":
PRINT"NUMBER TO LOG ON: ";:GOSUB
65Ø
312 IFCH$="?"THEN3Ø3ELSE BE=VAL(
CH$):IF BE<Ø OR BE>255 THEN 3Ø9
ELSE FOR X=1 TO LEN(BA$):IF BE=A
SC(MID$(BA$,X,1)) THEN 315 ELSE
NEXT X:GOSUB369:GOTO3Ø9
315 CLS:PRINT:PRINT:PRINT"  MENU
 PROMPTING":PRINT"  1 - FULL TEX
T PROMPT":PRINT"  2 - ABBREVIATE
D PROMPT":PRINT"  3 - KEY PROMPT
":PRINT"  4 - COMMAND ONLY":PRIN
T"ENTER PROMPT: ";:GOSUB6ØØ:P=VA
L(CH$):IF P>4ORP<1 THEN 315ELSEP
RINTCH$
318 PRINT:PRINT"WAIT...";
321 CLOSE:GOSUB92ØØ:GET#1,1:LSET
 UE$=CHR$(BE):POKE4618,BE
324 LSET UL$=CHR$(P):POKE4628,P:
UL=P:UX=P
326 PUT#1,1:X$=UN$+UP$:K7=PEEK(4
69Ø)*256+PEEK(4691):IFK7>LOF(1)O
RK7=ØTHENK7=2
327 GET#1,K7:Y$=UN$+UP$:IFX$=Y$T
HEN R=K7:GOTO33ØELSEFOR R=2 TO K
1:GET#1,R:IF UN$+UP$=X$ THEN 33Ø
 ELSE NEXT R:PRINT:PRINT"UNABLE
TO PROPERLY ACCESS.":PRINT"PLEAS
E TRY LATER.":CLOSE:RETURN
33Ø LSET UE$=CHR$(BE):LSET UL$=C
HR$(P):PUT#1,R:CLOSE:PRINT:RETUR
N
339 CLOSE:GOSUB94ØØ:BA$=""
342 FOR RM=1 TO K1 STEP 4
345 GET#1,RM:PB=ASC(M2$):PO=ASC(
M7$):IFP9=1ANDPO=ØTHEN372
348 GOSUB825:ON BX GOTO 357,366
```

```
357 IF PR<PB OR K7=1 THEN 372
36Ø PRINTTAB(3)ASC(M1$);TAB(1Ø)M
5$:BA$=BA$+M1$
363 GOTO372
366 IF ASC(M1$)<>BE THEN 372 ELS
E IF PR=>PB AND K7=Ø THEN 375
369 M=1:PRINT"YOU ARE NOT AUTHOR
IZED FOR THAT MENU.":GOTO 375
372 NEXT RM:IF BX=2 THEN 369
375 CLOSE:RETURN
4ØØ REM
4Ø5 RUNT117Ø
41Ø C=Ø:EXEC&H1ØDA
415 PRINT:PRINT"Please wait. Acc
essing userlog."
42Ø CLOSE:GOSUB92ØØ:GET#1,1:TT$=
UN$+UP$:K7=PEEK(469Ø)*256+PEEK(4
691):IFK7=ØORK7>K1 THEN425ELSE G
ET#1,K7:IFTT$=UN$+UP$ THEN A=K7:
GOTO44Ø
425 FOR A=2 TO K1
43Ø GET#1,A:IF TT$=UN$+UP$THEN44
Ø
435 NEXTA:CLOSE:PRINT"UNABLE TO
ACCESS PROPERLY.":RETURN
44Ø PRINT"PRESENT PASSWORD: ";UP
$:EXEC4314
445 PRINT"ENTER NEW PASSWORD: ";
:GOSUB65Ø
45Ø IFCH$=""THEN46ØELSEIFLEN(CH$
)>8THENPRINT"TOO LONG. MAXIMUM O
F 8 CHARACTERS.":GOTO445
455 CH$=LEFT$(CH$+STRING$(8,Ø),8
):LSET UP$=CH$:PUT#1,A:GET#1,1:L
SET UP$=CH$:PUT#1,1
46Ø CLOSE:RETURN
48Ø CLOSE:RETURN
5ØØ REM
5Ø5 K7=Ø
51Ø FOR Q=1 TO 8
515 IFMID$(F$,Q,1)="Ø"THEN525
52Ø IFMID$(PX$,Q,1)<>"1"THEN K7=
1:RETURN
525 NEXTQ:RETURN
55Ø REM
555 RUN T 117Ø
56Ø LOAD D$,R
6ØØ REM
6Ø4 GOSUB25:RUNT117Ø
6Ø5 EXEC&H1ØDA:CH$=CHR$(PEEK(448
1)):IC$=INKEY$:IFIC$<>""THEN CH$
=IC$:GOTO615
61Ø IFCH$=CHR$(Ø)THEN6Ø5
615 RETURN
625 GOSUB6Ø4:CH=ASC(CH$)
63Ø IFCH>96ANDCH<123THEN CH=CH-3
2
635 CH$=CHR$(CH):RETURN
65Ø REM
655 GOSUB25:RUNT117Ø:LINEINPUTCH
$:GOSUB25:RETURN
675 GOSUB655:G1$="":IFCH$=""THEN
RETURN
68Ø FOR G=1 TO LEN(CH$)
685 G1=ASC(MID$(CH$,G,1)):IFG1>9
6ANDG1<123THEN G1=G1-32
69Ø G1$=G1$+CHR$(G1):NEXTG:CH$=G
1$:RETURN
7ØØ REM
7Ø5 F3=Ø
71Ø GOSUB1ØØ:PT=(HR*6Ø)+MN:LT=(P
EEK(4615)*6Ø)+PEEK(4616):IF PT<L
T THEN PT=PT+144Ø
725 AX=PT-LT:HT=INT(AX/6Ø)
73Ø MT=AX-(HT*6Ø):PRINT"     STAR
T TIME:";STR$(ASC(CHR$(PEEK(4615
))));":";:A$=RIGHT$(STR$(ASC(CHR
$(PEEK(4616)))),2):IFVAL(A$)>9TH
ENPRINTA$ELSEMID$(A$,1,1)="Ø":PR
INTA$
732 PRINT"              TIME:";STR$(
HR);":";:A$=RIGHT$(STR$(MN),2):I
FVAL(A$)>9THENPRINTA$ELSEMID$(A$
,1,1)="Ø":PRINTA$
735 PRINT:PRINT"TIME ON SYSTEM:"
;STR$(HT);":";:A$=RIGHT$(STR$(MT
),2):IFVAL(A$)>9THENPRINTA$ELSEM
ID$(A$,1,1)="Ø":PRINTA$
736 '
745 RETURN
75Ø REM
755 PRINT:PRINT:PRINT"EXITING TO
 DOS...":PRINT"PASS: ";:GOSUB65Ø
:IFCH$=RIGHT$(D$,4)THEN TR$=TR$+
"+++EXIT TO BASIC+++":GOSUB9615:
GOTO31
78Ø PRINT:TR$=TR$+"**EXIT DENIED
**":GOSUB9615:PRINT:RETURN
8ØØ REM
8Ø5 GOSUB87Ø:PRINT:PRINT">>>";NA
$;"<<<":GOSUB88Ø
82Ø RETURN
825 REM
83Ø F$=M3$:GOSUB15Ø:F1$=F$:PX$=P
1$:GOSUB5ØØ:IFK7=1THEN84Ø
835 F$=M4$:GOSUB15Ø:F2$=F$:PX$=P
2$:GOSUB5ØØ
84Ø RETURN
85Ø REM
855 POKE&HØ16B,197:POKE&HØ16C,PE
EK(4684):RETURN
86Ø REM
865 POKE&HØ16B,16:POKE&HØ16C,2Ø5
:RETURN
87Ø REM
875 POKE&HØ168,PEEK(4681):POKE&H
Ø169,PEEK(4682):RETURN
88Ø REM
```

```
885 POKE&HØ168,&H1Ø:POKE&HØ169,&
HE6:RETURN
9ØØ REM
9Ø5 GOSUB92ØØ
91Ø FOR A=1TO5:PRINT:NEXT:PRINTT
AB(5)"1 - ENTIRE LIST":PRINTTAB(
5)"2 - SEARCH FOR USER":PRINTTAB
(5);">";
915 GOSUB6ØØ:PRINTCH$:PRINT:K=IN
STR("12",CH$):IFK=ØTHEN999
92Ø IF K=1 THEN935
925 PRINT" NAME: ";:GOSUB675:NT$
=CH$:IF NT$=""THEN999
935 PRINT:PRINT:PRINT"  PRESS 'S
' TO STOP - 'P' PAUSE":PRINT:FOR
 A=1 TO K1
94Ø GET#1,A:IF UR$=NU$ OR UR$=CH
R$(128) THEN 98Ø ELSEX=INSTR(UN$
,NT$):IF K=2 AND X=Ø THEN 96Ø EL
SE X$=UN$
945 X=INSTR(X$,CHR$(Ø)):N$=LEFT$
(X$,X-1):X$=RIGHT$(X$,5Ø-X):X=IN
STR(X$,CHR$(Ø)):L$=LEFT$(X$,X-1)
:PRINT:PRINTTAB(4)N$:PRINTTAB(4)
L$:IFINKEY$="S"THEN999
96Ø EXEC&H1ØDA:CH$=CHR$(PEEK(448
1)):GOSUB25:Y=INSTR("SsPp",CH$):
IF Y=Ø THEN 98Ø
97Ø IF Y<3 THEN 999
975 EXEC&H1ØDA:GOSUB25:IFPEEK(44
81)<>13THEN975
98Ø NEXTA
999 CLOSE:RETURN
1ØØØ REM
1ØØ5 GOSUB85Ø:GOSUB25:RUNT117Ø:S
S$=STRING$(2Ø,255):TT$="":CT$=""
:T$="":PRINT:PRINT"   WAIT....";
:PR=PEEK(462Ø):GOSUB94ØØ:EXEC&H1
ØDA
1Ø25 FOR R=1 TO K1 STEP 4
1Ø3Ø GET#1,R:IF ASC(M1$)<>BC THE
N 1Ø5Ø
1Ø35 GOSUB825:IFK7=1THEN1Ø46
1Ø45 IF ASC(M2$)<=PR THEN 1Ø6Ø
1Ø46 PRINT:PRINT:PRINT"YOU ARE N
OT AUTHORIZED ON THIS SECTION":P
RINT:CLOSE:CLOSE:BC=BØ:GOTO1ØØØ
1Ø5Ø NEXT R:GOSUB1ØØ
1Ø55 BC=BØ:GOTO1ØØØ
1Ø6Ø BN$=M5$:BT$=M6$:T$="":PL=Ø
1Ø65 FIELD#1,35 AS B$(1),35 AS B
$(2),35 AS B$(3),35 AS B$(4),35
AS B$(5),35 AS B$(6),35 AS B$(7)
,5 AS B$(8):GET#1,R+1:FORA=1TO7:
BC$(A)=B$(A):NEXTA
1Ø75 GET#1,R+2:FORA=8TO14:BC$(A)
=B$(A-7):NEXTA:GET#1,R+3:FORA=15
TO21:BC$(A)=B$(A-14):NEXTA:POKE6
5495,Ø
1Ø8Ø FOR A=1 TO 21
1Ø83 IF LEFT$(BC$(A),2Ø)=SS$ THE
N CO=A-1:GOTO11Ø9
1Ø84 TY$(A)=MID$(BC$(A),4,1):IFT
Y$(A)="L"ORTY$(A)="N"THENIFPL=ØT
HEN PL=ASC(LEFT$(BC$(A),1))
1Ø85 IF ASC(LEFT$(BC$(A),1))>PR
THEN 1Ø88
1Ø86 F$=MID$(BC$(A),2,1):GOSUB15
Ø:PX$=P1$:GOSUB5ØØ:IF K7=1 THEN
1Ø88
1Ø87 F$=MID$(BC$(A),3,1):GOSUB15
Ø:PX$=P2$:GOSUB5ØØ:IF K7=Ø THEN
1Ø9Ø
1Ø88 BC$(A)="":GOTO11ØØ
1Ø9Ø D$(A)=MID$(BC$(A),5,5)
1Ø95 KY$(A)=MID$(BC$(A),1Ø,1):TX
$(A)=RIGHT$(BC$(A),25):K=INSTR(T
X$(A),CHR$(Ø)):IFK=ØTHEN11ØØELSE
TX$(A)=LEFT$(TX$(A),K-1)
11ØØ CT$=CT$+KY$(A):NEXTA
11Ø9 PRINTSTRING$(11,8):GOSUB85Ø
:UX=UL:IFUX<1ORUX>4THEN UX=1
111Ø TT$=CT$:GOSUB1165:IFK>ØTHEN
114ØELSEGOSUB25:CLS:PRINT:IF UX>
2THEN1115ELSEX=INSTR(BT$,CHR$(Ø)
):IF X=Ø THEN 1115 ELSE FOR B=1
TO X:A$=MID$(BT$,B,1):IF A$="\"
THEN PRINTCHR$(13)ELSEPRINTA$;:N
EXT B
1115 GOSUB25:ML=Ø:PRINT:GOSUB8ØØ
:PRINT"*";BC;"*  ";BN$:GØ=Ø
1116 FOR A=1 TO CO
112Ø GOSUB25:IF TX$(A)=STRING$(2
5,32) ORBC$(A)="" OR LEFT$(BC$(A
),25)=STRING$(25,255) THEN 113Ø
1125 ON UX GOTO1126,1127,1128,11
29
1126 PRINTTX$(A):GOTO1129
1127 K=INSTR(TX$(A),CHR$(32)):IF
K=ØTHEN1126ELSEPRINTLEFT$(TX$(A)
,K-1):GOTO1129
1128 PRINTKY$(A);" ";
1129 GOSUB1165:IFK>ØTHEN114Ø
113Ø NEXT A:POKE65494,Ø:PRINT:IF
 PL=Ø THEN PL=25Ø
1131 GOSUB86Ø:GOSUB1ØØ:PRINT"COM
MAND: ";:XØ=Ø
1135 GOSUB625:IF CH$=CHR$(13) TH
ENPRINT:UX=1:PRINT:GOTO 111Ø ELS
EX=INSTR(CT$,CH$):IF X>Ø THEN PR
INT CH$ELSE1135
114Ø FOR A=Ø TO 19:IF MID$(BC$(A
),1Ø,1)=CH$ THEN 1145 ELSE NEXT
A:GOTO1135
1145 TY$=MID$(BC$(A),4,1):D$=MID
$(BC$(A),5,5)
115Ø G=ASC(TY$):CLOSE:B1=BC:GOSU
B86Ø
1155 GOSUB96ØØ:ON G-32 GOSUB33,3
```

```
4,35,36,37,38,39,4Ø,41,42,43,44,
45,46,47,48,49,5Ø,51,52,53,54,55
,56,57,58,59,6Ø,61,62,63,64,65,6
6,67,68,69,7Ø,71,72,73,74,75,76,
77,78,79,8Ø,81,82,83,84,85,86,87
,88,89,9Ø
116Ø IF BC=B1 THEN 111ØELSE1ØØØ
1165 EXEC4314:CH$=CHR$(PEEK(4481
)):CH=ASC(CH$):IFCH>96ANDCH<123T
HEN CH=CH-32:CH$=CHR$(CH)
1166 CV$=INKEY$:IFCV$<>"" THEN C
H$=CV$
1167 K=INSTR(TT$,CH$):IFK>ØTHENP
RINT:PRINTCH$:RETURNELSERETURN
117Ø REM
1175 PRINT:PRINT"SYSTEM ERROR, P
LEASE WAIT...";
1176 CLEAR1ØØØ:XØ=XØ+1:IFXØ>3THE
N982Ø
118Ø ER=PEEK(&HFD):EL=PEEK(&HFE)
*&H1ØØ+PEEK(&HFF)
1181 IF ER=>54 THEN EA=&HC242+ER
:GOTO1185
1182 IF ER=>5Ø THEN EA=&H88D9+ER
:GOTO1185
1183 EA=&HABAF+ER
1185 ER$=CHR$(13)+"***ERROR TYPE
 "+CHR$(PEEK(EA))+CHR$(PEEK(EA+1
))+" IN LINE"+STR$(EL)+"***":GOS
UB87Ø:PRINTER$:GOSUB9615:TR$=STR
ING$(32,"+")+CHR$(13)+ER$+CHR$(1
3)+STRING$(32,"+")+CHR$(13):GOSU
B96ØØ
119Ø RUNT117Ø:CLEAR1ØØ:GOSUB88Ø:
GOSUB86Ø
1195 RUN
12ØØ REM
12Ø5 RUNT1229:IFTD>2THENPRINT:PR
INT"FILE TO DOWNLOAD: ";:GOTO121
5
12Ø6 IF D9=Ø THEN D9=1:CH$="M":G
OTO1225
121Ø PRINT:PRINT:GOSUB1ØØ:PRINT"
[M]ENU, <CR> TO EXIT OR":PRINT"N
umber to download: ";:FT$="":F$=
""
1215 GOSUB675:RUNT1229:IFCH$=""T
HEN D9=Ø:GOTO111ØELSEFT$=CH$:IFT
D<3THENIFLEN(CH$)>4THEN12ØØELSE1
22Ø
1216 IFLEFT$(CH$,3)<>"DIR"THEN F
$=CH$:GOTO1226ELSE D=VAL(RIGHT$(
CH$,1)):IF TD<4THEN D=ZD
1217 IFD=1THEN DIR1:GOTO1219ELSE
IFD=2THEN DIR2:GOTO1219
1218 IF D=ØTHEN DIRØ ELSE DIR3
1219 PRINT"FREE:";FREE(D):GOTO12
ØØ
122Ø GOSUB25:IFCH$<>"M" THEN 123
Ø
1225 FT$="DOWN"+D$+"/MNU:":D=-1:
GOSUB1245:GOTO121Ø
1226 IFTD=3THEN D=ZD ELSEPRINT"D
RIVE: ";:GOSUB6ØØ:RUNT12ØØ:D=VAL
(CH$):IFD>3THEN1226ELSEPRINTD
1227 IFZK>ØTHEN X$=RIGHT$(F$,4):
IFX$="/SYS"ORX$="/BAK"ORX$="/BIN
"THEN12ØØ
1228 F$=F$+":"+DR$(D):RUNT129Ø:G
OTO123Ø
1229 GOTO111Ø
123Ø CLS:PRINT:PRINT:PRINT"1 - A
SCII, NO BUFFER":PRINT"2 - ASCII
, WITH BUFFER":PRINT"<CR> TO EXI
T"
1235 GOSUB6ØØ:RUNT12ØØ:X=VAL(CH$
):B=X-1:IFX=ØTHENRETURNELSEIFX>Ø
ORX<3THENPRINTTAB(3)CH$ELSE1235
1237 IFTD>3THEN1255
124Ø FT$=D$+FT$+"/DOW:":D=-1
1245 RUN T 1245
125Ø IFTD=3THEN129ØELSED=D+1:IF
D>3 THEN129ØELSE F$=FT$+DR$(D)
1255 CLOSE:UNLOAD:RUNT125Ø:OPEN"
I",#1,F$:RUN T 129Ø:IFCH$="M" TH
EN127Ø
126Ø IF B=Ø THENPRINT"OPEN BUFFE
R...":FOR O=1 TO 12ØØ:NEXT O
1265 IF B=1 THENPRINTCHR$(18);
127Ø IFEOF(1)THEN1275ELSEGOSUB25
:LINEINPUT#1,A$:PRINTA$:EXEC4314
:CH=PEEK(4481):IFCH=83ORCH=115OR
INKEY$="S"THEN1275ELSE127Ø
1275 RUN T 12ØØ:IFCH$="M" THEN12
8ØELSEIF B=1 THENPRINTCHR$(2Ø)EL
SEFORO=1TO12ØØ:NEXTO
128Ø CLOSE:TR$=TR$+CHR$(13)+"DOW
: "+F$+CHR$(13):GOSUB961Ø:GOTO12
ØØ
129Ø RUN T 12ØØ:PRINT:PRINT"UNAB
LE TO ACCESS FILE.":PRINT:GOTO12
ØØ
13ØØ REM
13Ø5 '
131Ø RUN T 111Ø:PE=Ø:PRINT:PRINT
:PRINT
132Ø X=FREE(TD):IFX<5ANDTU<3THEN
PRINT"SORRY, SPACE NOT AVAILABLE
.":FORA=1TO25Ø:NEXTA:RETURN
1325 PRINT"ENTER FILE NAME: ";:G
OSUB675:IFLEN(CH$)>8THEN1325ELSE
IFCH$=""THENRETURNELSEA$=CH$
133Ø PRINT"              EXT: ";:I
FTU<2THENB$="UPL":PRINTB$ELSEGOS
UB675:IFLEN(CH$)>3THEN133ØELSEIF
CH$=""THENRETURNELSEB$=CH$
1335 IFTU>2THENPRINT"          D
RIVE: ";:GOSUB625:D=VAL(CH$):IFC
```

```
H$=""THENRETURNELSEIFD>3THEN1335
ELSEPRINTCH$
134Ø IFTU<3THEN D=TD:RUN T 139Ø:
X=FREE(D):IFX<5THEN"SORRY, SPACE
 NOT AVAILABLE."
1342 X=FREE(D):IFX<3ANDTU<4THEN1
39ØELSE FL$=A$+"/"+B$+":"+DR$(D)
:IFTU<4THEN RUNT1345:OPEN"I",#1,
FL$:CLOSE:PRINT:PRINT"FILE ALREA
DY EXIST...":GOTO111Ø
1344 IFTU=4THENPRINTTAB(12)"FREE
:";X
1345 RUN T 139Ø:OPEN"O",#1,FL$:E
XEC4314
135Ø PRINT"READY TO RECEIVE. TYP
E":PRINT"'END' ON A LINE BY ITSE
LF":PRINT"TO EXIT UPLOAD.":PRINT
1355 LINEINPUT">";C$:GOSUB25:IFC
$="END"ORC$="end"THEN137Ø
136Ø PRINT#1,C$:GOTO1355
137Ø CLOSE:TR$=TR$+CHR$(13)+"UPL
: "+FL$:GOSUB96ØØ
1375 RUNT111Ø:CLOSE:RETURN
139Ø UNLOAD:PRINT"DRIVE NOT AVAI
LABLE.":PRINT:FORT=1TO15Ø:NEXTT:
GOTO111Ø
14ØØ REM
14Ø5 '
141Ø IF BL=255 THEN GOSUB166Ø:IF
 BL=255 OR CH$="" THEN RETURN
1411 X=FREE(PEEK(4671)):IF X<3 T
HEN PRINT:PRINT"MESSAGE BASE FUL
L!":RETURN
1415 MD$=CHR$(Ø):MI$=MD$
142Ø GOSUB25:PRINT:PRINT:MF$=NA$
:PRINT"FROM: ";MF$:EXEC&H1ØDA:PR
INT"  TO: ";:GOSUB675:IFCH$=""TH
ENRETURN
1435 MT$=CH$
144Ø PRINT"SUBJ: ";:GOSUB675:IFC
H$=""THEN144ØELSEMS$=CH$
1445 X=FREE(PEEK(467Ø)):Y=FREE(P
EEK(4671)):IF X<2ORY<2 THEN PRIN
T"MESSAGE BASE FULL!":RETURN ELS
E IFML>ØTHEN P$="1ØØØØØØØ":GOTO1
45ØELSEIFPF=ØTHEN P$="ØØØØØØØØ":
GOTO145ØELSEPRINT"MESSAGE PRIVAT
E (Y/N)? ";:GOSUB625
1446 IFCH$="Y"THEN P$="1ØØØØØØØ"
:PRINT"YES"ELSEIFCH$="N"THEN P$=
"ØØØØØØØØ":PRINT"NO"ELSEGOSUB625
:GOTO1446
145Ø K=LEN(MF$)+LEN(MT$)+LEN(MS$
)+2:IFK>64THENPRINT"SUBJECT TOO
LONG":PRINT"LIMIT TO ";64-LEN(MF
$)-LEN(MT$)-2:PRINT"TRUNICATING.
." ELSE 1452
1451 IFLEFT$(MS$,5)="REPLY"THEN
MS$=RIGHT$(MS$,LEN(MS$)-(K-64))
ELSE MS$=LEFT$(MS$,LEN(MS$)-(K-6
4)):GOTO145Ø
1452 GOSUB25:PRINT:PRINT:PRINTTA
B(5)"FROM: ";MF$:PRINTTAB(5)"  T
O: ";MT$:PRINTTAB(5)"SUBJ: ";MS$
1453 IFP$="1ØØØØØØØ"THENPRINTTAB
(5)"PRIVATE MESSAGE"ELSEPRINTTAB
(5)"PUBLIC MESSAGE"
1455 IF ML=2 THEN 1465 ELSE PRIN
T:PRINT"CORRECT (Y/N)? ";
146Ø GOSUB6ØØ:K=INSTR("NnYy",CH$
):IFK>2THENPRINT"YES":GOTO1463EL
SEIFK>ØTHEN1415ELSE146Ø
1463 PRINT:PRINT
1465 PRINT:PRINT"ENTER MESSAGE.
MAXIMUM OF 2ØØØ":PRINT"BYTES. MA
XIMUM OF 4Ø LINES.":PRINT"PRESS
<ENTER> ON LINE BY ITSELF":PRINT
"TO EXIT.":PRINT:LE=Ø:EXEC&H1ØDA
147Ø FOR LL=1 TO 4Ø
1475 PRINT">";:LINEINPUTCH$:IFCH
$=""THEN1497
1485 L$(LL)=CH$:GOSUB25:IF LE+LE
N(CH$)>1999THEN1496
149Ø '
1491 LE=LE+LEN(CH$):IF LL>35 THE
NPRINT"YOU ONLY HAVE";4Ø-LL;"LIN
ES LEFT."
1495 IFLE>15ØØTHENPRINT":";2ØØØ
-LE;" BYTES LEFT..."
1496 IF LE<2ØØØTHENNEXT LL ELSEP
RINT"BUFFER FULL!!!"
1497 POKE65494,Ø
15ØØ PRINT:GOSUB1ØØ:PRINT"<L>IST
 <E>DIT <C>ONTINUE <S>AVE <A>BOR
T ";
15Ø5 CH=1:GOSUB625:K=INSTR("LECS
A",CH$):IFK=ØTHEN15Ø5
151Ø PRINTCH$:ON K GOTO 1515,152
Ø,1475,1555:RETURN
1515 FOR C=1 TO LL-1:GOSUB25:PRI
NTC": ";L$(C):NEXTC:GOTO15ØØ
152Ø PRINT"LINE TO EDIT: ";
1525 CH=1:GOSUB65Ø:X=VAL(CH$):IF
X<1THEN15ØØ
153Ø IF X>LL THEN PRINT"NON-EXIS
TING LINE":GOTO152Ø
1535 PRINT"PRESENT LINE READS:":
PRINT:PRINTL$(X):PRINT:PRINT
154Ø PRINT"ENTER NEW LINE:":GOSU
B65Ø
1545 IFCH$=""THENPRINT:PRINT"LIN
E UNCHANGED":GOTO15ØØ
155Ø L$(X)=CH$:PRINT"LINE CHANGE
D":GOTO15ØØ
1555 LL=LL-1
156Ø PRINT:PRINT"SAVING MESSAGE.
 . .":GOSUB85Ø
1564 IF K8=1 THEN 1565 ELSE GOSU
```

```
B9ØØØ
1565 K1=LOF(1):IF K1=Ø THEN MN=1
:GOTO1575ELSE GET#1,1
157Ø MI=CVN(H1$)+1:IF MI>6ØØØØ T
HEN MI=Ø
1575 LSET H1$=MKN$(MI):F$=P$:GOS
UB175:LSET H2$=F$:LSET H5$=MF$+N
U$+MT$+NU$+MS$+STRING$(8Ø,NU$)
158Ø GOSUB1ØØ:Q$=CHR$(HR)+CHR$(M
N):LSET H4$=Q$:Q$=CHR$(PEEK(46Ø8
))+CHR$(PEEK(46Ø9))+CHR$(PEEK(46
1Ø)):LSET H3$=Q$:LSET H8$=CHR$(B
L)
1584 IF K8=1 THEN 1585 ELSE GOSU
B91ØØ
1585 K2=LOF(2):K2=K2+1:LSET H6$=
MKN$(K2)
159Ø A$=""
1595 B$="":LSET MG$=STR$(MI):PUT
#2,K2
16ØØ FOR A=1 TO LL
16Ø5 B$=L$(A)+CHR$(13):IFLEN(A$)
+LEN(B$)>25ØTHEN A=A-1:A$=A$+STR
ING$(8Ø,Ø):C$="":GOTO162ØELSE A$
=A$+B$
161Ø LØ=LEN(A$):IFLØ<8ØTHEN163Ø
1615 C$=RIGHT$(A$,LØ-8Ø):A$=LEFT
$(A$,8Ø)
162Ø LSET MG$=A$:K2=K2+1:PUT#2,K
2:A$=""
1625 IFLEN(C$)>ØTHEN A$=C$:GOTO1
61Ø
163Ø NEXTA
1635 IF LEN(A$)=Ø THEN 1645
164Ø LSET MG$=A$+STRING$(8Ø,Ø):K
2=K2+1:PUT#2,K2
1645 LSET MG$=STRING$(8Ø,255):K2
=K2+1:PUT#2,K2
165Ø LSET H7$=MKN$(K2):PUT#1,K1+
1:PUT#1,1:IFK1<2THEN PUT#1,2:RE=
RE+1
1655 GOSUB86Ø:IF K8=1 THEN RETUR
N ELSE CLOSE:RETURN
166Ø PRINT:PRINT:IF PR=255 THEN
1662 ELSE IF BA$=""THEN P9=1:GOS
UB339:GOSUB25:GOTO166Ø
1662 PRINT:PRINT:PRINT"BOARD # O
R '?' ";
1665 CLOSE:GOSUB65Ø:IF CH$="?" T
HEN 1675 ELSE CH=VAL(CH$)
167Ø IF CH>-1 AND CH<255 THEN IF
 PR=255 THEN  BL=CH ELSE 168Ø
1671 RETURN
1675 GOSUB339:GOTO166Ø
168Ø FOR X=1 TO LEN(BA$)
1685 IF ASC(MID$(BA$,X,1))=CH TH
EN BL=CH:GOTO1671
169Ø NEXT X:PRINT"NOT AUTHORIZED
 TO";CH
1695 GOTO166Ø
19ØØ REM
19Ø5 '
191Ø '
192Ø PRINT:PRINT"KILL MESSAGE #"
;
1925 GOSUB65Ø:X=VAL(CH$):IFX=ØTH
ENRETURN
193Ø GOSUB9ØØØ:PRINT
1935 FOR RØ=2 TO K1:GET#1,RØ:IFC
VN(H1$)=X THEN 1945
194Ø NEXT RØ:PRINT"UNABLE TO LOC
ATE OR ALREADY DELETED.":CLOSE:R
ETURN
1945 F7=1:GOSUB236Ø:PRINT:F7=Ø
195Ø IF NA$=MT$ OR NA$=MF$ OR PR
=255 THEN 196Ø ELSE PRINT"YOU AR
E NOT AUTHORIZED!":PRINT:CLOSE:R
ETURN
196Ø F$=H2$:GOSUB15Ø:MID$(F$,2,1
)="1":GOSUB175
1965 LSET H2$=F$:PUT#1,RØ
197Ø CLOSE:RETURN
2ØØØ REM
2Ø05 F5=Ø:K9=Ø:UX=UL
2Ø1Ø TT$="FRNIA"
2Ø2Ø PRINTCHR$(12):CLS:PRINT:PRI
NT:IF UX>2 THEN 2Ø5ØELSEPRINT"
   MESSAGE BASE:"
2Ø25 PRINTTAB(5)"<F>ORWARD":GOSU
B1165:IFK>ØTHEN2Ø6Ø
2Ø3Ø PRINTTAB(5)"<R>EVERSE":PRIN
TTAB(5)"<N>EW":GOSUB1165:IFK>ØTH
EN2Ø6Ø
2Ø4Ø PRINTTAB(5)"<I>NDIVIDUAL":P
RINTTAB(5)"<A>BORT TO MENU"
2Ø5Ø PRINTTAB(5)"COMMAND: ";
2Ø55 GOSUB625:IFCH$=CHR$(13)THEN
 UX=1:GOTO2Ø2ØELSEK=INSTR("FRNIA
",CH$):IF K=Ø THEN2Ø55ELSEPRINTC
H$
2Ø6Ø ON K GOTO 2Ø7Ø,2Ø7Ø,2Ø7Ø,2Ø
7Ø,2Ø65
2Ø65 UX=UL:CLOSE:RETURN
2Ø7Ø GOSUB9ØØØ:GOSUB91ØØ:GET#1,1
:RE=CVN(H1$)
2Ø8Ø FORB=2 TO K1:GET#1,B:F$=H2$
:GOSUB15Ø:IFMID$(F$,2,1)<>"1"THE
N RS=CVN(H1$):GOTO2Ø83 ELSE NEXT
 B:RS=RE
2Ø83 IFK=3THENGOSUB228Ø:GOTO219Ø
2Ø85 PRINT:PRINT"SYSTEM HAS MESS
AGES";RS;"TO";RE:EXEC&H1ØDA
2Ø9Ø PRINT"ENTER STARTING MESSAG
E: ";:K1=LOF(1)
2Ø95 GOSUB65Ø:R=VAL(CH$):IF R=Ø
THEN21Ø5
21ØØ IF R<RS OR R>RE THEN CLOSE:
RETURN
```

```
21Ø5 IF K=4 AND R=Ø THEN21ØØELSE
GOSUB228Ø:ON K GOTO 211Ø,2135,21
9Ø,216Ø
211Ø REM
2112 IF R=Ø THEN R=RS
2115 FOR RØ=2 TO K1
212Ø GET#1,RØ:RH=CVN(H1$):IF RH<
R THEN 213Ø
2125 GOSUB23ØØ:IFF6=1THENGOSUB22
2Ø
213Ø IFK9=1THEN2185ELSENEXT RØ:G
OTO2185
2135 REM
2137 IF R=Ø THEN R=RE
214Ø FOR RØ=K1 TO 2 STEP -1
2145 GET#1,RØ:RH=CVN(H1$):IF RH>
R THEN 2155
215Ø GOSUB23ØØ:IFF6=1THENGOSUB22
2Ø
2155 IFK9=1THEN2185ELSENEXT RØ:G
OTO2185
216Ø REM
2162 IF R=Ø THEN CLOSE:RETURN
2165 FOR RØ=2 TO K1
217Ø GET#1,RØ:RH=CVN(H1$):IF RH<
>R THEN 218Ø
2175 GOSUB23ØØ:IFF6=1THENGOSUB22
2Ø:GOTO2186ELSE2186
218Ø NEXT RØ:PRINT"MESSAGE NOT A
VAILABLE":GOTO2186
2185 IFF5=ØTHENPRINT:PRINT"NO ME
SSAGES FOUND!":PRINT
2186 CLOSE:RETURN
219Ø REM
2195 IF LM=>RE THEN2185
22ØØ FOR RØ=2 TO K1
22Ø5 GET#1,RØ:RH=CVN(H1$):IF RH<
=LM THEN 2215
221Ø GOSUB23ØØ:IFF6=1THENGOSUB22
2Ø
2215 IFK9=1THEN2185ELSENEXT RØ:G
OTO2185
222Ø REM
2225 IFF4=ØTHENRETURNELSEPRINT:G
OSUB1ØØ:PRINT"[N]EXT [R]EAD [S]T
OP ";
223Ø GOSUB625:IFCH$="D"ANDPR>25Ø
THEN T$="D":GOSUB2455 ELSE K=INS
TR("NRS",CH$):IFK=ØTHEN223ØELSEP
RINTCH$:PRINT:PRINT
2235 ON K GOTO 224Ø,225Ø,226Ø:GO
TO223Ø
224Ø RETURN
225Ø GOSUB2395:RETURN
226Ø K9=1:RETURN
228Ø PRINT:IFUL>2THENPRINT"PAUSE
? ";:ELSEPRINT"PAUSE AFTER EACH
MESSAGE? ";
2285 GOSUB625:C=INSTR("YN",CH$):
ON C+1 GOTO 2285,2295,229Ø
229Ø PS=1:PRINT"NO":RETURN
2295 PS=Ø:PRINT"YES":RETURN
23ØØ REM
23Ø5 K9=Ø:F4=Ø
231Ø ON GØ+1 GOTO 2315,2315,232Ø
,232Ø
2315 IF ASC(H8$)<>BQ THEN RETURN
 ELSE 2335
232Ø IF ASC(H8$)<=BQ THEN 2335 E
LSE233Ø
2325 '
233Ø RETURN
2335 F$=H2$:GOSUB15Ø:IFMID$(F$,2
,1)="1"THENRETURN
2345 GOSUB25:F4=Ø:BR=ASC(H8$):R1
=CVN(H6$):R2=CVN(H7$):GET#2,R1:I
FVAL(MG$)<>RH THEN RETURN
236Ø MØ=CVN(H1$):KF=INSTR(H5$,NU
$):MF$=LEFT$(H5$,KF-1)
2365 KT=INSTR(KF+1,H5$,NU$):XX=K
T-KF-1:IFXX<1THEN MT$="":GOTO237
ØELSE MT$=MID$(H5$,KF+1,XX):IFF7
=1THENRETURN
237Ø KS=INSTR(KT+2,H5$,NU$):XX=K
S-KT-1:IFXX<1THEN MS$="":GOTO237
1ELSE MS$=MID$(H5$,KT+1,XX):PF=Ø
2371 FOR YY=1 TO 5:PRINT:NEXT YY
:CLS:GOSUB8ØØ:IFMID$(F$,1,1)="1"
ORML=1THEN IF MT$=NA$ OR MF$=NA$
 OR MQ=1 OR GØ=1 OR GØ=3 THENPRI
NT:PF=1:PRINT"PRIVATE":GOTO2375E
LSERETURN
2374 PRINT:PRINT
2375 IFMID$(F$,3,1)="1"THEN X$="
 <RCVD>" ELSE IF NA$=MT$ THEN MI
D$(F$,3,1)="1":GOSUB175:LSET H2$
=F$:PUT#1,RØ:GOSUB15Ø:X$=" <RCVD
>"ELSEX$=""
2381 F5=1:GOSUB25:PRINT"MESSAGE
#";MØ;" *";ASC(H8$);"*"
2385 PRINTRIGHT$(STR$(ASC(LEFT$(
H3$,1))),2);"/";RIGHT$(STR$(ASC(
MID$(H3$,2,1))),2);"/";RIGHT$(ST
R$(ASC(RIGHT$(H3$,1))),2);"    "
;STR$(ASC(LEFT$(H4$,1)));":";:A$
=RIGHT$(STR$(ASC(RIGHT$(H4$,1)))
,2):IFVAL(A$)>9THENPRINTA$ELSEMI
D$(A$,1,1)="Ø":PRINTA$
239Ø PRINT"FROM: ";MF$:PRINT"  T
O: ";MT$;X$:F5=1:F4=1:PRINT"SUBJ
: ";MS$:PRINT:IFF6=1THENRETURN
2395 FOR C=R1+1 TO R2
24ØØ GET#2,C:A$=MG$:IFA$=STRING$
(8Ø,255)THEN2425
24Ø5 GOSUB25:KX$="NS"+CHR$(13)
24Ø6 K=INSTR(A$,CHR$(Ø)):IFK>1TH
EN A$=LEFT$(A$,K-1)
24Ø8 FOR CC=1 TO LEN(A$) STEP 1Ø
241Ø PRINTMID$(A$,CC,1Ø);:EXEC&H
1ØDA:CH$=CHR$(PEEK(4481)):GOSUB2
```

```
5:IF CH$=CHR$(Ø)THEN2416
2412 K=INSTR("NSP",CH$):IF K=Ø T
HEN K=INSTR("nsp",CH$):IF K=Ø TH
EN 2416
2414 ON K+1 GOTO 2416,245Ø,2465
2415 EXEC&H1ØDA:IFPEEK(4481)<>13
THEN2415
2416 NEXTCC:GOSUB25
242Ø NEXTC
2425 PRINT:PRINT:GOSUB1ØØ:IFPS=Ø
THEN243ØELSE IF MF$=NA$ OR MT$=N
A$ THEN 243ØELSE245Ø
243Ø GOSUB25:IF MF$=NA$ OR MT$=N
A$ OR PR=>25Ø THEN T$="DN":PRINT
"<D>ELETE <N>EXT ";:ELSET$=CHR$(
255)+"N":PRINT"<N>EXT ";
2431 IF PR=>PL THEN T$=T$+"RS":P
RINT"<R>EPLY <S>TOP ";:ELSE T$=T
$+CHR$(255)+"S":PRINT"<S>TOP ";
2435 GOSUB25
244Ø GOSUB625:K=INSTR(T$,CH$)
2445 IF K=Ø THEN 244ØELSEPRINTCH
$:ON K GOTO 2455,245Ø,2459,2465
245Ø IF ZK>ØANDMT$=NA$ THEN2455E
LSERETURN
2455 IFLEFT$(T$,1)="D"THEN MID$(
F$,2,1)="1":GOSUB175:LSET H2$=F$
:PUT#1,RØ
2456 PRINT"REPLY (Y/N)? ";:GOSUB
6ØØ:IFCH$="Y"ORCH$="y"THENPRINT"
YES":GOTO246ØELSEPRINT"NO":RETUR
N
2459 IF ZK>ØANDMT$=NA$ THEN MID$
(F$,2,1)="1":GOSUB175:LSET H2$=F
$:PUT#1,RØ
246Ø IFPR<PL THENPRINT:PRINT"YOU
 ARE NOT AUTHORIZED!":RETURN ELS
E PRINT:BL=BR:K8=1:MT$=MF$:MF$=N
A$:IFBR>249THEN BL=255:GOSUB166Ø
:GOSUB9ØØØ:GOSUB91ØØ:IFBL=255OR
CH$="" THEN K8=Ø:RETURN
2461 IFLEFT$(MS$,9)="REPLY TO "
THEN 2462 ELSE MS$="REPLY TO "+M
S$
2462 GOSUB1445:K8=Ø:RETURN
2465 K9=1:IFZK>ØANDMT$=NA$ THEN2
455ELSERETURN
7ØØØ REM
7ØØ5 '
7Ø1Ø UNLOAD
7Ø15 T$=RIGHT$(D$,5):BU=VAL(LEFT
$(T$,1)):E$=RIGHT$(T$,4):PRINT:P
RINT:FT$=X$+E$+"/TXT":D=-1
7Ø25 RUN T 7Ø7Ø
7Ø3Ø D=D+1:IFD>3THEN RUNT7Ø7Ø:RE
TURN
7Ø35 F$=FT$+":"+DR$(D)
7Ø4Ø RUNT7Ø3Ø:OPEN"I",#1,F$:IF B
=1THENPRINTCHR$(18);
7Ø41 RUN T 7Ø6Ø
7Ø45 IFEOF(1)THEN7Ø6ØELSEGOSUB25
:LINEINPUT#1,S$:GOSUB25
7Ø5Ø PRINTS$:EXEC&H1ØDA:CH$=CHR$
(PEEK(4481)):GOSUB25:IFCH$="S"OR
CH$="s"THEN7Ø6Ø
7Ø56 K=INSTR("Pp",CH$):IFK=ØTHEN
7Ø45
7Ø57 EXEC&H1ØDA:GOSUB25:IFPEEK(4
481)<>13THEN7Ø57ELSE7Ø45
7Ø6Ø RUNT7Ø7Ø:CLOSE:IF B=1THENPR
INTCHR$(2Ø):RETURNELSERETURN
7Ø65 D=D+1:IF D>3 THEN CLOSE::PR
INT:PRINT"FILE NOT AVAIABLE":RET
URN ELSE7Ø25
7Ø7Ø GOTO1ØØØ
71ØØ REM
71Ø5 '
7110 '
7115 PRINT:PRINT:PRINT"PAGING SY
SOP..."
712Ø FORC=1TO15:IFINKEY$<>""THEN
713ØELSEPRINT".";:SOUND15Ø,5:NEX
TC:GOTO7125
7125 PRINT:PRINT"SYSOP HAS BEEN
PAGED. PLEASE":PRINT"CONTINUE TO
 USE THE SYSTEM.":PRINT"HE WILL
BREAK IN IF AVAILABLE.":RETURN
713Ø GOSUB86Ø:GOSUB88Ø:POKE65494
,Ø:PRINT:PRINT:PRINT"SYSOP ENTER
ING CHAT"
7135 LINEINPUT">";A$:IFA$="\"THE
NRETURN
7145 GOTO7135
9ØØØ REM
9Ø1Ø F$="HDR/SYS:"+DR$(PEEK(467Ø
))
9Ø15 OPEN"D",#1,F$,11Ø:FIELD#1,5
 AS H1$,1 AS H2$,3 AS H3$,2 AS H
4$,8Ø AS H5$,5 AS H6$,5 AS H7$,1
 AS H8$,8 AS SP$:K1=LOF(1):RETUR
N
91ØØ REM
911Ø F$="MSG/SYS:"+DR$(PEEK(4671
))
9115 OPEN"D",#2,F$,8Ø:FIELD#2,8Ø
 AS MG$:K2=LOF(2):RETURN
92ØØ REM
92Ø5 F$="USERL/SYS:"+DR$(PEEK(46
72))
921Ø OPEN"D",#1,F$,96
9215 FIELD#1,5Ø AS UN$,8 AS UP$,
1 AS UR$,1 AS UA$,1 AS U1$,1 AS
U2$, 1 AS UU$,1 AS UE$,5 AS U5$,
5 AS UM$,5 AS UD$,1 AS U3$,1 AS
U4$,1 AS UO$,1 AS UL$,4 AS US$,9
 AS SP$
922Ø K1=LOF(1):RETURN
94ØØ REM
```

```
94Ø5 FF$="MENU/SYS:"+DR$(PEEK(46
73)):OPEN"D",#1,FF$,25Ø
941Ø FIELD#1,1 AS M1$,1 AS M2$,1
 AS M3$,1 AS M4$,16 AS M5$,2ØØ,A
S M6$,1 AS M7$,29 AS M8$:K1=LOF(
1):RETURN
95ØØ REM
95Ø5 '
951Ø GOSUB92ØØ:GET#1,1
952Ø K=INSTR(UN$,CHR$(Ø)):NA$=LE
FT$(UN$,K-1):PR=ASC(UA$):F$=U1$:
GOSUB15Ø:P1$=F$:F$=U2$:GOSUB15Ø:
P2$=F$:UL=ASC(UL$):IF UL<1ORUL>4
THENUL=1
955Ø BC=ASC(UE$):LM=CVN(UM$):POK
E4619,ASC(UO$):POKE4628,UL:POKE4
618,BC:POKE462Ø,PR:POKE4621,ASC(
U1$):POKE4622,ASC(U2$):CLOSE:RET
URN
96ØØ REM
96Ø5 TR$=TR$+BN$+"("+CH$+") "
961Ø IFLEN(TR$)<64THENRETURN
9615 FR=PEEK(4669):RUNT967Ø:IFFR
=ØTHEN9635ELSEIFFR=1THEN964ØELSE
IFFR=3THEN966ØELSECLOSE:F$="TRAC
ER/SYS:"+DR$(PEEK(4675)):OPEN"D"
,#1,F$,128
9616 FR=FREE(PEEK(4675)):IFFR<3T
HENCLOSE:RETURN
962Ø FIELD#1,128 AS X$:K1=LOF(1)
:K1=K1+1:LSET X$=TR$+CHR$(Ø):PUT
#1,K1
9635 CLOSE:TR$="":RETURN
964Ø OPEN"O",-1,"TRACER/SYS":PRI
NT#-1,TR$:GOTO9635
966Ø PT=PEEK(&HFF22)AND1:IFPT=1T
HEN9635ELSEPRINT#-2,TR$:GOTO9635
967Ø POKE4669,Ø:GOTO1ØØØ
98ØØ REM
98Ø5 '
981Ø RUN T 984Ø:FORA=1TO1Ø:PRINT
:NEXTA
9815 F3=1:GOSUB1ØØ:GOSUB71Ø:TR$=
TR$+CHR$(13)+TI$:GOSUB9615
982Ø PRINT:PRINT:PRINT"THANKS FO
R CALLING COBBS."
9825 PRINT"PLEASE CALL AGAIN..."
:PRINT:PRINT
983Ø POKE4615,HR:POKE4616,MN:POK
E4617,SS
984Ø PRINT"PLEASE HANG UP NOW.":
POKE65386,1Ø6:LOAD"USER/SYS",R
1ØØØØ POKE&HAC7D,&HØE:POKE&HAC7E
,&HØØ:MOTOROFF
1ØØ5 X=&H112B:POKE X,&H86:POKE
X+1,&HØD:POKE X+2,&HB7:POKE X+3,
&H11:POKE X+4,&H81:POKE &H1127,&
H1Ø
1Ø1Ø RUN
```

HOME HELP

32K ECB

Little Black Book

By Eric Hartley

Have you seen those disk address/phone number programs, but don't have a disk drive and can't find a good cassette program? Would you like a savable phone number system that doesn't leave data files everywhere, which can cause a "terminal" case of I/O Errors? If your answer to both questions is "yes", *Little Black Book* is for you!

Little Black Book is designed to save up to 100 four-line entries on any 32K ECB computer, and save data along with the program as one file. This is the key to the program.

Before using, choose a password for INIT and DEL (or use BLACK, which is the original password). Choose your password carefully because once the system is initialized, a change in the password can destroy all data. Place your password between the quotation marks, in place of BLACK, in Line 100. Next, if you have a printer width greater than 32, delete the semicolon from the end of lines 890-920. Finally, run and initialize the data file. You are now ready to operate the system.

To get maximum usage from *Little Black Book*, first familiarize yourself with the commands and their functions. These commands are as follows:

HELP	—	List commands and give brief summary
ENTER	—	Allow entry of data into file
LIST	—	Display formatted data on screen
PRINT	—	Display formatted data on printer
FORMAT	—	Choose what part of the file is to be displayed
EDIT	—	Edit specific entry
SEARCH	—	Search for specified string in file
SAVE	—	Save program and data as one on tape
DEL	—	Delete specific entry (requires password)
INIT	—	Initialize Data File (requires password)

All of the commands are self-explanatory, except EDIT. The instructions for this command are: 1) Enter *all* of the name line, 2) If data is correct, press ENTER and 3) If change is desired, type line with changes and press ENTER. □

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 $4.50 including postage and packing. Please enclose your cheque / money order made payable to: Australian Rainbow Magazine, PO Box 1742, Southport, 4215.

RATES

AUSTRALIAN CoCo / MiCo / softgold		AUSTRALIAN RAINBOW
☐ $3.45	Latest per copy	$4.50 ☐
☐ $19	6 months	$24.75 ☐
☐ $31	12 months	$39.95 ☐

BACK ISSUES

MiCo — First Issue Oct '83 ☐ $3/copy

MiCoOz (MiCo on Tape) $8 ea ☐ First Issue Dec '83

CoCo / MiCo / softgold First Issue Aug '84 ☐ $3.45/copy

AUSTRALIAN RAINBOW

Feb '85-Apr '85 ☐ $3.95

Aug '84-Jan '85 ☐ $3.25

to July '84 ☐ $3.00

DISKS & TAPES

CoCoOz (Aust CoCo on Tape)

☐ Latest $8

☐ 6 months $42

☐ 12 months $75

First Issue Mar '83

or Debit my Credit Card Monthly ☐

Blank tapes 12 for $18 or $1.70 ea ☐

Cassette Cases 10 for $5 ☐

Disks — $3.50 ea ☐ 10 for $29.99 ☐

Rainbow on Tape (program listings) $12 for month of ☐

Please note that RAINBOW on TAPE is issued irregularly

or Annually $144 ☐ or Debit my Credit Card Monthly ☐

BOOKS

Byte

Elementary $5.95 ☐

Help

Medium $9.95 ☐

Facts

Advanced $11.95 ☐

MiCo Help (MC-10 computer).

Medium $9.95 ☐

PLUS

☐ Best of CoCoOz # 1 (Education)
☐ Best of CoCoOz # 2 (Games)
☐ Part 1 for 16K ECB
☐ Part 2 for 32K ECB (Some 16k CB on P1)

TAPE $ 10.00, DISK $ 21.95

BULLETIN BOARD

Let your computer talk to ours!
CoCoLink — Annual Sub $29 ☐

☐ VISA ☐ BANKCARD ☐ MASTERCARD

Signature

☐ CASH
☐ CHEQUE
☐ MONEY ORDER

Authorised

Amount $ _____

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

New Subscription ☐ Renewal ☐

GOLDSOFT

Hardware & Software for your TANDY computer.

HARDWARE		
The CoCoConnection: Connect your CoCo to the real world and control robots, models, experiments, burglar alarms, water reticulation systems — most electrical things. Features two MC 6821 PIAs; provides four porgrammable ports; each port provides eight lines, which can be programmed as an input or output; comes complete with tutorial documentation and software; supplied with LED demonstration unit; switchable memory addressing allows use with disk controller or other modules via a multipack interface; plugs into Cartridge Slot or Multipack, uses gold plate connectors; a MUST for the hardware designer and debugger!		$195.00
Video-Amp: Connects simply to your CoCo to drive a Colour or Mono monitor.	With instructions With instructions and sound	$25.00 $35.00
SOFTWARE		
CoCoOz: The programs you see listed in Australian CoCo Magazine are available on CoCoOz! No laborious typing — just (C)LOAD and Go!	Each Tape Subscription, 6 months 12 months **NEW** for 1986 ONLY Each DISK Subscription on disk, 12 months	$8.00 $42.00 $75.00 $10.95 $102.50
Rainbow on Tape: Australian. The programs you see listed in Australian Rainbow Magazine are available on tape. A boon if you don't understand the language! American. We also supply the programs found in American Rainbow on tape. Please specify either Australian or American.	Each Tape Subscription, 12 months **NEW** for 1986 ONLY Each DISK Subscription on disk, 12 months	$12.00 $144.00 $15.00 $172.00
MiCoOz: The programs in the MiCo section of Australian CoCo Magazine. (For MC 10 computers only) Back issues of CoCoOz and MiCoOz are always available on tape only.	Each Tape Subscription, 12 months	$8.00 $75.00
To be released soon: **GOLDDISK 1000** — programs from 'softgold' for your Tandy 1000 on disk, and, **GOLDDISK 4** — programs from 'softgold' for your Tandy Model 4!		
The Best of CoCoOz: 1 . . . EDUCATION programs. Fourteen programs for the teacher or parent. 2 . . . Part 1 . . . 16K GAMES. (Mainly ECB). Sixteen programs to keep you on your toes! Part 2 . . . 32K GAMES. Adventures, simulations and arcade games for everyone! 3 . . . UTILITIES. Programs to make your use of the computer easier. Spoolers, Reverse Video, Disk utilities and more. 4 . . . BUSINESS programs. Invoicing, Accounts, Creditors and more.	Tape Disk 3 on Disk Any two tapes	$10.00 $21.95 $16.00 $17.95
CoCoLink: CoCoLink is our Bulletin Board which you can access with any computer if you have a 300 baud modem and a suitable terminal program. There is a free visitor's facility, alternatively membership entitles you to greater access of the many files available. We can also be contacted through Minerva (OTC) and Viatel (Telecom).	Subscription to CoCoLink, 12 months	$29.00
Magazines: Australian Rainbow Magazine — THE magazine for advanced CoCo users! Australian CoCo Magazine — THE magazine for the new user of a Tandy computer. Also suits owners of CoCos, MC 10s, Tandy 1000s, 100s, 200s & 2000s.		
Back Issues: Australian Rainbow Magazine. (Dec '81 to now.) Australian CoCo Magazine. (Aug '84 to now.) CoCoBug Magazine. For CoCo — usually 8 programs in each magazine. (Sep '84 to Oct '85) Australian MiCo Magazine. For Tandy MC 10 computers. (Dec '83 to Jul '84) Australian GoCo Magazine. For Tandy Model 100 users. (Jul '83 to Jul '84) See Subscription Page for further details.		
Books: HELP: A quick reference guide for CoCo users. FACTS: THE reference for CoCo machine language programmers. MiCo HELP: A quick reference for owners of MC 10 computers.		$9.95 $11.95 $9.95
Othello: by Darryl Berry The board game for your CoCo.	Tape 16K ECB	$15.95
Say the Wordz: by Oz Wiz & Pixel Software Two curriculum based speller programs for your Tandy Speech/Sound Pack.	Tape 32K ECB	$39.95
Bric a Brac: Blank tapes . . . 12 for $18.00 or $1.70 each. Cassette cases . . . 15 for $5.00. Disks . . . (they work!) $3.50 each or $28.95 per box of 10.		

HOW TO ORDER

Option 1: Use the subscription form in this magazine.
Option 2: Phone and have ready your Bankcard, Mastercard or Visa number.
Option 3: Leave an order on Viatel, Minerva or CoColink, but be sure to include your Name, Address, Phone Number, Credit Card Number and a clear indication of what you require, plus the amount of money you are authorising us to bill you.

user group CONTACTS

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

ADELAIDE	JOHN HAINES	08 278 3560
ADELAIDE NTH STUN	EISENBERG	08 250 6214
ALBURY	RON DUNCAN	060 43 1031
ARMIDALE	TOM STUART	067 72 8162
BAIRNSDALE	COLIN LEHMANN	051 57 1545
BALLARAT	MARK BEVELANDER	053 32 6733
BANKSTOWN	CARL STERN	02 646 3619
BLACKTOWN	KEITH GALLAGHER	02-627-4627
BLACKWATER	ANNIE MEIJER	079.82.6931
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
SOUTH	PATRICK SIMONIS	07 209 3177
SOUTH WEST	GRAHAM BUTCHER	07 376 3400
WEST	BRIAN DOUGAN	07 300 2072
BROKEN HILL	DEAN PARADICE	080 6701
BUNDABERG	JIM McPHERSON	071 72 8329
CAMDEN	KEVIN WINTERS	046.66.8068
CAMPBELLTOWN	LEO GINLEY	02 605 4572
CANBERRA NTH	JOHN BURGER	062 58 3924
CANBERRA STH	LES THURBON	062 88 9226
CHATSWOOD	BILL O'DONNELL	02 411 3336
' or	MARK ROTHWELL	02 817 4627
CHURCHILL	GEOFF SPOWART	051 22 1389
COLYTON TEENS	DWAYNE MANSON	02 623 5805
COOMA	ROSS PRATT	0648 23 065
COORANBONG	GEORGE SAVAGE	049 77 1054
DALBY	ANDREW B. SIMPSON	074.62.3228
DARWIN	BRENTON PRIOR	089.81.7766
DENILIQUIN	WAYNE PATTERSON	058 81 3014
DUBBO	GRAEME CLARKE	068 89 2095
EMERALD	LEIGH EAMES	059 68 3392
FORBES	JOHANNA VAGG	068 52 2943
FORSTER	GARY BAILEY	065 54 5029
GIPPSLAND STH	PAT KERMODE	056 74 4583
GOLD COAST	GRAHAM MORPHETT	075 51 0015
GOSFORD	PETER SEIFERT	043 32 7874
GOULBURN VALLEY	TONY HILLIS	058 59 2251
GREENACRES	BETTY LITTLE	08 261 4083
GUYRA	MICHAEL J. HARTMANN	067 79 7547
HASTINGS	MICHEAL MONCK	059.86.8288
HERVEY BAY	LESLEY HORWOOD	071 22 4989
HILLS DIST	DENNIS CONROY	02 671 4065
HOBART	BOB DELBOURGO	002 25 3896
HORNSBY	ATHALIE SMART	02 848 8830
IPSWICH	MILTON ROWE	07 281 4059
JUNEE	PAUL MALONEY	069 24 1860
KALGOORLIE	TERRY BURNETT	090.21.5212
KENMORE	GRAHAM BUTCHER	07 376 3400
KINGSTON	WIM DE PUIT	002 29 4950
LEETON	CHRIS NAGEL	069 53 2969
LITHGOW	DAVID BERGER	063 52 2282
LIVERPOOL	LEONIE DUGGAN	02-607-3791
MACKAY	LEN MALONEY	079511333x782
MacQUARIEFIELDS	KIETH ROACH	02 618 2858
MAFFRA	MAX HUCKERBY	051 45 4315
MAITLAND	LYN DAWSON	049 49 8144
MARYBOROUGH	NORM WINN	071 21 6638
MELBOURNE:		
' DANDENONG	DAVID HORROCKS	03 793 5157
' DONCASTER	JUSTIN LIPTON	03 857 5149
' FRANKSTON	BOB HAYTER	03.783.9748
' MELTON	MARIO GERADA	03 743 1323
' RINGWOOD	ANDREW RAWLINGS	03 700 4498
' SUNBURY	JACK SMIT	03.744.1355
MILDURA	SCOTT HOWISON	050 23 6016
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
NARROMINE	GRAEME CLARKE	068 89 2095
NEWCASTLE	LYN DAWSON	049 49 8144
NOWRA	ROY LOPEZ	044 48 7031
ORANGE	STEVE LOVETT	063.62.4025
' or	JIM JAMES	063 62 8625
PARKES	DAVID SMALL	068 62 2682
PENRITH	ALEX SCHOFIELD	047 31 5303
PERTH	IAN MACLEOD	09 448 2136
PORT LINCOLN	JOHN BOARDMAN	086 82 2385
PORT MacQUARIE	RON LALOR	065 83 8223
PORT NOARLUNGA	ROB DALZELL	08 386 1647
PORT PIRIE	KEVIN GOWAN	086 32 1368
ROCKHAMPTON	KEIRAN SIMPSON	079 28 6162
ROSEVILLE	KEN UZZELL	02 467 1619
SALE	BRYAN McHUGH	051 44 4792
SANDGATE	MARK HIGHELL	07 269 5090
SCARBOROUGH	PETER MAY	07 203 6723
SEACOMBE HTS	GLENN DAVIS	08 296 7477
SHEPPARTON	ROSS FARRAR	058 25 1007
SMYTHESDALE	TONY PATTERSON	053 42 8815
SPRINGWOOD	DAVID SEAMONS	047 51 2107
STURT	MARY DAVIS	08 296 7477
SUTHERLAND	IAN ANNABEL	02 528 3391
SWAN HILL	BARRIE GERRAND	050.32.2838
SYDNEY TEENS	ROD HOSKINSON	02 48 5948
SYDNEY EAST	JACKY COCKINOS	02 344 9111
TAMWORTH	ROBERT WEBB	067 65 7256
TAHMOOR	GARY SYLVESTER	046 81 9318
TARA	STEVEN YOUNGBERRY	
TONGALLA	TONY HILLIS	058 59 2251
TOOWOOMBA	GRAHAM BURGESS	076 30 4254
TOWNSVILLE	JOHN O'CALLAGHAN	077 73 2064
TRARALGON	MORRIS GRADY	051 66 1331
UPPER HUNTER	TERRY GRAVOLIN	065 45 1698
WAGGA WAGGA	CES JENKINSON	069 25 2263
WONTHAGGI	PAT KERMODE	056 74 4583
WYNYARD	ANDREW WYLLIE	004 35 1839
YARRAWONGA	KEN SPONG	057 44 1488

SPECIAL INTEREST GROUPS

' BUSINESS

BRIZBIZ BRIAN	BERE-STREETER	07 349 4696

' TDP - TELEWRITER, DYNACALC PROCOLOR

BRISBANE	GEOFF TOLPUTT	07 44 6084

' OS9 GROUPS

BANKSTOWN	CARL STERN	02 646 3619
BRISBANE	JACK FRICKER	07 262 8869
COOMA	FRED BISSELING	0648 23263
KALGOORLIE	TERRY BURNETT	090.21.5212
PENRITH	BOB THOMSON	047 30 2468
SYDNEY EAST	JACKY COCKINOS	02.344.9111
SYDNEY NTH	MARK ROTHWELL	02 817 4627

' MiCo GROUPS

LITHGOW	DAVID BERGER	063 52 2282
PORT LINCOLN	JOHN BOARDMAN	086 82 2385
ROCKHAMPTON	TIM SHANK	079 28 1846
SYDNEY	RAJA VIJAY	02 519 4106

' TANDY 1000 / MS DOS

BRISBANE	BRIAN DOUGAN	07 30 2072
SYDNEY	ROGER RUTHEN	047.39.3903

' FORTH

BRISBANE	JOHN POXON	07 208 7820
PORT LINCOLN	JOHN BOARDMAN	086 82 2385
SYDNEY	JOHN REDMOND	02 85 3751

AUSTRALIAN RAINBOW MAGAZINE
REGISTERED BY AUSTRALIAN POST —
PUBLICATION NO. QBG 4009
AUSTRALIAN COCO/softgold
REGISTERED BY AUSTRALIAN POST —
PUBLICATION NO. QBG 4007.
PO BOX 1742,
SOUTHPORT. QLD. 4215.

POSTAGE
PAID
AUSTRALIA

AUSTRALIAN COCO

THIS MONTH

* TOPPLE
* RECOVERY
* GOING TROPPO
* BASIC
* QUICK DRAW
* XMAS CARD
* SCOREBOARD
* COCKTAILS
* MICO
* SOFTGOLD
* AND MUCH MORE!

SO HURRY TO YOUR LOCAL NEWS STAND OR TANDY STORE NOW! BEFORE THEY ARE SOLDOUT!