TISHUG NEWS DIGEST

Focusing on the TI99/4A Home Computer

Volume 10, Number 6 July, 1991

Registered by Australia Post - Publication No. NBH5933

Come to the Buy, Swap and

Sell. Bargains for everyone.

Sydney, New South Wales, Australia $3

TIsHUG (Australia) Ltd
A.C.N. 003 374 383

TIsHUG News Digest

July 1991

All correspondence to:

P.O. Box 214
Redfern, NSW 2016
Australia

Membership and Subscriptions

Annual Family Dues	$30.00
Associate membership	$10.00
Overseas Airmail Dues	A$60.00
Overseas Surface Mail Dues	A$45.00

TIsHUG Sydney Meeting

The next meeting will start at 2.00 pm on 6th of July at Ryde Infant School, Tucker Street, Ryde. At 9 am, before the meeting, there will be a meeting of all those wishing to be involved in the train project followed at 12 pm by the beginners' Editor Assembler class for all those interested.

Printed by
The University of Wollongong
Printery Services.

TIsHUG News Digest ISSN 0819-1984

Index

TIsHUG Fairware Author of the Month

The Fairware Author this month is A.N. Onymous, for all the great public domain software that you would like to see written. All donations collected at the meeting and sent in will be put into a trust fund to provide an incentive for local software authors.

Editor's Comment

by Bob Relyea

IT hardly seems possible that we are already in the middle of the year and the club has entered its second decade of service and fun. I say 'fun' because I like working with the TI and it is 'fun' to me. Last week I helped my daughter, Rebecca, do a project for her university studies in which I had to do the formatting and the tricky bits on the computer after she typed it all in. She was amazed at how the TI manipulated all the data and she ended up with just what she wanted. I had fun doing it all and look forward to every moment that I am on the computer learning something new.

We could use a few more local articles to go with the current series that we are putting in the magazine. Do not let the same few do it all - have a go and put something in yourself. Remember, there are a lot of people out there at the same level as you and would appreciate hearing from you. o

Co-ordinator's Report

by Dick Warburton

I was delighted at the last monthly meeting to see the range of activities available. The assembly class was running as usual, the shop was open, even though Percy was overseas, a keen group of ramdisk fanciers were sorting out their difficulties together, two games machines were up and running, the train set was on display, the bulletin board was accessed (until one member, unnamed, used the computer to sort out a problem), and a rather keen group stayed late to discuss the development of a computer interface for a model train set. The library was open as usual, but no directors' meeting was held. The shop was busy, and helped a number of people who had not known of the club's existence. The people I spoke to were generally surprised at the range of club activities. The Eprom Ramdisk project is well under way. Two excited younger members rang me a number of times on the Sunday, to tell me of their progress. There was great excitement as the ramdisks came alive. There seems to be a new mood of enthusiasm as members see the real advantages of fitting a ramdisk with eproms to a TI at relatively low cost. The way the kits are selling, they will not last too long. If you want to get in and really expand your system cheaply and effectively, make sure you do not miss out. If the interest continues, we will get another run of ramdisk cards and kits. More memory chips and eproms should be available at the next meeting. If you have specific software you would like put on eprom, See Percy Harrison at the shop, and put your order in. At the next meeting there will be a ramdisk first aid group, all ready to give the kiss of life to your ailing card. If you have a ramdisk, which is not working as well as you would like, bring it in, and we will try to get it going well. If the software is a bit strange, come and practice using it so you can be self reliant. There are times when ramdisks glitch without warning, and the ROS is corrupted. Fixing it is quite easy and usually only a minute's work, if you know how.

Another exciting project, is the development of a computerised train control system. This has a number of applications, and can be readily adapted to a number of other tasks. Ross tells me that the electronics are basically simple, and the circuits repetitive. The group to do this will meet on the Saturday morning at 9.00 AM. Tasks will be shared out among the group members, and a range of learning experiences will be available. There will be practical training in electronics, some programming, and a chance to play with trains. It will be really good fun. We may need to call on members to donate the occasional train or carriage to get it going.

Another project I would like to see developed this year, is the production of of our own 80 column cards. They are still dear to buy overseas, and I am sure that with our resources we could develop one. If it is feasible I will try get it going. Perhaps you still have a really basic system. Do not forget that the club shop sells an interface card to build a simple expansion system. You can add a multifunction card, a ramdisk,etc in your own time, at low cost. New disk drives are only $65 and power supplies are available cheaply. Be on the lookout for people who want to sell Peripheral Expansion boxes. They are selling quite cheaply now. If there is a demand for expansion systems and other peripherals, the club may actively advertise to buy them for members. An expanded TI with a ramdisk is a really useful computer. It perhaps is one of the easiest systems to use of all the computers available. It certainly will do what the bulk of home users need to do.

Another major aim this year is to find an alternative source of colour monitors at a reasonable price. If any member hears of analogue monitors up for sale, let me know immediately. If anyone has any ideas about places or people we can try, please let me know. The colour monitor improves the TI tremendously. If we intend to expand to 80 column cards, we will need more colour monitors. Do not forget that a technical group meets at Ryde school on the Monday night two weeks after the meeting. So far we have put together a number of monitor interfaces, and we try to fix simple faults in consoles. If you have a technical problem, bring it along, and we might be able to help you.

At future meetings, I intend to have the bulletin board operating and a member available to help other members who are interested. Games activities will be expanded. Regular groups will meet for the train project, assembly class, and for Eprom Ramdisks. If you know young people who are interested in computers, bring them along and introduce them to TI games. Hopefully activity will characterise future meetings. Do not forget the coming meeting is a buy swap and sell day. These are always popular. Come early and get a bargain. There may be some ramdisks for sale. Do not forget that Geoff Trott and Lou Amadio are usually available to help and advise with technical problems.

Well I will say 'bye for now, and I will see you at the next meeting.

Dick Warburton

TI99/4A Emulator

by Thomas Opheys, Germany

Last month (March 1991) I started to write a TI99/4A emulator for the IBM PC. The program will emulate the full hardware of the TI99/4A including VDP, ROM, RAM, GROM, disk controller, RS232 card, p-Code card and much more. You will be able to copy original TI99/4A files, including the system ROMs/GROMs and all ROM/GROM based modules via a null modem cable to the PC and execute everything.

Now for some information about my TI99/4A emulator. I would have answered all the mail I received, but at my university I am not allowed to mail that much outside Germany, because of the cost. So I try to answer all questions in the COMP.SYS.TI file, hoping not to bore those of you who are not interested.

The emulator will require the following hardware: an IBM PC, any version of DOS and a standard VGA card capable of displaying the 640x480x16 mode. The faster the machine, the better.

The emulator is 100% written in assembler, for speed reasons. Someone asked me if it was possible to port it to the Amiga. Now, you know the answer: naturally I could give you the source code, but it will be a full time job to rewrite it for an Amiga or any other machine. You will not get a good TI99/4A emulator written in any higher level language, not even c.

How will the emulator work? You will see the original TI99/4A screen on your PC monitor. While the TI99/4A graphics only uses 256x192 pixels, there is enough space to display more information on the PC screen. When running in debug mode, the emulator displays the processor data like registers, status, debugged assembler instructions, VDP and GROM addresses and lots more. In addition there is a display of the current GPL instruction being executed by the GPL interpreter. The rest of the screen is reserved for switches (on/off, load interrupt) and lamps (access to peripheral cards, interrupts). Naturally the performance of the emulator is not too good with such a huge amount of data displayed, so you can switch it to speed mode which just displays the TI99/4A screen. As I now see it, the emulator will be only slightly slower on a 20MHz 386 than on the original TI99/4A, but I can get it still faster.

The whole hardware is supported: keyboard, video display processor with all modes (text, graphics, bitmap, multicolor), sound processor (at the moment only one sound channel out of four, because of the

continued on page 17

Techo Time

with Geoff Trott

Consoles

The number of consoles in for repair has dropped over the last few months. In fact, some of those that reached me did not have anything wrong with them. After a rush of consoles requiring new processors, there have not been any for quite a while. Just as well, as I do not have any spare processors at the moment. One console I looked at recently, mis-behaved when the Extended BASIC cartridge was inserted, even to making the console tester come up with the wrong checksums for all the GROMs in the console. After taking the console apart, cleaning the connectors around the cartridge port and putting it all back together again, everything worked fine so I guess there was a bad connection which caused the effects. An interesting symptom!

Another console had a black screen, would not run the console tester and the signals around the processor made me suspect it. After taking it out it tested fine, so I started to look at the signals more closely. The high half of the data lines on the processor seemed to be held low (D0 to D7), so I started to look at the 16 to 8 bit interface area. I was turning on and off while I was doing this and one time the console tester started to run. This then showed that the checksum for the high byte system ROM was not correct. Changing this ROM caused all problems to disappear so it must have been corrupting the data bus. I was just lucky to get it to release the bus so that the console tester could run and tell me where the problem was.

The other console problems related to a PEbox system I was repairing. This started out as a problem with a CorComp disk controller card in the PEbox which would not load the game "Hitchhiker's Guide to the Galaxy" without error. I swapped the 2114 memory chip in the disk controller card and this seemed to fix it. Then I was testing the whole system, TI memory expansion, TI RS232 card and the disk controller when it started to mis-behave again. This time the memory expansion would not test OK nor would the RS232 and the disk controller just hung up. I have a single I/O interface which plugs into the I/O port of the console and allows me to test PEbox cards in isolation. Each of the cards worked on their own but not in the PEbox. Now the I/O interface needs power supplies for the PEbox cards, +8, +16 and -16 volts. These are supplied by a power supply which has a red light to show that it is on. Carelessly, I pulled the disk controller out with the console turned off but the other supply turned on. Oops, suddenly the console did not work and neither did the disk controller when I tried it on another console. I borrowed Rolf's disk controller card to do some chip swapping to find the problem and confirmed that it was one of the PAL chips which do most of the logic on the card. Back to that later, but first to the console. It would not run the console tester and looked rather dead. I removed the processor but it tested OK. Then I noticed that the 74LS138 that does the main memory decoding was not working. I changed this but it still did not work. I then had a look at the pins next to the power supplies on the PEbox cards. The +8 volts is at one end of the connector and is next to a ground pin and the READY(H) line. At the other end of the connector the +16 is at the end with -16 volt next and MEMEN(L) and CRUIN(H) on the next pins. Pulling out the card with the power on is likely to short out adjacent pins. With ground next to one of the +8 volt lines this would probably protect the READY(H) line. At the other end, -15 volts into MEMEN(L) would and does cause all sorts of problems and sure enough a 74LS32 gate on the MEMEN(L) line was dead (the 74LS138 is also connected to MEMEN(L)). Replacing this IC and the console came to life. The CRUIN(H) line goes straight into the processor but all seemed OK there. There is one more connection to MEMEN(L) which was also dead, an inverter (74LS04). I found this a bit later as it is required by the 8 bit bus. So I learned the hard way to be careful about removing cards with the power on. Unfortunately, I did the same thing to Rolf's card and my console some time later. I was not very pleased with myself as I was just in the middle of getting the CorComp card going again.

CorComp disk controller card and PAL chips

The CorComp disk controller card is very like the AT and MiniPE disk controllers. It shares the same main ICs but differs in having PAL (Programmable Array Logic) chips to do most of the combinational logic required for decoding and so on. Both of these cards have a static RAM chip for saving information used by the disk controller chip. This chip is a 2114 (or equivalent 1K by 4 bits) and sometimes can cause problems which are hard to diagnose. In this case, the program would start to load in Extended BASIC and reach a point where it changes the colour of the screen, begins to read some more files and then come up with an error. Changing the 2114 fixed this problem. Then I damaged the card, as mentioned previously and found that I had destroyed one of the PAL chips, the one which has MEMEN(L) as one of its inputs. Using a working PAL chip from Rolf's CorComp card, I made an adaptor to convert the 20 pin PAL chip to a 24 pin EPROM chip and read the contents of the PAL chip as if it were an EPROM (2732 with 12 inputs and 8 outputs versus 12 inputs and 6 outputs of the PAL chip). This is fine as long as there is only combinational logic in the PAL chip which was the case here. I then programmed a 2732 with this pattern and was able to plug in 9 pins of the EPROM into the socket along with a bit of wiring for all the other pins. This worked but did not look all that good. I examined the contents of the PAL readout and worked out the logic equations and programmed another PAL chip. This chip did the job so that all is restored to working. I shall be more careful in future.

I have also had two AT cards to look at. One was a fully expanded version and the other was just a disk controller to which a PIO had been added. Both of these had similar problems, as it turned out. The first one had the RS232/PIO enabled all the time as shown by the LED being on all the time. This proved to be due to a short between tracks because of solder spread around a connection. The second one was more difficult to find as it was a short between tracks due to a failure in manufacture. It was only a problem when the PIO was added. It also showed up a problem with the instructions given previously for adding PIO to an AT card.

Adding PIO to AT disk controller

In the April 1990 edition of TND there are instructions for expanding your AT disk controller card by adding 32K memory, PIO and RS232. If you are going to add them all at once, things are relatively simple as all ICs and other components are added and all links are removed. If you do not add all of them, then some links will need to be left and even added as well as only some of the ICs installed. This can be confusing, even if you know what you are doing. As a help, I include a table of the chips on the AT card and what they are used for. Consult the drawing on page 16 of the April 1990 TND for the position of each chip on the board.

Chip		Disk	Memory	PIO	RS232/1	RS232/2
U2	62256		x			
U3	LS138	x	x	x	x	x
U4	LS02	x	x	x	x	x
U5	9902					x
U6	9902				x	
U7	LS86		x	x	x	x
U8	LS138			x	x	x
U9	LS32			x	x	x
U10	LS32	x		x		
U11	LS27	x				
U12	LS20	x	x			
U13	LS30	x				
U14	LS10	x		x	x	x
U15	LS245	x	x	x	x	x
U16	2793	x				
U17	9901	x				

Chip	Disk	Memory	PIO	RS232/1	RS232/2
U18 27256	x		x	x	x
U19 LS04	x				
U21 LS125	x		x	x	x
U22 LS00		x	x	x	x
U23 LS74			x		
U24 45406				x	x
U25 LS74				x	x
U26 2114	x				
U27 LS74	x				
U28 LS122	x				
U29 7438	x				
U30 7406	x				
U31 LS244	x	x	x	x	x
U32 LS244	x	x	x	x	x
U33 7805	x	x	x	x	x

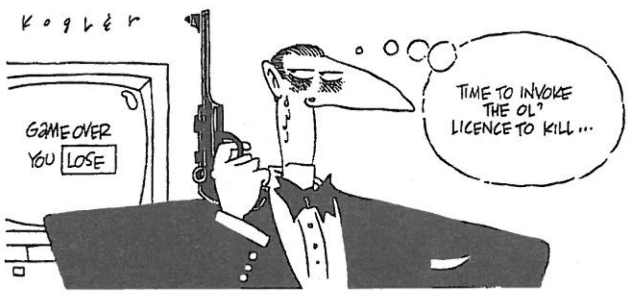
The jumpers required become very complicated as you mix and match the various options. If you wish to do something special, contact me and I will help you to set them up correctly for your choice of options.

RAMdisks

I have had a few RAMdisks to look at. One had a bad connection while another had a few bad connections as well as an 8K RAM chip which showed a few errors. These problems are sorted out by using the quite comprehensive memory test for the RAMdisks. One of the RAMdisks showed another problem, that of fast discharge of the battery. This was caused by the 74LS154 chip which is a 4 line to 16 line decoder. This provides the chip enable signal for each of the 32K RAM chips and when the power is removed these chip enable signals should be allowed to go to the battery voltage to put the RAM chips in their low current mode. To ensure this, resistors are connected from these lines to the battery voltage. What is happening on this RAMdisk is that the 74LS154 is drawing current through its outputs when the power is removed from the 74LS154. This discharges the battery plus the RAM chips are also not put into their low power mode. I have found a similar problem with another 74LS154 but only one output was drawing current in that case. I wonder if it is the manufacturer of the chip which determines whether it will cause this problem or not. The chips giving the problem are made by Hitachi while the ones which do not cause problems are made by National.

QED cartridge

The QED cartridge was designed and built in the Hunter Valley and is like a SuperCart with an Editor Assembler GROM and 32K bytes of RAM, battery backed and bank switchable in 8K lots. It is an interesting device for setting up a variety of programs which can be selected by menu. When I looked at the device, I was surprised to find that the voltage across the RAM chip was only about 1.5 volts although there was a 2.4 volt battery. What I found was that when the cartridge was plugged in to the console, a ground connection was made through that connector from one side of the board to the other. When the board was disconnected, the return path from the RAM chip to the battery was through some other ICs which caused the loss of voltage. I put a 1k resistor in to give a better path (which would be shorted out when inserted into the console) and the full battery voltage then appeared across the RAM chip. An unfortunate design fault I guess. o



TIsHUG Software Column

by Rolf Schreiber

I hope that those of you who look forward to the new software releases each month were not too disappointed by the absence of my column for the last two months. It was the result of a lack of new software from overseas and a lack of time on my part that caused me to omit my regular contribution. I still do not have any news of new software at this time, commercial or otherwise, so I dug deeply into the software archives and selected some disks that I hope you will find both interesting and stimulating. All disks are in SSSD format, unless otherwise indicated.

DISK A240 is STAR V1.1 (Super TI Assembly Routines), by Michael Riccio. This disk of assembly language routines was released in 1986 as FREEWARE and does very similar things to Craig Sheehan's XDP utilities. The routines allow the Extended BASIC programmer to gain access to the many features of our computer, which would normally require experience in assembly language programming.

DISK A249A and DISK A249B are a set of two disks of animation demonstrations and utilities. The software originated in Germany and has been translated into English. It was originally released as a "flippy" and is now available as FREEWARE. The first disk has been modified by Ray Kazmer and contains Comic Show Editor. This program allows you to create animated comic shows from TI-Artist pictures in a two step process. The pictures are first compressed into a squeezed format and then compiled into a stand alone comic show. Also on this disk is an E/A module loader and examples of animated art, including Garfield and Odie, Pluto and Ghost. The disk is menu driven, requires Extended BASIC, and includes full documentation. The second disk is also menu driven and contains three examples of animation by Thomas Opheys, the same person who wrote the TI99/4A emulator for an IBM PC.

DISK A261 is a disk of assorted assembly language games. The disk is menu driven and includes the following games: Maths Catch, an educational game to help lower primary school students; Mission -X-, a fast paced arcade game involving avoiding obstacles and shooting at moving targets; St-Nick, a maze game with a Christmas theme; Star Trap, a space game requiring good reflexes; and TI-Mazogz, a Pacman clone.

DISK A445 is the eighth disk in the TI-Base tutorial series from Martin Smoley. It includes all the sample data bases and setup files to work through the tutorial by yourself on your computer. o

Software Review

by Lou Amadio

The software review for this month is Rock Runner by Eric LaFortune, but, before getting into this marvelous game, a little diversion. I was fortunate enough to get hold of a copy of the Orphan Chronicles, a book on the early history of the TI99/4A, by Ronald Albright Jr. To say that the book was interesting would be an under statement. In my opinion, it is the most factual and entertaining account of the turbulent history of our computer that I have read to date. Try and get hold of a copy and read it as soon as possible.

Back to the review for this month - Rock Runner comes on a single floppy disk and requires the Editor Assembler module to load and run. Be warned that if you try to load it via your RAMdisk menu, you will probably lose your ROS, as I did. Had I read the documentation prior to starting this program, I could have saved myself a lot of time. Apparently the program needs some of the utilities that exist in the Editor Assembler module, so there are no short cuts to starting this game.

continued on page 8

TI-Bits Number 7

by Jim Swedlow, CA USA

[This article originally appeared in the User Group of Orange County, California ROM]

IF THEN IN XB

A number of the XB columns discussed alternatives to IF THEN. Here is another. Suppose that A$ depends on the value of I. You might use this code:

```
IF I=1 THEN A$="FRED" ELSE A$="PAUL"
```

A simpler way is to use the SEG$ function:

```
A$=SEG$("PAULFRED",1-4*(I=1),4)
```

Will this work if the two variables have different lengths? Yes! Remember that SEG$ does not produce an error if the length of the new string (the last number) is longer than the source string. If our two names are "PAUL" and "SAM", this works:

```
A$=SEG$("PAULSAM",1-4*(I=1),4)
```

QUOTES OF THE MONTH

"I shall never believe that God plays dice with the world."

---Albert Einstein (1879-1955)

"Genius is one percent inspiration and ninty-nine percent perspiration."

---Thomas A. Edison (1847-1931)
Attributed

A SAD TALE

A TI owner in New Jersey wrote that a thunder storm had destroyed his 4A and expansion system. I asked him if he used a surge suppressor. He responded:

"I am and always have been a strong advocate of the surge protector.

"All my equipment was plugged into a surge suppressor containing an on/off switch. The switch was in the off position. The surge protector was plugged into a wall outlet which is switch controlled. This switch was also in the off position.

"In my opinion, the air was highly charged with static electricity. Via some strange method of conduction, the charge dissipated to the chips. I have never heard of such occurrences. While discussing it with a few neighbours, I learned of VCR's and TV's being damaged during that same storm, none of which were in use at the time.

"How do you protect against this? My friends in the electronic field tell me that you cannot. We discussed possibilities such as a "grounded" station but, since most computers have a plastic case, this is not practical (unless you want to open the case and attach a ground lead to the mother board or another central ground).

"When considering the odds against this, it is not worth the effort. Had I known that Murphy's Law would select me, I would have tried it. Too late now."

ERROR TRAPPING AFTER RUN

The following will NOT work:

```
100 ON ERROR 200
110 RUN "DSK1.TEST"

. . . program continues

200 PRINT "TEST PROGRAM NOT FOUND"
210 PRINT "INSERT DISK IN DRIVE ONE"
220 PRINT "AND PRESS ANY KEY"
230 CALL KEY(0,K,S) :: IF S<1 THEN 230
240 RETURN 100
```

If you run this program with out a program called TEST in drive one, lines 200 thru 220 will print their message and then your program will fail in line 230 with 'SYNTAX ERROR IN 230'.

Why? As near as I can tell, when your TI executes the RUN part of RUN "DSK1.TEST", it clears memory including the variable table (think of it as 'un-pre-scanning' the program). The program and the line number table, however, remain.

When your TI tries to execute line 230, it looks for the variable table to find out where the values for S and K are stored. Finding none, the CPU decides that an error condition exists and ends the program.

What to do? One way is to add a disk directory reading routine to find out if the desired program is on the disk. This will significantly increase the time it takes to load the program, however.

Another way is to use RUN. This will recreate the variable table. Since you can specify the line number where program execution starts, you can control what happens. This works:

```
200 RUN 210
210 PRINT "TEST PROGRAM NOT FOUND"
220 PRINT "INSERT DISK IN DRIVE ONE"
230 PRINT "AND PRESS ANY KEY"
240 CALL KEY(0,K,S) :: IF S<1 THEN 240
250 GOTO 100
```

What about pre-scan? Even though you specified that program execution started at line 210, the entire program is pre-scanned. You must be careful, however, to return to a point in the program that will assure that all necessary variable initialisation is completed. o

Back-Up Copy?

by Earl Raguse, UGOC, CA USA

I found the following interesting story in North Jersey UG newsletter Sept 89 by Frank Decandia. I thought I would share it with you. The story is not only humorous, but Frank says it has a moral. I think it does too.

Lets now journey to the mythical land of I.P.M. (It is a Peculiar Machine). Jr. programmer Ted Nibble has just finished his first word processing program which he hopes to sell for $499.95. He corrects all the mistakes, checks the program for bugs, and saves the program to disk using a copy protect scheme.

Ted is not all there so he does not make a master (back up copy) of his program. In eager anticipation, Ted puts his disk in the disk drive and closes the door. He types the command to run his program and hits the "return" key with flairing confidence. As the disk spins it makes the sound of cash registers ringing all over computer land in his mind. He is interrupted by a message flashing on the screen. His program is running perfectly! He removes the disk putting it aside.

Ted sits in amazement as he explores all the wonderful features of his program. Ted finally decides he is going to be a famous author and he is going to use his word processing program to do it! The title of his best seller? "You Too Can Be A Great Programmer", By TED NIBBLE. Ted writes a while when he realizes that he will be late for an important meeting. He decides to format a disk and save his work. Ted knows formatting a disk will erase any information on it, what he does not know is he just stuck the only copy of his word processing program disk into the disk drive.

continued on page 23

XB tips Number 8

by Jim Swedlow, CA USA

[This article originally appeared in the User Group of Orange County, California ROM]

ASC CODES

At the February and March Users Group meetings a list of ASCII codes in both base 10 and hex was passed out. There were some questions about the BASIC commands listed on the chart.

When your computer saves a command on disk or cassette it changes the command statements to tokenized statements. CALL, for example, is saved as HEX 9D or CHR$(157). The chart I supplied lists all of these tokenized ASC codes.

WD 40

Recently Ramon mentioned using WD 40 to clean a disk drive. WD 40 is an excellent solvent but a lousy lubricant as it attracts dirt. Further, it does bad things to plastic. So use it to clean up gunk on metal but be sure and use a good quality oil to lubricate.

SUB PROGRAMS

XB has four (or more) ways of writing utility routines: sub routines, sub programs, assembly language routines and DEFinitions. Each of these has pros and cons. This month we will look at sub programs.

The best way to describe a sub program is to explain where it differs from a subroutine:

-- Sub routines are called by line number while sub programs are CALLed by name.

-- Sub routines use the same variables as the main program. In sub routines, variables are entirely different unless noted in the CALL statement.

-- Sub routines can be anywhere in the program while sub routine must be at the end.

Since you call sub programs by name, you do not lose their identity when you RESequence. You lose the power of ON GOSUB and passing variables can be tricky if quite a few are involved.

Enter and run this sample program:

```
10 A,B=10 :: PRINT A;B;C ::CALL TEST
20 CALL NEWS(B) :: PRINT A;B;C
30 SUB TEST :: A=5 :: PRINT A;B;C :: SUBEND
40 SUB NEWS(C) :: PRINT A;B;C :: C=4 :: SUBEND
```

You should get this output:

```
10 10  0
 5  0  0
 0  0 10
10  4  0
```

If you think this through, you will see that there are NINE variables in this program: A, B and C in the program; A, B and C in TEST; and, A, B and C in NEWS.

When you called NEWS(B) the value of B was passed to NEWS and processed as C in that sub program. Play with this program (including inserting BREAK statements) to test this out.

ORACLE

This program is loosely based on one that appeared in '99er years ago. Both programs were designed to demonstrate sub programs. Speech has been added to this version.

The CALL PEEK in line 160 works on my computer but may not work on yours -- if you have a problem, just delete it. If you do not have speech, change the CALL SAY's to PRINT's.

Various formats for sub programs are shown. The rules are in the XB book, especially for passing variables between the main program and the sub program. Note that SUB name must start a line and SUBEND must end a line.

A disclaimer: all answers are based on random numbers. If you think the answers fit, well, that is just random chance, right? A computer cannot give good answers, can it?

Enjoy!

```
100 ! ORACLE
110 ! VERSION XB.2.1
120 ! 08 MAR 85
130 ! BY JIM SWEDLOW
140 !
150 DISPLAY AT(10,4)ERASE ALL BEEP:"** I AM THE ORACLE
    **" :: CALL DELAY :: RANDOMIZE
160 CALL INIT :: CALL PEEK(-28672,I):: IF I=0 THEN DISPL
    AY AT(20,1):"I cannot operate without the Speech
    Synthesizer!" :: STOP
170 DISPLAY AT(15,1):" I answer all questions": : :"Ask
    questions with YES or NO answers -- When you are
    done press ENTER."
180 CALL DELAY :: DISPLAY AT(24,1):"PRESS ANY KEY TO
    BEGIN"
190 CALL KEY(0,I,S):: IF S=0 THEN 190 ELSE CALL CLEAR
200 PRINT : :"WHAT IS YOUR QUESTION?" :: LINPUT Q$ :: IF
    Q$="" THEN 220
210 CALL DELAY :: CALL REPLY(Q$):: CALL DELAY :: GOTO
    200
220 DISPLAY AT(10,1)ERASE ALL:"THANK YOU FOR CONSULTING"
    : : : : : : :" ** THE ORACLE **" :: CALL DELAY ::
    STOP
230 !
240 SUB DELAY :: FOR I=1 TO 200 :: NEXT I :: SUBEND
250 !
260 SUB REPLY(A$)
270 A$=SEG$(A$,1,2):: IF A$="WH" OR A$="HO" THEN CALL OT
    HER ELSE CALL YESNO
280 SUBEND
290 !
300 SUB YESNO
310 ON INT(10*RND)+1 GOTO 320,330,320,340,350,350,360,37
    0,380,390
320 CALL SAY("YES"):: SUBEXIT
330 CALL SAY("I THINK SO"):: SUBEXIT
340 CALL SAY("LOOKS POSITIVE"):: SUBEXIT
350 CALL OTHER :: SUBEXIT
360 CALL SAY("LOOKS NEGATIVE"):: SUBEXIT
370 CALL SAY("I DO NOT THINK SO"):: SUBEXIT
380 CALL SAY("NO WAY"):: SUBEXIT
390 CALL SAY("NO"):: SUBEND
400 !
410 SUB OTHER
420 ON INT(10*RND)+1 GOTO 430,440,450,460,470,480,490,50
    0,510,520
430 CALL SAY("I CAN NOT TELL YOU THAT"):: SUBEXIT
440 CALL SAY("SAY THAT A DIFFERENT WAY"):: SUBEXIT
450 CALL SAY("YOU DO NOT WANT TO KNOW"):: SUBEXIT
460 CALL SAY("I DO NOT KNOW"):: SUBEXIT
470 CALL SAY("I AM NOT SUPPOSED TO SAY"):: SUBEXIT
480 CALL SAY("I WILL NOT TELL YOU"):: SUBEXIT
490 CALL SAY("I CAN ONLY GUESS"):: SUBEXIT
500 CALL SAY("I CAN NOT ANSWER THAT"):: SUBEXIT
510 CALL SAY("I DO NOT REMEMBER"):: SUBEXIT
520 CALL SAY("TRY SOME THING ELSE"):: SUBEND
```

o

Wanted

I wish to buy a colour monitor for a friend who will join the club. If you have one you are not using, give me a ring on 9188132 (home) or leave a message for me on 8093181 during the day. Naturally I would buy the interface too if it is available. o

ASCII to Program Files

reprinted from CALL SOUNDS, Westchester PA, USA

Most users who have used an expanded TI system for some time have realised that you can convert a working Basic or Extended Basic program into a DV/80 file of that same program for editing and publishing purposes. This is done by using the 'List' command and is explained on page II-22 of the green and gold Users Reference Guide. This method ensures that once you have a working program you can publish it free from error because it is the computer that converts the program into a file.

What most users do not know is that you can change a program in text form back again to a working program. That is the subject of this article and the program that follows it converts from ASCII text to a program file. I trust that this is of some use to somebody. The information was taken from articles in the TI-Writer Supplement published by the Chicage Users Group.

Whether you use TI-WRITER to write and/or edit a program or you have a program that was downloaded by modem using CTRL 2, both types of files must be converted from DISPLAY VARIABLE 80 to usable programs. conversely, a program can be LISTed to disk resulting in a Display Variable 80 file that can be transmitted in ASCII text by modem and the recipient can use this program to convert it back to a program. The program will also delete any carriage returns. Extended Basic will interpret the output file as a program.

List the Display Variable 80 file on the screen and check to make sure none of the numbered program lines are more than 80 characters. Split the longer lines into two or more lines and mark the succeeding segments with a line number that will indicate it is part of the previous line. If lines contain more that 80 characters the conversion program will stop with BAD ARGUMENT IN 160. (I found that it merely chopped them off and you had to type the remainder on later. ED).

Other than having 80 characters or less on a line when editing the program with TI-WRITER or EDITOR ASSEMBLER be certain ALL blank lines are deleted. If a carriage return has been placed on the last numbered line of the program, a blank line will be automatically inserted when the cursor is moved down and to the left margin. Although the conversion program will automatically remove all carriage returns, when the conversion program reaches this, or any, blank line, it will look for a non-existing line number and the program will stop with BAD ARGUMENT IN LINE 160. Save the file to disk with the PRINT FILE (PF) command to eliminate saving Tab settings at the end of the file.

Run the conversion program and input the last file saved (no blank lines or lines longer than 80 characters) as the file to be converted. Input the name of the new merge file to be created. Let us call the merged file TEXT. After pressing ENTER the Display Variable 80 file will be read and a new file will be written to disk in MERGE format. When the drive has stopped completely, before doing anything else type NEW to clear the memory, then type MERGE DSK1.TEXT (or the name of the merge file you are using). Now list the program to the screen to edit any split lines into one complete line and delete the EXTRA BLANK SPACE after the line number. Save the resulting program using a final filename. There is your converted program (You also have to eliminate the exclamation sign ! at the beginning of each line. ED).

```
100 !************************
110 !   TRANSLATES FROM
120 ! DIS/VAR 80 TO MERGE
130 !       FORMAT
140 !************************
150 !
160 !USE A FULL SCREEN
170 !EDITOR TO CREATE
180 !EXTENDED BASIC PROGRAM
190 !
200 !CREATE A FILE USING
210 !TI-WRITER - MAKE
220 !SURE YOU DISABLE THE
230 !WORD WRAP MODE AND
240 !LIMIT THE LENGTH
250 !TO 80 CHARACTERS
260 !
270 CALL CLEAR
280 DISPLAY AT(3,7)BEEP ERASE ALL:"***TRANSLATE***"
290 DISPLAY AT(7,5):"DIS/VAR 80 FILENAME:": :"DSK1."
300 ACCEPT AT(9,4)SIZE(-12):IN$
310 DISPLAY AT(12,15)BEEP:"MERGED OUTPUT FILENAME:":
    "DSK1."
320 ACCEPT AT(14,4)SIZE(-12):OUT$
330 OPEN #1:"DSK"&IN$&""
340 OPEN #2:"DSK"&OUT$&"",VARIABLE 163
350 LINPUT #1:L$
360 S=POS(L$," ",1)
361 A$=SEG$(L$,S+1,80)
370 ON ERROR 490
380 N=VAL(SEG$(L$,1,S))
390 ON ERROR 440
400 !
410 PRINT L$
420 PRINT #2:CHR$(INT(N/256))&CHR$(N-256*INT(N/256))
    &CHR$(131)&A$&CHR$(0)
430 GOTO 350
440 PRINT #2:CHR$(255);CHR$(255)
450 CLOSE #2
460 PRINT : :"ENTER""NEW"" AND THEN ""MERGE"" THE
    TRANSLATED FILENAME:":"";OUT$: : :
470 PRINT "REMEMBER TO REMOVE THE      LEADING ""!""
    IN EVERY LINE." :: ::
480 END
490 ON ERROR 440
500 RETURN 350
```

continued from page 21

```
SCR #7
 0  (DISK DIRECTORY EGR 12 29 87) HOME
 1  ."        PAGE 1 OF MUSICAL MENU" CR CR CR
 2  ."          0. ABORT TO FORTH   " CR
 3  ."          1. Mary's Little Lamb" CR
 4  ."          2. Song At Twilight " CR
 5  ."          3. Little Boy Blue  " CR
 6  ."          4. Little Bo Peep   " CR
 7  ."          5. Baa Baa Black Sheep " CR
 8  ."          6. What Can The Matter Be " CR
 9  ."          7. Hickory Dickory Dock " CR
10  ."          8. Bass Note Demo   " CR
11  ."          9. PLAYNOTE DOCS    " CR CR CR
12  ."   ENTER <DIR> TO RETURN - DON'T FORGET"
13  12 17 AT ." ENTER SELECTION "
14  13 23 AT FREE  28 17 AT
15
```

Screen #67 does some new things, DT9 combines all the previous lyrics into one word and adds a second verse. The word ODC (Oh Dear Collected) finally puts it all together, so that when Screen #67 is LOADed, the immediate stream of words DTO PAK DD 2 WAIT ODC 2 WAIT ASK 2 WAIT CLS DIR IS executed in order. DTO (Screen #62) puts a title on the CRT, PAK waits for your key press to execute DD, a 2 second WAIT, then ODC tells of the second version, via ODO PAK CLS, DT puts up the text, OD plays it, CLS clears the CRT, DT9 puts up the second verse, and OD plays the music again. After a 2 second WAIT ASK is executed to inquire via .ASK, a UFW from Screen #89 of Beginning Forth #4, if you wish to repeat by pressing Space, which is checked for by KEY 32=, if this leaves a true flag on the stack, IF will execute DT OD again, ELSE the word DIR is executed to take you back to the DIRectory. If you have not installed DIR, just do not enter it, and the execution will stop there, also omit the words after ASK on Screen #67.

That is all I am going to say about music. There is more, but it gets technical and we have lots of other stuff to cover. Therefore, until next time,

May the FORTH be with you!

To Charge or not to Charge

by D.R.Y. Cell

It is very upsetting to think that after the better part of a century that you humans do not use us Dry Cells to our fullest measure, but then this is something that is never taught in your 'throw away society' and the misleading information which is often printed on us by the manufacturers, who would like to perpetuate the habit of throwing us away when we could be better utilised. As a human, how would you like to be discarded when you had still a lot to give? Pick up a packet of cells and somewhere on it you will probably find these cautions:-

DO NOT RECHARGE BATTERIES ... What they do not say is, if you do you will buy less batteries and that is not good for our business.

DO NOT USE NEW BATTERIES WITH OLD ONES ... This is a fair statement as the old cells will bring down the performance of the new ones.

DO NOT DISPOSE OF BATTERIES IN A FIRE ... This is also wise since heat would cause a well sealed battery to burst, however most cells are not all that well sealed and generally have parts of the outer casing which has been eaten away by the electrolyte and a small rupture can be caused, which in most cases would do no damage to a plastic container in which it is wise to house any charger.

DO NOT MIX MANGANESE AND ALKALINE CELLS ... Well our Alkaline cousins have a higher capacity and discharge rate capability and it would be wasteful since us Manganese types simply do not have the same capacity and would run down first and then help discharge our cousins faster. We would like to be with our own types anyhow.

It has been reported that in Japan it is not permitted for battery manufacturers to put the DO NOT RECHARGE warning on us since it can be done and is done. However that does not stop these same firms from writing this warning on cells they make for export. In this country politicians are mainly interested in getting re-elected and have little interest is such small deceptions when they often practice must greater ones while they hold office or try to get in.

This TISHUG Member (TTM) once bought a tape recorder many years ago for his son. The cells he bought for it had 'LEAKPROOF' written on the case. THEY CERTAINLY LEAKED and TTM called the manufacturer Union Carbide who promptly sent a PR man down. After seeing the damage he said 'Of course there is no such thing as a leakproof cell but if we did not put the Leakproof sign on we would not sell any since our competitors all put it on their cells.' Come to think of it we have noted that the Leakproof label has not been tagged on us for some time now. There were probably many more complaints.

TTM saw an article in a Popular Science magazine many years ago which gave a simple transistor circuit to allow humans to recharge us about five or six times before we are finally discarded. The secret is not to let our single cell voltage fall below 1.0 Volt ie. a 9 Volt Battery which has six of us in series, should not be allowed to fall below 6 Volts on load. This means that we have to be removed from circuit and checked and never allow us to have our potential fall below 1.0 Volt. With a heavy current drawing dump truck toy you would have to check the potential quite shortly in spite of Junior's protests, whereas with a TV remote control it would not be necessary more than once in every six months. If you look at the curves for typical cells of different types, you will see that when we reach 1 Volt on load, we do not have far to go. Much like you humans when you reach 100 years of age. This means that we have to be removed from circuit and checked more often so that our 'On Load' voltage is never allowed to fall below 1.0 Volt. With a heavy current drawing toy dump truck our potential will have to be checked quite shortly, in spite of Junior's protests whereas with a TV remote control it would not be necessary more than once every six months. If you look at curves for typical cells for the different types, you will see that when our potential falls to 1.0 Volt we are down to 10% of our energy level. Incidentally our mercury cousins should not be recharged because they can explode - just like you humans that have a mercurial temperament. If you have a charger you use to charge our Nickel Cadmium cousins and if the current is adjustable you can use it to charge us as well as long as you remember that our bigger brothers will stand a heavier current as shown in the table below. A very neat little charger can be found at Paddy's Market for about $17 (maybe $15 with a little bargaining) which allows four cells to be charged individually at one time, with a LED in series with each and with a built in battery checker. In the meantime, if you really wish to save money on us, keep your eye on our prices and you will find that they vary quite a lot from store to store. Make a point to avoid those of us that have been advertised on TV as the prices are loaded with the costs of the advertisements while they are really no better than some of us that are never brought into the limelight. We will try and encourage one of our Nicad cousins to write a further article on tricks and tips for using and caring for them.

Percent of energy remaining in carbon-zincs and alkalines for various ON LOAD voltages.

Volts	
1.5	= 100%
1.4	= 85%
1.3	= 65%
1.2	= 35%
1.1	= 20%
1.0	= 10%
0.9	= 0%

Recommended Charging currents for our brethern of various sizes.

	Current mA
AAA	20
AA	40
C	60
D	100
9 Volt	10
Lantern	140

o

continued from page 4

Apart from this minor inconvenience, Rock Runner is probably the best game to be released for the TI99/4A in the last 12 months. The graphics are excellent, game speed is good and there is the challenge of a game strategy to keep up the interest.

According to the documentation (3 easy to read pages) the game is a "technical feat that pushes the 4A to the limits". The program utilizes a "half bitmap" mode which mixes the capabilities of bitmap with the speed of pattern mode. Include fast action and sound and you have the recipe for an excellent game.

Rock Runner requires a joystick and begins with an instruction screen asking for the starting level (15 in all). Each level allows a limited time to solve the puzzle.

The object of the game is to move the character around the screen while gathering diamonds. Sounds simple enough, except that you have to avoid a number of dangers. At the lower levels, these consist mainly of falling rocks, but at the higher levels, watch out for nasty creatures that travel through the tunnels (that you have dug in search for diamonds) in search of prey. Fortunately, you have some weapons at your disposal, but there are other traps waiting for you as well.

Thankfully, the game has a pause mode, because you need a break now and then from the arduous task of surviving while digging for diamonds.

Like all good games, Rock Runner can be enhanced by utilizing certain strategies, but it will be up to you to discover and use them to best advantage.

This game is highly recommended. Check with the club shop for supplies or order a copy as soon as possible. (Rock Runner is distributed by Asgard Software). o

Lurking Horror part 2

Copyright 1987 Infocom
This walk through is by Scorpia, Copyright 1987.

You may have noticed a stairway going up near the outlet. Now is as good a time as any to see where it leads. Hmmm, it does not lead very far. And the ladder that goes up to the catwalk above seems to be missing. But there IS some sort of shiny rope hanging down.

Guess there is nothing for it but to climb the rope (and a good thing you have those gloves on right now). Huff! Puff! Not as easy as it looked. Ummmmm. Hmmmmm. Gee...that was not a rope at all....it was a tentacle! A tentacle attached to a really atrocious-looking creature! Fortunately, it decides to leave in another direction. Whew!

With the thing gone (and your heartbeat returning to something approximately normal), open the door and step north out into the freezing storm. Brrrr! And you with no coat, either! Ah well, if you move fast enough, you may be able to avoid frostbite.

Climb up to the very top of the dome, where you see a bronze plug. Get the plug and drop it. In the recess is a paper, which is what you came for. Take that, and hurry back to the catwalk. Now, drop everything you are carrying (the ladder is very heavy), get the ladder, and lower it. Pick up all your goodies, and climb down the ladder, then down the stairs to the corridor again. You are about to have fun times!

Head east along the corridor to the end, then go north to Fruit and Nuts, where you will find a stairway. Tiptoe down the stairs to the Cluttered Passage, then southeast to the Brown Building basement. Pick up the boots and wear them. Now move up the stairs into the Brown lobby, and all the way up the stairs to the top of the building (too bad the elevator here is not working).

Unlock the door with the master key, and once again step out (west) into the frigid night. Climb up to the weather dome. Ahh, nice and warm in here. Hmm, wonder what a potted peach tree is doing in a place like this?

Errrr....did you hear something out there? I think you did. Something nasty with claws, in fact. Something nasty with claws that is on its way in to do nasty things to you. Get out quickly. Once outside, and with a clear (if unsettling) view of the monster, throw the stone at it. SCREECH! Both monster and stone go over the side of the roof (note: if the monster shows up inside the dome first, throw the stone at it there, then get the stone and go outside).

Do not worry about that now. Return to the weather dome, and search the earth in the pot. UGH! You found a severed human hand! Trying not to feel squeamish, you take the hand (I will, with difficulty, refrain from making any "hand" jokes!).

Return to the Brown Lobby, then go outside to the courtyard. Fortunately, the monster is gone, but the stone is there. Get that, and go back inside. Warm up a bit, because you are about to pay a visit to the Alchemy Department.

Make your way back through the basement to the Infinite Corridor, and this time go south, through the Chem Building, until you reach the Alchemy Department door. Looks like someone is up pretty late in there.

The door is locked, and for some strange reason, your master key will not open the lock. No matter, I am sure if you just knock on the door, whoever is inside will be glad to open it for you.

Sure enough, there is a professor here, and he very kindly allows you into his office, even if he does not seem to be especially interested in you at the moment. As you look around, one thing catches your eye: a lab signup sheet taped to the wall. Is not one of the names on it familiar?

Come to think of it, the heaviest user of the lab is the same person whose note you found on top of the Great Dome. Could there be a connection here? Let's find out: show the note to the professor.

Ha! That sure changed his attitude, did it not? Maybe not for the better, though ! Now try going south to the lab. See how easy that was? Only, the professor seems to have some plans for you. Plans that have nothing to do with increasing your life span.

Before you know it, the doorway is magically barred, and you find yourself in a chalked pentagram, equally surrounded by magical forces that prevent you from leaving. Uh-oh. This does not look good.

Do not panic! You have the means to free yourself. While you wait for the proper moment, take a look around the lab. Note especially the vat of tarry liquid on the bench, as well as what the bench is standing on: a trap door.

Okay, the professor is now in his pentagram, and is starting a chant that will summon something you do not even want to think about for a second, let alone meet. This is the moment: cut the chalk line with the altar knife. Now, without wasting any moves (time is short here) move the bench, open the trap door, and go down!

Gasp! You just made it! Turn on the flashlight, and you will see you are in a cinderblock tunnel. This is the south end of the tunnel from the manhole in Ancient Storage. Once you get your breath back, climb up again to the lab (it is safe now, honest!).

Hmmmm...what a mess! There does not seem to be much left of the professor, except a red smear on the wall, and a brass hyrax (ring) on the floor. Fortunately, the vat of liquid escaped destruction. Have you been wondering what it is?

Well, one of the things Alchemists tried to make in the old days was the Elixir of Life. Say, you do not suppose....? Why not try it? Put the hand in the vat and let's see what happens.

This is creepy! The hand is moving... it is coming back to life! (Hey, while you are standing there, do not forget to pick up the ring!). It is trying to climb out of the vat! Now, I know that picking up a live severed hand is even more horrible than picking up a dead one, but...well, you will just have to do that.

No sooner do you pluck it out than it climbs up to rest on your shoulder. Gulp! At least it seems, err, friendly. Put the ring on the hand. Fits very nicely there, does it not? Now you are about ready for a date with the rats in the Steam Tunnel.

o

continued from page 14

This demonstrates a general principle (if there are any) that in well written programs you can usually trade storage cost for time cost or vice versa. Obviously, you trade in favour of storage in little used parts of the program and in favour of time in the inner loops. Do not be too quick to trade though because as we have shown in this series of articles, you can often improve both costs with a little thought. Once you have expended the effort to think about a coding problem and have "learned" how to code it efficiently or, better still, have encapsulated it in a macro then all future programs benefit.

Next time we will look at a variation of this same problem which I will call "counting".

"Write a program that even a fool can use, and only a fool will want to." o

```
1 GO TO 10000
2 GO TO 5050
3 GO TO 5060
4 GO TO 5070
5 GO TO 5080
6 GO TO 5090
7 GO TO 5100
8 GO TO 5110
9 GO TO 5120
10 GO TO 5130
11 GO TO 5140
12 GO TO 5150
13 GO TO 5160
14 GO TO 5170
15 GO TO 5180
16 GO TO 5190
17 GO TO 5200
18 GO TO 5210
19 GO TO 5220
20 GO TO 5230
21 GO TO 5240
22 GO TO 5250
23 GO TO 5260
24 GO TO 5270
25 GO TO 5280
26 GO TO 5290
27 GO TO 5300
28 GO TO 5310
29 GO TO 5320
30 GO TO 5330
31 GO TO 5340
32 GO TO 5350
33 GO TO 5360
34 GO TO 5370
35 GO TO 5380
36 GO TO 5390
37 GO TO 5400
38 GO TO 5410
39 GO TO 5420
40 GO TO 5430
41 GO TO 5440
42 GO TO 5450
43 GO TO 5460
44 GO TO 5470
45 GO TO 5480
46 GO TO 5490
47 GO TO 5500
48 GO TO 5510
49 GO TO 5520
50 GO TO 5530
51 GO TO 5540
80 CALL CLEAR
85 CALL SOUND(1000,262,2)
86 CALL SOUND(1000,262,2,330
,2)
87 CALL SOUND(2000,262,2,330
,2,392,2)
90 CALL SCREEN(12)
91 FOR Z=1 TO 700
92 NEXT Z
93 CALL SCREEN(8)
95 CALL HCHAR(1,1,35,768)
98 CALL CLEAR
99 CALL SOUND(100,500,2)
100 PRINT "HELLO! I AM YOUR
GREAT      INVENTIONS GAME!
           "

110 PRINT "I WAS CREATED BY.
.....


           "
114 PRINT "       ##########
##########       "
116 PRINT "       PARALLEL S
YSTEMS,INC.    "
117 PRINT "       ##########
##########


   1983     "
118 Z$="3C42B9A1A1B9423C"
119 CALL CHAR(130,Z$)
120 CALL HCHAR(22,22,130)
121 FOR Z=1 TO 1500
122 NEXT Z
123 CALL CLEAR
124 CALL SCREEN(8)
125 CALL SOUND(100,500,2)
126 WRONG=0
127 PRINT "THERE ARE 2 SEPAR
ATE GAMES!! "
128 WRONG=0
130 PRINT "GAME #1.I GIVE 50
 INVENTIONSIN ORDER,YOU GUES
S INVENTORS "
135 PRINT "GAME #2.I GIVE 25
 INVENTIONSAT RANDOM,YOU GUE
SS THE     INVENTORS
              "

136 Z=0
180 PRINT "TO SELECT GAME, T
OUCH 1 OR  2 ON THE KEYBOARD
!        "
300 GO TO 310
310 CALL KEY(0,KEY,STATUS)
311 IF STATUS=0 THEN 310
312 IF KEY>50 THEN 310
313 IF KEY<49 THEN 310
314 IF KEY=49 THEN 1000 ELSE
 1995
1000 CALL CLEAR
1001 PRINT "I WILL GIVE YOU
AN INVENTIONYOU MUST GUESS T
HE INVENTOR!
              "

1010 PRINT "LET'S BEGIN
               "

1011 FOR P=1 TO 750
1012 NEXT P
1015 GO TO 5000
1995 FLAG=0
1996 CALL CLEAR
1997 PRINT "I WILL GIVE YOU
25          INVENTIONS AT RA
NDOM        YOU GUESS THE IN
VENTORS!      "
1998 FOR B=1 TO 1500
1999 NEXT B
2000 CALL CLEAR
2005 GO TO 2014
2010 Z=0
2014 FLAG=FLAG+1
2015 IF FLAG>25 THEN 2050
2016 Z=1
2018 RANDOMIZE
2019 CALL CLEAR
2020 A=INT(50*RND)+1
2021 IF A<=25 THEN 2030 ELSE
 2033
2030 ON A GO TO 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
2033 RANDOMIZE
2034 B=INT(25*RND)+1
2035 ON B GO TO 27,28,29,30,
31,32,33,34,35,36,37,38,39,4
0,41,42,43,44,45,46,47,48,49
,50,51
2050 CALL CLEAR
2051 GOSUB 11006
2055 PRINT "GREAT! YOU GOT A
LL 25 RIGHT.   "
2056 PRINT "ALONG THE WAY YO
U HAD.....     "
2060 PRINT WRONG;" WRONG GUE
SS(ES)"
2061 PRINT
2062 PRINT
2065 PRINT "TO PLAY AGAIN..T
YPE RUN      THEN PRESS ENTER
               "

2066 END
5000 GO TO 5010
5010 CALL CLEAR
5050 PRINT " ADDING MACHINE"
5051 GOSUB 9000
5052 PRINT " 1=WESTINGHOUSE
 2=OTIS      3=BURROUGHS"
5053 GOSUB 9050
5054 IF X=3 THEN 5057 ELSE 5
055
5055 GOSUB 9100
5056 GO TO 5050
5057 GOSUB 9200
5059 IF Z=1 THEN 2000
5060 PRINT " AIR BRAKES"
5061 GOSUB 9000
5062 PRINT " 1=OTIS  2=WESTI
NGHOUSE      3=LEE       "
5063 GOSUB 9050
5064 IF X=2 THEN 5067 ELSE 5
065
5065 GOSUB 9100
5066 GO TO 5060
5067 GOSUB 9200
5069 IF Z=1 THEN 2000
5070 PRINT " AIRPLANE"
5071 GOSUB 9000
5072 PRINT " 1=WRIGHT BROS.
 2=BOEING    3=LOCKHEED"
5073 GOSUB 9050
5074 IF X=1 THEN 5077 ELSE 5
075
5075 GOSUB 9100
5076 GO TO 5070
5077 GOSUB 9200
5080 PRINT " HOT AIR BALLOON
"

5081 GOSUB 9000
5082 PRINT " 1=SMITH BROS.
2=RITTY       3=MONTGOLFIER B
ROS."
5083 GOSUB 9050
5084 IF X=3 THEN 5087 ELSE 5
085
5085 GOSUB 9100
5086 GO TO 5080
5087 GOSUB 9200
5090 PRINT " BALL POINT PEN"
5091 GOSUB 9000
5092 PRINT " 1=DRINKER   2=L
OUD   3=LEE"
5093 GOSUB 9050
5094 IF X=2 THEN 5097 ELSE 5
095
5095 GOSUB 9100
5096 GO TO 5090
5097 GOSUB 9200
5100 PRINT " BARBED WIRE"
5101 GOSUB 9000
5102 PRINT " 1=GLIDDEN 2=WHI
TTLE 3=BAIRD"
5103 GOSUB 9050
5104 IF X=1 THEN 5107 ELSE 5
105
5105 GOSUB 9100
5106 GO TO 5100
5107 GOSUB 9200
5110 PRINT " BAROMETER"
5111 GOSUB 9000
5112 PRINT " 1=LAENNEC     2=
TORRICELLI    3=CRISTOFORI"
5113 GOSUB 9050
5114 IF X=2 THEN 5117 ELSE 5
115
5115 GOSUB 9100
5116 GO TO 5110
5117 GOSUB 9200
5120 PRINT " BICYCLE"
5121 GOSUB 9000
5122 PRINT " 1=SPRAGUE 2=SHO
LES          3=MacMILLAN"
5123 GOSUB 9050
5124 IF X=3 THEN 5127 ELSE 5
125
5125 GOSUB 9100
5126 GO TO 5120
5127 GOSUB 9200
5130 PRINT " CASH REGISTER"
```

```
5131 GOSUB 9000
5132 PRINT " 1=WHITNEY   2=B
ELL  3=RITTY  "
5133 GOSUB 9050
5134 IF X=3 THEN 5137 ELSE 5
135
5135 GOSUB 9100
5136 GO TO 5130
5137 GOSUB 9200
5140 PRINT " CATERPILLAR TRA
CTOR"
5141 GOSUB 9000
5142 PRINT " 1=MORSE   2=HOL
T  3=HENRY"
5143 GOSUB 9050
5144 IF X=2 THEN 5147 ELSE 5
145
5145 GOSUB 9100
5146 GO TO 5140
5147 GOSUB 9200
5150 PRINT " CLOCK(PENDULUM
TYPE)"
5151 GOSUB 9000
5152 PRINT " 1=EDISON    2=W
HITTLE       3=HUYGENS "
5153 GOSUB 9050
5154 IF X=3 THEN 5157 ELSE 5
155
5155 GOSUB 9100
5156 GO TO 5150
5157 GOSUB 9200
5160 PRINT " COTTON GIN"
5161 GOSUB 9000
5162 PRINT " 1=WHITNEY    2=
FRANKLIN     3=MARCONI"
5163 GOSUB 9050
5164 IF X=1 THEN 5167 ELSE 5
165
5165 GOSUB 9100
5166 GO TO 5160
5167 GOSUB 9200
5170 PRINT " DYNAMITE:  CLUE
=BE CAREFULL "
5171 GOSUB 9000
5172 PRINT " 1=COLT   2=NOBE
L  3=LAND"
5173 GOSUB 9050
5174 IF X=2 THEN 5177 ELSE 5
175
5175 GOSUB 9100
5176 GO TO 5170
5177 GOSUB 11006
5180 PRINT " ELECTROMAGNET"
5181 GOSUB 9000
5182 PRINT " 1=EDISON 2=MORS
E 3=STURGEON"
5183 GOSUB 9050
5184 IF X=3 THEN 5187 ELSE 5
185
5185 GOSUB 9100
5186 GO TO 5180
5187 GOSUB 9200
5190 PRINT " ELEVATOR"
5191 GOSUB 9000
5192 PRINT " 1=WESTINGHOUSE
 2=OTIS       3=BURROUGHS"
5193 GOSUB  9050
5194 IF X=2 THEN 5197 ELSE 5
195
5195 GOSUB 9100
5196 GO TO 5190
5197 GOSUB 9200
5200 PRINT " DIESEL ENGINE"
5201 GOSUB 9000
5202 PRINT " 1=FORD    2=DIE
SEL   3=GM"
5203 GOSUB 9050
5204 IF X=2 THEN 5207 ELSE 5
205
5205 GOSUB 9100
5206 GO TO 5200
5207 GOSUB 9200
5210 PRINT " FOUNTAIN PEN"
5211 GOSUB 9000
5212 PRINT " 1=WATERMAN  2=P
ARKER  3=BIC"
5213 GOSUB 9050
5214 IF X=1 THEN 5217 ELSE 5
215
5215 GOSUB 9100
5216 GO TO 5210
5217 GOSUB 9200
5220 PRINT " GATLING(MACHINE
)GUN"
5221 GOSUB 9000
5222 PRINT " 1=COLT 2=REMMIN
GTON          3=GATLING"
5223 GOSUB 9050
5224 IF X=3 THEN 5227 ELSE 5
225
5225 GOSUB 9100
5226 GO TO 5220
5227 GOSUB 9200
5230 PRINT " IRON LUNG RESPI
RATOR"
5231 GOSUB 9000
5232 PRINT " 1=JANNEY 2=HENR
Y 3=DRINKER"
5233 GOSUB 9050
5234 IF X=3 THEN 5237 ELSE 5
235
5235 GOSUB 9100
5236 GO TO 5230
5237 GOSUB 9200
5240 PRINT " JET PROPULSION
ENGINE"
5241 GOSUB 9000
5242 PRINT " 1=ROLLS ROYCE
 2=WHITTLE   3=PRATT&WHITNEY
"
5243 GOSUB 9050
5244 IF X=2 THEN 5247 ELSE 5
245
5245 GOSUB 9100
5246 GO TO 5240
5247 GOSUB 9200
5250 PRINT " KNITTING MACHIN
E"
5251 GOSUB 9000
5252 PRINT " 1=LEE     2=LOUD
   3=WATT"
5253 GOSUB 9050
5254 IF X=1 THEN 5257 ELSE 5
255
5255 GOSUB 9100
5256 GO TO 5250
5257 GOSUB 9200
5260 PRINT " LIGHT BULB"
5261 GOSUB 9000
5262 PRINT " 1=BELL  2=EDISO
N  3=MARCONI"
5263 GOSUB 9050
5264 IF X=2 THEN 5267 ELSE 5
265
5265 GOSUB 9100
5266 GO TO 5260
5267 GOSUB 9200
5270 PRINT " LIGHTNING ROD:
  CLUE=KITE   "
5271 GOSUB 9000
5272 PRINT " 1=FRANKLIN 2=ED
ISON 3=BELL"
5273 GOSUB 9050
5274 IF X=1 THEN 5277 ELSE 5
275
5275 GOSUB 9100
5276 GO TO 5270
5277 GOSUB 9200
5280 PRINT " MICROSCOPE"
5281 GOSUB 9000
5282 PRINT " 1=MARCONI 2=HEN
RY 3=JANSSEN"
5283 GOSUB 9050
5284 IF X=3 THEN 5287 ELSE 5
285
5285 GOSUB 9100
5286 GO TO 5280
5287 GOSUB 9200
5290 PRINT " MINERS SAFETY L
AMP"
5291 GOSUB 9000
5292 PRINT " 1=WATT    2=LEE
   3=DAVY"
5293 GOSUB 9050
5294 IF X=3 THEN 5297 ELSE 5
295
5295 GOSUB 9100
5296 GO TO 5290
5297 GOSUB 9200
5300 PRINT " MOTION PICTURE
MACHINE"
5301 GOSUB 9000
5302 PRINT " 1=BAIRD  2=JENK
INS/EDISON   3=BELL"
5303 GOSUB 9050
5304 IF X=2 THEN 5307 ELSE 5
305
5305 GOSUB 9100
5306 GO TO 5300
5307 GOSUB 9200
5310 PRINT " ELECTRIC MOTOR"
5311 GOSUB 9000
5312 PRINT " 1=HENRY/FARADAY
   2=HOLT    3=LAENNEC"
5313 GOSUB 9050
5314 IF X=1 THEN 5317 ELSE 5
315
5315 GOSUB 9100
5316 GO TO 5310
5317 GOSUB 9200
5320 PRINT " MOVABLE TYPE FO
R PRINTING   PRESS"
5321 GOSUB 9000
5322 PRINT " 1=COLT 2=GUTENB
ERG 3=SHOLES"
5323 GOSUB 9050
5324 IF X=2 THEN 5327 ELSE 5
325
5325 GOSUB 9100
5326 GO TO 5320
5327 GOSUB 9200
5330 PRINT " PIANO"
5331 GOSUB 9000
5332 PRINT " 1=PULLMAN    2=
LAENNEC       3=CRISTOFORI"
5333 GOSUB 9050
5334 IF X=3 THEN 5337 ELSE 5
335
5335 GOSUB 9100
5336 GO TO 5330
5337 GOSUB 9200
5340 PRINT " PHONOGRAPH"
5341 GOSUB 9000
5342 PRINT " 1=EDISON  2=BEL
L  3=MORSE"
5343 GOSUB 9050
5344 IF X=1 THEN 5347 ELSE 5
345
5345 GOSUB 9100
5346 GO TO 5340
5347 GOSUB 9200
5350 PRINT " POLAROID CAMERA
"
5351 GOSUB 9000
5352 PRINT " 1=EASTMAN  2=LA
ND  2=PENTA"
5353 GOSUB 9050
5354 IF X=2 THEN 5357 ELSE 5
355
5355 GOSUB 9100
5356 GO TO 5350
5357 GOSUB 9200
5360 PRINT " ROTARY PRINTING
 PRESS"
5361 GOSUB 9000
5362 PRINT " 1=DAVY  2=HOE
3=SHOLES"
5363 GOSUB 9050
```

```
5364 IF X=2 THEN 5367 ELSE 5
365
5365 GOSUB 9100
5366 GO TO 5360
5367 GOSUB 9200
5370 PRINT " RADIO"
5371 GOSUB 9000
5372 PRINT " 1=SHOLES     2=J
ANNEY        3=DeFOREST "
5373 GOSUB 9050
5374 IF X=3 THEN 5377 ELSE 5
375
5375 GOSUB 9100
5376 GO TO 5370
5377 GOSUB 9200
5380 PRINT " RAILROAD CAR CO
UPLER"
5381 GOSUB 9000
5382 PRINT " 1=JANNEY   2=LA
ENNEC        3=PULLMAN"
5383 GOSUB 9050
5384 IF X=1 THEN 5387 ELSE 5
385
5385 GOSUB 9100
5386 GO TO 5380
5387 GOSUB 9200
5390 PRINT " REAPER"
5391 GOSUB 9000
5392 PRINT " 1=McCORMICK  2=
DEERE        3=SPRAGUE"
5393 GOSUB 9050
5394 IF X=1 THEN 5397 ELSE 5
395
5395 GOSUB 9100
5396 GO TO 5390
5397 GOSUB 9200
5400 PRINT " REVOLVER(GUN)"
5401 GOSUB 9000
5402 PRINT " 1=REMMINGTON  2
=WINCHESTER  3=COLT"
5403 GOSUB 9050
5404 IF X=3 THEN 5407 ELSE 5
405
5405 GOSUB 9100
5406 GO TO 5400
5407 GOSUB 9200
5410 PRINT " SLEEPER RAILWAY
 CAR"
5411 GOSUB 9000
5412 PRINT " 1=AMTRAK  2=PUL
LMAN         3=CONRAIL"
5413 GOSUB 9050
5414 IF X=2 THEN 5417 ELSE 5
415
5415 GOSUB 9100
5416 GO TO 5410
5417 GOSUB 9200
5420 PRINT " STEAMBOAT"
5421 GOSUB 9000
5422 PRINT " 1=HENRY 2=FULTO
N 3=JANSSEN"
5423 GOSUB 9050
5424 IF X=2 THEN 5427 ELSE 5
425
5425 GOSUB 9100
5426 GO TO 5420
5427 GOSUB 9200
5430 PRINT " STEAM ENGINE(CO
NDENSING)"
5431 GOSUB 9000
5432 PRINT " 1=AMPERE  2=VOL
TA  3=WATT"
5433 GOSUB 9050
5434 IF X=3 THEN 5437 ELSE 5
435
5435 GOSUB 9100
5436 GO TO 5430
5437 GOSUB 9200
5440 PRINT " STETHOSCOPE"
5441 GOSUB 9000
5442 PRINT " 1=LAENNEC    2=S
HOLES        3=GALILEI"
5443 GOSUB 9050
5444 IF X=1 THEN 5447 ELSE 5
445
5445 GOSUB 9100
5446 GO TO 5440
5447 GOSUB 9200
5450 PRINT " 1ST SUCCESSFUL
SUBMARINE"
5451 GOSUB 9000
5452 PRINT " 1=RICKOVER   2=
HOLLAND      3=JENKINS"
5453 GOSUB 9050
5454 IF X=2 THEN 5457 ELSE 5
455
5455 GOSUB 9100
5456 GO TO 5450
5457 GOSUB 9200
5460 PRINT " MILITARY TANK"
5461 GOSUB 9000
5462 PRINT " 1=SWINTON 2=CHR
YSLER 3=DAVY"
5463 GOSUB 9050
5464 IF X=1 THEN 5467 ELSE 5
465
5465 GOSUB 9100
5466 GO TO 5460
5467 GOSUB 9200
5470 PRINT " TELEGRAPH(ELECT
RIC)"
5471 GOSUB 9000
5472 PRINT " 1=BELL  2=MARCO
NI  3=MORSE"
5473 GOSUB 9050
5474 IF X=3 THEN 5477 ELSE 5
475
5475 GOSUB 9100
5476 GO TO 5470
5477 GOSUB 9200
5480 PRINT " TELEGRAPH(WIREL
ESS)"
5481 GOSUB 9000
5482 PRINT " 1=MARCONI 2=MOR
SE 3=EDISON"
5483 GOSUB 9050
5484 IF X=1 THEN 5487 ELSE 5
485
5485 GOSUB 9100
5486 GO TO 5480
5487 GOSUB 9200
5490 PRINT " TELEPHONE"
5491 GOSUB 9000
5492 PRINT " 1=EDISON   2=BEL
L  3=MORSE"
5493 GOSUB 9050
5494 IF X=2 THEN 5497 ELSE 5
495
5495 GOSUB 9100
5496 GO TO 5490
5497 GOSUB 9200
5500 PRINT " TELEVISION"
5501 GOSUB 9000
5502 PRINT " 1=BELL  2=JENKI
NS  3=BAIRD"
5503 GOSUB 9050
5504 IF X=3 THEN 5507 ELSE 5
505
5505 GOSUB 9100
5506 GO TO 5500
5507 GOSUB 9200
5510 PRINT " THERMOMETER"
5511 GOSUB 9000
5512 PRINT " 1=FAHRENHEIT
2=CENTIGRADE 3=CELCIUS"
5513 GOSUB 9050
5514 IF X=1 THEN 5517 ELSE 5
515
5515 GOSUB 9100
5516 GO TO 5510
5517 GOSUB 9200
5520 PRINT " PRACTICAL TROLL
EY CAR"
5521 GOSUB 9000
5522 PRINT " 1=SPRAGUE    2=W
ESTINGHOUSE  3=BUDD"
5523 GOSUB 9050
5524 IF X=1 THEN 5527 ELSE 5
525
5525 GOSUB 9100
5526 GO TO 5520
5527 GOSUB 9200
5530 PRINT " TYPEWRITER"
5531 GOSUB 9000
5532 PRINT " 1=SWINTON   2=SH
OLES/GLIDDEN 3=COLT"
5533 GOSUB 9050
5534 IF X=2 THEN 5537 ELSE 5
535
5535 GOSUB 9100
5536 GO TO 5530
5537 GOSUB 9200
5540 PRINT " X-RAY MACHINE"
5541 GOSUB 9000
5542 PRINT " 1=ROENTGEN   2=H
OLT  3=LEE"
5543 GOSUB 9050
5544 IF X=1 THEN 5547 ELSE 5
545
5545 GOSUB 9100
5546 GO TO 5540
5547 GOSUB 9200
5550 CALL CLEAR
5551 GOSUB 11006
5555 PRINT "GREAT! YOU GOT A
LL 50 RIGHT.
              ALONG THE WAY YO
U HAD.....      "
5560 PRINT WRONG;" WRONG GUE
SS(ES)"
5561 PRINT
5565 PRINT
5570 PRINT "TO PLAY AGAIN TO
UCH 1

                              "
5571 PRINT "TO STOP TOUCH 2
                              "
5580 CALL KEY(0,KEY,STATUS)
5581 IF STATUS=0 THEN 5580
5582 IF KEY>50 THEN 5580
5583 IF KEY<49 THEN 5580
5584 IF KEY=49 THEN 123
5585 CALL CLEAR
5586 PRINT "TO PLAY AGAIN, T
YPE RUN      THEN PRESS ENTER
                              "
5590 END
9000 GO TO 9005
9005 CALL SOUND(100,500,2)
9009 PRINT
9010 PRINT " WHO WAS THE INV
ENTOR???        "
9015 RETURN
9050 GO TO 9054
9054 PRINT "
                              "
9055 PRINT " WHAT IS YOUR GU
ESS ???         "
9060 PRINT " TOUCH 1, 2 OR 3
"
9064 CALL KEY(0,KEY,STATUS)
9065 IF STATUS=0 THEN 9064
9066 IF KEY>51 THEN 9064
9067 IF KEY<49 THEN 9064
9069 X=KEY-48
9070 RETURN
9100 CALL CLEAR
9105 CALL SCREEN(9)
9110 PRINT " WRONG
                              "
9115 CALL SOUND(750,-3,2)
9120 WRONG=WRONG+1
9121 CALL SCREEN(8)
9122 RETURN
9200 CALL CLEAR
9205 CALL SCREEN(4)
9210 PRINT " YOU ARE CORRECT
                              "
9214 JOE=110
```

continued on page 13

```
100 REM ** BARBIE BOUTIQUE *
* L. Dorais/Ottawa U.G./Marc
h 1989
110 REM
120 CALL CLEAR :: CALL SCREE
N(12):: CALL MAGNIFY(3):: CA
LL CHARPAT(47,A$,58,B$,89,C$
):: CALL CHAR(64,C$)
130 DISPLAY AT(1,7):"BARBIE
BOUTIQUE" :: DISPLAY AT(21,5
):" OPENING THE STORE": :"
 IF @OU USE THE JO@STICK
 RELEASE THE ALPHA LOCK"
140 DIM DR(15),DC(15),SZ(20)
,US(4),UC(4)
150 DATA 0,00C6C600004438,0,
0,0F1F3E78F0E0C0C0,C0C0C0C06
060E0E0,E0F0F83C1E0E0606,060
606060C0C0E0E
160 GOTO 480 :: AR,BC,C,CC,C
R,K,N,NC,S,V,X :: CALL HCHAR
 :: CALL VCHAR :: CALL GCHAR
 :: CALL COLOR
170 CALL SPRITE :: CALL LOCA
TE :: CALL DELSPRITE :: CALL
 C :: CALL CC :: CALL CUR !@
P-
180 DATA 071F3F7F7F7F7F7F,C0
F0F8FCFCFCFCFC,7F7F7F3F1F0F0
703,FCFCFCF8F0E0C080
190 DATA 0000000000010303,03
3F7FFFFFFFFFFF,80F8FCFEFEFFF
FFF,0000000000008080
200 DATA 070F1F1E3C787020,FF
FF7F3F3F3F3F3F,FFFFFDF8F8F8F
8F8,C0E0F0F0783C1C08
210 DATA 1F1F1F3F3F3F7F7F,F0
F0F0F8F8F8FCFC,7E7E7E7E7E7C7
C7C,FCFCFCFCFC7C7C7C
220 DATA 7C7C7C7C7C787878,7C
7C7C7C7C3C3C3C,7878787878787
830,3C3C3C3C3C3C3C18
230 DATA 183C7EFFFF7E3C18,0,
0,0
240 DATA 0000003F3F3F0000,0,
000000F8F8F80000,0
250 DATA 1C1E1E1E1F1F1F3F,3F
3F3F3F3F1F1F0F,1C3C3C3CFCFCF
CFE,FEFEFEFEFEFCFCF8
260 DATA 0,0103030707020000,
070F1F1F3F7F7FFF,FFEFC787070
70703
270 DATA 0707078F8FDFFFFF,FF
FFFFFFFFFFFFFE,0080C0C0E0F0F
0F8,FCBE1E0F07020000
280 DATA 3C7FFFFFFFFFFF7F,7F
3F0F0707070707,00078FFFFFFFF
FFF,FFFFFFFFFFFFFFFF
290 DATA E0F0F8F8F8F8F8F0,F0
E080000000000,0,0
300 DATA 0000000307070F0F,1F
1F1F3F3F3F7F7F,FFFFFFFFFFFFF
FFF,FFFFFFFFFFFFFFFF
310 DATA 808080E0F0F0F8F8,FC
FCFCFEFEFEFFFF,0,0
320 DATA 0F0F0F1F3F7F7F7F,7F
7F7F7F7F7F7F7F,F8F8F8FCFEFFF
FFF,FFFFFF7F7F7F7F7F
330 DATA 7F7F7F7F7F7E7E7E,7E
7E7E7E7E000000,7F7F7F7F7F3F3
F3F,3F3F3F3F3F000000
340 DATA 1F1F3F7FFFFFFFFF,FF
FFFFFFFFFFFFFF,F0F0F8FCFEFEF
EFE,FEFEFEFEFEFEFEFE
350 DATA 1F1F1F3F7FFFFFFF,FF
FFFFFFFEFE0000,F0F0F0F8FCFEF
EFE,FEFEFEFEFEFE0000
360 DATA 010303010103041F,7F
FFFFFFFF7F3F1F,80808038FC380
0C0,FCFEFFFFFFFFEFCF0
370 DATA 001F3F3F3F3F3F00,0,
00FCFEFEFEFEFE00,0
380 DATA 8,16,8,17,9,16,9,17
,10,15,10,16,10,17,10,18,11,
15,11,16,11,17,11,18
390 DATA 12,16,12,17,13,16,1
3,17,14,16,14,17,15,16,15,17
400 DATA 15,26,16,26,15,27,1
6,27,5,26,6,26,5,27,6,27
410 DATA 8,23,9,23,8,24,9,24
,8,25,9,25,8,26,9,26,8,28,9,
28,8,29,9,29,8,30,9,30,8,31,
9,31
420 DATA 10,3,11,3,10,4,11,4
,10,5,11,5,10,6,11,6,14,3,15
,3,14,4,15,4,16,3,17,3,16,4,
17,4
430 DATA 10,7,11,7,10,8,11,8
,14,7,15,7,14,8,15,8
440 DATA 4,4,5,4,4,5,5,5,5,7
,6,7,5,8,6,8
450 DATA 86,121,1,74,120,1,7
4,110,2,74,126,2,73,118,2,73
,134,2,89,116,2,89,132,2
460 DATA 89,120,2,105,120,2,
89,121,1,89,121,1,45,120,1,5
2,120,1
470 ! ** display **
480 CALL CC(32,63,1):: CALL
D("  FINDING THE DOLL"):: CA
LL CC(88,143,1):: CALL COLOR
(2,10,5,3,10,5,4,10,5):: X=8
490 CALL COLOR(X,13,1):: X=X
+1 :: IF X<15 THEN 490 ELSE
X=13
500 CALL VCHAR(6,X,61,13)::
X=X+1 :: IF X<21 THEN 500
510 CALL CC(40,59,2):: CALL
SPRITE(#16,32,2,54,125):: CA
LL SPRITE(#17,36,11,56,121):
: CALL D("CHOOSING THE CLOTH
ES"):: CALL CC(88,143,2)
520 CALL HCHAR(8,23,32):: CA
LL VCHAR(10,6,32,2):: CALL H
CHAR(16,26,32,2):: CALL D("
SORTING THE COLORS")
530 DISPLAY AT(7,2):"HATS";:
: DISPLAY AT(19,1):"BOTTOMS"
 :: DISPLAY AT(11,23):"TOPS"
 :: DISPLAY AT(17,23):"BELT"
540 CALL C(6,11):: CALL C(14
,11):: CALL C(6,22):: CALL C
(14,22)
550 V=8 :: CR=25 :: CC=121 :
: US(1),US(2),US(3),US(4)=20
 :: SZ(20)=1 :: UC(1),UC(2),
UC(3),UC(4)=13 :: BC=5 :: X=
2
560 READ DR(X),DC(X),SZ(X)::
 X=X+1 :: IF X<16 THEN 560 E
LSE CALL SPRITE(#1,60,16,CR,
CC):: CALL CHAR(68,A$,71,B$)
570 DISPLAY AT(21,1):RPT$("X
Z",14):"MOVE CURSORG ARROWSD
JO@STICK":"ACTIONG ENTERDFIR
E   QUITG Q":"  k NEW COLOR
  < TAKE OFF"
580 ! ** game **
590 CALL KEY(3,K,S):: IF S<>
0 THEN 620
600 CALL JOYST(1,X,Y):: IF A
BS(X)+ABS(Y)=4 THEN K=X+2*Y
:: K=9+(K=-4)-(K=-8)-2*(K=8)
:: GOTO 620
610 CALL KEY(1,K,S):: IF S=0
 OR K<>18 THEN 590 ELSE 690
620 IF K=81 THEN CALL CLEAR
:: END
630 IF K=11 OR K=69 THEN CAL
L CUR(CR,-V,17):: GOTO 680
640 IF K=10 OR K=88 THEN CAL
L CUR(CR,V,137):: GOTO 680
650 IF K=8 OR K=83 THEN CALL
 CUR(CC,-V,9):: GOTO 680
660 IF K=9 OR K=68 THEN CALL
 CUR(CC,V,233):: GOTO 680
670 IF K=13 THEN 690 ELSE 59
0
680 CALL LOCATE(#1,CR,CC)::
GOTO 590
690 AR=1-(CC+4>128)-(CR+4>72
)-(CR+4>72):: CALL GCHAR((CR
+4)/8,(CC+4)/8,C)
700 IF C=107 AND CC<180 THEN
 770 ELSE IF C=60 THEN GOSUB
 760 :: GOTO 590 ELSE IF C=6
1 THEN 800
710 IF C<88 THEN 590 ELSE N=
INT(C/4)-20 :: IF INT(N/2)<>
N/2 AND SZ(N)=2 THEN N=N-1 :
: C=C-4
720 GOSUB 760 :: CALL SPRITE
(#N,C,UC(AR),DR(N),DC(N))
730 IF SZ(N)=2 THEN CALL SPR
ITE(#N+1,C+4,UC(AR),DR(N+1),
DC(N+1))
740 US(AR)=N :: GOTO 590
750 ! ** routines **
760 CALL DELSPRITE(#US(AR)):
: IF SZ(US(AR))=2 THEN CALL
DELSPRITE(#US(AR)+1):: RETUR
N ELSE RETURN
770 NC=UC(AR)+1 :: IF NC=17
THEN NC=2 ! NC=new color
780 CALL COLOR(#US(AR),NC)::
 IF SZ(US(AR))=2 THEN CALL C
OLOR(#US(AR)+1,NC)
790 UC(AR)=NC :: GOTO 590
800 BC=BC+1 :: IF BC=12 OR B
C=10 THEN BC=BC+1 ELSE IF BC
=17 THEN BC=2 ! BC=backgroun
d color
810 CALL COLOR(2,10,BC,3,10,
BC,4,10,BC):: GOTO 590
820 !@P+ ** ud subs **
830 SUB CC(X,Y,Z)
840 IF Z=1 THEN READ A$ :: C
ALL CHAR(X,A$)ELSE READ A,B
:: CALL HCHAR(A,B,X)
850 X=X+1 :: IF X<Y+1 THEN 8
40
860 SUBEND
870 SUB C(R,C):: CALL HCHAR(
R,C,107):: CALL HCHAR(R+2+4*
(R=6),C,60):: SUBEND
880 SUB D(A$):: DISPLAY AT(2
1,5):A$ :: SUBEND
890 SUB CUR(X,Y,Z):: X=X+Y :
: IF X=Z THEN X=Z-Y
900 SUBEND
```

continued from page 12

```
9215 CALL SOUND(20,JOE,2)
9216 JOE=JOE+50
9217 IF JOE<600 THEN 9215
9218 CALL CLEAR
9219 CALL SCREEN(8)
9220 IF Z=1 THEN 2000
9221 RETURN
10000 GOSUB 11000
10010 GO TO 80
11000 CALL CLEAR
11006 CALL SCREEN(4)
11010 CALL COLOR(13,7,4)
11015 A$="18183C7E7E7E3C18"
11016 B$="FFFFFFFFFFFFFFF"
11050 CALL CLEAR
11055 CALL CHAR(129,B$)
11060 CALL VCHAR(14,16,129,1
0)
11070 CALL VCHAR(14,17,129,1
0)
11080 CALL VCHAR(14,18,129,1
0)
11088 CALL CHAR(60,A$)
11089 CALL COLOR(4,12,4)
11090 J=0
11091 CALL VCHAR(8,17,33,6)
11092 FOR X=1 TO 500
11093 NEXT X
11100 CALL SOUND(1500,-4,2)
11104 F=1
11105 J=J+1
11106 F=F+1
```

continued on page 16

A Look at Assembler

Calling Subroutines: by Art Green, Ottawa Users Group

Structured programming, that is, the use of subroutines, is as important in Assembler Language as it is in the higher level languages. In Assembler Language, however, it is possible and usual to use smaller subroutines and to use them more frequently. This makes it important to use an efficient calling sequence.

There are two instructions available for subroutine calls, Branch and Link (BL) and Branch and Load Workspace Pointer (BLWP), and their corresponding return instructions, Return (RT) and Return with Workspace Pointer (RTWP).

Generally, BL is used for internal subroutine calls. Internal subroutines are assembled as part of the calling program. The BL instruction saves the return address in R11 and branches to the subroutine using the same workspace as the calling code. Note that if the subroutine uses BL to call other subroutines, R11 must be saved in some way.

Generally, BLWP is used for external subroutine calls. External subroutines are assembled separately from the calling routine. The BLWP linkage is much more elaborate than the BL linkage. A new workspace is set up and as well as saving the return address, the workspace pointer and status registers are also saved. Note that a subroutine called with BLWP can call other subroutines with either linkage without having to save any registers. On return, RTWP restores the caller's workspace pointer and the status register. Restoring the status register, we should remember, restores the interrupt mask.

Now, let us look at the "cost" of these two methods of subroutine call. An example of BL is:

```
       BL   @ISUB           Call Subroutine
       .
       .
       .
ISUB   ...                  Subroutine, ISUB
       .
       .
       .
       RT                   Return to caller
```

The storage cost is 3 words (2 for the BL, 1 for the RT) and the time cost is 5 words (3 words to fetch the instructions, 2 words to save and restore the return address). An example of BLWP linkage is:

```
       REF  XSUB            Define External XSUB
       ...
       BLWP @XSUB           Call Subroutine
       .
       .
       .
       END

       DEF  XSUB            Define entry name
XSUB   DATA XWSP,$+2        Transfer vector
       .
       .
       .
       RTWP                 Return to caller
XWSP   BSS  32              XSUB's workspace registers
       END
```

The storage cost is 21 words (3 for the instructions, 2 for the transfer vector and 16 for the workspace) and the time cost is 11 words (3 words to fetch the instructions, 2 to fetch the transfer vector, 3 to save WP, PC and ST, and 3 to restore WP, PC and ST).

This cost analysis indicates that BLWP linkage is much more expensive than BL linkage. So why use BLWP? One reason is to reduce the third cost of a program, that is, to reduce the effort you expend writing the program. Since a routine called by BL uses the same workspace as the caller, the two pieces of code must coordinate their use of the registers. Sharing the registers is easy for very small subroutines, but gets very difficult if the subroutine is of any size or complexity. This sharing of registers also implies that both the caller and the subroutine "know" what the other is doing, which implies that the subroutine is a special purpose routine just for this program.

When using the BLWP linkage there is no sharing of workspace registers, the registers used by the subroutine belong to that subroutine. This situation is suited to external general purpose subroutines that are designed to be independent of the caller and to be used by several programs.

Since subroutine calling and its associated parameter passing is so important and used so frequently it is worthy of considerable thought to reduce all three costs associated with coding them. You should read the documentation for the CALL and RCALL macros (and their macro definitions) supplied with your favourite macro assembler.

We will continue this discussion next time when we will see that in some cases the higher cost of the BLWP linkage can be overcome.

"The Universe is not user friendly" o

A Look at Assembler

String Searching: by Art Green, Ottawa Users Group

In many programs you find that you have to search down a string looking for a specific character. For example, you have read in a variable length record and want to find the first blank character. You might write the following code.

```
       LI   R1,RECORD       R1->the string
       MOVB *R1+,R2         Load length of string
       SRL  R2,8            Make length a word
       A    R1,R2           R2->just past end
LOOK   CB   *R1+,@BLANK     Look for 1st blank
       JEQ  FOUND           Jump if we found it
       C    R1,R2           At end of string?
       JL   LOOK            Jump no, keep looking
       JMP  NOBLNK          Jump yes, no blank in string
FOUND  DEC  R1              R1->1st blank
       ....
RECORD BSS  81              Length, string
```

Seems straight forward enough, but we could do better. Often this type of code can be improved by ensuring that you will find what you are looking for. If you are sure you will find something then you do not have to check for the end of the item being searched. The example below shows this. It is also coded using macros (supplied with your favourite macro assembler) to demonstrate again how the use of macros reduces the third cost of a program.

```
       LI   R1,RECORD       R1->the string
       LDB  *R1+,R2         Load length byte
       A    R1,R2           R2->just past end
       MOVB @BLANK,*R2      Put blank at end
LOOK   IFB  *R1+,NE,@BLANK,LOOK  Find a blank
       DEC  R1              R1->1st blank
       IF   R1,EQ,R2,NOBLNK Jump if at our blank
       ....
RECORD BSS  82              Length, string, 1 blank
```

By adding the extra MOVB to put the blank at the end of the string (and allowing for that blank in the BSS), we have made a two instruction loop rather than a four instruction loop. We have also done away with the label "FOUND". If this is in a time critical part of your program the extra word of storage used is worth halving the time in the loop. continued on page 9

Fractal Graphics

by Stephen Shaw, England

The first set of programs generates planets or balls - this enables you to create animated sequences just a little bit like the NASA pics! Unlike the skeleton globes you often see generated by computer, frequently see-through wire frames only, these programs generate a true planet, and map all parts but only show what should be shown. Thus you may go into any orbit you wish, polar or equatorial or (with tiny adjustment) any other. You may approach or depart from the planet, and can set the speed of planetary revolution, and even decide whether the poles are to be vertical or horizontal. With a tiny tweak you can set the poles at an angle.

The global mapping algorithm is by Mark Datko, July 1989, and appeared in Issue 4 of Fractal Report, which is available on subscription only, for UK residents at Ten Pounds for Six Issues (irregular but about 6 issues a year) from Reeves Telecommunications Laboratories Ltd., West Towan House, Porthtowan, CORNWALL, TR4 8AX.

The first listing is in MYARC EXTENDED BASIC and requires that language- a program for standard TI ExBas plus JBM103 (from disk library) follows...

```
    {take care of off-screen plots:}
100 ON ERROR 110 :: GOTO 120
110 ON ERROR 110 :: RETURN NEXT
120 RANDOMIZE
130 CALL GRAPHICS(3)
140 REM SEEDS:
      { k high, a low seems to give more detail-
        experiment!!!! }
150 K=RND*.5 :: A=RND*.01
160 CALL WRITE(1,1,1,"  "&STR$(K)&"  "&STR$(A))
170 REM DENSITY OF IMAGE:-
180 NUMITS=42
190 REM VIEWPOINT:-
200 MYLAT=RND0
210 MYLONG=RND0
220 CALL WRITE(1,2,1,"  "&STR$(MYLAT)&"  "&STR$(MYLONG))
230 RADIUS=64
240 PIBY28=PI/28
250 PIPT8=PI+0.8
260 TWOPI=PI*2
270 TWOPI10=TWOPI/10
280 CONRAD=0.0174533
290 MYLAT=MYLAT*CONRAD
300 REM This sort of thing reduces processing time-
    honest....:-
310 SINMYLAT=SIN(MYLAT)
320 COSMYLAT=COS(MYLAT)
330 MYLONG=MYLONG*CONRAD
340 FOR H=1 TO 8
350 CALL DCOLOR(H+2,1)
360 FOR V=TWOPI10 TO TWOPI10*8 STEP TWOPI10
370 X=H*3 :: Y=V
380 FOR I=1 TO NUMITS
390 X=X-K*SIN(Y)
400 Y=Y+X*(1-A*X)
410 IF Y>TWOPI THEN Y=Y-TWOPI :: GOTO 410
420 IF Y<0 THEN Y=Y+TWOPI :: GOTO 420
430 LAT=(X-14)*PIBY28
440 LONG=Y+PIPT8
450 COSLAT=COS(LAT)
460 SINLAT=SIN(LAT)
470 LONG=LONG-MYLONG
480 SINLONG=SIN(LONG)
490 CLCL=COS(LONG)*COSLAT
500 IF CLCL*COSMYLAT+SINLAT*SINMYLAT<0 THEN CALL POINT(
    1,26+RADIUS*(SINLONG*COSLAT+1),22+RADIUS*(1+CLCL
    *SINMYLAT-SINLAT*COSMYLAT)+30)
510 NEXT I
520 NEXT V
530 NEXT H
540 CALL WRITE(1,24,1,"   PRESS SPACE FOR ANOTHER")
550 CALL KEY(5,A,B):: IF NOT B THEN 550
560 RUN
```

Now in TI EXTENDED BASIC using JBM103:

```
100 REM PLANET FOR JBM103
110 REM requires jbm103 disk  from group library
120 CALL LOAD(-31890,56,0)
130 CALL LOAD(-31964,56,0)
140 CALL CLEAR
150 RANDOMIZE
160 K=RND*0.5
170 A=RND*0.01
180 NUMITS=30
190 MYLAT=RND0
200 MYLONG=RND0
210 RADIUS=60
220 PIBY28=PI/28
230 PIPT8=PI+0.8
240 TWOPI=PI+PI
250 TP10=TWOPI/10
260 CONRAD=0.0174533
270 MYLAT=MYLAT*CONRAD
280 SINMYLAT=SIN(MYLAT)
290 COSMYLAT=COS(MYLAT)
300 MYLONG=MYLONG*CONRAD
310 CALL LINK("CLEAR")
320 CALL LINK("SCR2")
325 CALL SCREEN(16)
330 FOR H=1 TO 8
340 COLR=H+2
350 FOR V=TP10 TO TP10*8 STEP TP10
360 X=H*3 :: Y=V
370 FOR I=1 TO NUMITS
380 X=X-K*SIN(Y)
390 Y=Y+X*(1-A*X)
400 IF Y>TWOPI THEN Y=Y-TWOPI :: GOTO 400
410 IF Y<0 THEN Y=Y+TWOPI :: GOTO 410
420 LAT=(X-14)*PIBY28
430 LONG=Y+PIPT8
440 COSLAT=COS(LAT)
450 SINLAT=SIN(LAT)
460 LONG=LONG-MYLONG
470 SINLONG=SIN(LONG)
480 CLCL=COS(LONG)*COSLAT
490 IF CLCL*COSMYLAT+SINLAT*SINMYLAT>=0 THEN 510
500 CALL LINK("POINT",COLR,26+RADIUS*(SINLONG*COSLAT+1)
    ,50+RADIUS*(1+CLCL*SINMYLAT-SINLAT*COSMYLAT))
510 NEXT I :: NEXT V :: NEXT H
520 REM
530 REM SAVE PICS
540 PIC=PIC+1
550 PIC$="PIC"&STR$(PIC)&"_P"
560 S$="DSK2."&PIC$
570 CALL LINK("SAUVE",S$)
580 CALL LINK("SCR1")
590 GOTO 140
```

```
100 REM AUTO GENERATION OF A  PICTURE SEQUENCE FOR COMIC
    SHOW V 4.0
110 RAD@=26
120 REM FOR JBM103
130 RANDOMIZE :: K=RND*0.5 :: A=RND*0.01
140 REM requires jbm103 disk  from group library
150 CALL LOAD(-31890,56,0)
160 LAT@=135
170 LON@=5
180 CALL LOAD(-31964,56,0)
190 CALL CLEAR
200 RANDOMIZE
210 REM
220 REM
230 NUMITS=30
240 RAD@,RADIUS=RAD@*1.06
250 OFF=55-RADIUS/3
260 LAT@,MYLAT=LAT@+RADIUS/8
270 LON@,MYLONG=LON@+8+RADIUS/20
280 PIBY28=PI/28
290 PIPT8=PI+0.8
300 TWOPI=PI+PI
310 TP10=TWOPI/10
320 CONRAD=0.0174533
330 MYLAT=MYLAT*CONRAD
340 SINMYLAT=SIN(MYLAT)
350 COSMYLAT=COS(MYLAT)
360 MYLONG=MYLONG*CONRAD
370 CALL LINK("CLEAR")
```

```
380 CALL LINK("SCR2")
390 CALL SCREEN(16)
400 FOR H=1 TO 8
410 COLR=H+2
420 FOR V=TP10 TO TP10*8 STEP TP10
430 X=H*3 :: Y=V
440 FOR I=1 TO NUMITS
450 X=X-K*SIN(Y)
460 Y=Y+X*(1-A*X)
470 IF Y>TWOPI THEN Y=Y-TWOPI :: GOTO 470
480 IF Y<0 THEN Y=Y+TWOPI :: GOTO 480
490 LAT=(X-14)*PIBY28
500 LONG=Y+PIPT8
510 COSLAT=COS(LAT)
520 SINLAT=SIN(LAT)
530 LONG=LONG-MYLONG
540 SINLONG=SIN(LONG)
550 CLCL=COS(LONG)*COSLAT
560 IF CLCL*COSMYLAT+SINLAT*SINMYLAT>=0 THEN 580
570 CALL LINK("POINT",COLR,OFF+RADIUS*(SINLONG*COSLAT+1)
    ,OFF+RADIUS*(1+CLCL*SINMYLAT-SINLAT*COSMYLAT))
580 NEXT I :: NEXT V :: NEXT H
590 REM
600 REM SAVE PICS
610 PIC=PIC+1
620 PIC$="P"&STR$(PIC)&"_P"
630 S$="DSK2."&PIC$
640 CALL LINK("SAUVE",S$)
650 CALL LINK("SCR1")
660 GOTO 190
```

I ran the above program in Myarc XB, which can be used with JBM103, as follows: Place on your ram disk TIVDP from your Myarc XB disk, SCRO from the JBM103 disk, and the above program, say PROGRAM. Now type:

CALL LOAD("RD.TIVDP")::CALL LOAD("RD.SCRO") :: RUN "RD.PROGRAM"

The catch is if the program bombs and you are returned to GRAPHICS(1) mode, the screen is blank! You must type NEW (and lose the program) to see text mode again!

Having generated a set of pictures you can then animate them using COMIC SHOW Vn 4.0 from the disk library- here is a sample command file for use with that utility...

```
FP DSK2.P1_P
SC 31
SN DSK3.GREENBALL
MP
AP DSK2.P1_P
KW 50
AP DSK2.P2_P
KW 3C
AP DSK2.P3_P
KW 3C
AP DSK2.P4_P
KW 3C
AP DSK2.P5_P
KW 3C
AP DSK2.P6_P
KW 3C
AP DSK2.P7_P
KW 3C
AP DSK2.P7_P
KW 3C
AP DSK2.P8_P
KW 3C
AP DSK2.P9_P
KW 3C
AP DSK2.P10_P
KW 3C
AP DSK2.P11_P
KW 65
AP DSK2.P11_P
AP DSK2.P1_P
GO
```

You end up with a memory image machine code program which will run from Editor Assembler Option 5 or equivalent.

continued from page 13

```
11107 CALL VCHAR(8,17,60,F)
11108 CALL VCHAR(8,17,33,J)
11109 IF F>5 THEN 11110 ELSE
 11105
11110 FOR T=1 TO 500
11120 NEXT T
11121 CALL SCREEN(7)
11130 CALL CLEAR
11131 CALL SCREEN(12)
11132 W=-5
11135 CALL SCREEN(7)
11136 CALL SOUND(900,W,2)
11137 W=W-1
11138 IF W<-7 THEN 11240 ELS
E 11135
11240 CALL SCREEN(12)
11241 FOR V=1 TO 300
11242 NEXT V
11243 CALL SCREEN(7)
11251 FOR Q=1 TO 750
11252 NEXT Q
11253 CALL SCREEN(8)
11800 CALL COLOR(13,2,8)
11805 CALL COLOR(4,2,8)
11900 RETURN
```

HENON MAPPING

Here is a fractal program for TRITON SUPER XB module owners...

```
100 REM TRITON XB
110 REM BEFORE LOADING TYPE:
120 REM CALL FILES(2)
130 REM NEW
140 REM CALL INIT
150 REM CALL DRAWNPLOT
160 REM CALL LINK("GCLEAR")
170 REM AND OFF YOU GO...
180 REM
190 REM S SHAW 9/89
200 REM FROM FRACTAL REPORT
210 REM ISSUE 4
220 REM HENON MAPS
230 REM ANDY LUNNESS, BURY
240 REM
250 REM
260 A=RND*4 ! experiment with A from -4 to +4
270 COSA=COS(A):: SINA=SIN(A)
       {-.1 to +.8 is full map }
       { step .2 is about right.}
       { smaller steps take much longer but give
       finer detail}
       { larger steps lose detail fast }
280 FOR X=-.1 TO 0.8 STEP 0.20
290 FOR Y=-.1 TO 0.8 STEP .20
300 SX=X
310 SY=Y
       { 500 loop required for full inner plot }
       { larger loops seem to add little }
320 FOR IT=1 TO 500
330 IF SX>500 OR SY>500 OR SX<-500 OR SY<-500 THEN IT=
    500 :: GOTO 400
340 XX=SX*COSA-(SY-SX*SX)*SINA
350 SY=SX*SINA+(SY-SX*SX)*COSA
360 SX=XX
370 PLTX=SX+91 :: PLTY=SY+91
380 CALL LINK("MOVE",PLTX,PLTY)
390 CALL LINK("DRAW",PLTX,PLTY)
400 NEXT IT :: NEXT Y :: NEXT X
       { after 15-30 mins...}
410 CALL LINK("SHOW")
420 CALL LINK("GDUMP","PIO.CR")
430 RUN
```

You can use JBM103 to make interesting animated sequences, either vary the "step" in the loop to gradually lose or build up detail, or vary the value of A slightly, to slowly distort the image. This slow distortion is perhaps even more interesting than separate images! Try varying the STEP by about .005, or varying the value of A by .002 or .001 per frame.

The same thing in ordinary Myarc XB:

```
100 ON ERROR 110 :: GOTO 120
110 ON ERROR 110 :: RETURN NEXT
120 STP=0.2 ! MAX DETAIL
130 LOOP=500 ! COMPLETE INNER   LOOP
140 RANDOMIZE
150 A=RND*4-RND*2.5
160 COSA=COS(A):: SINA=SIN(A)
170 CALL CLEAR :: CALL GRAPHICS(3)
180 FOR X=-.1 TO 0.8 STEP STP
190 FOR Y=-.1 TO .8 STEP STP
200 SX=X :: SY=Y
210 FOR IT=1 TO LOOP
220 IF SX>LOOP OR SY>LOOP OR SX<-LOOP OR SY<-LOOP THEN
    280
230 XX=SX*COSA-(SY-SX*SX)*SINA
240 SY=SX*SINA+(SY-SX*SX)*COSA :: SX=XX
250 PLTX=SX+91 :: PLTY=SY+91
260 CALL POINT(1,PLTX,PLTY)
270 NEXT IT
280 NEXT Y :: NEXT X
290 CALL LINK("DUMP",0,16)
300 OPEN #1:"PIO" :: PRINT #1:"A=";A;"  LOOP=";LOOP;"
    STP=";STP:" ":" " :: CLOSE #1
310 RUN
```

RANDOM STARS:

I am not entirely happy with this coding, which I am sure can be improved to give more acceptable images than at present, but this one is a start. It yields similar images to the Rose program published a few issues ago. Some of the images will be unduly simplistic, some will not be at all nice. Some with plots partly off screen will be horribly distorted due to the way our error trapping is working - G handles such things much better! (See later!).

This code is again in Myarc XB:

```
1 RANDOMIZE
2 ON ERROR 3 :: GOTO 100
3 ON ERROR 3 :: RETURN NEXT
4 TRACE
100 REM STAR FRACTALS
110 REM L J VERSCHUEREN
120 REM FRACTAL REPORT 4
130 REM MYARC XB S SHAW 89
140 REM
150 CALL GRAPHICS(3)
160 N=INT(RND*8)+3 ! Number of points
170 R=.5+RND*.5    ! Reduction
180 F=INT(RND*5)+1 ! "fractionation"
182 CALL WRITE(1,4,4,STR$(F))
190 FOR Z=1 TO INT(RND*8)+1
191 READ A  !angle of rotation in radians
192 NEXT Z
200 DATA .25,.5,.75,1,2,3,4,5
201 RESTORE
210 IF INT(N/2)=N/2 AND INT(A/2)<>A/2 THEN 182
220 C=1
230 IF N/2-INT(N/2)=0 THEN IF A<>INT(A)THEN CALL GETC
    ELSE IF A/2-INT(A/2)<>0 TH
EN C=2
240 A=A*PI/N :: K=N-1
250 Y=RND*.5-RND*.5 ! "starting height"
260 REM S=SCALE
270 S=.7
280 X=-.5*S :: H=F-1
290 CALL POINT(0,X+127,Y+127)
291 CALL WRITE(1,24,4,"WAIT")
292 F=0
296 T1=C*N*K^H-1 :: LP=MIN(T1,35):: IF LP<12 THEN RUN
297 GOTO 800
298 CALL WRITE(1,24,4,"     ",4,3,"     ",5,3,"     ")
300 F,M,B,G,J=0
310 FOR I=0 TO C*N*K^H-1
320 M=I :: B=I*A :: G=0
330 IF M/K-INT(M/K)=0 AND G<H THEN G=G+1 :: M=M-K ::
    GOTO 330
340 J=H-G :: X=X+R^J*COS(B)
350 Y=Y+R^J*SIN(B)
360 CALL DRAWTO(F,Y*MULT+CO,X*MULT+CO)
361 IF F=0 THEN F=1
370 NEXT I
380 REM
390 FOR I=1 TO 600 :: NEXT I
400 RUN
410 REM
430 C=A-INT(A):: IF C>0.5 THEN C=1-C
460 STOP
500 END
800 REM
810 FOR I=0 TO LP
811 CALL WRITE(1,5,5,STR$(I))
820 M=I :: B=I*A :: G=0
830 IF M/K-INT(M/K)=0 AND G<H THEN G=G+1 :: M=M-K ::
    GOTO 830
840 J=H-G :: X=X+R^J*COS(B)
850 Y=Y+R^J*SIN(B)
860 MINX=MIN(MINX,X):: MAXX=MAX(MAXX,X):: MINY=MIN(
    MINY,Y):: MAXY=MAX(MAXY,Y)
870 DIFX=ABS(MAXX-MINX):: DIFY=ABS(MAXY-MINY):: SC=MAX(
    DIFX,DIFY):: MULT=110/SC:: SC1=MAX(MAXX,MAXY)::(SC2
    =MIN(MINX,MINY)::CO=190/ABS(SCI-SC2)/1.3+20
873 NEXT I
876 X=A1 :: Y=A2
880 GOTO 298
900 SUB GETC(A,C)
901 C=A-INT(A):: IF C>0.5 THEN C=1-C
902 C=2/C
905 SUBEND
```

o

continued from page 2

restrictions in PC sound, but SoundBlaster owners will get the full TI99/4A sound), 32k memory expansion, MiniMemory, GROM/ROM modules like Extended BASIC, Editor/Assembler and peripheral cards like the p-Code card, RS232 and disk controller.

The RS232 and disk controller would be the only source of incompatibility since, because of timing problems, the disk controller and RS232 hardware cannot be correctly emulated at the hardware level. The DSR ROMs will be patched so that every software level call to the card routines is handled by DOS. That means that you can print to "RS232.BA=9600.CR.LF" without any problems, but terminal emulation software like Telco will not work, because it directly accesses the hardware. Even more complicated will be the disk controller access. I will provide a patch for the ROMs that will enable you to access every filename on your hard disk just by giving the normal DOS name, eg RUN "C:\TI\LOAD" in Extended BASIC. Whenever accessing "DSKx" or using programs that read sectors from these disks (like Disk Manager 1000), the emulator will access special files on your PC hard disk that represent a TI99/4A disk. A utility to copy whole TI99/4A disks to the PC will be provided. Again, software that directly accesses the hardware of the controller will not work (eg COPY-C, Turbo-Copy and RapidCopy).

Another thing you can forget is the Speech Synthesizer, but I sure hope that you did not even expect that. SoundBlaster owners can perhaps play the words in ROM, but I do not think I can ever emulate the Speech Synthesizer, due to the lack of hardware information.

At the moment, the emulator is not absolutely ready. The processor is nearly ready (only three instructions are missing) and the TMS9900 instructions, registers, status and the GPL code are correctly displayed, debugged on screen. If anyone wants to play with the program in its present state, send me a disk and money for return mail (I am a poor student) because I am not allowed to email outside Germany.

Any suggestions, hints, hardware information (needed!) and testers are welcome. Anyone can get the source and I do not want any money. I just love that great machine! o

CQ...CQ...

by Ross Mudie, VK2ZRQ

There are a number of TI99/4A users who are also Amateur Radio Operators, but I am not aware of many programs to use the TI99/4A for Amateur Radio purposes. Some time ago, just before I took on the responsibility of the club's BBS (back in 1986), I wrote a morse code generation program. This program allows a 32K console to key an amateur radio transmitter via the cassette port whilst text was typed in from the keyboard. I am also working on a "Quick Log" program for one of our Radio Amateur members.

Recently I received a letter from an Amateur Radio operator in England asking for details of people who are using the computer for Amateur Radio purposes. What I propose, is to compile a reply and this is where I need some feedback. I would like to hear from all our members who are Amateur Radio operators, what programs they are using and what programs they are prepared to share for Amateur Radio purposes.

Members who are on the BBS could let me know via the BBS mail system. Any others could possibly give me a list at a meeting or drop me a note at 47 Berowra Waters Rd, Berowra NSW 2081. I would like also to provide access to any programs that our members would like to share, the BBS is a good medium. BBS members can upload programs into the Users' Upload/Download area, whilst others may be able to provide programs on disk or tape. o

TI-Base Tutorial #11

by Martin Smoley, North Coast 99ers USA

```
*       Reverse-Print File  RPF1/C

CLOSE ALL
LOCAL SEETOP N 3
LOCAL LINE C 75
  SET RECNUM OFF
  SET SPACES=2
 PRINT (E)
USE LSTFRST
 TOP
 REPLACE SEETOP WITH ANYTHING
 BOTTOM
 PRINT
  SET HEADING OFF
 MOVE -1
 WHILE (ANYTHING<>SEETOP)
 REPLACE LINE WITH ANYTHING | "        ";
 | LAND_DEAL | " " | COST | " " ";
 | SALEPRICE | " " | PROF'LOSS
  PRINT LINE
  MOVE -1
 ENDWHILE
 REPLACE LINE WITH ANYTHING | "        ";
 | LAND_DEAL | " " | COST | " " ";
 | SALEPRICE | " " | PROF'LOSS
  PRINT LINE
  SET RECNUM ON
  SET HEADING ON
 CLOSE ALL
RETURN
```

```
DB  LSTFRST       Sort off
REC ANYTHING LAND_DEAL       COST      SALEPRICE     PROF'LOSS
0000   1      Parcel#28   34456.00    39671.00      5215.00
0001   2      Parcel#84   74246.00    91342.00     17096.00
0002   6      Parcel#21   29876.00    28235.00     -1641.00
0003  44      Parcel#18  123965.00   189913.00     65948.00
0004  99      Parcel#237  49156.00    63945.00     14789.00
0005  33      Parcel#84   44232.00    89491.00     45259.00

DB  LSTFRST       Sort on PROF'LOSS
REC ANYTHING LAND_DEAL       COST      SALEPRICE     PROF'LOSS
0002   6      Parcel#21   29876.00    28235.00     -1641.00
0000   1      Parcel#28   34456.00    39671.00      5215.00
0004  99      Parcel#237  49156.00    63945.00     14789.00
0001   2      Parcel#84   74246.00    91342.00     17096.00
0005  33      Parcel#84   44232.00    89491.00     45259.00
0003  44      Parcel#18  123965.00   189913.00     65948.00

Printout of RPF1/C or RPF2/C
ANYTHING  LAND_DEAL        COST      SALEPRICE     PROF'LOSS

 44       Parcel#18     123965.00   189913.00     65948.00
 33       Parcel#84      44232.00    89491.00     45259.00
  2       Parcel#84      74246.00    91342.00     17096.00
 99       Parcel#237     49156.00    63945.00     14789.00
  1       Parcel#28      34456.00    39671.00      5215.00
  6       Parcel#21      29876.00    28235.00     -1641.00
```

```
Reverse-Print File II   RPF2/C
CLOSE ALL
LOCAL SEETOP N 3
  REPLACE SEETOP WITH -1
LOCAL SAVETOP N 3
LOCAL LINE C 75
  SET RECNUM OFF
  SET SPACES=2
 PRINT (E)
USE LSTFRST
 TOP
 REPLACE SAVETOP WITH ANYTHING
 REPLACE ANYTHING WITH SEETOP
 BOTTOM
 PRINT
  SET HEADING OFF
 MOVE -1
 WHILE (ANYTHING<>SEETOP)
 REPLACE LINE WITH ANYTHING | "        ";
 | LAND_DEAL | " " | COST | " " ";
 | SALEPRICE | " " | PROF'LOSS
  PRINT LINE
  MOVE -1
 ENDWHILE
 REPLACE ANYTHING WITH SAVETOP
 REPLACE LINE WITH ANYTHING | "        ";
 | LAND_DEAL | " " | COST | " " ";
 | SALEPRICE | " " | PROF'LOSS
  PRINT LINE
  SET RECNUM ON
  SET HEADING ON
 CLOSE ALL
RETURN
```

I received a question asking how to print down a page in descending order when TIB sorts everything in ascending order? This is a quick demo of reverse file printing. The size and shape of the DB does not matter, and the sorted item can be numbers or names, the CF will still work. RPF1 goes to the TOP of the file and saves a unique item which it will look for as it moves back up the file. Then it goes to the BOTTOM of the file, prints a record, moves up one record, prints that record, etc. until it finds the record it has saved from the top of the file, where it stops. This theory works fine if you have a field with unique (one of a kind) items. If not, RPF2 gives you an idea on how to handle that problem. It goes to the TOP of the file and saves whatever it finds there. It replaces the item with an item I know is unique and then proceeds in the same manner as RPF1. When it finds the top of the file it replaces the item it switched earlier with the original item, prints that record and stops. If this is a little confusing remember, I am always looking for more questions. So write me and ask. o

Jenny's Younger Set

Dear Jenny,

I have sent another program to you. I hope you like it."

VINCENT MAKER

```
100 RIGHT=0
110 RIGHT2=0
120 R=0
130 R2=0
140 REM QUIZ FOR TWO
150 REM BY VINCENT MAKER
160 CALL CLEAR
170 REM FOR MELANIE
180 PRINT "This is a quiz program that two can play.Just
   follow the directions given."
190 PRINT
200 PRINT
210 PRINT "There will be two chances   for two people to
    enter     numbers.may the best player win."
220 RANDOMIZE
230 INPUT "Give a number which you want to guess out of.
    ":A
240 INPUT "give a number your competitor wants to guess
    out of.":B
250 C=INT(RND*A)
260 D=INT(RND*B)
270 PRINT "Do you both want to guess out of the same
    number?":A$
280 IF A$="YES" THEN INPUT "Which number is that?":J
290 E=INT(RND*J)
300 CALL CLEAR
```

continued on page 23

Multiplan Exercises #5

by Herbert Schlesinger, USA

EDITING:

Now let us look at EDITING the spreadsheet. If you make a mistake in typing and have already pressed <ENTER> your mistake appears in the spread sheet. To correct this, simply re-enter that cell with the correct entry. Now the cell on the screen should hold the proper information. If you catch the mistake before it is entered you may press FCTN 9 to backspace over what you have written. This erases what was there and you may retype the proper entry. OR you can simply press CTRL = to cancel what you have started. This brings back the main Command Line and you start over. However, this section concerns Editing so if we find an error of spelling or of a formula we can select Edit from the main command line:

Bring up the screen "NAMES" When you finish this exercise on Editing your screen will look like "NAMES1".

Note the first label is "Principle" which is the wrong spelling.

1. Place the cell pointer in the cell containing the error.
2. Press E(dit) from the main menu by pressing "E".

The contents of that cell will appear in the menu area surrounded by quotation marks, thus:

EDIT: "Principle"_

Press the backspace key (FCTN 9) three times and you will have:

EDIT: "Princip_

Type in the correct letters (al) and the line is

EDIT: "Principal_

Remember to close the quotes and when the line looks like this:

EDIT: "Principal" PRESS <ENTER>

Using the same method of placing the cell pointer on the cell which needs editing, selecting the Edit option, we can correct numbers and formulas. Neither of these need quotation marks, only text is so enclosed. Now to the matter at hand: We see that the Payment in R5C2 is a negative number. This can not be right; so we Edit that cell and the screen shows:

(In order to get the caret ^ to show we had to redefine another character to print this out (transliteration).

R1C2*R2C2/(1-(1+R2C2) ^ R3C2)

This is not easy to correct but first write down the formula for calculating a payment on an ordinary annuity,

$$\text{PAYMENT} = \text{PRINCIPAL} * \frac{\text{INTEREST}}{\left[1-(1+\text{INTEREST})\right]^{-\text{TERM}}}$$

Our spread sheet stores the Principal in R1C2, the interest in R2C2, and the Term in R3C2 the formula is:

$$\text{PAYMENT} = \text{R1C2} * \frac{\text{R2C2}}{\left[1-(1+\text{R2C2})\right]^{-\text{R3C2}}}$$

OR as a formula for the spread sheet:

R1C2*R2C2/(1-(1+R2C2)^-R3C2)

The error is that the rightmost calculation. R3C2 should be negative. Make the correction by backspacing over that part of the formula and typing it in again after placing a - sign.

Easier in this case, simply change the term in R3C2 to a minus quantity and the formula need not be changed.

Once you make the correction, press <ENTER> and the formula is now altered.

Here is a chart of the editing keys:

Task	4A	Geneve	What it does
Character Left	FCTN 4	F4	Moves the cursor left over character without erasing it.
Character Right	CTRL 4	CTRL 4	Moves the cursor to the right a character without erasing.
Word Left	FCTN 5	F5	Moves cursor one word left without erasing.
Word Right	CTRL 5	CTRL 5	Moves cursor right a word not erasing.
Backspace	FCTN 9	F9	Moves cursor back one space long as it moves
Cancel	CTRL =	CTRL =	"Undoes" any editing, returns the original content to the cell
Delete Forward		F1	Deletes as cursor moves to right

Please note that the spreadsheet we have been working on does calculate the payment for a loan, but NOTE well that the interest is the rate per PERIOD, not per year; and the period or term is the number of PAYMENTS, not the number of years.

USING THE INSERT OR DELETE OPTION OF THE MAIN MENU:

Bring up the screen "INSERT"

INSERT A ROW:

We have a list of six names as labels as well as the cell labeled "Total". Now suppose we want to insert the name "Clement" in its proper alphabetical order:

1. Place the cell pointer to R4C1, where we want the name to appear.
2. Select the Insert command; there are two options on the screen:

   INSERT: Row Column

3. Select R(ow). Then the screen asks for more information:

   INSERT ROW # of rows: 1

   before row: 4 between columns: 1 and: 63

This says that Multiplan is going to insert 1 row, before row 4, for all columns (1 through 63).

4. Press <ENTER> and all the rows will move down a notch.

Now type the new data: Clement 1111.11

Before we made the insertion R7C2, the total, showed a formula: SUM(R[-6]C:R[-1]C and after the insertion it automatically adjusted to include the new row. It now shows: SUM(R[-7]C:R[-1]C. (this is shown on the bottom line of the screen between the hylited cell name and the % of Multiplan used. See pages 4 & 7).

TO INSERT A COLUMN:

Place the cell pointer to the place you want to insert the column, select the Insert and the Column commands and answer the prompts. Certain defaults are there and, unless changed, will be the extent of the insert. DELETEs are done exactly the same way. Just be sure of the number of rows or columns you want to insert or delete.

Be careful of the number of columns you are inserting or deleting as you can screw up a sheet by off-setting figures which are moved or not moved if the wrong numbers are used. Also note that while a formula adjusts itself if a row or column is inserted in the center or bottom of a group, if row 1 or column 1 is inserted, the adjustment is not made. Always check the formulas by using the Format Options command which will let you see the formulas instead of the figures in the spreadsheet. Sometimes you may delete a cell that was referenced by a formula while doing a delete. In that case the cell will display a message: #REF! which means that the cell the formula refers to does not exist. In that case you must re-enter the formula to make the proper references.

MOVING ROWS AND COLUMNS:

Place "MOVECOL" on screen.

The Move command allows the moving of rows or columns:

Suppose on our sample sheet, "Davis" marries and changes her name to "Baker", so we wish to put this name in its proper position.

1. We place pointer on the row to be moved, here Row 6
2. Select the Move command from the menu and we see:

```
MOVE: Row Column
```

If you select Row you get:

```
        MOVE ROW from row: 6
   to before row:6        # of rows: 1
```

3. Select row six (the default because we are on row 6) by pressing CTRL A (or CTRL2) the TAB key, which switches us over to the next option "before row:"
4. Type in 2 since Baker will before Bell. Leave the # key as it is.
5. Press <ENTER>

Davis will be moved to the new location and you may then Edit the name. (Change it to Baker)
X

Columns can be moved in just the same way. Try it by moving col 1 to the left of col 3. The submenu looks like:

```
        MOVE COLUMN: 1
  to left of column: 3 # of columns 1
```

Go further, move both of these columns into cols 4 and 5:

```
     MOVE COLUMN from column: 1
  to left of column:6 # of columns: 2
```

Even more complex moves can be made using the copy command.

If it is not already there, put "MOVECOL" on screen.

There are too many rows on this sheet to be able to see them at once so we are going to use the COPY command to "split" them into two columns which we can show on one screen. Since there are 30 rows it is natural to split them at 15. Here is how:(This works better on a 90 col Geneve than the 4A)

1. Position pointer in the cell you want to move the data to, here we use R1C4.
2. Select the Copy command giving us three options:

```
COPY: Right Down From
```

Select From (by depressing the space bar twice. Remember?) Or press F.

3. The next options are: (since our pointer is in R1C4)

```
COPY FROM cells: R1C4   to cells: R1C4
```

Change the cells to copy from to R16C1:R30C2. Do this by either typing or using the cell pointer to the cells in the range.

4. Press <ENTER>.

Notice that the cells in Rows 16 through 30, columns 1 and 2 will move to the new area, however we will have trouble when we perform the cleanup in the next section - the figures for the total should show an error - #REF!. This can be corrected by placing the cell pointer at R15C5 and entering this formula:

```
SUM(R1C2:R15C2)+SUM(R1C5:R14C5)
```

We are merely adding the two parts into which we split the spreadsheet. But since we copied rather than moved this data we must "clean up" our work.

CLEANING UP:

Using the BLANK command from the menu allows you to remove by erasing unwanted cells or groups of cells.

1. Place the pointer in the upper left corner of the range you wish to remove (R16C1).
2. Select the Blank command, and you have:

```
BLANK cells:R16C1
```

3. Press the colon : key for a range.
4. Move the pointer to the end of the group of cells in the range (here it is the bottom row of the second column).

```
BLANK cells:R16C1:R30C2
```

5. Press <ENTER>.

The specified cells will be erased and your spreadsheet will look like screen "MOVECOL1". NOTE that if the error of the total label was not taken care of before, it presents itself at this point.

To clear the entire spreadsheet from the screen use the Transfer and Clear options from the menu. You will be asked to confirm this so press "Y" if you want to clear the screen. If you do this all the data is lost forever unless you first Transfer-Saved it under some name. o

Treasurer's Report

by Geoff Trott

This is the time of year when we have to start to prepare the annual report for the past year. From my point of view, we have had a very successful year, with still many people showing interest in the group and our computer. I think we provide a good mix of activities. If you have other ideas, please tell someone. By the way, we did win our final, if you were going to ask!

Income for May	$2741.10
Payments in May	$1762.23
Excess of income over expenses for May	$978.87

o

Beginning Forth - part 7

by Earl Raguse, UGOC, CA USA

DISK DIRECTORY

In Beginning Forth #5, I promised a disk DIRectory word using MYSELF, instead I got carried away in Beginning Forth #6 and started into the subject of music. Now music is a very good subject, but I did not intend to skip my DIRectory word, I suppose its okay though, because now that you can write a disk full of musical works, you will need a DIRectory to keep track of them and to access them with a single keystroke. It so happens, that Screens #6 and #7 are the DIRectory on one of my nursery rhyme disks, and includes a song that I will be using as a demonstration of new music techniques that was promised last time.

But, to get on with DIR, it is relatively simple except for some new words KEY, CASE, OF, ENDOF, ENDCASE, and of course, MYSELF. Screen #6 contains the word DIR, and Screen #7 contains the text of the actual directory that is printed to the CRT when it is loaded by 7 LOAD, in DIR. The next word KEY is similar to CALL KEY in XB, when executed, it causes Forth to look for a keyboard entry, it then puts the ASCII code on the stack. Since DIR expects you to enter a number, 48 - converts the ASCII code to a value between 0 and 9. This value is DUPed and compared with zero and nine to see if it is a legitimate entry. If it is not, a true flag will be left for IF which will cause the incorrect value to be dropped and MYSELF executed to return to DIR, until you quit goofing off and enter an appropriate number.

Assuming that you did enter a proper number, the word CASE will be executed. Internal to CASE are the words OF and ENDOF. OF compares the value on the stack with the value just preceeding OF, if they match, the words between OF and ENDOF are executed, if not, on to the next OF. In this case, all values, but zero, cause a specific screen to be loaded. This screen number should be the screen where the program listed on Screen #7 is located.

Notice the similarity of CASE and the command ON from XB. Case of course is more versatile, because anything can be executed directly. After all the OF words have tested their values, we come to the word ENDCASE which terminates the comparison operation. If no match was found, execution proceeds to the word following ENDCASE. In this case, one cannot exit this loop without a match because of the comparison checks made earlier.

To use DIR for you own purposes, you need only to replace the titles listed on Screen #7 with your own titles, and to substitute the correct numbers to LOAD on Screen #6. If you do not have 9 programs to list for selection, you may leave blanks on Screen #7 and put 7's in front of those LOAD's. That will cause an erroneous selection to just load the directory again. If you have more than 9 programs, it is possible to make the directory in several pages, but that is a bit more complicated and best left until later. I do suggest you leave ABORT as one selection, so you can escape.

MUSIC AGAIN

Now, back to music. One of the things that the previous PlayNote words, PNn, did not allow, was for one to play one note continuously while playing other notes intermittently. Music is frequently written in this manner. This feat can easily be accomplished using a negative note duration. You will recall that a negative duration in XBASIC causes the executing CALL SOUND to cease and the new CALL SOUND to execute immediately without interruption of the sound. This works in much the same way, with the Forth words we defined last time. Forth, however is much faster, so we need to insert a delay. In XB, the loop execution is slow enough to provide its own delay, but we have very little control.

Screens #61 through #67 provide examples of nearly all the concepts discussed to date.

Please refer to the word CHD, defined last time, on Screen #47, to play chords, CHD expects 3 notes on the stack, to be read and stored by AN (All Notes). The volume of each sound generator can be specified separately and set with AV (All Volumes), and the note length can be stored in LENG by L. If one simply stores a negative duration in LENG, successive executions of CHD will cut off the prior sounds so fast you probably will not hear them. This was fixed by defining 3CH on Screen #63. 3CH executes, AN to read and store 3 notes from the stack for CHD, then a 250 millisecond delay insures that the next 3CH does not interrupt too soon.

Screens #61 and #62 are sort of a rehash of the techniques used last time for Song at Twilight. The words DT1 through DT8 print the lyrics while, the words DD1 through DD8 play the melody, using PN1 and PN2. DD sort of puts it all together. You will just have to trust that I read the sheet music correctly, or check it yourself, if you can find it. By now you should be able to read the musical staff, even if you have to have the text book handy. I always do. Not much new here, but necessary to demonstrate that even though this sounds quite good, it can be done a lot better, by using 3CH.

Its all rather easy as one can see by the listing of Screens #63 through #66, which includes the words OD1 through OD17, (OD= Oh, Dear, What Can The Matter Be? Johnny's So Long At The Fair). The word SOD (Set Oh Dear) sets Octave 2, a negative LENG of -1000, and generator volumes 1,2 and 3 to 2, and the noise generator volume to 30. In this case we do not set a TEMPO, because the speed of play is set by the 250 MS in 3CH. Note how the word OD1 plays the notes A and C continuously while the bass note sequence F A C is being played. This is what gives this version its rhythm.

Observe also, that instead of changing Octave, here the frequency for different generators is moved up and down an octave by 2* and 2/ respectively, these are UFW's from Beginning Forth #4 (Screen #2). This lets one play notes in three different octaves at one time. Also since CHD requires 3 notes on the stack, if less than 3 notes are to be played, a Rest must be used to fill in the blanks. Notice how easy it is to read this format. The measures are all divided into separate words, and each can be tested by itself. Even tin-ears, like myself, can figure where some thing is wrong this way. You may even become a musician. Again, as before, OD on Screen 66, puts it all together.

```
SCR #6
(DISK DIRECTORY EGR 12/29/87)   DECIMAL
: DIR  7 LOAD KEY 48 - DUP DUP 0 < SWAP
  9 > OR IF DROP MYSELF THEN  CASE
  0 OF ABORT            ENDOF
  1 OF 48 LOAD          ENDOF
  2 OF 53 LOAD          ENDOF
  3 OF 55 LOAD          ENDOF
  4 OF 57 LOAD          ENDOF
  5 OF 69 LOAD          ENDOF
  6 OF 61 LOAD          ENDOF
  7 OF 72 LOAD          ENDOF
  8 OF 68 LOAD          ENDOF
  9 OF 39 LOAD          ENDOF
  ENDCASE ;
: IT ;
```

EDITOR'S NOTE: There are many screens mentioned in this series. Many of them I edit and include with the article but some of them are difficult to edit within the limits of the 56 'elite' columns that I have to work with. From here on, if anybody has need of a particular screen that you do not find included with the article in the TND then give me a ring on the phone and I will bring it along to the next meeting on a disk and we will do an exchange. Bob).

continued on page 7

Newsletter update

by Geoff Trott

The Boston Computer Society of February has its usual article by Ron Williams on the UCSD P-code system. This time it is on assembler programming. It also has articles on configuring Funnelweb, Diskreview and the products from OPA all reprinted from other sources.

The March issue of Bug Bytes from Brisbane has the second of a TI-Base tutorial series, a review of the Red Baron flight simulator and a report on the products of OPA. OPA are promising to release an 80 column card which replaces the video processor in the console with a small board containing a 9958 video chip. They call it The Image Maker (TIM) but have struck some problems and so have re-programmed the operating system and produced a device called Son of a Board. TIM is supposed to sell for US$179 and SoB for US$49, or free with TIM.

The February issue of The Front Ranger has an article on Extended BASIC subprograms by Earl Raguse, character sets by Andy Frueh, New-age/99 by Jack Sughrue, a very detailed explanation of TIM and SoB from OPA by Steve Funkhouser and Asgard News update. This update contains information about their latest software releases, news about the MIDI interface package and their 80 column card which is like the Mechatronics 80 column card. They require orders (US$250) to go into production.

The March issue of the Ottawa Newsletter has Lucie Dorias' Fast Extended BASIC and some local news.

The March issue of The Pug Peripheral (Pittsburgh) has the second article on putting a PC power supply into a PEBox by John Willforth. He also shows how to quieten the fan by reversing its direction of rotation. There is an article on Multiplan version 4.0 by Audrey Bucher and a supportive article on Funnelweb by Jack Sughrue.

The March issue of ROM (Orange County) has TI Bits #34 from Jim Swedlow, how to produce large letters by Newt Armstrong and XBASIC Miscellany #1 by Earl Raguse.

Topics (LA) of March has articles by Jack Sughrue (New-Age/99 #14), Earl Raguse (How Subprograms are used), Chick De Marti (The Cracker Barrel) with some programs from Chick de Marti.

The March issue of Spirit of 99 (Central Ohio) has articles on Appreciating your Programmers, Simple Programs and Putting it all Together #9 by Jim Peterson, a program to add carriage returns to the end of lines by Jean Hall, New-Age/99 #12 by Jack Sughrue and TI-Base tutorial #20 from Martin Smoley.

The March issue of TI Focus (Hamilton Ontario) has news from Tom Arnold that this will be the last year for him in the TI99/4A community. That saddens me as I met Tom in 1987 at the Johnson's home. He has done a lot for the Hamilton group and I always enjoy reading TI Focus. Gary Bowser of OPA was present at their last meeting and talked about his latest products. The rest of the issue was made up of a reprint of some history of the TI99/4 by Charles Good of the Lima Group.

TIdbits of March (Memphis Tennessee) contains Some Curiosities by Tony McGovern, Copyrights by Shelby Jett, Poor Man's Loader program by Rick Rothstein and other interesting information including the announcement from Asgard that if you want one of their 80 column card please place an order. o

Newsletter update

by Bob Relyea

TIBUG (Brisbane), April, 1991: Editorial, Trading Post, What's New (release of Myarc MDOS V1.15 and 1.15h, TI-Base Tutorial No.3, News Digest, Tips from the Tigercub No. 28, DV/80 files to Merge format, To My Darling Husband, Q & A on ramdisks, RGB Interface, Programming Tips, TI Image Maker, German TI Users, Wonkapillar - Basic.

ATICC (Adelaide), April, 1991: Coordinator's Report, Editorial, Ingenuity by German TI Users, Users Notes, Printer Potpourri, Beginning Modems, Far Out by Dick Schaydel, Enhanced Basic by Stephen Shaw, Why Learn to Program? by Jim Peterson, Blackjack (program to type in), Programming Tips by Mark Schafer, Handy Tips for the TI99/4A.

TI UP TIt BITS, December, 1990: Editorial, TI Print Shop, Information on Horizon Ramdisk, Flow Chart for Government Contract, My Subprograms Listed by Earl Raguse, Music Subprogram Listings by Earl Raguse, Speak To Me by Chick De Marti, All About Character Sets by Andy Fruer, Converting Data Bases to TI-Base by Bill Gaskill.

February, 1991: Editorial, Using the 2Columns Program by Earl Raguse, Tips from the Tigercub No. 61 by Jim Peterson, Newsdigest, Colin's Colour Printer, The Cracker Barrel by Chick De Marti, Fish (program to type in), Yet Another Joystick Adapter (complete with wiring instructions), Part 1 of Setting Up a Spreadsheet, Personal Budget.

SPIRIT OF 99, April, 1991: Lima Conference Update, Programming Music The Easy Way by Jim Peterson, Tips From The Tigercub No. 63 by Jim Peterson, What's Hott by Irwin Hott, Puzzle-12 by Wesley Richardson, New-Age/99 No. 13 by Jack Sughrue, TI-Base Tutorial 21.1.1 & 21.1.2.

THE FRONT RANGER, April, 1991: The President Speaks, Address Labels from TI-Writer Valuefile by Sam Donato, Replace Your Cartridge Contacts by Rich Lynn, Asgard On-Line News No. 4 by Chris Bobbitt, OPA News No.1 by Gary Bowser, Rave Keyboard by Charles Good, TI Users by Charles Good.

TI-FOCUS, April, 1991: News and Views by Tom Arnold, Software Reviews by Tom Arnold, Club Page by Tor Hansen, More TI-Base by Rick Lilley, New-Age/99 No. 12 & 13 by Jack Sughrue.

THE BOSTON COMPUTER SOCIETY, March, 1991: Listen by Justin Dowling, Notes on Laser Printing With The TI99/4A by Larry Fairbanks, You Do Not Have To Have It All by Jim Peterson, Control of the CS1 Remote by Ed Hall.

THE OTTAWA NEWSLETTER, April, 1991: Fast Extended Basic by Lucie Dorais, Balldrop (program to type in) by Lucie Dorais, Ti Computing In France With The Myarc Disk Controller, Update on the 1991 TI Fest by Bill Gard, Myarc HFDC Hard Disk Controller (long article) by Jan Alexandersson.

UGOC ROM, April, 1991: The New Editor Speaks by Earl Raguse, Do You Have As MUc Sense As A Goose? by Earl Raguse, From The President by Ben Hathway, TI Bits No. 35 by Jim Swedlow, Shakespeare on The TI-99/4A, Board Meeting Report, Membership Corner, Subprograms And How They Work by Earl Raguse.

TIdbits, April, 1991: President's Bit by Gary Cox, In The News by Gary Cox, A Brief History of Bulletin Boards by Bill Garrard, Multiplan V.4.0 Users by Audrey Bucher, Why Does Multiplan Run On Different Systems? by Garth Potts, Mail LIst Manager by Bill Gaskill, After The Ball Was Over by Jim Deards, For Sale, Editor's Bit.

LEHIGH 99'ER, Jan/Feb, 1991: RLE Graphics, United Parcel Service by John Geisinger, The State of the Newsletter Address, various programs to type in - Wordcount 1 & 2, Movement, TVconverge, Colortest.

March, 1991: Jack's Basement, The Rave PS/2 Expansion Box - a review by Dave Ratcliffe, The Animator by Brad Snyder.

April, 1991: President's Comments, How Cheap Can You (I) Get? by John Geisinger, A Warning To Husbands, Program to type in (BOOT), Computer Info Storage and Disks by R. Lumpkin, Program - Diagnosis. o

Regional Group Reports

Meeting Summary For July

Banana Coast	14/07/91	Sawtell
Carlingford	17/07/91	Carlingford
Central Coast	13/07/91	Saratoga
Glebe	11/07/91	Glebe
Illawarra	08/07/91	Keiraville
Liverpool	12/07/91	
Northern Suburbs	25/07/91	
Sutherland	19/07/91	Jannali

BANANA COAST Regional Group
(Coffs Harbour area)

Regular meetings are held in the Sawtell Tennis Club on the second Sunday of the month at 2 pm sharp. For information on meetings of the Banana Coast group, contact Kevin Cox at 7 Dewing Close, Bayldon, telephone (066)53 2649, or John Ryan of Mullaway via the BBS, user name SARA, or telephone (066)54 1451.

CENTRAL COAST Regional Group

Regular meetings are normally held on the second Saturday of each month, 6.30pm at the home of John Goulton, 34 Mimosa Ave., Saratoga, (043)69 3990. Contact Russell Welham (043)92 4000.

GLEBE Regional Group

Regular meetings are normally on the Thursday evening following the first Saturday of the month, at 8pm at 43 Boyce St, Glebe. Contact Mike Slattery, (02)692 0559.

ILLAWARRA Regional Group

Regular meetings are normally on the second Monday of each month, except January, at 7.30pm, Keiraville Public School, Gipps Rd, Keiraville, opposite the Keiraville shopping centre. A variety of activities accompany our meetings. At the last meeting we looked at various functions of Multiplan. Contact Lou Amadio on (042)28 4906 for more information.

LIVERPOOL Regional Group

Regular meeting date is the Friday following the TIshug Sydney meeting at 7.30 pm. Contact Larry Saunders (02) 6447377 (home) or (02) 7598441 (work) for more information.

NORTHERN SUBURBS Regional Group

Regular meetings are held on the fourth Thursday of the month. If you want any information please ring Dennis Norman on (02)452 3920, or Dick Warburton on (02)918 8132.

Come and join in our fun. Dick Warburton.

SUTHERLAND REGIONAL GROUP

The month of May saw three of our members trek across to the Liverpool regional meeting at Larry Saunders house. The hospitality was first class as usual and a good night was had by all.

Larry had his usual box of tricks, with demonstrations of the latest software available.

On the local front, our May meeting concentrated on running the latest version of Page Pro. Seems as though there are new utilities being released to run with this software, more quickly than you can master the previous one.

Regular meetings are held on the third Friday of each month at the home of Peter Young, 51 Jannali Avenue, Jannali at 7.30pm.

Peter Young
Regional Co-ordinator

TIsHUG in Sydney

Monthly meetings start promptly at 2pm (except for full day tutorials) on the first Saturday of the month that is not part of a long weekend. They are held at the RYDE INFANTS SCHOOL, Tucker Street (Post Office end), Ryde. Regular items include news from the directors, the publications library, the shop, and demonstrations of monthly software.

The cut-off dates for submitting articles to the Editor for the TND are:

August	7 July
September	11 August

These dates are all Sundays and there is no guarantee that they will make the magazine unless they are uploaded by 6:00pm, at the latest.

****

TIsHUG Meetings for Sydney, 1991

July
The second Buy, Swap and Sell day. Bring things for sale and also your money for the terrific bargains. You have to be early to beat Rolf to a bargain!

August
Demonstrations of the latest software or hardware. Watch this space for details.

September
Demonstrations of the latest software or hardware. Watch this space for details.

October
The third Buy, Swap and Sell day. Your last chance this year to get some money for Christmas presents or to get an early present for yourself.

November
The second all day tutorial session.

December
The Annual General Meeting followed by some festive eats and drinks. Make sure you attend and give your support to all the workers in the club. o

continued from page 5

He gives the command to save his work and hits the "return" key. The disk drive spins and suddenly he realizes what he has done. This time the disk drive makes the sound of lost dreams, fame, and cash. Ted screams as he falls to his knees and bangs his head on the keyboard. What is this? His program is ok. He forgot to close the disk drive door. What do you know, saved by his own stupidity.

Ted decides now is a good time to make a back-up copy, next time he may no be so lucky. Now what? He cannot copy it! His copy protect scheme will not let him. What do you know, screwed by his own arrogance. Ted wimpers as he slumps into his chair. Suddenly, he perks up when he realizes he could rewrite his copy protect scheme and sell it to people who wish to protect their own programs. He will also use the example of his blunder in his book and change its name to: "Careful Computing is Crucial", Anonymous. o

continued from page 18

```
310 INPUT "Input the first players guess-":F
320 IF A$="YES" THEN 350
330 IF F=C THEN RIGHT=RIGHT+1
340 GOTO 370
350 IF F=E THEN RIGHT2=RIGHT2+1
360 INPUT "Input the 2nd players guess-":G
370 IF A$="YES" THEN 400
380 IF G=D THEN R=R+1
390 GOTO 410
400 IF G=E THEN R2=R+1
410 PRINT "The first player's total so far is";RIGHT,
    RIGHT2
420 PRINT
430 PRINT "The 2nd players total so far is";R,R2
440 PRINT
450 PRINT "Do you wish to play again?":AB$
460 IF AB$<>"YES" THEN END
470 PRINT "Do you wish to play for the same number
    again?":D$
480 IF D$="yes" THEN 300 ELSE 140
```
o