TISHUG NEWS DIGEST

Focusing on the TI99/4A Home Computer

Volume 10, Number 10 November, 1991

Registered by Australia Post - Publication No. NBH5933

Book now for Australia's first

TI-FAIRE

of the 1990's.
To be held in Sydney,
on November 14th, 1992
Contact Richard Warburton on
(ISD) 61-2-9188132
(STD) 02-9188132

Sydney, New South Wales, Australia $3

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

TIsHUG News Digest

November 1991

All correspondence to:

P.O. Box 214
Redfern, NSW 2016
Australia

The Board

Co-ordinator
Dick Warburton (02) 918 8132
Secretary
Terry Phillips (02) 797 6313
Treasurer
Geoff Trott (042) 29 6629
Directors
Rolf Schreiber (042) 84 2980
Russell Welham (043) 92 4000

Sub-committees

News Digest Editor
Bob Relyea (046) 57 1253
BBS Sysop
Ross Mudie (02) 456 2122
BBS telephone number (02) 456 4606
Merchandising
Percy Harrison (02) 808 3181
Publications Library
Warren Welham (043) 92 4000
Software library
Rolf Schreiber (042) 84 2980
Technical co-ordinator
Lou Amadio (042) 28 4906

Regional Group Contacts

Central Coast
Russell Welham (043) 92 4000
Coffs Harbour
Kevin Cox (066) 53 2649
Glebe
Mike Slattery (02) 692 0559
Illawarra
Lou Amadio (042) 28 4906
Liverpool
Larry Saunders (02) 644 7377
Northern Suburbs
Dennis Norman (02) 452 3920
Sutherland
Peter Young (02) 528 8775

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 9.30 am on 2nd of November at Ryde Infant School, Tucker Street, Ryde. It will be a full day tutorial with sessions to suit all tastes.

Printed by
The University of Wollongong
Printery Services.

TIsHUG News Digest ISSN 0819-1984

Index

Stop Press: $40 off TIM 80 column card

We are currently taking orders for the TI[illegible] upgrade for $165, a saving of $40 off the normal price. Send the m[illegible] with your order to Percy Harrison in the TIsHUG Shop [illegible]ctober. The TIM 80 column upgrade is compatible with RGB ana[illegible]onitors like the Wang monitors and Al Lawrence will be demonstrating his 80 column system at the November meeting.

Editor's Comment

by Bob Relyea

I was encouraged at the roll-up for the BUY, SWAP and SELL DAY at the last Sydney meeting. Money will only stretch so far and I was tossing between purchasing a system that was out for sale and taking advantage of this unexpected offer of getting the TIM to upgrade my system to 80 columns. This will make the aquisition of a high resolution monitor more useful. I choose the TIM and will look forward to receiving it later this year. Anybody got a colour monitor for sale? The other thing that is on my mind is the proposal for a FAIR next year in Sydney. It could be a real boon to the club if we put a lot into it. Putting anything less than a lot into it would be a waste of time and might possibly drain club resources to a dangerous level, especially if we are thinking about bringing somebody over from overseas. This fair could be a watershed for the club. If it is successful it could bring in enough new members to keep the club going for quite a few more years. I am willing to do my bit towards it, what about you?.

Co-ordinator's Report

by Dick Warburton

Directors' Annual Report for
Period Ending 31st December 1991

Directors

R Warburton	Co-ordinator
T Phillips	Secretary.
G Trott	Treasurer.
R Schreiber	Software
R Welham	Library.

The Directors are supported by other TIsHUG members who have accepted responsibilities for the following activities.

Shop	P Harrison.
Editor TND	R Relya.
BBS	R Mudie.
Technical	L Amadio.
Re-inking	R Peverill.
Re-typing	J Ryan.

Many other members have given time and effort towards making TIsHUG the success it is in a number of ways. Because of the hard work of its members, TIsHUG is able to provide the following services to its members.

1. A monthly News Digest of excellent quality.
2. A shop with a range of goods tailored to meet the needs of the members.
3. A Bulletin Board which operates 24 hours a day.
4. A console exchange service.
5. A mail order service for TIsHUG members all over Australia.
6. Club projects, eg EPROM RAMdisk.
7. An extensive library of books and magazines about the TI99/4A.
8. The distribution of some of the latest overseas software.
9. The purchase of hardware and its demonstration to members.
A. Special interest groups.
B. Technical help and advice for a wide range of problems.
C. Regional Group meetings.
D. An extensive software library.
E. A printer ribbon re-inking service.
F. Monthly meetings.

Membership

TIsHUG membership has declined very little over the past year. While some members have left the group, they have been replaced by new people. Since Texas Instruments (Australia) have given over to TIsHUG the right to repair and service the computer, new members are being recruited from people who want help with their systems. Advertising in the Trading Post has also brought in a number of people and a supply of TI99/4A equipment. The club at present is able to subsidise the News Digest and provide other services for members. If we are able to increase our membership we will remain a viable organization in the foreseeable future. It is obvious that, as membership falls, a higher proportion of members must be willing to help out if we wish to maintain our present level of service.

The Year in Review

1991 was a fairly satisfying year in many ways. The measures initiated in 1990 have lead to a small decline of about 3% in actual cash reserves over the year with a much larger increase in the value of stock in the shop. Services were maintained, even improved for members. Perhaps the success story of the year has been the operation of the Club shop. Members have been able to buy a whole range of hardware and software at low prices. Support for club projects has been available, meticulously organized by Percy Harrison. Percy has tried to anticipate members' needs at the right price. Where a bargain cropped up, the members received the benefit. The shop success has enabled us to purchase some overseas software and hardware for club use. We were also able to maintain the extremely high standard of our monthly News Digest. We have built a number of RAMdisks with EPROMs as a club project, completed Multi-function cards to upgrade our systems and are now moving towards 80 column cards.

We have maintained our monthly meetings at Ryde School and I have noticed that people stay longer at the meetings than used to be the case. Attendance at meetings has remained fairly stable during the year. We usually see in excess of 50 members each meeting. One of our aims has been to have more machines available for hands on use at the meetings. This is happening. Games machines are now always available for younger members. The bulletin board is usually available for display at each meeting. Members have the opportunity at meetings to raise any issues they wish. Communication between members and directors seems open and friendly. Three areas of continuing concern however relate to the declining use of the bulletin board, the reduction of local programming input and the reduced attendance at Regional Group Meetings. Despite some areas of change, TIsHUG remains strong and viable.

The Future

I suppose some members feel that each year will be the last for the TI99/4A. Well it does not seem to happen. The TI99/4A just keeps on going and getting better. Our particular project for 1992 will be a TI Faire late in 1992. We are already getting enquiries about our proposal from far and wide. There is no doubt that we can have another really good year with the TI99/4A. A number of our members have other computers which they use as well as the TI99/4A, but it remains for many, their fun machine. We may reach a point at some stage where we need to open our ranks to other computer users, but this has not been necessary at this this stage. There is continuing development of the TI99/4A. There is continuing challenge for us. Having an expanded system is getting cheaper all the time. Technology is getting cheaper and we can take advantage of it. During 1992 TIsHUG will continue to provide service for its members and will continue to be a strong and vital organization. The directors thank you for your support over the past year, and look forward to 1992. O

BBS Competition

by Ian Mullins

A competition is proposed for all BBS Users. A colour photo of the User's TI computer set-up (with modem displayed) to be submitted with a nominal entry fee. The Directors of TISHUG (AUST) have agreed to judge the entries at the December meeting. The prize will be in the form of selected hardware or software to a set value from the SHOP.

Please watch the BBS for further information. O

TIsHUG (Australia) Limited

Fifth Annual General Meeting

Saturday 7th December, 1991

Ryde Infants' School, Tucker Street RYDE NSW

AGENDA:

1. MEETING OPENING (2.00PM)
2. MEMBERS PRESENT AND APOLOGIES
3. READING AND CONFIRMATION OF MINUTES OF THE 4TH ANNUAL GENERAL MEETING HELD ON SATURDAY 1 DECEMBER, 1990
4. CORRESPONDENCE AND DEALING WITH SAME
5. READING AND DEALING WITH RECOMMENDATIONS FROM THE BOARD OF DIRECTORS - LIFE MEMBERS AND OTHER MATTERS
6. DIRECTORS REPORTS, PRESENTATION OF ACCOUNTS AND AUDITORS REPORT

continued on page 3

Secretary's Notebook

by Terry Phillips

A quick count of those present at the October meeting revealed that about 50 had turned out looking for bargains or trying to sell surplus equipment. There was not a great deal on offer and sales throughout the day were fairly slow. Perhaps this is another sign of the hard financial times the country is experiencing at the moment. The club purchased two complete systems and these will be available through the shop. So if you are thinking of expanding your system - and what better reason is there with Christmas fast approaching - check out what Percy has available.

Planning is well advanced for the November meeting which as you know will be a full day tutorial event. Russell has come up with some interesting sessions which will give a great variety throughout the day. Come along early and bring the family. There is plenty of room in the school grounds for the young ones to play in plus there will be several computers set up for them to have a go at their favourite games. Pack a picnic lunch or take advantage of the many and varied food outlets close by to the school.

Some members may not have attended meetings in the past because they may consider that Ryde is an out of the way area which is difficult to get to. This is not the case particularly if coming by public transport. All you need to do is catch a suburban train to Strathfield station, then take the bus, route 458, from the northern side. Suitable buses, for normal meeting start times, depart at 12.20pm, 12.50pm and 1.20pm with the journey time being approximately 20 minutes. On the return journey buses depart Ryde at 30 minute intervals, 3.28pm, 3.58pm etc. I believe that the bus fare is $2.50 for adults. If you wish to catch this bus to the full day tutorial November meeting then suitable buses depart at 9.52am, 10.27am then half hourly.

There is only one new member to welcome this month and he is David Peters of Wyoming on the Central Coast. I understand you were at the October meeting Peter. Sorry I did not get an chance to have a chat with you.

Last month I welcomed back Kerry Harrison to the club. That was my mistake. Sorry about that Kerry, it was Hilton Harrison I meant to welcome back.

At the last Directors meeting all gave a commitment to the idea of a TI Faire in 1992. Now we will be looking to a bunch of hard working volunteers to put their names forward so that the event can in fact become a reality rather than something that fades through a lack of support. If you would like to help in some way then let one of the Directors know.

As you know, December is Annual General Meeting time and in this issue is a suitable nomination form for members wishing to be candidates for election to the board of directors. Make sure you give this matter some thought. See you at the next meeting. o

continued from page 5

TC-750	Elementary Letters & Numbers with Speech
TC-751	Children's Programs with Speech
TC-752	Alphabet for Preschoolers
TC-753	Children's Programs with Speech, Disk #2
TC-755	Colors, Shapes, Directions
TC-760	Spelling Programs
TC-770	Vocabulary & Reading, Disk #1
TC-780	Preschool Math
TC-790	Elementary Addition and Subtraction
TC-791	Addition & Subtraction
TC-796	Multiplication and Division, Disk #1
TC-797	Multiplication and Division, Disk #2

Commercial Software

Page Pro Banner Maker is another Page Pro utility that enhances the versatility of Page Pro 99 as a desktop publishing tool for the TI99/4A. Asgard Software considers this program to be the most professional banner maker on the TI99/4A market.

You can incorporate both Page Pro pictures as well as text, and the pictures can be placed wherever you like. You can have one or two lines of text on your banner and the letters are well formed and look very smooth close up. There is no limit to the number of pictures or words that you can use in the banner and the results are very quickly printed out. You need an Epson compatible printer and a disk based system. The price is $20.00.

Page Pro Poster Maker allows you to create full-size posters and signs from your Page Pro pages. This utility will blow up your page by a factor of two times, four times and even eight times to create 40cm by 55cm signs, 85cm by 110cm posters and gigantic pictures measuring 1.75 metres by 2.25 metres. An Epson compatible printer is required.

The program is easy to use and is ideal for students, teachers, small businesses etc. All you need to do is to create the layout in Page Pro 99, then save the result to disk. You then load the file into Banner Maker, select the poster size you want, make sure that the printer is online with enough paper, and wait for the program to print out the result in strips. The full size artwork is completed when the strips are stuck together in the correct sequence. The price of Banner Maker is $20.00. o

continued from page 2

7. UNFINISHED BUSINESS FROM LAST AGM (IF ANY)
8. ELECTION OF RETURNING OFFICER AND TWO (2) SCRUTINEERS
9. ELECTION OF DIRECTORS
10. ELECTION OF AUDITOR
11. NEW BUSINESS (IF ANY)
12. CLOSE

NOTICE TO MEMBERS:

All members are advised that the 5th Annual General Meeting of TIsHUG (Australia) Limited will be held on Saturday 7 December, 1991 at Ryde Infants School, Tucker Street, Ryde NSW, commencing at 2pm.

Members attending are requested to arrive by 1.30pm to enable them to sign in and ensure a prompt 2pm start.

The following relevant paragraphs from the club's Articles of Association are brought to your attention:

16(i) - Nominations for the office of Director shall be delivered to the Secretary by 8.00pm on the twenty-first (21) day prior to the day fixed by the Board for the annual election of Directors.

17(b) - Nominations for election of the Directors shall be made in writing and signed by two (2) members of the Club and by the nominee who shall signify his consent to the nomination.

17(d) - If the full number of candidates for the positions of Directors is not nominated as prescribed then additional nominations may be made at the meeting. If there be more than the required number nominated an election by ballot shall take place but if there be only the requisite number nominated the Chairman shall declare those nominated duly elected.

In accordance with paragraph 16(i) nominations for the office of Director shall close with the Secretary at 8.00pm on Saturday, 16 November, 1991, while in accordance with paragraph 17(b) a suitable nomination form is enclosed. Terry Phillips Honorary Secretary o

TIsHUG (Australia) Limited

Minutes of the Fourth Annual General Meeting

Held on Saturday 1st December, 1990

Ryde Infants' School, Tucker Street RYDE NSW

1. OPENING:

Chairman Dick Warburton opened the meeting at 2.20pm welcoming those members present.

2. MEMBERS PRESENT AND APOLOGIES:

Present - 38 (highlighted on attached membership list).

Apologies were recorded and accepted from:

Tom Marshall, Karl Kuit, Warren Welham, George Meldrum, Cyril Bohlsen, Steven & Jenny Carr, Stephen Rivett and David Higgins.

3. READING AND CONFIRMATION OF MINUTES OF 3RD ANNUAL GENERAL MEETING HELD ON SATURDAY 3 FEBRUARY, 1990 AND READING AND CONFIRMATION OF MINUTES OF SPECIAL GENERAL MEETING HELD ON SATURDAY 4 AUGUST, 1990.

These minutes had been included in the November issue of the TND and had been mailed to all members.

a) AGM Minutes - Moved John Paine seconded Alf Culloden that they be confirmed. Motion carried unanimously.

b) SGM Minutes - Moved Ben Von Takach seconded Derek Wilkinson that they be confirmed. Motion carried unanimously.

4. CORRESPONDENCE AND DEALING WITH SAME:

Secretary Terry Phillips advised that there was no correspondence.

5. READING AND DEALING WITH RECOMMENDATIONS FROM THE BOARD OF DIRECTORS - LIFE MEMBERS AND OTHER MATTERS.

Chairman Dick Warburton advised that the Directors had no recommendations.

6. DIRECTORS REPORTS, PRESENTATION OF ACCOUNTS AND AUDITORS REPORT:

The Directors report had been included as a supplement in the December TND while the Accounts and Auditors report had been included as a supplement in the November TND. These reports had been mailed to all members.

a) Directors Report - Moved Derek Wilkinson seconded Alf Culloden that it be accepted. Carried.

b) Accounts and Auditors Report - Moved Percy Harrison seconded Lou Amadio that they be accepted. Carried.

7. UNFINISHED BUSINESS FROM LAST AGM (IF ANY):

Secretary Terry Phillips advised that item 10(b), viz, "Previous Auditor - on the motion of Allen Holmes, seconded Lou Amadio it was agreed that the group write to our previous auditor with a request for the return of $350 in recompense for the uncompleted 1988 annual corporate affairs return. This motion carried by the majority of members." had not been acted upon after the Directors had agreed that the alleged debt was unrecoverable and could create legal hassles in attempting its recovery.

8. ELECTION OF RETURNING OFFICER AND TWO (2) SCRUTINEERS:

Chairman Dick Warburton advised the meeting that at the close of nominations for the office of Director only the requisite number of five (5) candidates had nominated. On a point of order from John Paine suggesting that there still need be an election of a Returning Officer, the Chairman quoted paragraph 16(d) of the groups Memorandum and Articles of Association, viz - "If the full number of candidates for the positions of Directors is not nominated as prescribed then additional nominations may be made at the meeting. If there be more than the required number nominated an election by ballot shall take place but if there be only the requisite number nominated the Chairman shall declare those nominated duly elected."

John Paine withdrew his point of order and the Chairman declared those nominated, Messrs Phillips, Schreiber, Trott, Warburton and Welham duly elected.

9. ELECTION OF DIRECTORS:

Not required. See item 8 above.

10. ELECTION OF AUDITOR:

F H Spender (Wollongong) was nominated by Geoff Trott, seconded John Paine. Carried.

11. NEW BUSINESS (IF ANY):

a) Bob Relyea queried advantages of club incorporation. John Paine advised the members present that very advantageous reasons were on the limiting of liability of members in the event of the group being sued. He cited an example of an overseas user group where the group had been sued for software piracy. John further stated that under the groups articles of association the group could not become unincorporated.

Other discussion on this topic ensued however on the motion of Percy Harrison, seconded Don Gould it was carried that no further discussion on the topic take place.

b) Rolf Schrieber gave a rundown on software purchases to take place during 1991. Texaments had already given the group dealer status which involved a good discount rate while it was likely that Asgard would also do the same. Rolf hoped that the availability of good quality software at the right price would discourage software piracy.

c) Dick Warburton proposed a vote of thanks to all who had contributed to the groups running throughout the year.

d) Dick Warburton gave an update on the EPROM project and suggested it was now working. A demonstration would follow.

12. MEETING CLOSURE:

There being no further business for discussion, Chairman Dick Warburton closed the meeting at 3.15pm advising members that they could now partake of the Christmas party food.

Minutes recorded by:

Terry Phillips
Honorary Secretary

1 December, 1990

Minutes confirmed by:

Richard Warburton
Chairman

o

TIsHUG Shop with Percy Harrison

Once again Dick Warburton's selective buying has enabled us to drop the price of both the High Density 5.25 disks and the 3.5 disks - see new prices listed below.

There still has not been any further response to my request for those members who indicated that they would buy the Eprom Ram Card and kit to honour their commitment and it now looks like the club being left with a surplus of these boards and kits. Unfortunately it will mean that for future projects club members will have to pay for PC Boards and Kits prior to the club placing orders for the items as we cannot afford to be left with hundreds of dollars worth of surplus stock. It is with much regret that we have to impose this on our members but we have no other alternative if the club is to remain viable.

PRICE LIST.

5.25 in. DSDD Disks (Boxes of ten)	$6.00
5.25 in. HD Disks (Boxes of ten)	$10.00
3.5 in. DSDD Disks (Boxes of ten)	$10.00
5.25 in. DSDD Half Height Drive (New)	$65.00
12 Volt AC Transformer	$3.50
13 Volt Arlec Transformer	$12.00
8.5, 17 Volt Transformer	$25.00
60 VA Transformer	$20.00
MFC Printed Circuit Board	$30.00
MFC Kit (Disk Controller	$102.50
32K Kit for MFC	$19.50
PIO/RS232 (single port) Kit for MFC	$42.50
Combined 32K and PIO/RS232 Kit	$53.00
Music Kit with PCB	$65.00
32K Memory PC Board	$7.00
Eprom Ram PC Board	$45.00
Eprom Ramdisk Basic Kit	$35.00
Funnelweb Eprom Set (3 Eproms)	$36.00
TI Artist Eprom Set (2 Eproms)	$24.00
32K Static Ram IC (62256)	$10.00
8K Static Ram IC (6264LP)	$5.00
74LS08 IC (quad Schottky)	$0.50
1K Resistor	$0.05
Exchange Console	$30.00
ROS Version 8.14	$12.00

NOTE: ROS 8.14 must be purchased with first Eprom Set.

COMMERCIAL SOFTWARE.

Artoons SSSD	$12.00
Character Set Graphic Design Cataloguer	$6.00
Character Set Graphic Design I	$12.00
Character Set Graphic Design II	$10.00
Character Set Graphic Design III	$14.00
Display Master	$15.00
Genial Traveler (SSSD)	$6.00
Microdex 1 (SSSD)	$16.00
Microdex II (SSSD)	$11.00
Nuts and Bolts #1 (DSSD)	$6.00
Nuts and Bolts #1 (SSSD)	$7.00
Page Pro 99 version 1.6	$28.00
Page Pro Utilities	$17.00
Page Pro Applications #1	$2.00
Page Pro Line Fonts	$9.00
Page Pro Medical Clipart	$9.00
Picasso Publisher Version 2.0	$14.00
Picasso Publisher Support Disks	$6.00
Picasso Applications Disk	$2.00
Rockrunner (SSSD)	$15.00
Spell It! (DSDD version)	$24.00
Spell It! (SSSD version)	$27.00
The Missing Link (TML)	$28.00
The Missing Link Companion Disk	$2.00
TI Artist Plus	$25.00
TI Base Vers 3.01 (SSSD)	$25.00
TI Sort SSSD	$15.00
Tris Module	$25.00
Typewriter Module	$25.00

TIsHUG Software Column by Rolf Schreiber

November Software Releases

DISK A214 is PLUS! V1.0 a fairware word processing companion and utilities disk of templates, tutorials, articles, codes, programs for various versions of TI-Writer (particularly Funnelweb). The material was created, edited, adapted, rewritten and/or collected especially for the Fairware marketplace by Jack Sughrue. There is extensive documentation on this DSSD disk about using the various programs. PLUS! is also available in SSSD format as a two disk set.

DISK A447 is FILE READER, a utility disk by Martin Smoley, author of the TI-Base tutorial series.

DISK A456 is REMEMBRANCE, a disk of 24 marches from the era of the American Revolution by Bruce Harrison of Harrison Software. Bruce has done an excellent job at squeezing an incredible amount of music (written in assembly language) music onto this SSSD disk. Available as freeware only outside USA.

DISK A463 is TI-EXAM, a utility program which allows you to delve into the inner secrets of the TI99/4A's memory and operating system. This is definitely not a program for beginners, but experienced hackers. TI-EXAM requires the E/A module, or equivalent, to load the main program. A menu allows to to select the various options, but there is no documentation available to describe in detail what each option does.

Tigercub Software Releases

TCC-3 is the third disk in Jim Peterson's Tiger-Cub Collection. As is usually the case with all of Jim's software, this disk is just packed with many useful programs that do an amazing variety of things. I have not had the time to do a detailed assessment of all the programs on this disk, but the quick look that I did have revealed an interesting variety of useful utilities and subroutines that could very easily be incorporated into your own programs. A listing of the programs on this disk is given below.

1) Random Character Generator
2) Sort Tester
3) 4x4-Bit Graphic Utility
4) TC Graphics Designer
5) 4x4-Bit Doodler
6) Soundmaker
7) Catwriter
8) Compactor
9) Compare
10) Cross Reference Lister
11) Disklister
12) Disk Memory Check
13) Download Converter
14) Extractor
15) Go-Search
16) Line Edit
17) Single Pixel Drawing
18) Tigercub Quickloader
19) Shrink
20) Sprite Editor

I will also be releasing the Public Domain software that we recently received from Jim Peterson. To start off, I will be making available all the pre-school and primary school educational software that Jim has gone to great pains to collect and place on separate disks according to subject matter. There are twelve disks altogether, and they can be ordered separately for $2.00 each, or $20.00 for the complete set.

The subject category of each disk is listed below, together with its catalogue number.

continued on page 3

Sorting, part 1

by Ron Brubaker, USA

This is the first in a series that will introduce the user to the benefits and operation of sorting routines. We hope you find them to be useful.

INTRODUCTION

One of the oldest and most common applications of computers is sorting information into ascending or descending order. Both numeric data and character data can be readily sorted using BASIC programs.

Numerous algorithms (procedures) have been developed for sorting. These vary widely in complexity and speed. Unfortunately, some of the fastest sorting routines are the most difficult to understand. In the following sections of this chapter some of the fundamental concepts of sorting will be discussed. A simple sorting routine will be presented and its operation will be examined in detail. From this examination several ways of improving its efficiency and speed will be introduced and examined.

A ROUTINE FOR GENERATING RANDOM NUMBERS

The following program segment will be utilised to provide an array of random numbers ranging from one to one hundred. Duplicate values are quite possible since no provision was made to eliminate them.

```
10 REM **** GENERATION OF A LIST OF RANDOM NUMBERS ****
20 REM
30 DIM A(100)
40 RANDOMIZE
50 PRINT "HOW MANY NUMBERS DO YOU WANT";
60 INPUT N
70 PRINT
80 FOR I=1 TO N
90 A(I)=INT(100*RND)+1
100 PRINT A(I);
110 NEXT I
120 PRINT
```

The results of a typical run are shown below:

```
HOW MANY NUMBERS DO YOU WANT? 15

64 48 2 79 36 5 66 71 100 24 14 57 13 34
```

A Simple Sort Routine

The following program is a very simple sorting routine. It is neither elegant nor efficient but it does work.

```
130 REM
140 REM ***** SIMPLE BRUTE FORCE SORT
150 REM
160 FOR J=1 TO N-1
170 FOR I=1 TO N-1
180 IF A(I)<=A(I+1)THEN 220
190 T=A(I)
200 A(I)=A(I+1)
210 A(I+1)=T
220 NEXT I
230 NEXT J
240 REM
```

Let us examine this program in detail. Recall that N was defined in the first segment as the number of values to be sorted. Line 180 of the sorting routine compares a given value with the value that follows it in the array. Thus, the inner loop will start by comparing the first value with the second and should end by comparing the next to last value (N-1) with the last value (N). Line 180 is written for an ascending sort. If the value A(I) is less than or equal to A(I+1) then the program goes immediately to the next value of the loop index I. If, however, the value of A(I) is greater than that of A(I+1) then the value of A(I) is placed in a temporary location (T) and the value of A(I+1) is placed in A(I). The exchange is completed by placing the original value of A(I), which was sorted in T, into A(I+1). Thus, lines 190 through 210 effectively swap the values stored in two adjacent array locations.

Although a single pass through the list of numbers will make some improvements it will take more than once through to complete the task. The obvious question is how many passes are necessary. The answer may be obtained from the following reasoning. If a number is at the wrong end of the list how many times must it be swapped to move it to where it belongs? For a list 15 items long it would take 15 - 1 = 14 swaps to move it to the opposite end. Thus, the outer loop has also been set to run n-1 times.

Starting with the set of numbers obtained above:

64 48 2 79 36 5 66 71 100 24 14 67 57 13 34

The following shows the arrangement of the numbers after the indicated number of passes through the array (i.e. the number on the left corresponds to the value of J).

```
1) 48 2 64 36 5 66 71 79 24 14 67 57 13 34 100
2) 2 48 36 5 64 66 71 24 14 67 57 13 34 79 100
3) 2 36 5 48 64 66 24 14 67 57 13 34 71 79 100
4) 2 5 36 48 64 24 14 66 57 13 34 67 71 79 100
5) 2 5 36 48 24 14 64 57 13 34 66 67 71 79 100
6) 2 5 36 24 14 48 57 13 34 64 66 67 71 79 100
7) 2 5 24 14 36 48 13 34 57 64 66 67 71 79 100
8) 2 5 14 24 36 13 34 48 57 64 66 67 71 79 100
9) 2 5 14 24 13 34 36 48 57 64 66 67 71 79 100
10) 2 5 14 13 24 34 36 48 57 64 66 67 71 79 100
11) 2 5 13 14 24 34 36 48 57 64 66 67 71 79 100
12) 2 5 13 14 24 34 36 48 57 64 66 67 71 79 100
13) 2 5 13 14 24 34 36 48 57 64 66 67 71 79 100
14) 2 5 13 14 24 34 36 48 57 64 66 67 71 79 100
15) 2 5 13 14 24 34 36 48 57 64 66 67 71 79 100
```

Note that on the first pass the largest number has been moved all the way to the end of the list. On each suceeding pass the list becomes increasingly organised with the largest remaining number moving immediately to the end. The smaller numbers move toward the beginning of the list in a more gradual fashion. Note also that the list has been completely sorted in less than N-1 passes.

The observations suggest several improvements that can be made to make the routine more efficient and increase its speed. This will be the topic of next months article. o

Joystick Special

by Chris McCarthy

And now some good news. VIRGIN megastore in the Pitt St. Mall, Sydney has microswitch joysticks on sale for $9.95

They are good quality units having a total six microswitches inside (four for direction and two for the base mounted fire buttons). The handgrips have two additional fire buttons of the metal contact type as well as a steel shaft so they will be durable. As a bonus, they have an auto-fire capability which can be activated by way of a base mounted on/off switch.

This last feature seems to be of limited value on the TI99/4A, at least with the software I have at my disposal.

The best feature of these joysticks is that they are repairable. The microswitches are a standard type available from your local electronics supplier for about $3 each.

The joysticks will require the fitting of diodes and the use of an adaptor when used with the TI99/4A. o

TI-Bits Number 11

by Jim Swedlow, CA USA

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

QUOTE OF THE MONTH

"The computer 'doth make fools of us all."

---Weinberg

BELL COMPATIBLE?

Ever noticed that modem adds include a statement about Bell compatibility? This will give you an idea of what that means.

BELL 103A is the standard format for transmitting data by telephone at speeds of 300 baud or less.

BELL 202 is a standard format for transmitting data by telephone at 1,200 baud. Bell 202 format is half duplex only and has now largely been replaced by Bell 212A.

BELL 212A is the standard format for transmitting data by telephone at 1,200 baud.

WORD OF THE MONTH

BAUD: a unit that measures the speed of data transmission. One baud equals one bit per second. Or, 300 baud is 300 bits per second. That is why 1200 baud is four times faster than 300 baud.

MORE ACRONYMNS

Here are two more acronyms:

CD-ROM - "Compact Disk Read Only Memory" - using the same CD's that are becoming the rage for music to store data. One CD can hold an entire encyclopedia with space left over. The current price of the drives is high ($1000 or so) as are the few CD-ROM's on the market ($300 to $1000+). Prices will come down, however.

WORM - "Write Once Read Many" - the next generation for CD-ROM, these allow the user to write on CD disks once and then read many, many times. The drive has two laser devices, one to write and one to read.

TEXAMENTS

A while back I mentioned that I had some problems with a product from Texaments. I do not want to discuss the issue because I cannot document it, but I do want to say that I had excellent response from the folks at Texaments. They were more than fair.

MORE QUOTES

"PRICE is what you pay NOW;
COST is what you pay LATER."

---A Pilgrims' Pride Catalog

"You may not always get what you pay for but you never get what you do not pay for!"

---Ibid.

MAILING LABELS

As the United States Postal Service (USPS) uses more and more Optical Character Readers (OCR's), a problem is developing with computer generated mailing labels.

According to an article in a recent issue of InfoWorld, the USPS has purchased some 400 OCR's since 1984 and it recently signed a contract for over 400 more. These units can process mail at speeds up to 35,000 pieces an hour (the hand rate is about 900).

What is the mailing label problem? Seems that the OCR's have trouble reading mailing labels in compressed print or when the dots do not touch. The solution is to print your labels in Near Letter Quality or Emphasized Print and to use 10 to 12 characters per inch. If in doubt, check with your local post office.

ATTENTION OLIVETTI PR2300 OWNERS

Have you tried to read your manual and come to the conclusion that someone is unable to communicate in English? Help is available. Olivetti will send you a four page supplement that clarifies a number of things. It is available for the price of a stamped envelope. Call Customer Support (1-800-558-1303) for details.

TI LIVES!

I was looking through the current issue of MICROTIMES and spotted the listing of user groups. Most orphan computers have one or two user groups in the greater Southern California area. A few have three groups.

One computer stood out with ten (count them, ten) user groups. None other than our 4A.

Enjoy. o

continued from page 10

```
100 DISPLAY AT(3,6)ERASE ALL:"PROGRAM REGISTER":"":" Wi
11 reformat a LISTed XBasic program from any linelength
to any other length."
110 DISPLAY AT(8,1):" Each program line (not file line)
must end in a carriage return."
120 DISPLAY AT(12,1):"Input filename?":"DSK" :: ACCEPT A
T(13,4):IF$ :: DISPLAY AT(15,1):"Output filename?":"DSK"
 :: ACCEPT AT(16,4):OF$
130 DISPLAY AT(18,1):"Present line length?" :: ACCEPT AT
(18,22)SIZE(2)VALIDATE(DIGIT):A
140 DISPLAY AT(20,1):"Reformat to what length?" :: ACCEP
T AT(20,26)SIZE(2)VALIDATE(DIGIT):X :: IF X=A THEN 130
150 OPEN #1:"DSK"&IF$,INPUT:: OPEN #2:"DSK"&OF$,OUTPUT
:: IF X<A THEN 230
160 IF EOF(1)THEN 270 :: LINPUT #1:M$ :: L=LEN(M$):: IF
POS(M$,CHR$(13),1)=0 THEN 180
170 IF P+L<X+1 THEN PRINT #2:M$ :: P=0 :: GOTO 160 ELSE
PRINT #2:SEG$(M$,1,X-P)&CHR$(13):SEG$(M$,X-P+1,255):: P=
0 :: GOTO 160
180 IF L<A THEN M$=M$&RPT$("",A-L):: L=A
190 IF P=0 THEN PRINT #2:M$;
:: P=L :: GOTO 160
200 IF P+L<X THEN PRINT #2:M$;:: P=P+L :: GOTO 160
210 IF P+L=X THEN PRINT #2:M$&CHR$(13):: P=0 :: GOTO 160
220 PRINT #2:SEG$(M$,1,X-P)&CHR$(13):SEG$(M$,X-P+1,255);
:: P=LEN(SEG$(M$,X-P+1,255)):: GOTO 160
230 IF EOF(1)THEN 270 :: LINPUT #1:M$
240 L=LEN(M$):: IF L+P>X THEN PRINT #2:SEG$(M$,1,X-P)&CH
R$(13):: M$=SEG$(M$,X-P+1,255):: P=0 :: GOTO 240
250 IF M$=CHR$(13)THEN 230
260 IF POS(M$,CHR$(13),1)<>0 THEN PRINT #2:M$ :: P=0 ::
GOTO 230 ELSE PRINT #2:M$;:: P=LEN(M$):: GOTO 230
270 CLOSE #1 :: CLOSE #2
```

Another way of reformatting a program listing is to use Curtis Alan Provance's remarkable Textloader to convert the listing to program format and then listing it to disk in the desired format, using Super Extended Basic. However, Textloader can introduce some bugs, so be sure to test the program before you list it, and be prepared to fix the bugs.

Now that you have gone to all that work, are you going to print your article through the Funlweb Formatter and perhaps garble it again? Remember what I told you about the &, @, *, and the leading period! Program listings usually contain those characters. If you have not modified your FO file, and you do not replace and transliterate, you will print garbage! o

XB tips Number 12

by Jim Swedlow, CA USA

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

MORE ON PRINT SEPARATORS

If, in a PRINT USING, you follow the print list with a print separator, your 99/4A will honour it. Consider this:

```
10 FOR I=1 TO 30
20 PRINT USING "## ### ":I,I*I@;
30 NEXT I
```

Instead of one column of numbers and their squares (which you would have without the semi-colon after the I*I) you will get four right justified columns on your screen. Also works with printers and with DISPLAY USING.

TE II ABILITY

With a program in memory and your Speech Synthesizer attached, enter LIST "SPEECH" and see what happens. Strange! (Source: Regina in COMPUTE)

LINE NUMBER TABLE

What, you ask, is a line number table? Glad you asked! When your TI stores a program in memory or on disk, it is broken into two parts: the line number table and the line contents. In this program:

```
100 ! T1
110 FOR I=1 TO 10
120 PRINT I
130 NEXT I :: CALL CLEAR
```

The line number table looks like this:

HEX	DECIMAL
0082 37B5	130 14261
0078 37C2	120 14274
006E 37C6	110 14278
0064 37D3	100 14291

Note that the line numbers are in descending order. The second number is the memory location of the line contents. For example, the contents of line 100 start at >37D3. The > symbol means that the number is in hex or hexadecimal or base 16.

When you RUN a program your 4A, during pre-scan, reserves memory for variables and other things. That is why you get different responses to SIZE before and after RUNning a program.

While your computer always keeps the line numbers in descending order, it shifts the memory locations of the line contents around without regard to order as you edit/modify your program.

If you save a program to disk in MERGE format, enter NEW and then MERGE your program back into memory, your line number table should be returned to order. This may or may not have any real advantage, however.

What use is this? It depends. If you want to go looking in memory and learn how TI stores and executes a program, this is the first step. There is a program in our library (MEMORY DUMP on UTILITIES U02) that can help.

If not? Well, now you know that each line number takes four bytes of memory (two for the number and two for the memory address). A line separator (::) takes only one byte. Conclusion: multi-line statements save memory!

FILE*PRINT

Many files are stored in DISPLAY VARIABLE 80 format: TI writer files, Editor Assembler source files, text files, documentation files, etc. This month's program reads your disk for DV80 files and displays a memu on your screen.

You can then print up to 99 copies of any file with a single keystroke. Further, if your printer is GEMINI/EPSOM compatible, you can activate double strike and/or emphasized print.

If you get strange error messages when you first run this program, change the ON ERROR statements in lines 170 and 270 (twice) to ON ERROR STOP and change the !@P- in line 150 to !. After you have finished debugging, restore these commands.

Enjoy!

```
100 ! FILE*PRINT
110 ! VERSION XB.1.0
120 ! 26 MAY 85
130 ! By Jim Swedlow
140 !
150 N,@=1 :: DIM N$(24),S$(1),E$(1),D$(1),T$(1):: GOTO 1
60 :: CALL CLEAR :: CALL KEY:: A,B,D,E,H,T,A$,P$ :: !@P-
160 S$(0)="OFF" :: S$(@)="ON" :: P$=CHR$(27):: E$(0)=P$&
"F" :: E$(@)=P$&"E" :: D$(0)=P$&"H" :: D$(@)=P$&"G" :: T
$(@)=CHR$(12):: P$="PIO"
170 DISPLAY AT(@,10)ERASE ALL BEEP:"FILE*PRINT": : : :"
Reading Disk Directory" : : ON ERROR 330 :: H=0 :: OPE
N #@:"DSK1.",INPUT,RELATIVE,INTERNAL :: INPUT #@:A$,A,B
,B
180 INPUT #@:A$,A,B,B :: IF A$<>"" THEN IF ABS(A)<>2 OR
B<>80 THEN 180 ELSE H=H+@:: IF H<25 THEN N$(H)=A$ :: GO
TO 180
190 CLOSE #@ :: ON ERROR STOP :: IF H>24 THEN DISPLAY AT
(3,10):"WARNING": :"Only the first 24 files were": :"rea
d, there are more." :: GOSUB 310
200 DISPLAY AT(5,@):: GOSUB
300 :: B=14 :: FOR A=@ TO H :: B=15-B :: DISPLAY AT(INT(
(A+@)/2)+6,B):CHR$(64+A);" ";N$(A):: NEXT A
210 DISPLAY AT(20,@):"Press < > to change:": :"<1> Empha
sized <2> # Copies<3> TOF at End <4> Dbl strk<5> Printer
  <6> Stop"
220 GOSUB 300
230 CALL KEY(3,B,A):: IF A<@THEN 230 ELSE IF B>64 AND B
<65+H THEN 270 ELSE IF B=54 THEN CALL CLEAR :: STOP
240 IF B=49 THEN E=@-E :: GOTO 220 ELSE IF B=52 THEN D=@
-D :: GOTO 220 ELSE IF B=51 THEN T=@-T :: GOTO 220
250 IF B=50 THEN ACCEPT AT(3,26)SIZE(-2)VALIDATE(DIGIT)B
EEP:N :: N=MAX(N,@):: T=T-(N>@):: T=MIN(T,@):: GOTO 220
260 IF B=53 THEN ACCEPT AT(5,9)SIZE(-20):P$ :: IF P$=""
THEN 260 ELSE 210 ELSE 230
270 DISPLAY AT(20,@):"Printing ";N$(B-64): : : : : :: ON
ERROR 340 :: OPEN #2:P$ :: PRINT #2:E$(E);D$(D):: ON ER
ROR 330 :: OPEN #@:"DSK1."&N$(B-64):: FOR A=@ TO N :: RE
STORE #@
280 IF EOF(@)=0 THEN LINPUT#@:A$ :: PRINT #2:A$ :: GOTO
 280
290 PRINT #2:T$(T):: NEXT A :: PRINT #2:E$(0);D$(0):: CL
OSE #2 :: CLOSE #@ :: GOTO 210
300 DISPLAY AT(3,@):"Emphasized ";S$(E);TAB(17);"# Copie
s";N:"TOF at end ";S$(T);TAB(17);"Dbl Strk ";S$(D):"Prin
ter ";P$ :: RETURN
310 DISPLAY AT(20,@)BEEP:"Press any key to continue"
320 CALL KEY(3,A,B):: IF B<@THEN 320 ELSE RETURN
330 DISPLAY AT(3,@)BEEP: : :" Disk and/or file were not
": :" found in drive 1 ": :"Insert disk in driv
e 1" :: ON ERROR 350 :: CLOSE #@ :: ON ERROR 350 :: CLOS
E #2 :: GOSUB 310 :: GOTO 170
340 ON ERROR 350 :: CLOSE #2 :: DISPLAY AT(20,@)BEEP:"Pr
inter name is invalid" :: B=53 :: GOTO 260
350 RETURN NEXT
```

o

Treasurer's Report

by Geoff Trott

Income for September	$1404.10
Payments in September	$968.21
Excess of income over expenses for September	$435.89

Reformatting

by Jim Peterson, Tigercub Software, USA

Here is a handy article for those of you who do a lot of word processing. ED.

With the establishment of the Clearinghouse BBS, newsletter editors will have available more articles on disk, rather than having to xerox them or retype them from other newsletters. This will make it easier for them to reformat articles to their own requirements - but they will have to know how to do so.

I am by no means an expert on this subject, but I will offer a few ideas. It seems to me that the most practical column widths are 28, 40, 60 and 80.

I have always believed that Basic and XBasic program listings should be published in 28-column width, exactly as they appear on the screen. This makes it much easier to key them in accurately, especially when the listing contains strings of blank spaces or long strings of hex codes. For that reason, years ago I wrote a program to reformat listed programs accurately into 28- column width. Nowadays, the Super Extended Basic module will do that much more easily. Since my Tips From The Tigercub newsletters consisted mostly of program listings, I always published them in four columns of 28-character width. However, that is too narrow for primarily text articles, requiring too much hyphenation or creating too many gaps.

The 40-column width is perhaps best of all, because it can be printed in two columns of elite font or three columns of condensed font. The 60-column width can be printed in two columns of condensed font.

The 80-column width is suitable for regular D/V80 files in pica font, but this is somewhat wasteful of space; pica is considerably larger than most printed material. It is possible to write routines to reformat and print D/V80 text in even longer lines, up to 160 characters long in Epson condensed elite, but lines of more than 80 characters are difficult to scan.

Page Pro printing may require other widths, but I know nothing about that; I really do not consider the oversized crowded characters of Page Pro to be suitable for newsletters.

The first step in reformatting should be to separate any program listing portions from the rest of the text, and reformat them separately if at all. For instance, if you load the article THIS 'N THAT into Funlweb Editor and find that lines 200 to 300 of a 400-line article are a program listing, do this - FCTN 9, SF, 200 300 DSK1.PROGRAM and then FCTN 9, LF, 1 199 DSK1.THISNTHAT and then FCTN 9, LF, 199 301 E DSK1.THISNTHAT and FCTN 9, SF, DSK1.TEXT.

The next step is to make sure that the title and the paragraphs, any line that nothing should be added onto, ends in a carriage return. The carriage return (CR, ASCII 13) in the Funlweb Editor looks like a little square C above and to the left of a little upside-down L.

If the CR's are missing and you have a text file with indented paragraphs and centered headers (i.e., blank spaces before the title, etc.), this tinygram program will add the CR's.

```
100 DISPLAY AT(3,4)ERASE ALL:"CARRIAGE RETURN ADDER":"":
" This tinygram program will add carriage returns to any
text file which has centered"
110 DISPLAY AT(8,1):"headers and indented paragraphs."
120 DISPLAY AT(12,1):"Input filename?":"DSK" :: ACCEPT A
T(13,4):IF$
130 DISPLAY AT(15,1):"Output filename?":"DSK" :: ACCEPT
AT(16,4):OF$
140 DISPLAY AT(18,1):"Put blank lines between paragraph
s? Y/N" :: ACCEPT AT(19,17)SIZE(1)VALIDATE("YNyn"):Q$
150 OPEN #1:"DSK"&IF$,INPUT :: OPEN #2:"DSK"&OF$,OUTPUT
:: C$=CHR$(13)
160 IF EOF(1)THEN 190 :: LINPUT #1:M$ :: IF Q$="Y" OR Q$
="y" THEN 180
170 IF M$="" THEN PRINT #2:C$:M$;:: GOTO 160 ELSE IF ASC
(M$)<33 THEN PRINT #2:C$:M$; ::GOTO 160 ELSE PRINT #2:""
:M$;:: GOTO 160
180 IF M$="" OR M$=" " THEN PRINT #2:C$ :: GOTO 160 ELSE
IF ASC(M$)<33 THEN PRINT #2:C$:C$:M$;:: GOTO 160 ELSE P
RINT #2:"":M$;:: GOTO 160
190 PRINT #2:CHR$(13):: CLOSE #1 :: CLOSE #2
```

Another way to add CR's is to type CTRL U to get the underline cursor, then go through the text with FCTN 4 and FCTN 6 and the arrow keys, typing SHIFT M wherever you need a CR.

Now, here is a tip. FCTN 9, S, 1 to get to the beginning of the file, FCTN 9, RS, /CTRL-U M CTRL-U/CTRL-U M CTRL-U FCTN-W/ (slash,carriage return, slash,~,carriage return,slash) to replace each carriage return with a tilde followed by a carriage return. Why? We will get to that! Hit Enter, then A for All.

In order to use the Funlweb Formatter to reformat a file, it is necessary to print it back to disk - and when you do that, the Formatter can play some very nasty tricks! Any & symbol in the text will simply disappear! An @ in the text will disappear but cause the word following it to be repeated again and again on following lines. An * followed by two or more digits will disappear, along with the first two digits. A caret sign (shift 6) will disappear. And a period at the beginning of a line will cause the entire line to disappear!

So, to be on the safe side, get back to the top of your text, go into RS again with /&/\/ (the \ is FCTN Z). If you jump to the end of the file, there were no ampersands, so you will not have to restore them after reformatting. You might do the same thing with the @, just in case. The * bug is very unlikely to occur in a text file, and the is also unlikely (except in this article!), but you might scan down through the first character of each line to be sure that a decimal number in the text has not placed a period there. If so, the best solution is to insert a 0 before the period.

There is a better way to prevent the Formatter from garbling your text. I have never understood why Texas Instruments used characters that might appear in text as control characters, when other useless characters were available - and I have often wondered why the McGoverns did not do something about it. But you can, if you have John Birdwell's DSKU. Copy your Funlweb to a new disk, just in case. Boot up DSKU and select 1. File utilities, then 5. Find string. The filename is FO. Select H for hex. At the prompt for a string, type 2A23214026 and for replacement string type 7C2321605C. Select R for replace, then FCTN W, hit Enter twice to accept defaults, and it is done. From now on, if you want to underline a word, use FCTN Z instead of &, if you want to double-strike a word use FCTN C instead of @, and if you want to input a value from a data file use FCTN A instead of * . You will still have to watch out for those periods, and the caret sign - I do not know the fix for those.

Now, to get Funlweb to reformat the text, open a new line 1 with FCTN 8 and type in .LM 0;RM 79;IN 0;FI. The 79 would give you 80-column text; you can substitute any number, 1 less than the actual column width you want (because the computer is counting from a left margin of 0, not 1). If you wanted a pre-set left margin you could change that 0, and adjust the RM figure accordingly. If the text does not have paragraph indentations, and you want to add them, change the 0 after IN to whatever number of spaces you want to indent; you can also increase the amount of indentation that way. And, if you want Funlweb to insert additional blank spaces in the text in order to justify the right margin (line it up evenly), add ;AD after the FI . Now save the file to disk, then go to the Funlweb Formatter.

Accept that same default filename; instead of the printer option, enter DSK1. and a different filename, accept all the defaults, and the file will be printed back to the disk under that new filename.

Return to the Editor and load that new filename. You will find that your text has been reformatted to the desired width, but every line now ends in a line feed (a little L to the upper left of a little F); your carriage returns have also been converted to line feeds. And there are now three lines at the top containing nothing but a blank followed by a line feed, groups of similar lines throughout the text, probably a long series of such lines at the end, and sometimes a few lines in the text containing a few dashes followed by a line feed. Some of them also contain a form feed! You will have to go through the text with FCTN 4, FCTN 6 and the arrow keys, deleting those lines with FCTN 3.

Now, to get rid of those line feeds - FCTN 9, PF, C DSK1.(and a new filename). It is best to always use a new filename for each step in the process so that if you make a mistake - as you will occasionally! - you do not have to go all the way back to start over.

Load that new file, and you will find that printing to disk with the C option has apparently stripped out all the line feeds. Actually, it changed them to blank ASCII 32's, which could cause problem in multiple-column printing or concantenation. If you want to get rid of them, SF the file to disk, LF it back and PF it to get rid of the tab line.

You might find it easier to just use this handy-dandy routine to delete the line feed lines and line feeds -

```
100 DISPLAY AT(12,1)ERASE ALL:"Input filename?":"DSK" ::
ACCEPT AT(13,4):IF$ :: OPEN #1:"DSK"&IF$
110 DISPLAY AT(15,1):"Output filename?":"DSK" :: ACCEPT
AT(16,4):OF$ :: OPEN #2:"DSK"&OF$
120 A$=CHR$(32)&CHR$(10):: B$=CHR$(12)&CHR$(10)
130 IF EOF(1)THEN 150 :: LINPUT #1:M$
140 IF M$=A$ OR M$=B$ THEN 130 ELSE IF SEG$(M$,LEN(M$),1
)=CHR$(10)THEN PRINT #2:SEG$(M$,1,LEN(M$)-1):: GOTO 130
ELSE PRINT #2:M$ :: GOTO 130
150 CLOSE #1 :: CLOSE #2
```

But, now the carriage returns you worked so hard to put in have also disappeared! Not to worry. FCTN 9, S, 1, FCTN 9, RS, /FCTN-W/CTRL-U M CTRL-U/, Enter, A for All, and those tildes will magically turn into carriage returns! Occasionally one will appear at the beginning of a blank line instead of at the end of the preceding line, in which case you will have to delete it and put it where it belongs with CTRL-U SHIFT-M. Why did I use FCTN-W, the tilde? Just because I have never seen it used for anything else in a text file (oops! except in this article!) - I could have used FCTN A, Z, C, etc. In fact, if you want to preserve any CTRL-U type printer codes in the text, you could RS them back and forth in the same way. After CTRL-U to get the underline cursor, FCTN-R is the ASCII 27 escape code, SHIFT-2 is 0 and SHIFT-A is 1, etc.

Rather than go through all that, maybe you would rather just throw the Funlweb Formatter away and use my little handy dandy Text Reformatter in good old primitive Extended Basic. It is slow, but it avoids all those extra steps and all those pitfalls. The text must have carriage returns.

```
100 CALL CLEAR :: CALL SCREEN(5):: FOR SET=0 TO 12 :: CA
LL COLOR(SET,2,16):: NEXT SET :: CR$=CHR$(13)
110 DISPLAY AT(2,7):"TEXT REFORMATTER":"":" by Jim
 Peterson"
120 DISPLAY AT(6,1):"Input filename?":"DSK" :: ACCEPT AT
(7,4)BEEP:IF$ :: OPEN #1:"DSK"&IF$,INPUT
130 DISPLAY AT(8,1):"Output filename?":"DSK" :: ACCEPT A
T(9,4)BEEP:OF$ :: OPEN #2:"DSK"&OF$,OUTPUT
140 DISPLAY AT(11,1):"Present line length?" :: ACCEPT AT
(11,22)SIZE(2)VALIDATE(DIGIT):LL
150 DISPLAY AT(13,1):"Reformat to what length?" :: ACCEP
T AT(13,26)SIZE(2)VALIDATE(DIGIT):R
160 IF R=LL THEN 140 ELSE CALL CLEAR
170 IF EOF(1)THEN 290 :: LINPUT #1:M$ :: M$=P$&M$ :: P$=
"" :: IF R>LL THEN 230
180 L=LEN(M$)+(POS(M$,CR$,1)<>0):: IF L<=R AND POS(M$,CR
$,1)<>0 THEN PRINT #2:M$ :: GOTO 170 ELSE IF L<R THEN P$
=M$&" " :: GOTO 170
190 C$=SEG$(M$,1,R):: CALL LASTPOS(C$," ",P)
200 IF P<>0 THEN 210 ELSE PRINT #2:C$ :: M$=SEG$(M$,R+1,
255):: GOTO 180
210 GOSUB 300 :: GOTO 180
220 GOSUB 310 :: GOTO 180
230 IF POS(M$,CR$,1)<>0 AND LEN(M$)<=R+1 THEN PRINT #2:M
$ :: GOTO 170
240 IF LEN(M$)<R THEN P$=M$&" " :: GOTO 170
250 C$=SEG$(M$,1,R):: CALL LASTPOS(C$," ",P):: IF P=0 TH
EN PRINT #2:C$ :: M$=SEG$(M$,R+1,255):: GOTO 230
260 IF P=R THEN PRINT #2:SEG$(M$,1,P-1):: M$=SEG$(M$,R+1
,255):: GOTO 230
270 GOSUB 300 :: GOTO 230
280 GOSUB 310 :: GOTO 230
290 PRINT #2:P$ :: CLOSE #1 :: CLOSE #2 :: STOP
300 IF SEG$(M$,R+1,1)=" " THEN PRINT #2:SEG$(M$,1,R):: M
$=SEG$(M$,R+2,155):: RETURN
310 PRINT #2:SEG$(M$,1,P-1):: M$=SEG$(M$,P+1,255):: RETU
RN
320 SUB LASTPOS(A$,B$,Y):: X,Y=0
330 X=POS(A$,B$,X+1):: IF X>0 THEN Y=X :: GOTO 330
340 SUBEND
```

I have also written a Formatter+ program which will even reformat text which does not have carriage returns, if header lines and paragraphs are indented. It will also optionally allow you to hyphenate any word which breaks after the second character, and optionally right-justifies the text. It is too long to list here, but will be available on the Clearinghouse BBS and in my TI-PD catalog.

The next-to-last thing to do when reformatting (but, from my reading of many newsletters, often omitted!) is FCTN 9, S, 1, FCTN 9, RS, /-// (replace a hyphen followed by a blank with a null, to fix hyphenated words that have ended up in the middle of a line). DO NOT, repeat DO NOT do this until you have restored the carriage returns!! Enter, but this time to do not use A for All; use Y(es) and N(o) to go through the text, deleting unwanted hyphens but leaving those at ends of lines or elsewhere if they belong.

And, last of all, PLEASE, PF rather than SF, to print the file back to disk rather than saving it back, to get rid of that pestiferous tab line!

Now, as to reformatting program listings - preferably, do not! A 28-column listing combined with 40-column text is not going to waste much space. Remember that program listings are printed so that people can key them in and run them, and the least mistake in reformatting them will usually result in garbage.

Assembly source code and C99 source code (and probably most any other language) MUST NOT be reformatted! However, anything beyond the 25th character of assembly source code is just a comment, usually preceded by an asterisk, and so is anything after an asterisk in the first column. These do not affect the program. If a comment exceeds your desired line length, it is safe to open a new line below it and retype the comment there, preceded by an asterisk.

Basic and Extended Basic programs can be reformatted, but the method described above is not reliable. If you must reformat them, I think that the following program will do a foolproof job. Again, first you must put carriage returns at the end of each program line, and the only practical way is to get the CTRL-U underline cursor and go through inserting them with SHIFT-M. For those of you who are not programmers, I had better emphasize that I am talking about numbered program lines, not the numbered lines of text that appear in the Editor.

continued on page 7

Printing Graphics with a Tandy DMP-105

by Geoff Trott

I had a call from David Peters, a new member from the central coast, about using a printer he had with Page Pro. He is a Minister in a church and was given a TI99/4A with this printer by one of his parishioners and was wanting to use the system for preparing documents for the church. I said I would have a look at it if he sent me a copy of the printer manual, which he subsequently did. Looking at the manual, I could see what the problem is and how to allow the printer to print out the output from Page Pro and I thought that the problem and a possible solution would make a good article in printing problems and writing programs.

Firstly, I would like to discuss how dot matrix printers work, in general and then look at the particular printer and its differences with the normal printer. Dot matrix printers have a number of pins which can be "fired" electrically to make a series of dots on the page by pressing the ribbon onto the paper. The number of pins can be 7, 8, 9 or 24, depending on the type of printer. These pins are usually arranged in a vertical column in the printer head, which travels horizontally to make the printed output across the page. When the printer works normally, that is prints letters, numbers and punctuation, it only needs to be sent the ASCII code of the character to be printed and some intelligence inside the printer tells the pins to fire and the print head to move to make the character shape on the page. For example, if we assume an 8 pin print head and that each character is defined on an 8x8 grid, then each character requires a pattern which will cause the head to move horizontally 8 increments while the correct pins fire at each position. The shape of the characters (the font) is fixed by the printer itself. Of course some printers allow for a few different fonts and for different sizes as well, but the information sent to the printer simply selects the character and the printer determines what its shape will be.

If you wish to print graphics or different fonts or different sizes then the printer must be instructed as to how it should print each dot on the whole page. This means that as the print head moves across the page, the pins must be told when to fire at each column position. This means that 8 pieces of data must be sent for each character position in the previous example, rather than just one piece of data (the ASCII code of the character) in the previous case. For the Epson printer I use, single density graphics prints at 60 dots per inch both across the page and down the page. This means that for an 8 inch wide page, 480 dots are required and for 11 inch long page, 660 dots are needed down the page. For an 8 pin print head, 8 dots are printed down the page for each horizontal scan of the print head across the page so that 82.5 horizontal scans are needed to cover the whole page. For David's printer (DMP-105), 7 pins are used for graphics printing with a vertical spacing of the pins of 1/72 inch, which means that 113.1 (11x72/7) horizontal scans are needed to cover the whole page. His printer has the same single density spacing of 60 dots per inch or 480 dots across the page, but it does have two other densities possible, one of which is at 1/72 inch and the other at 1/100 inch.

A graphics program like Page Pro outputs the whole page to the printer in graphics mode. To do this, it first sends out a command sequence to the printer to initialise it and set its vertical line spacing to be equal to the distance between the 8 pins in the print head. This ensures that consecutive lines have no space between them in the vertical direction. The commands used are "<esc>@" to reset the printer and "<esc>A<8>" to set the line spacing. I am using the <> to enclose ASCII codes which do not print, either as a number (8) or as a name (esc) which is well known and stands for a number (27 for esc). I will use decimal numbers as these are what are used in BASIC. Following the initialising codes are 99 groups of data which define the bit patterns for the 99 horizontal scans used to define the whole page. Each of these 99 groups of data start with a command sequence to set the density and the number of bytes of data for that line. In the case of single density, the command sequence is "<esc>K<224><1>". This selects single density and 480 (1x256+224) bytes of data to follow. Then follow the 480 bytes of data broken up into 10 strings holding 48 characters each. These data are followed by a carriage return and line feed to move to the start of the next horizontal scan using "<13><10>". The next group of data follows the same format, with only the data bytes changing if the picture changes.

This format gives some clues as to the expectations of the printer to be used. The 480 data points across the page agree with a horizontal dot spacing of 1/60 (8/480) dots per inch. The 99 horizontal scans would suggest that the expected vertical spacing is 72 dots per inch (99x8/11). This suggests that the Epson printer I am using would produce a somewhat lengthened plot compared to the expected printer and the page length on this printer would be 13.2 inches (99x8/60). David's printer, however, would be just right in this regard. There are other problems to contend with before he will get a picture onto his printer. The main problem is to change the data so that the 8 bits which are sent out for the 8 pin printers are changed into 7 bits for his printer without losing or adding any bits. The simple problem is to set his printer up for graphics mode with the correct line spacing.

The DMP-105 has no initialise function, so the first thing that is needed is to set the printer to the required printing density. Single density would be selected using "<esc><19>". There is no need to select the line spacing as this is done automatically when graphics mode is selected. With this printer, all the graphics data has the most significant bit set (codes between 128 and 255) which means that there is no need to exit from graphics mode to do the carriage return and line feed. Thus there is no count required for the graphics mode data and graphics mode is entered with "<18>". The data are then sent just as previously in 10 strings of 48 bytes each (all codes between 128 and 255) followed by "<13><10>". There will need to be 113.1 (99x8/7) horizontal scans to cover the 11 inch page. Finally, the command to exit graphics mode should be sent, which is "<30>".

So with a few minor changes to the command sequence characters the printer can be made to do the job. The difficult part of the operation is to get the data in the right format. To do this, the output of Page Pro (or any other graphics program) should be sent to a file, rather than direct to the printer. This is usually done very easily by simply typing in a file name in place of the printer name (one of the strengths of the TI99/4A operating system). This file would then be the input to a program to reformat the data to make it suitable for the printer. The program could either send the data directly to the printer or to another file to be printed later.

The translation of the data can be understood as follows. Consider the data which describes the top 56 dots high by 480 dots wide of the page to be printed. This takes up 7 horizontal scans for an 8 bit print head and 8 horizontal scans for a 7 pin head. The first scan of the 7 pin head uses the top 7 bits of the 8 pin head. The second scan of the 7 pin head uses the last bit of the first scan of the 8 pin head and the top 6 bits of the second scan of the 8 pin head. The third scan of the 7 pin head uses the bottom 2 bits of the second scan of the 8 pin head and the top 5 bits of the third scan of the 8 pin head. Do you get the idea? The seventh scan of the 7 pin head uses the bottom 6 bits of the sixth scan of the 8 pin head and the top bit of the seventh scan of the 8 pin head. The eighth scan of the 7 pin head uses the bottom 7 bits of the seventh scan of the 8 pin head. I will leave it to you to draw a little picture to make it clearer. This pattern repeats every 8 scans of the 7 bit head.

This massaging of the bits is composed of basically shifting operations which are best done in assembler. The rest of the program would be easiest done in BASIC.

Unfortunately, c99 would be reasonably difficult to use, as a lot of the characters used in graphics are special in c99 string handling and so would require extra care and even difficult coding to avoid problems. Forth could be used but it uses a different file structure which would be a pain (for me anyway). Initially I propose to check that the ideas and algorithm work by writing it all in BASIC and then migrate parts to assembler to provide a bit more speed. How do we shift bits in BASIC? Shifting is done by multiplying and dividing. If you think of decimal numbers, if you multiply a decimal number by 10 (the radix), you shift all the digits one place to the left (and bring a zero in at the right). If you divide a decimal number by 10, you shift all the digits of the number one place to the right. The bit pattern in a byte can be thought of as a base 2 number. If we multiply it by 2 we move the digits (bits) one place to the left. If we divide it by 2, we move the bits one place to the right. That is all there is to it.

The listing of the Extended BASIC version of the program follows.

```
100 ! SAVE WDS1.GTRAN
110 ! A program to convert the output from Page Pro to a
    form suitable for a DMP-105 printer
120 DIM A$(11),B$(11),C$(11),E$(11)
140 FI$="WDS1.PPPIC" :: FO$="WDS1.PPPOUT"
150 CALL CLEAR :: DISPLAY AT(1,2):"Graphics data
    converter" :: DISPLAY AT(3,2):" Input file name" ::
    DISPLAY AT(4,1):FI$
160 ACCEPT AT(4,1)SIZE(-28)VALIDATE(UALPHA,"."):IN$
170 DISPLAY AT(6,2):"Output device name" ::
    DISPLAY AT(7,1):FO$ ::
    ACCEPT AT(7,1)SIZE(-28)VALIDATE(UALPHA,DIGIT,"."):
    OU$
180 OPEN #1:IN$,DISPLAY ,INPUT ,VARIABLE 254 ::
    OPEN #2:OU$,DISPLAY ,OUTPUT,VARIABLE 80
190 LINPUT #1:D$
200 LINPUT #1:D$ :: A=ASC(SEG$(D$,4,1))::
    B=ASC(SEG$(D$,3,1)):: LL=256*A+B :: NS=LL/48 ::
    PRINT #2:CHR$(27);CHR$(19);CHR$(18)
210 FOR I=1 TO 99/7 :: FOR J=1 TO NS ::
    LINPUT #1:A$(J):: B$(J)="" :: C$(J)=""
215 ! process first line. Shift right 1 bit, save one
    bit for next line
220 FOR K=1 TO 48 :: X=ASC(SEG$(A$(J),K,1))::
    Y=INT(X/2):: B$(J)=B$(J)&CHR$(128+Y)::
    C$(J)=C$(J)&CHR$(128+64*(X-2*Y)):: NEXT K
230 PRINT #2:B$(J)
240 NEXT J
250 LINPUT #1:D$ :: PRINT #2:D$ :: LINPUT #1:D$
255 ! process second line. Shift right 2 bits, save two
    bits for next line
260 FOR J=1 TO NS :: LINPUT #1:A$(J):: B$(J)="" ::
    E$(J)=""
270 FOR K=1 TO 48 :: X=ASC(SEG$(A$(J),K,1))::
    Y=INT(X/4)::
    B$(J)=B$(J)&CHR$(ASC(SEG$(C$(J),K,1))+Y)::
    E$(J)=E$(J)&CHR$(128+32*(X-4*Y)):: NEXT K
280 PRINT #2:B$(J):: NEXT J
290 LINPUT #1:D$ :: PRINT #2:D$ :: LINPUT #1:D$
295 ! process third line. Shift right 3 bits, save
    three bits for next line
300 FOR J=1 TO NS :: LINPUT #1:A$(J):: B$(J)="" ::
    C$(J)=""
310 FOR K=1 TO 48 :: X=ASC(SEG$(A$(J),K,1))::
    Y=INT(X/8)::
    B$(J)=B$(J)&CHR$(ASC(SEG$(E$(J),K,1))+Y)::
    C$(J)=C$(J)&CHR$(128+16*(X-8*Y)):: NEXT K
320 PRINT #2:B$(J):: NEXT J
330 LINPUT #1:D$ :: PRINT #2:D$ :: LINPUT #1:D$
335 ! process fourth line. Shift right 4 bits, save
    four bits for next line
340 FOR J=1 TO NS :: LINPUT #1:A$(J):: B$(J)="" ::
    E$(J)=""
350 FOR K=1 TO 48 :: X=ASC(SEG$(A$(J),K,1))::
    Y=INT(X/16)::
    B$(J)=B$(J)&CHR$(ASC(SEG$(C$(J),K,1))+Y)::
    E$(J)=E$(J)&CHR$(128+8*(X-16*Y)):: NEXT K
360 PRINT #2:B$(J):: NEXT J
370 LINPUT #1:D$ :: PRINT #2:D$ :: LINPUT #1:D$
375 ! process fifth line. Shift right 5 bits, save five
    bits for next line
380 FOR J=1 TO NS :: LINPUT #1:A$(J):: B$(J)="" ::
    C$(J)=""
390 FOR K=1 TO 48 :: X=ASC(SEG$(A$(J),K,1))::
    Y=INT(X/32)::
    B$(J)=B$(J)&CHR$(ASC(SEG$(E$(J),K,1))+Y)::
    C$(J)=C$(J)&CHR$(128+4*(X-32*Y)):: NEXT K
400 PRINT #2:B$(J):: NEXT J
410 LINPUT #1:D$ :: PRINT #2:D$ :: LINPUT #1:D$
415 ! process sixth line. Shift right 6 bits, save six
    bits for next line
420 FOR J=1 TO NS :: LINPUT #1:A$(J):: B$(J)="" ::
    E$(J)=""
430 FOR K=1 TO 48 :: X=ASC(SEG$(A$(J),K,1))::
    Y=INT(X/64)::
    B$(J)=B$(J)&CHR$(ASC(SEG$(C$(J),K,1))+Y)::
    E$(J)=E$(J)&CHR$(128+2*(X-64*Y)):: NEXT K
440 PRINT #2:B$(J):: NEXT J
450 LINPUT #1:D$ :: PRINT #2:D$ :: LINPUT #1:D$
455 ! process seventh line. Shift right 7 bits, save
    seven bits for next line
460 FOR J=1 TO NS :: LINPUT #1:A$(J):: B$(J)="" ::
    C$(J)=""
470 FOR K=1 TO 48 :: X=ASC(SEG$(A$(J),K,1))::
    Y=INT(X/128)::
    B$(J)=B$(J)&CHR$(ASC(SEG$(E$(J),K,1))+Y)::
    C$(J)=C$(J)&CHR$(128+X-128*Y):: NEXT K
480 PRINT #2:B$(J):: NEXT J
485 ! output the saved 7 bits at the end
490 LINPUT #1:D$ :: PRINT #2:D$ :: FOR J=1 TO NS ::
    PRINT #2:C$(J):: NEXT J :: PRINT #2:D$ ::
    LINPUT #1:D$
500 NEXT I
510 PRINT #2:CHR$(30) :: CLOSE(1):: CLOSE(2)
```

This is a fairly rudimentary program with very little error checking and probably has a few bugs left. I do not have access to the printer so I am sending the output to a file and checking the contents of the file to see if it is working. I have also just noticed on further checks in the manual that the dot positions are reversed in the DMP-105 with the top dot having the value of 1 and the bottom dot a value of 64. This means that the program as written will not work quite right. However it was not going to be satifactory as it stands because of the long execution time. A full page generates 200 sectors of data and this will take more than 1 hour to process by the above program. All the bit manipulations need to be done in assembler routines called from the BASIC program using CALL LINK. This will be the subject of the next article. Meanwhile you may like to study the above program to see what has to be done.

Line 120 introduces 4 arrays of strings. They would need to be a bit bigger for higher than single density. Lines 140 to 170 output some messages and prompt for input file name and output device name. Defaults are present and can be changed in the program or when the program is run. You may notice that I am using my hard disk for convenience. Line 180 opens the two files (or one file and one device). Page Pro outputs a display variable 254 file but I decided to use a display variable 80 format for the output of the program. Lines 190 and 200 read in two strings using LINPUT to avoid troubles with strange character codes. Then the number of data items is determined which gives the number of strings of characters making up each scan line. Line 210 starts the main loop to read in each of the scan lines of data in groups of 7 lines. Each line is contained in NS strings. The whole line of data is read into A$(J) and then processed into B$(J) which is output and the remainder into alternately C$(J) and then E$(J). The output contains information left over from the previous line combined with data from this line. The number of bits left over keeps increasing until after doing 7 scan lines there is one complete line left which is just output on its own. After each line is read in, the carriage return and line feed is read in and output. The next read gets the graphics command string which is ignored as it is the same as the first one. This is all done in groups of similar instructions which can be easily followed. The program as coded is only a first attempt to see what was required and will be tidied up for next month. I will then do all the bit processing, including reversal of the bit patterns, in assembler subroutines.

Extended BASIC Tips

by Bob Relyea

Making A Menu
Part 1

One of the most useful ways of 'moving around' a program that has several things to choose from in execution is the menu method. Such programs are said to be 'menu driven'. This means that you are continually going to and from (or in and out of) a menu while doing things while the program is running. This is really neat as it saves you from having to boot up the program each time to want to run a particular routine. It is really easy to do and involves the following basic steps:

1. Make the menu page by using Display At statements (or Print if you want to),
2. Send the execution of each number on the menu off to a different part of the program (called a subroutine),
3. At the end of each subroutine direct the flow of the program back to the menu line for another choice.

And believe it or not, that is the long and the short of it! Of course, to explain it in more detail with a working example will take more than a few lines, but the general idea is real simple. The best way to learn how to do this type of thing is to do what you can do after studying this example and experiment by adding just a little bit at a time to see what happens. That Grade Standardising program that I put in a few months ago was made in just that way. When I finally got one part working, I would say to myself, "it would be nice if I could add this feature to my program", and went about working out how to do it. This would sometimes involve asking questions.

Let us look at each of the above three steps in more detail and then we will put it all together for a short program which is menu driven.

1. Using the Display At statement to make a menu. We are going to have a short program with three choices, so we will need three subroutines. This is just a fancy way of saying that we send the computer off to a different line number to do the routine asked for and it automatically skips all the rest. Here is an example of a menu page:

```
100 DISPLAY AT(2,1):"1. CALCULATE AREA."
110 DISPLAY AT(5,1):"2.  CALCULATE VOLUME."
120 DISPLAY AT(8,1):"3. CALCULATE SURFACE AREA."
```

If you type this in and want to see it on the screen after entering 'run' then you will have to temporarily type in line 130 with something like the following on it so it remains on the screen for a while:

```
130 FOR X = 1 TO 1000 :: NEXT X
```

This is a way of making the computer 'slave away' at counting while you examine the goodies.

Now you have to admit, that is not hard! However, there is a more compact way of going about it that, in this simple case, only requires one programming line to accomplish. Here it is:

```
100 DISPLAY AT(2,1):"1. CALCULATE
AREA,": :"2. CALCULATE VOLUME,": :"3.
CALCULATE SURFACE AREA."
```

By placing the ":" symbols the way you see in line 100 with a space between each pair, the computer automatically places the next statement two lines after the previous one. In other words, choice two is place two lines (double spacing) after choice one on the screen. As far as I know, you cannot place them any closer using this method but you can space them further by using an additional ":" and a space for each additional spacing that you want. So, is you want triple spacing then instead of : :, use : : : , and so on for greater spacings.

Following the ": :" the line will be place at the beginning of the next line, that is on column 1. This could be a bit awkward for those like me who have an eprom fitted to their system that places everything too far to the left on the screen. To get around this write your line like the following:

```
100 DISPLAY AT(2,2):"1.  CALCULATE
AREA,": :" 2.  CALCULATE VOLUME,": :
"3.  CALCULATE SURFACE AREA."
```

You may notice that position (2,2) has replaced (2,1) which shifts the first line one column to the right. To make sure that the remaining lines (that is 2. and 3.) are placed one column to the right as well, you have to put a space before the number. Note the way that I did this in new line 100. This will ensure that everything lines up properly on the screen. This procedure can be repeated for any number of columns that you want.

Step two and three of the menu-making procedure will be covered next month as step two involves more than meets the eye and will make use of the CALL KEY statement. o

Death of a CAD!

by John W Ryan, Coffs Harbour

Having learned that Thomas Alva Edison with his team of professional inventors enclosed Nickel hydroxide in a perforated steel tube (+) and the (-) plate being either Iron or Cadmium also enclosed in a similar tube, using Potassium hydrate as the electrolyte (giving 1.2 volts per cell and that the R.A.A.F. recognized its unreliability), I took on Ni-Cads with some reserve.

* Reports by National, Sanyo, Sony, Philips, G.E.C., Arlec, and Dick Smith in his excellent booklet and finally Union Carbide (EVEREADY) have made the following confusing contradictions:

IT will take up to 1000 charges. Nine (9) volt cell type (actually 7.2V), a 6v radio using 4x 1.5v totalling 6v. or 4x 1.2v. totalling 4.8v total a drop of 20%. Charging is recommended for 14 to 16 hours. Recharge up to 300 times (SonY). Do not overcharge as doing so will often shorten their life. Sanyo says- "will not be damaged by moderate overcharging". Do not secure by solder. Short circuit (up to 100 Amps.) will burn the end off a screwdriver at 1.2volts! I did this to test out the claim, and it is true. Do not mix with other types of cells or destroy with fire- they may explode. Avoid storage in temperatures below 5C or over 35C. Do not charge outside these temperatures as it will reduce the life of the batteries. Let them run right down. Recharge before they are spent.

They are designed for dependability but will run down if not used over a certain time. That was meant to be as muddled as I have been using Ni-Cad cells.

However the good news is that I have taken the instructions from Union Carbide as Gospel.

Ni-Cads are NOT harmed by prolonged charging and may be left on charge indefinitely.

Capacity loss and reduced operating time and some of the associated causes may be explained as follows:

* Long term storage on the shelf. 'Wake up' the cells with two or three charge/discharge cycles. Usually about 40% capacity will be available after the first charge,70-80% after two cycles and more than 95% after the third cycle. Batteries should be stored in a fully charged state. Top up the charge on a monthly basis.

continued on page 14

Graphing Data with Multiplan

Author unknown, USA

This article was retyped from the PUG Peripheral of May, 1991.

Let us explore Multiplan's graphing capabilities. We will start by graphing a set of numbers. Clear your screen and use the FORMAT WIDTH command to change the width of column 1 to four (4) characters. Next, format column 2 so it will graph the data instead of printing numbers. This is done by selecting the FORMAT CELLS command. In the "cells:" field type C2. Tab over to the "alignment:" field, enter L to left justify the data and tab over to the "format code:" field and enter an asterisk. The asterisk is what selects the graphing capability.

Now type the following values in column 1, rows 1 through 10:

4,9,10,7,5,8,11,15,6,18

Move the cell pointer to R1C2, press Value and then move the cell pointer to R1C1. The command line displays:

Value:RC[-1]

When you press ENTER, four asterisks, left-justified, are displayed in column 2. This is the graphic representation of the value 4 in R1C1. Copy this formula down nine rows, hit the Function 8 to recalculate and your screen should display graphs for the remaining numbers in the column as shown below:

```
        1    2
 1      4 ****
 2      9 ########
 3     10 ########
 4      7 *******
 5      5 *****
 6      8 ########
 7     11 ########
 8     15 ########
 9      6 ******
10     18 ########
```

The numbers nine or less are represented by asterisks in column 2. If a value in column 1 is larger than 9, number signs (#) are printed in column 2 to indicate there is not sufficient room to graph the data. Multiplan does this because the current column width is 10 characters. Notice in row 3 that even though column 2 is 10 characters wide, Multiplan still requires that empty space on the right of the column. Therefore, the number signs are printed instead of asterisks for the number 10.

You can change the width of column 2 in order to graph the data accurately. Using the FORMAT WIDTH command, change the width of column 2 to the maximum 32. All of your data should now be graphed completely.

```
        1    2
 1      4 ****
 2      9 *********
 3     10 **********
 4      7 *******
 5      5 *****
 6      8 ********
 7     11 ***********
 8     15 ***************
 9      6 ******
10     18 ******************
```

SCALING YOUR GRAPHS

Even with a column width of 32 characters, the numbers you want to graph may be too large to fit the column. One way to correct this is to reduce all the numbers in the same proportion. For example, change the numbers in column 1 to the following:

27,43,37,25,41,38,31,29,36,35

Even at maximum column width, several of the rows in column 2 are filled with number signs. To correct this, reduce the values in column 1 by half. To do this, move the cursor to R1C2, press Edit, and divide the reference to RC[-1] by 2. Your command line now reads:

EDIT:RC[-1]/2

Press ENTER and copy the formula down the remaining nine rows. Recalc and your screen should now look like this:

```
 1     27 **************
 2     43 **********************************
 3     37 ******************
 4     25 ************
 5     41 ********************
 6     38 ******************
 7     31 ***************
 8     29 **************
 9     36 ******************
10     35 *****************
```

The process of changing data by a proportion is called scaling. Scaling down, as in the example above, is dividing data by a value so it can be displayed in an area of limited size, the graph represents half the actual data. Scaling is very useful in graphing values that are too large to fit in the column or are so large that the resulting string of asterisks are uninformative. Of course, you can also scale up, or increase the length of the string of the asterisks. To scale up you multiply the data by a constant instead of dividing. Scaling up is often useful when the values to be graphed are very small. o

continued from page 13

* Long term overcharge. Leaving your batteries on charge longer than the recommended charging time will often appear to diminish capacity, but this is not a permanent fault. A loss of 35% of the total capacity can be revitalised by a single charge/discharge cycle, After this treatment such batteries will typically exhibit 85-90% or more of their original capacity. Shallow discharge/full recharge of Ni-Cad's is perhaps the most well-known effect, yet is probably the least frequent problem, often called "memory effect" -it is the most mis-identified problem associated with Ni-Cads. The condition may be corrected by several DEEP charge/discharge cycles. (I tried to revive "dead" NO voltage reading cells at least three times by 16 hour charges before I finally discard the CADS!)

* Ultimate battery life. As time passes less and less capacity is available and at some point the batteries should be replaced. As many as 1000 charges are common before 80% max. capacity is unobtainable. Fast charging reduces these cycles to around 600/700. Finally when the ambient temp. is 25C or over full charge will not occur and the battery will appear to have lost capacity. Up to 10% may be lost at 35C. Ni-Cads do corrode and pressure contact connections should be neutralised with bi-carb when the tell tale "green" shows up.

It is not uncommon to find that the polarity has reversed!

These Cells may last from one week to at least 8 years and I love/hate them. My charger uses AAA AA C D and 9v and is always in use.

Sealed cells expand and stick in my torch. Open cells used in the model aircraft industry control the charging rate by thermostat, thus avoiding any rise in temperature over the dreaded 35C whilst being charged. Happy charging and thanks for the prompt by D.R.Y. Cell.

Decoding EPROM files

An exercise in File Handling Techniques

by Ben Takach

Fixed programs are frequently stored on Eproms. Eproms are convenient devices to store a set of codes which may be read and executed by a computer or processor. The Mechatronic Eprom reader/programmer attachment has enabled the TI community to copy or burn new eproms. One may also inspect the code, save it to disk or modify it at will.

Alas this handy peripheral will offer no more. One has to write a suitable program to work with the code in any other way. The following paragraphs will explain the ways and means to do just that.

My need to write an Eprom code manipulating program started with the necessity to modify a PLC program that controls a machine, which is no longer supported by its maker. This is a frequent problem in our fast changing electronic age when the electronic equipment you are likely to purchase is obsolete by the time you walk out with it from the store. It is a safe bet, that most of you will never have the need to pry into the inner secrets of a PLC, however you may find much useful information in the following pages. All the same one can never tell, if you do one day, some of this information may save you many hours of frustration.

PLC's are used in many industrial control applications. These are more or less simple, dedicated computers, hell bent on executing the pre-programmed task only, and the PLC can do nothing else. For better understanding Fig.1 shows the functional diagram of a PLC. The heart of the PLC is a simple minded, primitive processor chip. This is frequently a single bit processor working with a modest total of 16 instructions and has only a few registers.

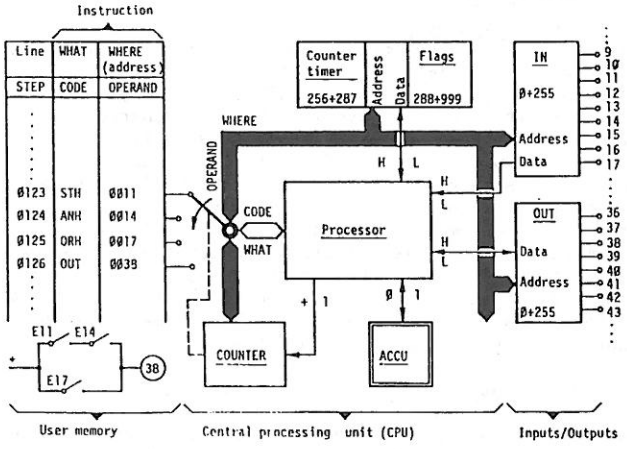


Figure 1: Function Concept of a PLC

One such popular device is the Motorola MC14500B. The IC is in a 16 pin 0.3"dil package. One would never guess that this unassuming chip could do so much! I will not bore you too much longer with the virtues of this chip, nor the PLC's, however a few more sentences are essential in order to understand the project. In addition the reader may study the architecture of this chip shown in fig.2. Using the MC14500B as an example, I will explain its programming syntax, and how one can reconstruct the original program from the Eprom code. The simple instruction set enables the processor to execute complex programs, which are only limited by the code one can burn in the associated eproms. Thus a complicated relay operated control circuit, having several hundred control relays, may be replaced by a PLC not larger than a shoebox.

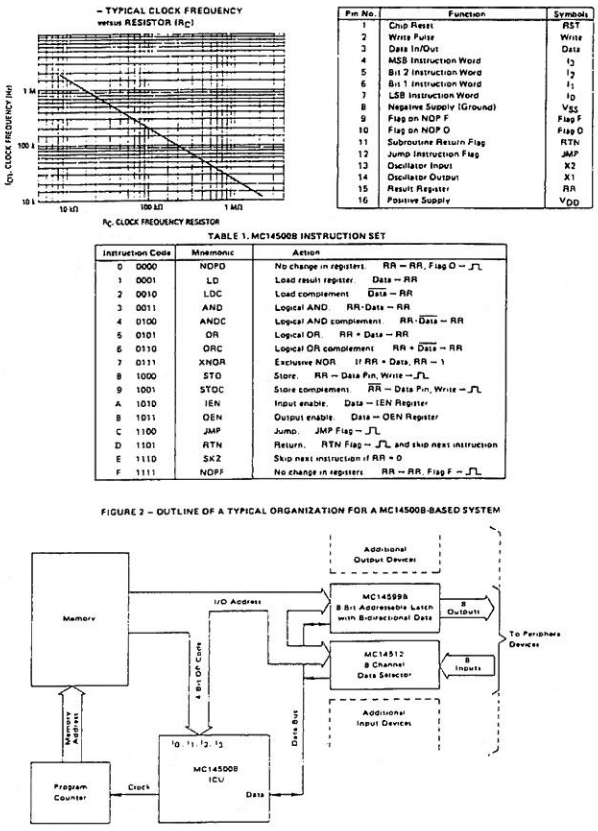
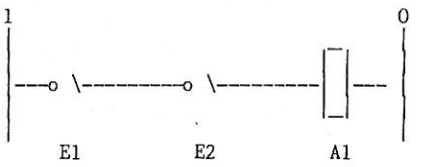


Figure 2: MC14500B Data Sheet Extract

The instruction set will include: NOP (no change,-no operation), LD (load register), AND, OR, STO (store), IEN (input enable), OEN (output enable), JMP(jump) as well as the compliments of these instructions. The 16 instructions are the hex digits from 0 to F. (see fig.2).

One may write a program using these instructions to emulate branches of a relay circuit. Let me give you just one example, and then we will close the subject of the PLC. Assuming we wish to program a relay coil which will be energised if or when 2 limit switches (or contacts) are closed:



The program will be:

Line No.	Address	Eprom Mnemonics (INSTRUCTION)	Operand
004	008	LD	E1
005	00A	AND	E2
006	00C	STO	A1

If we have defined that the address of:

E1 = 000
E2 = 001
A1 = 200

and the Hex code of the processors instruction set is:

LD = 1
AND = 3
STO = 8

then the eprom code for these three program steps will be:

1000 3001 8200

The eprom address of these 3 words will be:

008+009, 00A+00B, 00C+00D

I have started with line number 4, because lines 0,1,2 and 3 are used to reset and initialise the processor chip.

By now you have probably worked out that if one knows the definition assignments and the processor instruction set then the entire program can be written out line after line from an eprom by decoding the eprom code word by word. It is a straight forward and simple task, however it is also very time consuming, considering that the PLC may use 4x16K eproms, and the program may fill 35 to 40 A4 size pages. Hence the need for a program to decode the eprom and print out the entire program. I chose the XB language because it does not have to be compiled and in case of an error, the trouble is easily rectified.

The most important criterion is to access the eprom code file from XB. The epromer saves the code in a program image file, which is not directly accessible by XB. One may write a short assembly program to place the code into the memory, then subsequently the code can be "peeked" byte by byte for decoding. This approach would end in a no win situation. The eprom code is generally too long, the program would bomb out very early with a cryptic "memory full in nnn.line " message on the screen. The eprom code has to be in an XB compatible sequential file to conserve the 99/4A's limited memory capacity. Likewise, the Definition file of the PLC program may also be very large, thus this file also has to be in an XB compatible file format.

How can we convert a program image file to something XB can understand? Simple! Assuming you wish to access a file from an XB program, which has been created by TI-Writer. It is well known that such a file is always saved in a D/V80 format. You obviously cannot access such a file from XB. However, you may use the Print file command with a 'F' in front of the file name, and print your file to Disk. The resultant file will be a D/F80 file. XB can read this! We can use one of the sector editor programs to convert the program image file into a D/F80 format file in two steps, via TI-Writer.

Over the years a number of disk sector editor programs have been written for the TI. There is the original TI-Disk Fixer, The Navarone Disk Sector Editor, Disk Aid by M&T utility ware and John Birdwell's Disk Utilities. Each one of these is a useful program, each one more or less does the same thing in a different way, however none of these are perfect, nor universally useable and most of these have some bugs. The above sentence sums up the collective virtues of disk sector editors. We cannot use the TI-Disk fixer to print out sectors at all, because it has no print option, however it is the easiest to use program of the four. It will search for a file and provide information regarding the file header, the first and last sector number of a particular file in one step. One may also step through the sectors by a single key stroke in Hex or ASCII format. It also provides sector editing function.

My copy of the TI-Disk fixer will not work if the file is in Drive 2 or 3, although all 3 are identical drives. Further, it will happily step through an entire disk backwards (using FCN4), but will come up with an error message after a few increments using FCN6. The Navarone program as well as Disk Aide is useless if you wish to print to a disk. Both will do it, if the printer set up is changed to DSKn.File name, but the result will only contain the last nominated sector. Both will overwrite the sectors! In addition the Navarone program creates a D/V82 file, thus is not suitable for further processing by TI-Writer.

Disk Utilities will do the trick! Albeit not without problems! To create a D/V80 disk file using Disk Utilities proceed as follows:

1. Change the set up parameters to direct the printout to Disk; change the printer option to DSKn.File name. Also select standard mode, which will result in a D/V80 file.
2. Select sector Utilities option from the menu.
3. Enter the first sector, then push CTR P (for print) then push N(for next), then push ENTER.
4. Step through the sectors you wish to dump to your disk file repeating the key strokes CTR P, N, ENTER until the last desired sector is printed to disk.

Readers who are familiar with Disk Utilities may ask at this point, why not use the "File Print" option? Because this program also has a bug! The last word in each sector will disappear. It will unashamedly swallow the 255th and 256th byte of each disk sector without letting you know! Obviously its assembly equivalent of For-to-next loop will stop short of the last word.

Once the file is on disk it is easily edited by TI-Writer. Disk Utilities produce a 40 column wide file format (L=1 R=39), and each sector dump is followed by the identifying text and a few empty lines. One can quickly delete the text lines and the empty lines with a few function 3 key strokes until a continuous file of 40 col.Hex code is left on the screen. This will be in a word format (4 characters then space), 8 words per line (8x4+8 spaces = 40 char.). In fact this would be a useable format, however the resulting DF/80 file would be very long, as only 50% of the disk space would be filled with code. We can change the tab setting to 80 char. and reformat the file to 80 char/line format. Now we have 16 words per line (16x4+16 spaces = 80 char.).

The next step is to select the PF option and print the file to disk by responding to the print device prompt: F DSKn. FILE NAME. This process converts our eprom file to a D/F80 file, which is compatible with XB. The file will be used by the decoding program. The next task is to create an XB compatible, random access, fixed record length file of the definitions.

A predetermined 3 bit code has to be assigned to each operand. The operands are the input and output ports, the timers, the labels and the flag memories. The flag memories are internal outputs (phantom relays). The PLC treats these as outputs. The address of each is a 3 character code, from 000-FFF not including the labels. Most programs will use 2 labels, however in some rare instances one may find up to 4 labels in a program. Thus we have to create a file with 4100 records ((H)FFF=(D)4095+0000+4 labels=4100).

The program to generate this hefty file is quite simple. One may type in the example printed in the TI Disk memory System manual (pages 38,39). The program will work, but alas is not very user-friendly. It also has a bug, or rather an annoying short coming. If you do not know the last record number you can easily get trapped in the 10000-10040 sub-routine without a chance to exit from it gracefully. I have edited and augmented this program to make it more user-friendly. The program listing is reproduced below. I saved it as File-I/O.

```
100 CALL CLEAR
110 OPEN #2:"DSK1.GENFILE",RELATIVE 50,INTERNAL
120 PRINT "ENTER 'XXX' TO LEAVE ENTRY"::
130 REM ENTRY SECTION
140 TOTAL=0
150 INPUT "RECORD "&STR$(TOTAL+1)&" ":BUFFER$
160 IF BUFFER$="XXX" THEN 220
170 PRINT #2,REC TOTAL+1:BUFFER$
180 TOTAL=TOTAL+1
190 IF TOTAL<50 THEN 150
200 REM PRINT FULL MESSAGE
210 PRINT "FILE FULL"
220 REM CONTROL SECTION
230 PRINT ::"1 = PRINT FILE"
240 PRINT "2 = PRINT A RECORD"
250 PRINT "3 = CHANGE A RECORD"
```

continued on page 18

Beginning Forth - part 11

by Earl Raguse, UGOC, CA USA

FLOATING POINT ARITHMETIC

Last time I promised to get into FPA or Floating Point Arithmetic. This will permit us to do graphics which require the use of square root, transcendental and trigonometric functions. Please read the TIFM Chapter 7. I am not going to repeat all that. Read at least the first three pages anyway. The last two are pretty technical, and do not have to be understood to use FPA.

Floating Point numbers are represented in the computer with 8 (count em, eight) bytes, in Radix 100 notation. This is one of the reasons FPA is slow. This is equivalent to four sixteen bit words (64 bits total). Radix 100 representation uses the same built-in computer firmware capability that XBASIC or ASSY uses. If you are curious about this, read the TI EDITOR/ASSEMBLER Manual page 279.

The process of multiplying, for example, a pair of numbers represented by four sixteen bit numbers each is substantially more complex than integer multiplication of a pair of sixteen bit numbers. I will not explain in detail how this is done. I could say that is because you would probably be bored anyway, but the real fact is that I do not know. I have read books on computer design from the library on the general principle, but nowhere have I seen a discussion on exactly how TI does it. Fortunately it is not necessary to know in detail how the computer works to make it work for us.

The TI version of FPA provides every operation available to us for integer arithmetic, and then some. All the standard arithmetic and relational operators like +, -, *, /, >, <, =, 0=, and 0< are provided, by prefixing them with F. This applies to the stack manipulators too like FDUP, FOVER, and FSWAP. For some reason, FROT is not provided, but if needed, it could easily be defined. Fetch, Print and Store are done similarly as, F@, F. and F!. We also have some new words, some of which I will be using, SIN, COS, TAN, ATAN, SQR, LOG, PI and the exponential, all of which are identical to XBASIC commands of the same spelling.

X11 very good, but how do we get numbers into FP notation? TI has provided us with S->F and F->S to move numbers from S (Single) on the stack to F (Floating) on the stack, and vise versa. In addition, we may use >F from the keyboard or on a screen to convert a string of digits to an FP number. Note also, the nice words F.R, FF., and FF.R, explained on page 3 of the TIFM, Chapter 7, as a means of controlling number formatting in lists.

TI Forth also provides >FAC, >ARG, FAC>, S->FAC, FAC->S and FAC>ARG, and other related words like FADD, FSUB, FMUL, and FDIV, all of which have the same meaning as the similar words in ASSY. I will not use or explain these. If you understand such stuff, read on in the TIFM Chapter 7, and the E/A manual.

GRAPHICS EXAMPLES

This brings me to the point where I am forced to provide some examples. Screens #66 through #68 are examples of of both DOT and LINE graphics using FPA, which I did back when I was just beginning to figure out Forth. The first Screen #66 defines several variables, namely Q (angle), R1 (radius, cannot use R that is a resident Forth word), XX (X axis), YY (Y axis), W (width), H (height), and Z (a parameter). The words LOC (LOCate), MENU> (to the MENU), E1 (EXIT word), and QUIT? (looks for FCTN 4), should be obvious.

HOT, GALAXY and their supporting cast, H1 through H8, actually do the graphic work. Do not ask why I named them that way, I had to have names, so I made them short. When you see GALAXY you will understand that one. HOT is an infinite loop using MYSELF (Remember those loops from BFORTH #5?), which initializes R1 and Z, H7 announces, and H8 puts you into a DO +LOOP which calls H5 which plots a DOT after H2, H3, and H4 compute the coordinates. You can exit H8 with FCTN 4. The formulae executed by H2, H3 and H4 are:

x=r cos Q + W and y=r sin Q + H

You all remember that from your high school Trigonometry classes do you not? These days they do not teach kids about trigonometry, after all, you do not need to know that to sell insurance, or used cars, so they have to make it up to attend most colleges.

Execute HOT (as in, Some Like It) and write down the numbers of the patterns you would like to see again. That is what GALAXY does, it expects a number on the stack, then performs the galaxy that HOT did when is displayed that number.

What kind of patterns are they? Well, they vary, I have never seen one exactly duplicated, and I have watched pattern numbers up into the hundreds. I do not really pretend to understand exactly what is happening. Galaxy #6 is what I set out to program originally, but the thing got out of hand. I did not really give it much thought before I started, I was thinking degrees, but the computer was thinking radians. These are related by an infinitely non-repeating decimal number called PI.

You will discover the the numbers 11, 33, 55, 77, 99, etc tend to be related. Also the numbers 22 (the weirdest), 44, 66, and 88 produce another series. These all have the common factor 11, but that does not help much toward understanding, just enjoy.

The basic principle of the algorithm is simple, I am just using a loop and computing coordinates XX YY at the end of a radius vector, using SIN and COS, at which a DOT is plotted. The vector is rotated each time by incrementing the angle Q and the length of the radius R1. The values W and H are the X and Y coordinates of the center, and you can control these (see the menu for CIRC) before executing GALAXY, however, HOT resets them using LOC to the middle of the CRT.

```
     SCR #66
 0  ( DOT GRAFIK ER 4 85 )  FG IT : IT ; 0 VARIABLE Q
 1    >F 2 VARIABLE R1 6 ALLOT 0 VARIABLE XX 0 VARIABLE YY
 2    120 VARIABLE W  90 VARIABLE H  0 VARIABLE Z CLS
 3    5 12 AT ." LOADING, AIN'T WAITING FUN ? "
 4  : LOC 120 W ! 90 H ! ;  : MENU> TEXT 69 LOAD QUIT ;
 5  : E1 8 9 AT  ." TO EXIT ENTER <FCTN 4> " 1 WAIT ;
 6  : QUIT? ?TERMINAL IF TEXT 0 Z ! MENU> ENDIF ;
 7  : H1 GRAPHICS2 -1 DCOLOR ! 0 DMODE ! 1 SCREEN ;
 8  : H2 R1 F@ >F .4 F+ >F 1.01 F*  R1 F! ;
 9  : H3 R1 F@ Q @ S->F COS F* F->S W @ + XX ! ;
10  : H4 R1 F@ Q @ S->F SIN F* F->S H @ + YY ! ;
11  : H5 H2 H3 H4 XX @ YY @ DOT ;
12  : H7 TEXT 10 7 AT ." DOT GALAXY NO. " Z @ . E1 ;
13  : H8 H1 120 Z @ * 0 DO I Q ! H5 QUIT? Z @ +LOOP ;
14  : HOT LOC >F 2 R1 F! 1 Z +! H7 H8 1 WAIT MYSELF ;
15  : GALAXY ( Z --) >F 2 R1 F! Z ! H7 H8 MENU> 0 Z ! ; -->
```

```
SCR #67
 0  ( EXPERIMENT IN DOT GRAPHICS 4 85 )
 1   0 VARIABLE RSQ  0 VARIABLE SQT 6 ALLOT
 2  : X W ! ; : Y H ! ;
 3  : ARC  R1 @ DUP 1 + SWAP 1 MINUS * DO RSQ @ I I * - S->F
 4   SQR SQT F! SQT F@ W @ S->F F+ F->S I H @ + DOT QUIT?
 5   SQT F@ MINUS  W @ S->F F+ F->S I H @ + DOT LOOP ;
 6  : CIRC  DUP R1 ! DUP * RSQ ! H1 ARC MENU> ;
 7  : JOT1 35 W ! ;
 8  : JOT2 60 H ! ;
 9  : JOT3 15 DUP R1 ! DUP  * RSQ ! ;
10  : JOT4 W @ 25 + W ! ;  : JOT5 H @ 25 + H ! ;
11  : JOT6 3 1 DO JOT5 ARC QUIT? LOOP ;
12  : JOT7 7 1 DO JOT2 JOT4 JOT6 LOOP ;
13  : JOT  CLS E1 H1 JOT1 JOT3 JOT7 MENU> ;
14    CLS 12 12 AT ." ALMOST DONE!"
15  -->
```

On Screen #67, we see how x and y coordinates may be entered from the keyboard using words X and Y. Two new VARIABLEs RSQ and SQT are defined to hold the values of R squared, and a square root, to be used in computing x=(R 2-y 2) (-1/2), which can be used to plot a circle. I wanted to see if the square root approach was faster than the SIN COS approach. You be the judge. All the computation is done by ARC, which computes two points on each side of the vertical axis of a circle. The JOT series just uses ARC in two loops to draw circles on the CRT, they may be a little confusing, but not mysterious if you study them.

Screen #68 is a repeat of #66 except that I now use LINE to connect pairs of DOTs. This often changes the appearance of the figure compared to plain dots. To make this simpler, I had to invent 2ROVER, which saves the point at the end of the current line and pushes it down into the stack as the 5th and 6th numbers which will become the top when the LINE is drawn. To do 2ROVER, I had to invent 2SWAP, both of these are on my Stack Manipulator #33 screen of BFORTH #8, OVER OVER is the same as 2DUP. One new thing here is that instead of H1, I have G1 which changes the values in DCOLOR and SCREEN color. G2, G3 and G4 are the same as the H version. FIG, like HOT, lets you see all the patterns, and DOODLE shows any specific number that you put on the stack. Note that if you decide to GALAXY and DOODLE often, they have abbreviated to G and D. Forth is nice that way.

If you have any questions about how all this works, please do not hesitate to ask me. Next time I am going to show how we can speed all this up by substituting approximate values for the FPA stuff.

CU next time; May the Forth be with U!

```
SCR #68
 0  ( SPIRAL GRAPHIC ER 9 27 85 )
 1  : D3 CLS 12 7 AT ." DOODLE NO. " Z @ . E1 ;  HEX
 2  : 2SWAP   ROT >R ROT R> ;
 3  : 2ROVER ( n1 n2 n3 n4 -- n3 n4 n1 n2 n3 n4 )
 4     OVER OVER >R >R 2SWAP R> R> ;
 5  : G1 GRAPHICS2 F0 DCOLOR ! 0 DMODE ! 4 SCREEN ; DEC
 6  : G2 R1 F@ >F .3 F+ >F 1.03 F* R1 F! ;
 7  : G3 R1 F@ Q @ S->F COS F* F->S W @ + XX ! ;
 8  : G4 R1 F@ Q @ S->F SIN F* F->S H @ + YY ! ;
 9  : G5 G2 G3 G4 XX @ YY @ QUIT? ;
10  : G6 G1 G5 OVER OVER DOT ;
11  : G7 G6 62 Z @ * 0 DO G5 2ROVER
12     I Q ! LINE QUIT? Z @ +LOOP ;
13  : FIG  LOC >F 5 R1 F! 1 Z +! D3 G7 1 WAIT MYSELF ;
14  : DOODLE >F 5 R1 F! Z ! D3 G7 0 Z ! MENU> ;
15  : D DOODLE ;  : G GALAXY ;   -->

SCR # 69
 0  ( DOT GRAPHICS  INSTRUCTIONS) CLS
 1  3 3 AT ."        TO DRAW A CIRCLE  "
 2  3 4 AT ."    OF RADIUS R CENTER @ X,Y "
 3  3 5 AT ."   DEFAULT IS AT CENTER SCREEN "
 4  3 7 AT ." ELSE, ENTER x COORDINATE THEN X  "
 5  3 8 AT ."       ENTER y COORDINATE THEN Y "
 6  3 9 AT ."       THEN PRESS <ENTER>. "
 7  3 11 AT ."  NEXT ENTER THE CIRCLE RADIUS "
 8  3 12 AT ."     THEN <CIRC> <ENTER> "
 9  3 15 AT ." ELSE JUST FOR FUN TRY <JOT>, or  "
10  3 16 AT ." <HOT>,<n GALAXY>,<FIG>,<n DOODLE> "
11  3 17 AT ." THEN <ENTER>. "
12
13
14
15                                              o
```

continued from page 21

I see I have brought up two topics- ANALOG JOYSTICKS and VHS TAPE OUTPUTs for GAMES SIMULATIONS AND SPEECH. Well, they are connected! People who break Joysticks will perhaps want a VHS TAPE showing Munchman getting pushed over a hundred thousand. It may be possible to get a million in Munchman, but so far I have not achieved it, and I still recommend building ANALOG Joysticks for those serious about getting a hundred thousand or better in it. o

continued from page 16

```
260 PRINT "4 = ADD A RECORD"
270 PRINT "5 = LEAVE THE PROGRAM":::
280 INPUT "ENTER YOUR CHOICE? ":X
290 PRINT
300 IF X<1 THEN 220
310 IF X>5 THEN 220
320 ON X GOTO 1000,2000,3000,4000,5000
1000 REM PRINT A FILE
1010 RESTORE #2,REC 1
1020 FOR COUNT=1 TO TOTAL
1030 INPUT #2:BUFFER$
1040 PRINT COUNT;BUFFER$
1050 NEXT COUNT
1060 GOTO 220
2000 REM PRINT A RECORD
2010 GOSUB 10000
2020 INPUT #2,REC RECNUM:BUFFER$
2030 PRINT RECNUM;BUFFER$
2040 GOTO 220
3000 REM CHANGE A RECORD
3010 GOSUB 10000
3020 INPUT "ENTER NEW DATA? ":BUFFER$
3030 PRINT #2,REC RECNUM:BUFFER$
3040 GOTO 220
4000 REM ADD RECORD
4010 IF TOTAL=50 THEN 210
4020 TOTAL=TOTAL+1
4030 INPUT "ENTER DATA? ":BUFFER$
4040 PRINT #2,REC TOTAL:BUFFER$
4050 GOTO 220
5000 REM LEAVE THE PROGRAM
5010 CLOSE #2
5020 END
10000 INPUT "WHICH RECORD? ":RECNUM
10010 IF RECNUM<1 THEN 10030
10020 IF RECNUM<=TOTAL THEN 10050
10030 PRINT "INVALID RECORD NUMBER"
10040 GOTO 10000
10050 RETURN
```

Figure 3: The File I/O Program Published by TI

This program will create my internal, fixed 40, relative file of 4100 records. The original TI program was further edited for general use. The input statement in line 150 used to input the record and assigned it to the string BUFFER$. The string was not tested for length, thus one could type in a longer than 40 char string. In such a case the computer will end the program with a file error. In my case although none of the definitions will exceed the 40 character limit, however the program will check the length of each entry before writing it to disk. If the program is to be used for general file handling routines then one ought to add a few lines before the open statement to define the desired record length from the keyboard.

Incidentally, the 40 character record length is a good compromise. Each disk sector will store 6 records, thus a DSDD disk could store a very large file.

My modification to test the length of the string in line 155 represents but one of many options. The original TI program will run in console BASIC or extended BASIC. I did try to use commands which are compatible with console BASIC. The few multi-statement lines were added during final testing, these are easily broken up into separate program lines. Program lines 6000-6090 and 7000-7090 are added to execute the menu options 6&7.

Option 6, to fill the entire file with Hex20 (spaces) is a very wise precaution. It may save you some embarrassment if your record lengths are not the same throughout the file. If you happen to use a disk, which has had some previous files deleted, you may end up with the telephone numbers of your favourite girlfriends printed out on your Christmas card list. The computer will not delete characters already on the track, unless it is in a location, which will be overwritten by the new record. For that reason line 6030 prints 40 space characters to deliberately clean the entire record length of any garbage.

Now you may recycle any of your old disks without the risk of some odd litter-padding on your newly created records, whenever these happen to be shorter than the designated record length. o

TI-Base Tutorial #15

by Martin Smoley, North Coast 99ers USA

I am reserving the copyright on this material, but I will allow the copying of this material by anyone under the following conditions. (1) It must be copied in its entirety with no changes. (2) If it is retyped, credit must be given to myself and the NorthCoast 99ers, as above. (3) The last major condition is that there may not be any profit directly involved in the copying or transfer of this material. In other words, Clubs can use it in their newsletters and you can give a copy to your friend as long as it is free.

```
*                       FORTST9
CLOSE ALL
USE CHKBK
PRINT (Drft),(f)
* (1)
* PRINT ALL ;FOR CHKNO="    "
* - - - - - - - - - - - - - - - - - - - - *
* (2)
* PRINT ALL ;FOR ((CHKNO>" 269").AND.;
*                (CHKNO<" 276"))
* - - - - - - - - - - - - - - - - - - - - *
* (3)
* PRINT ALL ;FOR ((DATE>"02/29/89").AND.;
*                (DATE<"04/00/89"))
* - - - - - - - - - - - - - - - - - - - - *
* (4)
* PRINT ALL ;FOR ((DATE>"02/29/89").AND.;
* (DATE<"04/00/89")).OR.(CHKNO="    ")
* - - - - - - - - - - - - - - - - - - - - *
* (5)
* PRINT ALL ;FOR (DATE<"01/01/89").OR.;
* ((CHKNO>"    ").AND.(DATE="00/00/00"))
* - - - - - - - - - - - - - - - - - - - - *
* (6)
* PRINT ALL ;FOR (DATE>"09/29/88").AND.;
* (DATE<"04/00/89").AND.(CHKNO="    ");
* .AND.(DEP'CREDIT< .01)
* - - - - - - - - - - - - - - - - - - - - *
* (7)
* PRINT ALL ;FOR ((DATE>"02/29/89").AND.;
* (DATE<"04/00/89")).AND.((CHKNO="    ");
* .OR.(REMARKS="Gas    ,"))
* - - - - - - - - - - - - - - - - - - - - *
* (8)
* DISPLAY ALL DATE,DEP'CREDIT,BALANCE ;
*   ;FOR CHKNO="    "
* - - - - - - - - - - - - - - - - - - - - *
* (9)
DISPLAY ALL DATE,DEP'CREDIT,BALANCE ;
  ;FOR ((DATE>"02/29/89").AND.;
  (DATE<"04/00/89")).AND.((CHKNO="    ");
  .OR.(REMARKS="Gas    ,"))
* - - - - - - - - - - - - - - - - - - - - *
CLOSE ALL
PRINT (Drft),(E)
RETURN  Copyright Martin A. Smoley 1989
*       ;FOR clause demonstrations.
```

This is a continuation of last month's tutorial. The Databases I used last month are not important, and you need not type them in. I hope that my rambling explanations of the TI-Base language will give you enough information to adapt the CFs I present to your own Dbs, without the need to create all my examples. I also hope to give you more information about how I write my own CFs, find bugs in those CFs and reconstruct those CFs to produce the end results which are needed.

Last month I gave you FORTST1. This month we have FORTST9. I created this CF for your benefit, but I have done the same thing for myself in the past. All nine statements will work if the asterisks are removed from the beginning of those lines, as in number (9). However, we will start at the top. As I have stated previously, the first thing I like to do is CLOSE ALL of the Dbs that I may have forgotten in an open condition. Next I USE CHKBK. In this way I have TIB keep track of what I am doing and I also have a visual reference to the name of the DB (CHKBK) this CF handles. Remember, without changing the fieldnames this CF or the commands in it will only work on the DB named CHKBK. "PRINT (Drft),(f)", is my personal printer setup. (Drft), is my command to reset the printer back to a simple draft mode and (f) is my command to set the printer to condensed mode. You will notice that before I leave this CF I reset my printer and then set it to Emphasized (E) mode. These are my personal preferences, so you may substitute your own or leave these lines out of the CF. Remember, each of the nine commands in FORTST9 can be used in a CF, as I have, or by themselves if typed in as a command line. Command (1) is fairly simple. It says PRINT ALL of the Fields that are contained in all the Records that have blank check numbers, (;FOR CHKNO=" "). In the case of CHKBK, only record numbers 85 through 97 would be printed. Suppose you wanted to see the Records for check numbers 270, 271, 272, 273, 274 and 275. Command (2) will print those records for you. These commands are quite simple but at the same time they are very very tricky, because you must tell TIB ((EXACTLY!!!)) what to do. This EXACTLY! business is the part that discourages many programmers. Let us take a quick look at this EXACTLY stuff. There is no way to tell TIB (in loose terms), that I want to see the data from 270 through 275. I know that these numbers fall between (but do not include) 269 and 276, notice the (and). Between 269 and 276 sounds grammatically correct, between 269 or 276 does not. It is more complicated than that, but this is a good way to keep yourself straight as far as .AND. and .OR. are concerned. .AND., will give you the records between 269 AND 276, but .OR. (in this case), would give you all the records in the Db. So now we have two distinct qualifications for a record to be printed. It is CHKNO must be greater than (>) 269 and less than (<) 276, or CHKNO>269 and CHKNO<276. "Sounds pretty close does it not? Well it is not." For one thing we must tell TIB every detail about the data it will encounter in the CHKNO field. For instance the quotation marks, ("1234"). The quotation marks tell TIB that when it looks into the CHKNO field it will see a character (C) type field. TIB must know this. If you look at the end of Command (6) you will see (DEP'CREDIT< .01). In that command TIB will be looking for a numeric (N) type field and the quotes must not be used. Also Note that the CHKNO field is 4 characters wide, with a space in column one (" 269").

Both the width and justification of a character type field are important. TIB would not find a match for "273 ", because the space is in the wrong place. TIB also needs to know that (CHKNO>" 269") is one small but complete question which it must answer. Thus the reason for the parenthesis. I have also placed parenthesis around the upper search limit (CHKNO<" 276"), and for myself, I placed parenthesis around the whole thing. The outer parenthesis are helpful when more queries are added to are ;FOR clause. Take a look at number (3). I took a copy of number (2) and changed our field of interest. My idea was to find everything (deposits, debits etc.) for March (03/nn/nn). Because February could have 29 days, I set our low limit to "02/29/89". I then set the high limit to April (04/00/89). Because I did not want items from either Feb. or April, I made the command (DATE>"February") and (DATE<"April"). I told TIB that it would be looking for a date (D) type field "nn/nn/nn" and I used quotation marks. Note: All nine of the commands in FORTST9 work, but they are to demonstrate the makeup of command lines and not necessarily to find meaningful data. You should also take note of the way I break my lines in this CF. The Command Processor will only read lines up to 40 characters in length. You can continue a command line, in a CF, up to 255 characters by placing a semicolon at the end of each line (40 chars. or less) and continuing on the next line down, as you see in FORTST9. TIB does not require that you start the next line at the very beginning or far right column, see (2) and (3). So this is the way I like things. If you do not need every space then do not fill every space, waste space. Stop a line before column 40 at a convenient, easy to read, easy to understand, eye pleasing spot. Start and stop

the next line under the same conditions. You will notice in (2) and (3) that I have even lined up the parts of my ;FOR limits directly above and below each other. This allows me to easily see what I am doing and to check for matching quotation marks or parenthesis. While you are writing the program you will remember everything, but if you must make a change six months after the program is finished, you will not have the slightest idea what you had in mind when you wrote it. OK, back to the CF. Number (4) is a slightly more complicated set of limitations. You should see that (4) is a direct copy of (3) with the addition of ".OR.(CHKNO=" ")", which is actually the same as number (1). In many cases where I have complicated problems I find it easier to solve pieces of that problem separately, such as (1) and (3), and then combine the pieces, such as (4), to handle the whole problem in one shot. Number (4) will print all records for March "03/nn/nn" and it will also print all records that have no check number (CHKNO). You should think of ".OR." as "Either Or" and you should think of ".AND." as "Must satisfy both, or more than one requirement". In this command TIB will think (if the record is either for March OR it has no check number print it). Buried within the statement on the left hand side is the March limitation, which says that the DATE must fall above "02/29/89" and below "04/00/89". "Figuring out these examples is making me crazy, it must be doing the same thing to you." I created number (5) to reinforce the thought that an .AND. requires more than one portion of the clause to be true. I have enclosed the two items that are directly related to the

This should remind me that they must both be true. Command number (6) is to demonstrate that several .AND.s or .OR.s can be used in the same statement. In the case of number (6), all four of the requirements must be true before a record will be printed. In number (7) three items must be true. The two DATE items must be true and either the (CHKNO=" ") must be true .OR. the (DEP'CREDIT< .01) must be true. Number (8) is roughly number (1) and number (9) is roughly number (7). In these two I am merely showing that DISPLAY is interchangable with PRINT, and that you can request that specific fields be DISPLAYed or PRINTed rather than the complete record. It is interesting to note that (REMARKS="Gas ,") will be found if REMARKS has no characters other than Gas ,. I am covering this as thoroughly as possible and I will probably cover it again because the logic used with the ;FOR clause is the same as that used with IF or WHILE statements. They must be understood clearly if you are to recognize the logic which is built into TI-Base.

```
* MOVE NUMBERS TO CHKBK   FLCKBKP/C
SELECT 2
 USE CHKBK
SELECT 1
 USE BILLS
  WHILE .NOT. (EOF)
   SELECT 2
PRINT
   FIND 1.PCHK
* NOTE: TIB is looking through CHKBK
* and attempting to find a match for
* PCHK, which is the Phone Check No.
* in BILLS.  CHKBK must be SORTed ON
* CHKNO for FIND to work.
PRINT
    IF .NOT. (EOF)
     REPLACE 2.DATE WITH 1.MODATE
     REPLACE 2.PAY'DEBIT WITH 1.PHONE
     REPLACE REMARKS WITH "Phone  ,"
    ENDIF
   SELECT 1
    MOVE
PRINT
  ENDWHILE
 CLOSE ALL
RETURN Copyright Martin A. Smoley 1989
```

Let us go back to the CF named FLCHKBKP/C for just a moment or two. When I first wrote it, it was a piece of junk. It was not worth beans. I got a "database not sorted" error message, I selected the wrong slots and I was trying to move data to and from the wrong fields. If you think that you are the only one who makes mistakes, your wrong. My favourite debugging tool, aside from watching the screen for asterisks and error messages, is PRINT. If you place the command PRINT at a key, or problem spot in the CF, TIB will print out the DB Heading and the complete record it is using at that moment. I find this technique to be most helpful after a SELECT or SELECT, MOVE. I watch the CF scroll up the screen and when my printer dumps some data I hold the spacebar to stop the CF. I can then observe the CF on the screen and read the record which has been printed, to see if the CF is actually where I thought it was. I can then press the S key to get TIB started again. o

Jenny's Younger Set

Again, the only mail I received this month was a letter from Vincent Maker who has a follow-up program to last month's program.

```
100 RIGHT=0
110 WRONG=0
120 REM BY VINCENT MAKER
130 CALL CLEAR
140 DISPLAY AT(5,7):"THIS IS A QUIZ TO TEST YOUR
    KNOWLEDGE OF ANCIENT HISTORY."
150 INPUT "PRESS ENTER TO GO ON."
160 CALL CLEAR
170 PRINT "WHO RECORDED THE BATTLE OF THEROPYLEA FOR
    US;."
180 PRINT
190 PRINT "A) HERODOTUS"
200 PRINT "B) THUCYDIDES"
210 PRINT "C) KING LEONIDAS"
220 PRINT "D) WHO KNOWS"
230 PRINT "PRESS YOUR ANSWER"
240 CALL KEY(0,J,K)
250 IF K=0 THEN 240
260 IF J=65 THEN RIGHT=1 ELSE WRONG=1
270 IF J=65 THEN PRINT "WELL DONE, RIGHT" ELSE PRINT
    "SORRY, WRONG"
280 INPUT "PRESS ENTER TO CONTINUE":JKL$
290 CALL CLEAR
300 PRINT "HOW MANY CONSULSHIPS DID MARIUS HAVE BEFORE
    HE DIED?"
310 PRINT
320 PRINT "A) 1"
330 PRINT "B) 3"
340 PRINT "C) 7"
350 PRINT "D) 8"
360 PRINT "PRESS THE ANSWER"
370 CALL KEY(0,K,L)
380 IF L=0 THEN 370
390 IF K=67 THEN RIGHT=RIGHT+1 ELSE WRONG=WRONG+1
400 IF K=67 THEN PRINT "WELL DONE - RIGHT" ELSE PRINT
    "SORRY, WRONG"
410 INPUT "PRESS ENTER TO GO ON":GH$
420 CALL CLEAR
430 PRINT "WHO WON THE BATTLE OF MANTINEA?"
440 PRINT
450 PRINT "A) SPARTA"
460 PRINT "B) ATHENS"
470 PRINT "C) THEBES"
480 PRINT "D) WHO KNOWS"
490 PRINT "PRESS THE ANSWER"
500 CALL KEY(0,U,I)
510 IF I=0 THEN 500
520 IF U=65 THEN RIGHT=RIGHT+1 ELSE WRONG=WRONG+1
530 IF U=65 THEN PRINT "WELL DONE, RIGHT" ELSE PRINT
    "SORRY, WRONG"
540 INPUT "PRESS ENTER TO GO ON":JKL$
550 CALL CLEAR
560 PRINT "HOW LONG DID THE PEACE OF NICIAS LAST?"
570 PRINT
580 PRINT "A) 100 YEARS"
590 PRINT "B) 50 YEARS"
600 PRINT "C) 75 YEARS"
610 PRINT "D) WHO KNOWS"
```

continued on page 22

Analog Joystick

by Daniel Harris

I am finally doing the talk on the Analog Joystick. Big yawn at last meeting- so some members have heard of the thing and it is just a couple of crossed potentiometers.

Well, I know lots of people in TISHUG use their Computer to play games. I made myself learn BASIC and for a while the only plug-in I had was Touch Type because half the skill of using a Computer is touch-typing. I have been tested at 40 Words Per Minute by Secretarial Job Centres so it was not a waste of time. Other games tend to train people to be devils in their motor cars. I have been driven by a person who regards cars as another way to do a SLALOM type of game. I have scored a million plus at Buck Rogers and got over six hundred thousand in Parsec, up to forty thousand in TI Invaders, and recently managed over five hundred thousand in Star Trek. I suppose if I was impressed into the Martian Space Navy I would be not unusable as a Shuttle Pilot or something.

The use of ANALOG JOYSTICKS in the above simulations introduces massive oversteer and crashes. There is some advantage of such a device in Star Trek where the way to go forwards is to push the Joystick forwards. With ANALOG Joysticks that means THROWING the thing and having a nice ride. The wrist is not wrenched and one does not isotonically develop to the point of snapping a Joystick which is where I got the chassis to build an ANALOG joystick. Instead of throwing out the solid-looking piece which had a metal sleeve over a plastic core in the joystick, I resolved to connect up the makings of an ANALOG joystick I had earlier purchased for a couple of dollars at Dick Smith.

About eight hours of work later, reshaping and drilling plastic and fitting self-tappers boldly going where no self-tappers had gone before, into the deep black unexplored regions of jagged plastic, culminated in about six out of ten for the carpentry. But it looks fair and is firm. I also had to locate where all the wires went on the digital circuit and map them, screwing up one's courage to the sticking point and solder to wire and wire to contact on the ANALOG joystick. I only had to change a connection or two and find some way to earth the "Fire" button. At least I have been to Tech. and learned how to solder properly, and how to trace circuits. I also have a Certificate for a course involving BASIC and FORTRAN and of course SYSTEMS-among various other qualifications. Patients will be accepted for minor brain surgery by appointment! At any rate, I wound up with an 80% functional Joystick. It does up, down, left and right, which is four points. It shoots- that makes five points. Add two more points- it gives the diagonals downwards, so that comes to seven features. It does not give the up diagonals. I am thinking of putting a parallel resistor in the up circuit to reduce the up resistance. Two functions out of nine do not work.

Skipping the perfectionism, it is worth mentioning that I came by the chassis out of Munchman, which could be paraphrased to Munchjoystick as I am not the only one thus to demolish or become enraged at the response of the system in Munchman. It is in Munchman, Paint 'n Print, and some features of Star Trek that the ANALOG Joystick is most advantageous. A featherlight touch of the fingers is enough to change direction, eliminating the Wham-Wham-Wham that slides the Joystick across the table and pulls the Adaptor out. Or perhaps, pushes the computer off the table, wrecks the wrist quite seriously, and overdevelops the wrist muscles effectively self- destructing the joystick at the shaft. Whereas so far I have found the Analog Joystick puts the plane into the Electron Posts in Buck Rogers, crashes the ship into the Fuel Tunnels in Parsec, and is a bit tricky for shooting Nomad in Star Trek, the fact of breezing through to a hundred thousand in Munch Man without getting up tight. Nor do I get a sore wrist in an argument in favour of members going to Dick Smith and Radio House and buying up Analog Joysticks, which then need their centre terminals yoked for mounting on snapped-off former joysticks. For that matter, I have found there are people who cannot win at Video Games and have thought of doing a Video Cassette with Voice Dub by yours truly for members who cannot win at their game, whatever it might be. Victory is a lot of special tricks that must be mastered progressively. For instance to get the last sprite in a level of Texas Instruments Invaders is like swatting a fly, but in "The Attack" you have to think like a funnelweb Spider and hide under a Spore to ambush the Alien. To break the million in Buck Rogers you have to apply at least five special techniques.

I think the next project is to start doing VHS tapes of Winning Games. Such tapes are portable presentations of the output of a Texas Instruments Home Computer which other people can put on their own screens regardless of what the system of computer they are using. A good player of a Computer Game can draw a crowd. A VHS Tape will make the replay visible for others. Also the tape can be stopped at various points to look at tricks and examine strategies. It is possible members have never considered VHS Tape as a medium. Look again, you can save your games and simulations to VHS tape. I once had a program edited for publication, the point being that the output was for a Speech Simulator, and somebody changed all the S's into C's and made it look pretty. There were reasons for the funny spelling and unusual punctuation marks- they operated the Speech Synthesizer to make the poem (or spell) SOUND the way the incantation was intended. To this day Gremlins must be laughing at the way "Magic Machine" sounds as coming from the edited version, with order more and chaos less, to avoid which I had purposely made it into chaos, a point speech programmers have found out. Other words like chimera or Christ must have a "k" or else out comes a "tch" where you write a "ch". Other characters such as ">" and "//" have special meanings in Speech Synthesizer.

Now if I had done the output onto a VHS tape the way it ought to have been done, a really emotional chant would come out of the TV set, with the comical evocation of ABRAXAS as the warm-up. Then the words of the spell setting out the war of LIGHT VERSUS DARKNESS, in this case the Order increasing and the Chaos (pronounced Khaos) decreasing until my habit's no distress and all my deeds are sweet success. So Mote it be..(*repeat*) a useful spell to run after a spate of accidents and muddles. You can chant along with it if you like. I hope the Library has the original with the Speech Codes, not the edited version which changes all the Ss into Cs. One symptom of Gremlins is lights blowing when you turn them on, say three times in two days. In Olden Times the symbol of Abraxas was displayed as a preventive of such events. A full expansion of the Mantram for Abraxas comes to Abrasad, Abrasad, Abrasad-LATIN for polish, polish, polish, in other words PRACTICE MAKES PERFECT, doubtless a Masonic slogan of the time. The point is that the will has telekinetic effects as well as acting in the muscular system, and mains currents are very susceptible to telekinetic effects, especially at the highest current values and about 415 Volts. Those of you with some Rosicrucian training will recall causing blackouts and down time in factories during certain phases of development exercises. The mains transformer for the whole district goes down when the Neophyte has, for some reason, to concentrate near a machine plugged into the system. A precisely shaped well-aimed-at useful, not disruptive, effect can work the other way. "Order More and Chaos Less", SO MASTERING THAT PATTERN COULD SAVE MILLIONS OF DOLLARS IN DOWN TIME. I am sure other useful slogans and incantations can be made to come out of a Speech Synthesizer. Ministers of religion and politicians with laryngitis or perhaps wanting to make a point could do speech synthesis voice tapes. The thing about that is that you can get the exact pronounciation you want even if you have a frog in the throat or whatever.

continued on page 18

Newsletter Update

by Bob Relyea

TIBUG (Brisbane), June, 1991: Editorial; Bits & Pieces by Col Christensen; mention of a 'bug' in EMU; TI-Base Tutorial #5 by Martin Smoley; What's News - MS Express have released more of their popular Sliding Block Puzzles; Mike Maksimik has set up Crystal Software Inc to handle distribution of his MIDI Master 99; Asgard scheduled the release of the Extended Graphics Interface for last month and have released a program called Screen Preview (described as a replacement for the TI-Writer formatter where the output can be sent to the screen in miniature format so that the whole page can be viewed; Trading Post; Posters; The Disk Library; Tips From The Tigercub #30; Multiplan Exercises #2; TML Graphics by Stephen Shaw; TI 99/4 - The Early Days by Chuck Neil; Memex, a Review by John Koloen; The Joy of Ramdisks by Ted Peterson; The "Clock".

August, 1991: Editorial (a boost for our fair!); Transliteration (&Foreign Languages); TI-Base Tutorial #7 by Martin Smoley; Bits and Pieces by Col Christensen; Trading Post; In the P.O. Box; Tips #32; Traffic Cop (program to type in); Multiplan Exercises #6; Treasurer's Report; TI-Writer Graphics by Anne Dhein; Financial Planning One Liners by Tony Falco;

ATICC (Adelaide), July, 1991: Words from the Coordinator; Editorial; Basic Training by Boyd Shugert; Variable Accuracy by Tom Halfhill; TI Cadette: Computer Aided Design by Bradley Rogers; TI Randomness Test; Basic, The DEFinition Statement by Regena; TI Free Memory Techniques; TI Call Key; Glossary of the Computer; TI Disassembler by James Dunn.

September, 1991: Words from the Coordinator; Editorial; Dips Tips; Did you know? (programming tips); Getting your words correct (spell checkers); Translating from other Basics to TI Basic by Jerry Stern; Subscripted Variables (programming tips); Notes on Laser Printing with the TI-99/4A.

TI UP TI t BITS (WESTERN AUSTRALIA), JUNE, 1991: Editor's Comments; The Newsletter Column; The RS-232C (V.24) Serial Data Interface; More Tips on Tips; The Ribbon Re-Inker; Kooky Fortunes by Ed Machonis; Of Modems and Speed; Uncle miltie's Corner; Combining Two Hobbies - Revisited.

SPIRIT OF 99, June, 1991: What's Hot by Irwin Hott;Comments on the Lima Fair by Dick Beery; TI World News by Jim Peterson; TI-99/4A Limitation? by Jan Knapp; New Age/99 #14 by Jack Sughrue; Tex Talker by Ed Machonis; It's Later That You Think by Sister Pat Taylor; The Other Guys vs TI-994A by Lynn Gilbertson; New Age Special by Jack Sughrue (good plug for Funnelweb!); Cassette tricks by Ed Hall and Harold Hoyt; News of Up-coming Funnelweb upgrades.

THE OTTAWA NEWSLETTER, June, 1991: Fast Extended Basic by Lucie Dorais; TI Computing in France, edited by William Gard; Personal Record Keeping Basic Part 0 by Jan Alexandersson, Sweden.

TI-FOCUS, Summer, 1991: News and Views From CH99 by Tom Arnold; Channel 99, a retrospective (long article on the club's history by Tor Hansen; Technology (photocopy from Time magazine, 3 June, 1991; New Age/99 #14 by Jack Sughrue.

THE BOSTON COMPUTER SOCIETY, May, 1991: Listen by Justin Dowling; Language Drills by Don Shorock, reviewed by Jim Peterson; Air Taxi by Don Shorock, reviewed by Jim Peterson; Confidential File, a Rave review by Dave Ratcliffe; Introduction to the UCSD P-System by Ron Williams.

July/August, 1991: Listen; Midimaster update; Notes on Laser Printing with the 99/4A; New Ribbons or Old by Marshal Ellis; Word Processing Commands; TI still cares; Word Processing Tips; Introduction to the USCD P-System; Taking the "Buzz" out of Buzz Words.

UGOC ROM, June, 1991: President's Message by Silas Bazerman; Earl Raguse, Computer Enthusiast by Earl Raguse; How My Subprograms Work by Earl Raguse.

July, 1991: Coming Events; John Birdwell, TI Programmer; Make your own Graph Paper; The Membership; Thoughts for the Day; Long Distance Telephone Companies by Earl Raguse; TI Bits #16 by Jim Swedlow; Computer Glossary for Realists by Earl Raguse; XBasic Miscellany #3 by Earl Raguse; Disk Manager Functions - a Comparison; History Re-Written by Earl Raguse.

August, 1991: President's Message by Siles Bazerman; UGOC Hall of Fame by Bill Nelson; Biographical Sketch of Gene Smith; Pandemonium; The Membership; Auto Computer Music by Earl Raguse; Forth in Real Time Control; Warning, Warning, Warning by Earl Raguse; Programmer's Dilemma by Don Lester; XBasic Miscellany #4 by Earl Raguse; Who Own's The Zebra?; Fibonacci Numbers by Earl Raguse; Computer Music by Earl Raguse.

THE PUG PERIPHERAL, April, 1991: TI-Bits by Jim Swedlow; Call Sound by Dan Eicher; The Kiddie Corner by Sue Harper; New-Age/99 #12 by Jack Sughrue; Tips From The Tigercub #54; Pc Ps in the PEB (part 3) by John F. Willforth.

May, 1991: TI News by Gary Taylor; New-Age/99 #13 by Jack Sughrue; From the Librarian by Sue Harper; The Kiddie Corner by Sue Harper; Programs that write programs by Jim Peterson; Why should you learn to program? by Jim Peterson; Graphing Data with Multiplan; Modems, Part 1 by Al Kinney.

June, 1991: Club News by Gary Taylor; Modems Part 1b by Al Kinney; Programs That Write Programs, Part 2 by Jim Peterson; New-Age/99 by Jack Sughrue #14; PIXEASE from Comprodine by BJ Mathis; Tinycat-A Tinygram by Ed Machonis; Tips #55 by Jim Peterson.

TIdbits, June, 1991: President's Bit and In The News by Gary Cox; Buying a Modem by Gary Cox; New Ribbons for Old by Marshall H. Ellis; a list of TI suppliers; Notes on Laser Printing with the TI-99/4A by Larry Fairbanks; DEZIP - Utility Software for the TI-99/4A by Rich Link.

August, 1991: Presidents' Bit; In the News; The Altmam Fairware List; Modem Use, Part 3; Computing, The Future; Real Programming; Memory Savers; Editor's Bit.

LA TOPICS, July, 1991: Tips on Funnelweb 4.31;Crackerbarrel by Chick De Marti; Computer Controlled Robot; You're Not Getting Older, You're Getting Better; Hints in IBM Conversion; Financial Planning One LIners by Tony Falco; Mail List Manager by Bill Gaskill; Program to type in; The Editor's Column.

August, 1991: Thoughts From The President; Crackerbarrel by Chick De Marti; XBasic 2 by Earl Raguse; Printer Codes by A Cassidy; Market Place by Fred Moore; Editor's Column - Is the TI 994a Extinct?

LEHIGH 99'ER COMPUTER GROUP, July, 1991: President's Message; The Broken Key; New Info Disk Available; Statistical Analysis; Some Notes on Video by Delbert Wright.

August, 1991: The Broken Key; Second Issue of SOS On-Line; New Info Disk Available; Statistical Analysis by David Strachin; Some Notes on Video. o

continued from page 20

```
620 PRINT "PRESS YOUR GUESS"
630 CALL KEY(0,A,B)
640 IF B=0 THEN 630
650 IF A=66 THEN RIGHT=RIGHT+1 ELSE WRONG=WRONG+1
660 IF B=66 THEN PRINT "WELL DONE, RIGHT" ELSE PRINT
    "SORRY, WRONG"
670 IF RIGHT=0 THEN A$="POOR SHOW.0/5"
680 IF RIGHT=1 THEN A$="NOT YOUR OCCUPATION.1/5"
690 IF RIGHT=2 THEN A$"PLENTY OF ROOM FOR
    IMPROVEMENT.2/4"
700 IF RIGHT=3 THEN A$="NEARLY THERE.3/4"
710 IF RIGHT=4 THEN A$="PERFECT.4/4"
720 CALL CLEAR
730 PRINT "YOUR SCORE IS";RIGHT;" RIGHT AND ";WRONG;"
    WRONG"
740 PRINT A$
750 INPUT "WOULD YOU LIKE ANOTHER GO(Y/N)":AS$
760 IF AS$="Y" THEN 120
770 END
```

o

Regional Group Reports

Meeting Summary For November

Banana Coast	10/11/91	Sawtell
Central Coast	09/11/91	Saratoga
Glebe	07/11/91	Glebe
Hunter Valley	09/11/91	Boolaroo
Illawarra	11/11/91	Keiraville
Liverpool	08/11/91	
Northern Suburbs	28/11/91	
Sutherland	15/11/91	Jannali

BANANA COAST Regional Group
(Coffs Harbour Environs)

Regular meetings which are in our 6th year are held on the second Sunday of the month at 2 pm sharp. (There is a new venue but the message I got from the BBS was garbled so please contact one of the names & numbers below for accurate details. ED). Kevin Cox, 7 Dewing Close, Bayldon, telephone (066)53 2649. You could also contact John Ryan of Mullaway via the BBS, user name SARA, or telephone (066)54 1451, as well as Rex De-Mouilpied via the BBS, user name REX, or telephone (066)51 2485. All visitors are most welcome.

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.

HUNTER VALLEY Regional Group

The first regional meeting of the group will be held at the Boolaroo Ambulance Station to discuss future plans. All welcome. Please contact Geoff Phillips on (049) 428 176 for details.

ILLAWARRA Regional Group

Regular meetings are normally held on the second Monday of each month after the TIsHUG Sydney meeting, except January, at 7.30pm, Keiraville Public School, Gipps Rd, Keiraville, opposite the Keiraville shopping centre. A variety of activities accompany our meetings, including Word Processing, Spreadsheets and hardware repairs. 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 REPORT

Our September meeting saw Herbert Shade's ramdisk spring to life, following a few anxious moments during the previous week. All problems were solved with the installation of a new ROS and the cleaning of a troublesome Extended Basic module.

Kevin Taylor brought along a misbehaving disk system which was still causing problems. Geoff Trott had previously checked out the disk controller card which tested OK. The problem later turned out to be in the drive itself.

After losing a number of floppies due to this drive, Kevin has decided to cut his losses and invest in a new drive through the club shop.

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.

NOVEMBER MEETING - 2ND NOVEMBER

The November meeting will be a full-day tutorial, which is the second and last one for the year. The wide range of activities and tutorials available are listed below. We hope you find one that interests you.

* Multiplan	-	Geoff Trott
* Tips	-	Alf Ruggeri
* Modems	-	Peter Mudie
* Console Repairs	-	Geoff Trott
* Word Processing	-	Percy Harrison
* Extended Basic Tips	-	Bob Relyea
* Educational Software	-	Dick Warburton
* Eprom Ramdisk Simplified	-	Dick Warburton

****

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

December	10 November
February	12 January

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.

****

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.

For Sale

2 TI CONSOLES (1 hardly used)
1 TI EXPANSION BOX (with a quiet fan)
1 RS232 CARD
1 TI DISK CONTROLLER CARD
1 TI MEMORY EXPANSION
1 HORIZON RAM DISK DSSD
1 FLOPPY DRIVE DSSD (in expansion box)
1 WANG COLOUR MONITOR with interface modules
2 EXTENDED BASIC
1 EDITOR/ASSEMBLER with manual
1 HC99 BK Cartridge
1 DISK MANAGER
1 VIDEO CHESS
1 RETURN TO PIRATES ISLE
1 OTHELLO
1 PARSEC
1 MUNCH MAN

I have all original manuals plus TI Writer manual, as well as disks with all popular programmes. Also, I have a complete set of HV99er magazines, and all TIsHUG Digests from November, 1984.

I will accept best offer around $600 with shipping costs to be negotiated.

Contact Barry Ridgeway,
BEECHWORTH, Victoria
Ph 008 032264