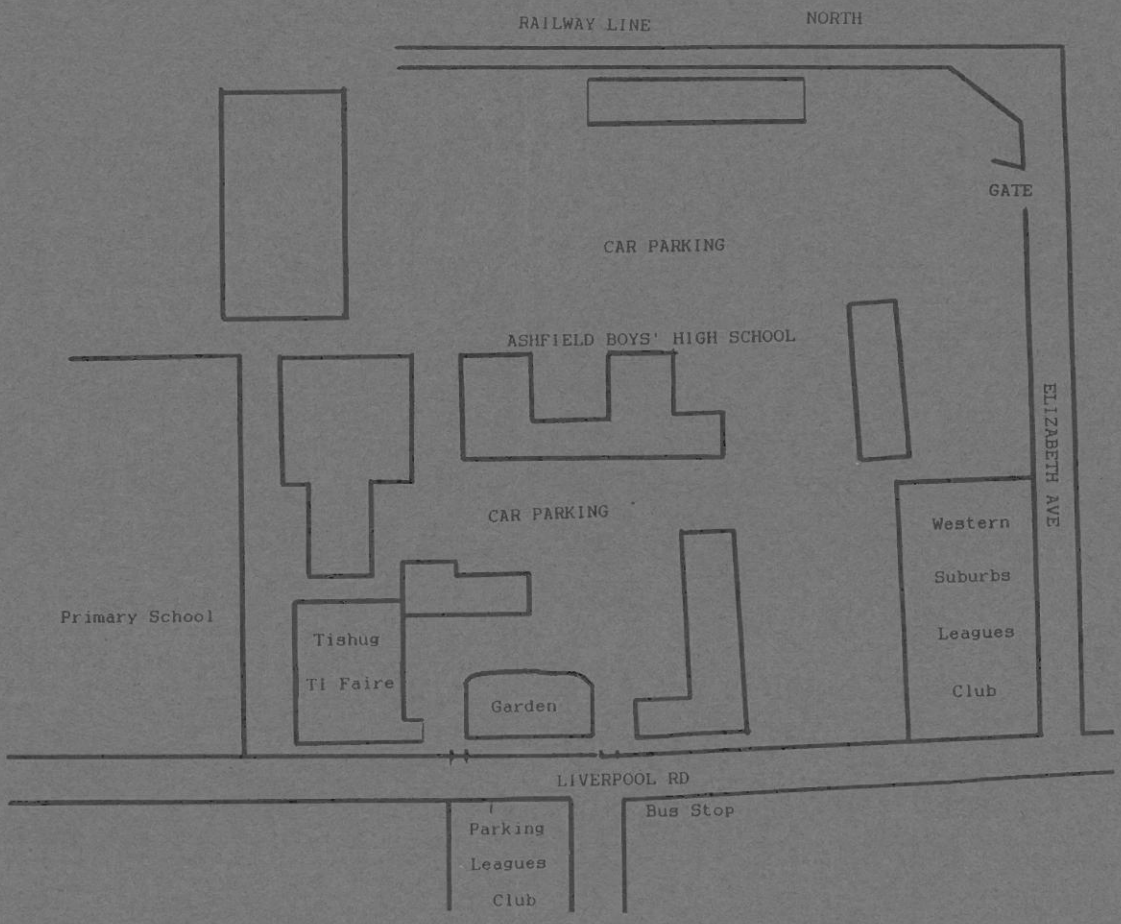
TIsHUG NEWS DIGEST

Focusing on the TI99/4A Home Computer

Volume 11, Number 10 November, 1992

Registered by Australia Post - Publication No. NBH5933

TIsHUG's TI-Faire



Saturday and Sunday, 28th and 29th November

Sydney, New South Wales, Australia **$3**

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

TIsHUG News Digest

November 1992

All correspondence to:

P.O. Box 1089
Strawberry Hills, NSW 2012
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) 85 5519
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
Russell Welham (043) 92 4000
Software library
Larry Saunders (02) 644 7377
Technical co-ordinator
Geoff Trott (042) 29 6629
TI-Faire co-ordinator
Dick Warburton (02) 918 8132

Regional Group Contacts

Central Coast
Russell Welham (043) 92 4000
Coffs Harbour
Kevin Cox (066) 53 2649
Glebe
Mike Slattery (02) 692 8162
Hunter Valley
Geoff Phillips (049) 42 8176
Illawarra
Geoff Trott (042) 29 6629
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	$35.00
Associate membership	$10.00
Overseas Airmail Dues	A$65.00
Overseas Surface Mail Dues	A$50.00

TIsHUG Sydney Meeting

The November Meeting will start at 2.00 pm on 7th of November at Ryde Infant School, Tucker Street, Ryde. There will be an Assembler Class before the meeting starting at 10.00 am.

Printed by
The University of Wollongong
Printery Services.

TIsHUG News Digest ISSN 0819-1984

Index

Wanted...
Members contributions to the BBS and the TIsHUG News Digest. Everyone can help, please do!

Editor's Comment

by Bob Relyea

This is the big month for the faire. I trust that all the work put into it will bear fruit and that, most of all, we will have a good time fellowshipping with ourselves and other TIers and learn something. I trust that the change-over to a new magazine team will be a smooth one as Geoff and Rolf, who have certainly done their 'whack', are passing on the baton to others. In the 11 years that the club has been operating, there have been only five (5!!) users who have edited or done the paste-up, etc for the magazine. It is about time a few others got involved in this way. I am not talking about people like Ross, for instance, who have done heaps in other capacities but those who have been 'in the shadows' waiting for their chance to make a contribution, even if it is only for a year or two. There are a lot of interesting things you can learn as well as keeping our magazine going. I trust that by the time this is read that volunteers have come forward to answer Dick's call. o

Notice to Members

All members are advised that the 6th Annual General Meeting of TIsHUG (Australia) Limited will be held on Saturday, 5th December, 1992 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 that 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, 14th November, 1992, while in accordance with paragraph 17(b) a suitable nomination form is enclosed.

Terry Phillips (Honorary Secretary) o

TIsHUG (Australia) Limited
Sixth Annual General Meeting
Saturday 5th December, 1992
Ryde Infants School, Tucker Street, Ryde NSW

Agenda

1. Meeting opening.
2. Members present and apologies.
3. Reading and confirmation of minutes of the 5th Annual General Meeting held on Saturday 7th December 1991 and reading and confirmation of minutes of Special General Meeting held on Saturday 6th June 1992.
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 Auditor's Report.
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 Meeting. o

Secretary's Notebook

by Terry Phillips

It seems that most members may have been out of town over the long weekend as there was not a very large turn-out for the October meeting. Still, those that did come along had an enjoyable, if informal meeting.

By the time you are reading this there will be two important events on the horizon:

1. The TI-Faire is just around the corner - 28th and 29th of November and on the Saturday night we will be having a social mixer. The venue will be the Western Suburbs Leagues Club - Stars Carvery and Grill. A number of seats will be set aside for those who wish to attend from 6.30pm onwards. Meals at very reasonable prices, $3 for the daily special plus grills and other menu items at about $5 to $6, are available up to 9pm. Drinks are available, at club prices. Under 18's are welcome in the bistro but are not allowed into the poker machine and bar areas. Normal club dress rules apply. This is probably one of the few opportunities available for members to mix socially so come along if you can.

2. The AGM will be held on the Saturday following the TI-Faire, on the 5th December. In this issue is the notification of the AGM plus an enclosed nomination form which is suitable for nominating a member for the office of Director. Remember, any nomination must be seconded and the nominee has to sign to signify his/her acceptance of the nomination.

In the past few years ballots have not been required for director's positions due to the fact that only the required number of nominations, 5, have been received at the closing date. Will this year be the same?

It is now 9 years since TI, in their wisdom, pulled the plug on our computer. In that period there have been marvellous advancements in TI99/4A hardware and software that probably none of us would have imagined possible back in 1983. But the cold hard facts are that membership is on the decline, not only in our club, but throughout the TI99/4A world. I believe that the major problem which will face the directors, and members, during 1993 is how to go about keeping, and possibly, increasing a membership base. Like it or not there are other computer systems out there and quite likely the majority of members either own or use some other system. Should we then start to cater for these other systems? Time, I guess, will tell.

See you at the November meeting. o

TI-Faire News

Meals will be available on the two days of the Faire at the Western Suburbs Leagues Club, next door to the school, at the following times.

Saturday:	12.00 to 2.30pm	#2 Lunch Special
	5.30 to 9.00pm	#3 Dinner Special
Sunday:	12.00 to 2.30pm	#2 Lunch Special
	5.30 to 8.30pm	#3 Dinner Special

Accomodation for TI-Faire

Motels

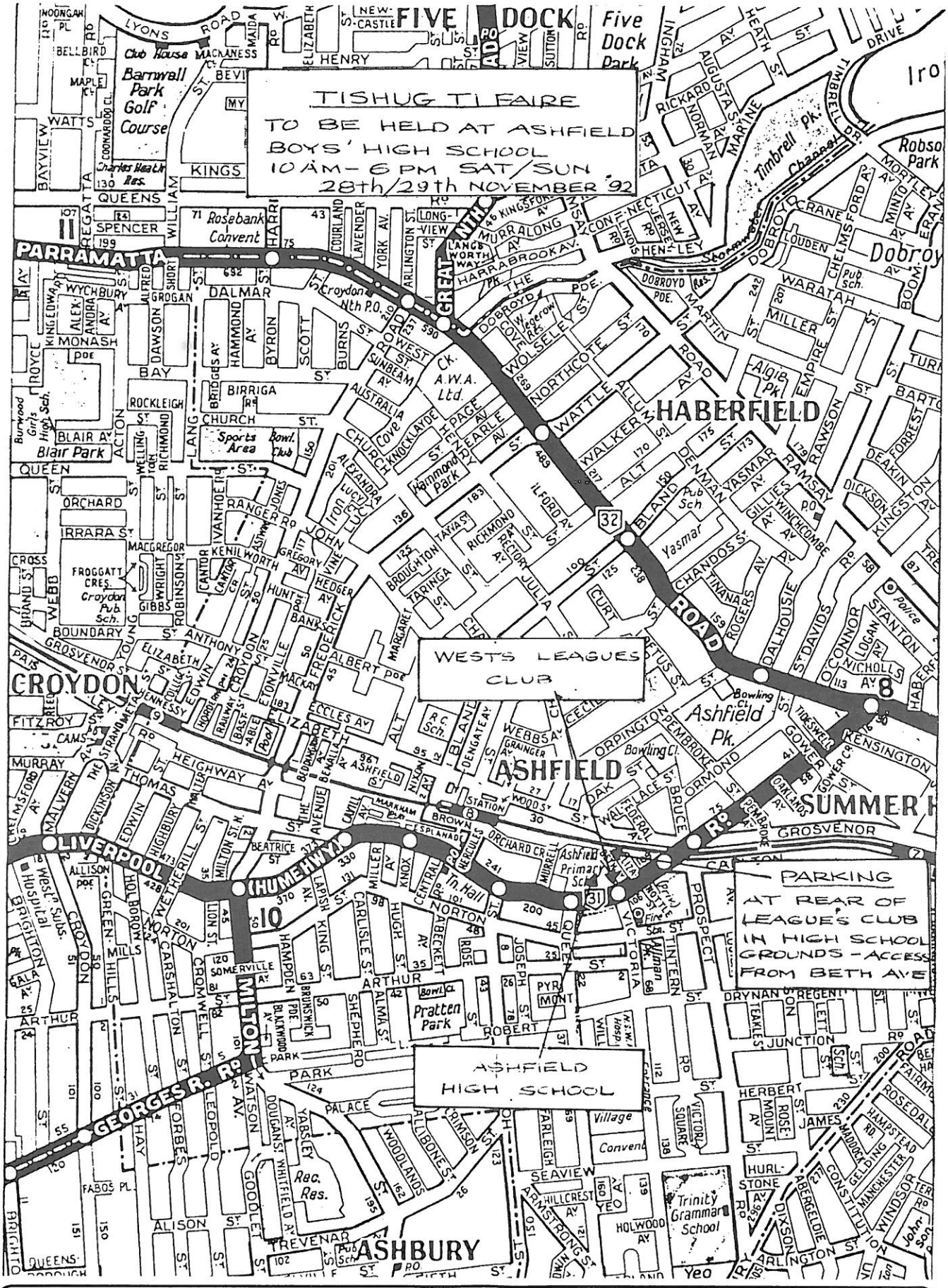
**** Palm Court (M): 17 Parramatta Road, Haberfield
Room only, single $78 to $88; 02-797 6111

*** Metro Motor Inn (M): 63 Liverpool Road, Ashfield
Room only, single $89 to $95; 02-798 0333

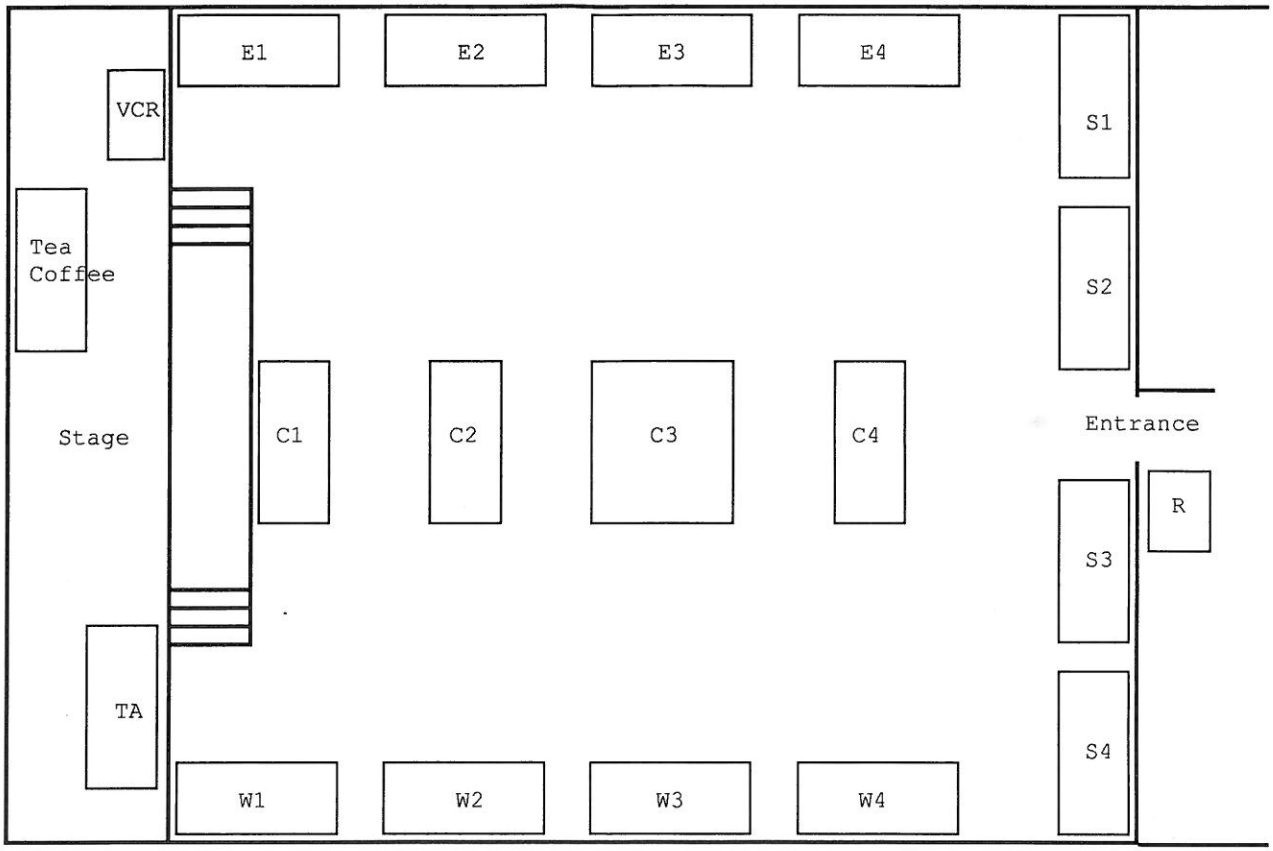
*** Philip Lodge (M): 156 Parramatta Road, Ashfield
Room only, single $78 to $88; 02-798 7666

** Charlotte Motor Inn: 62 Charlotte Street, Ashfield
Bed breakfast, single $70, double $80; 02-798 8918

TI-Faire Map Reference



TI-Faire Hall Layout



Stand Identification	Attendants
E1 TIsHUG Shop	Percy Harrison Alf Culloden
E2 Miscellaneous Software Demonstrations	Peter Mullins Clyde Thornton
E3 Page Pro Demonstrations	Larry Saunders
E4 Greeting Card Production	Alf Ruggeri
S1 Ribbon Re-inking	Robert Peverill
S2 Fully updated TI99/4A System	Ben von Takach
S3 TI-Midi Music System	Arto Heino Gary Wilson
S4 Adventure Games	Vince Maker
W1 Music Maker	Russell Welham
W2 80 Column card, Funnelweb Demonstration	Eric Ockenden Chris McCarthy Bob Relyea

Stand Identification	Attendants
W3 RAMdisk/EPROM Demonstration	Dick Warburton Mike Slattery
W4 Australian Designed Hardware Demonstration	Peter Schubert
C1 Cadet Console Expansion System	Col Christensen Garry Christensen
C2 Computer Control of Model Train	Ross Mudie Peter Mudie
C3 Sale of Surplus TI99/4A Hardware and Software	Rolf Schreiber
C4 Unique Items Display	Bruce Boese
TA Technical Advice	Geoff Trott Lou Amadio Tony McGovern Will McGovern
R Reception	Tom Marshall Jack Scott Jim Mable Peter Young

TIsHUG (Australia) Limited

Minutes of the Fifth Annual General Meeting

Held on Saturday 7th December, 1992
Ryde Infants School, Tucker Street, Ryde NSW

1. Opening:

   Chairman Dick Warburton opened the meeting at 2.05pm welcoming those members present.

2. Members present and apologies:

   Present - 33 (highlighted on attached membership list). Several other members arrived while the meeting was in progress.

   Apologies were recorded and accepted from -

   Larry Saunders, Michael Ball, Lou Amadio, Peter Mullins, Ashley Lynn, Eric Ockenden and Gordon Smith.

3. Reading and confirmation of minutes of 4th Annual General Meeting held on Saturday 1st December, 1990

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

   Moved Tom Marshall, seconded Percy Harrison that they be confirmed. Motion carried.

4. Correspondence and dealing with same:

   Secretary Terry Phillips advised that there were two items of correspondence:

   a) Letter from Warren Welham advising he could not continue as Publications Librarian due to being selected for Youth Exchange Programme and he would be spending next 12 months in Japan.

   b) Letter from Geoff Warner (Perth UG) giving support to TI-Faire.

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' Report, presentation of accounts and Auditor's Report:

   The Directors report, Accounts and Auditors report had been included as a supplement in the November TND. Reports had been mailed to all members.

   These reports were accepted on the motion of Don Gould, seconded Tony Bell.

7. Unfinished business from last AGM (if any):

   Nil.

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. 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 Percy Harrison, seconded Tom Marshall. Carried.

    John Robinson queried the fees charged by the Auditor - these were advised to him by Geoff Trott.

    John Robinson raised Taxation issues - he advised that the club he was in did not have to pay income tax. John is to supply contact details in ATO so that appropriate enquiries can be made.

11. New Business (if any):

    a) Dick Warburton outlined to members that membership fees would rise to $35 from the 1st January 1992. Concurrent with this, the present $5 BBS access fee would be abolished. Reasons cited for the increase were increasing costs with production and mailing of TND, declining membership base and in the case of the BBS fee being dropped, to encourage more members to use this facility.

    b) Peter Schubert queried high expenses shown in December TND - Geoff Trott explained this was outgoings for purchase of TIM Cards.

    c) John Robinson queried how advanced was planning for the TI-Faire - Dick Warburton explained that a committee had been formed and he invited others who may be interested to join this committee. Ian Mullins gave a run-down on places he had contacted and their pricing structure.

    d) Ian Mullins advised that judging would take place after the AGM on entries received in the BBS competition.

12. Meeting closure:

    There being no further business to discuss, Chairman Dick Warburton closed the meeting at 2.45pm advising members that the Christmas Party food would be served shortly.

Minutes recorded by:

Terry Phillips
Honorary Secretary

7 December, 1991

Minutes confirmed by:

Richard Warburton
Chairman

/ /

o

TIsHUG (Australia) Limited

Minutes of the Special General Meeting

Held on Saturday 6th June, 1992
Ryde Infants School, Tucker Street, Ryde NSW

1. Opening:

   Chairman Dick Warburton opened the meeting at 3.05pm.

2. Members present:

   34 members were present to vote on the proposed amendment to Clause 4.

3. Business:

   Geoff Trott outlined the reasons for calling the special meeting. Advice from the Australian Securities Commission (ASC) had been received to the effect that our club was liable for an annual $600 fee. However, if the proposed amendment as promulgated in the May 1992 NewsDigest was approved, then the annual fee would be reduced to $30.

continued on page 9

Rambles

by Stephen Shaw, England

One heart-felt cry, from more than one member, involves the printing of text files using TI Writer. TI Writer NORMALLY send a carriage return and line feed at the end of each line, and life is much easier if you switch the internal dip switches in your printer so that your printer does not itself add an automatic carriage return and line feed at the end of each line! If both the printer and TI Writer add a line feed, you end up with double spacing, whether you want it or no.

The TI RS232 card contains a number of software switches, and if you use RS232 as your printer name you will certainly know all about those! However PIO users generally have little call to use them and so remain unaware!

You can instead of using just PIO, name your printer as PIO.CR or PIO.LF - if you use the Formatter, you may NEED to use PIO.LF even if PIO on its own is OK when printing from the Editor. If you ever use a Graphics program, you will need to use PIO.CR to prevent an automatic carriage return every 80 characters- something that can make your graphics look a little untidy. In case of difficulty with line feeds, go through all the possible printer names and use the one that is best, be it PIO.CR or PIO.LF or possibly even PIO.CR.LF It makes life easier if you can switch the auto line feed off at the printer- consult your printer manual.

Another problem is that several members use printers which are not 100% compatible with the Epson control codes, and when printing text which has these embedded, you may meet problems, such as a printer freeze. Again, TI Writer has been written to take care of this problem. You can instruct TI Writer to print the text file but to strip out the control codes- that is to print only the "printable" characters ASCII 32 to 126- to do this, instead of using the output device name of PIO you use C PIO - that is, a capital C followed by a space followed by the normal printer name.

And as a reminder, if you want TI Writer to save in DF80 format instead of DV80, select PF then type F DSK1.FILENAME- use the print file command instead of the usual save file, and add an F and a space in front of the output device name. You remember of course that TI Writer can load both DF80 and DV80 files! In fact it will even handle DF and DV files longer than 80, but only load the first 80 characters of each record. TI Writer is an unusually well written program! And remarks regarding TI Writer also hold true for Funnelweb!

Reminder to PR Base users about the bugs in that database:

1. Output Device Name:

* When you initialise a disk (any disk) the sectors are filled with >E5's.

* When you select CREATE using PRBASE, the default output device name appears, IF you have nominated one previously. Otherwise you will see an apparently blank input field. Not so- what you cannot see is an input field full of >E5's, so if you type "PIO", the program will then try to use a device called PIO followed by several CHR$(229)'s and tell you that you do not have any such device!

This is mainly a problem when using PIO, as RS232 users usually make use of the cards software switches and put a full stop in there someplace- the card is so programmed that the excess >E5s are then ignored.

PIO users should type PIO and then fill the input field with blank spaces by holding space bar down! OR first use ERASE (FCTN 3) before entering the printer name.

2. Bill Warren is aware that some users with TI disk controllers have been unable to format a double sided disk using the PR Base utility. Their solution is simple- use another disk manager to format your disks! [I suspect that Bill's problem may have been with the stand-alone disk controllers which are for single sided use only?]

3. Mis-Alignment of columns in TAB reports:

This time due to >00's.! When you type the header, do not use FCTN D to move to the right- it will leave >00's behind. You MUST type spaces with the space bar. Bill does not fill the screen with spaces, he used nulls!

Note: Bill is NOT working on updates to PR Base. Version 2.0 is his last. However Mike Dodd has written a version 2.1 which makes normal use of sectors 0 and 1, but is incompatible with V 2.0 data files. Both versions are available from the disk library.

Just as I thought the supply of new software was declining, along comes the postman with a small package with three innocent looking disks inside- containing -in archived format- a bit more than 3000 sectors of software catalogue- lots that we have, some duplications, some we have no interest in, and one or two (ha!) items that we have been looking for for years and years, so watch out for some nice new goodies in the coming quarter!

One disk library user had a problem using HBMPRINT, a utility to print the data from Home Budget Manager in various formats- it would only load with Editor Assembler. Now- this program was in Forth, which if you recall Versions 3.x of Funnelweb, had to be flagged K=60 in the Funnelweb load menu. That menu is no longer available, and it would appear that Funnelweb is no longer able to produce that particular load environment. No problem! Some time ago, the peculiar original TI Forth Load requirement was removed, and an amended format produced which would load as a standard Load and Run file- you can find it on the MiniMemory version of Forth in the disk library. All I had to do was swap the load routine and now HBMPRINT loads fine with Funnelweb.

Hands up anyone who remembers the ADAM computer! It was on sale in this country for about a week I think! It was a Z80 based micro- like the ZX80 and ZX81. It appears that some surplus Adam PCB's duly found their way into some TI PEBs over in the states, providing a 2nd (Z80) processor! I shall try to find out a little more, it seems to have been a well kept secret... maybe we can do something with ZX80's!

o

Stephen Shaw

continued from page 19

```
char *str;
{
  int sgn;
  str = str + sz;
  *str = '\0';
  while (0 < --sz) {
    sgn = nbr & 15;
    if (sgn > 9)
      sgn = sgn + 7;
    *--str = sgn + '0';
    if (!(nbr = nbr >> 4))
      break;
    nbr = nbr & 4095;
  }
  while (0 < --sz)
    *--str = ' ';
  return str;
}

* Script-Load for loading and running c99 files

      AUTO
      FILE "DSK1.FIND;O"
      FILE "DSK1.CSUP"
      FILE "DSK1.TCIO;O"
      LAST START
```

o

TIsHUG Shop with Percy Harrison

As the TI-Faire is to be held at the end of this month, I have decided that it would be an opportune time to list most of the items that we have for sale in this months TND in the hope that we may move quite a lot of the stock that we have been holding for some time.

The sale of commercial software has been somewhat disappointing although we have priced it very competitively with overseas prices. Some of these programs have been in stock for twelve months and if we do not make any sales in November, either at the club meeting or at the Faire I am afraid that we will have to remove them from our stocks as it is pointless having to carry them to and from the meeting each month when no sales are made. Because of the poor response to this commercial software we will have to be much more selective in future purchases of such software.

Now on a different topic, we have had quite an influx of new members over the last six to twelve months, all of whom have only a basic system with very little knowledge of the full capabilities of the TI99/4A, so at the last meeting of the Directors I requested that we devote considerably more space in our magazine for programs and tutorials that would assist our new members get the most out of their machine and hopefully keep them interested in the club and its activities. This proposal was well received by the Directors and we can expect to see an increase in articles slanted towards the new members within the next couple of months. To help us achieve this it would be greately appreciated if some of our long standing members would take a little time to prepare articles which would be suitable for the newer members.

Price List

Commercial Software

Artoons SSSD $12
BABA Brewery Beer Labels SSSD $10
Bride of Disk of Dinosaurs SSSD $14
Character Set & Graphic Design Cataloguer SSSD ..$6
Character Set & Graphic Design I SSSD $12
Character Set & Graphic Design II SSSD $10
Character Set & Graphic Design III SSSD $14
Disk Utilities (Memorial Edition) DSSD $11
Disk Utilities (Memorial Edition) SSSD $12
Disk of Dinosaurs SSSD $10
Disk of Horrors SSSD $14
Disk of Pyrates SSSD $12
Display Master SSSD $15
Edu-pak Module + Book $25
FilmLib Vers 3.0 (TI-Base) SSSD $8
Fonts and Borders I SSSD $8
Fonts and Borders II SSSD $8
Fonts and Borders III SSSD $10
Fonts and Borders IV SSSD $8
Genial Traveler SSSD $6
GIF-Mania SSSD $15
Legends (2 Disk Set) SSSD $30
McPaint (5 Disk Set)-DSDD $10
McPaint (10 Disk Set)-DSSD $20
Microdex 1 SSSD $16
Microdex II SSSD $11
Nuts and Bolts #1 DSSD $6
Nuts and Bolts #1 SSSD $7
Page Pro Applications #1 SSSD $2
Page Pro Line Fonts SSSD $9
Page Pro Medical Clipart-DSDD $10
Page Pro Medical Clipart-DSSD $13
Page Pro Templates Vol1-SSSD $8
Page Pro Templates Vol3-SSSD $8
Page Pro Utilities SSSD $17
Picasso Publisher Version 2.0 SSSD $14
Picasso Publisher Support Disk SSSD $6
Picasso Applications Disk DSSD $2
Rockrunner SSSD $15
Screen Preview SSSD $20
Smart Connect SSSD $15
Son of Disk of Dinosaurs SSSD $12
Spell It! (DSDD version) $24
Spell It! (SSSD version) $27
Star Trek (Calender) DSDD $14
The Missing Link Companion Disk SSSD $2
The Ring Companion SSSD $12
TI Casino SSSD $16
Word Processor Harrison Software SSSD $10
X Basher SSSD $15
XB : Bug SSSD $22
Typewriter Module $25

Club Software Disks

A119 J P Drawing Vers 3.0 DSSD $2
A145 Adventures (Scott Adams) DSSD $2
A214 Plus Vers 1.0 SSSD $2
A245 Telco Vers 2.3 DSSD $2
A261 Assembly Language Games SSSD $2
A380 Super-Cataloger SSSD $2
A382 Boot (40 Column Vers) SSSD $2
A386 Boot (Hard Disk Vers) SSSD $2
A401 Pix Version 1.2 SSSD $2
A411 Miner/49er, Espial SSSD $2
A417 TI Utilities #2 SSSD $2
A420 Atari Games #1 SSSD $2
A430 Configuring Funnelweb SSSD $2
A431 Object Linker Vers 3.0 SSSD $2
A438 More Assembly Games SSSD $2
A439 Multiplan Exercises DSSD $2
A448 Tips Vers 1.7 SSSD $2
A448A Tips Graphics #1 SSSD $2
A448B Grips (Tips Companion) SSSD $2
A450 Funnelweb 4.40 DSDD $2
A450A Funnelweb 4.40 (3 Disks) SSSD $4
A451 Multiplan Vers 4.02 SSSD $2
A453 The Nutcracker Suite SSSD $2
A456 Rememberance Disk (Music) SSSD $2
A457 Anna Magdalene Music SSSD $2
A462 Rediskit SSSD $2
A463 TI-Exam SSSD $2
A464 Il Pastor Fido Vivaldi DSSD $2
A465 Lute Music SSSD $2
A466 Best of DOM #5 DSSD $2
A467 The Singing TI SSSD $2
A468 Speech #1 SSSD $2
A472 TI Writer Supplement SSSD $2
A473 DM 1000 Version 5.0 SSSD $2
A474 GIF Pictures (80 column card) DSDD $2
A475 Clubline 99 Vol 4 No 8 SSSD $2
A476 Clubline 99 Vol 5 No.5 SSSD $2
A481 Artconvert DSSD $2
A481A Artconvert (2 Disk Set) SSSD $3
A482 Horizon Utilities SSSD $2
A483 TI Tiler SSSD $2
A484 Mac-labels SSSD $2
A485 YEO SSSD $2
A486 Genealogy Record Keeping DSSD $2
A487 XHI (80 column card) DSDD $2
A488 Utilitied LA 99ers SSSD $2
A489 Fontart #1 SSSD $.....$2
A490 Fontart #2 and #3 DSSD $2
A491 Designing Graphic Screens SSSD $2
A492 Microkey SSSD $2
A493 Disk Manager 99 Vers 2.0 SSSD $2
A494 XB #0 Money Money SSSD $2
A495 Directory SSSD $2
A504 The Director SSSD $2
A505 Sorting DSSD $2
A506 Memory Manager SSSD $2
A507 Implanting SSSD $2
A508 Booklet SSSD $2

TCC1 Tigercub Collection #1 SSSD $2
TCC2 Tigercub Collection #2 SSSD $2
TCC3 Tigercub Collection #3 SSSD $2
TCC4 Tigercub Collection #4 SSSD $2
TCC5 Tigercub Collection #5 SSSD $2
TCC6 Tigercub Collection #6 SSSD $2
TCC7 Tigercub Collection #7 SSSD $2
TCC8 Tigercub Collection #8 SSSD $2
TCC9 Tigercub Collection #9 SSSD $2
TCC10 Tigercub Collection #10 SSSD $2
TCC11 Tigercub Collection #11 SSSD $2

continued on page 9

TIsHUG Software Column

by Rolf Schreiber

This will be my last software column. I have found it increasingly difficult to devote the necessary time to the task, so I have finally decided to call it a day. As of next month, Larry Saunders will be taking over all the resposibilities of Software Coordinator.

Software Releases for November 1992

DISK A417 contains three TI99/4A utility modules that were transferred to disk using the GPL Linker. The modules include Household Budget Management, Tax/Investment Record Keeper and Personal Real Estate. The disk is menu driven and runs out of Extended BASIC, or individual programs can be directly loaded from the Editor Assembler module.

A417 Diskname: UTM/2 Format: SSSD

Filename	Size	Type / Length	
BUDGETMNG1	13	PROGRAM	3072
BUDGETMNG2	33	PROGRAM	8192
BUDGETMNG3	18	PROGRAM	4108
INVE	29	PROGRAM	7168
INVF	23	PROGRAM	5632
INVG	25	PROGRAM	6144
INVH	17	PROGRAM	4096
INVI	17	PROGRAM	4096
LOAD	9	PROGRAM	2025
REALESTAT1	13	PROGRAM	3072
REALESTAT2	33	PROGRAM	8192
REALESTAT3	33	PROGRAM	8192
REALESTAT4	33	PROGRAM	8192

DISK A432 is MM UTILITY V1 from R. A. Green. The disk comes with source code and object code, in memory image and DIS/FIX 80 formats, and a documentation file.

A432 Diskname: MMUTILITY Format: SSSD

Filename	Size	Type / Length	
DOC/MMUTL	21	DIS/VAR	80
MMUTILITY	12	PROGRAM	2576
OBJ/MMUTL	30	DIS/FIX	80
SRC/MMUTL	87	DIS/VAR	80

DISK A483 is the latest version (V3.0) of Col Christensen's TI-Tiler, which we released as Club software earlier this year. This new version is a great improvment over the original version, which was only a beta release.

A483 Diskname: TI-TILER Format: SSSD

Filename	Size	Type / Length	
DOCS/TILER	69	DIS/VAR	80
ELECDEMO_T	23	PROGRAM	5406
GRAFDEMO_T	23	PROGRAM	5406
OL	33	PROGRAM	8192
OM	10	PROGRAM	2196
OVERLAY_T	23	PROGRAM	5406
T1EL_T	9	PROGRAM	2039
T2EL_T	9	PROGRAM	2039
T3GR_T	9	PROGRAM	2039
TILESETS_T	23	PROGRAM	5406
TT	33	PROGRAM	8192
TU	10	PROGRAM	2196

DISK A509 is Ken Gilliland's 1991 Girlie Calendar. The artwork was all hand drawn and can be edited using TI-Writer or Funnelweb. It should be fairly easy to update the calendar part so that it will be suitable to use in 1993, whilst retaining the existing artwork. The disk is DSDD and the pictures must be printed out in order to view them.

A509 Diskname: GIRLIE'91 Format: DSDD

Filename	Size	Type / Length	
!READTHIS	15	DIS/VAR	80
00/91	90	DIS/VAR	80
01/91	88	DIS/VAR	80
02/91	88	DIS/VAR	80
03/91	88	DIS/VAR	80
04/91	88	DIS/VAR	80
05/91	88	DIS/VAR	80
06/91	88	DIS/VAR	80
07/91	88	DIS/VAR	80
08/91	88	DIS/VAR	80
09/91	88	DIS/VAR	80
10/91	88	DIS/VAR	80
11/91	88	DIS/VAR	80
12/91	88	DIS/VAR	80
CONFIG	2	DIS/VAR	80
DSKLABEL_P	25	PROGRAM	6144
LOAD	27	PROGRAM	6429
PICTURE_C	25	PROGRAM	6144
PICTURE_P	25	PROGRAM	6144

Tigercub Releases for November 1992

TCC-11 is the eleventh disk in the Tigercub Collection from Jim Peterson. The following programs can be selected from the menu:

1 Butterfly and Flowers
2 Tirkle
3 Slinky
4 Kid Stuff
5 Pot of Gold
6 Wawaland
7 Kwik Draw
8 Marksman
9 Miss Muffet
10 Automatic Mouse Maze
11 Pi-Ring Squad
12 Ribbit
13 School Daze
14 Shape Art
15 Turtle Hop

TC-1102 is a disk of Sorts, Scrambles and Searches. The following items can be selected from a menu:

1 2-Dimensional Sort
2 2-Dimensional Swap Sort
3 Binary Search Demo
4 Bubble Sort
5 Disksort
6 Easy Sort
7 Heap Sort
8 High Scramble
9 Insert Sort
10 Integer Flag Sort
11 Jeb's Sort
12 Long Shell Sort
13 Peek Scramble
14 Quick Scramble
15 Quick Sort
16 Relative File Sort
17 Resort Sort
18 Selection Sort
19 Shaker Sort
20 Short Shell Sort
21 Shuttle Sort
22 Simulated Card Shuffle
23 Sort Demo
24 Sort Demo #2
25 Sorts
26 Tigercub Sortwatcher
27 Swap Sort
28 TI-Writer Text Sort
29 Table Sort
30 Wazzit? Sort
31 Assembly Sort Demo
32 TI-Writer Chartbase Sort
33 Shell-Metzner Assembly

continued on page 23

The Communicators

by Ross Mudie

Over the last few months I have been rather busy with work and involvement with several other events where I have provided public address and/or communications, etc. I was feeling guilty over the lack of time that I could devote to the BBS Sysop duties. As a little more time became available, I decided to review the usage patterns of the BBS to see where I should devote my time for the maximum benefit of users. The BBS has 3 areas which require maintenance by the Sysop, Programs, Information Files and the All Mail File. The All Mail file has been maintained fairly regularly and does not present a problem.

The Program download area over the 10 week period reviewed was accessed by only 7 users out of 19 who logged on. Two users downloaded 41 programs. The other 5 users downloaded an average of 2 programs each.

13 users out of the 19 read the ALL MAIL file an average of 5 times.

The 19 users read an average of 12 NEWS files each.

No user has uploaded a program to the user Program Upload/ Download area for a year.

There has been a note on the BBS asking for members contributions since early August. There has not been any response to this request.

The BBS has the capability for Directors and Sub-Editors to upload files directly to the BBS. The last time a Director uploaded a file was four months ago. It has been so long since a sub-editor provided a file that it does not warrant talking about.

I am concerned that the BBS has become a one way street. I am now the only person who has contributed any new material on the BBS of more than a few lines, for over 5 months. The only other items are short items that users place on the ALL MAIL file. If the BBS is to remain viable it must have a number of users contributing material to keep an interesting range of subjects available.

I am quite happy to continue to provide accomodation and physical maintenance of the BBS, but in future users may find that either material has not changed or there is not much on the BBS, if others are not prepared to contribute some material.

The bottom line to the present trend is that if members do not want to help, then the BBS will probably close within a few months due to lack of interest. o

continued from page 5

After some discussion a motion to amend Clause 4 (see May 1992 TND) was put to the meeting by Geoff Trott, seconded Percy Harrison. On a show of hands the motion was passed unanimously.

4. Close:

The special general meeting was closed at 3.20pm.

Minutes recorded by: Minutes confirmed by:

Terry Phillips
Honorary Secretary

Dick Warburton
Chairman

6 June, 1992 / / o

continued from page 7

TC820 Health and the Human Body SSSD $2
TC830 Physics SSSD $2
TC850 Chemistry SSSD $2
TC860 Astronomy Disk #1 SSSD $2
TC890 Teacher's Helper SSSD $2
TC911 Display Calculator SSSD $2
TC990 Sports (Requires XB) SSSD $2
TC1015 Word Processing Utilities SSSD $2
TC1102 Sorts,Scrambles,Searches SSSD $2
TC1119 Hardware Utilities #1 SSSD $2
TC1120 Sound Effects SSSD $2
TC1122 Screen Fonts-Peterson DSDD $2
TC1131 Gemini Printer Utilities SSSD $2
TC1145 Telecommunications SSSD $2
TC1210 Graphics Printing SSSD $2
TC1211 TI Artist Pictures #1 SSSD $2
TC1212 TI Artist Pictures #2 SSSD $2
TC1213 TI Artist Pictures #3 SSSD $2
TC1219 R Kazmer's Xmas Card SSSD $2

Club Hardware

5.25 inch DSDD Disks (Boxes of ten) $6
5.25 inch HD Disks (Boxes of ten) $10
3.5 inch DSDD Disks (Boxes of ten) $10
5.25 inch DSDD Half Height Drive (New) $65
12 Volt AC Transformer $4
13 Volt Arlec Transformer $12
8.5, 17 Volt Transformer $25
TI Power Transformer $25
TI Keyboard $15
60 VA Transformer $20
MFC Printed Circuit Board $30
Music Kit with PCB $65
32K Memory PC Board $7
Eprom RAM PC Board $45
Eprom RAMdisk Basic Kit $35
Funnelweb Eprom Set (3 Eproms) $36
TI Artist Eprom Set (2 Eproms) $24
32K Static RAM IC (62256) $10
8K Static RAM IC (6264LP) $5
Exchange Console $30
Peripheral Expansion Box $150
TI Disk Control Card $60
TI 32K Memory Card $40
RS232/PIO Card $100
AT Card (DSDD Format) $150
AT Card (DSDD,32K,RS232/PIO) $220
Modem PE Card (300 Bd) $60
PE Eprom RAMdisk (248K) $195
PE Horizon RAMdisk (184K) $155
PE Horizon RAMdisk (256k) $200
Printer (Serial) $120
Midimaster 99 .....$60
Mini PE RS232/32K Card $80
Mini PE RS232/PIO PC Board $30
Mini RAMdisk PC Board $30
Null Modem Cable $15

Club T-Shirts Sizes 14, 16, 18, 20 .....$11

Bye for now. o

Treasurer's Report

by Geoff Trott

Firstly, I must apologise for the lateness of the last TND. We had a few production problems which delayed things past the normal deadlines. Also I must apologise for the error I just noticed in my last report. Please ignore the mention of July; it was of course August. I am still feeling like there is not enough time in each day. At least I have the club finances sorted out. We had a good stable year, thanks largely to Percy's efforts in the Shop, which has led to all of us spending more there this last year and helping to maintain the financial viability of the club.

Income for September	$262.65
Payments for September	$1003.18
Excess of expenses over income for September	$740.53 o

XB Tricks - Sprites

by Andy Frueh, OH USA

Sprites are handy little things. If you are interested in graphics or games at all, I can recommend buying the Extended BASIC cartridge simply for sprites. For those who do not know, sprites are characters that can be set in real motion. Instead of moving character by character (8 pixels) they can move 1 pixel at a time, giving very smooth movement.

Extended Basic has several built-in CALLs that relate to sprites. A few of the familiar CALLs such as CHAR and COLOR have been expanded so that the controls either the shape or colours of sprites. I will get into that in a minute.

The first, and most essential, statement for sprite programming is the CALL SPRITE. It literally calls a sprite into being. The syntax for it is as follows: CALL SPRITE(sprite #,ASCII value,colour,row,column,row speed,col speed) where the sprite # is a number between 1 and 28 used to label the sprite. The ASCII value can be pre-defined or defined by the user. It must be between 32 and 143. If you use double sized sprites (see below) than the character value you input is the first of a set of 4, the next three having consecutive values. For example, if you used 32 as your value, a double size sprite also uses 33, 34, and 35.

The sprite colour can be any colour between 1 and 16. Note that a colour of 1, or one that matches the screen colour will make the sprite invisible. Next, you specify the row. This is a dot (pixel) value between 1 and 192. It can go up to 256, but this is off of the screen. The column can be between 1 and 256. The sprite's position is where the upper leftmost part of the sprite will appear, rather than the center of the sprite as you might think. Keep this in mind if you need to place sprites exactly.

The next two numbers are optional. The row and column velocity can be between -128 and 127. Negative numbers go to the left, and positives to the right, with numbers being close to zero the slowest. A row and column speed of 0 stops the sprite.

A little while ago, I mentioned the use of COLOR and CHAR in sprites. If CHAR is used to define a character before it is turned into a sprite, that sprite assumes the defined shape. Thus, a sprite can be any shape. To use the COLOR subprogram, use CALL COLOR(sprite #,foreground). You do not define the background because with sprites, the background is always 1.

Next we have the CALL COINC routine. This is handy for chase-type games because it detects when sprites are touching. It comes in three forms. CALL COINC(sprite,sprite,tolerance,variable) lets you know when two sprites have met. The two sprites are named by their number. The tolerance is given in dots. For example, if the tolerance was 5, the sprites would "be touching" when they were 5 dots apart at any angle. This is preceded set to 1 or 0 in games, to enhance the precision of the judging of the game. The last part is the variable which returns different values depending on the sprite's coincidence.

A second form of COINC is CALL COINC(sprite,row,column,tol,var). This form reports whether or not a sprite is at or (depending on the tolerance) near a certain location. The final form is CALL COINC(ALL,var). The reports on all the sprites in use. It determines a coincidence if one or more of the dots that make the sprites are overlapping. The accuracy of this statement depends on many things. For example, a sprite that moves very quickly may not always register a coincidence every time one occurs. Also, the routine checks only when it is called, so if a sprite has a coincidence condition while the program is doing something else, it may not be detected. Programmers beware!

Another statement we have to use is the CALL DELSPRITE. It erases certain sprites, or all of them. The syntax is CALL DELSPRITE(sprite/ALL). Note that with all routines that call for a sprite number, you can provide a numeral and perform some operation on it. For example, sprite #3-X is an acceptable sprite number, as long as the operation does not make the sprite number exceed the values between 1 and 28. Also, sprite numbers must be preceeded with a # sign. Anyway, either place a sprite number, list of numbers, or the word ALL after a DELSPRITE to completely erase that sprite. After it is erased, it must be recreated with a CALL SPRITE.

The DISTANCE subprogram is next in line. It tells us how far a sprite is from a certain location or another sprite. To use it, pick one of two forms. CALL DISTANCE(sprite,sprite,variable) to tell the distance between two sprites. The variable will return a value that equals the distance in dots. It is found in the following way. First, the distance between the dot-rows of the sprites (or the sprite and the location) is found and squared. Next, the square of the dot-columns is found. These two values are added together. If the value is higher than 32767 (what IS it with this number?) than 32767 is returned. The distance between one sprite and the other, or the location, is square root of the number returned by the variable. The other form is CALLS DISTANCE(sprite, row,col,var).

CALL LOCATE is the subprogram that lets us change the location of any sprite without having to actually move it, or recreate it. The syntax is CALL LOCATE(sprite,row,col). As soon as this statement is executed, it locates the sprite with the number you gave, and where it is at. It then makes it skip to the new spot.

The next, and most interesting, of the sprite controllers is the MAGNIFY subprogram. It has a simple syntax of CALLS MAGNIFY(size). All sprites are affected by it. Mag factors are between 1 and 4. A size of 1 is the standard size. A factor of 2 causes the sprites to be simply magnified in size. Factor 3 causes sprites to be double size, but unmagnified. The sprite is defined by 4 characters, not one. The first character is the upper left. The next goes in the lower left, then upper right. The final character is in the lower right corner. A factor of 4 is similar to 3, except that the sprites are magnified as well as doubled up.

Another interesting command is the MOTION command. It lets you change the speed an/or direction of any sprite. The syntax is CALL MOTION(sprite,row velocity,col velocity). The values may be from -128 to 127, with the rules being the same as those with the CALL SPRITE command.

A seldom-used, but nonetheless useful command is PATTERN. It lets you alter the character pattern of any sprite instantly. Type in CALL PATTERN(sprite #,value) where the sprite is the number of the one to change, and the value is the new ASCII code for the sprite.

Yet another command is POSITION. It returns two values for any sprite's current row and col. CALL POSITION(sprite,row,col).

Something I should mention is that with most of the sprite command subprograms, you can repeat the data over and over, so that one command will control several sprites. For example, you could use CALL LOCATE(sprite 1,row,col,sprite 2,row,col,...) and so forth. You can repeat with the COLOR, DELSPRITE, LOCATE, MOTION, PATTERN, POSITION, and SPRITE commands.

I hope this clears some of the confusion with using sprites. The Extended Basic manual does a poor job, because it references other parts constantly, and repeats itself when you do not want it to. The examples are worth looking at, though, and to keep this article short, I did not include any example programs. However,

continued on page 20

Tips from the Tigercub #66

by Jim Peterson, Tigercub Software, USA

My three Nuts & Bolts disks, each containing 100 or more subprograms, have been reduced to $5.00 each. I am out of printed documentation so it will be supplied on disk.

My TI-PD library now has well over 500 disks of fairware (by author's permission only) and public domain, all arranged by category and as full as possible, provided with loaders by full program name rather than filename, Basic programs converted to XBasic, etc. The price is just $1.50 per disk(!), post paid if at least eight are ordered. TI-PD catalog #5 and the latest supplement is available for $1 which is deductible from the first order.

In Tips #65 I said that the TI could calculate to 14-digit accuracy, rather than the 8-digit accuracy of a PC. Actually the number in memory is calculated to 13- or 14-digit accuracy, depending on the number, but is rounded to 10 digits on the screen display, or shown in exponential notation if the number is extremely large or extremely small. If you want to see the complete number, this routine will show the normal screen display and the full number in memory. To see the complete range of numbers our little TI can handle, try inputting the following:

-9.9999999999999E127 and
-1.0000000000000E-128 and
1.0000000000000E-128 and
9.9999999999999E127.

```
100 OPEN #1:"DSK1.INT2",INTERNAL,RELATIVE,UPDATE
110 INPUT N
120 PRINT #1,REC 1:N
130 INPUT #1,REC 1:N$
140 X=ASC(SEG$(N$,1,1)):: Y=ASC(SEG$(N$,2,1)):: IF Y>99
THEN Y=256-Y :: N$=SEG$(N$,1,1)&CHR$(Y)&SEG$(N$,3,255)
150 FOR J=2 TO LEN(N$):: X$=STR$(ASC(SEG$(N$,J,1)))
160 IF ASC(SEG$(N$,J,1))<10THEN X$="0"&X$
170 P$=P$&X$ :: NEXT J
180 IF X<63 THEN Y$="."&RPT$("00",63-X)&P$ :: GOTO 230
190 IF X>191 THEN Y$="."&RPT$("00",X-192)&P$ :: GOTO 230
200 IF X>185 THEN Y$=SEG$(P$,1,14-(X-185)*2)&"."&SEG$(P$
,14-(X-185)*2+1,255):: GOTO 230
210 Y$=SEG$(P$,1,(X-63)*2)&"."&SEG$(P$,(X-63)*2+1,255)
220 IF ASC(Y$)=48 THEN Y$=SEG$(Y$,2,255)
230 IF N<0 THEN Y$="-"&Y$
240 PRINT TAB(2);N :: PRINT TAB(3);Y$ :: P$="" :: GOTO
110
```

But even the smart little TI has its limits. Try this- X=2/3-1/3-1/3 :: PRINT X. See the TI User's Reference Guide page III-13 for the explanation of all this.

Solving an equation such as X X/X-X would be very difficult to solve by mathematical means, but our computer can find the answer quickly by systematic trial and error, to the 14-point limit of its accuracy.

```
100 DISPLAY AT(3,1)ERASE ALL:"This program will solve
even the most difficult equations with one variable."
110 DISPLAY AT(6,1):"Put your own equation in line 21
0, using A for the known value and X for the unknown
value."
120 DISPLAY AT(24,6):"PRESS ANY KEY" :: DISPLAY AT(24,6)
:"press any key" :: CALL KEY(0,K,S):: IF S=0 THEN 120
130 CALL CLEAR
140 DISPLAY AT(8,1):"KNOWN VALUE? " :: ACCEPT AT(8,14):C
150 X=1 :: DISPLAY AT(12,1):""
160 GOSUB 210
170 IF A<C THEN Y=X :: X=X*2 :: GOSUB 210 :: GOTO 170 EL
SE 190
180 IF A>C THEN Y=X :: X=X/2 :: GOSUB 210 :: GOTO 180
190 Z=(ABS(X-Y))/2 :: Y=X:: IF A<C THEN X=X+Z ELSE X=X-Z
200 GOSUB 210 :: GOTO 190
210 A=X X/X-X/2
220 IF A<C THEN DISPLAY AT(11,5):X ELSE IF A>C THEN DISP
LAY AT(13,5):X
230 IF A=C OR A=B THEN DISPLAY AT(12,5)ERASE ALL:X :: GO
TO 140 ELSE B=A :: RETURN
```

In Recreational and Educational Computing, published 8 times a year at 909 Violet Terrace, Clarks Summit PA 18411, $27 per year, I found a neat routine to find the greatest common divisor and least common multiple of any two numbers - so I converted it to TI Basic and modified to do the same with multiple numbers.

```
100 CALL CLEAR :: PRINT "PROGRAM TO FIND THE GREATESTCOM
MON DIVISOR AND LEAST COMMON MULTIPLE OF ANY NUMBER OF
NUMBERS.": :
110 DIM N(100)
120 PRINT "INPUT ZERO WHEN FINISHED":
130 T=T+1 :: INPUT "NUMBER "&STR$(T)&"? ":N(T):: IF N(T)
=0 THEN 140 ELSE IF N(T)<>INT(N(T))OR N(T)<1 THEN T=T-1
:: GOTO 130 ELSE 130
140 AA=N(1):: GCD=N(2)
150 GOSUB 170 :: FOR J=3 TO T-1 :: AA=N(J):: GCD=ABS(GCD
):: GOSUB 170 :: NEXT J
160 GOTO 210
170 R=AA-INT(AA/(GCD+ABS(GCD=0)))*GCD
180 IF R<2 THEN GCD=R+GCD*(1-R):: GCD=GCD*ABS(GCD>0)+ABS
(GCD=0):: GOTO 200
190 AA=GCD :: GCD=R :: GOTO 170
200 RETURN
210 PRINT "THE GREATEST COMMON DIVISOR OF YOUR";T-1;"NUM
BERS IS";GCD
220 L=N(1)*N(2)/GCD :: FOR J=3 TO T-1
230 IF L/N(J)<>INT(L/N(J))THEN L=L*N(J)
240 NEXT J
250 LL=L/2 :: FOR J=1 TO T-1 :: IF LL/N(J)<>INT(LL/N(J))
THEN J=T-1 :: GOTO 270
260 NEXT J :: L=LL :: GOTO 250
270 PRINT "AND THE LOWEST COMMON MULTIPLE IS";L
```

Joy Warner called from the L.A. group, and mentioned that it would be nice to have a program to print out a page of math problems, and a page of answers. So here is one that will randomly create any number of either addition or subtraction problems, within any specified range of numbers, and output the desired number of copies to a printer in two columns of expanded print, numbered in sequence, plus a numbered answer sheet to make it easy for the teacher.

```
100 DISPLAY AT(1,4)ERASE ALL:"MATH PROBLEM PRINTER" !by
Jim Peterson
110 DIM A(200),H(200),L(200) :: OPEN #1:"PIO":: PRINT #1
:CHR$(27)&"@"&CHR$(27)&"W"&CHR$(1);
120 M$(1)="ADDITION" :: M$(2)="SUBTRACTION" :: D$(1)="+
" :: D$(2)="- " :: ON$=CHR$(27)&"-"&CHR$(1):: OFF$=CHR$(
27)&"-"&CHR$(0)
130 DISPLAY AT(3,1):"Do you want?":"":"1. "&M$(1):"2. "&
M$(2):: ACCEPT AT(3,14)VALIDATE("12")SIZE(1)BEEP:C
140 DISPLAY AT(8,1):"Range of numbers?":"From":"To" :: A
CCEPT AT(9,6)VALIDATE(DIGIT)BEEP:LN :: ACCEPT AT(10,4)VA
LIDATE(DIGIT)BEEP:HN :: IF LN>=HN THEN 140 ELSE HN=HN-LN
150 DISPLAY AT(13,1):"How many problems?" :: ACCEPT AT(1
3,20)VALIDATE(DIGIT)BEEP:P
160 DISPLAY AT(15,1):"How many copies?" :: ACCEPT AT(15,
18)VALIDATE(DIGIT)BEEP:CC
170 FOR J=1 TO P :: GOSUB 290 :: H(J)=N1 :: L(J)=N2
180 IF C=1 THEN A(J)=H(J)+L(J)ELSE A(J)=H(J)-L(J)
190 NEXT J
200 FOR J=1 TO CC :: GOSUB 310 :: FOR K=1 TO P STEP 2
210 T1$=STR$(K)&".  "&STR$(H(K)):: T2$=STR$(K+1)&".
 "&STR$(H(K+1))
220 PRINT #1:TAB(15-LEN(T1$));T1$;TAB(35-LEN(T2$));T2$
230 T1$=D$(C)&STR$(L(K)) :: T2$=D$(C)&STR$(L(K+1))
240 PRINT #1:TAB(15-LEN(T1$));ON$&T1$&OFF$&RPT$(" ",20-L
EN(T2$))&ON$&T2$&OFF$
250 PRINT #1:"":"":"" :: IF K/19=INT(K/19)THEN PRINT #1:
CHR$(12);
260 NEXT K :: PRINT #1:CHR$(12) :: NEXT J
270 PRINT #1:TAB(16);"ANSWERS":"":""
280 FOR J=1 TO P STEP 2 :: PRINT #1:TAB(6);STR$(J)&".  "
;A(J);TAB(26);STR$(J+1)&".  ";A(J+1):: NEXT J :: STOP
290 RANDOMIZE :: N1=INT(RND*HN+LN):: N2=INT(RND*HN+LN)::
 IF N1=N2 THEN 290
300 IF C=2 AND N2>N1 THEN T=N2 :: N2=N1 :: N1=T :: RETUR
N ELSE RETURN
310 PRINT #1:" "&M$(C) &" PROBLEM PRINTER":"":"":""
:"" :: RETURN
```

And this one will do the same with multiplication problems.

```
100 DISPLAY AT(1,4)ERASE ALL:"MULTIPLICATION PROBLEMS":"
PRINTER" !by Jim Peterson
110 DIM A(200),H(200),L(200) :: OPEN #1:"PIO" :: PRINT
#1:CHR$(27)&"@"&CHR$(27)&"W"&CHR$(1);
120 ON$=CHR$(27)&"-"&CHR$(1):: OFF$=CHR$(27)&"-"&CHR$(0)
130 DISPLAY AT(8,1):"Range of multiplicand?":"From":"To"
:: ACCEPT AT(9,6)VALIDATE(DIGIT)BEEP:L1 :: ACCEPT AT(10
,4)VALIDATE(DIGIT)BEEP:H1 :: IF L1>=H1 THEN 130 ELSE H1=
H1-L1
140 DISPLAY AT(12,1):"Range of multiplier?":"From":"To"
:: ACCEPT AT(13,6)VALIDATE(DIGIT)BEEP:L2 :: ACCEPT AT(14
,4)VALIDATE(DIGIT)BEEP:H2
150 IF L2>=H2 THEN 140 ELSE R=LEN(STR$(H2)):: H2=H2-L2
160 DISPLAY AT(16,1):"How many problems?" :: ACCEPT AT(1
6,20)VALIDATE(DIGIT)BEEP:P
170 DISPLAY AT(18,1):"How many copies?" :: ACCEPT AT(18,
18)VALIDATE(DIGIT)BEEP:CC
180 FOR J=1 TO P :: GOSUB 310 :: H(J)=N1 :: L(J)=N2
190 A(J)=H(J)*L(J)
200 NEXT J
210 FOR J=1 TO CC :: GOSUB 320 :: FOR K=1 TO P STEP 2
220 T1$=STR$(K)&".  "&STR$(H(K)):: T2$=STR$(K+1)&".
 "&STR$(H(K+1))
230 PRINT #1:TAB(15-LEN(T1$));T1$;TAB(35-LEN(T2$));T2$
240 T1$="X "&STR$(L(K)):: T2$="X "&STR$(L(K+1))
250 PRINT #1:TAB(15-LEN(T1$));ON$&T1$&OFF$&RPT$(" ",20-L
EN(T2$))&ON$&T2$&OFF$
260 FOR S=1 TO R+3 :: PRINT#1:"" :: NEXT S
270 LC=LC+5+R :: RC=LC+5+R :: IF RC>=60 AND K<P THEN PRI
NT #1:CHR$(12):: LC=5
280 NEXT K :: PRINT #1:CHR$(12);:: NEXT J
290 PRINT #1:TAB(16);"ANSWERS":"":""
300 FOR J=1 TO P STEP 2 :: PRINT #1:TAB(3);STR$(J)&".  "
;A(J);TAB(23);STR$(J+1)&".";A(J+1):: NEXT J :: PRINT #
1:CHR$(12):: STOP
310 RANDOMIZE :: N1=INT(RND*H1+L1):: N2=INT(RND*H2+L2)::
 RETURN
320 PRINT #1:" MULTIPLICATION PROBLEMS":"":"":"":
"" :: LC=5 :: RETURN
```

And division -

```
100 DISPLAY AT(1,6)ERASE ALL :"DIVISION PROBLEMS":"
    PRINTER" !by Jim Peterson
110 DIM A(200,2),H(200),L(200):: OPEN #1:"PIO" :: PRINT
#1:CHR$(27)&"@"&CHR$(27)&"W"&CHR$(1);
120 DISPLAY AT(8,1):"Range of dividend?":"From":"To" ::
ACCEPT AT(9,6)VALIDATE(DIGIT)BEEP:L1 :: ACCEPT AT(10,4)V
ALIDATE(DIGIT)BEEP:H1 :: IF L1>=H1 THEN 120
130 DISPLAY AT(12,1):"Range of divisor?":"From":"To" ::
ACCEPT AT(13,6)VALIDATE(DIGIT)BEEP:L2 :: ACCEPT AT(14,4)
VALIDATE(DIGIT)BEEP:H2
140 IF L2>=H2 THEN 130 ELSE R=LEN(STR$(INT(H1/H2)))*2 ::
 H2=H2-L2 :: H1=H1-L1
150 DISPLAY AT(16,1):"How many problems?" :: ACCEPT AT(1
6,20)VALIDATE(DIGIT)BEEP:P
160 DISPLAY AT(18,1):"How many copies?" :: ACCEPT AT(18,
18)VALIDATE(DIGIT)BEEP:CC
170 FOR J=1 TO P :: GOSUB 310 :: H(J)=N1 :: L(J)=N2
180 A(J,1)=INT(H(J)/L(J)):: A(J,2)=H(J)-A(J,1)*L(J)
190 NEXT J
200 FOR J=1 TO CC :: GOSUB 320 :: FOR K=1 TO P STEP 2
210 LC=LC+1 :: T1$=STR$(K)&".  "&RPT$(" ",LEN(STR$(L(K
))))&RPT$("_",LEN(STR$(H(K))))
220 T2$=STR$(K+1)&".  "&RPT$(" ",LEN(STR$(L(K+1))))&RP
T$("_",LEN(STR$(H(K+1))))
230 PRINT #1:TAB(15-LEN(T1$));T1$;TAB(35-LEN(T2$));T2$
240 T1$=STR$(L(K))&"|"&STR$(H(K)):: T2$=STR$(L(K+1))&"|"
&STR$(H(K+1))
250 LC=LC+1 :: PRINT #1:TAB(15-LEN(T1$));T1$;TAB(35-LEN(
T2$));T2$
260 FOR S=1 TO R+3 :: LC=LC+1 :: PRINT #1:"" :: NEXT S
270 IF 66-LC<5+R AND K<P THEN PRINT #1:CHR$(12):: LC=5
280 NEXT K :: PRINT #1:CHR$(12);:: NEXT J
290 PRINT #1:TAB(16);"ANSWERS":"":""
300 FOR J=1 TO P :: PRINT #1:TAB(3);STR$(J)&".  ";A(J,1)
;"REMAINDER ";A(J,2):: NEXT J :: PRINT #1:CHR$(12) ::
STOP
310 RANDOMIZE :: N1=INT(RND*H1+L1):: N2=INT(RND*H2+L2)::
 RETURN
320 PRINT #1:" DIVISION PROBLEMS":"":"":"":""
:: LC=5 :: RETURN
```

Bud Wright wrote this one for Irwin Hott, so he could listen to lower case text with the Speech Synthesizer. Imbed it with ALSAVE, access it with CALL LINK("CAPS",A$) and it will instantly convert any lower case letters to upper case. I found it invaluable in writing key-word search programs.

```
* CAPS/S BY BUD WRIGHT
* VERSION 1.1 10/17/86
STRREF EQU >2014
STRASG EQU >2010
MREG   BSS 32
STRBUF BYTE 255
       BSS 255
       DEF CAPS
CAPS   LWPI MREG
       CLR R0
       LI R1,1
       SETO @STRBUF
       LI R2,STRBUF
       BLWP @STRREF
       MOVB @STRBUF,R2
       SRL R2,8
       JEQ CAPOUT
       LI R1,STRBUF+1
CAPS2  MOVB *R1,R3
       SRL R3,8
       CI R3,96
       JGT CAPS1
CAPS3  SWPB R3
       MOVB R3,*R1+
       DEC R2
       JNE CAPS2
       CLR R0
       LI R1,1
       LI R2,STRBUF
       BLWP @STRASG
CAPOUT LWPI >83E0
       B @>006A
CAPS1  CI R3,122
       JGT CAPS3
       AI R3,-32
       JMP CAPS3
       END
```

Memory full, Jim Peterson o

PRINT USING

by Mark Schafer, USA

This is a new and different animal for me, as I normally do not write tutorials. It came up at a recent meeting that some people are having trouble with PRINT USING. Since I consider it to be no problem, I volunteered to write an article about it. As for the title, remember that USING can also be used in a DISPLAY USING statement. There is a lot of ground to cover, so let's get started with....

Beginner's Stuff

USING represents a method by which you can get data to print in a specific format instead of the default format. The syntaxes are:

```
PRINT USING format:print-list
DISPLAY [option-list:] USING format[:print-list],
```

where format is a line number or a string expression. A line number would be the line where an IMAGE statement is found which would contain the string expression which is the format. A string expression can be as simple as a string literal enclosed in quotes or a complicated expression made up variables, function calls, whatever. Now let's compare the two methods of printing:

```
100 PRINT -56.7;109.2850
110 PRINT USING "####.# ###.####":-56.7,109.285
```

RUN this program and it looks like this:

```
-56.7 109.285
-56.7 109.2850
```

USING allows you to better control the spacing as well as put trailing zeroes to the right of the decimal point.

Let's look at the format string. It is made up of fields with optional text. Fields are like fill-in-the-blanks. They mark the place where an unknown value will be printed. Text is printed the same way every time. Fields are made up of pound signs (#) with possibly a decimal point, a minus sign, or maybe some circumflexes (^). I will cover circumflexes in a later section.

Pound signs take the place of a digit or sign. So in line 110 above, I printed -56.7 with the field "####.#". -56.7 has only three characters to the left of the decimal point, so the initial character is left blank; the second one is where the minus sign went. Numbers are always aligned with the decimal point. If there is none, it is assumed to be at the end.

If there is a minus sign in the field, it always goes at the beginning. It indicates that you want the minus sign to appear immediately to the left of the number if it is negative. However, this is the same thing another pound sign would do, so you never really need to use a minus sign (except for a more advanced purpose to be discussed later.) In other words, "-##.##" does the same thing as "###.##".

The decimal point indicates where you want it to be. The number of pound signs you put to the left of it dictates how many digits you want to the left; the number you put on the right indicates how many decimal places you want. The decimal is always printed even if there are no pound signs after it.

If there are fewer digits in the value to the left of the decimal point than there are in the field, spaces are used to fill it out. If there are too many, the computer will refuse to print your value and fill the field with asterisks (*). This means your value is too high and/or there is not enough places in your field. This includes the minus sign, if any.

If there are fewer digits in the value to the right of the decimal point than there are in the field, zeroes are used to fill in the remainder. If there are too many, the computer will estimate the number to the number of places given. So .## can be used to estimate to the nearest 100th, .# to the nearest tenth, and if there are no decimal places, it will estimate to the nearest unit.

Below is a table of how various values will be printed with various fields. The left side represents the value; across the top are the fields.

	#	##	###	#.#	##.##	###.###
2	2	2	2	2.0	2.00	2.000
-13	*	**	-13	***	*****	-13.000
9.671	*	10	10	9.7	9.67	9.671
125.678	*	**	126	***	*****	125.678
-.385	*	-0	-0	-.4	-.39	-.385
-3.05	*	-3	-3	***	-3.05	-3.050

Generally, Extended Basic handles format strings like this: It keeps printing text until it comes to a field (characterized by a pound sign). Then it looks for the next value to be printed. If there is one, it prints it in the format specified; if there is none, it terminates there and does not print anything else. However, you must specify at least one value and there must be at least one field in the format. Except, curiously,

DISPLAY USING

does not need any values. Why anyone would use DISPLAY USING without any values is beyond me. Anyway, if it reaches the end of the string, and there is more values to be printed, it goes to the next line and starts over at the beginning of the string. So it is ok if the number of fields does not match the number of values.

Going back to syntax, if you want to use the format "I HAVE $###.##", you have three ways of doing it:

```
120 IMAGE I HAVE $###.##
130 PRINT USING 120:M

140 A$="I HAVE $###.##"
150 PRINT USING A$:M

160 PRINT USING "I HAVE $###.##":M
```

Notice that if use the first method using IMAGE, you do not need quotes. The only time you need quotes with an IMAGE statement would be when you have leading or trailing spaces. If you are going to use it repeatedly in a program, IMAGE is the most efficient method. If you are only using it once, go with the third method. Practically the only time you need the second method would be when the format string is constructed so you may not know exactly what it looks like. Remind me, and I might discuss that later.

The IMAGE statement must be on a line by itself. The computer ignores it when it comes to it in a program the same as it does the DATA statement, so you can put it anywhere. Circumflexes are used to denote scientific notation which leads us to the next section....

Intermediate Stuff

You may want the number to come out in scientific notation (E format) even if the number normally would not. To do this you put four or five circumflexes at the end of the field. Four means you want two digits in the exponent; five means you want three. If you put less than four, they will be treated as text; if you put more than five, the first five will be used, and the rest will be treated as text.

There is a little something different about E format. It always reserves the first character for the sign, so you will need at least two #'s in the mantissa (or precede the field with a sign). Using the same numbers as above, the table can be amended thusly:

```
##      ####      #.####
2 2E+00 200E-02 .2000E+001
-13 -1E+01 -130E-01 -.1300E+002
9.671 1E+01 967E-02 .9671E+001
125.678 1E+02 126E+00 .1257E+003
-.385 -4E-01 -385E-03 -.3850E+000
-3.05 -3E+00 -305E-02 -.3050E+001
```

This format also estimates whenever it is not given enough places to express the exact value or tacks on zeroes when given too many. You will probably not need this format unless you are dealing with exceedingly low or high numbers.

Time to switch to another concept. USING can also be used to format text values. No bells or whistles here, though. Any field that can be used to format a number can also be used to format text. Text is just left-justified, and all characters within a field are treated the same. I can give you a table, but I assure you, it is quite boring:

```
### ###.### -###
BOB BOB BOB BOB
JOHN *** JOHN JOHN
25% OFF! *** ******* 25% OFF!
```

In general, it makes more sense to use only #'s when constructing a text field. You may want to do something like column 2 if you are printing a table of numbers, and you want to use the same format string for the column headers as in the table.

Of course you can mix numeric and text values for the same format as in something like:

```
170 IMAGE ##### HAS $###.##.
180 PRINT USING 170:"MARY",123.4
```

This will print:

```
MARY HAS $123.40.
```

Note the computer recognizes the period at the end as a period and not as a decimal point since the field already has one. Even if it did not, it would not make any difference since the decimal point would be printed at the end.

Now suppose you want part of a line formatted instead of a whole line. Do not worry; you can always put a semicolon at the end of a PRINT or PRINT USING statement, so you can stay on the same line for the next PRINT. You can also use this technique if you need to print a # as text instead of a field character. However, you cannot put a comma at the end of a PRINT USING statement or the computer will think you left out a value.

Another problem you may have is suppose you want to concatenate two fields, say ## with ###. If you put them together, you get #####, and the computer will see that as one field. Well, you could try using "##-##". The computer will then recognize that as two fields, ## and -##. But if your second field can have three digits, this will not work. If this is the case, you will need to split up the format string into two PRINT USING statements. Perhaps like this:

```
190 PRINT USING "##":A;
200 PRINT USING "###":B
```

You cannot confuse the computer with pound signs, minus signs, circumflexes, decimal points, etc. It will always know where one field ends and another begins. Except in the case like above where two fields collide.

I will give you another way to handle that in....

Advanced Stuff

There is not very much in the way of advanced stuff because this is such a small subset of a much bigger entity. I always consider undocumented features to be advanced stuff. That is right, boys and girls, USING has an undocumented feature! By "etc." in the above paragraph, I meant the plus sign! It can also be used like the minus sign in a field. What it will do is float the sign in front of the number be it positive or negative. Let me illustrate:

```
+# +### +###.##
2 +2 +2 +200.00E-02
-13 ** -13 -130.00E-01
9.671 ** +10 +967.10E-02
125.678 ** +126 +125.68E+00
-.385 -0 -0 -385.00E-03
-3.05 -3 -3 -305.00E-02
```

If it is negative, you get a minus sign; if it is positive or zero, you get a plus sign. There is no word about this in the Extended Basic manual. This can be used to write format strings for equations, like this:

```
200 IMAGE ###x 2+##x+##
210 PRINT USING 200:2,7,10
220 PRINT USING 200:-15,-11,1
230 PRINT USING 200:+9,33,-14
```

This will print:

```
2x 2 +7x+10
-15x 2-11x +1
9x 2+33x-14
```

Suppose you do not want it to print those leading spaces when there are not enough digits to fill the field. This is where you would use a variable for the format string. You would construct it so that each field would only have as many #'s as it needed to print the number (since they are all integers, that can easily be accomplished with LEN(STR$(X))) and concatenate the necessary text and use the variable as the format string. For instance, I have a program that uses a define function like this:

```
240 DEF MF$(X)=" $"&RPT$("#",LEN(STR$(INT(X))))".##"
```

That function generates a format string for money so that there will no spaces between the dollar sign and the amount. The reason I did it this way was to get trailing zeroes after the decimal point to look good. However, user-defined function calls take a long time in Extended BASIC, so if need speed or if you only need it once, put a formula like this directly where you need it.

There is another manual correction that must be made (although it may have been made already in some addendum I do not know about). The syntax for PRINT USING with files is wrong. I will not reprint it here because I do not like glorifying the incorrect. The correct syntax is as follows:

```
PRINT[#file-number,[REC record-number,]USING format:
print-list
```

That translates specifically as something like:

```
250 PRINT #1,USING 200:A,B,C
260 PRINT #2,REC 15,USING 200:A*4,B*4,C*4
```

The file being referenced is most likely the printer, but it could be a disk file. If it is, it must be a DISPLAY format file; you will get a FILE ERROR if is INTERNAL format. Obviously line 260 is not referring to the printer.

PRINT USING is very good for printing to a printer because the printer has so many more columns to print in. So there is a good chance that you may have a program in which a particular PRINT USING is only being used to print to the printer and never to the screen. If this is the case, there is another solution to the concatenated field problem mentioned in the last section. You can print some unprintable character between the two fields. The computer will then recognize them as two fields but they will be together on the printout! Something like this would be typical:

continued on page 16

Programming Music part 4

by Jim Peterson, Tigercub Software, USA

The first three parts of this series were written and published some time ago, so I had better review. In Part 1, I showed you this one-line routine to set up a musical scale.

```
100 DIM N(36):: F=110 :: FOR J=1 TO 36 :: N(J)=INT(F*1.0
59463094 (J-1)+.5):: NEXT J :: N(0)=40000 ::GOTO 110
101 D,T,A,B,C,V1,V2,V3,J,X,V
102 CALL SOUND
103 !@P-
```

That sets up a scale of three octaves beginning with A. If you decide to change the music to a higher key, just change the 110 to 117, 123, 131, 139, 147,156,165, 175, 185, 196, 208 or 220. In fact, for some music you will have to change it, if the program crashes with a BAD VALUE error message.

If you have programmed the music with high notes, you can lower the key by changing 110 to 104, 98, 92, 87, 82, 78, 73, 69 or 65. Again, if you try to go too low you will get that BAD VALUE message.

I have given N(0) a value of 40000, which creates a tone too high to be heard. This can be used to silence a note, but it can also cause a crash when used with some of the following routines. If you are programming three voices and want to play a single note, the easiest way is to give all three notes the same number, such as A,B,C=10. If you need a silent rest, play all the notes at an inaudible volume by V1,V2,V3=30 and then, after the GOSUB, restore their original volume by V1= (whatever is in line 110) and the same for V2 and V3.

Lines 101-103 are a pre-scan routine to start the music playing sooner. There will still be a few seconds delay while that array is set up in line 100. You can perhaps shorten that delay slightly by changing the 36 to the highest note number you have used in programming your piece.

However, Bruce Harrison wrote for me an assembly link which eliminates any delay; this also makes it possible to change key while the music is playing. I will not list the source code here, because everyone is afraid to key in source code anyway, but it is available on my TI-PD disk #1143 and will also be on a tutorial disk on this type of music programming.

Part 2 of this series contained a listing of a program to easily give you the numbers you would need in order to key in a particular piece of sheet music. If you do not have that, you can just take a piece a paper and list the scale A Bf B C C# D Ef etc., on through as many as you will need, and then number them consecutively. For the length of the notes, give the shortest note a value of 1 unless it also appears as a dotted note, in which case it must be 2, and then number the others according to their relative length - for example, 2 for a quarter note, 3 for a dotted quarter, 4 for a half note, 8 for a whole note.

Part 2 showed you how to key in single-note music, and Part 3 showed how to do 3-part harmony. To recap briefly-

First, save yourself a lot of work by identifying any groups of notes in the sheet music that are repeated two or more times. Mark them off wherever they appear. Key them in first, starting with line number 500; at the end, put RETURN. If you find another such series, label it 600 and do the same; you may find several such series. Just stay below line number 1000, which is reserved for mergeable routines. Then, while you are programming the music and come to such a series of notes, just put in GOSUB 500 or whatever.

Start keying in your music in line 120; line 110 is reserved for a line to be merged in. To key in the music, just give T the number for the length of the first note, and give A, B and C the numbers for the melody and first and second part harmony. Then GOSUB 1000. For instance, T=1 :: A=23 :: B=18 :: C=12 :: GOSUB 1000 .

And for each succeeding note, give a new value to whatever changes; if T is still 1 and B and C are still the same, all you need is, for instance, A=19 :: GOSUB 1000.

Merge in one of the following routines, put in a line 999 STOP, and after every several notes enter RUN and listen to what you have done so far, to catch any errors while it is still easier to find them.

You can merge in any of the following routines to create many different musical effects. The D in line 110 controls the tempo of the music; change it as you wish. V1, V2 and V3 are the volume (loudness) of the three voices; adjust them as you like.

Key this in and save it by SAVE DSK1. .PLAY1,MERGE

```
110 D=500 :: V1=1 :: V2=5 :: V3=7 CALL SOUND(D*T,N(A),
V,N(B),V,N(C),V):: RETURN
```

That plays simple 3-part music, all at the same volume, which may sound rather harsh to your years. Try changing the second V to V2 and the 3rd one to V3. Save that as PLAY2.

For a bass accompaniment in the 3rd voice, change that to

```
CALL SOUND(D*T,N(A),V1,N(B),V2,N(C)*3.75,40,-4,V3)
```

For a bass melody with accompaniment, change the A to C, V1 to V3, C to A and V3 to V1.

For the melody in two voices two octaves apart, change the C back to A and the V3 back to V1. Are you beginning to see how many different effects can be created by making changes in just this one line? Save any ones you like in merge format with a different name for each.

Perhaps those bass notes sound too deep. Try changing the 3.75 in any of those routines to 7.5 . Better yet, change it to X and add :: X=3.75 to line 110. Then you can switch back and forth in your music by simply X=7.5 or X=3.75. Getting interesting, no?

Music played in that way has a strong throbbing beat, so try this method-

```
110 D=4 :: V1=1 :: V2=5 :: V3=7
1000 FOR J=1 TO T*D :: CALL SOUND(-4250,N(A),V1,N(B),V2,
N(C),V3):: NEXT J :: RETURN
```

I will be referring back to this one as the negative duration method. Again, you can change the tempo by changing the value of D, but sometimes not as exactly as I would like. With this method, you will find that a series of the same note runs together into a single long note. To avoid this, use different harmony notes each time, or different volumes for V2 and V3.

There is no law that says the harmony has to be lower than the melody, so try changing N(B) to N(B)*2 or even N(B)*4 or do the same with N(C), or both. Or, use *X, add X=1 to line 110, and then in the middle of your music program you can switch by X=2 or X=4 (do not try 3!)

For a vibrato effect, we alternate a note with the same note multiplied by 1.01-

```
1000 FOR J=1 TO T*D :: CALL SOUND(-4250,N(A),V1,N(B),V2,
N(C),V3):: CALL SOUND(-4250,N(A)*1.01,V1,N(B),V2,N(C),V3
):: NEXT J :: RETURN
```

For vibrato in the harmony rather than the melody, multiply N(C) or N(B), or both, by 1.01 instead- or multiply all three.

For a stronger vibrato, change the 1.01 to 1.02 or even 1.03. Of course, you can also multiply the harmony notes in both CALL SOUNDs by 2 or 4, as above. Or for a "chop" effect, multiply them in one CALL SOUND but not the other. The possibilities are almost endless!

For a tremolo, we alternate the volume rather than the frequency. Add X=3 to line 110 and use this routine-

```
1000 FOR J=1 TO T*D :: CALL SOUND(-4250,N(A),V1,N(B),V2,
N(C),V3):: CALL SOUND(-4250,N(A),V1+X,N(B),V2,N(C),V3)::
 NEXT J :: RETURN
```

You can vary the value of X as much as you want (V3+X cannot total more than 30) for any amount of tremolo from a flutter to a wobble or a stutter, and you can put the +X after V1 or V2 or all three. You can even change it in the middle of your music, by X= whatever you want. And you can multiply any or all by 1.01 for different combinations of vibrato and tremolo.

To enhance a note, play it twice in the CALL SOUND but multiply one of its voices by 1.01-

```
110 D=4 :: V1=1 :: V2=5 :: V3=7
1000 FOR J=1 TO T*D :: CALL SOUND(-4250,N(A),V1,N(A)*1.0
1,V1,N(B),V2):: NEXT J :: RETURN
```

Of course, with this trick you can only have 2-part harmony, but you can choose to enhance the harmony rather than the melody.

Now, try combining the enhanced note with the vibrato and/or tremolo, for many more effects. For enriched vibrato, use N(A),V1,N(A)*1.01,V1 in the first CALL SOUND and N(A)*1.01,V1,N(A)*1.02, V1 in the second.

The bass notes do not go well with this method because interrupting them through a loop introduces a rattle, but the baritone works well and gives a unique reedy sound. To do this, place the note you want in the 3rd position, multiply it by 7.5, give it a volume of 30, and add the -4 noise at whatever volume you want. You can also combine this with other effects, for instance with vibrato

```
1000 FOR J=1 TO T*D :: CALL SOUND(-4250,N(A),V1,N(B),V2,
N(C)*7.5,30,-4,V3)
1010 CALL SOUND(-4250,N(A)*1.01,V1,N(B),V2,N(C)*7.5,30,-
4,V3):: NEXT J :: RETURN
```

Now for the real fun- the "piano" effects that we get by decreasing the volume gradually. This is the basic routine-

```
1000 FOR J=1 TO T*D :: CALL SOUND(-4250,N(A),J+V1,N(B),J
+V2,N(C),J+V3):: NEXT J :: RETURN
```

Of course, with all of these you must also have that line 110 to define the duration and volume.

If you want a little more percussion in your piano, try this-

```
1000 FOR J=1 TO T*D :: CALL SOUND(-4250,N(A),J*1.5,N(B),
J*1.5,N(C),J*1.5):: NEXT J :: CALL SOUND(-4250,N(A),15,N
(B),15,N(C),15):: RETURN
```

And, of course, all those tricks we learned above- vibrato, tremolo, baritone, enhanced, high harmony, chop- can also be used with piano. This will give you the vibrato-

```
1000 FOR J=1 TO T*D :: CALL SOUND(-4250,N(A),J+V1,N(B),J
+V2,N(C),J+V3):: CALL SOUND(-4250,N(A)*1.01,J+V1,N(B),J+
V2,N(C),J+V3):: NEXT J :: RETURN
```

And an increasing tremolo-

```
1000 FOR J=1 TO T*D :: V=J*2 :: CALL SOUND(-4250,N(A),J,
N(B),J,N(C),J):: CALL SOUND(-4250,N(A),V,N(B),V,N(C),V):
: NEXT J :: RETURN
```

And just one more, the "reverse piano" with an increasing volume-

```
1000 FOR J=T*D TO 1 STEP -1 :: CALL SOUND(-4250,N(A),J+V
1,N(B),J+V2,N(C),J+V3):: CALL SOUND(-4250,N(A),J+V1,N(B)
,J+V2,N(C),J+V3):: NEXT J :: RETURN
```

By the time you get through exploring all the possible combinations of those, you should have a hundred ways of making music. Save each one you like, complete with line 110, in merge format, so you can try them all with each piece of music you create.

I had intended this to be the last part of this series, but I still have not told you about autochording, so there will have to be one more. o

continued from page 22

```
                    UNITWRITE(REMOUT,COMMAND,1,0,12);
                  END;
      END;
    UNTIL COMMAND[0]=UNSLAVE;
  END;

BEGIN
  buffsize:= 1 + ((varavail(res_segs)-slop) div 256);
  if varnew(buff,buffsize*256) = 0
    then
      begin
        writeln('program error allocating buffer');
        exit(program);
      end;
  WRITELN('REMTALK [',version,'] - press S(lave first');
  REPEAT
    WRITE('M(aster  S(lave  Q(uit ');
    CASE GETCHAR(['M','S','Q']) OF
      'M':BEGIN
            REPEAT
              WRITE('  S(end  R(eceive  Q(uit ');
              INCH:=GETCHAR(['S','R','Q']);
              CASE INCH OF
                'S',
                'R':DOCOMMAND(INCH);
                'Q':BEGIN
                      COMMAND[0]:=UNSLAVE;
                      UNITWRITE(REMOUT,COMMAND,82,0,12);
                    END;
              END;
              WRITELN;
            UNTIL INCH='Q';
          END;
      'S':DOSLAVECOMMANDS;
      'Q':EXIT(REMOTETALK);
    END;
    WRITELN;
  UNTIL FALSE;
END.
```

This tutorial was downloaded from TI-SOURCE BBS. o

continued from page 18

5) QUIT

SIDEWRITER 2.0 returns back to calling program.

NOTE: FILES CH1 TO CH5 ADDED BY S SHAW They are alternate character sets. RENAME THEM CHARA1 OR CHARA2 TO USE. Can also be used with TI Writer or Funlwriter.o

continued from page 14

```
270 A$="##"&CHR$(0)&"###"
280 PRINT #1,USING A$:A,B
```

CHR$(0) does not do anything on most printers, so the two fields will appear consecutive. Note that you could not do that in an IMAGE statement since you can only use characters and not expressions. The expression could have also been constructed within line 280 itself.

This should about exhaust the subject of USING except to remind you that you can also use DISPLAY USING if you need to format values somewhere else on the screen (by using the AT option) or if want to BEEP or clear the screen before doing it (ERASE ALL). o

Sidewriter Version 2.1

by Mauro Tomietto, Ottawa, Canada

Sidewriter 2.1 is a FAIRWARE print utility for Multiplan and TI-Writer. Sidewriter (SW) is written by Mauro Tomietto and is distributed by the Ottawa TI-99/4a User's Group. It is being made available via the Fairware concept. If you like the package and use it, send a donation of $5 to $10 to:

OTTAWA TI-99/4A USER'S GROUP
P.O. BOX 2144, STATION D,
OTTAWA, ONTARIO
CANADA K1P 5W3

Please feel free to distribute copies to your friends. If you have any suggestions for improvements or noted any errors please forward them to above address. Presently this program only works with EPSON or EPSON compatible printers. If you would like another printer configured for this program send all the appropriate escape codes (ie how to send graphic characters to the printer) and I will try to modify the program, provided no major changes are needed.

RUNNING SIDEWRITER 2.1:

To run SW select option 5 on Editor/Assembler module or Utilities option from TI-Writer module and type DSKn.SW1 (or use any other suitable memory image file loader) SW requires 32K memory expansion.

SW is contained in three linked files on the system disk called SW1, SW2, and SW3. SW only works with Epson Compatible printers that use escape codes ESC K+n1+n2 or ESC *+n+n1+n2 for printing in graphics mode. Check your printer manual. SW expects to load Alternate character sets from DSK1. SW is a self-contained program with its own support routines included:

i) Viewing or printing (the regular way) of Spreadsheet or Wordprocessor files
ii) Allows on-line cataloguing of files
iii) Prints any DIS/VAR 80 file SIDEWAYS using variable margins, line spacing, character spacing, with up to four separate character sets
iv) Brief helpful instructions and defaults with every prompt or for more information allows viewing or printing of the help file from Main Menu

With Spreadsheets, printing SIDEWAYS allows chaining of Print File pages from Multiplan for one continuous form. No more staples, glue or tape. With TI-Writer or any other text processor pages can be printed in 2 columns for an attractive book-type look.

FEATURES OF SIDEWRITER

1) VIEW/PRINT TEXT(HELP) FILE:

The Sidewriter help file is a text file called SIDEHELP contained on the system disk and can be viewed using this option. Using an extension of this option the file can also be printed. Viewing occurs screen by screen as as SW loads each page individually. If the printing option is engaged SW will close the file and then reopen it to begin printing from the top of the file.

The real power of this option can be realized by noting that in actuality any text file (ie dis/var 80) can be viewed or printed. Pressing "CLEAR" stops print.

2) CATALOG DISK:

The source code for this program can be found on the system disk. "CAT" allows a maximum number of files to be catalogued on the screen at one time. For multiple screen catalogs (ie. more than 23 files) the individual screens can be selected by simply pressing the appropriate page number. The source code itself is valuable learning tool for routines on sector accessing, number conversions, and overall assembly language programming.

3) LOAD ALTERNATE CHARACTER SET:

This option loads a character set which has lower case descenders unlike that in the TI GROM. This option is sometimes desirable when printing with the SIDE option. SW gets its character patterns from the Pattern Descriptor Table in VDP RAM and therefore would normally use TI's small capital set. However in some cases such as printing out spreadsheets small capital characters actually look attractive. Another character set could be used provided the program is a display/fixed 80 file using a filename of DSK1.CHAR and autorunning when loaded. SW will look for DSK1.CHAR on the system disk so before selecting new characters make sure the system disk is in drive 1. Version 2.1 now loads the Funlwriter/TI-Writer alternate character sets CHARA1 and CHARA2. CHARA1 is quite useful set for printing using 158 character spacing. It gives the impression of printing in compressed print in the SIDEWAYS direction. CHARA2 is also an attractive character set for those that do not want the small size.

4) SIDE PRINT OPTIONS:

This is really the backbone of SW. Most of the menu options and the future options which will be incorporated later (more on this later) are actually supplements to the body of this program. In order to print 'SIDEWAYS' many suboptions are available. These suboptions allow one to match PRINT FILE options from MULTIPLAN, or to configure a text (TI-WRITER) file.

i) SW allows the loading and printing of any DIS/VAR 80.
ii) In order to print out the data, set up your printer up for graphics type dump (ie PIO.CR or RS232.....).
iii) Presently SW supports EPSON compatible printers with two types of graphic dot densities.

Most EPSON capable of 480 dots per line (dpl) in single density mode. They use a 'ESC K+n1+n2' sequence to enter this mode. If your printer is capable of this (set your printer manual) then select "1" when prompted. Other EPSON compatible printers have 640 dpl for the same sequence in single density mode, most notably the popular Mannesman Tally Spirit 80 printer comes to mind. Choose "2" when prompted if you have this type of printer.

However some of the newer Epson compatibles printers have new selectable graphic densities selectable by control codes. Using a code such as 'ESC *+n+n1+n2' a graphic density of 640 dpl can be achieved. This makes the Epson compatible basically equivalent to the Spirit 80 with respect to the source code needed which effectively gives the Epson the ability to use the allowed lines per side page as the Spirit 80. If your printer is capable of this mode choose "3".

iv) PRINT FILE OPTIONS:

These options allow one to match Print file options used in Multiplan Print File Command, or to configure the printout for text files. However one main drawback is the fact the Print File commands should be printed to disk from Multiplan and TI-Writer, and must have been configured with SW in mind. SW allows for some variability in formatting but not all.

Allowable left margin is 0-40, and allowable right margin is 40-80. Use these variables to configure your printing or to align your spreadsheets.

For example:-

Your spreadsheet has 13 columns; one 15 characters, and 12 ten character wide columns (like for twelve months of the year). You want to print the file sideways and concatenate the two pages. The total number of characters wide is 135. Print file from MULTIPLAN groups by 80 character wide page so you will want to align your field to match this. If Print File from MULTIPLAN is used with LM=5, MULTIPLAN will print,

on the first page, a 5 character wide blank column, followed by one 15 characters wide and 6 columns of 10 characters wide (5+15+6(10)=80). On the second page it will print the 5 character wide blank column and the remaining 6 columns of 10 (5+6(10)=65). Using SIDEWAYS function with LM=5 and RM=80 you will find that the printout will concatenate properly. similarity with TI-Writer a text file must be saved to disk with the control codes erased (ie. C DSK1.filename). With the margins of the text file set at Left margin=10 and Right margin=70 it is possible to get a side printout in two columns attractively set out on the page. The file when printed will be formatted with 10 spaces, 60 characters of printed material and another 10 spaces. To print these pages load SIDEWRITER 2.0, and load the alternate character set which will give the printout lower case characters with true descenders. Select the SIDEWAYS option, and choose the file you want to load. In the PRINT FILE options select LM=7, RM=73 and choose 132 characters per side page. The 8 1/2 x 11 inch page will have a printout in two columns rotated 90 degrees, and formatted 3 spaces, 60 columns text print, 3+3 spaces between the two columns, 60 more columns of text print and 3 spaces to end of page. To get even more text into a page in a legible fashion choose the alternate character set and 158 characters/side page.

The page length option allows one to have 9 defined page lengths. SW will automatically space the number of lines accordingly. Page length will printers capable of 480 dpl using ESC K n1+n2 can use 60, 53,48, 43, 40, 36, 34, 32, and 30 lines per page. Most EPSON compatible printers fall into this bracket. Other EPSON compatible printers have 640 dpl using the same escape code, or in the newer versions, using the escape code ESC *n+n1+n2. They are capable of 80,. 71, 64, 58, 53, 49, 45, 42, 40 lines per side page.

In order to get MULTIPLAN print files properly it is very important that one of these page lengths are chosen when printing a MULTIPLAN print file to disk! When printing normal text files in the same Side format the proper print file length is not important.

This help file has margins at 10 and 70. Set top of page on the printer and try printing 2 pages of this file using 132 char/side page. Set the form length at 53 LM=8, and RM=73.

v) PAGES TO PRINT:

Number of page for SW to print sideways is 0-9. A zero selects print till end of file. SW will load each page separately, the number of lines loaded is determined by page length.

vi) CHARACTERS/SIDE PAGE:

Not only line spacing can be changed but spacing between characters can also be varied. Four character spacings are allowed (labelled 1-4). Selections 1-4 represent character spacing of 99, 113, 132, and 158 respectively. Character spacing "1" is the largest spacing between characters while spacing "4" is the smallest spacing (approximately 1 pixel between each character). In this way one can select the character width of the side page (ie the longest side of page) from 99-158 characters. If "4" is selected then it is advised that the alternate character set is loaded. The characters in this character set are slightly narrower which make the printout more legible.

Once all the options are selected SW will open the file from disk and printer, load in one page of the file and print it out. Pressing <CLEAR> stops printing and branches back to first SIDEWAYS prompt.

WARNING!! When the printer is halted in this way the printer crashes (will not go off line). At this point it is necessary to turn the printer off and then on again.

continued on page 16

Let's Round Up the Mavericks

by Jim Peterson, Tigercub Software, USA

A maverick, for the information of you tenderfeet, is a young Texas critter which has lost its mama. There are over a million of them hiding in the closets of America (and a fair few in Australia!) and I think it is time for a roundup!

There are perhaps 200, possibly 300, TI user groups in the United States and elsewhere in the world. A few boast of several hundred members, but some have no more than a dozen, and I doubt that the average is more than 50 users actually paying dues and attending meetings. That computes to at most 15,000 members of the "organized" TI world. Of course, there are many others who keep in contact by subscribing to those magazines which support the TI, and still others who are kept up to date on new developments by the catalogs from the big mail order houses. Still, no matter how you compute it, there are certainly well over a million owners of the TI-99/4A who have no way of knowing that our computer is still alive and well.

These people have read that Texas Instruments abandoned the computer. They have seen the supplies of hardware and software disappear from the big retail stores. Many of them bought their computer during the final suicide sales, therefore never got on the mailing list for the Texas Instrument newsletter.

And yet, relatively few of the TI-99/4A are showing up in the classified ads and in the garage sales. A recent national survey found that the TI-99/4A was owned by more people than any computer except the Commodore.

True, many of these owners are only interested in plugging in a module and playing a game. But some have a deeper interest - and even five percent of a million is an awful lot of people!

When I bought my TI, in March of 1982, I searched in vain through the articles and ads of every magazine on the newsstand, for anything relating to my computer. It almost seemed that there was a conspiracy of silence. I had taught myself to program, and written dozens of programs, before I finally made contact with the TI world. I was once a maverick, and I can sympathize with those who are mavericks now.

Is your user group dwindling away, as some of your members move on to bigger but not necessarily better computers, while others become so polarized in their interests that they have little in common with each other? Are your givers tired of giving to your getters, and your doers tired of being used by your users? Do you miss the enthusiasm and excitement of your first meetings, when everyone was learning together? Does your group need a transfusion of fresh blood? The donors are out there and waiting, if you can find them!

Do you want to see new hardware, new software, new publications for your computer? The bigger the market, the more that will be produced to be marketed. And the market is there - it just does not know that it is there!

The user groups are the only ones who can round up the mavericks. You can do it by publicizing your meetings, by letting the TI owners in your community know what you can do for them. You can get newspaper publicity and television publicity. Some of you are already offering classes in programming or in computer use to the general public, to the schools, to libraries, to senior citizens, to foster children, to the handicapped. These are very fine endeavours in themselves, and they can also bring the publicity which will attract new members. And here and there among those new members will be an ingenious hardware hacker or programming genius who will make our computer better than ever.

To See or Not to C

by Geoff Trott

I was pondering what I should talk about to continue this series when I read an article in the "Front Range 99ers" about finding a file whose name you are not too sure of. Have you ever tried to find a file whose name you cannot quite remember? Perhaps you are sure it has "DOC" or ";C" somewhere in the name but you cannot remember exactly what the name is. Of course you can load in your favourite Disk Manager program and look through a number of disks at all the files and try to pick out the one you are looking for. The program in the article allowed you to put in a string and it would then look in all the drives connected to the computer for all files containing this string. The program was written in Extended BASIC and was relatively simple, but I thought this would be a good program for conversion to C as it would introduce the concepts of accessing files as well as using the string routines we have looked at so far in the series of articles. It would also execute faster than the BASIC program, which is a bit slow. It would be a particularly useful program for owners of hard disks, where it is easy to lose files.

To plan the program operation, consider the following scenario. First, select the disk units whose contents are to be examined. If you only have one drive, this is easy. However you may have several drives, RAMdisks and hard disks. This selection should involve as little typing as possible and should not require the program to be re-written to change. The string should then be entered in response to a prompt. Then each drive should be opened and the file names examined to see if they contain the string. This can be done with the index() routine. If the string is present, the name of the file is sent to the screen along with the disk name. If any sub-directories are found (mainly on hard disks), these are stored at the end of the disk name list so that once all the disk names have been searched, the sub-directories are also searched.

The only difficult areas are reading the catalogue of each disk and sub-directory and setting up the list of disk names and sub-directory names so they can be accessed one at a time. The first of these requires a bit of fiddling with the disk access mechanisms available in c99 while the second requires storing strings in a way that they are sequentially accessible and do not take up too much memory. I have only looked at the first area so far and I will leave the second area until next month, when the whole program should be complete.

How to read the catalogue of a disk? In the disk manual (Disk Memory System) it tells us that the catalogue file is "an unnamed, INTERNAL-format, FIXED-length file". An example BASIC statement to open this catalogue file is:

```
100 OPEN #1:"DSK1.",INPUT,RELATIVE,INTERNAL
```

The disk name ends with the "." and each record in the catalogue file contains four items: one string and three numbers returned in internal floating point format (normal BASIC format). There are exactly 128 records in the file numbered from 0 to 127. These correspond to the disk name and the 127 possible file names on the disk. The string, which is either the disk name or the file names, may be up to 10 characters in length. A floating point number takes 8 bytes to store so the maximum length of all the information is 10 + 3 * 8 = 34 bytes, plus any length bytes. In fact there are 4 length bytes, making a total of 38 bytes for each record. The first thing I decided to do was to see if c99 could read this catalogue file and what form the data read would take. I also needed to make sure that the program would handle RAMdisks and hard disk controllers as well as the more common disk controllers.

The standard release of c99 has file operations but only for display type files. I tried to use these by using display fixed mode. This worked fine for the RAMdisks and returned data which showed me that the format of the data was as follows:
length byte, string, length byte, number, length byte, number, length byte, number.
The length bytes for the numbers always contain 8 as there are 8 bytes in each number. The length byte of the string depends on the number of characters in the string. The output of the catalogue file from the RAMdisks also stopped after one empty file name indicating that the RAMdisk returned an end of file code at this time. This did not work with the Myarc hard disk controller, however, with that controller not opening the catalogue file at all as display fixed.

This meant that I had to use the Tom Bentley file I/O routines which do support Internal type files. This program is the one given in this article. The problem with this is that now printf() does not appear to work so that I have to use the more simple puts() for output and to use functions like itox() to convert the numbers into hexadecimal characters for output. This function is only used during the exploratory phase of the programming until I am sure I know what the data looks like. Using the internal format causes the first length byte to be lost (in an attempt to convert BASIC strings to C strings which do not have a length byte) which will cause a small problem. Another annoyance is that the Myarc controller says that it has returned 145 bytes for each record, while the RAMdisks return the correct number (depending on name length) and the Myarc controller does not return an end of file until it has returned all 128 records.

The topen() command uses a zero size to allow the system to define the size of the file. With the advantage of hind-sight, the number 38 could be used here. Each record is read into a buffer and then the contents of the buffer are output in hexadecimal to enable the contents to be examined. A routine to convert an integer to a hexadecimal string of characters was written and is included at the end of the program. The Script-loader file is also included. I called the main program FIND;C and the Script-loader file FIND;T.

```
#include "DSK1.CONIO"
#include "DSK1.TCIOI"
extern printf;
main()
{
  char filename[16], buffer[256], str[16];
  int fp, eof, rec, rsiz, *pbuf;
  putchar(FF);
  puts("Type in disk name ");
  gets(filename);
  fp = topen(filename,INPUT+RELATIVE+FIXED+INTERNAL, 0);
  if (fp < 1) {
    puts("\nBad disk name!!\n\n");
    puts("file status = ");
    puts(itox(fp, str, 5));
    puts("\n");
    exit(0);
  }
  putchar(FF);
  puts("Valid disk name\n");
  eof = tread(buffer, RELSEQ, fp, &rsiz);
  while (eof != TEOF) {
    puts(itox(rsiz, str, 5));
    for (rec = 0, pbuf = buffer; rec<19; rec++, pbuf++)
      puts(itox(*pbuf, str, 5));
    puts("\n");
    for (rec = 0, pbuf = buffer; rec<19; rec++, pbuf++)
      *pbuf = -1;
    eof = tread(buffer, RELSEQ, fp, &rsiz);
  }
  tclose(fp);
}
/*
** s = itox (nbr, str, sz)
**
** convert nbr to hexadecimal string of width sz
** right justified, blank filled, result in str[].
** sz includes NULL string terminator.  Returns str.
*/
itox (nbr, str, sz)
int nbr, sz;
```

continued on page 6

Writing in Machine Code

by J.E. Banfield

Writing to VDP (continued)

The TMS9900 is a powerful microprocessor, quite advanced for its time, but was dramatically under-utilized in the TI99/4A. Its autoincrementing addressing mode is valuable in many ways (one of these is exemplified in the 7C60 routine given in my previous contribution number 6). This use enables data to be passed to a subroutine by writing it after the calling instruction. In this case the calling routine is at 7A46.

```
7A46:   06 A0       JSP @.+2
7A48:   7C 60
7A4A:   41 10       data, may be altered
```

The JSP, jump and save program counter, moves the current contents of the program counter, incremented by 4 and thus reading 7A4A, to accumulator Ac B (R11) and inserts the value 7C60 into the program counter so that control passes to the instruction at that address (JUMP). So the next instruction to be executed is:

```
7C60:   C0FB
```

i.e.

```
1 1 0 0   0 0 0 0   1 1 1 1   1 0 1 1
\_____/   \_____________/ \_____________/
   C            03             3B
```

The opcode C is MOVE. The source address is address mode 3 and refers to Ac B. The "3" mode is indirect autoincrementing so that the data is taken from the address given in Ac B; that is address 7A4A (which contains the data 4110). After this procedure the contents of Ac B is increased by two (since the move was not a byte instruction) so that it now reads 7A4C and now points to the instruction after the DATA. The destination address is Accumulator 3 so that the value 4110 is transferred to Ac 3.

For economy, the DATA word contains two pieces of information, each in 8 bit bytes. The program now separates this data into these components.

```
7C62:   C1 03       MOVE 4,3
```

D and S both being accumulator mode. This duplication is needed to avoid destructio of data in later instructions.

```
7C64:   02 43
   0 0 0 0   0 0 1 0   0 1 0 0   0 0 1 1
   \_____________________________/ \_____/
                 024                   3
```

The opcode 024 is ANDI, AND immediate and its address mode is simple accumulator (source and destination) with the immediate data in the word following the instruction.

```
7C66:   FF 00
```

The AND operation was described in the last contribution and in this case selects the left hand 8 bit byte (a sharp observer will notice that this instruction is not really necessary since these bits will be selected in the MOVB, move byte, instruction later). Thus:

```
Ac 3:   41 10      is ANDed with   FF 00     thus:
        0 1 0 0   0 0 0 1   0 0 0 1   0 0 0 0
        1 1 1 1   1 1 1 1   0 0 0 0   0 0 0 0
        ---------------------------------------
        0 1 0 0   0 0 0 1   0 0 0 0   0 0 0 0
```

and this result remains in Ac 3.

The next instruction (and this is necessary) selects the right hand 8 bits of Ac 4.

```
7C68:   02 44       ANDI 4,
7C6A:   00 FF
        0 1 0 0   0 0 0 1   0 0 0 1   0 0 0 0
        0 0 0 0   0 0 0 0   1 1 1 1   1 1 1 1
        ---------------------------------------
        0 0 0 0   0 0 0 0   0 0 0 1   0 0 0 0
```

and this result, in fact a count of 10 (16 in decimal) remains in Ac 4.

We now enter a loop:

```
7C6C:   D8 03
   1 1 0 1   1 0 0 0   0 0 0 0   0 0 1 1
   \_____/   \_____________/ \_____________/
      D             20              03
```

The opcode D, MOVB, moves a byte from the source address, which is Ac 3 and since the accumulator addressing mode points to the left hand byte in the accumulator, the data there, 41, is selected. The 20 address mode, as we have seen many times before, takes the address of the destination from the next word.

```
7C6E:   8C 00
```

to which the byte 41 is transferred. From Table 6-1 (previous contribution) we see that this address, with A14, the MODE, being low, corresponds to VDP Data Write; so that 41, ASCII for "A", is written to the current VDP address, which is autoincremented.

```
7C70:   06 04       SOS 4
```

subtracts one from the value in Ac 4 which, first time around the loop, now reads 00 0F. Status condition bits are set in the 9900 CPU so that a conditional branch, the next instruction, will read those conditions.

```
7C72:   16 FC
```

The opcode 16, SKIPNE, will skip the count (signed) number of instructions from the next if the condition "not equal" is true (and since the result of the SOS was 0F, the skip of FC instructions (minus 4) is made).

If you have a Casio fx-100B or similar calculator, you can do the following address calculation:
Enter FFFFFFFC, x 2 = then + 7C74 (the address of the next instruction now in the program counter), then = to bring up the address 7C6C to which the skip is made. Or, you can count back 4 instructions; easy to do in this case.

So the first loop is completed and so it goes until Ac 4 is reduced to zero. At this time the skip condition no longer being true, the next instruction is executed.

```
7C74:   04 5B       JUMPA @B
   0 0 0 0   0 1 0 0   0 1 0 1   1 0 1 1
   \_____________________/ \_____________/
              044                 1B
```

Thus the contents of Ac B, now 7A4C, are transferred to the program counter and the return from this subroutine is completed.

Colour Table

The VDP processor has several modes as set by the contents of VDP registers. We will not pursue these complex matters at this stage, except to demonstrate how the colour table can be changed. Each byte in the colour table contains two 4 bit nybbles; the most significant (left) four bits defines the colour of the 1's, the right nybble the colour of the 0's, as given in Table 2-4 of the VDP Processor Manual. (Note that the colour definition table for BASIC differs in that the codes are one more than in the VDP processor manual; black is 2 in BASIC, 1 in the VDP register.) For the current VDP mode, as set by the MiniMemory module, the colour table is 20 bytes long and starts at VDP address 380 (all numbers in hexadecimal of course).

If the programs described in contribution #6 are still resident in MiniMemory, then type:

```
M7A4A   enter
M7A4A = 41   type 91 enter
M7A4B = 10   type 20 enter
             type .
```

and also type:

```
M7A44   enter
M7A44 = 40   type 43 enter
M7A45 = E3   type 80 enter
             type .
```

Then execute 7A40 by typing E7A40 enter. Change the byte at 7A4A to run other values and observe the results when the program is re-run. O

continued from page 10

I hope I cut through some of the stuff TI threw at us in the manual. Note that the section on MAGNIFY is wrong, and that they mixed up magnification factors 2 and 3. There may be other manual errors, but none that I could find out about (by the way, the Extended Basic manual WAS laced with errors. Some of us have an addendum about these problems. TI may still let you have a copy.). O

RemTalk

UCSD Pascal Utility

This program will allow any two P-Systems to transfer files via RS232 cable not over the phone,IBM to TI-99 uses a straight cable 1-8 20-20 The others I cannot help you with maybe someone can help with this.

Source and object code for the REMTALK utility are provided in "as is" condition. No warranty is made, particularly with respect to fitness for a particular purpose.

Copies of source and object code for the REMTALK utility may be used for any lawful purpose providing each copy retains all markings and legends that appear on or in the source and object code items. Failure to include such markings and legends is a violation of U.S. Copyright Laws.

```
(*$R-,I-*)
(*NATIVE*)
(*$L PRINTER:*)
PROGRAM REMOTETALK;

CONST version='IV.0 a1';
      res_segs='fileops,pascalio,extraio,heapops';
                                {resident segments}
      slop=2000; {extra slop for buffer allocation}

      REMIN=7;
      REMOUT=8;
      FINALBLOCK=50;
      NOTLASTBLOCK=51;
      SENDAGAIN=52;
      SENDNEXT=53;
      ABORT=54;
      UNSLAVE=55;
      CLOSEFILE=56;
      RECEIVEFILE=57;
      SENDFILE=58;
      OPENFILE=59;
      FILEOPENED=60;
      BUMFILE=61;
      FILECLOSED=62;

TYPE BYTE=0..255;
     BLOCK=PACKED ARRAY[0..511] OF BYTE;
     TWOBYTES=PACKED ARRAY[0..1] OF BYTE;
     SETOFCHAR=SET OF CHAR;
     BLOCKARRAY=ARRAY[0..0] OF BLOCK;
     BLOCKPTR=^BLOCKARRAY;

VAR BUFF,FOON:BLOCKPTR;
    PACKBLOCK:BLOCK;
    FILENAME:STRING;
    INCH:CHAR;
    F:FILE;
    COMMAND:PACKED ARRAY[0..81] OF BYTE;
    FIRSTBLOCK,UNITNUM,LASTBLOCK,UNOCNTR,BUFFSIZE:
                                          INTEGER;

  PROCEDURE SIGNAL(COMMAND:INTEGER);
  VAR WART:TWOBYTES;
  BEGIN
    WART[0]:=COMMAND;
    UNITWRITE(REMOUT,WART[0],1,0,12);
  END;

  FUNCTION WAIT:INTEGER;
  VAR WART:TWOBYTES;
  BEGIN
    UNITREAD(REMIN,WART[0],1,0,12);
    WAIT:=WART[0];
  END;

  PROCEDURE UNO(CH:CHAR);
  BEGIN
    UNOCNTR:=UNOCNTR+1;
    WRITE(CH);
    IF UNOCNTR=40 THEN
    BEGIN
      WRITELN;
      UNOCNTR:=0;
    END;
  END;

FUNCTION GETCHAR(OKSET:SETOFCHAR):CHAR;
VAR CH:CHAR;
BEGIN
  REPEAT
    READ(KEYBOARD,CH);
    IF CH IN ['a'..'z'] THEN
      CH:=CHR(ORD(CH)-ORD('a')+ORD('A'));
  UNTIL CH IN OKSET;
  WRITELN(CH);
  GETCHAR:=CH;
END;

PROCEDURE RECEIVEIT;
VAR INBLOCK:PACKED ARRAY[0..1025] OF BYTE;
    JUSTONE:TWOBYTES;
    BADOUTPUT:BOOLEAN;
    BYTENUM,CHECKSUM,BUFFPTR,BYTE0,BYTE1,ANSWER:
                                          INTEGER;

  FUNCTION PUTBLOCK(VAR ONEBLOCK:BLOCK):BOOLEAN;
  BEGIN
    PUTBLOCK:=TRUE;
    BUFF^[BUFFPTR]:=ONEBLOCK;
    BUFFPTR:=BUFFPTR+1;
    IF BUFFPTR=BUFFSIZE THEN
    BEGIN
      PUTBLOCK:=BLOCKWRITE(F,BUFF^,BUFFSIZE)=BUFFSIZE;
      BUFFPTR:=0;
    END;
  END;

BEGIN
  BUFFPTR:=0;
  UNOCNTR:=0;
  BADOUTPUT:=FALSE;
  REPEAT
    ANSWER:=WAIT;
    IF ANSWER=NOTLASTBLOCK THEN
    BEGIN
      UNITREAD(REMIN,INBLOCK,1026,0,12);
      CHECKSUM:=0;
      FOR BYTENUM:=0 TO 511 DO
      BEGIN
        BYTE0:=ORD(ODD(15) AND ODD(INBLOCK[BYTENUM+
                                          BYTENUM]));
        BYTE1:=ORD(ODD(15) AND ODD(INBLOCK[BYTENUM+
                                        BYTENUM+1]));
        PACKBLOCK[BYTENUM]:=BYTE0*16+BYTE1;
        CHECKSUM:=CHECKSUM+BYTE0+BYTE1;
      END;
      IF CHECKSUM=ORD(ODD(127) AND ODD(INBLOCK[1024])
                                          )*128+
        ORD(ODD(127) AND ODD(INBLOCK[1025])) THEN
      BEGIN
        IF PUTBLOCK(PACKBLOCK) THEN
        BEGIN
          UNO('.');
          SIGNAL(SENDNEXT);
        END ELSE
        BEGIN
          BADOUTPUT:=TRUE;
          SIGNAL(ABORT);
        END;
      END ELSE
      BEGIN
        UNO('?');
        SIGNAL(SENDAGAIN);
      END;
    END ELSE
      IF ANSWER=ABORT THEN
        BEGIN
          WRITELN;
          WRITE(' ERROR in input file');
        END;
  UNTIL ANSWER IN [FINALBLOCK,ABORT];
  BADOUTPUT:=BADOUTPUT OR (BLOCKWRITE(F,BUFF^,BUFFPTR
  CLOSE(F,LOCK);                          )<>BUFFPTR);
  IF (IORESULT<>0) OR BADOUTPUT THEN
  BEGIN
    SIGNAL(ABORT);
    WRITELN;
    WRITE(' ERROR in output file');
```

```
  END ELSE
    SIGNAL(FILECLOSED);
END;

PROCEDURE SENDIT;
VAR ANS,BYTEO,BYTE1,BYTENUM,CHECKSUM,BLOCKSREAD,
     BUFFPTR:INTEGER;
    BADINPUT:BOOLEAN;
    UNPACKBLOCK:PACKED ARRAY[0..1023] OF BYTE;
    JUSTTWO:TWOBYTES;

  FUNCTION GETBLOCK(VAR ONEBLOCK:BLOCK):BOOLEAN;
  BEGIN
    BUFFPTR:=BUFFPTR+1;
    IF BUFFPTR>=BLOCKSREAD THEN
    BEGIN
      BLOCKSREAD:=BLOCKREAD(F,BUFF^,BUFFSIZE);
      BADINPUT:=IORESULT<>0;
      BUFFPTR:=0;
    END;
    GETBLOCK:=(BLOCKSREAD<>0) AND (NOT BADINPUT);
    ONEBLOCK:=BUFF^[BUFFPTR];
  END;

BEGIN
  BADINPUT:=FALSE;
  UNOCNTR:=0;
  BUFFPTR:=-1;
  BLOCKSREAD:=0;
  ANS:=SENDNEXT;
  WHILE GETBLOCK(PACKBLOCK) AND (ANS<>ABORT) DO
  BEGIN
    CHECKSUM:=0;
    SIGNAL(NOTLASTBLOCK);
    FOR BYTENUM:=0 TO 511 DO
    BEGIN
      BYTEO:=PACKBLOCK[BYTENUM] DIV 16;
      UNPACKBLOCK[BYTENUM+BYTENUM]:=BYTEO;
      BYTE1:=ORD(ODD(PACKBLOCK[BYTENUM]) AND ODD(15));
      UNPACKBLOCK[BYTENUM+BYTENUM+1]:=BYTE1;
      CHECKSUM:=CHECKSUM+BYTEO+BYTE1;
    END;
    UNITWRITE(REMOUT,UNPACKBLOCK,1024,0,12);
    JUSTTWO[0]:=CHECKSUM DIV 128;
    JUSTTWO[1]:=ORD(ODD(CHECKSUM) AND ODD(127));
    UNITWRITE(REMOUT,JUSTTWO,2,0,12);
    ANS:=WAIT;
    CASE ANS OF
      SENDNEXT :UNO('.');
      SENDAGAIN:BEGIN
                  BUFFPTR:=BUFFPTR-1;
                  UNO('?');
                END;
    END;
  END;
  CLOSE(F);
  IF BADINPUT THEN
  BEGIN
    WRITELN;
    WRITE(' ERROR in input file');
    SIGNAL(ABORT);
  END ELSE
    SIGNAL(FINALBLOCK);
  IF WAIT<>FILECLOSED THEN
    BEGIN
      WRITELN;
      WRITE(' ERROR in output file');
    END;
END;

PROCEDURE DOCOMMAND(SENDORRECEIVE:CHAR);
VAR CH:CHAR;
    I,TRANSFERUNIT:INTEGER;
    ANSWER:TWOBYTES;
    S:STRING;
BEGIN
  FILLCHAR(COMMAND,82,0);
  IF SENDORRECEIVE='S' THEN
  BEGIN
    COMMAND[0]:=SENDFILE;
    REPEAT
      WRITE('  Send what file? ');
      READLN(S);
      IF LENGTH(S)=0 THEN
        EXIT(DOCOMMAND);
      RESET(F,S);
    UNTIL IORESULT=0;
    REPEAT
      WRITE('  Send to what remote file? ');
      READLN(S);
      IF LENGTH(S)=0 THEN
      BEGIN
        CLOSE(F);
        EXIT(DOCOMMAND);
      END;
      FOR I:=0 TO LENGTH(S) DO
        COMMAND[I+1]:=ORD(S[I]);
      UNITWRITE(REMOUT,COMMAND,82,0,12);
    UNTIL WAIT=FILEOPENED;
    SENDIT;
  END ELSE
  BEGIN
    REPEAT
      WRITE('  Receive what remote file? ');
      READLN(S);
      IF LENGTH(S)=0 THEN
        EXIT(DOCOMMAND);
      COMMAND[0]:=OPENFILE;
      FOR I:=0 TO LENGTH(S) DO
        COMMAND[I+1]:=ORD(S[I]);
      UNITWRITE(REMOUT,COMMAND,82,0,12);
    UNTIL WAIT=FILEOPENED;
    REPEAT
      WRITE('  Write to what file? ');
      READLN(S);
      IF LENGTH(S)=0 THEN
      BEGIN
        COMMAND[0]:=CLOSEFILE;
        UNITWRITE(REMOUT,COMMAND,82,0,12);
        EXIT(DOCOMMAND);
      END;
      REWRITE(F,S);
    UNTIL IORESULT=0;
    COMMAND[0]:=RECEIVEFILE;
    UNITWRITE(REMOUT,COMMAND,82,0,12);
    RECEIVEIT;
  END;
END;
PROCEDURE DOSLAVECOMMANDS;
VAR I:INTEGER;
    S:STRING;
BEGIN
  REPEAT
    UNITREAD(REMIN,COMMAND,82,0,12);
    FOR I:=0 TO COMMAND[1] DO
      S[I]:=CHR(COMMAND[I+1]);
    WRITELN;
    CASE COMMAND[0] OF
      CLOSEFILE  :CLOSE(F);
      SENDFILE   :BEGIN
                    REWRITE(F,S);
                    IF IORESULT=0 THEN
                    BEGIN
                      WRITE('Opening new file: ',S);
                      COMMAND[0]:=FILEOPENED;
                    END ELSE
                    BEGIN
                      WRITE('ERROR opening new file:
                                                  ',S);
                      COMMAND[0]:=BUMFILE;
                    END;
                    UNITWRITE(REMOUT,COMMAND,1,0,12);
                    WRITELN;
                    IF COMMAND[0]=FILEOPENED THEN
                      RECEIVEIT;
                  END;
      RECEIVEFILE:SENDIT;
      OPENFILE   :BEGIN
                    RESET(F,S);
                    IF IORESULT=0 THEN
                    BEGIN
                      WRITE('Opening old file: ',S);
                      COMMAND[0]:=FILEOPENED;
                    END ELSE
                    BEGIN
                      WRITE('ERROR opening old file:
                                                  ',S);
                      COMMAND[0]:=BUMFILE;
                    END;
```

continued on page 16

6. 11, 12, 13, 14 13, 14, 15, 16

Regional Group Reports

Meeting Summary for November

Banana Coast	8/11/92	Sawtell
Central Coast	14/11/92	Saratoga
Glebe	12/11/92	Glebe
Hunter Valley	14/11/92	
Illawarra	16/11/92	Keiraville
Liverpool	13/11/92	
Northern Suburbs	26/11/92	
Sutherland	20/11/92	Jannali

BANANA COAST Regional Group
(Coffs Harbour and Environs)

We never miss meeting at Kerry Harrison's residence 15 Scarba St. Coffs Harbour, 2 pm second Sunday of the month. Visitors are most welcome. Contact Kerry 52 3736, Kevin 53 2649, Rex 51 2485 or John 54 1451.

CENTRAL COAST Regional Group

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

GLEBE Regional Group

Regular meetings are normally on the Thursday evening following the first Saturday of the month, at 8pm at 43 Boyce Street, Glebe. Contact Mike Slattery, (02) 692 8162.

HUNTER VALLEY Regional Group

The meetings are usually held on the second Saturday of each month at members homes starting at 3:15 pm. Check the location with Geoff Phillips on (049) 428 176. Note that after 9:00 pm this number is used for the ZZAP BBS which includes TI-99 information. Geoff.

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, at the home of Geoff & Heather Trott, 20 Robsons Road, Keiraville. 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 Group

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 .CE2

TIsHUG in Sydney

Monthly meetings start promptly at 2pm (except for full day tutorials) on the first Saturday of the month except for January. They are held at the RYDE INFANTS SCHOOL, Tucker Street (Post Office end), Ryde. Regular items include news from the directors, the publications library, the shop, and demonstrations of monthly software.

****

The cut-off date for submitting articles to the Editor for the TND via the BBS or otherwise is:

December 15 November

This is a Sundays and there is no guarantee that they will make the magazine unless they are uploaded by 6:00pm, at the latest. Longer articles should be to hand well before the aboves dates to ensure there is time to edit them.

****

TIsHUG Meetings for Sydney

November

The TI-Faire will be a few weeks after this meeting so it may be taken up with the organizational requirements of this big day. New software and hardware to be demonstrated. Watch this space for more details. Time to think about nominating for positions on the board. I am sure there will be some vacancies this year!

December

The Annual General Meeting followed by some festive eats and drinks, There will probably be a bit of celebration after the TI-Faire, if we are all still friendly after the event. Make sure that you attend and give your support to all the workers in the club. O

continued from page 8

TC-1119 is Hardware Utilities #1, and contains the following programs:

1 Burglar Alarm
2 Memory Test #2
3 Bill Gronos Sound Show
4 Screen Test
5 Disk Drive Speed Check
6 TV Test Patterns
7 Dator Testbild
8 TV Adjustment
9 TV Color Test
10 VCR Counter Table
11 VCR Title Generator
12 Keyboard/Joystick Test
13 TV Alignment Screen
14 Serial Printer Test

TC-1120 is a disk of sound effects from various sources, collected by Jim Peterson.

1 CALL SOUND EFFECTS
2 CALL TONE
3 Church Bells
4 Drums
5 Experimental Sound
6 Funny Noises
7 Quickie
8 Shaping TI Sounds
9 Sound Developer
10 Sound Effects #1
11 Sound Effects #2
12 Sound Catalog #1
13 Sound Catalog #2
14 Sounds #1
15 Sounds #2
16 Sounds of Space
17 Music Synthesizer
18 Soundmaker
19 Sound Effects Collection
20 Sounds Library O

How you can contribute to your TI-Faire

We would welcome any new TI99/4A material which can be brought to the Faire for display.