(042) 8709

NEWS DIGEST

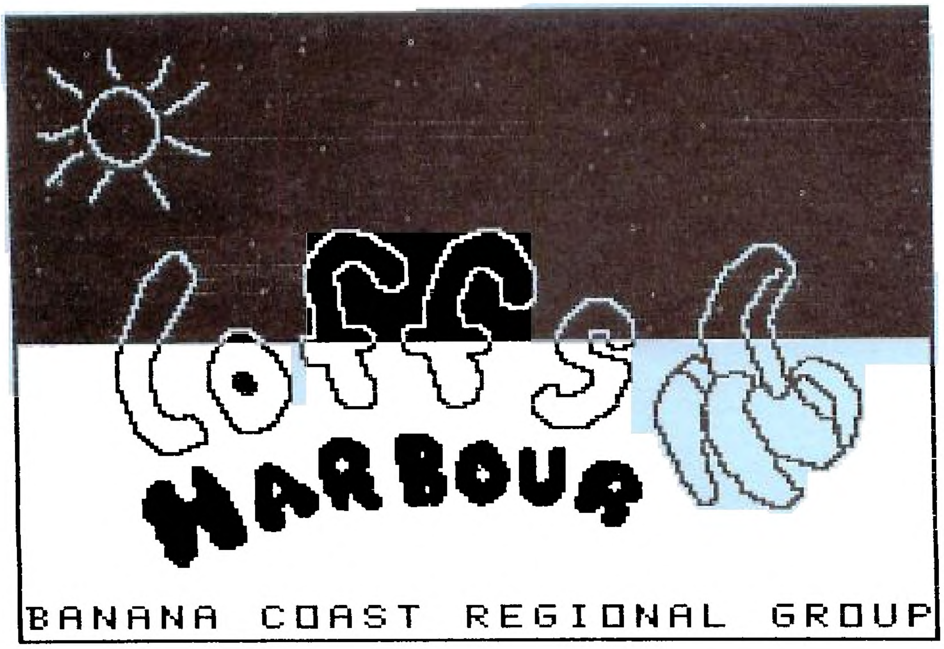
TISHUG

Focusing on the TI-99/4A Home Computer

Volume 6, Number 8 September 1987

Registered by Australia Post - Publication No. NBH5933

TI USERS OF THE NORTH



P.O. Box 214, Redfern, New South Wales, Australia, 2016 $2

TISHUG NEWS DIGEST

TI99/4A Owners Home Computer
User Group
TIsHUG NEWS DIGEST

SEPTEMBER 1987

Correspondence to:

PO Box 214
REDFERN NSW 2016

Texpac BBS: Tel.: (02)319.1009

MEMBERSHIP AND SUBSCRIPTIONS:

Joining Fee	$ 8.00
Annual Family Dues	$25.00
Dues O'seas Airmail	US$30.00
Publications Library	$ 5.00
Texpac BBS	$ 5.00
BBS Membership:	
Other TI User Group Members	$10.00
Public Access	$25.00

GROUP GENERAL MEETING:

First Saturday of each Month at Woodstock Community Centre, Church Street Burwood. Starts 2:00pm

COMMITTEE MEETINGS:

Before the main monthly meeting. Starting at 12:30 pm.

TIsHUG NEWS DIGEST ISSN 0819-1984

CONTENTS PAGE

This month's cover is the start of a theme of recognition of the regional groups. The support of a cover design and a demographic article giving details of the area and why it exists etc. by the leader would be appreciated.

Since taking on the editorship of the magazine, the TI-Writer program has been used extensively. This issue highlights a number of facts about that program. These are also applicable to Funelwriter.

How are the PICASSO competition entries coming on?

There has been a number of articles submitted in the past few months that have been hand written. While this is acceptable it takes time to have them typed. It may not get into the TND for some months. A better method is to type them using REM statements in program format and submit a disk or tape. TI-Writer files are even better. Articles printed using a dotmatrix printer frequently comes out poorly when reprinted. Refer to the April TND.

Bob Montgomery

CO-ORDINATOR'S REPORT

.... Chris Buttner

The Club is now on the way to incorporation. An application for reservation of name is with the NSW Corporate Affairs Commission, and once I receive confirmation that we can use our proposed name, the Memorandum and Articles of Association will be lodged. Our costs in this exercise will be minimal, so the whole exercise will achieve what we originally set out to do - protect the legal liability of each and every member. At present, it is possible everything will be finalised by the time you read this but if I were to be a little more realistic bearing in mind who we are dealing with, we should be incorporated some time in September.

There is a growing trend in the USA to which a number of TI users are falling victim so I am going to share a few thoughts with you. For some time now the only way to get some items for our computers has been to order by mail from overseas; principally the USA. If you have been watching the overseas magazines, you have no doubt seen there is now a growing number of small retail outlets attempting to satisfy the demand for user support. Some of them are doubtless quite reputable and I can think of a couple which I would have no hesitation dealing with. On the other hand there are those whose activities are questionable to say the least, if not criminal.

Some of these "retail" outlets do not operate from shop premises at all. Their phone number is an answering service and the only other way to contact them is via a post office box address . (If the truth be known, they probably hold down a regular 9-5 job and do the TI work as a side-line.)

I now know of some people who have had very unfortunate experiences with such operators. The story is very simple: the money was paid and either (a) the goods were not supplied at all; (b) items were short supplied with no refund given; or (c) different items of a lesser standard were supplied with no refund given.

The moral of all this seems to be to deal with the larger companies or if a small company, at least one which operates from shop premises and one which has given good service in the not too distant past (if you don't want to get together for a bulk order through the club shop).

It's the same old story: Buyer Beware! Not so much for the merchandise as the trading honesty of the proprietors. If you are tempted to go it alone and get "bitten" your most likely redress will be through the local law-enforcement agency (wouldn't it be dreadful if the local Sheriff was moonlighting as a TI dealer) and/or the Australian Foreign Affairs Department which appart from its major offices in the USA also has commercial officers located in important trade cities. They can make representations on your behalf but in the long run you will be depending on the integrity of the supplier - something which has already been called into question.

Elsewhere in this issue you will find tutorials which have been designed to help you develop programming skills. Within the club, we have two broad categories of members - programmers (of varying competence) and program users. If you fall into the user only category, I hope you will accept the challenge to become a programmer.

By knowing how to program, we are better able to understand what the other fellow is trying to do with the program we use. Secondly, if it doesn't do what we want, we have a better chance of making an informed decision on whether the program can be successfully modified (as opposed to a complete rewrite), and then carrying out that modification. Remember, you are not on your own: if there is something you don't understand, ask. That is why the club exists - to help you get the most from your computer.

On a different note I have received a letter from Garry Christensen, Secretary of the Brisbane TI Users Group. His letter deals with a TI-Faire to be held in conjunction with EXPO88. The full text of his message is reproduced in this issue. I hope you will take the time to read it. o

LETTER FROM G. CHRISTENSEN

Dear Chris,

I would like to take this opportunity to invite you and all the members of the Sydney Harbour Users Group to attend the TI-Faire that is to be held in Brisbane in May next year.

We have invited many major manufacturers and retailers who cater for the TI-99/4A computer and to this date the response has been favourable. A list of the business houses attending the faire will be forwarded to you at a later date. It is hoped that on display will be most of the latest hardware and software available for the TI computer and for the Myarc 9640 computer.

I would also like to invite the Sydney Harbour Users Group to set up a table and display any software that the members have written or any hardware upgrades that have been pioneered through your Group. Representation from every Users Group in Australia would help demonstrate to our American counterparts that the support for the TI computer is still strong in Australia.

There are many good programmes that have been or are being developed in Australia as well as some inovative hardware. The faire will provide an opportunity to share the knowledge with the other users in the country. Central meetings of the interstate users are too few and far between. At present the only official contact between Users Groups is through the exchange of newsletters. The continued success of the TI computer requires co-operation between the Users Groups and a free flow of information.

We are also inviting computer retailers throughout Australia to attend the Faire. A strong showing at the faire may induce some interest from computer retail businesses in the market for software and hardware associated with the TI-99/4A. I am tired of having to send to the United States to purchase programmes. There is a need for a distributor in Australia.

Aside from seeing the "latest", there are other reasons to come to Brisbane at that time. The faire is timed to coincide with the opening of the International Exposition several weeks before. While there may be other TI-Faires in Australia in the future, another EXPO will not be hosted in this country on our life-times. This is a once in a life-time chance and you will have the opportunity to attend the TI-Faire as well.

Continued on P30

* Secretary's Notebook *

It was great to see a good number of enthusiastic members at the August meeting increasing their programming knowledge in Basic and Extended Basic. Workshops, such as these, are great ways to increase programming skills and a good way of learning in a relaxing environment.

Welcome to the following members who have recently joined us:

R Doyle	- Ravenshoe Qld
A W Pruszinski	- Broken Hill
A L Szemere	- Glen Waverley Vic

There's a scattering from far and wide. Hope you all enjoy your membership of the group.

New magazines recently received, and passed to Warren for inclusion in the library include:

Melbourne Times	- April/May 1987
HV 99'ers News	- June 1987
Northern NJ 99'ers	- June 1987
Spirit of 99	- May 1987
Clubline	- February 1987

At the Committee meeting held on 1/8/87, the following program was mapped out for meetings for the remainder of this year:

5 SEPTEMBER - This meeting, at time of writing, should be held in the main Woodstock Building. If there is nay change to this it will be notified on the BBS and appropiate signs at Woodstock. The topic on this day will be software swapping. Members can bring along disks or tapes and download from the library. We hope to have three systems running so there shouldn't be any real long delays in you obtaining the programs you want. Also remember that if you have a piece of software that others may not have seen, bring it along for a demonstration.

3 OCTOBER - The theme this month will be an auction, which will run along similar lines to those held in the past. This will be a great opportunity to offload unwanted items and also perhaps, pick up something wanted at a bargain price. Full details of how the auction will operate will be in the next issue.

7 NOVEMBER - This month's theme will be a full day tutorial workshop. Topics have yet to be refined, but a full description of the day will also be included in the next 2 issues.

5 DECEMBER - The last meeting for 1987, and, traditionaly takes the form of a Christmas party with the club providing all the food and drink. Given a fine day we can look forward to a great BBQ in the spacious grounds of Woodstock. Further details will appear in later issues of the TND.

Well I hope you enjoy all of these planned activities and I look forward to meeting and chatting with as many as I can at the meetings. o

* * * * *

LETTERS

My sincere thanks to John Paine for the speedy and efficient job he did in repairing my console. It is good to know that such a reliable technical back up is available to club members who, like me, are babes in the wood when it comes to the technical aspect of our computer. Thanks again John, and I would be remiss if I failed to include Mel Copeland who acted as my intermediary.

Sincerely,

Noel Richards
VINCENTIA

The Editor,TISHUG News Digest

Dear Sir,
Please find enclosed a copy of a letter sent to me by Geoff Trott whilst in the USA. Geoff has been on sabbatical leave for the past 4 months and has spent some time in Canada and the USA. He should now be in England and we expect to see him back in Australia in about 3 months time.

Geoff has made some valuable contacts with TI-99/4A User Groups, and, at the same time, has sold some of his famous console testers. The message that Geoff is trying to convey in all of his letters is evident in the attachment. If possible would you please publish it for all TISHUG members.

Sincerely Yours,
Lou Amadio

18 July 1987

Dear Lou,
You should be getting two letters from people wanting console testers. The Philadelphia group want two and they should be sending you a cheque for them. Could you send at least one, if not both, depending on packaging, by air mail please. The other one should be from the Delaware Valley Group who should also be sending a cheque. This should come from Tony DiFebbo in New Jersey. It would seem to me to be a good idea to try to maintain contacts and perhaps extend them as the development of software happens quite quickly here it seems.

I was at the users meeting and Tony DiFebbo gave a talk on software for the Super Cart which appeared to do a lot of the things that the GRAM Kracker did. Things like saving cartridges to disk and loading them back again, and it looked much easier to use than the GRAM Kracker. It also allowed multiple menus on power up and general loading was much simplified. There is also a new version of the ROS for the Horizon Ram Disk card which I will be getting with the Geneve. Yes I will be bringing a Geneve back with me, at great expense.

It seems to me that the TISHUG group has a lot of money which could be used to establish communications with user groups around the world. People are working all over doing good work which everyone could benefit from. There are good programs everywhere, even in Australia. I hope I have stirred a few people to write to us about one little device. What we need is to maintain and expand these contacts and exchange as many newsletters as we can.

By the way, I suppose there are no console testers left after these three go out - at least I hope there are three left to go out. I may have to go into production when I get back!

Well we are really planning to leave Philadelphia in 12 days time and take our chances with the terrorists in Europe. Hopefully they will inhabit the places rich people go to and not those we will be in. There have also been rather too many near misses in the air for our peace of mind. It all makes for a good holiday. We have a lot of hot weather in July here so I quess we cannot expect it to follow us over the Atlantic.

We heard that Bob managed to increase his majority because of Joh's campaign but very little before the event and not much after. I guess Australia is too quiet compared with all this Iran-Contra mess and the state of Central America.

Regards
Geoff

As I have mentioned in an accompanying article, several members wrote requesting publication of a list of Funelwriter/TI Writer Editor and Formatter commands. I hope the TND Editor has room in this issue to publish the article. The article will also be available on disk from the club shop at the September meeting along with this other selection of software.

ON DISK:

1. SEPTEMBER:

The formentioned Editor and Formater commands together with a utility from Craig Sheehan which works similar to the CALL FILES command. Craig has also included a heavily documented source code, which should be of great benefit to budding assembly language programmers. Thank you for this Craig.

2. FUNPLUS:

My copy of this came via member Barry Ridgeway. FUNPLUS is an enhancement package for FUNELWRITER, and contains a heap of useful routines to add that litle bit extra to your printed document. There are loads of documentation files on the disk so make sure you print all these out before attempting some of the routines. This disk will come either in DOUBLE SIDED or DOUBLE DENSITY format, so make sure you ask for the correct version for your system. Those members with only SSSD drives will need to ask for the 2 disk package.

3. WADE'S PROGRAMS:

From member Wade Bowmer came a tape of interesting games and a great graphic designer program. Here's what you get on this disk:

i) Graphic Designer - the drawing and designing utility.

ii) Graphic Designer Information - a help file to get you started using Graphic Designer.

iii) Bouncing Bug - a game where you have to avoid the black bugs. Wade recommends that all programmers have a look at the code in this game as it contains some powerful programming tricks.

iv) Music Box Dancer - Wade has enhanced this good piece of music by adding his own bass line.

v) Balloon Voyage - has been released before but Wade has souped it up a bit and has added an interesting title.

vi) Heydey 7 - an interesting 7X7 checkers game where the computer plays for keeps.

All of these programs require Extended Basic but not memory expansion.

ON TAPE:

TAPE 1987/9 will contain the same as Wades programs on disk less the documentation file which will be available as a printed sheet. Make sure you ask for it when you get your tape. Also on the tape to take it up to 8 programs will be these:

OXYGENE - a very well written piece of music in Extended Basic.

CAVERNS OF MARS - an action arcade style game written in Basic.

LEMONADE STAND - where you try to make your fortune selling and buying to thirsty customers.

That will be the issues for this month, but next time it is again hoped to release some more 32K expansion games converted by George Meldrum to run out of Extended Basic.

Several members have taken advantage of the offer in last month's TND and have sent in for some of the earlier tape releases. By the time they read this I trust they will have received their orders.

John Scott of Penshurst writes - "With the Mini PE Box now creating new interest for members expanding into a disk system it follows that the demand for previous issued disks will start.

I have been looking over my back issues of TND trying to compile a list of disks I will buy but without any success because I do not know what the various utilities etc will do.

Is it possible to give a list, maybe in order of merit or a top 10 or 20 of the utilities disks and maybe later a list of entertainment/educational type disks"

Thanks for taking the time to write, John. I will make this a project and hopefully have the list published within the next 2 issues. In the meantime for any member expanding into a disk system there is no finer disk to start with than FUNELWRITER, probably the best piece of software yet released for this computer. After that I guess it depends on where your interests are.

That's it for this month. More software news next time.

* * * * * * * * *

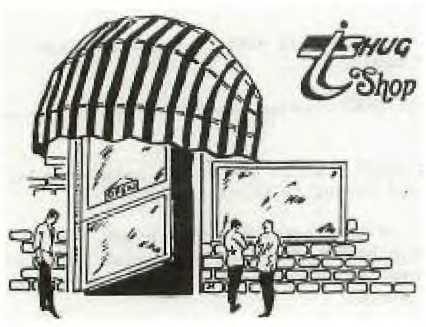
Return To Pirate's Isle.

Chapter 2

Well, how did you go last month? How many of you dusted that old module and tried to your wit's end and still could not get it out? Fear not, I could not think of any rums this time. So water va your thinking here are more clues.

Get paint, remove frame, drop frame, look paint, wrap paint, in oils, take paint, go up, drop blade, go sea, hold breath, swim down, swim west, feel silt, feel boat, swim opening, so boat, wear glass, drop brooch, drop ring, drop watch, drop paint, unwrap paint, drop oils, drop map, drop screw, drop oyster, drop mask.

Wish that clock would stop ringing. o



TIsHUG SHOP SEPTEMBER 1987

SHOP INVENTORY :-
- (a) HFi DS/DD 5 1/4" Disks......per box $19.00
- (b) Spike Protectors....................$29.00
- (c) Consoles Ver.2-2....................$65.00
- (d) T.I. Joystick handles..............$00.50
- (e) Peter Schubert's mini-expansion unit
  Mini-PE mother board (with one of either- 32K mem :: PIO :: RS232 port.$85.00
  Extra options on mother board
  RS232...............................$50.00
  PIO.................................$50.00
  32K memory exp. for Mini PE..........$45.00
  DS/DD disk controller card..........$190.00
  Finished painted box for Mini PE.....$30.00
- (f) Stand alone RS232.(old style)........$80.00

SECOND HAND ITEMS :-
- (a) Keyboards..........................$15.00
- (b) Grom Ports.........................$12.00
- (c) Ivory Console Cases................$ 2.00

BOOKS :-
- (a) Back issues of SND.................$ 1.00
- (b) Technical manual...................$15.00
- (c) TI-writer manual...................$15.00
- (d) Editor Assembler manual............$28.00
- (e) TI-LOGO curriculum guide...........$10.00
- (f) TI 3 ring binders..................$ 4.00
- (g) Micropendiums......................$ 3.00
  1986-June to Dec./1987-Jan.to July

SOFTWARE :-
- (a) Club Software Tapes................$ 3.00
- (b) Club Software Disks................$ 5.00
- (c) Picasso Publisher V1.1 (Arto Heino).$20.00

POSTAGE
Please NOTE that with all mail orders YOU have to pay postage and packaging.

If you are phoning the SHOP please note that I am NOT normally available before 7pm week days. (02)639 5847

Could all the people who picked up their 6264LP-15 memory chips at the last meeting (1/8/87) please check them as someone has a Z80 chip mixed up in their lot. Please bring it back for replacing.

RAM DISK CARD :-

We have sold out of RAM card auxiliary component kits, and sets of sockets, but still have a couple of P.C.Bs. and a few memory chips left.

The prices are:-
- RAM CARDS.................$ 35.00
- 6264L-15 RAM CHIPS (13)...$ 71.50 (SINGLE SIDE)
- " " " (24)...$132.00 (DOUBLE SIDE)
- 256K Ram expansion kit....$ 47.00
- Batteries (AAA)...........$ 14.00

PRINTER BUFFER :-

After a lot of hunting around for components for this project.

The prices are
- P-BUFF P.C.B. and EPROM..............$42.00
- P-BUFF components incl.PCB and EPROM..$75.00
- 41256 memory chips (8 reqd) are about.$ 6.00 each
- BPIO kit complete.....................$27.00
- Serial P.C.B. and components..........$63.00
- Serial P.C.B. only....................$18.00
- Computer sharer board and components..$20.00
- Printer sharer board and components...$20.00
- Plastic box small (DS 2508)...........$ 9.00
- Plastic box larger (DS 2505)..........$ 9.50
- 9 volt transformer ....$ 6.50

IBM type printer cables (25 pin D to 36 pin Centronic) are available from GEOFF WOOD ELECTRONICS PTY.LTD. LANE COVE WEST phone 4271676 for $15.00 each.

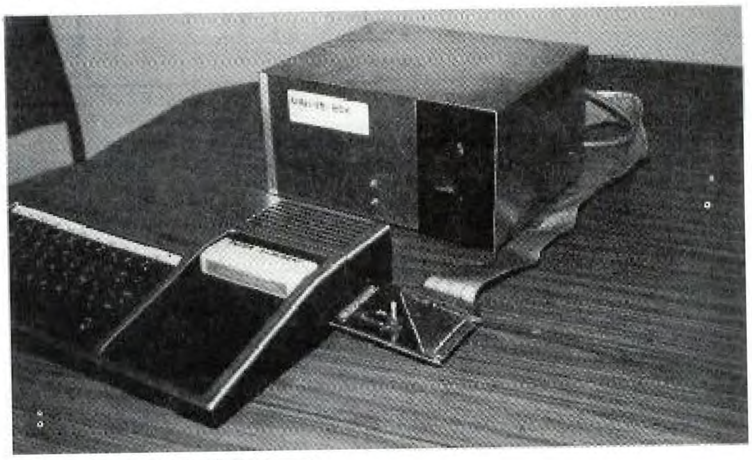
FOR SALE
selling spare system, surplus to my needs.

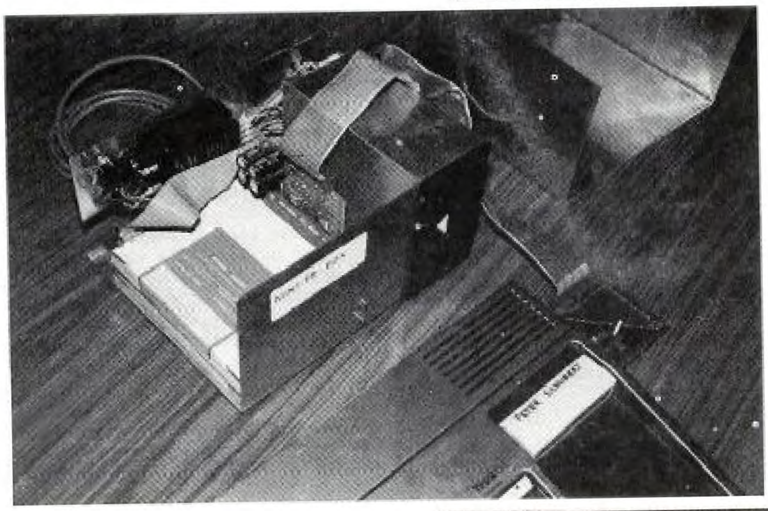
No! No! I am not quitting the TI 99, just selling surplus items!
Please ring Keith de Haan on 451 3032.

35 issues of MICROPENDIUM (all issues except Oct.86, Feb, Mar, Apr, May 87)	$ 45.-
6 issues of Softex plus 2 duplicates (Australian TI99 Mags.)	$ 8.-
24 issues of 99er-Home Computer Magazine (collector's items)	$ 60.-
1 SLICK- STICK Joy Stick	$ 10 -
1 Box of Memorex 5 1/4" SSDD diskettes (new)	$ 15.-
1 Cartridge Expander Navarone (new)	$ 30 -
1 TI Multiplan, with original books, diskettes and Module	$ 60.-
1 TI Writer, with original books, diskettes and Module	$ 40 -
1 Extended Basic Module, with orignal book and Quick Reference Guide	$ 35.-
2 TI Disk Drives full height in perfect condition, will demonstrate	$ 60 - each
1 TI RS 232 Card for Expansion System, in perfect condition, will demo.	$140.-
1 Modem Card 1200/75, plugs into Expansion System (Peter Schubert type)	$ 220 -
1 TI Disk Controller Card, with book but no Disk Manager Module or cables	$ 80.-
1 TI Disk Controller Card, with book, Disk Manager Module and cables	$100 -
1 TI Expansion Box, with Interface Card and Cable	$250.-
1 TI 99 Console Beige, with all books, modulator, power supply etc.	$100 -
1 Touch Typing Tutor Module, with book	$ 15.-

Hardware Products by Peter Schubert

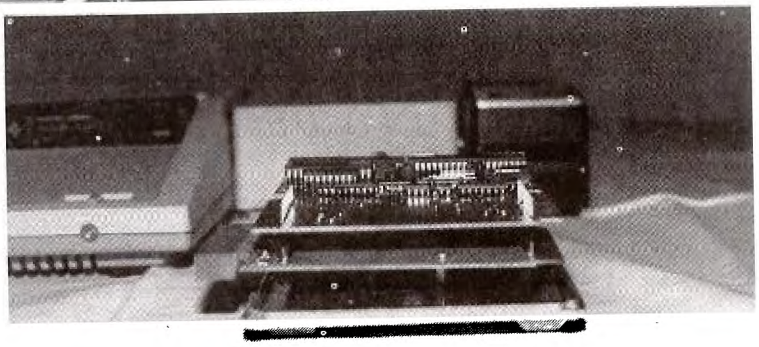
In the past nine months Peter Schubert has demonstrated two new products that he has developed. You may have seen them at a regional group meeting or at the monthly club meeting. Because not everyone is able to get to either meetings Peter intended to have photographs accompanying the articles he wrote decribing his products. Because of some editorial problems those photographs did not appear. For a full decription, refer to the Oct and Dec 86 tnd for the Mini-PE Box and to the Mar, June and July TND issues for the Mini Expansion System.

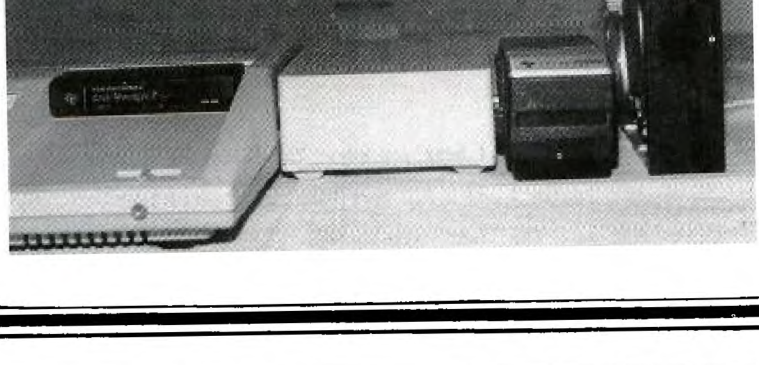




The accompanying two photos are reproduced for those members who do not have the Dec magazine. As you can see, fron the photos, the original TI cards were used.

Because of the difficulty in obtaining the original cards, Peter designed the Mini Expansion System. The photos show that the printed circuit boards are stacked on top of each other. Previously it was mentioned that the top board is the disk controller and will handle double sided double density drives and up to four drives.





The second board has a mini RS232 and 32K memory expansion.

The third board is for a PIO output and the fourth board is intnded for a 220K RamDisk. For prices refer to the shop article. o

TECHO TIME

.... John Paine

With the month of September almost upon us perhaps we should reflect on the achievements of our small but dedicated group. On the technical / hardware scene we now have members who, in the past would have hesitated at even looking at a soldering iron but now can say that they have successfully build a complicated peice of hardware called a RAMDISK.

Some of these people have now decided to attempt other projects like the PRINT BUFFER. The SHOP has been doing a roaring trade in semiconductors and other bits and peices for these projects. As you may have noticed I feel very proud of the members for attempting these projects and I daresay that that all members who have participated in these projects must have felt some pride in the fact that they have now got extra hardware which because of financial constraints or the sheer lack of confidence in their own ability, they would never have tried to attempt.

I believe that these projects have instilled a new lease of life into the group and in the future even more exciting projects will come to light. Terrific, you may say, but now spare a thought for those members that only have a bare console, maybe TE II and an early stand alone RS232 and modem, and no disk system. Although Peter Schubert's new MINI-EXPANSION system is now alive and running, and this product is available through the club shop, I wish to announce a new constructional project which could be of interest to those of us that would like to access VIDEOTEXT systems like VIATEL, WESTPAC and other similar databases.

This project is ideal for those of us who have only the basic equipment for communicating, and to some extent may be considered as reverting to old times.

This new project is the construction of a COMMAND MODULE that contains an EPROM to give VIDEOTEXT Terminal Emulation that does NOT require Extended Basic, Editor Assembler, Mini-Memory or 32K Memory Expansion but will require modem, and RS232 Card or Standalone.

This module kit will not come with Module Case but will include PCB, components, Instructions and a Pre-programmed EPROM. There will be two versions of the EPROM available. One will be for modem connection on RS232/1 and the other for RS232/2. No more loading cassette or disk software to utilise VIATEL, just plug in the module and dial away.

The software in ROM is based on the ENGLISH PRESTEL system and gives fast colour screens when used on VIATEL. No more black and white screens as provided by the software available up until now.

The only thing to remember with this module is that the VIATEL symbol # is defined on the TI keyboard as FUNCTION U (_ or underline symbol). Prototypes of the module are now being tested and the kits should be available mid to late September, for those that are interested. The PCB has been designed to accept up to 8 K bytes of EPROM and as such will the basis of other module projects in the future.

Projected cost will be about $20.00 and now is the time to indicate to the shop if there is any interest in becoming involved with this project. Please note that the kit will not have a module case. I would suggest that you could use the case of one of those old modules that do not get used anymore. You may remember that over a period I have asked members to donate their old 'dead' modules to the club so that a stock of cases would have been available for supply. Unfortunately, the response to this request was relatively poor, so I can only assume that all those old modules must still be working OK.

The software for the EPROM will be available in a format to load into the SUPER CARTRIDGE as well but will require EDITOR ASSEMBLER to load as it will be in program memory image format.o

Hardware News

By Peter Schubert 6th August 87

A "PIO" PORT

For MINI-PE system

The option of a true "PIO" printer port inside the Mini-PE Box is now available with the design of an all-new main PCB (or Motherboard) for the Mini-PE System. This new board is actually a Multi-function board consisting of the following functions;

32K MEMORY EXPANSION RS232 PORTS 1 AND 2 PIO PRINTER PORT

It also has the 44 way connection to console and thru-connect for other peripherals, and of coarse, the expansion buss for other Mini-PE boards to attach (such as the amazing little DSDD Disk Control card). The board is designed so that any single function can be provided, or a combination of functions, or all of them can be provided. A partial board can be expanded over a period of time as the TI budget requires, by returning the board (or Mini-PE) to me for fitting. The PIO is TI standard in operation and the connector used to attach the printer cable. In addition the "PIO" can also be opened as "PRINT", and because it has a different CRU address of >1400 it can possibly be used with existing TI hardware (or the PIO add on board described in June TND) as a second PIO port). The RS232 connector is similar to the TI RS232 Card for TI PEB and has both ports in same connector. However this board does not use all connections on port 2. Only transmit and receive data are connected. The 32K is the new single-chip version(plus some logic) which has proven itself on earlier board, and has now been adopted by the club for console 32k memory kits. A most important feature of this new board(as with earlier RS232 board) is its ability to run from console power supply (no external power required). I have also provided the option to power it from the regulated supply on the Disk control board when fitted with external DC Plugpak. This may be advisable in the future when the board is complete with all options and another board may be added to the Mini-PE system. A RAMDISK of up to 400K or 1600 sectors has been designed.

PRICES

Mini-PE motherboard $85 (with one of either 32K Memory :: PIO :: RS232 port)

Extra options on motherboard;

RS232 $50

PIO 50 more RAMDISKS I hear you say. The following will be provided either singly, or together as desired; 32K MEMORY RS232 ports 1 and 2 PIO port DSDD Disk Controller

The last also gives you the chance to upgrade to double density and so save heaps of disks. Watch out for this one soon.o

Fix for Excessive PEB Power Supply Voltages

Contributed by a member of the Sutherland Regional Group.

PROBLEM : Overheating of PE Box regulator.
Failure of card voltage regulators.

CAUSE : PE Box power supply voltage on unregulated rails is excessive.

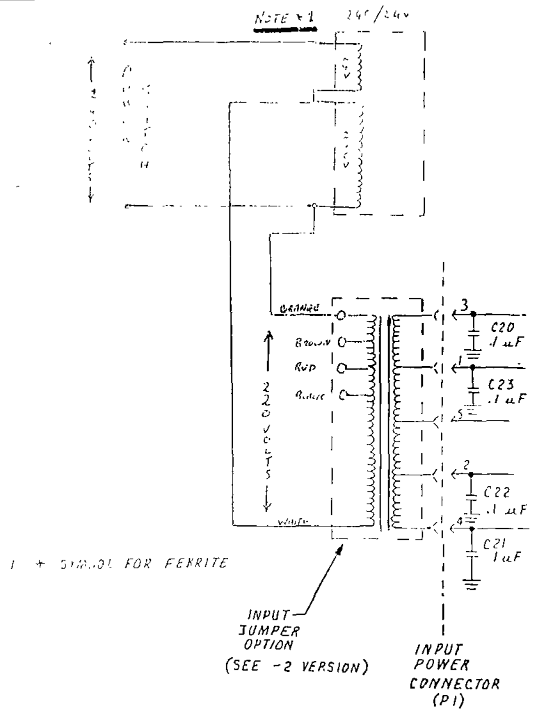
Rail Colour	Design Voltage	Actual Voltage	Modified Voltage	
Yellow	-16	-25	-18	
Blue	+5	+5	+5	Regulated
Green	+8	+14	+10	
Violet	+12	+12	+12	Regulated
Brown	+16	+25	+18	

CORRECTION : One successful way of overcoming the problem is to insert a small 240V/24V transformer between the mains supply and PE Box transformer. The transformer is connected in such a way that the primary and secondary windings are connected in series across the 240V mains supply.

The PE Box transformer primary winding is connected across the 240V winding of what is now effectively an "autotransformer" (see diagram). This has the effect of reducing the unregulated voltages close to the specified values (see above).

RESULT: The PE Box runs much cooler and reduces the likelyhood of failure of card regulators through over voltage and heating.

Note : The new transformer is small enough to fit inside the PE Box near the main PE Box transformer.



EDITOR'S NOTES:
Correct phasing of the windings on the new transformer is essential.

A multitap 240/240V isolation transformer may be used to provide a similar drop in unregulated voltage by connecting the PE Box to the 220V tapping on the isolation transfomer.
CAUTION!!
Due to the presence of mains voltages, check with your regional group technical co-ordinator before proceeding with this modification. o

REPRINTED...
FEB 1987, CLUBLINE MAGAZINE

```
1000 REM GAME LOADER NOV85
1010 REM 06:0:15>3>>;:18;000
1?820:17=:18;020032<8=830:19
805819810:0=71302=<7010?;060
1<801
1020 REM :0<002215>79<801:18
606:0:1420:00:17>001:04>0837
<02000:0><800835602000004<80
08354
1030 REM 020<1100<80<83=01=0
0<0>04008<083136><0?2<2729<:
0835516?9<1400206:1889<;616?
40605
1040 REM 16?<069910601>00<02
0:18006:0:16083>60006<1:0:13
81602<805:138<084<0450220000
60222
1050 REM ???:0285:0001608020
60200:006<800:1806086:046100
006:0:164;820:0090000<0<316;
69820
1060 REM :198:17<132506:0:15
>0800:000002006:0:1420380:00
<002006:0:15>0400:000002006:
0:142
1070 REM 0900:0000020:820:0?
4:0>6:820:0?4:0?08820:0>6:0=
21:><0201:17802020004=7?1060
216?=
1080 REM 0200001402018300020
2:164<<72064016?=02000=06020
1:0000202020020;00000460830
00420
1090 REM 0000<03;<07;0260400
006<0=7<006<0=7<0<0;;=8318<0
0060216?<045;<03;<07;<0;;06<
0=7<0
1100 REM 06<0=7<01000=<60880
0060216?<045;0>8301845830050
<0=0000002400000044534;302>
1110 CALL CLEAR :: CALL CHAR
(128,"0000FF0000FF"):: CALL
COLOR(13,7,8):: DISPLAY AT(2
,3):"GAME MODULES" :: CALL H
CHAR(3,1,128,32)
1120 DIM A$(14):: CALL HCHAR
(1,1,128,32):: FOR X=0 TO 13
 :: READ A$(X):: DISPLAY AT(
X+6,1):CHR$(X+65);" ";SEG$(A
$(X),12,26):: NEXT X
1130 DATA XATTACK1    Attack
,XBLASTO1    Blasto,XHUSTLE1
    Hustle,XAMAZING1   Amazi
ng,XCARWARS1   Carwars,XHANG
MAN1   Hangman,XOTHELLO1   O
thello
1131 DATA XYAHTZEE1   Yahtze
e,XFOOTBALL1  Football,XZERO
-ZAP1  Zero Zap,XADVENTURE1
Adventure,XVCHESS1    Video
Chess
1132 DATA XVGAMES1    Video
Games,XBJ/POKER1  Blackjack
& Poker
1140 CALL SOUND(250,440,2)::
 CALL INIT
1150 CALL LOAD(12288,2,0,160
,0,2,1,0,10,192,160,131,50,6
,2,6,66,6,66,208,226,0,1,6,1
95,208,210)
1160 CALL LOAD(12314,5,195,2
,4,0,40,209,115,2,69,15,0,10
,69,209,179,2,70,15,0)
1170 CALL LOAD(12334,241,133
,220,6,6,4,22,245,6,1,22,234
,4,224,131,196,4,96,160,0)
1180 CALL KEY(0,K,X):: IF X<
1 THEN 1180 ELSE IF K<65 OR
K>78 THEN CALL SOUND(250,220
,2):: GOTO 1180 :: ELSE K=K-
65
1190 DISPLAY AT(2,3):SEG$(A$
(K),12,26):: FOR X=1 TO 12 :
: CALL LOAD(12999+X,ASC(SEG$
(A$(K),X,1))):: NEXT X
1200 CALL LOAD(-31804,48)
1210 GOTO 1210
```

Starting a Database from Scratch.

Pt 1.

by Chris Buttner of TIsHUG, July 1987.

There are many programs which allow us to organise a mailing list, create a data base and so on. Some are very powerful, some commercial, some shareware, but have you ever tried designing one yourself? It's not such a hard thing to do provided you know clearly what you expect of the system and set about designing it in a systematic and logical manner.

What I plan to do is start from scratch and I hope you will join me on this learning experience. It will be spread over a number of issues of the magazine so don't be put off thinking it will all be too much and beyond your comprehension.

In the commercial world of that other (IBM) machine, there are software packages galore which extol their virtues in sorts, data file size and so on and compete with one another in the race to have more and more "go gear" installed and menu selectable by the operator. I am often prompted to ask the question "why". They are terrific routines but will I ever use them. As this project develops, don't for one moment think it is meant to be the definitive answer to anything and everything. By the time we get to the end however, you should be able to design a program to suit YOUR particular needs, and what's more, have it operate efficiently.

Our starting point is really the think tank stage. There is no paper - just thoughts, a glimmer of hope and finally the light at the end of the tunnel, which in this case is the desire to have "something" which will allow me to retrieve names quickly from a file. You may decide on something else but always bear in mind how you expect to use the program to retrieve your data. I'm not interested in sorting the list of names; just finding them and their associated data; regularly and quickly. It doesn't matter to me whether the correct record is numbered 5 or 955 - the important thing is the name.

On paper, this conceptual idea looks something like this:-

```
          DATA BASE (FILE)
               |
           SEARCH
               |
       DISPLAY RESULT
```

This is the real meat in the package. Whatever I do from this point onward must be directed at achieving that end.

Stage 2 of the idea starts to take shape like so:-

```
          DATA BASE (FILE)
               |
     ------------------
     |     |     |    |
    ADD UPDATE DELETE SEARCH
                      |
              DISPLAY RESULT
```

I now have a framework within which I can design the various modules of the program necessary to get it to all hang together. These various sections in turn can be broken down into smaller tasks. The smaller the task the easier it to to write the program and the more feasible the project becomes. Notice that at this stage I haven't attempted to write a single line of program. The extra time spent now will save me countless hours of heartbreak and frustration later.

In the programs which follow some conventions have been observed:-

(1) all are written in XBasic;
(2) NO attempt has been made to fully utilise the multiple statement capability of XBasic;
(3) many of the program lines have been kept short in the interests of clarity;
(4) comments (REMarks) are liberally made to assist your understanding;
(5) the style is not necessarily the most elegant or efficient;
(6) your are free to modify/doctor any or all of the program/s as you see fit. In fact you are encouraged to do so if it will (a) aid your understanding or (b) make the program work better for YOU!
(7) the programs and sub-programs will be available from the club in MERGE format so you can easily incorporate them into this and any subsequent programs you are inspired to write.

As a general rule there are three aspects to any database. They are (1) Structure, (2) Records, and finally (3) Fields. The field is the smallest working part of the database. Various fields make up each a Record. Structure is the way the fields and records are organised in the database file.

STRUCTURE

Since the main program will use another file to store all the data it is necessary to define some of the parameters. Firstly, the data file will be a DIS/FIX type to allow manipulation of the data with simple basic programs.

The next step is deciding on the method of retrieving the information. The common methods of doing this are (1) checking every record until a match is found (long and tedious); (2) implementing a shell type search if the database is in sorted order; (3) using pointers to indicate the desired record and (4) hash coding. For this exercise, I plan to follow hash coding techniques.

If you are new to hash coding, it is a procedure which structures (builds) a list (records). The address in the list is derived from a mathematical equation. In layman's terms, this means I will have to decide which part of each individual record will be the "key". (If you have seen the Navarone Database Management System you should be familiar with the expression "Key Field"). Once I have done this, everything else becomes automated because the program will decide the record number (a job to which it is well suited and something which I don't want to worry about). For my application, the record number is immaterial; what counts is that I can quickly find the record if it exists.

For best results with hash coded lists, the actual number of records should be somewhere between 50% and 60% of the total list capacity. If I anticipate 60 records, my file should be capable of holding about 120 records. This may seem to be something of a waste but in fact it is a compromise between record retrieval speed and file size. If the file is 70% full the search time for my record will increase by about 50% and at 90% capacity if will have increased 400%.

Additionally, the list will work better if the maximum number of records is a prime number which yields a remainder of 3 when divided by 4. A small program will calculate this number for us.

RECORD and FIELDS

The final consideration relating to the file is the number of individual parts or fields and their respective size. For this example we will use the following fields and sizes:-

SURNAME	16
FIRSTNAME	12
STREET	21
TOWN	18
POSTCODE	4
TELEPHONE	7
AREA CODE	3

giving a total record size of 81(bytes) for 7 fields.

The first program creates the database file. Some of the things you need to keep in mind are:-

(a) try to keep your field names as short as possible: for example use SNAME for Surname;

(b) record 0 will store the total number of records in the file and also the parameters for each record;

(c) 3 bytes are reserved at the start of record 0 to record the total number of records;

(d) each record is limited to a maximum of 21 lines;

(e) if your field titles exceed (record size -3) bytes, you will be forced to start defining your records again;

(f) take care your "design work" all fits within the parameters of the TI Disk System. A single sided, single density drive will leave you roughly 350 sectors. Each sector contains 256 bytes. Divide 256 by the number of bytes in each record. This will give you the number of records per sector. Multiply this by 175 (because the number of actual records is roughly half the maximum) and you will have some indication of the number of records you will get on a SS/SD disk.

(g) you are allowed a maximum of only 6 characters to name your file. The program will add on the suffix "_BDF" so you will always be able to tell your database files from other files on the same disk.

(h) the moral of all this is PLAN before you start committing anything to paper (or in this case the program).

One of these fields has to be the key and I have chosen the surname. What happens if there are two people with the same surname? The algorithm we develop for coding the field will take care of this and ensure we finish with two distinct records rather than overwriting one with the other.

One further aspect must be settled before we put pen to paper and create the database program and that is how we will interface with it. The options are almost limitless and range from menu selections through to primitive prompts. I propose having a command line where commands (or more precisely nmemonics) will be entered along with a parameter. It should now be obvious this is a very specific program: not one designed to do a multitude of tasks for various users but one which will fulfil the specific need I have defined. (With modifications, you will customise it to suit your specific applications.)

My working screen will look something like this:-

```
+------------------------------------------+
| (Command Area)                           |
| ---------------------------------------- |
|                                          |
| (Data Display Area)                      |
|                                          |
|                                          |
|                                          |
+------------------------------------------+
```

Commands are entered above the line and normally this area is reserved for operator input.

The Data Display area is where information is displayed by the program and where the entry/updating of data takes place. Think of the data area as a blackboard if you want to simplify the explanation.

Almost immediately you should now start to envisage some of the smaller subtasks which must be designed. The most obvious one is is "wipe" to clear the blackboard. If you think of the tasks in real terms you fill find it easy to define and then write the subprograms to carry out those tasks. Another task which can be included is a help menu to increase user friendliness.

Getting down to the "nitty-gritty", the line dividing commands from data will be at row 3. Commands will appear on row 1. With three rows remaining "untouched" that leaves 21 rows to be wiped (24-3). This can easily be done by successively wiping 3 sections each of 7 rows. One way to "wipe" is to write the space character successively. Here is an example without program line numbers:

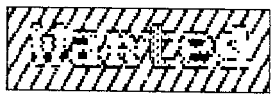
```
A$=" " (A$ takes the meaning of the space.)
A$=RPT$(A$,196) (A$ is redefined to now mean
                 196 space characters.)
```

By writing the newly defined A$ three times starting at locations 7 rows appart I can wipe the entire data area. This is easilly done in a for next loop. Because I will use this task repeatedly, it is ideally suited to being written as a subprogram. When I need it, I call the subprogram - simple!

By resequencing the actual program to start at a high line number, I can save it as a MERGE file (after testing to make sure it does what I want) and at the end, merge it back in to my base program.

This process of breaking the program down into smaller parts is repeated. The partial listing which I have for you is definitely not elegant and I deliberately have not attempted to make full use of Extended Basic's multiple statement lines. You should however start to get a feel for the way the program is developing and posibly how you can modify it to suit your needs.

A program to create a data file is in the downloadable software area of the BBS with the name CREATEDBF . O



One Speech Synthesizor. Will pay around $65 for it. Please include any manuals that came with it. Phone 543-5180 After 4:30pm, ask for "Wade Bowmer".

RTTY

ON THE 99/4A HOME COMPUTER

By G.Wilson (VK2YGW) and D.Crawford

The aim of this project was to enable the reception of Radio Teletype (RTTY) transmissions on the TI 99-4A home computer together with the minimum of external hardware.

Requirements for the system are a standard 4A console and mini memory module along with the appropriate software. Externally we need a radio receiver with good sensitivity and high signal to noise ratio, along with a demodulator capable of decoding F.S.K. (frequency shift keyed) analog signals and converting them to digital signals which are then inverted and applied to pin 4 off the joystick port.

HISTORY. Teletype is an early form of coding based on 5 data bits and was invented in 1874 by a Frenchman, Emile Baudot. This code was in use until 1925 when a New Zealand farmer, Donald Murray rationalised it on the basis of the most frequently used characters of the alphabet having the least number of data bits. This standard is used today by all commercial and amateur stations. The first public telegraph service started in England in 1927 and was known as the G.P.O. Telex Service. A similar service is opperated localy by Telecom Australia.

After W.W.2 teletype comunications expanded rapidly with manufacturers such as Teletype Corp., I.T.T., Klienschmidt Co., Creed Ltd. and Siemens producing there mechanical teleprinters, many of which are still in use today. The Amateur Radio fraternaty seized upon this surplus equipment and established their own international standards.

THEORY. The Baudot or Murray Code is the 5 bit international teleprinter code and at first appears to have a maximum of 32 combinations for a character set. As there are 26 letters in the alphabet ,plus numerals and punctuation, space is a problem. Cunningly the code is shifted by two control characters called LETTERS and FIGURES,achieving a possible 60 combinations. The actual number used is 58, as the NUL.character,or all bits zero,is not used.

The Baud is the shortest single unit in a code, and can be expresed as the reciprical of the time of that unit, eg.if the shortest unit is 20 MS.,the speed of the signal will be 1 over .02 secs. or 50 Baud. Amateur Radio speeds are 45.5 or 50 Baud with the former being common in Australia and Europe. Commercial RTTY transmitions,wire services etc.often use 75 Baud along with encripted code to secure there privacy. The speed used in this project is 45.5 Baud or approx. 22 MS. and the value for this speed is used in the full and half bit delay routines documented in the software.Other speeds may can be obtained by altering the value loaded into REGISTER 3 and can be accurately monitored at pin 7 of the joystick port with an oscilloscope. The data displayed on the screen is also stored in V.D.P.Ram from >1000 upwards and this may be serially dumped to a printer at a later date via pin 7 of the joystick port. The printer subroutine is written for 300 Baud operation and will need some individual hardware modifications to achieve RS.232 compatability with the printer that is used.The source code for this routine will be released when fully tested.

INSTRUCTIONS. Select the Debug option from M/MEM. menu using the M.....Mem Modify comand and enter the source code from listing starting at address 71FC and ending at 75FE. After the program has been checked for errors,you can then store it on cassette tape using the S...Save command. This will make reloading the program a lot easier if it is lost due to battery failure in the M/MEM module.The output of the RTTY. demodulator is applied to pin 4 of the joystick port through a simple transistor inverter as this point is a normal -INT.3 keyboard line and is active low. There is no chassis ground available at this port and it will have to be obtained elsewhere eg.casette port pin 3.

To run the program use the E....Execute command from 71FC and then via a switch turn on the data input to pin 4. NOTE: If the data from the demodulator is applied to pin 4 before the RTTY.program is running the data will appear as keyboard inputs and the computer will print garbage on the screen. A simple single pole switch at pin 4 that is enabled after the program is running will stop this problem. To escape from the program at any time all you have to do is press the FCTN. key and disable the data switch and the program will return you to DEBUG.

Testing the operation of the complete receive system requires a known error free signal. Murphys Law says there will not be any RTTY. stations on the air just when you need one. A standard audio cassete recorder can be used to record signals from your receiver headphone socket and this can then be replayed many times, thus making monitoring and setup less tedious. For those of you that have a FM.receiver Eg.a scanner, you will find RTTY transmissions on certain Amateur Repeaters located in the major cities. These Repeaters operate in the 2METRE band and in Sydney the call sign is VK2RTY and the receive frequency is 146.675 MHZ. The demodulater that I used was a simple PLL. type called the ETI.733 and pc. boards are still available from RCS RADIO in SYDNEY. Alternately any RTTY. demodulator may be pressed into service and this is left to the experimenter's choice.

I have used a cassette recorder to store the WIA. weekly news broardcast on sunday mornings and then reviewed it when time permitted. This system is quite flexable and is cheap memory.

In conclusion I hope this little project will lead to more experimenting using the 99/4A.,as other fields and applications for it's use are limited only by imagination.

```
*****************************************************
BAUDOT TO ASCII CONVERSION FOR RTTY DECODER ON 99/4A
        USING MINI-MEMORY AND JOYSTICK PORT.
*****************************************************

 DATA COMES IN ON PIN 4 OF THE JOYSTICK PORT,IS
CONVERTED TO ASCII IN A LOOK-UP TABLE AND IS DISPLAYED
ON THE SCREEN.AS WELL IT IS STORED IN VDP RAM FROM
>1000 FOR LATER DUMPING TO A PRINTER.

REGISTERS USED:

         R0  CURRENT VDP ADDRESS
         R1  DISPLAYED BYTE IN ASCII
         R2  HIGHEST VDP ADDRESS TO DISPLAY
         R3  TIMER COUNT VALUE
         R4  SCRATCH LOCATION
         R5  BYTE BEING READ AS BAUDOT
         R6  SHIFT COUNT
         R7  LETTER OR FIGURE SHIFT
         R8  HIGHER VDP ADDRESS (NOT VISIBLE)
         R10 TEMPORARY PROGRAM COUNTER STORAGE
         R11 PC LINK
         R12 CRU BASE ADDRESS
         R13 PC
         R14 WORKSPACE POINTER
         R15 STATUS

LOCN   DATA   LABEL   MNEMONIC
```

```
*******************************************************
  INITIAL SETUP OF SOME REGISTERS
*******************************************************
71FC  0208          LI R8,>1000    STARTING HIGH VDP
71FE  1000
7200  020C          LI R12,>0006   CRU ADDRESS
7202  0006
7204  C28B          MOV R11,R10    SAVE RETURN LINK
7206  0200          LI R0,>0002    FIRST SCREEN ADDR
7208  0002
720A  0202          LI R2,>0240    MAX SCREEN ADDR
720C  0200
720E  0207          LI R7,0000     "LETTERS" DEFAULT
7210  0000
*******************************************************
  DO START BIT AND WAIT FOR CHARACTER OR CHECK FOR
                    "FCTN"KEY
*******************************************************
7212  0206   START  LI R6,0005     SHIFT COUNT
7214  0005
7216  0205          LI R5,0000     CLEAR CHARACTER
7218  0000
721A  1F00   PIN4   TB 0           TEST PIN 4
721C  1604          JNE DEL1       JUMP IF LOW
721E  1F04          TB 4           TEST "FCTN" KEY
7220  13FC          JEQ PIN4       JUMP IF HIGH
7222  C2CA          MOV R10,R11    RESTORE LINK
7224  045B          B *R11         RETURN TO EASYBUG
*******************************************************
  WAIT ONE AND A HALF-BIT TIMES AND SAMPLE FIVE TIMES
*******************************************************
7226  06A0   DEL1   BL @HDELAY     DO HALF DELAY
7228  7300
722A  06A0   SAMPLE BL @DELAY      DO DELAY
722C  7380
722E  1F00          TB 0           SAMPLE PIN 4
7230  1602          JNE +2         JUMP IF LOW
7232  0225          AI R5,>8000    SET MSB TO 1
7234  8000
7236  0606          DEC R6         DEC SHIFT
7238  1302          JEQ +2         JUMP IF R6=0
723A  0915          SRL R5         SHIFT R5 RIGHT 1
723C  10F6          JMP SAMPLE
723E  09B5          SRL R5,>0B     RIGHT JUSTIFY
*******************************************************
  CONVERT BAUDOT TO ASCII,DISPLAY BYTE,ADJUST R0
*******************************************************
7240  C105          MOV R5,R4      SAVE R5
7242  06A0          BL @BASCII     CONVERT TO ASCII
7244  7400
7246  0420          BLWP @VSBW     DISPLAY R1
7248  6024
724A  06A0          BL @RZERO      TREAT R0
724C  7480
724E  06A0          BL @HDELAY
7250  7380
7252  C100          MOV R0,R4      SAVE R0
7254  C008          MOV R8,R0      HIGH VDP RAM
7256  0420          BLWP @VSBW     PUT INTO VDP
7258  6024
725A  C200          MOV R0,R8      SAVE HI VDP ADDR
725C  C004          MOV R4,R0      RESTORE SCREEN,
725E  0588          INC R8         ADDR
7260  0288          CI R8,>3F00    MAX VDP ADDRESS
7262  3F00
7264  130A          JEQ QUIT       EXIT
*******************************************************
  PRINT SPACES TO PREVIOUS PRINT
*******************************************************
7266  C100          MOV R0,R4      SAVE R0
7268  0220          AI R0,>0020
726A  0020
726C  0201          LI R1,>002E    ASCII FULL STOP
726E  002E
7270  0420          BLWP @VSBW     DISP FULL STOP
7272  6024                         ERASE
7274  C004          MOV R4,R0      RESTORE R0
7276  1000          JMP 0          CONTINUE
7278  10CC          JMP START
727A  045B   QUIT   B *R11         RETURN TO E/BUG
///////////////////////////////////////////////////////
  SUBROUTINES
///////////////////////////////////////////////////////
```

```
*******************************************************
  DELAY FOR HALF OF A BIT TIME
*******************************************************
7300  020C   HDELAY  LI R12,>0024   CRU FOR OUTPUT
7302  0024
7304  1D00           SBO 0          SET P2 HIGH
7306  1D01           SBO 1          SET P3 HIGH
7308  1D02           SBO 2          SET P4 HIGH
730A  1000           JMP 0          SHORT DELAY
730C  1000           JMP 0
730E  1E02           SBZ 2          RESET P4
7310  1E01           SBZ 1          RESET P3
7312  1E00           SBZ 0          RESET P2
7314  020C           LI R12,6       CRU FOR PIN 4
7316  0006
7318  0203           LI R3,>0490    DELAY FOR 45.5
731A  0490                         BAUD
731C  0603           DEC R3
731E  16FE           JNE -1         11 MS DELAY
7320  045B           B *R11
*******************************************************
  DELAY FOR WHOLE BIT TIME THEN PULSE OUT ON PIN 7
        AT TIME OF SAMPLING
*******************************************************
7380  0203   DELAY   LI R3,>0920    45 BAUD VALUE
7382  0920
7384  0603           DEC R3         22 MS DELAY
7386  16FE           JNE -1
7388  020C           LI R12,>0024   CRU FOR OUTPUT
738A  0024
738C  1D01           SBO 1          SET P3
738E  1D02           SBO 2          SET P4
7390  1000           JMP 0          SHORT DELAY
7392  1000           JMP 0
7394  1E01           SBZ 1          RESET P3
7396  1E02           SBZ 2          RESET P4
7398  020C           LI R12,6       CRU FOR PIN 4
739A  0006
739C  045B           B *R11
*******************************************************
  CONVERT BAUDOT VALUE IN R5 TO ASCII AND PUT IN R1
*******************************************************
7400  0285   BASCII  CI R5,>001B    FIGURE SHIFT?
7402  001B
7404  1604           JNE LETTER
7406  0207           LI R7,0002     OFFSET VAL OF 2
7408  0002
740A  1000           JMP 0
740C  1006           JMP LOOK
740E  0285   LETTER  CI R5,>001F    LETTERS SHIFT
7410  001F
7412  1603           JNE LOOK
7414  0207           LI R7,0000     NO OFFSET IF LTRS
7416  0000
7418  1000           JMP 0
741A  0A25   LOOK    SLA R5,2       SHIFT R5 LEFT TWO
741C  A147           A R7,R5        ADD OFFSET
741E  C065           MOV (R5),R1    VALUE INDEXED R5
7420  7580                          BASE ADDRS TABLE
7422  045B           B *R11
*******************************************************
  INCREASE R0,AND PUT IN A LEFT MARGIN
*******************************************************
7480  C144   RZERO   MOV R4,R5
7482  0285           CI R5,>001B    LETTERS SHIFT?
7484  001B
7486  1304           JEQ +4
7488  0285           CI R5,>001F    FIGURE SHIFT?
748A  001F
748C  1301           JEQ +1
748E  0580           INC R0         INCREASE CURSOR
7490  C100           MOV R0,R4
7492  0244           ANDI R4,>001F  MASK
7494  001F
7496  0284           CI R4,>001F    R0 AT RIGHT MASK
7498  001F
749A  1602           JNE +2         JUMP IF NOT
749C  0220           AI +3          MAKE LEFT MARGIN
749E  0003
74A0  0280           CI R0,>0282    MAX SCREEN ADDRS
74A2  0282
74A4  1602           JNE +2
74A6  0200           LI R0,2        BACK,TOP OF SCREEN
74A8  0002
74AA  045B           B *R11
```

Continued on P13

BACK UP AND FLIP!

by Jim Peterson
(Tigercub)

Or rather, flip and back up. There is a good deal of controversy over whether disks should be flipped. It is claimed that the liner inside the disk jacket collects dust, and that running a disk backward will release this dust and foul the disk.

I can only say that I have a file of about 250 flipped disks, including utility disks which I run many times every day, and have never had a problem which I could blame on the flipping. And, I am using low-cost no-brand disks.

However, disks do occasionally go bad, whether flipped or not, whether cheap or expensive, whether "certified" or not, and the "certified" won't get you your data back. So it certainly pays to have a backup copy. And, if you put the backup on the flip side of another disk, it has cost you nothing at all. If there is 1 chance in 50 that a disk will go bad, there is 1 chance in 2500 that both will go bad!

Almost all of the bargain disks I have flipped have had 358 good sectors on the back side. Occasionally, one of them will have a bad sector, but this is no problem unless it is sector 0 or 1. However, I would not use one of those sector-copy programs to copy onto a disk which had bad sectors.

Don't waste your money on one of those disk-flipping kits. They consist of nothing but a template and a paper punch. You can make your own template from the cover of a disk that has gone bad, or from a piece of cardboard. The single-hole hand paper punch is sold in office supply stores for less than $1.50. Be sure to get the kind that has a plastic catcher on the lower jaw to catch the punchings. You don't need a square write-protect notch, a half-round one works just as well.

Several people have told me that their flipped disks wouldn't work, and it turned out that they had asked an Apple peddler how to flip them. To flip a disk for the TI, you must punch another reading hole on each side, as well as another write-protect notch. This is where the template comes in handy, although the placement of the holes is not all that critical. I use typist's "white-out" correction fluid to mark the spots to be punched, but I'm not sure I will recommend it - don't let dried chalk from the bottle top get into your disk!

In simple terms, if you hold a disk so that a notch is on the upper left edge and a small hole is to the left of the center hole - you must punch a notch in the same position on the upper right edge, and another small round hole in the same position on the right side of the center hole. Then you must turn the disk over and punch a corresponding small hole to the left of the center hole.

The notch on the edge is easy, although some brands of disk have a cover with a very thick hard edge. Now for those center holes. You DO NOT punch them through the disk itself, just through the cover. Cut a piece of index card about an inch wide. Slip it into the center hole and under the place you want to punch. Now slip the lower jaw of the punch into the center hole, with the card protecting the disk surface, and punch your hole. Do the same on the other side.

The only trouble you might have is that the inner lining of the disk cover may not punch out cleanly, probably because your punch is not uniformly sharp. The instruction sheet with one of the disk punching kits recommends that you "nibble" at the hole with the punch. DON'T! You might get thin shavings of the disk cover inside the cover. It is better to leave the card in place, lift up the bit of liner with the tip of a knife and slice it off. A bit of fuzz around the edges is nothing to worry about.

It is probably a good idea to flip your disks before you use them, even before you initialize them, but I have yet to damage a disk with this procedure.

I should mention that I am running single-sided single-density disks in a TI drive. I have no idea what trouble flipping might cause in your souped-up systems. Can someone tell us? o

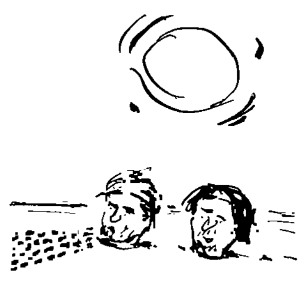
* * * * * * *

Continued from P12

```
*******************************************************
            LOOK UP TABLE
*******************************************************
```

7580	2000	SPACE
7582	2000	SPACE
7584	4500	E
7586	3300	3
7588	0A00	L/F
758A	0A00	L/F
758C	4100	A
758E	2D00	-
7590	2000	SPACE
7592	2000	SPACE
7494	5300	S
7596	2100	!
7598	4900	I
759A	3800	8
759C	5500	U
759E	3700	7
75A0	0D00	C/RET.
75A2	0D00	C/RET.
75A4	4400	D
75A6	2A00	*
75A8	5200	R
75AA	3400	4
75AC	4A00	J
75AE	0700	BELL
75B0	4E00	N
75B2	2C00	,
75B4	4600	F
75B6	2400	$
75B8	4300	C
75BA	3A00	:
75BC	4B00	K
75BE	2800	(
75C0	5400	T
75C2	3500	5
75C4	5A00	Z
75C6	2200	"
75C8	4C00	L
75CA	2900	)
75CC	5700	W
75CE	3200	2
75D0	4800	H
75D2	2300	#
75D4	5900	Y
75D6	3600	6
75D8	5000	P
75DA	3000	0
75DC	5100	Q
75DE	3100	1
75E0	4F00	O
75E2	3900	9
75E4	4200	B
75E6	3F00	?
75E8	4700	G
75EA	2600	&
75EC	2000	SPACE
75EE	2000	SPACE
75F0	4D00	M
75F2	2E00	.
75F4	5800	X
75F6	2F00	/
75F8	5600	V
75FA	3B00	;
75FC	2000	SPACE
75FE	2000	SPACE o

```
100 REM *** Start of Main Pr
ogram - CREATEDBF ***
110 REM ***Start of Title an
d Instruction screen***
120 DISPLAY AT(11,3)ERASE AL
L:"My DataBase File Creator"
130 CALL PAUSE(600)
140 DISPLAY AT(20,1):"Answer
 the Questions on the follow
ing screens to create the da
tabase file."
150 CALL PAUSE(1500)
160 REM ***End of Title/Inst
ruction screen***
170 REM ***Subprogram Recno
will determine a suitable si
ze for the DataBase File***
180 CALL RECNO(F)! The numbe
r of records in the file wil
l be returned in the variabl
e F
190 A$=STR$(F):: A$=RPT$("0"
,3-LEN(A$))&A$ ! Make A$ 3 b
ytes long with leading zeros
200 REM ***Subprogram Param
will collate file to enable
creation of DB file
210 CALL PARAM(F$,L)
220 REM ***The record parame
ters will now be combined wi
th the max. # of records in
a single string variable***
230 F$=A$&F$ ! Place the num
ber of data records at the s
tart of the Data string
240 REM ***The DataBase file
 will now be created***
250 CALL CREATE(F,L,F$)
260 REM *** The main program
 will auto-run if you remove
 the REM statement at line 2
70 and name the main file. *
**
270 REM RUN"DSK1.FILENAME"
280 END
290 REM *** End of Main Prog
ram ***
300 REM *** Subprogram PAUSE
 to create a Screen delay.
Changes is "A" cause a chang
e in the delay time. ***
310 SUB PAUSE(A)
320 FOR P=1 TO A :: NEXT P
330 SUBEND
340 REM *** End of PAUSE Sub
program ***
350 REM *** Start of RECNO S
ubprogram to calculate the n
umber of records to create i
n the file ***
360 SUB RECNO(A)
370 DISPLAY AT(10,1)ERASE AL
L:"How many records do you w
antto have in the file?"
380 ACCEPT AT(11,22)SIZE(3):
A
390 REM ***Reset variable A
to 1 less than twice its val
ue***
400 A=A*2-1
410 FOR D=2 TO 7
420 IF INT(A/D)=A/D THEN 450
 ! If evenly divisible, exit
 and increment
430 NEXT D
440 IF (A-3)/4=INT(A/4)THEN
460 ! Test for prime number=
4K+3 and if true, exit
450 A=A+1 :: GOTO 410 ! Incr
ement and recalculate
460 DISPLAY AT(16,3):"File S
ize =";A;"Records" :: CALL P
AUSE(600)! Display result on
 the screen
470 SUBEND
480 REM *** End of RECNO Sub
program ***
490 REM *** Start of PARAM S
ubprogrm to define Field Nam
es and Size. ***
500 SUB PARAM(P$,RL)! Subpro
gram for Parameters
510 DISPLAY AT(3,1)ERASE ALL
:"  FIELD NAME        LENGT
H" !Heading Titles
520 DISPLAY AT(4,1):"END=End
 Input" :: CALL PAUSE(500)!
Prompt explains how to end i
nput <21
530 P$="" :: RL=0 :: A=4 ::
B=0 ! Initialise P$ (Paramet
ers) and RL (Record Length)
and counters (A,B)
540 ACCEPT AT(A,1)SIZE(15):A
$ ! Field name input. Max si
ze is 15 characters
550 IF (A$="END")+(B>=24)THE
N 640 ! If END or beyond lin
e 24, exit
560 ACCEPT AT(A,25)SIZE(2)VA
LIDATE(DIGIT):L$ :: IF L$=""
 THEN 560 ! Input for field
size - check digit
570 IF LEN(L$)=1 THEN L$="0"
&L$ ! Right justify field si
ze
580 DISPLAY AT(A,25):L$
590 RL=RL+VAL(L$)! Increment
 Record Length
600 B$=STR$(LEN(A$&L$))! Cal
culate length of 2 fields an
d convert result
610 IF LEN(B$)=1 THEN B$="0"
&B$ ! Pad out B$ to 2 charac
ters
620 P$=P$&B$&A$&L$ ! Convert
 smaller strings to one larg
e string
630 A=A+1 :: B=B+INT((LEN(A$
)+2+VAL(L$))/28)+1 :: GOTO 5
40 ! Calculate next display
row number and return
640 A=LEN(P$)! Calculate len
gth of P$
650 W=0 :: CALL CHECK(W,A,RL
)
660 IF W=0 THEN 680
670 GOTO 530
680 P$=P$&RPT$("~",RL-A-3)!
Pad out the record with the
~ character allowing 3 bytes
 for the # of records
690 DISPLAY AT(8,5)ERASE ALL
:"Your file will have";INT(2
56/RL);"        records/secto
r."
700 DISPLAY AT(14,2):"A Sing
leSided SingleDensity Disk c
an hold";INT(256*175/RL);"re
cords."
710 CALL PAUSE(1500)
720 SUBEND
730 REM *** End of PARAM Sub
program ***
740 REM *** Start of CREATE
Subprogram to make the DataB
ase File ***
750 SUB CREATE(FL,RL,P$)! Su
bprogram to create Database
File
760 DISPLAY AT(8,2)ERASE ALL
:"Insert Data Disk in Drive"
 :: CALL PAUSE(800)
770 DISPLAY AT(12,1):"What D
rive Number? (1/2)" :: ACCEP
T AT(12,26)VALIDATE("12")SIZ
E(1):A
780 A$="DSK"&STR$(A)&"."
790 DISPLAY AT(10,4)ERASE AL
L:"Now name the new file." :
: DISPLAY AT(14,6):A$ :: CAL
L KEY(3,K,S)
800 REM *** The CALL KEY com
mand ensures the filename is
 in Upper Case ***
810 DISPLAY AT(16,3):"(Maxim
um of 6 Characters)" ! File
naming instruction
820 ACCEPT AT(14,11)SIZE(6):
F$ :: IF F$="" THEN 820
830 CALL KEY(5,K,S):: A$=A$&
F$&"_DBF" ! Return to Upper/
Lower Case and put _DBF on t
he end of the filename.
840 OPEN #1:A$,RELATIVE(FL),
DISPLAY ,UPDATE,FIXED(RL)
850 PRINT #1,REC 0:P$
860 FOR A=1 TO FL
870 PRINT #1,REC A:RPT$("~",
RL)! Fill blank records with
 the ~ chracter.
880 NEXT A
890 CLOSE #1
900 SUBEND
910 REM *** End of CREATE Su
bprogram ***
920 REM *** Start of CHECK S
ubprogram to verify record s
ize parameters ***
930 SUB CHECK(W,A,RL)
940 IF A+3>RL THEN W=A+3-RL
950 IF RL>=255 THEN W=MAX(W,
(RL-255))
960 IF W<>0 THEN CALL WARNIN
G(W)
970 SUBEND
980 REM *** End of CHECK Sub
program ***
990 REM *** Start of WARNING
 Subprogram to display warni
ng prompts at battom of scre
en ***
1000 SUB WARNING(W)
1010 DISPLAY AT(22,8):"**WAR
NING**" :: DISPLAY AT(23,1):
"Your record is too long by"
 :: DISPLAY AT(24,1):"a tota
l of ";W;" characters"
1020 CALL PAUSE(600):: DISPL
AY AT(22,8):"*TRY AGAIN*" ::
 DISPLAY AT(23,1):RPT$(" ",5
6)
1030 CALL PAUSE(600):: DISPL
AY AT(22,1)
1040 SUBEND
1050 REM *** End of WARNING
Subprogram ***
```


'Hey, you know this would make a great micro game!'

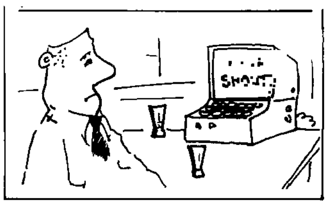
```
100 REM ** PT 109 **
110 REM *SEPTEMBER 1983*
120 CALL CLEAR
130 DISPLAY AT(6,9):"*******
*****"
140 DISPLAY AT(7,9):"*
     *"
150 DISPLAY AT(8,9):"*  PT 1
09  *"
160 DISPLAY AT(9,9):"*
     *"
170 DISPLAY AT(10,9):"******
******"
180 DISPLAY AT(12,14):"by"
190 DISPLAY AT(14,7):"George
 Yarmoluk"
200 DISPLAY AT(23,5):"Instru
ctions?  (Y/N)"
210 CALL KEY(0,K,S)
220 IF S=0 THEN 210
230 IF K=78 THEN 350
240 IF K=89 THEN 260
250 GOTO 210
260 CALL CLEAR :: PRINT " U-
Boat is hiding in a 10x10gri
d. It can move one squareat
a time Horizontally, Ver-"
270 PRINT "tically, or Diago
nally.": :
280 PRINT "You command PT-10
9 and have 20 depth charges
to sink thesubmarine.": :
290 PRINT "If a shot is:":"
 HOT - the sub Must move"
300 PRINT "  WARM - the sub
May move":"  COLD - the sub
Cannot move": :
310 PRINT "Use:":" ARROW KEY
S to get on Target":" SPACE
BAR to fire Charges": : :
320 PRINT "       press any
key"
330 CALL KEY(0,K,S)
340 IF S=0 THEN 330
350 CALL CLEAR
360 CALL SCREEN(11)
370 DISPLAY AT(12,4):"ONE MO
MENT PLEASE . . ."
380 V=16 :: H=24
390 CALL CHAR(128,"000101030
3FF7F00000000000000000000000
080C0FFFF0000")
400 CALL CHAR(132,"000001031
F0F0000000000000000000000000
0C0F8F8000000")
410 CALL CHAR(136,"000001030
3010000000000000000000000008
0C0C080000000")
420 CALL CHAR(140,"8001094C0
4010700000000000000000000122
0224C90E00000")
430 CALL CHAR(43,"009E9090FE
1212F2")
440 DATA FFFFFFFFFF8F8F8,1F
1F1FFFFFFFFFFF,F8F8F8F8F8F8F
8F8,FFFFFFFFFF1F1F1F
450 DATA F8F8F8FFFFFFFFFF,1F
1F1F1F1F1F1F1F,FFFFFFFFFF,FF
FFFFFFFF181818
460 DATA 1F1F1FFFFF1F1F1F,00
0000FFFFFFFFFF,181818FFFFFF
FFF,F8F8F8FFFFF8F8F8
470 DATA 1818181818181818,18
1818FFFF181818,000000FFFF,3C
7EE7C3C3E77E3C,00000008,00
480 FOR I=97 TO 114
490 READ A$
500 CALL CHAR(I,A$)
510 NEXT I
520 CALL COLOR(9,6,16)
530 CALL COLOR(10,6,16)
540 CALL COLOR(11,2,16)
550 RANDOMIZE
560 CALL MAGNIFY(3)
570 CALL CLEAR
580 CALL DELSPRITE(ALL)
590 GOSUB 820
600 GOSUB 1080
610 GOSUB 1210
620 IF X=98 THEN 680
630 GOSUB 1600
640 IF M=0 THEN 680
650 GOSUB 2570
660 IF R<1 THEN 2990
670 GOTO 610
680 DISPLAY AT(9,23):"S.O.S.
"
690 DISPLAY AT(11,23):"HILFE
!"
700 DISPLAY AT(1,1):"+ VE VI
LL RETURNEN! +"
710 DISPLAY AT(17,23):"SCORE
:"
720 DISPLAY AT(19,24):200-(R
3*10+R2*5+R1*2)
730 RESTORE 780
740 FOR TUNE=1 TO 11
750 READ TIME,NOTE
760 CALL SOUND(TIME*200,NOTE
,0,NOTE*2.01,5)
770 NEXT TUNE
780 DATA 4,523,2,659,2,784,3
,880,1,659,4,880
790 DATA 4,1047,2,1175,2,784
,4,1047,1,20000
800 GOTO 3240
810 REM *PRINT GRID*
820 CALL CLEAR
830 PRINT "aghghghghghghghgh
ghgd"
840 PRINT "crmrmrmrmrmrmrmrm
rmrf"
850 PRINT "lononononononon
onoi"
860 PRINT "crmrmrmrmrmrmrmrm
rmrf"
870 PRINT "lononononononon
onoi"
880 PRINT "crmrmrmrmrmrmrmrm
rmrf"
890 PRINT "lononononononon
onoi"
900 PRINT "crmrmrmrmrmrmrmrm
rmrf"
910 PRINT "lononononononon
onoi"
920 PRINT "crmrmrmrmrmrmrmrm
rmrf"
930 PRINT "lononononononon
onoi"
940 PRINT "crmrmrmrmrmrmrmrm
rmrf"
950 PRINT "lononononononon
onoi"
960 PRINT "crmrmrmrmrmrmrmrm
rmrf"
970 PRINT "lononononononon
onoi"
980 PRINT "crmrmrmrmrmrmrmrm
rmrf"
990 PRINT "lononononononon
onoi"
1000 PRINT "crmrmrmrmrmrmrmr
mrmrf"
1010 PRINT "lonononononono
nonoi"
1020 PRINT "crmrmrmrmrmrmrmr
mrmrf"
1030 PRINT "ejkjkjkjkjkjkjkj
kjkjb": :
1040 CALL SPRITE(#2,132,4,V+
1,H-3)
1050 DISPLAY AT(5,26):"20"
1060 RETURN
1070 REM *HIDE TARGET*
1080 L=INT(RND*100)+1
1090 IF L<11 THEN 1080
1100 IF L>88 THEN 1080
1110 T=INT(L/10)
1120 IF T*10-L=0 THEN 1080
1130 S1=0
1140 R=20
1150 R1=0
1160 R2=0
1170 R3=0
1180 X=0
1190 Y=0
1200 RETURN
1210 REM *PLAYER INPUT*
1220 CALL KEY(0,K,ST)
1230 DISPLAY AT(2,24):"SHOTS
"
1240 DISPLAY AT(3,24):"LEFT:
"
1250 PH=INT(L/10):: PV=L-PH*
10
1260 IF K=32 THEN 1490
1270 IF K=69 THEN 1320
1280 IF K=88 THEN 1360
1290 IF K=83 THEN 1400
1300 IF K=68 THEN 1440
1310 GOTO 1220
1320 REM NORTH
1330 V=V-16
1340 IF V<16 THEN V=16
1350 GOTO 1470
1360 REM SOUTH
1370 V=V+16
1380 IF V>160 THEN V=160
1390 GOTO 1470
1400 REM EAST
1410 H=H-16
1420 IF H<24 THEN H=24
1430 GOTO 1470
1440 REM WEST
1450 H=H+16
1460 IF H>168 THEN H=168
1470 CALL SPRITE(#2,132,4,V+
1,H-3)
1480 GOTO 1220
1490 REM FIRE
1500 CALL POSITION(#2,VER,HO
R)
1510 X=INT((VER-16)/16)
1520 Y=INT((HOR-20)/16)
1530 CALL HCHAR(X*2+3,Y*2+4,
113)
1540 R=R-1
1550 DISPLAY AT(5,25):R
1560 S=Y+X*10
1570 RETURN
1580 REM
1590 REM *PRINT CLUES*
1600 IF S<>L THEN 1690
1610 DISPLAY AT(23,6):"YOU G
OT ME!"
1620 CALL SPRITE(#3,140,9,V,
H)
1630 FOR I=1 TO 30
1640 CALL SOUND(-50,110,I,13
3,I,137,I,-7,I)
1650 NEXT I
1660 CALL DELSPRITE(#3)
1670 M=0
1680 RETURN
1690 M=1
1700 D=L-S
1710 A=ABS(D)
1720 FOR I=1 TO 2
1730 IF A=M*1 THEN 1870
1740 IF A=M*9 THEN 1870
1750 IF A=M*10 THEN 1870
1760 IF A=M*11 THEN 1870
1770 M=2
1780 NEXT I
1790 M=3
```

```
1800 DISPLAY AT(23,7):" "
1810 DISPLAY AT(23,7):"* COL
D *"
1820 FOR I=0 TO 30 STEP 5
1830 CALL SOUND(-30,110,I,-7
,I)
1840 NEXT I
1850 R3=R3+1
1860 RETURN
1870 IF M=2 THEN 1970
1880 DISPLAY AT(23,7):" "
1890 DISPLAY AT(23,7):"* HOT
 * "
1900 FOR I=0 TO 30 STEP 5
1910 CALL SOUND(-30,110,I,-7
,I)
1920 NEXT I
1930 CALL SOUND(150,2222,0,2
233,0)
1940 R1=R1+1
1950 RETURN
1960 DISPLAY AT(23,7):" "
1970 DISPLAY AT(23,7):"* WAR
M *"
1980 FOR I=0 TO 30 STEP 5
1990 CALL SOUND(-30,110,I,-7
,I)
2000 NEXT I
2010 CALL SOUND(100,1111,0,1
116,0)
2020 R2=R2+1
2030 RETURN
2040 REM *NORTH/SOUTH*
2050 IF P=1 THEN 2190
2060 IF D<0 THEN 2130
2070 T=L-1
2080 GOSUB 2770
2090 IF P=9 THEN 2120
2100 IF T=999 THEN 2150
2110 L=T
2120 RETURN
2130 T=L+1
2140 GOTO 2080
2150 D=D1
2160 GOTO 2050
2170 REM *EAST/WEST*
2180 IF P=1 THEN 2060
2190 IF D<0 THEN 2260
2200 T=L-10
2210 GOSUB 2770
2220 IF P=9 THEN 2250
2230 IF T=999 THEN 2280
2240 L=T
2250 RETURN
2260 T=L+10
2270 GOTO 2210
2280 D=D1
2290 GOTO 2180
2300 REM *SOUTHWEST/NORTHEAS
T*
2310 IF P=1 THEN 2450
2320 IF D<0 THEN 2390
2330 T=L-9
2340 GOSUB 2770
2350 IF P=9 THEN 2380
2360 IF T=999 THEN 2410
2370 L=T
2380 RETURN
2390 T=L+9
2400 GOTO 2340
2410 D=D1
2420 GOTO 2310
2430 REM *NORTHWEST/SOUTHEAS
T*
2440 IF P=1 THEN 2320
2450 IF D<0 THEN 2520
2460 T=L-11
2470 GOSUB 2770
2480 IF P=9 THEN 2510
2490 IF T=999 THEN 2540
2500 L=T
2510 RETURN
2520 T=L+11
2530 GOTO 2470
2540 D=D1
2550 GOTO 2440
2560 REM *PATTERN TESTS*
2570 P=1
2580 IF S1<>S THEN 2600
2590 S=S+1
2600 D=S1-S
2610 D1=S-S1
2620 S1=S
2630 IF M=3 THEN 2750
2640 A=ABS(D)
2650 FOR I=1 TO 2
2660 IF A=P*1 THEN 2050
2670 IF A=P*10 THEN 2180
2680 IF A=P*9 THEN 2310
2690 IF A=P*11 THEN 2440
2700 P=2
2710 NEXT I
2720 P=3
2730 IF M=2 THEN 2750
2740 GOTO 2190
2750 RETURN
2760 REM *LEGAL MOVE?*
2770 IF T<0 THEN 2940
2780 IF T>99 THEN 2940
2790 REM *IF LOW-T IS 0, LOW
-L CAN'T BE 9*
2800 REM *IF LOW-T IS 0, LOW
-L CAN'T BE 0*
2810 T9=0
2820 L9=0
2830 IF T=0 THEN 2850
2840 T9=INT(T/10)
2850 IF L=0 THEN 2870
2860 L9=INT(L/10)
2870 IF T-T9*10=0 THEN 2900
2880 IF T-T9*10=9 THEN 2920
2890 RETURN
2900 IF L-L9*10=9 THEN 2940
2910 RETURN
2920 IF L-L9*10=0 THEN 2940
2930 RETURN
2940 IF M=2 THEN 2970
2950 T=999
2960 RETURN
2970 P=9
2980 RETURN
2990 DISPLAY AT(1,3):"+ TORP
EDO LOSS! +"
3000 DISPLAY AT(17,23):"SCOR
E:"
3010 DISPLAY AT(19,24):200-(
R3*10+R2*5+R1*2)
3020 CALL SPRITE(#1,128,2,PH
*16+17,PV*16+21)
3030 CALL SPRITE(#3,136,9,PH
*16+17,PV*16+21)
3040 CALL POSITION(#2,PTH,PT
V)
3050 CALL POSITION(#3,TH,TV)
3060 CALL COINC(#2,#3,4,HIT)
3070 IF HIT<0 THEN 3110
3080 CALL MOTION(#3,(PTH-TH)
/4,(PTV-TV)/4)
3090 CALL SOUND(-30,2000,0)
3100 GOTO 3050
3110 CALL SPRITE(#2,140,9,PT
H,PTV)
3120 CALL DELSPRITE(#3)
3130 FOR I=1 TO 30
3140 CALL SOUND(-40,110,I,13
3,I,137,I,-7,I)
3150 NEXT I
3160 CALL DELSPRITE(#2)
3170 RESTORE 3220
3180 FOR TUNE=1 TO 18
3190 READ TIME,NOTE
3200 CALL SOUND(TIME*100,NOT
E,0,NOTE*2.01,5)
3210 NEXT TUNE
3220 DATA 6,175,2,196,4,220,
4,196,4,233,4,220,2,196,2,16
5,2,175,2,22000
3230 DATA 4,294,4,262,4,233,
4,220,4,196,2,220,2,175,4,26
2
3240 DISPLAY AT(23,2):"ANOTH
ER GAME? (Y/N)"
3250 CALL KEY(0,K,S)
3260 IF S=0 THEN 3250
3270 IF K=89 THEN 580
3280 IF K=78 THEN 3300
3290 GOTO 3250
3300 CALL CLEAR
3310 DISPLAY AT(12,6):"AUF W
IEDERSEHEN!"
3320 CALL DELSPRITE(ALL)
3330 END
```

* * * * * * *

```
100 GOSUB 260
110 DISPLAY AT(6,1):"Text of
 first line" :: DISPLAY AT(8
,1):A$ :: ACCEPT AT(8,1)SIZE
(-20):A$
120 DISPLAY AT(12,1):"Text o
f second line" :: DISPLAY AT
(14,1):B$ :: ACCEPT AT(14,1)
SIZE(-20):B$
130 DISPLAY AT(18,1):"Text o
f third line" :: DISPLAY AT(
20,1):C$ :: ACCEPT AT(20,1)S
IZE(-28):C$
140 GOSUB 260
150 DISPLAY AT(6,1):A$ :: DI
SPLAY AT(8,1):B$ :: DISPLAY
AT(10,1):C$
160 DISPLAY AT(14,1):"How ma
ny labels? " :: ACCEPT AT(14
,18):LABELS
170 OPEN #1:"PIO"
180 FOR I=1 TO LABELS
190 PRINT #1:CHR$(14);:: PRI
NT #1:A$ :: PRINT #1
200 PRINT #1:CHR$(14);:: PRI
NT #1:B$ :: PRINT #1
210 PRINT #1:C$ :: PRINT #1
:: PRINT #1 :: PRINT #1
220 NEXT I
230 PRINT #1:CHR$(27)&"@";::
 CLOSE #1
240 DISPLAY AT(24,1):"ANOTHE
R RUN? Y":Y$ :: ACCEPT AT(24
,14)SIZE(-1):Y$ :: IF Y$="Y"
 THEN 100 ELSE IF Y$<>"N" TH
EN 240
250 END
260 CALL CLEAR :: DISPLAY AT
(1,5):"*****************" ::
 DISPLAY AT(2,5):"* LABEL PR
INTER *" :: DISPLAY AT(3,5):
"*****************" :: RETUR
N
```

```
110 REM SHEEPDOG
120 REM BY S. BONNETT
130 REM
140 CALL CLEAR
150 PRINT "DO YOU WANT INSTR
UCTIONS"
160 INPUT "(Y/N) ":A$
170 IF SEG$(A$,1,1)="N" THEN
 420
180 CALL CLEAR
190 REM PRINT INSTRUCTIONS
200 PRINT "MOVE THE DOG USIN
G"
210 PRINT
220 PRINT "THE ARROW KEYS (E
SXD)"
230 PRINT
240 PRINT "SPACE MAKES THE D
OG BARK"
250 PRINT
260 PRINT "HERD THE SHEEP IN
TO THE"
270 PRINT
280 PRINT "PEN BY MOVING ROU
ND BEHIND"
290 PRINT
300 PRINT "THEM AND BARKING"
310 PRINT
320 PRINT "GET ALL THE SHEEP
 INTO THE"
330 PRINT
340 PRINT "PEN BEFORE YOU RU
N OUT OF"
350 PRINT
360 PRINT "TIME"
370 PRINT
380 PRINT
390 PRINT "PRESS ANY KEY TO
CONTINUE"
400 CALL KEY(0,K,S)
410 IF S=0 THEN 400
420 RANDOMIZE
430 CALL CLEAR
440 CALL COLOR(10,1,1)
450 CALL COLOR(13,1,1)
460 DIM XSHEEP(10),YSHEEP(10
)
470 TIME=450
480 NUMSHEEP=INT(RND*8)+1
490 REM PLACE SHEEP
500 FOR SH=1 TO NUMSHEEP
510 XSHEEP(SH)=INT(RND*15)+1
5
520 YSHEEP(SH)=INT(RND*22)+2
530 IF (XSHEEP(SH)<10)*(YSHE
EP(SH)<10)THEN 510
540 CALL GCHAR(YSHEEP(SH),XS
HEEP(SH),CH)
550 IF CH<>32 THEN 510
560 REM 8 IS PROBABILTY OF W
HITE
570 IF RND>8 THEN 610
580 CALL HCHAR(YSHEEP(SH),XS
HEEP(SH),128)
590 GOTO 620
600 REM BLACK SHEEP
610 CALL HCHAR(YSHEEP(SH),XS
HEEP(SH),110)
620 NEXT SH
630 CALL CHAR(104,"0000FF")
640 REM INITIALISE EDGE CHAR
ACTERS
650 CALL CHAR(105,"080808080
80808")
660 CALL CHAR(106,"00000F080
80808")
670 CALL CHAR(107,"0000F8080
80808")
680 CALL CHAR(108,"08080F")
690 CALL CHAR(109,"0808F8")
700 REM DRAW BOUNDRY
710 CALL HCHAR(1,1,104,32)
720 CALL HCHAR(24,1,104,32)
730 CALL VCHAR(1,1,105,24)
740 CALL VCHAR(1,32,105,24)
750 REM DRAW PEN
760 CALL HCHAR(6,6,104,6)
770 CALL VCHAR(6,6,105,5)
780 CALL VCHAR(6,12,105,5)
790 CALL HCHAR(10,6,104,2)
800 CALL HCHAR(10,11,104,2)
810 REM DRAW CORNERS
820 CALL HCHAR(1,1,106)
830 CALL HCHAR(6,6,106)
840 CALL HCHAR(1,32,107)
850 CALL HCHAR(6,12,107)
860 CALL [illegible](10,6,108)
870 CALL [illegible](24,1,108)
880 CALL HCHAR(24,32,109)
890 CALL HCHAR(10,12,109)
900 REM PLACE DOG
910 YD=5
920 XD=5
930 CALL SCREEN(12)
940 REM DEFINE SHEEP
950 A$="003C7FDFFF3F1212"
960 CALL CHAR(128,A$)
970 CALL CHAR(110,A$)
980 CALL COLOR(13,16,1)
990 CALL COLOR(10,2,1)
1000 REM DEFINE DOGS
1010 DOG$="1030F1323E1E1236"
1020 CALL CHAR(98,DOG$)
1030 CALL COLOR(9,9,1)
1040 CALL VCHAR(YD,XD,98)
1050 TAIL$="1034F2323E1E1236
"
1060 CALL CHAR(99,TAIL$)
1070 REM START OF MAIN LOOP
1080 CALL KEY(0,K,S)
1090 GOSUB 1400
1100 TIME=TIME-1
1110 IF TIME<1 THEN 1910
1120 IF S=0 THEN 1170
1130 NEWY=YD
1140 NEWX=XD
1150 ON POS("EXDS ",CHR$(K),
1)+1 GOTO 1170,1200,1220,124
0,1260,1290
1160 REM NO INPUT-WAG TAIL
1170 CALL GCHAR(YD,XD,CH)
1180 CALL HCHAR(YD,XD,98+99-
CH)
1190 GOTO 1080
1200 NEWY=YD-1
1210 GOTO 1330
1220 NEWY=YD+1
1230 GOTO 1330
1240 NEWX=XD+1
1250 GOTO 1330
1260 NEWX=XD-1
1270 GOTO 1330
1280 REM BARK
1290 CALL SOUND(65,131,2,-6,
2)
1300 GOSUB 1590
1310 GOTO 1080
1320 REM RE-DRAW DOG
1330 CALL GCHAR(NEWY,NEWX,CH
)
1340 IF CH<>32 THEN 1080
1350 CALL HCHAR(NEWY,NEWX,98
)
1360 CALL HCHAR(YD,XD,32)
1370 XD=NEWX
1380 YD=NEWY
1390 GOTO 1080
1400 REM MOVE SHEEP
1410 REM 3 IS PROBABILY OF N
O SHEEP MOVING
1420 IF RND>3 THEN 1580
1430 SH=INT(RND*NUMSHEEP)+1
1440 X=XSHEEP(SH)
1450 Y=YSHEEP(SH)
1460 REM 5 IS FLOCKING FACTO
R
1470 FLOCK=-(RND>.5)
1480 X=X+(1-FLOCK)*(2*(RND>.
5)+1)+SGN(XSHEEP(1)-XSHEEP(S
H))*FLOCK
1490 Y=Y+(1-FLOCK)*(2*(RND>.
5)+1)+SGN(YSHEEP(1)-YSHEEP(S
H))*FLOCK
1500 CALL GCHAR(Y.X.CH)
1510 IF CH<>32 T[illegible] 1580
1520 CALL GCHAR([illegible]EP(SH),X
SHEEP(SH),CH)
1530 CALL HCHAR(Y,X,CH)
1540 CALL HCHAR(YSHEEP(SH),X
SHEEP(SH),32)
1550 XSHEEP(SH)=X
1560 YSHEEP(SH)=Y
1570 GOSUB 1760
1580 RETURN
1590 REM MOVE SHEEP AWAY FRO
M BARK
1600 FOR SH=1 TO NUMSHEEP
1610 X=XSHEEP(SH)
1620 Y=YSHEEP(SH)
1630 IF ((X-XD)^2+(Y-YD)^2)>
17 THEN 1740
1640 X=X+SGN(X-XD)
1650 Y=Y+SGN(Y-YD)
1660 CALL GCHAR(Y,X,CH)
1670 IF CH<>32 THEN 1740
1680 CALL GCHAR(YSHEEP(SH),X
SHEEP(SH),CH)
1690 CALL HCHAR(Y,X,CH)
1700 CALL HCHAR(YSHEEP(SH),X
SHEEP(SH),32)
1710 YSHEEP(SH)=Y
1720 XSHEEP(SH)=X
1730 GOSUB 1760
1740 NEXT SH
1750 RETURN
1760 REM COUNT SHEEP IN PEN
1770 SHINPEN=0
1780 FOR I=1 TO NUMSHEEP
1790 X=XSHEEP(I)
1800 Y=YSHEEP(I)
1810 IF (X<7)+(X>11)+(Y<7)+(
Y>10)THEN 1840
1820 SHINPEN=SHINPEN+1
1830 IF SHINPEN=NUMSHEEP THE
N 1860
1840 NEXT I
1850 RETURN
1860 M$="WELL DONE YOU WER G
REAT"
1870 FOR I=1 TO LEN(M$)
1880 CALL HCHAR(20,5+I,ASC(S
EG$(M$,I,1)))
1890 NEXT I
1900 GOTO 1930
1910 CALL CLEAR
1920 PRINT "YOU RAN OUT OF T
IME"
1930 PRINT "DO YOU WISH TO P
LAY AGAIN"
1940 INPUT "(Y/N) ":A$
1950 IF SEG$(A$,1,1)="Y" THE
N 420
1960 STOP
```



Programs That Write Programs

Part 1

by Jim Peterson

Way back in 1982, in the old 99'er Magazine, Vol. 1 Nos. 3 and 4, John Clulow wrote two articles entitled "How To Write a Basic Program That Writes Basic Programs". At that time I thought I would never understand what he was writing about!

But really, it's simple. Have you ever LISTed a program to the disk, and noticed that the resulting D/V80 file took up many more sectors than the program itself? That is because the TI saves programs in a compacted form, with each statement represented by a single token ASCII.

When a program is saved in MERGE format, by SAVE DSK (file- name),MERGE it is saved in this same compacted form, but in a D/V 163 file. And of course a D/V file can be created by a program - so you can write a program which will create a D/V 163 file in the form of a program, and then MERGE that file into memory and RUN it as a program, and SAVE it as a program.

You ask, why use this roundabout way of writing a program? Why not just key it in directly? Well, for one thing you can write program lines that could not possibly be keyed in directly. As for instance, the famous "line zero". Key this in, run it with a disk in drive 1, then enter MERGE DSK1.ZERO and LIST the result.

```
100 M$="BETCHA CAN'T DELETE
THIS!"
110 OPEN #1:"DSK1.ZERO",VARI
ABLE 163,OUTPUT :: PRINT #1:
CHR$(0)&CHR$(0)&CHR$(131)&CH
R$(200)&CHR$(LEN(M$))&M$&CHR
$(0)
120 PRINT #1:CHR$(255)&CHR$(
255):: CLOSE #1 :: END
```

Actually, there is an easy way to delete that line - but no way to key it in directly.

Here's another one - the full ASCII string.

```
100 OPEN #1:"DSK1.FULLSTRING
",VARIABLE 163,OUTPUT
110 LN=100 :: GOSUB 190 :: A
$=L$&"M$"&CHR$(190)
120 FOR J=1 TO 127 :: C$=C$&
CHR$(J):: NEXT J :: A$=A$&CH
R$(199)&CHR$(127)&C$&CHR$(0)
130 PRINT #1:A$
140 GOSUB 190 :: B$=L$&"M2$"
&CHR$(190)
150 FOR J=128 TO 255 :: D$=D
$&CHR$(J):: NEXT J :: B$=B$&
CHR$(199)&CHR$(128)&D$&CHR$(
0)
160 PRINT #1:B$
170 GOSUB 190 :: F$=L$&"M$"&
CHR$(190)&"M$"&CHR$(184)&"M2
$"&CHR$(0)
180 PRINT #1:F$ :: PRINT #1:
CHR$(255)&CHR$(255):: CLOSE
#1 :: END
190 L$=CHR$(INT(LN/256))&CHR
$(LN-256*INT(LN/256)):: LN=L
N+10 :: RETURN
```

Run that, then enter NEW, then MERGE DSK1.FULLSTRING. The string contains every ASCII from 0 to 255 in sequence, and there is no way to ènter many of the unprintable ASCII codes from the keyboard. You can of course create that string in a program - FOR J=0 TO 255 :: M$=M$&CHR$ (J):: NEXT J but it saves a few seconds to have it predefined.

The full ASCII string is very useful for a quick shuffle without duplication. For instance, to scramble the numbers 200-250, -

```
100 M$="
                 !""#$%&'()*+,-./
0123456789:;<=>?@ABCDEFGHIJK
LMNOPQRSTUVWXYZ[±]²_°abcdefg
"
110 M2$="



                             "
120 M$=M$&M2$
130 M$=SEG$(M$,200,50)
140 L=LEN(M$):: RANDOMIZE ::
 X=INT(L*RND+1):: N=ASC(SEG$
(M$,X,1)):: M$=SEG$(M$,1,X-1
)&SEG$(M$,X+1,255)
150 PRINT N;:: IF LEN(M$)=0
THEN STOP ELSE 140
```

One more example - can you run this program and get these results? You won't even be able to key in that last line!

```
>LIST
100 FOR J=1 TO 7 :: READ M$
:: PRINT M$ :: NEXT J
30000 DATA AAAAAAAAAAAAAAAA
AAAAAAAAAAA,BBBBBBBBBBBBB,BB
BBBBBBBBBBBB,CCCCCCCCCCCCC,
             DDDDDDDDDDDDDD
30010 DATA "TESTING",,,,,,,,
,,,,,,,,,,,,,,,,,,,,,""TEST
ING""
>RUN
AAAAAAAAAAAAAAAAAAAAAAAAAAA
BBBBBBBBBBBBB,BBBBBBBBBBBBBB
CCCCCCCCCCCCC
              DDDDDDDDDDDDDD
"TESTING"

,,,,,,,,,,,,,,,,,,,,,,,,,,,,
""TESTING""

*READY*
```

Next month - the answer to that puzzle, and a more useful program that writes a program, and then we will start learning how you too can write programs that write programs! o

* * * * * * *

ALGORITHM DESIGN

An algorithm is a formula, a recipe, a step-by-step procedure to be followed in order to obtain the solution to a problem. To be useful as a basis for writing a program, the algorithm must:

a) arrive at a correct solution within a finite time,
b) be clear, precise and unambiguous, and
c) be in a format which lends itself to an elegant implementation in a programming language.

Algorithm specification has traditionally been done by drawing flowcharts which are a diagrammatic representation of the steps involved. Pseudocode is now generally used in preference to flowcharting.o

THE COMMUNICATORS

Special Interest Group for Users
of the TEXPAC Bulletin Board Service.
by Ross Mudie, SYSOP, 3rd August 1987.

1. SIZE OF DOWN LOADABLE PROGRAMS INCREASED.

The maximum size of programs capable of being down-loaded from the TEXPAC BBS has been increased from 40 disk sectors to 50 disk sectors. A one byte value in the assembly part of the download routine was found to be the limiting factor. Users who have disk systems will need to use CALL FILES(1) before typing OLD RS232 to free up sufficient space in the VDP RAM for programs which exceed 46 disk sectors in size. The BBS will also provide a warning message with beeps when CALL FILES(1) is required.

At 300 bauds, which is the current speed of communications with the BBS, each disk sector takes 8 seconds to transfer. As a consequence a program of 45 disk sectors will take 6 minutes to transfer.

2. TE2 GRAPHICS MODE.

During spare time in the July holiday, son Peter and I developed a file which switched the caller's TE2 into the 32 column GRAPHICS mode in lieu of the normal 40 column TEXT mode. Character redefinition for the Graphics mode of TE2 is very time consuming, but it was an interesting exercise. The file was placed on the BBS using the REMOTE SYSOP facilities early in July using the file name TE2_ONLY . If the file is read using another Terminal Emulator, such as TE1, Fast Term or 4A-Talk, then the file appears as lots of funny characters and a bit of text, often in illogical order.

To reset TE2 back to the 40 column TEXT mode, a file named TE2_RESET is included to enable ease of restoration to the TEXT mode.

The BBS mail system will accept TE2 graphic screens (or RLE screens) providing that they are in DIS/VAR 80 format with characters in the range of ascii 16 to ascii 127 inclusive. The files may be sent to the BBS from your disk system using the program SENDMAIL4 which is available on the downloadable programs menu. The SENDMAIL4 program replaces all records which consist of only a carriage return (ascii 13) with a space (asc 32) to prevent premature departure from the mail routine.

Full details for TE2 graphics mode, written in classic TI language, (how to make something which is reasonably complex into absolutely impossible for the layman), is to be found in the publication TERMINAL EMULATOR PROTOCOL MANUAL which is available for loan from the club publications library. (Anyone who is battling with the Editor-Assembler manual will understand what I mean here!!)

3. CLEAR SCREEN WITH TE2.

TE2 users will note that the BBS clears the screen when going to a new file or function, menu etc. If this is too quick for you here is a method of reading the text which was lost from the screen.

PAUSE the BBS with <ctrl> S , then once paused use <fctn> X, (down arrow), to review the information which just disappeared from view. To UNPAUSE use <ctrl> Q . It is recommended that <ctrl> X is not attempted unless PAUSE is invoked or the BBS is waiting for a response to a prompt.

4. LOG ON PROBLEMS.

There have been 3 cases of people having problems logging on recently due to faulty modems. These are often difficult problems to diagnose. If you decide to try your modem on someone elses computer take your full system along including the cords to try to resolve the problem at your friend's place by mixing and matching. Don't fall for the trap of taking the modem without the connecting cords.

One of the modem problems was due to the fact that the telephone was left connected in parallel with the modem, even though the phone was hung up when the modem was on line. The only way to make this particular modem interwork with the BBS was to unplug the telephone when the modem was on line. The standard Telecom method of connecting modems switches ALL telephones off the line when the modem is on line. Double adaptor phone plugs may appear to be a cheap and easy method of connecting both modem and telephone but make sure of the proper disconnection of all phones when on line.

If all else fails then contact the SYSOP, preferably weekdays between, 7.30am and 2.30pm, on (02) 663 0141. Often problems can be diagnosed by observing the actual operation of the BBS and checks for both audio level & frequency may be conveniently conducted.

Don't forget to wait for the HIGH tone, which is heard when the modem answers, to be replaced by the LOW tone from the BBS modem BEFORE switching your modem on line. This should overcome most of the log on problems, including the problem of the screen colour not changing to white on blue when TEII is in use.

5. BBS USAGE, TIME LIMITS CHANGED.

Following previous items on the subject of falling usage of the BBS some additional reasons are emerging. Some users agree with the cold weather theory, whilst others suggest school examinations, the natural loss of interest from a "fad", users graduating to bigger and better (IBMs & UNIX) & even, (these are not my words), "their imaginations and not their temperatures are too low!!"

It is interesting to note that 34% of TIsHUG's members are registered to use the BBS.

The time limits on usage have been again reviewed in light of usage patterns and the 60 minute limit now applies from midnight to 2pm daily. A 30 minute limit applies from 2pm to midnight. The time limit operates as a reminder only and does not cut the call off, so if what you are doing on the BBS takes a little over the time limit, then thats a fair thing. The whole purpose of having a time limit on calls is to share the access to the BBS to the benefit of ALL users. If you want more time than the already generous limits then leave a break of 30 minutes before calling again to allow others to have a chance of getting on. Function 9 on the Main Menu of the BBS will not extend the time limit as I have observed some users have tried when running over time.

The times are taken from the time at log on. The 60 minute time applies to calls which commence after midnight and before 2pm. The 30 minute time is for calls which commence after 2pm and before midnight.

6. CHANGE IN PROMPT AFTER NEWS ITEMS.

There has been another minor change to the prompt which is received after reading a file. The prompt is now:

```
   [M]ain[#], [N]ews menu or News # >
```

The change is the inclusion of the optional number # after the M . If a user types M2 they will go directly to the Program Download Menu which is option 2 on the Main Menu. M7 will give the User Log etc. By typing M at this prompt the Main Menu will be given, N will give the News Menu and a number will give the corrosponding file number from the News Menu.

The valid range of M numbers is M1 to M9 only, since the range of the Main Menu is 1 to 9.

7. SENDMAIL FOR CASSETTE USERS.

A version of SENDMAIL has been written for Cassette only users who have 32K Memory Expansion in their console. These users can compose the text of their mail in DATA statements in the special version of SENDMAIL and then save the program containing the mail back to tape. When ready to send the mail item the user may exit TE2 with <CTRL> 0 then switch over to Extended Basic & load the SENDMAIL program containing the mail from tape. When the program is run the user selects the RS232 port and the mail is quickly transferred into the BBS.

Cassette users who are calling STD into the BBS are best advised to send mail manually, unless they are a very slow typist as the STD meter is still ticking up whilst the SENDMAIL program is loading from tape. Once a user becomes used to the program it could be slightly shortened by removing some of the instructions and remarks.

The SENDMAIL program is named SNDMAIL2-4 in the Program Menu on the BBS.

8. BBS MEMBERSHIP.

a) Lost Renewals.

There have been two cases where members paid BBS renewal with their TIsHUG membership renewal but the fact of BBS renewal has not been properly recorded and these members have been removed from the BBS as unfinancial. If any financial user can not get past the INVALID message, then please contact either SECRETARY, Terry Phillips or SYSOP, Ross Mudie.

b) How to Join the BBS.

Members of TIsHUG pay an additional $5 membership fee to gain access to the BBS. All requests for BBS membership should be sent to the SECRETARY TIsHUG, PO Box 214, Redfern. N.S.W. 2016. You should nominate your preferred user name (a nickname is usually suitable). The minimum system for BBS access is a TI99/4A, TV, TE2 module, RS232 card or stand alone and a Modem, with ready access to your telephone line. If living outside the Sydney metropolitan area, then remember that using the BBS can do wonders for your phone bill! o

HOW TO CONDENSE ASSEMBLY IN EXTENDED BASIC USING CALL LOADS.

by Ross Mudie of TIsHUG, 4th August 1987.

The ACE program by Paulo Bagnaresi converts an assembly program which may be linked from extended basic into CALL LOADS. This allows the assembly object code to be included in the extended basic program which is suitable for use with cassette loading or to simply overcome the need to load the assembly object file from disk.

A shortcoming of the ACE program is that it can not differentiate between a memory assignment which is made using BSS and the assignment of constants or even program. The resultant CALL LOAD format assembly is unnecessarily lengthy due to the inclusion of whatever was in the BSSed assignments when the code was passed through the conversion process.

After running ACE the assembly is in extended basic MERGE format which must then be MERGEd into extended basic PROGRAM format. The CALL LOADS may then be edited using the extended basic program editor to remove the unnecessary bytes. It is best to place the BSS assignments either near the start or end of the source file to make it easy to find. Take care not to affect the DEF table, (8 byte entries at 16376, working down) and the pointers to the start of the free space in low RAM plus the start of the DEF table (8194). The absolute addressing must not be affected but all unnecessary CALL LOADS may be removed.

A program to convert assembly to CALL LOADS appeared in the March 1987 Micropendium and will be again available in the selection of downloadable programs on the TEXPAC BBS during September 1987. o

SORTS.

by R.N.Harris

In a Bubble Sort you have the process rather like one guy sorting mail in a Post Office, finding the things and putting them into pigeon holes.

In Shell Sort the guy has an assistant to put things away.

In Quick Sort the guy has TWO assistants to put things away.

I have already submitted a Quick Sort and Terry may have it on Disk. The programme may need a little experimentation, but what it assumes is a Speech Synthesizer and a Cassette Recorder and a Printer so that when the Sort is finished you can take notes, or listen hands free, save the data to Cassette so you don't have to rerun the entry, and give you a printout as well. The output loop speaks, tapes, and prints. The input loop assumes you have compressed the data into 32-character strings which it displays across the screen. I have put a display loop in so that the programme will print to the screen in 32-column displays. The irritation of a 28-column screen is thereby palliated, and the CALL KEY method of inputting is utilized. Most of my programmes are self-documenting and I think the Quicksort will be useful to anybody as it really is quick, and amazes people who think only Assembler can give such speedy output!

One can find SORTs in one of the SAMs books on the Texas Instruments Computer. o

TI-WRITER HELP

by Tom Kennedy

Reprinted from the BOSTON COMPUTER SOCIETY
TI 99/4A User Group TI-Writer Help Disk

PART 1: THE EDITOR

How many of you have a typewriter, please raise your hand. Keep your hand up if your typewriter has interchangeable text. How about automatic bold and underline? Or some amount of memory storage (for letter heads, etc.)? How about an erase key? Those of you left have probably got a pretty expensive piece of machinery, but TI-WRITER has ten times the functions, or features of the best typewriters. With TI-WRITER, your only limitation is your own creativity.

To start off with, what will you need to operate your Word Processor? You must have the 99/4A console (TI-WRITER won't work with the 99/4), a TV or monitor, the cartridge and disk package, the disk system, memory expansion, the RS232 interface, and a printer. In other words, the whole works. The printer is something you definitely want to be careful in choosing because all of your work will be in vain if you can't print out exactly what you type in, and with an attractive appearance.

First, let's look at the command line. That's the line at the top of the screen when you're in the command mode. There are seven commands shown and sixteen sub-commands that are options of the main seven. The commands are selected by typing only the letters that are capitalized in the word. For instance: "F" for Files, "SH" for SearcH, or "LF" for Load File. That's an interesting point: you can access any of the sub-commands from the main command menu. In other words, to ShowDirectory (which is a disk catalog) you would enter the command mode, (FCTN 9), and either type "F" for files, and "SD" for ShowDirectory, or just type "SD" immediately. This feature saves a lot of time and keystrokes.

The first command is Edit. This simply enters you into the text-edit mode in which text is created.

Next is Tabs. When you hit "T", the top part of your text is shown with a scale across the top showing the current tabs and margins. Changes are made by simply typing over existing entries with the appropriate symbol (L,R,T, or I).

"F" for files allows you to work with your text file as a whole. To Load, Save, Delete, Print, Purge, or ShowDirectory. "PF" for print file is not what you'll get when you print out through the text formatter; it just prints a "hard copy" of the whole file, just as you see it on the screen. It doesn't print with any of the modifications made by the format commands (more on those later). "PF" is useful for making a fast copy of a long letter, or whatever, in order to check for errors without having to scroll back and forth or up and down. Purge simply erases the file from memory to prepare for a new entry. It is similar to the "NEW" command in BASIC.

Next is "L" for Lines. This allows you to work with whole lines or groups of lines by moving them to somewhere else in the text, copying to somewhere else and leaving the original intact, to delete groups of lines, or to quickly move the cursor to some line in the text with the ShowLines option.

Search (or "SH") gives you the option of either the FindString routine or the ReplaceString routine. FindString will move the cursor to the first and/or each successive use of the word string you give. ReplaceString searches the text for a given string and replaces all or one occurrence with the new string. This is great for correcting a repetitive spelling error.

RecoverEdit is a failsafe repair in case the text buffer was purged in either the File or Quit command. It will pull back everything but the first line and restore the file. I guess the loss of the first line is the penalty paid for accidentally erasing a file, which can't be done very easily.

Finally, Quit, as the name implies, blows it all apart and leaves you with the title frame. But before it goes, all open files are closed (such as to disk or printer) so no data is lost. Fortunately, it first gives you the option of saving your file (in case you forgot to do that already) or just purging the file and going back to the edit mode. But if you really want to quit, you type "E" for Exit and it shuts down.

Now let's go over the keyboard. TI-WRITER makes extensive use of the FCTN and CTRL keys and uses every possible function of the top line of keys (the numbers). There are also many functions that have duplicate methods of keystrokes to activate them. For instance, to enter the command mode, you either press FCTN 9 or CTRL C. The reason for this duplication is to allow you to choose which is easiest to use depending on where your fingers are at. The problem though, is that it can be very confusing trying to remember the fifty different key combinations that activate the thirty functions. A better method is to just pick which keys you're going to use for what function and ignore the rest. What I do is use the number line keys for anything shown on the overlay strip and just memorize the few functions hidden down in the keyboard. Let's start by going down the overlay strip, left to right.

```
*************************************************************************
OOPS!     * CTRL 1 *    This can be a real lifesaver. It recovers, or "backs up"
          *(CTRL Z)* a function that you didn't mean to hit. Like if you goofed
          *        * and hit "Delete Line" instead of "Insert Character", you
          *        * just hit "OOPS!" and the line comes back.
Del Char  * FCTN 1 *    This is the same as "DEL" in console BASIC. It deletes
          *(CTRL F)* one character under the cursor and pulls the rest of the
          *        * line up to fill.
Reformat  * CTRL 2 *    This is used to close up the text after using Insert
          *(CTRL R)* Character. It deletes all spaces between the cursor and the
          *        * next word in the text. Then it draws all subsequent words up
          *        * through the paragraph until it encounters a Carriage Return.
Ins Char  * FCTN 2 *    In the Word Wrap mode (solid cursor), thirty two blank
          *(CTRL G)* characters are inserted after the cursor and the bulk of the
          *        * text is pushed down the line. After insertion of new text,
          *        * you hit Reformat and any remaining spaces are removed. In
          *        * the Fixed mode (hollow cursor), this operates the same as in
          *        * console BASIC.
Screen    * CTRL 3 *    This allows you to choose which of the five color
  Color   *        * combinations of text/screen you prefer. The default, for no
          *        * good reason, is white on dark blue. But I find this hard on
          *        * the eyes. I prefer to turn down the color on my monitor and
          *        * use either black on green or black on light blue.
Del Line  * FCTN 3 *    Deletes the entire line that the cursor is on, including
          *(CTRL N)* the space of the line.
Next      * CRTL 4 *    This  advances  the cursor  to the beginning  of the
Paragraph*(CTRL J)* following paragraph and puts the first line at the top of
          *        * the page.
```

```
Roll Down* FCTN 4 *    This is called a "vertical block scroll", which means
         *        * that the next 24 lines of text are shown. This is handy for
         *        * scanning quickly down the text to get to some point.
Dupe Line* CTRL 5 *    This creates an exact duplicate of the line the cursor
         *        * is on and places it directly below. Some have questioned
         *        * its value in writing text, especially since the Move/Copy
         *        * function can do the same, but this key makes it faster and
         *        * easier to create repetitive lines such as a double row of
         *        * asterisks under a title.
Next     * FCTN 5 *    This is a "horizontal block scroll". It jumps across to
  Window *        * display the next block of 40 characters, in increments of
         *        * 20. For example, the screen starts out on column one to
         *        * forty, then twenty to sixty, then forty to eighty.
Last     * CTRL 6 * The opposite of "Next Paragraph"
Paragraph*(CTRL H)*
Roll up  * FCTN 6 * The opposite of "Roll Down"
         *(CTRL B)*
Word Tab * CTRL 7 *    This moves the cursor down the line to the first letter
         *(CTRL W)* of each word.
Tab      * FCTN 7 *    Just like on a typewriter, this moves the cursor to next
         *(CTRL I)* setting, defined using the Tab function on the command line.
New      * CTRL 8 *    This places a Carriage Return symbol at the end of the
Paragraph*        * line you're on and skips down to the next line. If you have
         *        * preset an auto-indent, (by using an "I" in Tabs) then it
         *        * also indents over to the proper column.
Ins Line * FCTN 8 *    Inserts a blank line above the line the cursor is on.
         *(CTRL O)*
New Page * CTRL 9 *    Inserts a blank line with a Np and Cr symbol at the
         *        * beginning. This causes the printer to feed to the next page.
Command/ * FCTN 9 *    This is how you exit from the edit mode to get to the
  Escape *(CTRL C)* command line and the functions above it. It also is used to
         *        * cancel a command already in progress.
Word Wrap* CTRL 0 *    This switches from the "Word Wrap" mode to the "Fixed"
         *        * mode. In Word Wrap, when you reach the end of the line the
         *        * cursor jumps to the next line. If you're in the middle of
         *        * a word at the end of the line, the whole word you were on
         *        * moves down too. This allows you to just type continuously
         *        * without looking up to see when to hit enter. In the fixed
         *        * mode, when you reach the end of the line your letters just
         *        * pile on top of each other and you hit enter to move to the
         *        * next line.
Line     * FCTN 0 *    This removes or displays the four-digit line numbers
 Numbers *        * at the left side of the screen. The numbers are used for
         *        * reference when manipulating blocks or lines of text, just
         *        * like when you're editing a BASIC program. You need line
         *        * numbers to refer to where changes will be made.
Quit     * FCTN = *    Quit is the same as in console BASIC. Use Quit option
         *        * of the Command line to safely exit TI-WRITER.
Back Tab * CTRL T *    The same as Tab except it backs up one setting.
         *        *
Beginning* CTRL V *    Moves the cursor to the beginning of the line you're on.
 of Line *        *
Del.End  * CTRL K *    This is just like Delete Character (FCTN 1), except it
  of Line*        * takes out everything to the right of the cursor.
Home     * CTRL L *    This moves the cursor to row 1, column 1, on the screen
  Cursor *        * only. Unfortunately, it doesn't move to first line of text,
         *        * which would be more convenient when you were at the a long
         *        * document and wanted to jump to the top. [For that, enter S,
         *        * then enter 1.]
Left Mrgn* CTRL Y *    Allows you to temporarily back-arrow beyond the left
  Release*        * margin when it has been set past zero.
********************************************************************************
```

The last four key functions to mention are the cursor arrows: UP, DOWN, LEFT, & RIGHT. These stay the same as in console BASIC.

Now, if you're still following along you may be quite confused with this onslaught of information. The point is, you can't learn all of this in one sitting, but after using TI-WRITER for a while you start to pick things up as you need them. Rest assured, you do spend the majority of your time typing. The purpose of most of the functions I've mentioned are to manipulate the text which is already in the file. I have simply tried to cover all of this in order to bring something to your attention that you might have missed, or to peak your interest in the capability of the TI-WRITER software.

To review, in the command mode we can choose between Edit, Tabs, Files, Lines, SearcH, RecoverEdit, or Quit. As sub-commands of those seven, we can choose Load File, Save File, Print File, Delete File, Purge, ShowDirectory, Move Lines, Copy Lines, Delete Lines, Showlines, FindString, ReplaceString, or Exit.

[Added note: TK-WRITER does not support the SD option.]

PART 2: THE FORMATTER

Now I want to cover the Text Formatter, which prints out the document. Most importantly, the special symbols, called Format Commands, that the formatter uses to alter the print-out of the document, which are installed in the Text Editor.

In other words, you put these commands into the text when you write it and as the formatter comes across them it changes the text accordingly but doesn't actually print the symbols.

There are six groups of formatter commands that are all applied in a similar manner. All commands must be in capitals and must be on a line that starts with a period.

Continued on P23

Continued from P22

```
*******************************************************************************
     Text Dimension commands, as the name implies, move or shape the words in
the document (margins, linespacing, right justify, etc.)
 .FI   : FILL     : PUTS AS MANY WORDS ON A LINE AS WILL FIT.
 .NF   : NO FILL  : CANCELS FILL.
 .AD   : ADJUST   : ALIGNS THE TEXT TO THE LEFT AND RIGHT MARGINS. (RT. JUSTIFY)
 .NA   : NO ADJUST: CANCELS ADJUST.
 .LM n : LF MARGIN: SETS LEFT MARGIN TO "n".
 .RM n : RT MARGIN: SETS RIGHT MARGIN TO "n".
 .IN n : INDENT   : CREATES AN AUTO-INDENT FROM LEFT MARGIN.
 .LS n : LINE SP  : SETS LINE SPACING TO "n" LINES.
 .PL n : PG LENGTH: DEFINES NUMBER OF LINES TO A PAGE.
 .BP   : BEGIN PG : DEFINES FIRST LINE OF NEW PAGE.

     Internal Format commands control the spacing of characters on a line.
 .SP n : SPACE    : SIMILAR TO THE TAB FUNCTION.
 .CE n : CENTER   : CENTERS NEXT "n" LINES BETWEEN MARGINS.

     Highlighting commands control functions such as underline or bold and
allow you to redefine characters to use them to send CTRL codes to the printer.
  ^    : REQUIRED : JOINS WORDS TOGETHER WHEN REQUIRED TO PREVENT SPLITTING IN
       :   SPACE  :      REFORMATING, UNDERLINE, ETC.
  &    : UNDERLINE: (UNDERSCORE) UNDERLINES ALL TEXT FOLLOWING UNTIL NEXT PACE.
  @    : BOLD     : (OVERSTRIKE) RETYPES FOLLOWING TEXT FOUR TIMES.
 .TL xx: TRANS-   : ALLOWS REASSIGNMENT OF ONE CHARACTER TO REPRESENT A NUMBER.
       : LITERATE : OF CHARACTER VALUES TO SEND CODES TO THE PRINTER.
 .CO t : COMMENT  : SIMILAR TO REM IN BASIC--ALLOWS NOTES THAT DONT PRINT.

     Page identification commands print notes in the upper or lower corner of
each page, either headers or footers.
 .HE t : HEADER   : PRINTS TEXT (t) AND PAGE NUMBER AT TOP OF EACH PAGE.
 .FO t : FOOTER   : PRINTS TEXT (t) AND PAGE NUMBER AT BOTTOM OF EACH PAGE.
 .PA   : PAGE #   : RESETS PAGE NUMBER IN .HE AND .FO

     File management commands
 .IF f : INCLUDE  : MERGES A FILE TO PRINT A DOCUMENT TOO LARGE FOR ONE FILE.
       :  FILE    :

     Mail Merge option commands are used to supply values to the variables in a
letter that has been set up for the mail merge option
 .ML f  :MAIL LIST: IDENTIFIES VALUE FILE (f) FOR MAIL LIST.
 *n*    :VARIABLE : INSERTED IN TEXT AS VARIABLE FOR ASSIGNMENT FROM VALUE FILE.
 .DP n:t:DISPLAY  : PROMPTS YOU USING TEXT "t" TO ASSIGN TO VARIABLE (*n*).
        :  PROMPT :
*******************************************************************************
```

The use of these commands in your text is what separates the word processor from a typewriter. They allow you to get the most out of your printer.

So, now you've written your document, and inserted all the format commands, now how do you print it out? First, save the document and exit the Text Editor. At the title menu, select Text formatter, (make sure the program disk is in the drive) and the screen will blank with the prompt "ENTER INPUT FILENAME". Enter the name of the file you just saved, (ex. DSK1.MYFILE) and hit enter.

Next, the prompt "ENTER PRINT DEVICENAME" appears after the file is loaded. If you use a serial printer, the device name would be RS232.BA=xxx with xxx being the baud rate. If you're using a parallel printer, the device name is PIO. Also, you must add either .CR or .LF to the end of the device name. This tells TI-Writer whether your printer will handle the carriage return or the line feed. Check your printer manual and the TI-Writer manual in detail to find out which you use.

The next prompt is "USE MAILING LIST". If you aren't printing "form letters" just hit enter to accept the default of N (NO).

Next is "WHAT PAGE(S)? <ALL>. If you want to print the whole document, accept the default for all pages. Otherwise, you can print any of the pages or groups of pages.

The prompt "NUMBER OF COPIES: 1" tells how many copies of each page are to be printed.

The last prompt is "PAUSE AT END OF PAGE? N". The main purpose of this function is if you are using separate sheets of paper it will stop and wait for you to align the next sheet.

Now, about the Mailing List Option. Let's say you've written a form letter to send out to various individuals, maybe a resume. You write the letter as normal, but when you come to a name or an address or something that will change with each letter, you put in its place a variable in the form of *n*, where n is a number to identify the order. So instead of starting off with:"Dear Mr. Smith" you would use "Dear Mr. *1*" and so on. When you're all through with your letter, save it and purge the memory. Now you must create what is called a Value File, which is the mailing list from which TI-Writer will draw the variables. A value file consists of a list of values to be inserted into the letter, listed one to a line, preceded by the number of the variable and ending with a carriage return symbol. Groups of values must be separated by a line with just an asterisk and a carriage return. For example:

```
1 John Smith
2 123 STREET
3 Seattle, WA
*
1 Jane Doe
2 456 STREET
3 Seattle, WA
```

At the top of your letter you insert the .ML f command where f equals the filename of your value file. After selecting the mailing list option the computer will use this command to fill in the variables. If there is no .ML command in the letter then when you are prompted for "MAILING LIST NAME:" you supply the filename. This allows you to call on a number of files for different groups.

TI Writer BUGS
by Jim Peterson
TIGERCUB SOFTWARE

Dr. Guy-Stefan Romano has confirmed and explained the pestiferous asterisk bug in TI-Writer. If you are printing out of the Formatter mode and your text contains an asterisk followed by two or more numeric digits - the asterisk and two digits will disappear! For instance, A*256 becomes A6, and I've noticed that A6 in programs published in several newsletters recently.

The TI-Writer program misinterprets the asterisk and two digits as an instruction to input data from a "value file" (see Alternate Input on p. 111 of the TI-Writer manual).

The solution to this bug is to type two asterisks followed by two dummy digits, then the actual digits. For instance, instead of A*256 type A**25256. Trouble is, the bug usually shows up in a program which has been LISTed to disk and then MERGEd into TI-Writer, and is usually not noticed. The solution? Run the program through my 28-Column Converter.

Dr. Romano informs me that there is an even worse bug in the Transliterate command coding, erratic and sometimes destructive. It is triggered by certain sequences of characters, but these have not been documented. Dr. Romano says that he does not use transliteration.

I would suggest that you also avoid the use of the & and @. The & will only underline a single word unless you tie words together with the ^ sign. If you tie words together, the Fill and Adjust will leave gaping blanks in your lines and if you tie too many together the line will extend beyond the right margin! Also, the underlining is a broken line. It is better to use the escape codes (these are for an Epson or Epson compatable printer) CTRL U, FCTN R, CTRL U, SHIFT -, CTRL U, SHIFT @, CTRL U, which will give a solid underline until you turn it off with CTRL U, FCTN R, CTRL U, SHIFT -, CTRL U, SHIFT @, CTRL U.

The @ is handy to emphasize a single word, but if you want to double-strike a whole sentence or paragraph it is better to use the escape code CTRL U, FCTN R, CTRL U, SHIFT G, and turn it off again with CTRL U, FCTN R, CTRL U, SHIFT H.

The period bug is another killer - the Formatter thinks that any line which begins with a period is a formatter command, and deletes the whole line! If your text contains a decimal value such as .11 and the wrap-around puts it at the beginning of a line, the line disappears! There are two ways around this - put a 0 in front of all your decimals, as 0.11, or trans-literate all your periods.

In all the TI-Writer formatter is a temperamental and unpredictable piece of software, prone to unwanted line feeds and unexpected paper- wasting form feeds. I like to use it to right-justify text back to the disk, but from then on I prefer to print it out of the editor mode, or out of my own program. o

MAKING AN OVERLAY FOR FUNELWRITER/TI WRITER

In the August 1986 issue of MicroPendium, there was an article and program which would enable you to run off your own overlays for all sorts of software packages. Now I am aware that a lot of members have a copy of Funlwriter and may not be aware of the range of keystrokes needed to access a lot of the Editors functions. A related article in this issue will give you the commands, but it is always difficult having to refer to a printed document until you have memorised them. So here's what I reckon you should do.

1. Get a copy of the forementioned Micropendium and type in the program. If you're lucky someone might have already typed it in and you can get a copy from him or her.

2. Having gotten the program up and running you will be prompted to input the various activities accessed by the Function and Control keys. Here's a list of them for you to type in:

UPPER KEY 1	- OOPS!
UPPER KEY 2	- REFORMAT
UPPER KEY 3	- SCREEN COLOR
UPPER KEY 4	- NEXT PARAGRAPH
UPPER KEY 5	- DUPE LINE
UPPER KEY 6	- LAST PARAGRAPH
UPPER KEY 7	- WORD TAB
UPPER KEY 8	- NEW PARAGRAPH
UPPER KEY 9	- NEW PAGE
UPPER KEY 0	- WORD WRAP
UPPER KEY +	- (LEFT BLANK)

LOWER KEY 1	- DEL CHAR
LOWER KEY 2	- INS CHAR
LOWER KEY 3	- DEL LINE
LOWER KEY 4	- ROLL DOWN
LOWER KEY 5	- NEXT WINDOW
LOWER KEY 6	- ROLL UP
LOWER KEY 7	- TAB
LOWER KEY 8	- INS LINE
LOWER KEY 9	- COMMAND/ESCAPE
LOWER KEY 0	- LINE #'S
LOWER KEY =	- QUIT

The Upper Keys are accessed with the Control Key, while the Lower Keys are accessed with the Function Key.

3. Having entered all the information, do a print out when prompted.

4. Neatly trim the resulting printout and glue onto a piece of stiff cardboard and allow to thoroughly dry.

5. Again trim up and place the overlay on your keyboard and there you will have it, a constant reminder of the most used commands. o

Continued from P23

Another way to insert values is to use the Define Prompt command. With this command you do not insert a .ML comand calling a value file and instead you insert lines containing the format: .DP n:t - where n is the number of the variable and t is the prompt text. Now, when you come to the prompt "USE MAILING LIST?" you select "N" for NO and as the document is printed when a variable is encountered the printing stops and the text you chose appears on the screen asking you for the appropriate value. If you don't include a ".DP n:t" command in your text, the computer responds with "ENTER DATA FOR VARIABLE *n*" and it can get confusing trying to remember which item you're on. This method is handy for letters which you only want to print one copy at different times to different people.

Let me tell you, this is why I bought a computer. I'm sure we all went through that period of time before buying a computer when we would ask: "what am I going to use a computer for, anyway?". Well I decided there were two things I wanted to do: 1) Store files of data (recipes, albums, etc.) and 2) Use my computer as a typewriter. I didn't know about TI-WRITER when I bought the 99/4A, but now I know that I made the best choice possible. I hope you will all find TI-WRITER as easy to use and as powerful as I have.

If you have any questions, I'll do my best to help. Just ask!

Tom Kennedy o

FORUM

Following a number of requests from members who have Funelwriter and not the TI Writer reference manual the following article has been prepared to show the commands available for the Editor and Formatter. Members are advised that copies of the TI Writer reference manual are now available from the shop. This file will also be available at the shop in September for those that would like a disk copy

TEXT EDITOR - EDITING OPERATIONS.

Back Tab - CTRL T - Moves the cursor one tab setting to the left.

Beginning of Line - CTRL V - Moves the cursor to the beginning of the line on which it is located.

Command/Escape - FCTN 9 or CTRL C - Exits from Edit Mode into Command Mode. Escapes a command.

Delete Character - FCTN 1 or CTRL F - Deletes character by character including the space.

Delete End of Line - CTRL K - Deletes all text to the right of the cursor to the end of the line, including the chaaracter under the cursor.

Delete Line - FCTN 3 or CTRL N - Deletes the entire line of text and the line space.

Display Line Numbers - FCTN 0 (Zero) - Removes or redisplays line numbers on the screen.

Down Arrow - FCTN X or CTRL X - Moves the cursor down. If the cursor is on the last line of the screen, subsequent lines scroll onto the screen one at a time.

Duplicate Line - CTRL 5 - Duplicates the line above and replaces the line the cursor is on with the duplicated line.

Home Cursor - CTRL L - Repositions the cursor at the upper left corner of the screen without altering the display.

Insert Blank Line - FCTN 2 or CTRL G - WORD WRAP: Splits a line into two lines for insertion. Termiated by Reformat. FIXED: Pushes the remainder of the line to the right during insertion. Text pushed past the right margin is deleted.

Last Paragraph - CTRL 6 or CTRL H - Moves the cursor to the beginning of the preceding paragraph in the text.

Left Arrow - FCTN S or CTRL S - Moves the cursor to the left without blanking our text.

Left Margin Release - CTRL Y - Temporarily disables the left margin until the cursor recrosses the left margin.

New Page - CTRL 9 or CTRL P - Inserts a blank line with a Page symbol and Carriage Return symbol. Causes the printer to begin a new page in both Editor and Formatter.

New Paragraph - CTRL 8 or CTRL M - WORD WRAP: Begins a new paragraph by inserting a Carriage Return symbol and a blank line and returning the cursor to the beginning of the blank line. If Automatic Paragraph Indentation is set, the cursor is returned to the indentation point. FIXED: Does not function.

Next Paragraph - CTRL 4 or CNTL J - Moves the cursor to the beginning of the following paragraph.

Next Window - FCTN 5 - Horizontal Block Scroll. Displays the next of the three overlapping windows of the full 80 column screen width - 1,2,3,1,2,etc.

OOPS! - CTRL 1 or CTRL Z - Recovers text deleted by Delete Character, Delete Line, Delete End of Line, Duplicate Line, typing over or blanking out. Removes characters typed on a blank screen.

Reformat - CTRL 2 or CTRL R - WORD WRAP: Recloses text after an insertion and fills text to incorporate deletions and insertions. Reformats stopes when a Carriage Return is encountered. FIXED: Terminates Insert Character.

Right Arrow - FCTN D or CTRL D - Moves the cursor to the right without blanking out text.

Roll Down - FCTN 4 or CTRL A - Vertical Block Scroll. Displays the 24 lines that follow the last line on the screen.

Roll Up - FCTN 6 or CTRL B - Vertical Block Scroll. Displays the 24 lines that precede the first line on the screen.

Screen Color - CTRL 3 - Displays the next screen background/character color combination in a sequence of five.

Tab - FCTN 7 or CTRL 1 - Moves the cursor to the next tab setting to the right.

Up Arrow - FCTN E or CTRL E - Moves the cursor up. If the cursor is on the top line, previous lines scroll onto the screen one at a time.

Word Tab - CTRL 7 or CTRL W - Moves the cursor to the first character of the next word.

Word Wrap - CTRL 0 (Zero) - Switches from Word Wrap Mode (solid cursor) to Fixed Mode (hollow cursor), and from Fixed Mode back to Word Wrap Mode.

TEXT EDITOR - COMMANDS IN COMMAND MODE

Edit: Exits Command Mode and returns to Edit Mode.

1. Type E for Edit
2. Press ENTER

FindString: Locates a word or phrase in the text buffer.

1. Type FS and press ENTER
2. Type /s/ where "s" is the word or phrase to be found.
3. Press ENTER

LoadF(Whole File): Loads a file from a disk into the text buffer.

1. Type LF for Load F.
2. Press ENTER
3. Type any valid filename.
4. Press ENTER

LoadF(Part of File): Loads part of a file from disk into the text buffer.

1. Type LF for Load F.
2. Press ENTER
3. Type the line number of the first line to be loaded, a space, the line number of the last line to be loaded, a space, and the filename of the file to be loaded.
4. Press ENTER

LoadF(Merge whole File): Merges a file on disk with the contents of the text buffer.

1. Type LF for Load F.
2. Press ENTER
3. Type the line number of the line in the text buffer after which the file is to be merged, a space, and the filename of the file to be merged.
4. Press ENTER

LoadF(Merge part of File): Merges part of a file on disk with the contents of the text buffer.

1. Type LF for Load F.
2. Press ENTER
3. Type the line number of the line in the text buffer after which the file is to be merged, a space, the line number of the first line of the file to be merged, a space, and the line number of the last line of the file to be merged.
4. Space once and type the filename of the file to be merged.
5. Press ENTER

Move: Moves a line or block of consecutive lines from one point to another point in the text buffer.

1. Type M for Move.
2. Press ENTER
3. Type the line number of the first line to be moved, a space, the line number of the last line to be moved, a space and the line number of the line after which the moved text is to be inserted.
4. Press ENTER

PrintF: Prints the contents of the text buffer.

1. Type PF for Print F.
2. Press ENTER
3. Type your printers device name.
4. Press ENTER

TO STOP/CANCEL PRINTING - press FCTN 4.

Purge: Clears the text buffer. The contents of the text buffer may be recovered by RecoverEdit.

1. Type P for Purge.
2. Press ENTER
3. Type Y for Yes or N for No.
4. Press ENTER

QUIT: Save a file, Purge a file or Exit the Text Editor.

1. Type Q for Quit.
2. Press ENTER
3. Type ONE of the following letters:

   S for SaveF(See SaveF)
   P for Purge(See Purge)
   E for Exit
4. Press ENTER

RecoverEdit: May recover all but the first line of the contents of the text buffer after purging.

1. Type RE for RecoverEdit.
2. Press ENTER
3. Type Y for Yes or N for No.
4. Press ENTER

ReplaceString: Replaces a word or words with another word or words in the text buffer.

1. Type RS for ReplaceString.
2. Press ENTER
3. Type a slash (/), the string to be replaced, a slash, the string that is to replace it, and a slash.
4. Press ENTER
5. Options:
A = All - replace the string in every subsequent instance.
Y = Yes - replace the string in this instance; find the next instance.
N = No - do not replace the string in this instance; find the next instance.
S = Stop - escape the command with the cursor on the last instance found.
6. When replacement is complete, return to Edit Mode is automatic.

SaveF(Whole File): Saves the contents of the text buffer including the Tabs settings to a file on disk.

1. Type SF for Save F.
2. Press ENTER
3. Type any valid filename.
4. Press ENTER

SaveF(Part of a File): Saves part of the contents of the text buffer to a file on disk. Tabs settings are saved if the part contains the last line of the text buffer contents.

1. Type SF for Save F.
2. Press ENTER
3. Type the line number of the first line of the part to save, a space, the line number of the last line of the part to save, a space, and any valid filename.
4. Press ENTER

Show: Locates a line in the text buffer by line number and displays it as the top line on the screen. Zero can equal line 0001, E can equal the last line of the file.

1. Type S for Show.
2. Press ENTER
3. Type the line number of the line to be shown
4. Press ENTER

ShowDirectory: Catalogs a disk on the screen.

1. Type SD for ShowDirectory.
2. Press ENTER
3. Type the number of the disk drive that contains the disk to be catalogued.
4. Press ENTER

Tabs: Sets margins, tabs and paragraph indentation for the text buffer.

1. Type T for Tabs.
2. Press ENTER
3. Beneath the appropriate column number, type:
L for Left Margin
I for paragraph Indent
T for Tab
R for Right Margin
4. Blank out any undesired settings
5. Press ENTER

TEXT FORMATTER - FORMAT COMMANDS

.AD Adjust - Justifies the right margin. Cannot be used without using the Fill Command. If Fill and Adjust are both used, and Fill is turned off with No Fill, Adjust is also turned off.

n Alternate Input - Used with the Mailing List option to position up to 99 variables that can be assigned values from the screen or from a value file called by a Mailing List command. (Mail Merge Option)

.BP Begin Page - Forces a page break. The printer begins subsequent text on a new page.

.CE n Center - Centers the next "n" lines. The command .CE centers the next line only.

.CO t Comment - Puts a comment "t" in text that is not printed with the document by the text formatter.

.DP n:t Define Prompt - Defines a prompt "t" for Alternate Input "n" to cue definition from the screen. (Mail Merge Option)

.FI Fill - Puts as many words on the line as fit without exceeding the right margin.

.FO t Footer - Puts "t" (text) as the footer on each page. If % is used in the header text, it is replaced with the appropriate consecutive page number.

Continued on P27

Continued from P26

.HE t Header - Puts "t" (text) as the header on each page. If % is used in the header text, it is replaced with the appropriate consecutive page number.

.IF f Include File - Calls filename "f" at that point. Does not permit nesting. Files can be called by disk drive number or by disk name.

.IN n Indent - Indent the first line of a paragraph "n" spaces. Absolute value indents to the column number "n" regardless of the left margin. Relative values: "+n" is added to the left margin value; "-n" is subtracted from the left margin value to "outdent" to the left of a left margin. Each time the Left Margin is reset, the indent command must be reset also.

.LM n Left Margin - Set the left margin at column "n". Absolute or relative values can be used.

.LS n Line Space - Causes the printer to skip "n" lines before printing each line. The default is single line spacing.

.ML f Mailing List - Calls value file "f" from the main file to assign values to variables defined by Alternate Input commands. (Mail Merge Option)

.NA No Adjust - Default. Turns of the Adjust command.

.NF No Fill - Default condition. Prints lines as they appear in the file. All of the Left Margin and Indent commands that follow a No Fill commard are ignored.

@ Overstrike - Causes the printer to overstrike subsequent characters until a space is encountered.

.PA n Page Number Reset - Resets the consecutive page number in Header and/or Footer commands to "n". Absolute or relative values may be used.

.PL n Page Length - Sets the number of lines per page to "n". The default is 66 lines per page.

.RM n Right Margin - Sets the right margin at column "n". Absolute or relative values may be used.

^ Required Space - Joins words for the purposes of filling, adjusting, underling and overstriking.

.SP n Space - Causes the printer to skip "n" lines beofre printing the next line. The command .SP skips one line.

& Underscore - Causes the printer to underscore subsequent characters until a space is encountered.

The following handy TI-WRITER commands are reprinted from the June issue of the 99'er News published by the TI Users Group of Will County, Romeoville, Ill. This puts the most used commands on one page for handy access at your computer.

EDITOR COMMAND	FCTN	CTRL	EDITOR COMMAND	FCTN	CRTL	EDITOR COMMAND	FCTN	CTRL
Back tab		T	Ins. Blank line	8	O	Quit	=	
Beginning/line		V	Insert character	2	G	Reformat		2orR
Command/escape	9	C	Last paragrapph		6orH	Right arrow	D	D
Delete character	1	F	Left arrow	S	S	Roll down	4	A
Del. end of line		K	Left margin rel.		Y	Roll up	6	B
Delete line	3	N	New page		9orP	Screen color		3
Line #'s(on/off)	0		New paragraph		8orM	Tab	7	I
Down arrow	X	A	Next paragraph		4OrJ	Up arrow	E	E
Duplicate line		5	Next window	5		Word tab		7orW
Home cursor		L	Oops!		1orZ	Word wrap/fixed		O

```
Load files: LF (enter) DSK1.FILENAME  (load entire file)
            LF (enter) 3 DSK1.FILENAME  (merges filename with data in memory
                                         after line 3)
            LF (enter) 3 1 10 DSK1.FILENAME  (lines 1 thru 10 of filename are
                                              merged after line 3 in memory)
            LF (enter) 1 10 DSK1.FILENAME  (loads lines 1 thru 10 of filename)
=================================================================================
Save files: SF (enter) DSK1.FILENAME  (save entire file)
            SF (enter) 1 10 DSK1.FILENAME (save lines 1 thru 10)
=================================================================================
Print Files:PF (enter) PIO (prints control characters and line numbers)
            PF (enter) C PIO (prints with no control characters)
            PF (enter) L PIO (prints 74 characters with line numbers)
            PF (enter) F PIO (prints fixed 80 format)
            PF (enter) 1 10 PIO (prints lines 1 thru 10)
NOTE: The above assumes PIO. DSK1.FILENAME, and RS232 are also valid!
      To cancel the print command press FCTN 4.
=================================================================================
Delete file:DF (enter) DSK1.FILENAME
=================================================================================
Setting Margins and Tabs: (16 tabs maximum)
      L - Left margin      R - Right margin     I - Indent       T - Tab
         Use ENTER to execute or COMMAND/ESCAPE to terminate command.
=================================================================================
Recover Edit: RE (enter) Y or N
=================================================================================
Line move:  M (enter) 2 6 10  (moves lines 2 thru 6 after line 10)
            M (enter) 2 2 10  (moves line 2 after line 10)
=================================================================================
Copy:       same as move except use C instead of M.
=================================================================================
Find String: FS (enter) /string/  (will look for string in entire file)
             FS (enter) 1 15 /string/  (will look for string in lines 2 thru 15)
=================================================================================
Delete:     D (enter) 10 15 (deletes lines 10 thru 15 in memory)
=================================================================================
```

THE MECHATRONIC EIGHTY COLUMN CARD

by BEN TAKACH

The long awaited hardware was cleared through customs on the eve of our August meeting. The card made its "dry" debut during the TISHUG meeting. Time just did not permit a full scale demonstration.
Even now, due to the lack of a suitable monitor, yours truly has to rely entirely on the information provided by Mechatronic -in German language- to review its features.

The 80 column card is the most recent addition to the range of peripherials developed for the TI-99/4A. It was recently released by the West German company Mechatronic GmbH.
It will produce 80 columns in 26 rows under program control. It also offers an enhanced graphic display. One can by suitable program produce 256x212 dots in 256 colours, or 516x212 dots in 16 colours. In addition it is compatible with most of the known programs and hardware (e.g. these will work as before on your TI-99/4A). However a few enhancements are also available with these. The 16 standard colours of the 99/4A may be chosen from 512 colour tones, as well as the position of the display on the VDU may be shifted within certain limits.
The card plugs into the expansion socket on the right hand side of the console. One has to start the installation by opening the console, in order to gain access to the video processor chip. This will be removed and replaced by a 40 way ribbon cable, which will also be plugged into the 80 col. card. Step by step installation instructions are supplied with the card. The card is also fitted with a centronics port for the direct connection of a parallel printer (RS232 card is not needed). Power for the card is taken from a 6v dc. plugpack, which plugs into a concentric socket on the rear of the card. This must be switched on before the console is powered up.
The monitor connector is on the side of the card. It is a 9 pin sub D type socket (identical to the joystick port or the tape recorder interface).
The monitor output is an RGB analogue signal with TTL synchronisation. Thus one needs an RGB monitor. Mechatronic offer a modulator as an optional extra, which has a composite video output and also an HF signal for TV use. The use of a TV set is not recommended. The 80 col. display on the TV set is not very satisfactory.
Audio signal is taken -as before- from the appropriate pin of the DIN socket from the rear of the console.

The operation of the computer for all existing programs remain the same as before. The monitor attached to the 80 col. card is used for all display purposes. The usual title screen appears after powerup, and you may proceed as usual.
Some consoles and a few modules are exceptions. These will be detailed in the following paragraphs.

The card is fitted with 3 sets of DIP switches. Switch-bank 1 consists of 8 switches, these are used to centre the display on the VDU.
Switch-bank 2 is used to select the PAL resp. NTSC op. mode, as well as selection of US or GERMAN character sets. A switch is also provided to enable or disable the push-buttons (more of these in the next paragraph).
Switch-bank 3 sets the DSR address. The switches must be set for CRU base address 1000 to guarantee trouble free operation (1000 is the most popular address lately! The MYARC Ramdisk is also set at 1000, this can not be changed as it resides in a Pal chip.). Other addresses are not useable (One must question the wisdom of providing DIP switches to select the one and only permissible address).

There are 2 push buttons on the PC board. These can be accessed through the steel cover plate by a ball point pen or a pencil. Mechatronic was forced to incorporate these control units as TI has used in some of the consoles and modules "false" VDU control codes.

In practice, if the display is blank or the screen is filled with colourfull garbage then the left hand side button is pressed to correct the situation. In such case one also has to push the QUIT key when the console is to be reset. The push button enable/disable dip switch is turned off with standard consoles to avoid malfunction and possible lockup due to inadvertent operation of the push)button. The second UM! button so far has no function (Perhaps it could be wired for interrupt routine?).

A resident EPROM contains the operating system and software. Its basic operation mode is the same as that of the console based video processor. The only difference is that the available VDP register space is 2 bytes shorter. It may result in the inability to load special copy protected programs. However these are very rear since the introduction of the various third party DISK CONTROLLERS, as these are also not compatible.

HOW DOES IT WORK.
Text Mode, General, OPEN.
One invokes the 80 col. text mode from console BASIC or XB exactly as if it would be a printer. The device name is "TEXT 80". Only DISPLAY 80 mode is allowed. The open statement also contains an up to 8 bytes long (16 Hex digits) VDP RAM resp. VDP Register definer. This may be omitted in XB mode (XB mode Definer is the default). Optionally, the cursor starting address may also included in the definer. E.g. "TEXT80.0000E00020000000" is the definer for XB and cursor start address at the top left hand corner of the screen.
The 80 col. mode is established by an open statement, and a close command/statement will restore the standard TI format.

Control Character Codes.
Special screen functions are implemented by the use of control characters, just the same as it is with printers. These may be embedded in the text. For example CHR$(17) will kill the display. Others are designated for the following screen functions:

-cursor positioned at the upper left-hand corner,
-highlighting display segments by blinking,
-to stop all blinking (works on the entire screen)
-to position the cursor anywhere on the screen,
-define the size of an input value or string,
(just like ACCEPT AT SIZE ... statement in XB)
-colour definition resp. selection,
-define the blink frequency
-CHAR definition,
-Input prompt default accepted as input.

The resident software also permits the redefinition of colours (e.g. inverse display mode), by the "DEFCOL" open mode. A file may be opened in DISPLAY VARIABLE 80 format by the statement OPEN #1:"DEFCOL", to print selected data in the redefined colour.

A supplementary disk is also supplied with the card. This software enables the use of the TI-WRITER module in the 80 col. format.

The card comes in a solid steel enclosure, it is fitted with 20 IC-s, a 21 MHz crystal, several transistors in addition to the dip switches and push buttons as well as the usual passive components. The heart of the unit is the Yamaha V9938 Video Processor Chip. The components mount a high quality double sided glass fibre PC board.

Present cost of the card ex Sydney (not including P.&P. to other Australian destinations) is $385.50
The Mechatronic 80 col. Card may be obtained from:

Ben von Takach, P.O.Box 114 Wahroonga NSW, 2076,
Tel: (02) 489 4492. Telex 10718194 (tuet) O

Comments on Game Loader Programme

by Geoff Trott

In The Clubline Magazine of February 1987 (Vol. 5, No. 5), there was a plea by Tor Hansen for help in understanding an Extended BASIC programme submitted by Pierre Drouin. The object of the programme is to load assembly language programmes into expansion memory from a menu.

The strange thing about the programme is that the first 11 statements are all REM statements, with the last 10 of these, containing numbers and other special characters. After the REM statements the main body of the programme follows, and this includes a small number of CALL LOADs after the particular game is selected from a menu. The filenames for the games all start with "X". The programme does not appear to finish, but just enters a continuous loop.

Analysing the BASIC programme CALL LOADs first, the first set store bytes starting at 12288 (>3000) and continuing to 12353 (>3041), or a total of 66 bytes. Then the filename is stored starting at 13000 (>32C8), without its disk name. Finally the value 48 (>30) is stored at -31804 (>83C4), which is the high byte of the address of the user defined interrupt routine (both bytes are normally 0 unless an interrupt routine is already in use). This means that at the next interrupt (each 16.7 milliseconds at 60 Hz, or 20 milliseconds at 50 Hz) the operating system will start executing the routine at address >3000 (12288), which is the programme just stored with the CALL LOADs.

These 66 bytes must be a machine language programme, which does something with the data in the REM statements, so if we look at this programme it should be possible to find out what is going on.

Before doing this, if the data in the REM statements is examined carefully, it can be seen that there are only 16 different symbols used, namely "0123456789:;<=>?". This would imply that these symbols are used to represent the Hexadecimal code, and this supposition is further strengthened by the fact that these 16 symbols are consecutive members of the ASCII code, namely "0"=>30 up to "?"=>3F. This means that the hexadecimal value for each symbol can be extracted by using the low 4 bits of its ASCII code. This is what the first programme does.

The first programme converts data stored as a series of REM statements into binary code to be then used as a machine language programme by the processor.

The programme must take the data one character at a time, obtain its hexadecimal value, put that digit together with a second hexadecimal digit from the next character to make a byte, and store it in the correct location. The first programme is stored in low expansion memory (>3000 to >3041), and it stores the second programme in high expansion memory (>A000 to >A18C).

The first programme is put into memory using CALL LOADs by the Extended BASIC programme, which also has the REM statements in it. The REM statements must start at the second line of the programme and each contain 80 characters for the 80 Hexadecimal digits of the 40 bytes to which they refer.

Now follows the first programme in assembler mnemonic form, with the machine code down the left margin in its CALL LOAD format.

```
                  AORG 12288     >3000 in decimal

          START  EQU  >A000  start of next programme
          LTADR  EQU  >8332  line table address
          INTADR EQU >83C4 interrupt routine address
          MASK   EQU  >0F00     Hexadecimal bit mask

          FIRST
2,0,160,0        LI   R0,START  save address
2,1,0,10         LI   R1,10     10 REM statements
192,160,131,50   MOV  @LTADR,R2 look for first
6,2              DEC  R2        adjust address
          LOOP1
6,66             DECT R2        next line number
6,66             DECT R2
208,226,0,1      MOVB @1(R2),R3 get place in memory
6,195            SWPB R3
208,210          MOVB *R2,R3
5,195            INCT R3        skip REM token + 1
2,4,0,40         LI   R4,40     40 bytes per REM
          LOOP2
209,115          MOVB *R3+,R5   get first character
2,69,15,0        ANDI R5,MASK   Hex value
10,69            SLA  R5,4      shift to ms digit
209,179          MOVB *R3+,R6   get second character
2,70,15,0        ANDI R6,MASK   Hex value
241,133          SOCB R5,R6     OR them together
220,6            MOVB R6,*R0+   save byte
6,4              DEC  R4        finished this REM?
22,245           JNE  LOOP2     jump if not
6,1              DEC  R1        finished all REMs?
22,234           JNE  LOOP1     jump if not
4,224,131,196    CLR. @INTADR   turn off programme
     which was started by putting its start address
      in the interrupt service routine address.
4,96,160,0       B    @START    go execute next one.
```

The second programme is the one which was in the REM statements. It is thus stored at location >A000, and it reads in the file from the disk and starts it executing. It has to avoid being overwritten in the process, and uses a flag at the start of the filename to decide if the colour table and pattern table need to be moved in VDP RAM. If the filename starts with "X", then it uses the same VDP registers as Extended BASIC and no changes are needed. Any other character at the start of the filename and E/A environment is assumed. Then the pattern table and colour table are moved in VDP memory and the VDP registers 3 and 4 are changed. The filename is set up by the Extended BASIC programme and stored in low expansion memory.

The code is given below in assembler mnemonic form as before, while the machine code is given in hexadecimal using the symbols used in the REM statements.

```
                        AORG    >A000
              FNAME    EQU  >32C8  The Ex-BASIC programme
                                   puts filename here
              XBFLG    EQU  >A198  Flag to change VDP regs
              VPAB     EQU  >0A00  VDP address for PAB
              STATUS   EQU  >837C  system status byte
              PNAME    EQU  >8356  address of disk name
              PNUMB    EQU  >8354  number of bytes in name
              DSRSV    EQU  >83D0  save the CRU address
              REG3     EQU  >83E6  address of WS register 3
              PAD      EQU  >8300  Start of PAD area
              RESET    EQU  0      reset on error address
              DSRSUB   EQU  >4008  address DSR name table
              DSKDSR   EQU  >1100  disk CRU address
              C200H    EQU  >200   a constant = 512

              START
06:0 :15>          BL   @VMBR       read disk number
3>>; :18;          DATA >3EEB,NUMBER  from VDP
              ONE  EQU  $+1
0001               DATA 1           used as a constant
?820 :17= :18;     SOCB @ASC0,@NUMBER  OR ascii code
0200 32<8          LI   R0,FNAME     address of filename
=830 :198          MOVB *R0+,@XBFLG save first byte flag
0581               INC  R1           after . in disk name
              L1
9810 :0=7          CB   *R0,@SPACE until " " found.
1302               JEQ  ST1       copy filename to end of
=<70               MOVB *R0+,*R1+    diskname in PAB area
10?;               JMP  L1
              ST1
0601               DEC  R1          adjust address to last
<801 :0<0          MOV  R1,@LNAME character and store it
0221 5>79          AI   R1,>5E79    subtract addr length
<801 :186          MOV  R1,@LENG    store name length
              L2
06:0 :142          BL   @VMBW       move the PAB into VDP
0:00 :17> 001:     DATA VPAB,PAB,26
04>0 837<          CLR  @STATUS     clear status
0200 0:0>          LI   R0,VPAB+14 address of name
<800 8356          MOV  R0,@PNAME    saved
0200 0004          LI   R0,4        # bytes in disk name
<800 8354          MOV  R0,@PNUMB   saved
020< 1100          LI   R12,DSKDSR disk CRU address
<80< 83=0          MOV  R12,@DSRSV    saved
1=00               SBO  0           enable disk CRU
<0>0 4008          MOV  @DSRSUB,R3 DSR sub. table addr.
              L3
<083               MOV  R3,R2
136>               JEQ  ERROR       if zero big problem!
<0?2               MOV  *R2+,R3     link to next name
<272               MOV  *R2+,R9     address of routine
9<:0 8355          CB   *R2+,@PNUMB+1 check for 4 bytes
16?9               JNE  L3          no, try next one
<140               MOV  R0,R5       set up counter
0206 :188          LI   R6,DNAME    get disk name addr
```

```
        L4
9<;6              CB    R6+,*R2+   check for same name
16?4              JNE   L3         no, next one
0605              DEC   R5         all characters
16?<              JNE   L4
0699              BL    *R9        found name-execute
1060              JMP   ERROR      error return
1>00              SBZ   0          turn off disk DSR
<020 :180         MOV   @VBUFF,R0  get VDP buffer
06:0 :160         BL    @VMBR1     read first 6 bytes
83>6 0006         DATA  REG3,6      into regs R3,R4,R5
<1:0 :138         MOV   @BEGIN,R6  Check if start addr
1602              JNE   ST2         has been saved.
<805 :138         MOV   R5,@BEGIN  if not, save it!
        ST2
<084              MOV   R4,R2      number of bytes
<045              MOV   R5,R1      init'l store address
0220 0006        ·AI    R0,6       add 6 to VDP address
0222 ???:         AI    R2,-6      take 6 from count
0285 :000         CI    R5,START   check overwrite this
1608              JNE   ST3         programme
0206 0200         LI    R6,C200H   yes, save 512 bytes
:006              A     R6,R0       in VDP buffer, and
<800 :180         MOV   R0,@VBUFF   adjust the current
6086              S     R6,R2       addresses and count
:046              A     R6,R1       to allow for this.
        ST3
1000              NOP
06:0 :164         BL    @VMBR2     Xfer VDP buffer
        LNAME     EQU   $+4        last char in name
;820 :009 0000    AB    @ONE,@0    inc last byte of name
<0<3              MOV   R3,R3      see if last file read
16;6              JNE   L2         jump if not
9820 :198 :17<    CB    @XBFLG,@ASCX check BASIC flag
1325              JEQ   ST4        jump if BASIC
06:0 :15>         BL    @VMBR    else shift colour table
0800 :000 VADDRE DATA >800,START
0020      SPACE DATA >20
06:0 :142         BL    @VMBW      to >380
0380 :00< 0020    DATA >380,START+12,32
        L5
06:0 :15>         BL    @VMBR      pattern table from
0400:0000020 VADDRS DATA >400,START,32     >400
06:0 :142         BL    @VMBW       to >900, 32 bytes
0900 :000 VADDRD DATA >900,START  at a time
0020      K32     DATA 32
:820 :0?4 :0>6    A     @K32,@VADDRS
:820 :0?4 :0?0    A     @K32,@VADDRD
8820 :0>6 :0=2    C     @VADDRS,@VADDRE
1:><              JL    L5
0201 :178         LI    R1,VDPWR   change VDP write
0202 0004         LI    R2,4       registers
        L6
=7?1              MOVB *R1+,*R15   registers 3 and 4
0602              DEC   R2
16?=              JNE   L6
.       ST4
0200 0014         LI    R0,20      move that first 512
0201 8300         LI    R1,PAD     bytes from VDP buffer
0202 :164         LI    R2,VMBR2   to >A000
        L7                         First move VMBR subr
<<72              MOV   *R2+,*R1+ into PAD memory, then
0640              DECT R0          use it to do move,
16?=              JNE   L7         make its return
0200 0=06         LI    R0,>D06    addr start addr of
0201 :000         LI    R1,START   prog just loaded, it
0202 0200         LI    R2,C200H   will self start.
        BEGIN     EQU   $+2
020; 0000         LI    R11,0
0460 8300         B     @PAD
        ERROR
0420 0000         BLWP @RESET      On error do reset
        VMBW                       Subroutine to write
<03;              MOV   *R11+,R0    many bytes to VDP
<07;              MOV   *R11+,R1    memory. Arguments
0260 4000         ORI   R0,>4000    follow the call in
06<0              SWPB R0           order; VDP, memory,
=7<0              MOVB R0,*R15      count.
06<0              SWPB R0
=7<0              MOVB R0,*R15     first the write addr
<0;;              MOV   *R11+,R2
        VL1
=831 8<00         MOVB *R1+,@>8C00 then do the transfer
0602              DEC   R2
16?<              JNE   VL1
045;              RT
        VMBR                       Subroutine to read
<03;              MOV   *R11+,R0    many bytes from VDP
        VMBR1                       memory. Arguments
<07;              MOV   *R11+,R1    follow the call in
<0;;              MOV   *R11+,R2    order; VDP, memory
                                    count.
```

```
        VMBR2
06<0              SWPB R0
=7<0              MOVB R0,*R15     first the read addr
06<0              SWPB R0
=7<0              MOVB R0,*R15
1000              NOP
        VL2
=<60 8800         MOVB @>8800,*R1+ then do the move
0602              DEC   R2
16?<              JNE   VL2
045;              RT

        VDPWR                      **** data ****
0>83              BYTE >0E,>83     register 3 contents
0184              BYTE 1,>84       register 4 contents
        ASCX
58                TEXT "X"         for checking flag
        ASC0
30                TEXT "0"         to make number ascii
        PAB
·050<             BYTE 5,>0C       PAB load command
        VBUFF
0=00              DATA >D00        VDP buffer address
0000 2400         DATA 0,>2400     memory image file
        LENG
00                BYTE 0
        DNLEN
00                BYTE 0  length of whole filename
        DNAME
44534;302>        TEXT "DSK0." diskname for filename
        NUMBER    EQU  $-2         disk number address
        NAME      EQU  $       start address of filename
```

If you compare the data here with the original version you will find two changes. The first change is that a ";" has been added to statement 1090 (second last line), and the second is an "=" has been changed to ">" in statement 1030 (second line). The first of these looks like a typing error although the programme will not work as given because of it, while the second is an error which will not cause a problem, as it only occurs on an error and then it will get to the correct place eventually anyway!

I hope these listings help provide an understanding what this programme tries to do. There are probably other ways of doing the same thing, and the idea of using REM statements to enter the machine language programme seems to be a good one. The symbols used here are not the only ones which could be used. Any 16 symbols which occupy consecutive ASCII codes with their least significant hexadecimal digit could be used. For example, "0123456789JKLMNO" or "PABSTEFGXYJKLMNO" could be used to confuse the casual reader. O

* * * * * * *

Continued from P2

Once again let me stress that the continued success of the TI Computer and compatibles will only be through the support of the users. Your support is vital. We can only guess, but in excess of 20,000 TI-99/4A computers must have been sold in Australia. Less than 750 users are members of Users Groups. There is an awful lot of people who are interested in the computer but are not members, and they must be interested because TI-99/4A computers are not being sold. These people probably know nothing of the enormous leaps that have been made in the field. Promote this faire in your own area and the membership of the Users Groups must increase.

The work that all users put in today will benefit them in the future. I ask that the Sydney Harbour Users Group support this faire. Perhaps you would consider publishing this letter in your newsletter. The invitation is open to all your members as they are the ones who will benefit.

The invitation remains open to all who may be visiting the EXPO88 later in the year. Drop in and say "Hello". Interstate visitors are always welcome.

I wish the Sydney Harbour Users Group success in the future and hope to see many members at the TI-Faire next year.

Yours faithfully,

(Garry J Christensen) o

REGIONAL GROUP REPORTS

BANANA COAST REGIONAL GROUP

Coffs Harbour is a resort town, about 570 km, north of Sydney and has a population of about 16,700. It is in the heart of the banana country and hence the name "Banana Coast Regional Group"

The group has had a steady growth since its formation about November last year. Expatriate, Keir Wells, migrated to the banana republic to start the group. There was comment when Keir left that the IQ of Sydney would go up. As would the Coffs Harbour area.

A number of members have installed, on board, 32K memory expansion. Support from other regional groups, as well as head office, with software has been appreciated.

Distance from Sydney always makes it difficult for members to attend TIsHUG monthly meetings. For the same reasons, telephone costs are also high when using the BBS. As a regional group of TIsHUG they receive the full support of the club.

They deserve our support.

Well done, Banana Coasters.

* * * * * * *

BANANA COAST RG REPORT:

The July meeting of the Banana Coast Regional Group was held on 12 July with 10 members attending. The highlight of the day was the attempted demonstration of the Mini Expansion System. After getting the title screen to appear it was found that the SDSS disk drive wouldn't load the software. After consultation with Peter Schubert this problem was solved. Three members also brought along printers to see if they were compatable with the new PE System, but without proper cables little was achieved in this area. With half the members with a full system the future looks good for the TI. A hearty afternoon tea was enjoyed by all at the the conclusion of the meeting. Next meeting will be held at Kerry Harrison's home.

* * * * * * *

LIVERPOOL REGIONAL GROUP

The meeting at Stans place was a quite evening with only a few members turning up.

Robert Pevrill demontrated a device he built that will help you diagnose Disk Drive problems, something we will surely use when things go FZZZ!@?.... Thanks Robert for showing that at least some members can come up with something when most of us just take not give.

Arto gave a small demo on a routine that will be available to Picasso owners when he is finished. The routine can use the files from CSGD to create large FONTS that can be either Merged or loaded straight into PICASSO PUBLISHER.

NEXT MEETING: ROSS HARDY
TIME & DATE : 11-9-87 7.30 pm
PHONE ROSS : (02) 637 6772

GLEBE Regional Group.

10th September 1987, 8pm, 43 Boyce St, Glebe. Contact Mike Slattery, 692 0559. Regular meetings on the Thursday evening following the first Saturday of the month.

CENTRAL COAST Regional Group.

Meetings are normally held on Second Saturday of each month at 6.30 pm at
Toukley Tennis Club hall,
Header St, Toukley.
Next meeting 12th September 1987.

Contact Russell Welham (043 92 4000)

ILLAWARRA Regional Group.

Next meeting 17/8/87 7.30pm, Keiraville Public School, Gipps Rd, Keiraville.
Opposite Keiraville Shopping centre.

Regular meetings are third Monday of each month except January.

CARLINGFORD Regional Group.

The next Carlingford Regional Group meeting will be 19th August 1987.

Commencing Time 7.30 pm at the home of

Claude Fra,
10 MacKenzie Bvde,
Seven Hills. phone 622 9423

Ample parking on nature strip opposite house.
For more details contact Chris Buttner Ph 871 7753 or mail to CO-ORD .

Regular meetings are third wednesday of each month.

NORTHERN SUBURBS Regional Group.

Contact Dennis Norman on 452 3920 or Dick Warburton on 918 8132 for further information.

20/8/87 8pm, at Dick Warburton's home,
7 Milga Rd, Avalon Beach.
ring Dick for more details.

Regular meetings are third or fourth Thursday of the month.

BANKSTOWN Regional Group.

Next meeting date unknown to SYSOP but if interested contact Peter Pedersen, (02) 772 2396. Meeting are held at 11 Bastille Close, Padstow Heights.