042
8704

TISHUG NEWS DIGEST

Focusing on the TI-99/4A Home Computer

Volume 6, Number 3 **April 1987**

Registered by Australia Post - Publication No. NBH5933

TEXAS INSTRUMENTS

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

TISHUG

TISHUG NEWS DIGEST

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

MARCH 1987

Correspondence to:

PO Box 214
REDFERN NSW 2016

Texpac BBS: Tel.: (02)319.1009

COMMITTEE MEMBERS:

Co-Ordinator:
Chris Buttner..Tel.(02)8717753
Secretary:
Terry Phillips.Tel.(02)7976313
Treasurer:
Bert Thomas....Tel.(047)541535
Publications:
Bob Montgomery.Tel.(042)286463
Sysop:
Ross Mudie.....Tel.(02)4562122
Merchandising:
Cyril Bohlsen..Tel.(02)6395847
Technical:
John Paine.....Tel.(02)6256318
Librarian:
Terry Phillips.Tel.(02)7976313

REGIONAL COMMITTEE MEMBERS:

Glebe:
Mike Slattery..Tel.(02)6920559
Penrith:
John Paine.....Tel.(02)6256318
Central Coast:
Russell Welham.Tel.(043)924000
Liverpool:
Stan Puckle....Tel.(046)256157
Illawarra:
Rolf Schreiber.Tel.(042)842980
Bankstown:
Peter Pederson.Tel.(02)7722396
Carlingford:
Chris Buttner..Tel.(02)8717753
Sutherland:
Peter Young....Tel.(02)5288775
Manly Warringah:
Dennis Norman..Tel.(02)4523920
Coffs Harbour:
Keir Wells.....Tel.(066)551487

MEMBERSHIP AND SUBSCRIPTIONS:

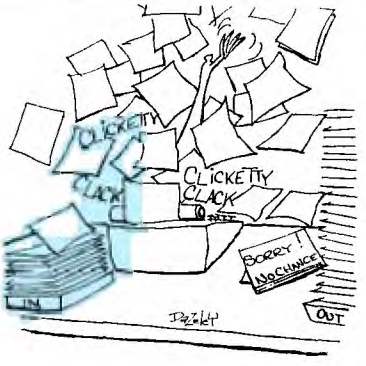
Joining Fee.....$ 8.00
Annual Family Dues.....$25.00
Dues O'seas Airmail...US$30.00
Publications Library....$ 5.00
Texpac BBS.....$ 5.00
BBS Membership:
Other TI User Group
Members.....$10.00
Public Access.....$25.00

GROUP GENERAL MEETING:

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

COMMITTEE MEETINGS:

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



CONTENTS Page

* * *

This month's magazine has three articles relating to how every member can submit articles.

Many of the aricles have been downloaded from the TEXPAC BBS, while others have been passed on by means of disk files.

It has been very gratifying to have had so many local articles submitted this month. Unfortunately not all have made this issue. Special mention must be given to Ben Takach and Arno Heino for their efforts. Ross Mudie was very patient as I learnt how to use the BBS . I had managed to wipe the mail files of articles before they had been properly saved.

The April meeting will see John Paine give a presentation on the RAMDISK.

Space limitations has not allowed everything to be included in this month's magazine.

Please remember:
Fees are due in April. If you haven't already paid, then get it in.

CO-ORDINATOR'S REPORT

with CHRIS BUTTNER

This issue marks the start of what I hope will become a regular and useful feature in the magazine. If you have any queries relating to either hardware or software, address them to FORUM. Your problems are likely also problems to others. Where appropriate, answers will be given through this column and you will be helping others to understand their computers and software.

****

Last month we had an article in the magazine mentioning an IBM upgrade for the TI 99/4A. That is now a reality as is the Myarc Geneve Card. Computer manufacturers, tend to be a bit like car manufacturers - there is a lot of "hpye" at release time, glowing reports, fanfares and many claims about why we need the product. As users, we should be like car buyers: Cautious.

****

A regular correspondent has made some observations which I believe are worth mentioning to all TI users.

Firstly, let's deal with the Triton product. It sells for US$449, comes without a keyboard, and is still going to take up extra space on your table. On the other hand you could buy a clone with 640K and keyboard for US$469. The difference in price is less that the cost of a RAVE keyboard which you would probably buy to use with the Triton computer.

Secondly, we have the Myarc Geneve card. As with a car, the after sales backup and support is crucial. Our computer is now generally accepted as an "orphan" although TI still provides limited support. By all accounts the production version of the card is a joy to use with the IBM type keyboard and a real screamer where speed is concerned when compared with our pedestrian TI but where do I get the support for a new card and at what cost? I don't know of any short answer to the questions, but at least if I do buy a unit I will be doing so with both eyes open and no rose tinted glasses.

****

All this deep thinking brings me back to the question - Why did I buy the computer in the first place?. I thought it would be good for the children, I knew I was going to have fun with it and it might just be handy to do some spreadsheets or word-processing if I took the plunge and expanded the system. All that is now history and I am pleased to say I have not been disappointed. I still use my unit regularly (in fact more than I first imagined) the young children use it; my young adults use it and will probably kick me off it in time so as to get their college work done, and the marvelous little machine keeps ticking along while others I know have had more than one other computer suffer from that human disease called stress.

****

In 1987 I wouldn't buy a TI 99/4A to help run a business but for home use, its on a short list of one.

****

Last month I gave you a taste of the administrative problems we have to tackle this year. I am pleased to tell you I have had productive talks with a solicitor regarding our incorporation and it is back on the rails again. I will keep you up to date with progress there.

Our technical group has put in a marvellous effort manufacturing the 32K console expansion boards. A lot of club money has been invested in hardware and we must work to budget. The memory chip market is a volatile one with wide price fluctuations which can substantially effect the final cost of any project. Because of this, we are forced to buy at short notice (if our selling prices are to be realistic) and naturally its a case of "first in best dressed", so if you are thinking about one of our club projects, don't ponder too long - you may well miss out. If you are genuine about wanting items, we want to help you but at the same time we need you support - financial and moral.

****

The Source Information Network (Virginia, USA) is considering a proposal from the club to allow members to subscribe to The Source without having to pay the normal joining fee of US$49.95. If they agree, the offer will be open for the month of APRIL only.

Our proposal also includes the opportunity to purchase the Sourcepak Manual for US$9.95 + shipping (currently US$15.00 but we're working on that too.)

Most of you will recall the small brown folders packed in with the console offering the services of TEXNET. That service no longer exists having been replaced by TISIG which offers XModem downloading, Bulletin Board, Members' Directory as well as the other standard Source features - US Stock market information, access to the Official Airlines Guide, Crolier's Academic American Encyclopedia on line, communications, and a tutorial section where you are not charged access fees by The Source.

I mention this now so you can consider ALL the implications of joining at your leisure rather than in haste in April.

There are several financial matters you need to bear in mind. All Source charges are in US Dollars and you will require a credit card such as Mastercard or Visa to which your monthly charges can be billed. The MINIMUM monthly charge is $10.00. On-line charges vary depending on baud rate and time of access. The minimum on-line charge is 10 cents/minute and the maximum is 46 cents/minute. If you want specific rates, see Ross MUDIE or myself.

You will also incur charges in accessing The Source from Australia. The dearest way is simply to dial direct. The alternative, cheaper way, is to use Austpac or Midas/Minerva. Austpac has a $50 joining fee whereas Midas/Minerva doesn't but the hourly charges are higher. With regular use spread over a 12 month period, Austpac becomes cheaper if you are on-line for more than 1 hour/month.

Well, that's it. I expect the club proposal will be accepted, so you need to now decide if you want access and secondly, your estimated monthly use so you can get the lowest charges. Remember, sign up time is APRIL if you want to participate.

****

Want to help the club? Get your renewal in early. They all expire in April but we can serve you better if they don't all come in at the last minute. After all, its the cheapest "insurance policy" for your computer.

o

* Secretary's Notebook *

By Terry Phillips

Welcome to the following new members:

Dennis Remmer - Mermaid Waters QLD
Tony Smith - Kelso NSW
Peter Hogan - Bray Park QLD

Several enquiries have been received and having responded I hope we have at least another 3 to 4 new members shortly.

NEW ADDRESS - FOR ALL CORRESPONDENCE

Would you all please amend your records to show that the new address for TIsHUG is PO BOX 214 REDFERN NSW 2016. This box is near where I work so I will be able to clear the box daily and give quicker attention to your needs.

At the Committee meeting held in February it was decided to dispense with the Programmers Crisis Line which has been run by Graeme Holliss for the past few years. The main reasons for this decision were that it was felt members with programming problems have 3 main options available to them by which to solve those problems. These are:

1. Look up examples given in your reference manuals. It is surprising the amount of detailed information you will find.

2. Consult your local regional group. It is through this type of contact that learning takes place.

3. If you don't live near a regional group, or can't attend a group meeting, then write to TIsHUG outlining your problem. It is through this means that perhaps your query can be published in this news digest and give others the benefit of a response by one of our programming experts.

A letter has been forwarded to Graeme expressing appreciation for his efforts.

I'm sorry that I don't have much else to report in this column at this time. Hopefully the column will grow so bear with me.

PUBLICATIONS REPORT - APRIL 1987

By Terry Phillips

NEW MAGAZINES IN THE TIsHUG PUBLICATIONS LIBRARY.

Publib members can contact Russell Welham on 043.92400 for availability or see him at the meetings.

NORTHWEST OHIO 99'ER NEWS	- November 1986
HOUSTON USERS GROUP NEWS	- February 1987
MELBOURNE TIMES	- January 1987
HUNTER VALLEY 99'ERS NEWS	- February 1987
99'ER ON LINE	- February 1987
OTTAWA UG NEWSLETTER	- January 1987
KANKAKEE UG NEWSLETTER	- December 1986

The publications library contains many overseas newsletters and other books on the TI. If you haven't yet joined why not do it now and take advantage of this wealth of information that is available for the TI.

TIsHUG Tutorial Days, June & November 1987.

Please tell your committee what you want.

The committee of TIsHUG is planning Tutorial workshop days for Saturday 6th June and 7th November 1987. It would be of great assistance to the committee if interested members would advise what subjects are wanted and how many people are likely to attend.

Here are some suggestions for the tutorial days:

1. Beginners Basic. A full day course. Bring your console, power supply, modulator, TV & cassette recorder.

2. Extended Basic. A full day course. This class requires a knowledge of TI Basic and will concentrate on the "Extended" commands & statements. Bring your own extended basic manual for reference.

3. Beginners TI LOGO. A full day course. Bring your own 32K console or expansion system, TV, TI LOGO module and books.

4. Beginners FORTH. A full day course. Bring your own console, disk expansion system,TV, E/A module, FORTH disk & spare disk.

5. Beginners Assembly. A full day course. This is a theory course for people who are already familar with the E/A Editor. Bring your own E/A manual for reference purposes.

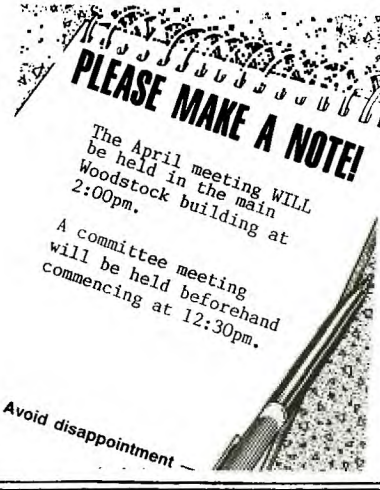
6. File handling in assembly linked from Basic or Extended Basic.
This is a full day advanced level class which will use the assembly source files from the TEXPAC BBS as the text. You should bring your own E/A manual.

Participants in all classes will need to bring their own note books and pens.

The above classes are suggestions only. If you are interested in attending then please advise the committee by writing to TIsHUG at PO Box 214, Redfern 2016, or by leaving mail on the BBS to SYSOP .

Please remember that without your response the committee will be unable to plan classes which meet the needs of the membership and there may not be sufficient copies of the printed notes.

People willing to plan and present classes are also required, please advise if you are able and willing to assist.



Producing the TND

Writing Articles for the News Digest

This is the start of a new year and a new Editorial staff for the News Digest. Thanks to Shane we have enjoyed a first class publication for the last 6 years, but now we must do without him and go in another direction. As a start, we would like to have a greater proportion of articles written by members of TIsHUG, which means YOU. Anyone with a TI99/4A can write an article and send it to us, or bring it to a monthly meeting, or give it to someone to bring to the monthly meeting. There are many ways of doing this and here are some of them.

1. On disk, in DIS/VAR 80 format. These can be prepared using TI-Writer, E/A editor or Console Writer.

2. On cassette using Console Writer, or as REM statements in a BASIC or XB programme.

3. On a piece of paper, either printed or hand written.

The first two of these are preferred and will ensure quick publication of the material supplied. Also the club will swap your disk or cassette full of articles with a blank disk or cassette if the article is handed in at a monthly meeting. Of course if you would prefer to get your own disk back, this can also be done. Regular contributors will be supplied with a disk or cassette in advance.

If you have a console and only BASIC or Extended BASIC with cassette, do not dispair. BASIC contains a very good editor so that you can write articles as a series of comments starting with REM or !. The main difficulty with this method is the fact that the lines have a maximum length which may not appear to suit your needs. Do not worry about this as by using a few special characters you can indicate clearly how you want the final result to appear.

The first problem is how to indicate the end of a paragraph, or force a new line. This is done by using the } character to indicate this. If you want to leave a blank line, put in two of them (}}), either at the end of a BASIC line, or one at the end of a BASIC line, and the next on its own BASIC line.

The second problem is what to do when the beep comes to indicate the end of a BASIC line and you are in the middle of a word. Either delete the start of the word and terminate that line beginning the next line with that word, or simply continue the word on the next BASIC line by making the first character on that line a "-". This will not appear when the listing is processed. Your article is then saved as a programme, which is far faster for cassettes than any other way. Of course you can load and edit your article, just as you can do that with a programme.

In general when writing articles it is better to put in too many space characters, rather than too few. There should be one space after every punctuation character, and two after every period at the end of a sentence.

Everyone has the tools for producing articles one of these ways, so there are no excuses for not trying. Tell all of us what you are doing with your computer, what you think of the club, what could be done to improve the club, and so on. Do not let the ones brave enough to stand up and talk dominate everyone. You can work your thoughts out in the privacy of your own home and send them to us. It is a good way to learn about word processing and a good use for your computer. o

THE MECHANICAL DETAILS OF YOUR TND MAGAZINE.

Contributions are typed or printed if these are submitted as a disk file. The finished printed pages are cut up and pasted on A3 format sheets in a 2 column format. These masters are then photographically reduced to A4 size and get printed in the conventional manner. It saves a lot of work if contributions are typed or disk files are formatted appropriately .

Contributions should be typed with margins set at 55 characters per line. Disk files should also be set out at 55 characters per line. Any text submitted for direct reprint should be typed at 12 characters per inch spacing. Text submitted for direct paste up prepared on a machine which will print only at 10 char./inch spacing should be typed at 46 characters per line. Any text not prepared in the above format has to be retyped by the editors. Disk files are preferred for the sake of uniformity. Note: The club uses a daisy wheel printer, its copy is preferred to a dot matrix printer.

Two columns of max. 85 lines make up a page. A page is therefore may have a maximum of 170 lines. Heading and spacing out of paragraphs take up about 20% of the space. Thus the average page may have 130-135 lines.

The Editor would appreciate if contributing authors would plan their articles to be in multiples of half pages long. It will improve the magazine's presentation and makes life easier for the volunteer staff. It is always a headache to decide what to do with the last 8 to 12 lines of a poorly set out contribution, which does not fit the page.

Drawings, illustrations, graphics and other non printable line drawings will be pasted in as received. These have to fit the original A3 column format. One column width is 125 mm. Of course one have to have a margin on both side, therefore the submitted original should not exceed 110 mm in width. The printer charges $6.- per item to photographically bring it to the correct size. It also takes some time! Large diagrams which may not be drawn 110 mm wide would use both columns. The original should not exceed 240 mm in such case.

Photos can not be printed direct. These have to be screened first. Screening takes time, and it is costly. Colour photos are not suitable. These will not produce a satisfactory black and white print. Generally the printer has to reshoot the pix. on a black & white negative, then prepare a b&w enlargement and screen it. One can see that this is a costly process. So, be careful with photos! The rule is: if you have to include a photo, then this should be a crisp, well cropped black and white enlargement exactly the right width and somewhat more contrasty than an average snap shot. Acres of sky, foreground or blank walls etc. have to be cut away. The average amateur photo is seldom good enough for a publication.

Be a ruthless judge, and ask yourself the question before submitting a photo for publication: -is it worth the 10-15 dollars of additional expense?

Well, armed with this brief inside information, now is your chance to see your name in print. You never can tell, one day you may even be the author of a best seller. One got to start somewhere! o

* * *

TECHO TIME

with John Paine

Welcome again fellow users,

This month I wish to inform you of the progress of the RAMDISK and to announce a new section to this regular column called appropriately FAULT OF THE MONTH which will outline some of the symptoms, faults and fixes encountered on this machine of ours.

This section is for the benefit of all users, hackers and general techo types that may care to know what can go wrong.

RAMDISK PROGRESS.

A complete Manual and bill of material will be made available to ALL regional group leaders and the actual parts list is now available from the club shop and will be passed to all on receipt of the funds for the bare board. Remember, the first production run is for 50 boards only and these will be distributed on the basis of first paid, first served.

Space limitations prevent the parts list from appearing in this column but I have passed the file on to the EDITOR for inclusion elsewhere.

A few more words on the RAMDISK would now seem to be appropriate.

The RAMDISK is a peripheral card for the 99/4A computer which comprises of battery backed, CMOS RAM chips and emulates all the functions of a TI floppy disk drive. The primary advantage of the RAMDISK is speed; data transfer takes place at approximately 20-30 times faster than a standard disk drive. Those of you that may have MYARC or CORCOMP cards with head step times of 3-6 mSec may find the speed about 15-20 times faster than normal.

To use the RAMDISK you will require the following hardware:

1) TI 99/4A Peripheral Expansion Box, Hunter Valley Expansion System, or 2 Slot Poormans Expansion System.

2) TI, Corcomp or Myarc Disk Controller Card

(Mechatronic System is as yet untested.)

3) At least one floppy disk drive.

4) One of the following Command Modules;

Editor Assembler

Extended Basic

Mechatronics Extended Basic

Mini Memory Module

Super Cartridge or Maximem

5) 32k Memory expansion is desirable but not necessary.

The testing of the prototype has shown that the RAMDISK appears to be compatable with TI and 3rd party software except where non standard direct disk accesses are used within programs.

Programs tested successfully for compatability include TI Writer, FunnelWriter, Extended Basic, Basic, Assembly Language programs, MultiPlan, Logo and Forth.

Some programs which do direct access to disk controller appear not to be compatable, examples include CorComp disk manager, Pascal and Plato.

The DSR (Device Service Routine) will have provision for a number of CALL Subprograms which will allow maximum utilisation of the RAMDISK out of Basic and Extended Basic. Some typical examples are;

CALL DN(x)	Set Drive Number
CALL MS(xxx)	Set maximum number of sectors
CALL WO	Turn on write protect
CALL WF	Turn off write protect
CALL EX(xxxx)	Execute Machine Language Program
CALL DM	Load and execute Disk Manager 1000

Other CALL's are also to be implemented and there is provision for you to implement your own CALLs as needed.

As this project is considered to be relatively advanced, I believe that construction should be attempted at the local Regional Group level as a group activity. Country and interstate members should not despair if there is no Regional Group nearby, as I will try to arrange for technically competent constructional help during the course of events.

I appeal to all REGIONAL CO-ORDINATORS to contact me shortly, so that the necessary construction details and software can be distributed, for work to begin by the time the boards become available in APRIL.

I would now like to pay tribute to the team of dedicated fellow TI club members who now make up the nucleus of a TECHNICAL SPECIAL INTEREST GROUP. This group devoted a whole Saturday last month to preparing, manufacturing and final testing of 32K memory boards for the club to pass on to you, the members, so that those bare consoles now can be upgraded with the absolute minimum effort and worry.

These people are a special group in my eyes and I wish to thank John O'Brien, Cyril (The Shop) Bohlsen, Ross (Sysop) Muddie, Peter Schubert (Modem Fame) and a special thanks to Robert Peverill who patiently cut and prepared those terrible ribbon cables which allowed the rest of us to do the glamorous job of wiring, soldering and testing.

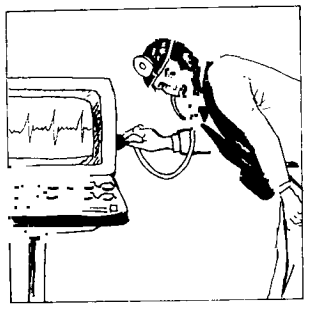
The end result of this activity is that YOU, the member can have that much needed 32K expansion for your console at the same price that kitsets were sold at last year, without the fear of the unit not being functional. All boards that will be on the shop shelf are 100% assembled and tested. All that is necessary is for you to solder 5 wires to the motherboard and even that task is made simpler by the inclusion of a four page instruction sheet.

One major point that needs to be made now is that this 'Special Interest Group' does NOT comprise of TECHO TYPES exclusively. I am sure that John O'Brien would not think it unkind of me to say that he could be classed as a pure novice, BUT he has contributed with his time and efforts to help others. WILL YOU ?

From time to time during the year other projects will be attempted, so please, if you wish to participate, please let me know if you can help. Do not be timid about your lack of Technical Knowledge. TECHO's are a dime a dozen, helpers and contributers are rare.

See page 6 for Parts List for Ramdisk

TECHO TIME



FAULT OF THE MONTH

During the last 6 months I have diagnosed over 200 consoles and although not successful with all, some interesting facts have emerged.

The majority of faulty consoles that I have seen were manufactured around or just after August 1983.

This period was at the height of the great selloff of home computers in the US and here just a little later on. The only assumption I can make now is that the quality control and heat soak testing of late model consoles must have been allowed to slip somewhat, after all , these procedures are extremely costly to maintain and we know that at one stage, the US retail price fell as low as $27.00 US. Little wonder that Q.C. fell away.

Anyway, on to a very common fault with a multitude of symptoms, which luckily is very easy to fix.

SYMPTOMS

1) Screen ocassionally flashes and sometimes a loud crack is heard on audio.
2) When screen flashes background color shows up as pink.
3) Console seems to sometimes ignore PEB and and cannot address any peripheral devices including cassette recorder.
4) After running for some time screen slowly fades away and console appears to take a long time to boot up again after reset.

Sounds like a terrible tale of woe but generally easy to diagnose and repair.

THE FIX

The culprit is usually a small choke that TI used to decouple the +5v rail to the video display chip crystal oscillator circuit.

This choke is a honeycomb wound type which has all it's windings interleaved. The choke is encapsulated in a browny coating which has the ability to soak up as much moisture as possible. I have dissected a number of these chokes to find that the interleaved windings have all manner of green stuff growing in and around them.

A resistance check will show a low impedance path which would normally be expected, but the crunch comes when you look look at the circuit.

This choke was designed to provided a relatively low DC resistance for the oscillator and a high impedance path back to the DC rail from the crystal. With the corrosion and moisture it has absorbed, this little sucker is effectively allowing the low impedance +5v rail to shunt the high impedance crystal circuit which is trying it's heart out to drive the relatively drive hungry VDP chip.

A good crystal circuit will drive the VDP chip with a signal swing from about 0.2V to about 3.2V which just barely makes the grade if you read TI Semiconductor Data sheets on the TMS9918/28/29 chip.

If you suspect that this problem is causing some of the symptoms mentioned above, look at this circiut first. Using a scope with probe look at waveform on pin 39 of VDP . Make sure that the vertical amplifier of the scope is switched to DC and ensure that the voltage swing is from about 0.2V to 3.2V peak to peak. Most bad choke circuits will show a voltage from 1V to 2V peak to peak.

The choke used by TI is unspecified in the technical manual I have, but it would appear to be around 120uH. I have successfully replaced this choke with axial type chokes as low as 4uH with no apparent ill affects.

For those that wish to examine the VDP circuit a little more closely, the TI reference number for this device is L207 and connects directly to pin 39 of VDP chip.

That concludes this month's Techotime and I need your feedback on the material presented. Do you want more of this type of technical information? Do you want a continuation of FAULT OF THE MONTH. Please let me know.

I can be contacted via Club Mail Box, BBS username TECHOTIME or by telephone on
(W) 02 8197200
(H) 02 6256318
(INT)61-2-6256318

PARTS LIST FOR RAMDISK

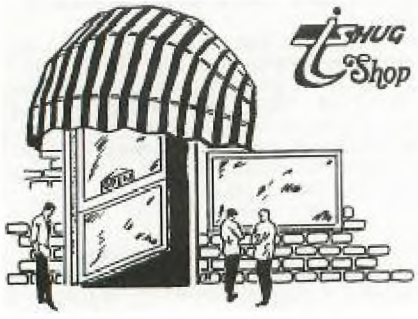
The follawing list is a general parts list for the RAMDISK Project
Check the club shop for any specials on same af these parts.

COMPONENT LIST

C1, C5, ALL Bypass	.1 uf, 20%, 15v, Capacitors	25
C2, C3, C4	10 uf, 20%, 25v, Tantalums	3
R3, R4, R6, R7, R8, R10	2.7K, 1/4 Watt, 10%, Resistors	6
R1	33 OHM, 1/4 W, 10%, " "	1
R5	6.8K, 1/4 W, 10%, " "	1
R2	270 OHM, 1/4 W, 10% " "	1
R9	270-470 OHM, 1/4 W, 10% "	1
CR1, CR2	Light Emitting Diades	2
CR8	IN4001 Diode	1
CR3, CR9, CR10	IN914 Ddodes	3
CR4, CR5, CR6, CR7	IN34A Diodes	4
Q1	2N2222 Transistor	1
Q2	7805 Regulator	1
U1, U19, U20	74LS138 IC	3
U2	74LS154 IC	1
U9	74LS259 IC	1
U3, U4, U5, U6, U7, U8, U11, U12, U13, U14, U15, U16, U17	6264L15 IC (CMOS RAM)	13
U10	74LS02 IC	1
U18	74LS156 IC	1
U21, U22	74LS244 IC	2
U23	74LS245 IC	1
Ni-Cad. Rechargeable Batteries	AAA	3
Single Cell Battery Holders	AAA	3
Heat Sink (suit 7805 regulator)	screw mounting	1
Dip-Switch	B Position	1

FOR 720 SECTORS ONLY

TOP1, TOP2, TOP3, TOP4, TOP5, TOP6, TOP7, TOP13, TOP14, TOP15, TOP16	6264L15 IC (CMOS RAM)	11



TIsHUG SHOP APRIL 1987

Well the first day of running the shop is over, and I have just about recovered from the ordeal. I would like to thank all the people who assisted me on the day.

There was a fair amount of interest in the second-hand computer parts we had on sale. Enough to have made it worthwhile bringing them to the next meeting.

SHOP INVENTORY.
This is very similar to last months:

HARDWARE
Micropendiums .....$ 2.90
(1986-June to Dec.)(1987-Jan.Feb.)
Spike Protectors.....$29.00
Consoles Ver.2-2.....$65.00
1 only Double Disk Drive Cable Set..$30.00
32k Matchbox Memory Expansion $55.00
(All these units are fully assembled and include a $10.00 deposit on the GROM PORT.)

SECOND HAND ITEMS
Keyboards.....$15.00
Power Supply Boards.....$15.00
Grom Ports.....$12.00
Ivory Console Cases.....$ 2.00
No Good Mother Boards (spare parts).$ 5.00

BOOKS
Millers Graphics Smart Prog.Guide...$ 7.50

SOFTWARE
Console Writer Modules.....$45.00
Auto Spell-Check (TI-Writer).....$45.00
Club Software Tapes.....$ 3.00
Club Software Disks.....$ 5.00
Boxes of Blank Disks (10).....$19.00

RAM DISK
There are still a couple of the RAM DISK circuit boards remaining $35.00

The Shop is now taking orders for
6264LP-15 RAM CHIPS
(13 req'd. for single side configuration)
(24 req'd. for double side configuration)
A MINIMUM DEPOSIT OF 50% IS REQUIRED WITH ORDER
Price per chip .....$ 5.50

As a service to the builders of the Ram card the shop is interested in supplying some of the components in kit form, but it will require at least 25 people to pay their money first.

The cost will be approximately.....$20.00

The kit will include all components required with the exception of :-
Printed circuit board.
62641p-15 Ram chips.
Chip Sockets.
This price is only available through the shrewd bargaining of our Tech. Co-ord. John Paine. The price is about half of what it would be if you did the purchasing of single items yourself.

If you are interested in ordering this kit please let me know A.S.A.P. by phone or on the BBS. with credit card authorization included.

Cyril.

*** For Sale ***

TI RS232 Card	$160
TI 32K Card	$80
TI Disk Controller	$120
Shugart Disk Drive	$120
Plato Module	$40
Video Chess Module	$20
PRK Module	$15
Music Maker Module	$15
Parsec (two modules)	$15 ea
Touch Typing Module	$15
Tunnels of Doom	$20

CONTACT LOU AMADIO
042-284906

FOR SALE

$50

Contact: F. BARTSCH

Phone: 042 28-7270

FOR SALE

EDITOR/ASSEMBLER

$70

Contact: Justin Richards

Phone: 86-3860 (A/H)

****
FOR SALE
****
CICADA 300 MODEM
$80 ono. COMES WITH SOFTWARE.
PHONE 639-1031 (A/H)
GAVAN ELEMENT.

SOFTWARE column by Terry

First up some news of recently acquired software from local and overseas sources.

Disk Manager 1000 - Version 5.5, which includes new features in the file utilities area. Of special interest is the ability to use the "P" command to dump a D/V80 file to your printer. Watch for this one at the shop shortly.

Diskhacker - continuing on with the good work from our friends at Funnelweb Farm (Newcastle) comes this extremely versatile utility from Will McGovern. It is being released under the Fairware option so if you come by a copy, and it will also be available at the shop shortly, do the right thing and send off your contribution to Will. And speaking of Funelweb Farm, the latest version (3.4) of Funlwriter has arrived on my desk. I haven't used it as yet, so cannot tell you what the changes may be.

Freddy - a colorful, addictive game from West Germany. Unfortunately copyrighted so it can't be distributed through the shop. If time permits at an upcoming meeting I will demonstrate the game for those who have not seen it. I know many of you have seen it at various Regional Group meetings.

Midnight Mason and 4A Flyer - again 2 copyrighted games, the former similar to TI Runner, though not as complex, while the latter is a fair attempt at a flight simulator for the TI. Still no where near as good as the industry standard Microsoft Flight Simulator but never-the-less a reasonable attempt. And speaking of Microsoft and the upcoming IBM compatability for the TI, I thought you may be interested in this bit of news recently received from an overseas correspondent.

"Things are hotting up in the TI world here. MYARC have finally come out with their new Geneve computer which is undoubtedly a fantastic computer - but Myarc products have been known to be full of bugs when first released, and if their venture fails the buyers of the computer will be stuck with the orphan of an orphan with no support whatsover! Not for me.

And MG has gotten together with Triton to offer some kind of a hybrid which amounts to an IBM clone on a card stuck in the PEB and attached to a TI keyboard. Since a TI keyboard can't handle many IBM programs, and a TV can't handle an 80 column screen, it seems to me you would have to also buy a keyboard and a high resolution monitor. So, $499 gets you the guts of an IBM clone without a case, keyboard or anything else - and self standing IBM clones are going for not much more than that!"

So there you have it. Of course the correspondents comments are his own and in no way endorsed by TIsHUG.

Well enough of that and down to more serious business such as the software to be released at the shop in April. Regrettably here I must mention this. While there is still plenty of good software around for disk users, the supply of good software suitable for distribution on cassette is rapidly drying up. I guess the main reason for this is that the majority of TI users, especially software authors, have a full system and write their software acordingly. This is not to say that there won't be any more cassettes issued. There certainly will be, but the point is that the quality of the programs contained on the tapes may not be as good a deal as disk users are getting. My apologies for this but rest assured I will do the best I can.

ON DISK:

There are 3 disks and while they are heavily musically oriented, I urge you not to dismiss them as not being the thing for you. Each in their own way show a different facet of your TI and are fine examples of programming skills. Here they are:

STARTREK THEMES - from Ken Gilliland. Themes included are from the TV series and the first 3 movies. Some very nice high-res graphics add to the class of this disk. The disk auto-loads through XB and contains a documenation file from Ken which will tell you how he went about his programming task. Requires 32K expansion.

SOUTH PACIFIC VOLUME 1 - also from Ken Gilliland. This one contains 3 songs from the classic movie with the high-res graphics characters actually singing along to the music. This has to be seen to be believed. Exceptionally well done. Again auto-loads in XB and requires 32K expansion.

TI SINGS - a remarkable fairware offering from Barb Berg. For this one you will need the TE2 module. TI SINGS will enable you to create songs, refine them and then play them back all done with alphons. This is a very exciting and new way to create music. On the disk are 7 examples for you to see how it's done. This is really a disk not to be missed.

ON TAPE:

TAPE 1987/04 will contain the following programs:

3D LABYRINTH - a maze escape game in Extended Basic. Full instructions are included. It takes a while to set up so be patient and good luck in finding your way out. I think you will need it as all my attempts failed.

A-B GAMES - a fairly simple bouncing ball type of game with instructions included. It will run in either basic or extended basic.

ACHILLUS AND THE MOLE - is a chasing/capture game where you help Achillus catch the mole. It is quite complex and not all that easy to play. XB required.

ANGRIFF - a fast reaction shooting game, simple but good fun. Optional speech option is included and will run in either basic or extended basic.

CAR RACE - a well done car racing game in XB. Avoid the oncoming cars by manouvering with the No.1 joystick.

BRIDGE ON RIVER KWAI - in XB. The idea is to build a bridge across the river with your limited supply of building materials. When you think it's ready, cars drive across to test it. Remember, you will need to build a strong bridge to succeed in this game.

US FLAGS - is an extremely well done historical program in extended basic. Flags from 1777 to the present day are graphically shown. Perhaps this program isn't for everyone but students of US history should find it very useful.

BALLOON JUMP - a nicely programmed game in XB where you have to jump from balloon to balloon using the arrow keys. Not all that easy so it should keep you amused for a fair time.

Programs on tape 1987/04 will also be released on disk for those who prefer that medium.

OK. That's it for this month. Have fun and read this column next time for news on software releases. o

Sound Off

The Illawarra Regional Group

This group holds regular meetings on the third Monday of each month (except January) at 7.30 p.m. at the Keiraville Public School, Gipps Road, Keiraville, opposite the Keiraville shopping centre.

We also hold occasional hardware and other special interest group meetings at irregular intervals. We are offering memory expansion and other simple hardware expansions upon request, and are working on software for systems without disks but with memory expansion. Basic service facilities for members are also available.

The meetings normally start with a tutorial session on Extended BASIC, followed by a talk and demonstration of some other topic of interest. This leads to some refreshments while members meet each other and chat about problems and interests. We maintain various libraries for the use of members.

SUTHERLAND REGIONAL USERS GROUP

After a one month break during January the Sutherland Regional Group met at the home of Peter Young on Friday 20th February 1987.

A faulty 32k card was repaired during the preceeding week thanks to a memory test program in the June 1986 edition of Micropendium and the replacement of a failed chip by fellow member, Garry Wilson.

Like most other Users, the Sutherland Group has been on the lookout for a good data base program. The latest offering "Data Base 1" appears to fit the bill as it is a well structured program and also well documented.

A new release of assembly language games from EMI proved popular with the kids.

Next meeting at the same location on 20th March 1987.

Regards
Peter Young

BANANA COAST REGIONAL GROUP REPORT

The fourth meeting of the Banana Coast Regional Group was held at Kevin Cox's home and 7 TI members attended. Great interest was shown as Keir Wells demonstrated how to clean the GROM PORT inside the console. A lot of dismay was evident having found being an LED inside the console connected to the ON/OFF switch. Many modifications are assured in this group so the LED will be visible.

Other modules were demonstrated: e.g. Home Budget Management, Personal Record Keeper plus some assembly programmes with the Mini-memory Module.

Kerry Harrison showed a Frogger programme written by her brother. She, also, brought along a tasty afternoon tea for all.

By the next mmeting we hope to have the 32K memory expansion fitted to at least one console. The Illawarra Group has kindly sent us a tape of programmes that require the 32K memory expansion.

Glebe Regional Group

TO ALL IN OR NEAR THE CITY OR WHO DO NOT MIND TRAVELLING. DONT FORGET THE GLEBE REGIONAL GROUP IS NOW FULLY OPERATIONAL AGAIN AND ALL ARE WELCOME AT THE MEETINGS.

MEETINGS ARE HELD ON THE THURSDAY FOLLOWING THE MONTHLY MEETING AT 43 BOYCE ST GLEBE AT 8 PM.

FOR FURTHER INFO PLEASE CONTACT Mike Slattery on 692-0559.

CARLINGFORD REGIONAL GROUP

The April Meeting of the group will be held at Chris Buttner's home - 79 Jenkins Road, Carlingford. The February meeting was taken up largely with demonstrations and a clinic on assembly language programmes. It is hoped to continue the Assembly Language courses held in abeyance since February with assistance from Shane Ferrett.

MORE
Reports from our Regional Home-group leaders...

```
90 REM CRAZYFRAZEII
100 REM PAUL YORKE 12/16/83
105 REM 1200 STARFISH LN. ST
UART FL 33494
110 REM   RE-WRITTEN 5/14/84

120 REM PRINTER OPTION
130 REM  TEII SPEECH OPTION
140 REM FILL IN THE BLANKS
150 DIM NE$(30),NEE$(30),B$(
30)
160 CALL CLEAR
170 PRINT "*****************
***********"
180 PRINT "*****************
***********"
190 PRINT "*****
      *****"
200 PRINT "*****   CRAZYFRAZ
EII   *****"
210 PRINT "*****
      *****"
220 PRINT "*****   BY PAUL Y
ORKE  *****"
230 PRINT "*****
      *****"
240 PRINT "*****************
***********"
250 PRINT "*****************
***********": : : : :
260 INPUT "INSTRUCTIONS?(Y/N
) ":F$
270 IF SEG$(F$,1,1)="N" THEN
 360
280 REM ********************
START OF INSTRUCTIONS ******
**************
290 PRINT "THIS GAME IS BASE
D ON THE   PARLOR GAME ""TEL
EPHONE"".":"YOU ONLY SEE THE
 SCREEN":
300 PRINT "WHEN IT'S YOUR TU
RN.":"SO IT'S BETTER IF PLAY
ERS ":"ARE IN ANOTHER ROOM."
: : :
310 PRINT "THIS PROGRAM WILL
 ACCEPT A  ": :"LINE OF TEXT
.  EVERY THIRD": :
320 PRINT "WORD IN THAT LINE
 WILL BE": :"REPLACED BY THE
 NEXT PLAYER": :"WITH A WORD
 OR WORDS.": :
330 PRINT "THIS WILL CONTINU
E UNTIL ": :"SOMEONE INPUTS
""STOP STOP""": :
340 INPUT "PRESS ENTER TO CO
NTINUE ":DUM$
350 REM ********************
 END OF INSTRUCTIONS *******
**************
360 CALL CLEAR
370 REM  ********************
* START OF ORIGINAL TEXT INP
UT AND PROCESSING **********
**********************
380 PRINT "NOW IT'S TIME TO
TYPE IN AN    ORIGINAL LINE
OF TEXT.": : :
390 PRINT "  MAKE IT MORE TH
AN THREE           WORDS LON
G.": : :"   DON'T USE ANY CO
MMAS.": : :
400 PRINT "   ALWAYS END TEX
T WITH A              PERIOD.
": : :
410 PRINT "INPUT YOUR TEXT"
420 INPUT A$
430 IF A$="" THEN 380
440 GOSUB 1330
450 CALL CLEAR
460 IF ASC(SEG$(A$,LEN(A$),1
))=46 THEN 560
470 PRINT "YOU DID NOT PUT A
 PERIOD AT THE END OF YOUR T
EXT": :
480 PRINT "PRESS ""A"" TO AD
D A PERIOD": :
490 PRINT "PRESS ""R"" TO RE
-INPUT YOUR             TEXT"
: : :
500 CALL KEY(0,KEY,STATUS)
510 IF STATUS=0 THEN 500
520 IF KEY=67 THEN 670
530 IF KEY=82 THEN 410
540 IF KEY<>65 THEN 490
550 A$=A$&"."
560 PRINT "THE TEXT YOU JUST
 TYPED IN    IS NOW BEING PR
OCESSED": : :"   GO GET THE
NEXT PLAYER": : : :
570 FOR P=1 TO LEN(A$)
580 IF ASC(SEG$(A$,P,1))=46
THEN 670
590 IF ASC(SEG$(A$,P,1))<>32
 THEN 660
600 IF ASC(SEG$(A$,P+1,1))<>
32 THEN 660
610 FOR PP=P+2 TO LEN(A$)
620 IF ASC(SEG$(A$,PP,1))<>3
2 THEN 640
630 NEXT PP
640 A$=SEG$(A$,1,P)&SEG$(A$,
PP,LEN(A$)-P)
650 P=P+1
660 NEXT P
670 FOR P=1 TO LEN(A$)
680 IF ASC(SEG$(A$,P,1))=32
THEN 750
690 IF ASC(SEG$(A$,P,1))=46
THEN 740
700 C$=C$&SEG$(A$,P,1)
710 IF ASC(SEG$(A$,P,1))=46
THEN 750
720 NEXT P
730 GOTO 790
740 B$(G+2)="."
750 G=G+1
760 B$(G)=C$
770 C$=""
780 GOTO 720
790 GOSUB 1330
800 X=4
810 REM ********************
****** END OF ORIGINAL TEXT
PROCESSING ****************
*****
820 REM ********************
**** START OF PLAYER INPUT C
YCLE ***********************

830 X=X-1
840 IF PLA>28 THEN 1350
850 IF X<1 THEN 800
860 F=X
870 Z=Z+1
880 IF B$(Z)="" THEN 1000
890 IF LEN(N$)<196 THEN 910
900 GOTO 2060
910 IF B$(Z)="." THEN 930
920 IF Z=X THEN 950
930 N$=N$&B$(Z)&" "
940 GOTO 870
950 N$=N$&"..."&STR$(X)&"..."
"&" "
960 BLANK=BLANK+1
970 Q=Q+1
980 X=X+3
990 GOTO 870
1000 CALL CLEAR
1010 PRINT "YOU ARE PLAYER N
UMBER ";PLA+2: : :
1020 PRINT "YOU HAVE  ";BLAN
K;" BLANKS TO FILL.": : :
1030 BLANK=0
1040 PRINT N$;" ";NN$: : : :
1050 FOR INPU=F TO X-3 STEP
3
1060 N$=""
1070 NN$=""
1080 PRINT "INPUT ";INPU;
1090 TEMP$=B$(INPU)
1100 INPUT B$(INPU)
1110 IF B$(INPU)<>"" THEN 11
40
1120 PRINT "YOU MUST TYPE IN
 A WORD OR  PHRASE TO FILL I
N THE BLANK AND THEN PRESS E
NTER"
1130 GOTO 1040
1140 IF B$(INPU)="STOP STOP"
 THEN 1380
1150 NEXT INPU
1160 CALL CLEAR
1170 PRINT "IF YOU JUST FINI
SHED TYPING GIVE THE NEXT GU
Y A TURN": : :
1180 INPUT "IF YOU ARE THE N
EXT PLAYER  PRESS ENTER.":DU
M$
1190 PLA=PLA+1
1200 Z=1
1210 IF B$(Z)="" THEN 1270
1220 IF LEN(NE$(PLA))<200 TH
EN 1240
1230 GOTO 2020
1240 NE$(PLA)=NE$(PLA)&B$(Z)
&" "
1250 Z=Z+1
1260 GOTO 1210
1270 Q=0
1280 Z=0
1290 X=F
1300 REM
1310 GOTO 830
1320 REM ********************
***** END OF PLAYER INPUT CY
CLE************************
*
1330 ORIG$=A$
1340 RETURN
1350 CALL CLEAR
1360 PRINT "YOU HAVE RUN OUT
  OF MEMORY GO GET THE GANG
AND LISTEN  TO THE CRAZY FRA
ZE": : : :
1370 INPUT "PRESS ENTER":DUM
$
1380 REM FINAL SCREEN
1390 CALL CLEAR
1400 INPUT "DO YOU HAVE SPEE
CH(Y/N) ":F$
1410 IF SEG$(F$,1,1)="N" THE
N 1620
1420 CALL CLEAR
1430 REM ********************
** SPEECH OUTPUT OPTION ****
**********************
1440 OPEN #1:"SPEECH",OUTPUT
1450 PRINT "PRESS ANY KEY TO
 CONTINUE": : : : : :
1460 PRINT "THIS IS THE ORIG
INAL TEXT": :ORIG$: :
1470 PRINT #1:"THIS IS THE O
RIGINAL TEXT":ORIG$: :
1480 PRINT #1:"PRESS ANY KEY
 TO CONTINUE"
1490 FOR I=1 TO PLA
1500 PRINT NE$(I)
1510 PRINT NEE$(I): :
1520 PRINT #1:NE$(I)
1530 PRINT #1:NEE$(I)
1540 CALL KEY(0,K,STATUS)
1550 IF STATUS=0 THEN 1540
1560 NEXT I
1570 CLOSE #1
1580 GOTO 1760
```

Program continued on page 11

Continued from page 10

```
1590 REM
1600 REM
1610 REM ********************
*** SCREEN ONLY OPTION *****
************************
1620 CALL CLEAR
1630 B$(INPU)=TEMP$
1640 PRINT "PRESS ANY KEY TO
 CONTINUE": : : : : :
1650 PRINT "ORIGINAL TEXT:"
1660 PRINT ORIG$
1670 PRINT
1680 FOR L=1 TO PLA
1690 PRINT NE$(L);
1700 PRINT NEE$(L)
1710 PRINT
1720 CALL KEY(0,K,STATUS)
1730 IF STATUS=0 THEN 1720
1740 NEXT L
1750 PRINT ORIG$
1760 PRINT
1770 INPUT "PRINTOUT(Y/N)":F
$
1780 IF SEG$(F$,1,1)="N" THE
N 1900
1790 REM ********************
***** PRINTER OPTION *******
************************
1800 OPEN #1:"RS232/2.BA=960
0.DA=8"
1810 PRINT #1:ORIG$
1820 PRINT #1
1830 FOR L=1 TO PLA
1840 PRINT #1:NE$(L);
1850 PRINT #1:NEE$(L)
1860 PRINT #1
1870 NEXT L
1880 CLOSE #1
1890 REM ********************
****** REINITIALIZE FOR NEXT
 GAME ***********************
**
1900 Q=0
1910 G=0
1920 PLA=0
1930 N$=""
1940 C$=""
1950 FOR Z=1 TO 30
1960 B$(Z)=""
1970 NE$(Z)=""
1980 NEE$(Z)=""
1990 NEXT Z
2000 Z=0
2010 GOTO 360
2020 REM OVERFLOW PROTECTION

2030 NEE$(PLA)=NEE$(PLA)&B$(
Z)&" "
2040 Z=Z+1
2050 GOTO 1210
2060 REM INPUT OVERFLOW
2070 IF B$(Z)="." THEN 2090
2080 IF Z=X THEN 2110
2090 NN$=NN$&B$(Z)&" "
2100 GOTO 870
2110 NN$=NN$&"..."&STR$(X)&"
..."&" "
2120 GOTO 960
```


```
1 REM ********************
2 REM *CAVERNS OF CARNAGE*
3 REM *    BASIC ONLY    *
4 REM ********************
10 CALL CLEAR
20 CALL SCREEN(5)
30 PRINT "CAVERNS OF CARNAGE
"
40 PRINT
41 PRINT "PRESS S(LEFT)OR D(
RIGHT) TO MOVE"
50 PRINT "PRESS THE SPACE BA
R TO FIRE."
60 PRINT "PRESS ANY KEY TO B
EGIN."
70 CALL KEY(0,K,S)
80 IF K<0 THEN 70
90 CALL CLEAR
100 RANDOMIZE
110 SCORE=0
120 LL=16
130 RR=4
140 L=16
150 SW=7
160 CALL SCREEN(2)
170 REM**CHAR DESIGN
180 CALL CHAR(152,"1819FF3D3
C3C2770")
190 CALL CHAR(144,"1819FFBC3
C3C4280")
200 CALL CHAR(126,"183D7EBC3
CCD4208")
210 CALL CHAR(33,"FFFFFFFFFF
FFFFFF")
220 CALL CHAR(87,"1818181818
181818")
230 CALL CHAR(96,"FFFFFFFFFF
FFFFFF")
240 CALL CHAR(101,A$)
250 CALL COLOR(10,2,2)
260 CALL COLOR(2,4,4)
270 CALL COLOR(9,7,7)
280 CALL COLOR(5,8,2)
290 CALL COLOR(7,2,7)
300 CALL COLOR(11,2,2)
310 CALL COLOR(12,16,7)
320 CALL COLOR(15,8,7)
330 CALL COLOR(13,11,4)
340 CALL COLOR(6,15,2)
350 CALL COLOR(3,16,2)
360 CALL COLOR(4,16,2)
370 CALL COLOR(14,16,4)
380 CALL COLOR(16,6,7)
390 FOR JK=24 TO 1 STEP -1
400 CALL HCHAR(JK,5,96,25)
410 NEXT JK
420 CALL HCHAR(RR,LL,152)
430 CALL HCHAR(RR,LL-1,144)
440 LET RN=RND
450 IF RN>.3 THEN 480
460 L=L+1
470 GOTO 500
480 IF RN<.7 THEN 500
490 L=L-1
500 IF L>=1 THEN 540
510 L=15
520 IF L>=1 THEN 540
530 L=1
540 CALL HCHAR(24,1,33,L)
550 CALL HCHAR(24,L+1,96,SW)
560 CALL HCHAR(24,L+SW,33,32
-L-SW)
570 CALL HCHAR(24,((SW-1)*RN
D+L+1),126)
580 CALL SOUND(100,1000*RND+
110,L)
590 CALL HCHAR(RR,LL-1,96,2)
600 PRINT
610 CALL HCHAR(RR,LL,152)
620 CALL HCHAR(RR,LL-1,144)
630 CALL GCHAR(5,LL,N)
640 IF N=33 THEN 1120
650 IF N=126 THEN 1120
660 CALL GCHAR(5,LL-1,N)
670 IF N=126 THEN 1120
680 IF N=33 THEN 1120
690 CALL KEY(0,K,S)
700 IF K=32 THEN 940
710 IF K=83 THEN 760
720 IF K=68 THEN 850
730 CALL HCHAR(RR-1,LL-1,96,
2)
740 GOTO 420
750 REM**INITIAL POSITIONS
760 CALL GCHAR(RR,LL-1,N)
770 IF N=126 THEN 1120
780 IF N=33 THEN 1120
790 CALL HCHAR(RR,LL-1,96,2)
800 LL=LL-1
810 IF LL<1 THEN 890
820 CALL HCHAR(RR,LL,153)
830 CALL HCHAR(RR,LL-1,144)
840 GOTO 1110
850 CALL GCHAR(RR,LL+1,N)
860 IF N=126 THEN 1120
870 IF N=33 THEN 1120
880 CALL HCHAR(RR,LL-1,96,2)
890 LL=LL+1
900 IF LL>32 THEN 800
910 CALL HCHAR(RR,LL,153)
920 CALL HCHAR(RR,LL-1,144)
930 GOTO 420
940 REM**POSITRON BLAST
950 FOR LK=4 TO 22
960 CALL GCHAR(LK+1,LL,N)
970 IF N=126 THEN 1030
980 IF N=33 THEN 420
990 CALL HCHAR(LK+1,LL,87)
1000 CALL HCHAR(LK+1,LL,96)
1010 NEXT LK
1020 GOTO 420
1030 CALL HCHAR(LK+1,LL,96)
1040 CALL SOUND(100,-7,0)
1050 CALL SOUND(100,-7,5)
1060 REM**SCORE
1070 SCORE=SCORE+23-LK
1080 FOR BB=1 TO LEN(STR$(SC
ORE))
1090 CALL HCHAR(1,BB+16,ASC(
SEG$(STR$(SCORE),BB,1)))
1100 NEXT BB
1110 GOTO 420
1120 FOR BB=1 TO LEN(STR$(SC
ORE))
1130 CALL HCHAR(1,BB+16,ASC(
SEG$(STR$(SCORE),BB,1)))
1140 NEXT BB
1150 X$="ANOTHER GAME?"
1160 FOR BB=1 TO LEN(X$)
1170 CALL HCHAR(24,BB+3,ASC(
SEG$(X$,BB,1)))
1180 NEXT BB
1190 CALL KEY(0,K,S)
1200 IF K=89 THEN 90
1210 IF K=78 THEN 1220 ELSE
1190
1220 END
```

```
100 REM   ** DON QUIXO-TI **
105 REM   JOHN&NORMA CLULOW
106 REM      DECEMBER 1983
110 REM
120 REM  DESIGN FROM K.SHEET
S
130 REM   -- UCC  MARRIAGE -
-
140 REM   --   ENCOUNTER   -
-
150 REM
160 CALL CLEAR
170 CALL SCREEN(12)
180 FOR I=1 TO 16
190 CALL COLOR(I,12,12)
200 NEXT I
210 FOR I=1 TO 79
220 READ R,C,CH,P$
230 CALL CHAR(CH,P$)
240 CALL HCHAR(R,C,CH)
250 NEXT I
260 CALL COLOR(9,7,12)
270 FOR I=10 TO 16
280 CALL COLOR(I,2,12)
290 NEXT I
300 CALL COLOR(1,2,12)
310 CALL COLOR(2,2,12)
320 CALL HCHAR(18,8,41)
330 CALL HCHAR(18,9,42)
340 CALL HCHAR(16,22,44)
350 CALL HCHAR(16,23,45)
360 CALL HCHAR(17,22,46)
370 CALL HCHAR(17,23,47)
380 CALL HCHAR(18,24,42)
390 CALL HCHAR(18,25,41)
400 CALL HCHAR(18,12,42)
410 CALL HCHAR(18,13,41)
420 CALL HCHAR(18,10,37)
430 CALL HCHAR(18,11,38)
440 CALL HCHAR(18,22,37)
450 CALL HCHAR(18,23,38)
460 A=110
470 B@=117
480 B=123
490 C=131
500 D@=139
510 D=147
520 E@=156
530 E=165
540 F=175
550 G@=185
560 G=196
570 A1@=208
580 A1=220
590 B1@=233
600 B1=247
610 C1=262
620 D1@=277
630 D1=294
640 E1@=311
650 E1=330
660 F1=349
670 G1@=370
680 G1=392
690 A2@=415
700 A2=440
710 B2@=466
720 B2=494
730 C2=523
740 D2@=554
750 D2=587
760 E2@=622
770 E2=659
780 F2=698
790 G2@=740
800 EN=300
810 QN=2*EN
820 DQ=3*EN
830 HN=4*EN
835 DQQ=5*EN
840 DH=6*EN
842 QDQ=7*EN
845 DHQ=8*300
850 CALL SOUND(EN,D1,0)
860 CALL SOUND(HN,D1,0,F1,0)
870 CALL SOUND(EN,B1@,0,D1,0
)
880 CALL SOUND(EN,B1@,0,E1@,
0)
890 CALL SOUND(EN,B1@,0,F1,0
)
900 CALL SOUND(EN,B1@,0,E1@,
0)
910 CALL SOUND(EN,B1@,0,D1,0
)
920 CALL SOUND(DHQ,D1,0,F1,0
)
930 CALL SOUND(EN,F1,0)
940 CALL SOUND(HN,E1@,0,G1,0
)
950 CALL SOUND(EN,E1@,0)
960 CALL SOUND(EN,B1@,0,F1,0
)
970 CALL SOUND(EN,B1@,0,G1,0
)
980 CALL SOUND(EN,B1@,0,F1,0
)
990 CALL SOUND(EN,B1@,0,E1@,
0)
1000 CALL SOUND(DHQ,E1@,0,G1
,0)
1010 CALL SOUND(EN,E1@,0,G1,
0)
1020 CALL SOUND(HN,F1,0,A2,0
)
1030 CALL SOUND(EN,D1,0)
1040 CALL SOUND(EN,E1@,0)
1050 CALL SOUND(EN,D1,0,F1,0
)
1060 CALL SOUND(EN,E1@,0,G1,
0)
1070 CALL SOUND(EN,C1,0,A2,0
)
1080 CALL SOUND(EN,E1@,0,B2@
,0)
1090 CALL SOUND(QDQ,C1,0,E1@
,0)
1095 IF FLAG=2 THEN 2040
1100 CALL SOUND(EN,D1,0,F1,0
)
1110 CALL SOUND(HN,E1@,0,G1,
0)
1115 IF FLAG=1 THEN 1190
1120 CALL SOUND(EN,C1,0)
1130 CALL SOUND(EN,D1,0)
1140 CALL SOUND(EN,C1,0,E1@,
0)
1150 CALL SOUND(EN,D1,0,F1,0
)
1160 CALL SOUND(EN,E1@,0,G1,
0)
1170 CALL SOUND(DHQ,E1@,0,A2
,0)
1175 FLAG=1
1180 GOTO 850
1190 CALL SOUND(EN,C1,0,E1@,
0)
1200 CALL SOUND(EN,D1,0,F1,0
)
1210 CALL SOUND(EN,E1@,0,G1,
0)
1220 CALL SOUND(EN,E1@,0,A2,
0)
1230 CALL SOUND(EN,E1@,0,B2@
,0)
1240 CALL SOUND(DH,E1@,0,C2,
0)
1250 CALL SOUND(EN,E1@,0,B2@
,0)
1260 CALL SOUND(EN,E1@,0,C2,
0)
1270 CALL SOUND(EN,E1@,0,B2@
,0)
1280 CALL SOUND(DQQ,F1,0,D2,
0)
1290 CALL SOUND(EN,D1,0,B2@,
0)
1300 CALL SOUND(EN,D1,0,B2@,
0)
1310 CALL SOUND(EN,D1,0,C2,0
)
1320 CALL SOUND(EN,D1,0,B2@,
0)
1330 CALL SOUND(DQQ,G1,0,D2,
0)
1340 CALL SOUND(EN,G1,0,B2@,
0)
1350 CALL SOUND(EN,D1,0,B2@,
0)
1360 CALL SOUND(EN,D1,0,C2,0
)
1370 CALL SOUND(EN,D1,0,B2@,
0)
1380 CALL SOUND(EN,F1,0,D2,0
)
1390 CALL SOUND(HN,F1,0,A2,0
)
1400 CALL SOUND(EN,F1,0,B2@,
0)
1410 CALL SOUND(EN,F1,0,C2,0
)
1420 CALL SOUND(EN,F1,0,B2@,
0)
1430 CALL SOUND(EN,F1,0,A2,0
)
1440 CALL SOUND(DQQ,G1,0,C2,
0)
1450 CALL SOUND(EN,G1,0)
1460 CALL SOUND(EN,G1,0,B2@,
0)
1470 CALL SOUND(EN,C1,0,A2,0
)
1480 CALL SOUND(EN,C1,0,G1,0
)
1490 CALL SOUND(HN,D1,0,B2@,
0)
1500 CALL SOUND(EN,D1,0,G1,0
)
1510 CALL SOUND(EN,D1,0,A2,0
)
1520 CALL SOUND(EN,D1,0,B2@,
0)
1530 CALL SOUND(EN,D1,0,A2,0
)
1540 CALL SOUND(EN,D1,0,G1,0
)
1550 CALL SOUND(HN,D1@,0,B2@
,0)
1560 CALL SOUND(EN,D1@,0,B2@
,0)
1570 CALL SOUND(EN,C1,0,C2,0
)
1580 CALL SOUND(EN,D1,0,D2,0
)
1590 CALL SOUND(EN,C1,0,C2,0
)
1600 CALL SOUND(EN,D1,0,B2@,
0)
1610 CALL SOUND(EN,F1,0,D2,0
)
1620 CALL SOUND(EN,F1,0,C2,0
)
1630 CALL SOUND(EN,D1,0,B2@,
0)
1640 CALL SOUND(EN,D1,0,D2,0
)
1650 CALL SOUND(EN,D1,0,B2@,
0)
1660 CALL SOUND(EN,D1,0,C2,0
)
1670 CALL SOUND(EN,D1,0,D2,0
)
```

```
1680 CALL SOUND(EN,G1,0,E2@,
0)
1690 CALL SOUND(EN,G1,0,D2,0
)
1700 CALL SOUND(DH,E1@,0,C2,
0)
1710 CALL SOUND(EN,A,30)
1720 CALL SOUND(EN,E1@,0,C2,
0)
1730 CALL SOUND(EN,D1,0,D2,0
)
1740 CALL SOUND(HN,E1@,0,E2@
,0)
1750 CALL SOUND(EN,G1,0,D2,0
)
1760 CALL SOUND(EN,E1@,0,C2,
0)
1770 CALL SOUND(EN,G1,0,E2@,
0)
1780 CALL SOUND(EN,G1,0,D2,0
)
1790 CALL SOUND(EN,E1@,0,C2,
0)
1800 CALL SOUND(HN,E1@,0,E2@
,0)
1810 CALL SOUND(EN,E1@,0,D2,
0)
1820 CALL SOUND(EN,E1@,0,E2@
,0)
1830 CALL SOUND(EN,E1@,0,D2,
0)
1840 CALL SOUND(EN,E1@,0,C2,
0)
1850 CALL SOUND(EN,E1@,0,B2@
,0)
1860 CALL SOUND(QDQ,G1@,0,A2
,0)
1870 CALL SOUND(EN,D1,0,A2,0
)
1880 CALL SOUND(EN,D1,0,B2@,
0)
1890 CALL SOUND(HN,E1@,0,C2,
0)
1900 CALL SOUND(EN,E1@,0,B2@
,0)
1910 CALL SOUND(EN,E1@,0,A2,
0)
1920 CALL SOUND(EN,E1@,0,C2,
0)
1930 CALL SOUND(EN,G1,0,B2@,
0)
1940 CALL SOUND(EN,G1,0,A2,0
)
1950 CALL SOUND(HN,G1,0,C2,0
)
1960 CALL SOUND(EN,B2@,2)
1970 CALL SOUND(EN,C2,4)
1980 CALL SOUND(EN,B2@,6)
1990 CALL SOUND(EN,A2,8)
2000 CALL SOUND(EN,G1,8)
2010 CALL SOUND(DQQ,G1@,8)
2015 CALL SOUND(EN,A,30)
2020 FLAG=2
2030 GOTO 850
2040 CALL SOUND(EN,E1@,0,B2@
,0)
2050 CALL SOUND(HN,F1,0,B2@,
0,D2,0)
2055 EN=600
2060 CALL SOUND(EN,D1,0,F1,0
,B2@,0)
2070 CALL SOUND(EN,D1,0,F1,0
,C2,0)
2080 CALL SOUND(EN,E1@,0,G1,
0,D2,0)
2090 CALL SOUND(EN,E1@,0,G1,
0,C2,0)
2100 CALL SOUND(EN,E1@,0,G1,
0,B2@,0)
2110 CALL SOUND(4000,D1,0,B2
@,0,F2,0)
2120 FOR I=1 TO 700
2130 NEXT I
2140 REM   CALL SOUND(666)
9000 REM  GRAPHICS DATA
9010 DATA 8,12,96,0000000800
0804
9020 DATA 8,13,97,0000002024
480102
9030 DATA 8,14,98,0000000040
80
9040 DATA 8,17,104,000101010
101
9050 DATA 8,18,105,008080808
0808080
9060 DATA 9,12,99,0001028444
140201
9070 DATA 9,13,100,F80402020
20204F8
9080 DATA 9,14,101,408000000
04020
9090 DATA 9,18,106,80818F8F9
EA78F8D
9100 DATA 9,19,107,C0C0C080
9110 DATA 10,12,102,00000000
102040
9120 DATA 10,13,103,00000202
4284
9130 DATA 10,16,108,00000000
00000303
9140 DATA 10,18,109,85858583
87808181
9150 DATA 10,19,110,63F7FFEF
EFEFEFEF
9160 DATA 10,20,111,C0E0E0F0
F0F0F0F0
9170 DATA 11,16,112,03070703
07060604
9180 DATA 11,17,113,80C16163
61616170
9190 DATA 11,18,114,8187C7C3
83FFFF3F
9200 DATA 11,19,115,FFFFEFEF
FFFFF8FC
9210 DATA 11,20,116,F0F0E0E0
C080
9215 DATA 12,15,43,000000000
00000C0
9220 DATA 12,17,117,7078F8FC
FF7E3F1F
9230 DATA 12,18,118,8383ABAB
DF9FBFFF
9240 DATA 12,19,119,FAE2FEFF
EFCFBEBF
9250 DATA 12,20,120,000020E0
F6FEEBFA
9260 DATA 12,21,121,00000000
70F0F0F0
9270 DATA 13,14,122,010301
9280 DATA 13,15,123,F0F0F070
7C7EFFFF
9290 DATA 13,17,124,1F1F1F0F
0707 0F0F
9300 DATA 13,18,125,FFBFBFBE
3FFFF7F7
9310 DATA 13,19,126,FBBF7F7F
7F7F7F7D
9320 DATA 13,20,127,37353C2D
AFBAB8BC
9330 DATA 13,21,128,F0F0F0D0
D0B0F0B0
9340 DATA 14,14,129,03030301
071F3F7F
9350 DATA 14,15,130,FFFFFFFE
FB87BFEE
9360 DATA 14,16,131,82820202
030FFFF7
9370 DATA 14,17,132,8F838383
07070703
9380 DATA 14,18,133,F1FB7F79
7979797B
9390 DATA 14,19,134,F9E1A1F0
60600080
9400 DATA 14,20,135,8EDEFDE0
FFFFC3C3
9410 DATA 14,21,136,B8B8BCB4
90100000
9420 DATA 15,14,137,7FFD7F77
7F0F0F07
9430 DATA 15,15,138,EEFFFFFF
FFFEFEFE
9440 DATA 15,16,139,F7870706
0606
9450 DATA 15,17,140,03030303
03010101
9460 DATA 15,18,141,7B797878
58B8B8B8
9470 DATA 15,19,142,80800000
00010101
9480 DATA 15,20,143,C36666C6
C6C68686
9490 DATA 15,21,144,00000000
41221408
9500 DATA 16,13,145,00000001
00000700
9510 DATA 16,14,146,03030100
E0A0FE00
9520 DATA 16,15,147,FA929292
8A8A8A8A
9530 DATA 16,16,148,00002010
08E70303
9540 DATA 16,17,149,216161C1
C1FBE770
9550 DATA 16,18,150,B9B8B1BF
9BD9DFD8
9560 DATA 16,19,151,63C3E3C7
F6C6460C
9570 DATA 16,20,152,86860C0C
1C3878FF
9580 DATA 16,21,153,48FF0000
000000F8
9590 DATA 17,14,154,00010107
0C3E7FFF
9600 DATA 17,15,155,8A8A8AE2
7FABDBDF
9610 DATA 17,16,156,070D0D07
86FFFFE5
9620 DATA 17,17,157,78404C4C
74FBD5B2
9630 DATA 17,18,158,D8D8D8D8
D8FB7FF8
9640 DATA 17,19,159,3B3F7F3F
EBF7EE23
9650 DATA 17,20,33,FEFFFDFAF
4802A51
9660 DATA 17,21,34,FFFFFFFDD
F3096A4
9670 DATA 18,14,35,A01F
9680 DATA 18,15,36,9EE30C
9690 DATA 18,16,37,0D056D
9700 DATA 18,17,38,6D2F68
9710 DATA 18,18,39,EF
9720 DATA 18,19,40,F6
9730 DATA 18,20,41,810E1B01
9740 DATA 18,21,42,5D73FF80
9750 DATA 16,10,44,000020100
8070303
9760 DATA 16,11,45,202060C0C
080E070
9770 DATA 17,10,46,070D09070
6FFFFE5
9780 DATA 17,11,47,78404C4C7
4FAD5B2
```

THE FACTS

CARTRIDGE PORT INFORMATION

by Mack McCormick

This is a text file discussion of the ROM cartridge port for the TI-99/4A. It represents information I have been able to obtain from various references. Cartridge programs must operate from >6000 to >7FFF. When the computer is RESET or turned on, the power up routine looks for a Header or Control block at location >6000 in the cartridge port. This contol block establishes the linkage into your cartridge program and allows you to have multiple entry points. Here is an example of a contol block used to provide one entry point;

```
0000 AA01        DATA    >AA01      6000   Valid
                                            cartridge
                                            header
0002 0000        DATA    >0000      6002
0004 0000        DATA    >0000      6004
0006 000C        DATA    CHAIN      6006   Address of
                                            menu list
0008 0000        DATA    >0000      6008
000A 0000        DATA    >0000      600A
000C 0000 CHAIN  DATA    >0000      6010   Chain
                                            pointer
000E 0020        DATA    SLOAD      6012   Entry point
0010   0F        BYTE    SLOAD-$-1  6014   Length of
                                            menu text
0011   54        TEXT    'CARTRIDGE NAME'
0020 0460 SLOAD  B       START
0022 092E
```

Let's examine the control block. If the TI operating system finds >AA at >6000 it knows that a cartride is plugged in. The next byte must be a >01 at location >6001. This informs the operating system that the code in the cartridge contains executable machine language. Other values at >6001 are found in GROM catridges, but that's another story. The data at location >6002 - >6005 is zero. Location >6006 must contain a word pointer to a list which identifies the menu text and associated entry point when that item is selected. This location usually contains a >600C. Locations >6008 - >600B must be zero. The chain list at >600C contains the following:

Bytes 1 & 2 = chain pointer to the next menu list - or >0000 if this is the only entry in the chain.

Bytes 3 & 4 = entry point associated with this menu selection.

Byte 5 = length of the menu text.

Byte 6 onwards = Menu Text - this is displayed on main menu. Craig Miller's newsletter has additional information on the power up routine for the computer. Remember all dynamic data must be in CPU RAM, usually in the >8300 area. This area is used for registers while VDP RAM is used for variable storage. Cartridges cannot REFerence any label or routine outside the cartridge. This means that the cartridge program must provide it's own VSBW, VSBR, VMBW, and VMBR routines which are normally loaded from the E/A cartridge. Examples of what these routines look like may be found in the source for Tombstone City or Craig Miller's newsletter. Armed with this information, it is possible to disassemble code to see how the program works. I hope you find this information useful.

LOGICAL AND, IS IT A CRAZY WAY TO ADD NUMBERS?
by Ben Takach

I would hazard a bet not many of you would know the result of the command: PRINT 7 AND 12. Well no prize for guessing, the answer is 4. This is what happens:
7 is binary 0111 and
12 is binary 1100.

If we add the two, using Boolean logic (0+0 or 0+1=0, 1+1=1) then we get: 0100, which is the binary number for 4.

Here is a one liner to display and print some weird tables of additions.

```
2 CALL CLEAR::INPUT "ENTER A NUMBER ":A:: FOR I=0 TO
20 :: PRINT A;"AND";I"IS";A AND I::NEXT I::FOR J=1 TO
3000 ::GOTO 2! LOGICAL "AND" EXERCISE
```

This "program" may be changed to print out the calculation so it may be studied at length.

```
2 OPEN#1:"PIO",OUTPUT::INPUT"ENTER A NUMBER ":A::FOR
I= 0 TO 20::PRINT#1:A;"AND";I"IS";A AND I::NEXT I::FOR
J=1 TO 3000:: CLOSE#1:: GOTO 2 ! LOGICAL "AND"
EXERCISE
```

Logical AND-ing may seem to be a useless and crazy exercise , however it is a useful and important function used for internal housekeeping and data management by the computer. Have fun with the one liner and the amazing results.

DM1000 Modifications

Ever held down a key while using Disk Manager 1000 and whizzed through several menu screens? This somewhat over speedy action can be slowed down with the following modification.

Search for hex string 0603 16F9 0380 00A0 FF00 C01D in the MGR1 file. Change the code 00A0 to a larger value. Acceptable values range between 00A0 and 07D0 with the larger values producing the slower key repeat action.

If you copy MGR1 to a blank disk then you can locate the 00A0 code as follows :-

For version 2.3	Sector 53	Bytes 16 and 17
For version 3.1	Sector 36	Bytes 42 and 43
For version 3.3	Sector 54	Bytes 84 and 85

* * *

To speed up loading Infocom games, don't use Extended Basic. Use Minimemory or Editor / Assembler instead. To use these, select the load and run option and type DSK1.BOOT. Who this is finished loading, press ENTER until you get the program name, then type START.

On the Minimemory, you will get an error after BOOT loads, but keep pushing enter and proceed as above.

(From Houston Users Group) o

DID YOU EVER WONDER WHY?

The Console Tester

A device to locate problems in TI99/4A consoles.

by Geoff Trott
Illawarra Regional Group - TIsHUG
20 Robsons Road,
Keiraville, NSW, 2500
Australia

Background

The TI99/4A is quite a sophisticated computer which relies on a number of parts for its correct operation. All microprocessors require a programme in ROM to be accessible upon power up. This is called the monitor programme and for most computers is the only programme which runs the computer until something like BASIC is started. This monitor programme is written in assembler and is run directly by the processor. In the TI99/4A there is a monitor programme in ROM but its main function is to provide an interpreter for another language called GPL (Graphics Programming Language), which is the language in which the operating system of the computer is written. This GPL interpreter expects to find the GPL commands in another sort of read only memory called GROM. The contents of GROM can only be read in such a way that a GROM cannot be used to store assembler language which is to be executed directly by the processor from that GROM. Thus for the correct operation of the computer, both ROM and GROM must be operating correctly.

The 9900 processor requires some RAM for its registers and other system constants. There are 256 bytes of RAM provided in the console called system RAM. For most of the other storage requirements of the computer, VDP RAM is used. This is 16K of dynamic RAM which also is used by the Video Processor chip to contain the information required for the screen display. Once again this memory cannot be used to store assembler language programmes to be executed directly by the processor, whereas the System RAM can be.

If there is a problem with one of the major parts of the console, it is quite difficult to determine which one is at fault because of all the interactions between them. If the problem is only in VDP RAM the computer will usually start up and produce a recognisable title screen. This is because the VDP RAM is made up of 8 ICs, one for each bit in every byte. If one IC is faulty it only affects 1 bit of each byte, so there are 7 bits correct and a recognisable screen results. If there is a black screen on start up however, the problem could be due to any one of the following being faulty:-

Video Modulator
Video Processor
CPU - TMS9900
System ROM
System RAM
GROM
A large number of other components and ICs

For these reasons it was considered necessary to have a way of determining which parts of the computer were working, and even to pinpoint the actual faulty part. The easiest way to do this is by using the computer itself, but if it was not even giving a title screen this would seem to be impossible.

The LOAD Interrupt

The solution to this dilemma lies in using the LOAD interrupt to start another programme running in hardware external to the console, but using the processor in the console, to check out the System RAM, VDP RAM and hence VDP processor, System ROM and GROM. This must be done without relying on the screen display, but using it if it is working to give more information than would be otherwise be possible. What is this LOAD interrupt?

All microprocessors have a RESET input for power up or panic restarts. The TMS9900 has a RESET which is used for this purpose on power up and whenever a cartridge is pushed into the cartridge port. RESET causes the processor to do an interrupt sequence through addresses 0 to 3 and thus to enter the System ROM and produce the title screen. The TMS9900 has another interrupt input like RESET called LOAD, not normally used in the 99/4A, which causes the processor to do an interrupt sequence through addresses 65532 to 65535, at the top of memory expansion. This LOAD signal is very like the non-maskable interrupt of other processors. LOAD is not very useful normally as one cannot rely on a programme and its vectors to be present in these locations of expansion RAM. However, if a diagnostic programme is put into EPROM with the vectors at these addresses, and some RAM was made available also, then the LOAD signal could be used to start this programme executing regardless of the state of all but the processor in the console. All that would then be necessary would be some indicators to show any errors found, in case the screen display does not work.

The Hardware

The hardware is quite simple, consisting of an EPROM containing the program and the vectors and occupying the last 8K of the expansion memory address space, a RAM chip in the next to last 8K of memory (up to 8K bytes in size), a push button and circuitry for the LOAD signal, address decoding for the EPROM and RAM, and an 8 bit latch which is enabled by a write to any EPROM address. The output of six of the bits of the latch are connected to 3 red LEDs and 3 green LEDs. There is a 44 way edge connector on the printed circuit board and this plugs into the I/O port on the console, and uses the +5 volt supply from the console for power to the board. If a console was in trouble, any internal memory expansion would need to be removed before this device is attached to ensure no address conflicts.

Operational Procedure

The diagnostic board is plugged into the I/O port of a "dead" console and the power turned on. The LOAD button is pressed. This starts the diagnostic programme, and if the processor is working the three red LEDs turn on and the three green LEDs are off. The first red LED starts blinking to show that the System RAM is being tested. This test is done by writing a pattern into the entire memory and then reading the entire memory checking for any errors, swapping the bits in the pattern, and doing it again. After doing this 100 times, if there are no errors the first green LED turns on, and the blinking red LED is off. If there are errors, the green LED stays off and the formerly blinking red LED remains on.

A similar test is then run on the VDP RAM 21 times while the middle red LED blinks. Since the VDP RAM is attached to the Video processor, some of its functions are also checked. At the end of this test the middle green LED will come on if there are no errors while the red one will remain on if there are errors.

If all is OK so far, the programme sets up the VDP RAM and processor with a green background and black foreground and a character set of 256 characters, using the 128 TI-Writer characters repeated once with a red background colour for those codes between 128 and 255. All the characters appear on the bottom of the screen in reverse numerical order, taking up the last 8 lines of the screen. Then the rest of the screen appears with a heading in the first two lines followed by diagnostic information. There is a message about the System RAM and another about the VDP RAM, which will mirror the state of the first 2 pairs of LEDs. Then the checksum of the System ROMs and the 3 GROMs in the console and 2 GROMs which may be in a cartridge in the cartridge port are calculated and displayed as they are calculated. They are displayed using the full character set and so the actual value can be determined if required. The programme loops around calculating the checksums indefinitely, with the third red LED blinking as it does so. Interrupts are enabled for one instruction at the end of the loop. If the System ROM is faulty the diagnostic may not loop, as the interrupt service routine is in System ROM.

The Checksums and Error Codes

The checksums are calculated by adding the bytes of whichever ROM is being checked. In the case of the System ROMs there are two of them, one for the High or even address byte, and the other for the Low or odd address byte. The checksum is calculated for each of these ROMs and displayed side by side, using 3 characters for each. For the GROMs the bytes of each are added together to produce the checksum. Each checksum requires 3 bytes and is displayed as 3 characters using the character set displayed on the bottom of the screen. The values of the checksums of some ROMs and GROMs are:

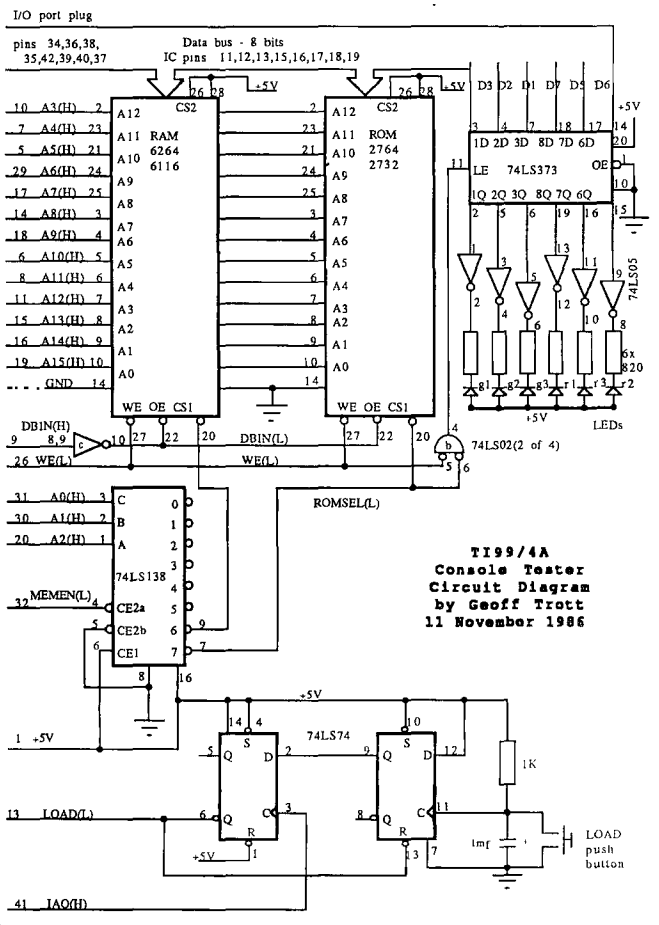
System ROM High byte >048181
System ROM Low byte >05FC8C
System GROM
0 >075574 (1981) >0731D7 (V2.2)
System GROM 1 >091B99 (BASIC)
System GROM 2 >089ADC (BASIC)

The codes for the checksums and RAM errors can be worked out from the characters on the bottom of the screen. If a character has a green background, its hexadecimal value is less than >80. If it has a red background, its hexadecimal value is greater than >80. If it is a recognisable character, its value can be determined using the ASCII code, or by counting characters on the screen. For example, the characters in the last column starting from the top would have the values >E0, >C0, >A0, >80, >60, >40, >20, >00. The character "U" with a green background would have the value of >55, while with a red background it would have the value >D5.

Interpretation of Results

If the System RAM has errors, then two error code characters will appear on the screen alongside the message. If these characters are decoded into hexadecimal and then into binary, they show in which of the 16 bits errors have been found. Since this RAM uses two chips, one for the most significant 8 bits, and the other for the least significant 8 bits, the chip which should be changed can be determined. The most significant bits are in the left character and a 1 indicates an error, while a 0 indicates no error in the bit.

If the VDP RAM has errors, a single character is output. This can be decoded as before to determine which bit of the memory has errors. For this memory, each bit is contained in a single chip so the code allows the chip in error to be determined and replaced.



The checksums given are the only ones found so far, but there have been other versions of things like system ROMs, or so we believe. The most frequent problem encountered is the failure of a system ROM, the low byte ROM. A guide could be that if one of a pair of checksums is as expected, but the other is not, then there is probably an error.

If there are errors in VDP RAM and GROMs, then check the -5 volt supply. If the tester does not start flashing when activated, it may be the processor, power supply, address buffers between the processor and the I/O port or the data buffers in the 8 to 16 line interface area. Looking at various signals around these areas with an oscilloscope is the only way to determine which of these is the problem. ©

Editor's n[illegible]. The success, that John Paine talks of in TE[illegible]IME, is largely due to this Console Tester.

Prior to this development, troublesome consoles could take many hours to diagnose the fault.

A number of these testers have been shipped around Australia and are proving their worth.

Geoff is currently in Canada on sebatical leave. He has taken a number of testers with him to demonstrate with several User Groups with whom he hopes to make contact.

* * *

```
*************************************************
*                                               *
*  CREATING RLE SCREENS FROM EXTENDED BASIC     *
*                                               *
*************************************************
```

by Arto Heino

How many times have you made a great picture created in EXTENDED BASIC and wished you could add a few more frills, but run out of chars to define!!

Using this ASSEMBLY LINK and an XB MERGE subroutine you can now use ALL those SCREENS you did when you still had a CASSETTE and nothing else.

The routine "BINARY" is used mainly to speed up screen I/O by reading one screen row of 32 chars and putting the dot patterns into 8 strings. The first string argument represents the top row of pixels of the line read. The reason I used this method is so the routine could be used for other programmes that need quick pixel screen access.

To use this routine you need XB & 32K & DISKDIVE.
Save the SOURCE code as "DSK1.WRITERLE/S"
OBJECT code as "DSK1.WRITERLE/O"
MERGE file as "DSK1.WRITERLE"

```
*************************************************************
*    CALL LINK("BINARY",Y,L$())  by Arto Heino 1987         *
*************************************************************
        DEF  BINARY
NUMASG  EQU  >2008
NUMREF  EQU  >200C
STRASG  EQU  >2010
XMLLNK  EQU  >2018
VMBR    EQU  >202C
FAC     EQU  >834A
STATUS  EQU  >837C
GPLWS   EQU  >83E0
SAVRTN  DATA >0000              XB RETURN ADRESS STORED
MYWS    BSS  32                 YOUR WORKSPACE
BIN     TEXT '0000000100100011' BINARY REPRESENTATION
        TEXT '0100010101100111' OF HEX CODES
        TEXT '1000100110101011' 0123456789ABCDEF
        TEXT '1100110111101111'
CHR     BSS  32                 CHARS FOR 1 LINE
CHRPAT  BSS  256                CHARPATTERNS FOR 1 LINE
CBIN1   BSS  256                PIXEL LINE 1
CBIN2   BSS  256                PIXEL LINE 2
CBIN3   BSS  256                PIXEL LINE 3
CBIN4   BSS  256                PIXEL LINE 4
CBIN5   BSS  256                PIXEL LINE 5
CBIN6   BSS  256                PIXEL LINE 6
CBIN7   BSS  256                PIXEL LINE 7
CBIN8   BSS  256                PIXEL LINE 8
PASS    BYTE >FF                STRING LENGTH FOR STRASG
PASS1   BSS  255                STRING LOCATION FOR STRASG
        EVEN
*-------------------------------------------------------------
BINARY  MOV  R11,@SAVRTN        SAVE RETURN ADRESS
        LWPI MYWS               LOAD YOUR WORKSACE
        CLR  R0                 NOT AN ARRAY
        LI   R1,1               FIRST ARGUMENT
        BLWP @NUMREF            GET VALUE INTO FAC
        BLWP @XMLLNK            CONVERT VALUE
        DATA >12B8               TO INTEGER
        MOV  @FAC,R0            MOVE VALUE TO R0
        CI   R0,1               IS R0<1
        JLT  OUT                YES ? GO OUT
        CI   R0,24              IS R0>24
        JGT  OUT                YES ? GO OUT
        JMP  CC                 NO GO ON
OUT     B    @EXIT              LONG JUMP OUT
CC      DEC  R0                 CONVERT R0 TO
        SLA  R0,5               SCREEN ADRESS
        LI   R1,CHR             CPU AREA FOR WHOLE LINE
        LI   R2,32              READ 32 BYTES
        BLWP @VMBR              READ IT
        MOV  R1,R5              R5 = CPU AREA FOR LINE
        MOV  R2,R6              R6 = NUMBER OF LOOPS
        LI   R1,CHRPAT          CPU AREA FOR CHARPATTERNS
        LI   R2,8               8 BYTES TO READ
*-------------------------------
GETPAT  CLR  R0                 RESET R0
        MOVB *R5+,R0            MOVE CHR TO R0
        SWPB R0                 MOVE TO MSB
        AI   R0,-125            STRIP OFFSET 96 & 29
        SLA  R0,3               MULTIPLY BY 8
        AI   R0,>3E8            ADD START OF VDP CHARPATS
        BLWP @VMBR              READ IT
        AI   R1,8               NEXT CPU SAVE AREA
        DEC  R6                 32 LOOPS DONE YET
        JNE  GETPAT             NO ? BACK FOR MORE
        LI   R7,CHRPAT          CPU AREA FOR CHARPATTERNS
        MOV  R7,R6                   ALSO IN R6
        LI   R9,CBIN1           CPU AREA FOR BINARY REPRS
        LI   R4,32              NUMBER OF LOOPS
        LI   R2,BIN             CPU AREA OF BINARY CODES
        CLR  R5                 RESET R5
*-------------------------------
GETBIN  CLR  R8                 RESET R8
        CLR  R3                 RESET R3
        MOVB *R7,R8             MOVE CHRPAT BYTE TO R8
        SWPB R8                 MOV TO MSB
        SRL  R8,4               DIVIDE BY 16
        SLA  R8,2               MULTIPLY BY 4
        A    R2,R8              ADD BIN TO R8 FOR 1ST PNTER
        MOVB *R7,R3             MOVE CHRPAT BYTE TO R3
        SWPB R3                 MOV TO MSB
        ANDI R3,>000F           LEAVE NYBBLE
        SLA  R3,2               MULTIPLY BY 4
        A    R2,R3              ADD BIN TO R3 FOR 2ND PNTER
*-------------------------------------------------------------
* VALUE AT 1ST PNTER   TO CBIN    10---10 01 11 01
* VALUE AT 1ST PNTER+1 TO CBIN+1  01-------  |  |
* VALUE AT 2ND PNTER   TO CBIN+2  11---------  |
* VALUE AT 2ND PNTER+1 TO CBIN+3  01------------
*
        MOV  *R8+,*R9+          SEE TEXT
        MOV  *R8,*R9+           "
        MOV  *R3+,*R9+          "
        MOV  *R3,*R9+           "
        AI   R7,8               NEXT CHRPAT
        DEC  R4                 PIXEL LINE DONE YET
        JNE  GETBIN             NO ? GO BACK
        INC  R6                 NEXT PIXEL LINE
        MOV  R6,R7              MOV TO R7
        INC  R5                 R5+1
        LI   R4,32              RESET R4 TO 32
        CI   R5,8               ALL 8 PIXEL LINES DONE YET
        JNE  GETBIN             NO ? GO BACK
*-------------------------------
        CLR  R0                 ARRAY ELEMENT 0
        LI   R1,2               SECOND ARG
        LI   R2,PASS            STRING LENGTH ADRESS
        LI   R3,CBIN1           START OF BINARY STORED
        LI   R8,8               EIGHT ELEMENTS
MMEM    LI   R5,255             255 LOOPS
        LI   R4,PASS1           STRING RETURN AREA
MMEM1   MOVB *R3+,*R4+          MOV STORE TO STR RETURN
        DEC  R5                 DONE YET
        JNE  MMEM1              NO ? GO BACK
PUTSTR  INC  R0                 NEXT ELEMENT
        BLWP @STRASG            ASSIGN STRING TO AGRUMENT
        INC  R3                 NEXT STORE LOCATION
        DEC  R8                 ALL 8 ELEMENTS DONE YET
        JNE  MMEM               NO ? GO BACK
EXIT    LWPI GPLWS              LOAD XB REGISTERS
        MOV  @SAVRTN,R11        RECOVER RETURN ADRESS
        CLR  @STATUS            CLEAR ERRORS
        RT                      GO TO EXTENDED BASIC
        END                     END OF CODE
```

After ASSEMBLING the CODE using only option 'R',load as usual.

```
CALL INIT
CALL LOAD("DSK1.WRITERLE/O")
```

Make sure you have saved your WRITERLE file in MERGE format first. Now merge WRITERLE into your PROGRAM and add a LINE to your PROGRAM at a point you want to SAVE the SCREEN at:

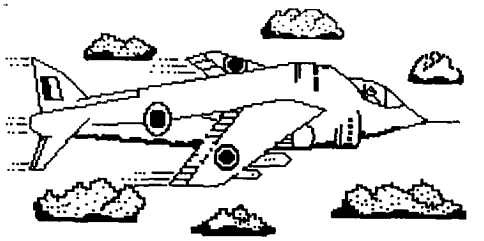
```
nnn CALL RLE("DSK1.SCREEN",1
,24,1):: END
```

This will save screen lines 1 TO 24 (whole screen) to the DISK in a format for use with your MAX/RLE DISK.

```
1 !*************************
2 ! CALL RLE(DV$,Y1,Y2,FN) *
3 !  DV$=DEVICE NAME       *
4 !  Y1 =START ROW         *
5 !  Y2 =END ROW           *
6 !  FN =FILE NUMBER       *
7 !*************************
8 !   by Arto Heino 1987   *
9 !*************************
11000 SUB RLE(NAME$,Y1,Y2,I)
:: OPEN #I:NAME$ :: PRINT #I
:CHR$(27);"G";:: FOR Y=Y1 TO
 Y2 :: CALL LINK("BINARY",Y,
L$()):: IF Y>Y1 THEN 11020
11010 IF SEG$(L$(1),1,1)="0"
 THEN B=0 :: PRINT #I:" ";EL
SE B=1
11020 FOR V=1 TO 8 :: FOR W=
1 TO 255 ::IF B<>ASC(SEG$(L$
(V),W,1))-48 THEN GOSUB 1104
0
11030 R=R+1 :: NEXT W :: R=R
+1 :: NEXT V :: NEXT Y :: GO
SUB 11040 :: PRINT #I:CHR$(2
7);"G" :: CLOSE #I :: GOTO 1
1070
11040 Z=INT(R/94):: IF Z THE
N R=R-(Z*94):: E$=RPT$("~ ",
Z):: IF R=0 THEN E$=E$&" " :
: GOTO 11060
11050 E$=E$&CHR$(R+32)
11060 PRINT #I:E$;:: E$="" :
: B=-(B=0):: R=0 :: RETURN
11070 SUBEND
```

Another use for the link routine is for displaying the char patterns in a magnified form. eg

```
1 !*************************
2 !   CHARACTER MAGNIFIER  *
3 !*************************
4 !   by Arto Heino 1987   *
5 !*************************
100 DISPLAY AT(1,1)ERASE ALL
:"ABCDEFGHIJKLMNOPQRSTUVWXYZ
{\": :" Press <S D> Keys to
View" :: CALL LINK("BINARY",
1,L$())
110 CALL CHAR(48,"FF81818181
8181FFFFFFFFFFFFFFFF"):: P
=1 :: GOTO 130
120 CALL KEY(0,K,S):: IF S=0
 THEN 120 ELSE P=P-(K=68 AND
 P<255)+(K=83 AND P>1)
130 FOR Z=8 TO 15 :: DISPLAY
 AT(Z,1):SEG$(L$(Z-7),P,28):
: NEXT Z :: GOTO 120
```



New Flight Simulator Program.

The company, NOT-POLYOPTICS, has released a Flight Simulator Program named SPAD XIII .

A demo disk has been supplied by the company which provides a graphic screen apparently from the program.

The following text is from the leaflet advertising the program.

SPAD XIII
Flight Simulator

Fly your own classic World War 1 fighter plane in Not-Polyoptics' new, all assembler language flight simulator. Set in wartime France, Spad XIII gives you all the thrills of other, better known simulators, plus features only a TI99/4A can offer!

* a full 48K program

* an all graphic world in 3D perspective

* look up, down, left, right, forward or back

* full acrobatic control of the airplane with algorithms that mimic all of the physics of flight

* continuous instrument readouts with graphic throttle and stick controls

* joystick or keyboard inputs; accepts two commands at once

* scenery includes: Eiffel Tower, Seine River, trenches, French villages, clouds; more!

* engage enemy planes in deadly dog- fights

* down enemy obersavation balloons in flames or bomb enemy hangers; but watch out for the flak!

SPAD XIII

Requires: TI99/4A, Extended Basic cartridge, 32K Memory, Disk.

Retail cost: $29.95 US funds from:

Not-Polyoptics,
PO Box 4443,
Woodbridge, VA 22191.
USA.
Phone 0011-703-491 5543

The ext'd basic demo disk program has been reformatted into an extended basic program which is small enough to download via the download feature of the TEXPAC BBS. These program is named SPAD_XIII .

If anyone gets a copy of the program please let all users of the BBS know what you think of it.

SENDING MAIL ON TEXPAC BBS.

by Ross Mudie, SYSOP, February, 1987.

A TEXPAC BBS user can send electronic mail to another user by selecting Main Menu option 3. The mail is addressed to User Names which appear in the list on the file BBS_USERS.

The mail may be sent manually, direct from the user's keyboard or from a previously prepared DIS/VAR text file. There is an extended basic program in the program download menu which will mail to be sent from a disk file.

SENDING MAIL.

Enter the mail option using 3 from the main menu, the BBS will advise how many sectors are free on the mail disk. If you are sending a file ensure that your file is smaller than the free space.

After you enter the addressee user name the BBS will validate your entry and a file will be opened for the addressee in the "append" mode. This means that if there is already other mail for your addressee then your mail will be added on to the end of the existing file. You will then receive the same date and time stamped header as is placed on the addressee's file and the > prompt will then be received. After the > is received you may proceed to send your electronic mail from your keyboard or disk file.

SENDING FROM THE KEYBOARD.

Upper and lower case may be used in addition to numbers and symbols. Commas and quotes, (once a no no on this BBS), may be used freely.

If you make a mistake then <ctrl> H will back space in the same line, wiping out all that you back space over. If FAST-TERM is in use then <fctn) 1 and <fctn> S will also perform a destructive back space.

Check that the line you have typed is correct BEFORE pressing <enter>. Once <enter> is pressed the line of text is placed automatically in the disk buffer before the > prompt is received again.

Lines of mail may be up to 79 charact- ers in length, a warning bell sounding at 72 characters. Any characters typed after the 79th in a line will be lost.

If you want to enter a blank line then press the space bar at the start of the line then press <enter>.

To exit mail press <enter> at the start of a new line and you will be prompted:
[S]ave or [C]ontinue.

C will let you continue where you left off whilst S (or just <enter> will finish saving the file to disk and close the file.

The word END on its own in upper case at the start of a line will exit to the [S]ave or [C]ontinue prompt.

SENDING A FILE TO MAIL.

This is best done using a cartridge expander with TE2 and Extended Basic plugged in. Save your SENDMAIL program on the disk under the name LOAD. Using TE2 tell the BBS who the mail is for and wait for the > prompt after validation and header. Exit TE2 with <ctrl> 0 (to close any open disk files) and switch to extended basic. Load and run the SENDMAIL program which then for the file name to be sent.

It is not practical to use a program in console basic since commas and quotes can not be used in text with INPUT.

After entering the file name and specifying to which RS232 the modem is connected the SENDMAIL program will show you what is being sent. On completion exit with BYE and switch back to TE2.

When back in terminal mode press <enter> ONCE only and the [S]ave or [C]ontinue prompt will be received.

If you want to experiment with sending mail then address your test mail to yourself. You can then read the test mail without having to hang up and call again by simply pressing 9 when in the main menu. This special feature, (which is not shown in the menu), allows you to log on again. After giving your user # and password you will immediately receive your own test mail.

Please consider the needs of other users and refrain from completely filling the mail disk if the free space is small.

RECEIVING MAIL, THE WHOOPS FUNCTION.

If you receive mail but do not manage to capture it on your printer or disk, or just want to make sure you have saved it sucessfully before it is deleted, then at the end of the mail when you receive the prompt:

Press ENTER to continue >

press E then <enter> . After checking use 9 from the main menu to log on again for another go and when all is OK allow deletion of the mail by just pressing <enter> at the prompt.

Please remember that uncleared mail causes unnecessary congestion on the mail disk.

If, for some obscure reason, you want some of your own mail to another user deleted then the only way is to leave E-Mail for the SYSOP stating the date that the mail was sent and the addressee user name. Please note that this is a particularly time consuming process for the SYSOP and should only be requested as a last resort.

o

FORUM

I have a modem which I bought from Peter Schubert and software to run on the club BBS and VIATEL. These all work but I can't get it to run with the Westpac online banking system. I have tried their technical section but they have not been able to help me. Can you help?
Les Andrews, Waterloo.

Your system (modem and software) would have to be tested before a meaningful answer could be given. Your problems may however be the result of software incompatability. The PRESTEL based systems used in Australia do not all work in exactly the same way. Some require all aspects of your transmission format to exactly match the host system while others are content with just baud rate, data bit and stop bit equality. If you are able to generate the host system logo the problem may well be one of parity which I suspect is hard coded into your programmes.

Do you know if it is possible to convert TI99/4A data files to IBM format so I can convert them to dBase III without having to retype? Of course even having achieved that I still have the problem of making the file dBase III compatible. Any Clues?
John Kerr, Aldgate.

HELP! Systems Analyst Wanted.

o

FORUM

Can I upload a file to the BBS? I know some of the large commercial systems allow this and I was wondering if it could be done on TEXPAC. Joseph P.

If the file is an ASCII file, you have two choices. The first is to use a programme called SENDMAIL which was in the download programme section of the BBS in February. The other is to use the upload facility of FAST-TERM. (I am not able at this stage to achieve a similar successful transfer with MASS TRANSFER.) By following the steps below, you will be able to implement a limited form of "scripting" which will allow you to send messages to more than one person from the one file.

As the first step, you must create your text file. Use the Editor from the Editor/Assembler. Your line length must be no longer than 79 characters - I suggest you set the right margin at 75.

The following example should make the file creation clear. I have shown line numbers in the example (don't type them in your file).

```
001 SYSOP
002 Dear Ross,
003 Start text of message for Ross.
...
...
022 Regards....Joseph
023 END
024 Y
025 SHOP
026 Dear Cyril,
027 Start text of message for Cyril.
...
...
035 Regards....Joseph
036 END
037 N
```

NOTE that the Texpac BBS conventions have been observed. Usernames and the commands (END, Y, and N) are in upper case.

Line 001 contains only the user name of the person to receive the first message. Lines 002 to 022 are the body of the message. Line 023 is the COMMAND to end the first message (on a line of its own) and line 024 is the COMMAND to open another file to accept the second message. This process is repeated for the second message except that the last line contains the COMMAND "N" which tells the system to close the message files.

Now that the file has been created, it must be PRINTED to disk with the PRINT FILE command. You can name the file whatever you like e.g. DSK1.SESSION.

If you haven't already done so, set up your parameter file so that there is NO line feed supplied with each carriage return. Log onto the BBS with FAST-TERM.

Select Option 3 to send mail and wait for the prompt for the recipient's Username. Now use the correct function key combinations to select your file (DSK1.SESSION) and to TRANSMIT LINE BY LINE. This last step is absolutely essential because the BBS does not operate quickly enough to let you upload the whole file in one hit. MAKE SURE YOU HAVE THE CORRECT PROMPT ">" on your screen before pressing the space bar to send subsequent lines to the system.

After receiving the N command the BBS will start to close the mail files. Press the space bar once more! This displays the message "End of File" on your screen and takes you out of the upload mode of the programme. You are now once again in normal terminal mode with the BBS.

If you are an experienced BBS user and have uploaded files previously you should have no problems by following these instructions to the letter. If you are unsure what to do or have only limited BBS experience, please make contact with a regional group in the fist instance.

Illawarra Regional Group

Coordinators Report

The Illawarra Regional Group held 11 regular monthly meetings in 1986, in a Church hall whose rental was paid by the local members. We had 32 families take part in our meetings, and enjoy the facilities offered. The group has gathered together useful items of hardware, modules (many donated by members), useful and interesting books, and software on disk and cassette which are available for loan to members.

At each meeting Extended BASIC tutorials were held, before a talk and demonstration on another topic of special interest. Topics covered included LOGO, FORTH, Assembler, Hardware, Speech, and Music. We also had a picnic and a fun night for Christmas.

People in the group have a large range of interests and expertise. A group is interested in the hardware, and has produced a console tester which has enabled 15 or so consoles to be repaired in our area. Many members have been helped and encouraged to install 32K memory expansions, and work is progressing on other expansions. Another group has been working on the software to allow members with memory expansion to enjoy the benefits of running assembler language programmes from cassette with Extended BASIC.

In conclusion, we have always encouraged our members to support TIsHUG with less than spectacular success. As was noted by the Coordinator Fred Morris during the year, our members have made considerable contributions to the News Digest, which we hope you all enjoyed reading. We also wrote to the executive of TIsHUG towards the end of last year with our thoughts on how to improve the relations between the centralised TIsHUG and the members out in the regional groups and so make membership of TIsHUG more attractive. This lead to some vigorous discussions but no actions. We wish the new committee well in the rather uncertain times ahead.

Jenny's YOUNGER SET
under 18's page

When we tried to contact Jenny for her contribution to this month's magazine, we were very distressed to learn that she had been admitted to Wollongong Hospital. Apparently she had been feeling very depressed lately because there hasn't been any mail from the younger set members. Perhaps the best way to cheer her up is to send in your contributions for next month's magazine NOW!!

Edited from.... NORTHEAST TARRANT HOME COMPUTER USER'S GROUP, HURST, TI. January, 1986.

THE MUSIC CORNER
BY GATLIN

Whenever I think about programming music, I think of the time it takes to key in the CALL SOUND statements. My next concern is which programming trick to use to make the desired song 'sing' through the computer instead of spit and hiccup its way through. So far I have encountered 4 distinctive methods: 1) The CALL SOUND method, 2) the DATA method, 3) the GOSUB method, and 4) the ARRAY method.

CALL SOUND is simple: A series of CALL SOUND statements each with different information, such as :

```
CALL SOUND(800,10,220,10,370,10)
CALL SOUND(800,165,5,247,5,415,5)
CALL SOUND(1600,110,0,227,0,440,0)
```

This produces smooth, precise sounds, but gets very tiresome for the nontypist. My first variation was setting up variables for all the notes to avoid constantly referring to the manual for the appropriate frequency. It worked, but didn't save any typing time or memory.

The DATA method involves setting up DATA statements which contain the notes, a READ statement to assign the notes to variables, a single CALL SOUND statement with variables read from data, and a clever FOR ... TO ... NEXT statement.
Example :

```
100 FOR REP=1 TO 4
110 READ A,B,C
120 CALL SOUND(400,A,0,B,0,C,0)
130 NEXT REP
140 RESTORE
150 GOTO 100
DATA 100,40000,40000,139,330,400,
147,40000,40000,165,277,440
```

The notes with frequencies of 40000 are used to create silence without having to key in a separate CALL SOUND statement with only one note. The disadvantage of this method is when you have a program with oodles of data (lots of notes), you occasionally get hiccups. The cause lies somewhere in the BASIC language. I've been told that **BASIC** generates garbage that has to be taken out occasionally. When the garbage is dumped, the computer hiccups causing the flow of the music to be interrupted. However, it usually takes quite a bit of data or an extreme tempo (speed) to cause hiccups.

The GOSUB method was introduced to me by Gerry Myers. It involves keying in one or more CALL SOUND statements (as needed) and follow them with a RETURN statement. Once done, your programming consists of redefining the CALL SOUND variables and adding a GOSUB statement to initiate the sound. Just to make things easier, set up variables for each note within an octave (see lines 100 to 110 below). Now instead of having a variable for every note, you alter the base variable (is this making sense). For example:BF is B flat, BF*2 is B flat 1 octave high8, BF*4 is 2 octaves higher, BF*8 is 3 octaves higher an so on. BF/2 is 1 octave lower, and BF/4 is 2 octaves lower. Explanation: well there is one, its just that if I try to explain it correctly, you'll probably put down the article and grab the TV guide. Very simple: double the frequency of any note and you'll have a note one octave higher, halve the frequency of any note and you'll have a note one octave lower.

Example :

```
100 B=493.88 :: AS,BF=466.16 :: A=440.00 :: GS,AF=415.30
    :: G=392.00 :: FS,GF=369.99 :: F=349.23
110 E=329.63 :: DS,EF=311.13 :: D=293.66 :: CS,DF=277.18
    :: C=261.63 :: R=40000 :: L=250
120 FOR REP=1 TO 3
130 X=G*2 :: Y=E*2 :: Z=C :: GOSUB 500
140 X=F*2 :: Y=D*2 :: GOSUB 500 :: X=G*2 :: Y=E*2 ::
    GOSUB 500
150 Z=G/2 :: GOSUB 500 :: X=A*2 :: Y=F*2 :: Z=C :: GOSUB
    500 :: GOSUB 500
160 X=BF*2 :: Y=G*2 :: Z=G/2 :: GOSUB 500 :: X=A*2 ::
    Y=F*2 :: GOSUB 500
170 X=BF*2 :: Y=G*2 :: GOSUB 500 :: GOSUB 500 :: GOSUB
    500
180 X=A*2 :: Y=F*2 :: GOSUB 500
190 NEXT REP
200 X=G*2 :: Y=E*2 :: Z=C/2 :: GOSUB 500
499 END
500 CALL SOUND(L,X,0,Y,0,Z,0):: RETURN
510 CALL SOUND(L*7,X,0,Y,0,Z,0)::RETURN
```

In the above example, "L" is the defined length of the note. Notes longer than "L" can be lengthened with successive GOSUBs or altered in a separate SOUND statement (as in line 510). The "voices" are defined as X,Y,Z. X is the highest voice, Y is the middle, and Z is the bass. Although it is not necessary to keep them in that order, it does help make the editing of mistakes easier. Once a voice is defined, it will remain until you change it. This causes the illusion of sustained notes behind moving notes (lines 130, 140). Once again, 40000 can be used for silence (defined as R but not used in the example).

The last method is the ARRAY method. It is similar to DATA in that all the notes are in DATA statements. However, instead of reading each voice and then playing them, the notes are read into ARRAYS, then played via a FOR .. TO .. NEXT statement. Even better, each array can be a musical line. The melody can be one array, counter melody in another, and bass line in a third.

Combining this method with a negative "duration" within the SOUND statement creates a remarkably smooth and incredibly fast musical line. When a negative duration is specified, the previous sound is stopped and the new sound is started immediately. The first question that comes to mind is how do you use negative values without getting ridiculously fast music ? EASY ! Just put some sort of delay between the SOUND statements. In the Bach Invention example, I've used a math unction that I saw used in a program by Robert Gagle. The statement "P=2^ 50 " causes the computer to think for a few extra milliseconds before it plays the next sound. A higher number than fifty creates a longer delay and thus a lower number creates a shorter delay. Why? It's a mystery to me but it works like a charm so I don't complain. (To get an idea of how fast the 4A can play, change line 240 to read "for N=1 TO 104 :: CALL SOUND(X,A(N),V1,B(N),V2):: NEXT N", remove line 270 and run the program.)

Once you've completed your data statements, you can write the data to a disk file to conserve program memory. This can allow you to create programs that execute extremely long songs without running out of memory while in the middle of programming (it happened to me, really !).

In the example program, I've added the option of changing the volume of either voice while the program is running and without sacrificing the smoothness of execution (well, maybe a little, occasionally). Hope you enjoy the program, and I hope this article has helped someone.

Program on page 22

THE MATCHBOX TESTER.
by Ben Takach

One is forced sooner or later to do some continuity testing on wire harnesses or on components. Whilst the multimeter is not an expensive test gear, it is not really a wise investment for those who do not have a constant need for it. Also the number of measuring range may be confusing for the non technical minded. I used an inexpensive home made matchbox tester, which is often more useful than an expensive multimeter. It will test continuity. It is most frustrating to remove the shells of a multi pin plug to visually observe the wire terminations for shorts or open circuit!

It will test diodes for polarity, short or open junction. It will also be useful to determine the type and polarity of unknown transistors.

It is safe to use on any diodes or transistors. It will cost about 30 cents plus 2 AAA batteries. You can build it in any suitable small plastic container (mine is in a discarded plastic core of a cash register printing roll). The electronic parts consist of a 100-150 ohm 1/10W resistor a red LED indicator lamp and 2 test leads. The device works with 3 volts at approx. 10 mA.

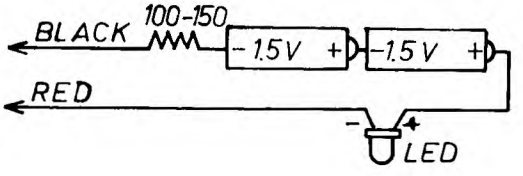


Fig.1. The tester will comfortably fit in a plastic container of the size of a matchbox.

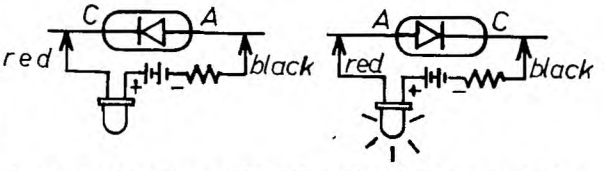


Fig.2. Diode testing. The LED will only be on when the pos. lead of the tester is on the anode terminal. If the LED is on both ways then the diode is shorted, if it does not light at all then its junction is open.

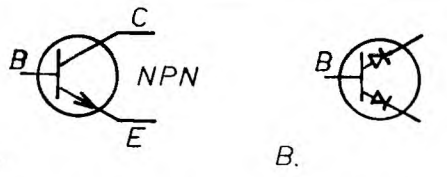


Fig.3. Transistors are like two diodes joined by their anodes or cathodes. Fig.3a shows an NPN transistor, 3b is its equivalent for testing with the LED tester. The arrow of the emitter points inwards if the transistor is a PNP type, and the diodes are inverted in its test equivalent.

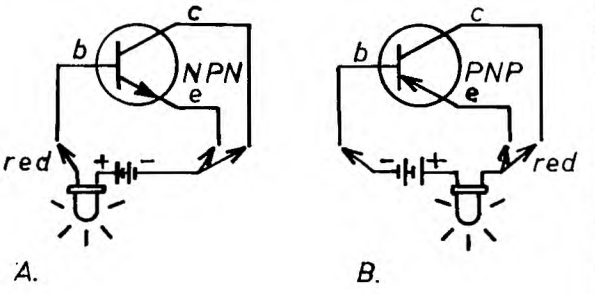


Fig.4. Testing a transistor. A. NPN transistors. The LED will only be on when the pos. lead is on the base and the neg. lead is connected to the emitter or the collector . B. PNP transistors are the reverse of NPN. The tester will show the type and the base terminal.


```
 10 ! THIS PROGRAM USES
 20 !ONE SOUND STATEMENT!!!
 30 !AND TWO SIMPLE ARRAYS!!
 40 !
 50 !PROGRAMMED BY
 60 !JEFF GATLIN
 100 CALL CLEAR
 110 PRINT TAB(5):"INVENTION
NO.13"
 120 PRINT
 130 PRINT TAB(7):"BY BACH"
 140 FOR T=1 TO 5 :: PRINT ::
  NEXT T
 150 PRINT "PROGRAMMED BY JEF
F GATLIN"
 160 PRINT
 170 PRINT "CONTROL VOLUME OF
  VOICES     USING '1'&'2' FOR
  LOUDER     AND 'Q'&'W' FOR S
OFTER"
 180 DIM A(104):: DIM B(104)
 190 FOR N=1 TO 104 :: READ A
(N):: NEXT N
 200 FOR N=1 TO 104 :: READ B
(N):: NEXT N
 210 X=-999 :: V1=10 :: V2=10
  :: P=0
 220 INPUT "READY! PRESS ENTE
R.":UU$ :: GOTO 240
 230 INPUT "PLAY AGAIN? PRESS
  ENTER.":UU$
 240 FOR N=1 TO 104 :: CALL S
OUND(X,A(N),V1,B(N),V2):: P=
2^50
 250 CALL KEY(0,K,S):: IF K=4
9 THEN V1=V1-1 ELSE IF K=50
THEN V2=V2-1 ELSE IF K=81 TH
EN V1=V1+1 ELSE IF K=87 THEN
  V2=V2+1
 260 IF V1<0 THEN V1=1 ELSE I
F V1>30 THEN V1=29 ELSE IF V
2>0 THEN V2=1 ELSE IF V2>38
THEN V2=29
 270 NEXT N
 280 GOTO 230
 290 DATA 40000,659,880,1047,
988,1175,1047,1047,-880,831,
831,659,659
 300 DATA 880,1047,1319,1047,
880,1047,740,880,1047,880,74
0,880,622,1047,988,880
 310 DATA 831,988,1175,988,83
1,988,587,698,831,698,587,69
8,494,698,659,587
 320 DATA 523,659,880,659,523
,659,440,523,662,523,440,523
,370,523,494,440
 330 DATA 415,415,988,988,831
,831,659,659,40000,659,880,1
047,988,659,988,1175
 340 DATA 1047,880,1047,1319,
1175,988,1175,1397,1319,1047
,1319,1568,1397,1319,1175,10
47
 350 DATA 988,1047,1175,1319,
1397,1175,1661,1175,1976,117
5,1047,1760,1397,1175,988,11
75
 360 DATA 831,988,1047,880,65
9,880,988,831,880,659,523,65
9,440,440,440,440
 370 DATA 523,523,440,440,415
,415,330,330,440,330,440,523
,494,330,494,587
 380 DATA 523,659,880,659,523
,659,440,523,370,440,523,440
,370,440,311,370
 390 DATA 330,330,415,415,494
,494,415,415,330,330,247,247
,208,208,165,165
 400 DATA 220,220,262,262,330
,330,262,262,220,220,262,262
,156,156,40000,40000
 410 DATA 40000,494,415,330,2
94,494,415,294,262,262,330,3
30,208,208,330,330
```

? ? ?

Loading Large Machine Language Files

by George Meldrum
Illawarra Regional Group

A problem occurred in our regional group for producing a software library suitable for both cassette and disk based systems. BASIC programs were no problem but many machine language programs needed an E/A module and could not be SAVEd by the cassette user.

A solution was found by producing a BASIC file that linked to the machine language file embedded behind it. With a machine code program sitting behind a BASIC header there is no problem of loading (OLD CS/DSK) or saving (SAVE CS/DSK) albeit the minimum requirements being an XB module and 32K memory to run such a program.

The only limitation of the machine code embedded BASIC program is the size of file loaded. Some machine language files can total 24K or more. Our little TI does not allow BASIC programs of that magnitude.

The solution our group came up with was to break the machine language code into two (or three) groups with BASIC headers. With the machine code partitioned in this way each section still contained familiar BASIC code which provided a connection from one file to the next. Essential of course was an assembly routine included to move each block of code back to its original position. The move routine being best placed at the tail end of the first BASIC file so as not to be overwritten in the reshuffling of code.

This is the way it works :-

1) The first BASIC file is loaded as normal into high memory. It contains three important statements :
```
CALL INIT
CALL LOAD(-31868,0,0)
RUN "CS1"     ! or "DSK1.2ND_FILE"
```
The CALL LOAD tells the computer that there is no memory expansion.

2) The second BASIC file loads when the RUN "filename" statement is executed. This time it loads into vram so as not to overwrite the first file. The second BASIC file links to the move routine which was loaded within the first BASIC file still in high memory. For example :
```
CALL LOAD(8192, HBYTE, LBYTE)
CALL LINK("MOVE")
! RUN "CS1"     (if needed)
```
Loading the address of your routine at 8192 (>2000) is cheating as it by-passes the name link routine. Because of this any name of one to six characters can be used in the link name. HBYTE and LBYTE are normally numeric values representing the entry address of the move routine.

As portion of the machine language main program still resides in vdp ram (loaded within the second BASIC file) our transfer routine needs to move this to the appropriate cpu ram location. For example a transfer program may look like :

```
      VPTR EQU >8370   ptr highest vram address
      CRAM EQU >xxxx   cpu ram address
      LEN  EQU >xxxx   length of portion of code
      VMBR EQU >202C   vram multi byte read
ENTRY1 MOV @VPTR,R0
      INC R0
      AI R0,-LEN       from vdp ram
      LI R1,CRAM       to cpu ram
      LI R2,LEN        number of bytes to move
      BLWP @VMBR
```

At this stage you could return to BASIC to load yet another file (for big programs), or continue to relocate the machine language portion loaded with the first BASIC file. Note that BASIC files in high memory always end at byte >FFE7. Finally character definitions usually need relocating and vdp registers need resetting to E/A defaults before jumping to the machine language program proper.

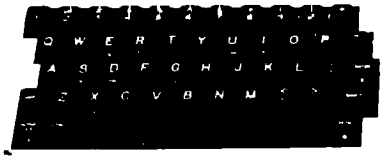
Although this method requires a fair effort to set up, once done it provides lots of advantages. Programs load in bigger chunks (than E/A opt5), no special loaders are required, and they can be loaded and saved in an environment familiar to BASIC users. Our group members, especially the cassette users, love it !

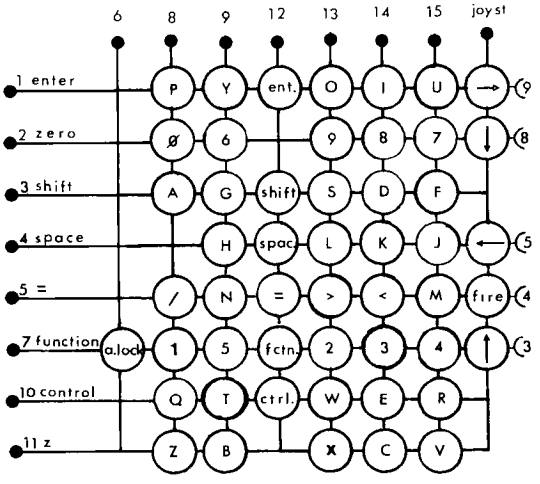
o

KEYBOARD MATRIX.
by Ben Takach

The Technical Data Manual contains much useful information. One of the notable omissions is the lack of the keyboard matrix information. Reconstruction of the matrix by tracing a keyboard could be tedious. TI has used several different keyboards made by suppliers from around the Globe, some of which are quite difficult to trace. None of the club members were able to help me with a keyboard matrix, so I mapped it myself. The matrix is reproduced below.

Terminal numbering is from right to left, viewed from the foil side of the PC board. The numbering will thus coincide with the circuit diagram published in the Technical Data Manual.





The TI-99/4A Keyboard Matrix. The keyboard is connected to the motherboard by a 15 pin plug.

o