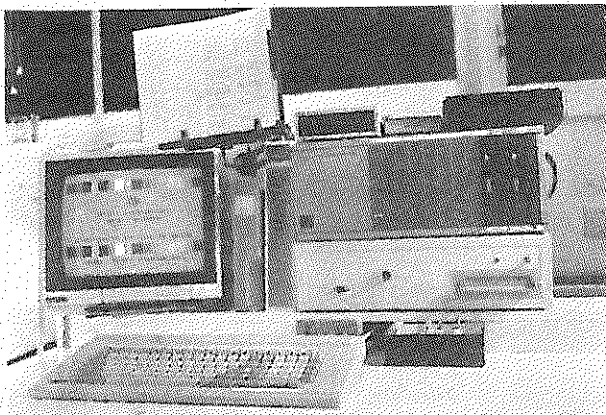


TI-Faire News Digest

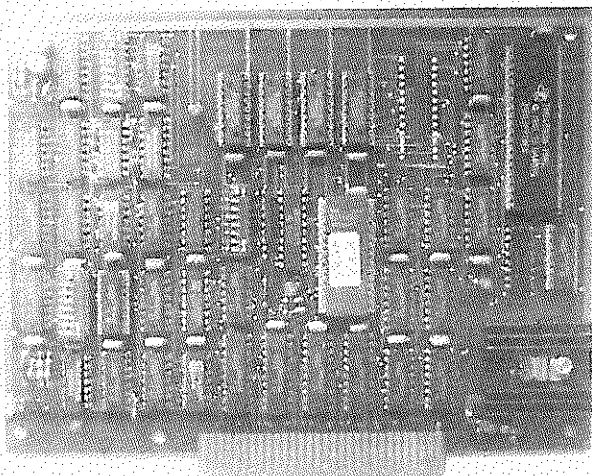
Focusing on the TI99/4A Home Computer

TI-Faire News Digest

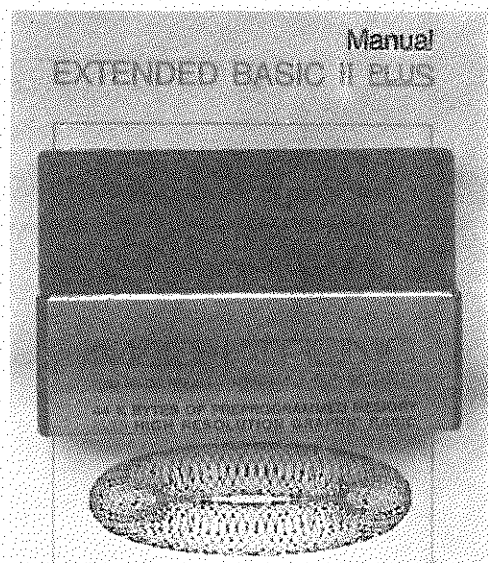
November 28 and 29, 1992



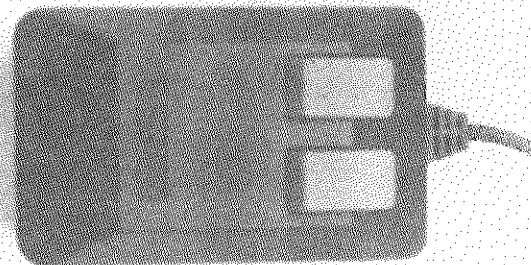
A highly modified TI99/4A!



Mechatronic GRAM card



Mechatronic XBII



Mechatronic Mouse

Sydney, New South Wales, Australia

\$3





SMILE!

There's a source
to help you make
better use of the TI99/4A
and Myarc Geneve 9640

MICROpendium, in its 9th great year,
brings you news, programming tips and
product reviews. And we've done this
every month since February 1984.

SIGN ME UP!

- I want Air mail at \$42.00 US
 Surface mail at \$30.00 US
- Charge my  
- I enclose a money order in US
funds

NAME _____

ADDRESS _____

CITY _____ STATE _____ POSTAL CODE _____

COUNTRY _____

Mail to: MICROPENDIUM
P.O. Box 1343
Round Rock, TX 78680
USA

Letter to the TI-Faire

from Jim Peterson, TigerCub Software, USA

Your invitation to be the guest speaker at your TI-Faire was the greatest honour I have ever received and I am most grateful. At the time, I had said that I would gladly accept if my health permitted. Since then, my health has deteriorated somewhat. I have emphysema, which causes me to tire very easily and have been having a series of minor complaints which would make it impractical to travel. I am therefore reluctant to go halfway around the world. Also, I do not want to leave you expecting me as your speaker when I might not be able to come. Therefore I must regretfully decline.

I wish you the greatest of success with your TI-Faire. Thanks also for continuing to send me your newsletter. It is undoubtedly by far the best in all the TI99/4A world.

You wanted a picture of the TI99/8. Lee Bendick of Newark, Ohio, has one, along with just about everything else for the TI99/4A. I just tried to call him but he has an unlisted number and I do not know his address. However, Lee demonstrated it at the Lima Fair in May and all the demonstrations were taped. They are available on three VHS video tapes for \$15, plus maybe a few dollars more for overseas postage, from the Lima TI User Group, PO Box 847, Venedocia OH 45894. I gave one of the demonstrations, but I am told that I forgot to turn on the lapel microphone so my demonstration was silent on the tape!

The TI99/4A world in this country is shrinking fast. Many user groups are down to half a dozen hard core members, others have become a TI99/4A and PC group which soon becomes a PC and TI99/4A group and then drops the TI99/4A altogether. My own group, the Central Ohio 99ers, has had to retrench drastically to keep afloat.

It is too bad, because some great advances are still being made in hardware, although little new software is coming out. I do not think much of Don O'Neill's pipe dreams and Chris Taylor's "Concept 99" but Gary Bowser has some great ideas and so does Bud Mills. Bud Wright has been digitizing photographs with a digitizing camera - that is also on the video tapes.

Personally, I do not go in for the hardware enhancements, but I did get a Midi Master 99 and have been having a lot of fun with it, even to writing a mergeable routine to convert my Extended BASIC music to a MIDI SNF file. The trouble is that inexpensive electronic keyboards with a MIDI interface are difficult to find, probably impossible to find overseas. Also, Mike Maksimik is not very good at filling orders promptly (to paraphrase Jim's comments).

Best wishes from Jim Peterson

o

continued from page 12

for administrators to assist them in finding students (and the class they are in) quickly. Once you have them sorted you can change all this to a DV 80 file (using the print option) and use FunnelWeb to make any changes. It is easy to insert new names or delete the names of students who have left. A similar thing could be done on TI BASE but I have not as yet done it this way. With TI-SORT the sorting might be faster as it took 15 minutes on Multiplan with floppies. If I had had the Mini-expansion System Ramdisk at the time it would have been a lot faster.

3. Anything that involves typing up documents I do on FunnelWeb. You can well imagine how many tests, excursion notes, etc. this would involve during the year. Last year I had a go at typing up lab practical tests and with Picasso I was able to make some good electric circuit drawings right on the computer, showing resistors, power supplies, globes and all.

continued on page 12

Contents of the Issue

by Geoff Trott and Rolf Schreiber

We decided that this special TI-Faire issue should show as many local authors as possible, so we have looked back through past issues to identify articles by TISHUG members from NSW. We only went back to the 1986 issues as we did not have a complete set before that time and we did not want to have to type in too many articles ourselves. Out of all those articles that we found, we chose the selection which you see in this News Digest. Some authors may only have contributed one article while others will have been much more active. This is not noted in this issue. There were a number of criteria used as the basis of this selection, which I will not elucidate here other than to note that we were looking for articles that would still have some interest and were not too recent, but there were a number of authors whose articles will not appear for one reason or another. In order to give them some credit, these authors will be listed here as a tribute to their contribution to the production of the TND over the years. In alphabetical order they are (and abject apologies if we have missed anyone):

Shane Andersen
J.E. Banfield
Wade Bowmer
Percy Harrison
Laurie Marsh
Bob Montgomery
Fred Morris
Terry Phillips
John Robinson
Larry Saunders
Steven Schraibman
Rolf Schreiber
Peter Schubert
Phil Thompson
Russell Welham
Warren Welham
Derek Wilkinson

I hope you find the selection presented in this News Digest interesting and that it encourages you to look through past issues of the TND for articles which are still interesting and provide stimulation for future projects. Past issues of the TND are available from the Publications Library.

XB Screen Dump

Arto Heino (December 1986, p5)

A screen dump allows the contents of the screen to be sent to a printer. This program runs from Extended BASIC and is a little slow. It should be adequate for poor people without a full system. If you want to use it as a routine, just make DV\$ your print device name (always with the .CR flag) and resequence to higher line numbers (or save it in merge format).

```
100 !XB SCREEN DUMP*
110 ! MINI UTILITY *
120 !BY ARTO HEINO *
130 ! 30/8/86 *
140 !*****
150 PRINT "RS232.BA=4800.DA=8.CR";:
ACCEPT AT(24,1)SIZE(-26)BEEP:DV$ :: OPEN #1:DV$ ::
PRINT #1:CHR$(27);"A";CHR$(8);
160 FOR X=1 TO 32 ::
PRINT #1:CHR$(27);"K";CHR$(192);CHR$(0);:
FOR Y=24 TO 1 STEP -1 :: CALL GCHAR(Y,X,C)::
C=MAX(C,32)::
IF C=CA THEN 190 ELSE CALL CHARPAT(C,CH$)
170 CA=C :: DP$="" :: FOR U=16 TO 1 STEP -2 ::
C1=ASC(SEG$(CH$,U,1)):: C2=ASC(SEG$(CH$,U-1,1))::
C1=C1+7*(C1>57) :: C2=C2+7*(C2>57) :: V=0
180 FOR I=0 TO 3 :: V=V+(C1 AND 2^I)+16*(C2 AND 2^I)::
NEXT I :: DP$=DP$&CHR$(V):: NEXT U
190 PRINT #1:DP$:: NEXT Y :: PRINT #1:CHR$(10):: NEXT X
:: CLOSE #1
```

Word Processing

Harl Davis and Geoff Trott (February and July 1986, p15 and p21)

The purpose of word processing is to produce a printed document of as good a quality as a professional typist, with a minimum of training. Typical applications include letters and papers for publication. If you do not have a printer, then it becomes difficult to produce letters or any other document. Why then would someone without a printer become interested in word processing?

As Editor of the Regional Group Newsletter, I hope that all our members would like to contribute something to our Newsletter at some time or other. I receive contributions from members in various different forms, on disk, on cassette, or on paper. To make up the Newsletter, I have to fit these into the available space. If the contribution is printed already, it may not fit in the available width, or waste some valuable space by being not wide enough. It may also have some typing errors, or require some other editing. In these cases there is no alternative other than to type it into a word processor again. If the contribution comes on cassette or on disk, then it is easier to put it into the word processor for printing in the correct format. So a printer is not necessary for preparing articles for the Newsletter. For a basic system with cassette, all that is needed is a way of entering the text of the article and saving it on cassette.

There are two ways word processors work to prepare a document. The first way is to prepare the document on the screen to look exactly how it will when it is printed. The second way is to use another programme to interpret the document into the form that is required for the printed output, using some commands in the document. TI-Writer allows both methods to be used if you have a disk based system. The advantage of the second method is that the original document can be prepared without regard to the final layout of the text, by using commands which will tell the Formatting programme how to set out the document. This means that any editor can be used to type in and correct the original document, and many have been written in BASIC and other languages for just that purpose.

When restricted to the use of cassette to save the results of hours of typing, the time taken to save and reload the data can be as long as 20 minutes. This is very tedious at both ends of the processing, so a major factor in the choice of editor for cassette systems involves how quickly the data can be stored on cassette. The fastest way to store data on cassette is in PROGRAM mode. To use this mode the system has to be fooled into thinking that this text is a programme. The easiest way of doing this is to make it into a programme by using BASIC and a REM statement at the start of each line (or ! for Extended BASIC). BASIC (especially Extended BASIC) has a very nice editor available for the entry of programmes. It provides for automatic line number generation (NUM command) and the ability to edit any existing line without having to retype the entire line. If lines need to be added the line numbers can be RESequenced afterwards.

Using BASIC to prepare documents

The documents prepared by BASIC will have to be processed by the Formatter programme to end up looking reasonable. This is because there is no direct way to indicate that the next line of text must start on a new line (end of paragraph). Also the text displayed in BASIC is only 28 characters wide and is split up by line numbers and REM or ! statements. So we need to define how to handle some of the problems which arise because of the use of this editor.

1. End of paragraph.

The end of paragraph is defined by using a special character wherever it is needed. The character chosen was }. When a } is encountered in the text, a new line will be generated.

2. Continuation of a line.

When typing in a line of text, eventually BASIC signifies that it has had enough with a beep, and the cursor stops moving. Any further characters typed in just change the character in the last position. This may happen in the middle of a word, and rather than waste time deleting back to the nearest space, the word can be continued on the next line. To signify this, the first character on the next line must be a - character. On processing, this character will not appear, and the rest of the word will be added to the end of the previous line, without a space.

3. Spacing of text.

The Formatter works by breaking lines as close to the set margins as possible. It only breaks a line at a space character, so it is important to use as many spaces as possible. In particular, make sure that punctuation marks and brackets are preceded or followed by a space.

4. Formatter commands.

Commands are identified by being on a line which starts with a period (.) in the first character position. This means that an end of paragraph character must precede the period, or it must be on the first line. More than one command can be on a line separated by semi-colons, and the line must end with an end of paragraph character. The following commands are of most use.

LM n;RM m This pair of commands allow the Left Margin and the Right Margin to be set to the values n and m respectively. These will define the width of the printing, as well as the position on the page.

FI;NF Fill puts as many words on each line as will fit without exceeding the right margin. The lines are broken at a space character. No Fill causes the lines to be printed as they appear. It turns off the Fill command.

SP n;CE m The SPace command causes n blank lines before printing the next line. CEnter causes the next m lines to be centred between the margins. If n or m are not present, the value of 1 is assumed in either case.

BP Force the printer to Begin a new Page.

LS n;PL m Line Space sets the number of lines moved to get to each line of printing to n. Page Length sets the number of lines in each page to m.

IN n INdent the first line of each paragraph n spaces. If n is absolute, the indent is to column n regardless of the margins. If n is relative (+ or -), the indent is that number of spaces about the left margin. Positive is to the right and negative is to the left.

AD;NA ADjust the text to the right margin, only when fill is turned on. No Adjust turns off the right adjust, as does NF.

TLn1:n2,...,nz TransLiterate the character with ASCII code n1 to the characters with ASCII codes n2,...,nz. This is a means of getting printed special characters, or of sending to the printer non printing characters.

There are 3 special characters which can be used to get the printer to perform special functions. & causes the printer to underline subsequent characters until the next space. To carry this on over a space to the next word requires the use of a defined space, which joins words together for filling, adjusting, underlining, and overstriking. The character used for this is ^ . For overstriking to get a darker printout use the @ character in the same way as the & for underlining.

An example of a document prepared in this way and its final appearance follow.

```

>num 100
>100 !.LM 0;RM 55;FI}
>110 !&An^Example}
>120 !.CE 1}
>130 !by Geoff Trott}
>140 !.SP;IN +5}
>150 !This is an example of a
n article prepared using Ext
ended BASIC. The first line
shows the command to provid
e automatic number generatio
>160 !-n for input. This lin
e will not be saved and is o
nly included for completenes
s.}
>170 !Line number 100 has the
commands to set the Left Ma
rgin to 0, the Right Margin
to 41, giving a column width
of 42 characters. FI cause
>180 !-s the formatter to dis
regard the line structure of
the document and use the ma
rgins as set.}
>190 !Line 110 is the heading
underlined, both words.}
>200 !Line 120 is the command
to centre the next line, li
ne 130.}
>210 !Line 140 causes the a b
lank line and all subsequent
paragraphs to be indented b
y 5 spaces.}
>220 !The rest of the documen
t follows in the subsequent
lines, with the } character
used to force an end of para
graph. When words flow over
>230 !the end of a line, a hy
phen is used to show the con
tinuation. Compare the two
outputs for the differences.
}

```

An Example

by Geoff Trott

This is an example of an article prepared using Extended BASIC. The first line shows the command to provide automatic number generation for input. This line will not be saved and is only included for completeness.

Line number 100 has the commands to set the Left Margin to 0, the Right Margin to 41, giving a column width of 42 characters. FI causes the formatter to disregard the line structure of the document and use the margins as set.

Line 110 is the heading underlined, both words.

Line 120 is the command to centre the next line, line 130.

Line 140 causes the a blank line and all subsequent paragraphs to be indented by 5 spaces.

The rest of the document follows in the subsequent lines, with the } character used to force an end of paragraph. When words flow over the end of a line, a hyphen is used to show the continuation. Compare the two outputs for the differences.

There is a program available which takes as input the listing of the original BASIC source and converts it into a TI-Writer file which can then be edited if necessary, or passed through the formatter to a printer. This program is listed below.

```

100 REM *****
110 REM * BASICWP 27/9/85 *
120 REM * G.W.TROTT *
130 REM * ILLAWARRA REGION*
140 REM * TIsHUG *
150 REM *****
160 CALL CLEAR
170 PRINT TAB(11);"BASIC-WP": : " To take the line
numbers and REM or ! from a listing, and join lines
together to";
180 PRINT "produce a file suitable for TI-WRITER
formatter."

```

```

190 PRINT : " Files are assumed to be on DSK2 unless
specified, and the output file is assumed to go
on the same disk as the input file.": :
200 S$=" " :: H$="-" :: E$="}" :: C$=CHR$(13)
210 DL=10 :: LN=100 :: LN$=STR$(LN)
220 INPUT "Filename for input ":I$ :: X=POS(I$,"DSK",1)
:: IF X=1 THEN 230 :: IF LEN(I$)=0 THEN 500 ::
I$="DSK2."&I$
230 OPEN #1:I$,INPUT ,VARIABLE 80
240 INPUT "Filename for output ":O$ ::
Y=POS(O$,"DSK",1):: IF Y=1 THEN 260 :: IF X=1 THEN
250 :: O$="DSK2."&O$ :: GOTO 260
250 O$=SEG$(I$,1,5)&O$
260 OPEN #2:O$,OUTPUT,VARIABLE 80 ::
PRINT #2:".LM 0;RM 40;FI"&C$
270 IF EOF(1)THEN 490
280 LINPUT #1:A$ :: IF LEN(A$)=80 THEN 290 :: A$=A$&S$
290 X=POS(A$,LN$,1):: IF X=0 THEN 270
300 LLN=LEN(LN$)+1 :: LN=LN+DL :: LN$=STR$(LN):: DX=1 ::
R$="!"
310 IF SEG$(A$,X+LLN,DX)=R$ THEN 330 :: DX=4 ::
R$="REM "
320 IF SEG$(A$,X+LLN,DX)=R$ THEN 330 :: PRINT : "First
line must be a comment": : GOTO 490
330 A$=SEG$(A$,X+DX+LLN,LEN(A$))
340 IF LEN(A$)>80 THEN 430 :: Y=POS(A$,E$,1)::
IF Y>0 THEN 460 :: IF EOF(1)THEN 480
350 LINPUT #1:B$ :: IF LEN(B$)=80 THEN 360 :: B$=B$&S$
360 X=POS(B$,LN$,1):: IF X=0 THEN 400
370 LLN=LEN(LN$)+1 :: LN=LN+DL :: LN$=STR$(LN)::
IF SEG$(B$,X+LLN,DX)=R$ THEN 390
380 PRINT #2:A$&C$ :: A$=B$ :: GOTO 340
390 B$=SEG$(B$,X+DX+LLN,LEN(B$))::
IF SEG$(B$,1,1)=H$ THEN 410
400 A$=A$&B$ :: GOTO 340
410 IF SEG$(A$,LEN(A$),1)<>S$ THEN 420 ::
A$=SEG$(A$,1,LEN(A$)-1):: IF LEN(A$)>0 THEN 410
420 A$=A$&SEG$(B$,2,LEN(B$)):: GOTO 340
430 FOR I=81 TO 1 STEP -1 :: IF SEG$(A$,I,1)=S$ THEN 450
440 NEXT I
450 PRINT #2:SEG$(A$,1,I-1):: A$=SEG$(A$,I+1,LEN(A$))::
GOTO 340
460 PRINT #2:SEG$(A$,1,Y-1)&C$
470 A$=SEG$(A$,Y+1,LEN(A$)):: Y=1 ::
IF POS(A$,S$,1)=1 THEN 470 ELSE 340
480 IF LEN(A$)=0 THEN 490 :: PRINT #2:A$&C$
490 CLOSE #2 :: CLOSE #1 :: PRINT RPT$(" ",28):: GOTO
200
500 STOP

```

continued from page 22

```

130 CALL LOAD(8196,63,248,"",16376,71,65,77,69,
32,32,255,156)
140 CALL LINK("GAME")

```

(10) You MUST remember to save the program now !
Type :
SAVE DSK1.filename
SAVE CS1

You can now try RUNNING the program now. Not all programs meant for the Editor/Assembler environment will work with Extended BASIC. You may edit the BASIC program with the only restriction being that you cannot RESEQUENCE (resequencing can effect the embedded machine code).

** Full system users can, if they wish, skip step D and continue with steps E, F, and 10. They can then create and assemble the following code :-

```

* Emulate E/A environment + code relocation
*
AORG >FF9C . = >FFE8 - length of this file
VMBW EQU >2024
VMBR EQU >202C
VWTR EQU >2030
START EQU >dddd enter load addr found in step 6
LOAD EQU >dddd enter load addr found in step 6
LEN EQU >nnnn enter length calculated in step 7
FROM EQU >ssss enter location found in step 8

```

continued on page 26

Greeting Cards

Alf Ruggeri (November 1990, p13)

The subject of this presentation is to show to you how to use the TI99/4A for producing greeting cards. By this demonstration Larry and I hope to stimulate those who have a little creative inclination and certainly young people. In this way you or your children, if you have any, will begin to explore a new facet of our computer through the use of the latest text graphics programs such as PAGE PRO 99.

If you have read recent newsletters you will be aware of some of the excellent features of PAGE PRO 99 as a desk top page publisher, but it is only in the most recent material that Larry has received from ASGARD that greeting cards production has been featured. After you see this demonstration we hope you will convey your sentiments to Rolf Schreiber so that he can import PAGE PRO 99, its supporting utilities and pictures for our user group in general. I certainly will encourage Rolf to import PAGE PRO as having seen Larry's copy in action, the word 'brilliant' does not do it sufficient justice.

My involvement in this presentation was in part prompted by a query from Rolf. When I last requested some MAX RLE files from Rolf about four months ago, he was curious as to what use they were being put. On my reply as graphic illustrations for greeting cards, he suggested that I bring a few sample cards to the next monthly meeting to show the rest of the members a novel use for our computer. I decided that more information on card production could be passed on through a demonstration, so with Larry's agreement to put his PAGE PRO 99 programs through their paces this presentation came about.

For my part of this presentation I will use PICASSO VN.2. This program was purchased by me in support of what was considered to be the work of a fellow TISHUG member, it is not the current version marketed by ASGARD but once again having seen Larry's imported one in action it has been vastly improved and I certainly recommend that you suggest to Rolf that it be imported also. I chose to use PICASSO for my part of the demonstration mainly because most of you will have bought a copy in the past. After I relate some background information as to what was my motivation to producing greeting cards, and what I think are important considerations I will fully demonstrate my method of card production.

Prelude to Card Production

At the beginning of this year I bought a printer that was readily capable of reproducing graphics. The printer that it replaced could produce graphics but with a lot of user effort. The new printer, an EPSON LX850, was basically bought for two reasons:

- 1) A bit more creativity in the program user instructions that I write for myself and my children.
- 2) Motivation for my year 10 son to prolong his stay at school.

At this point I do not think either of the reasons have had much realisation. My son is still in year 10 but there is no clear indication that he will be there for year 11 or 12.

In the process of wading through the printer handbook, I experimented with the concept of greeting cards mainly for the purpose of determining that the machine was able to reliably download graphics from programs available for the TI99/4A. The results were gratifying I had not bought a lemon after all. These early attempts at card production were only meant to be printer tests and nothing more, but several members of my family saw fit to suggest that cards could be made for friends' and relatives' birthdays. There was very little attempt on my family's part to disguise their

suggestion as nothing more than a prompt for me to produce something useful with the computer, for all the time I spent in front of it. Being basically a frustrated artist, that is, a draftsman, I found the artistic challenge very provocative.

It was at a regional meeting at Larry's place, that I happened to mention one of the card problems that I was currently trying to solve. Ashley Lynn came up with a simple solution borrowed in concept from similar software for IBM machines. Yes! a dynamic example of healthy cross fertilization of ideas. With this problem behind me it remained only to standardize on card formats and choice of what pictures and character fonts to use.

Some Considerations

The card is printed on one side of a single A4 sheet. It is then carefully folded over once, narrow end to narrow end with the printed material appearing on the exposed surfaces. The sheet is now carefully folded again, the new narrow end to the new narrow end with the graphics that are intended for the front and back of the card appearing on the two exposed surfaces. This folding gives the card rigidity and of course makes it suitable for postage in readily available envelopes. I prefer single sheets to fanfold paper because my printer has a very effective single sheet feeder mechanism and at the time I developed the card technique, the only coloured paper suitable for printers, that I could locate, were A4 single sheets. Larry has since sourced coloured fanfold paper. The method that I will describe deals with single A4 sheets but with a little modification fanfold paper can also be used.

At this point I should explain my choice of coloured rather than plain paper, though the reason should be fairly obvious. A card produced on coloured paper provides a more artistic flair and is far more presentable for festive and personal occasions. It is certainly more likely to receive a favourable comment than the same product on ordinary white paper which would definitely have the appearance of my early printer tests, that is, a computer hacker's output from his latest toy! Of course suitably colour matched envelopes, if available, would definitely add that aesthetic finishing touch.

A few more important points if you will bear with me, are:

- A) Those of you who have used PICASSO know that it will require a disk based system with 32K memory and the use of the Extended BASIC module. It is also recommended although not absolutely essential that you have TI-ARTIST VN.2 or TI-ARTIST PLUS mainly for the character fonts. There are some large character fonts on the PICASSO companion disks identified by a /CH suffix, they are quite good and although generating the font character string can be a bit tedious, the results are quite rewarding. There is certainly a wealth of graphics available from our RLE library disks that can be used as pictures or clip-parts not to mention the 2000 odd pictures from Ron Woolcot's TIPS program that Rolf is presently trying to source from the USA. Although not a topic of this presentation a comment on the TIPS program seems timely, it is most encouraging to note that more and more of the newer software being written is geared to further expanding the graphics creative use of our humble orphaned computer.

- B) A few considerations when choosing pictures for the cards are:

- (1) The horizontal picture size and font character string should be limited to 30 screen columns, the reason for this is that the width of the card when folded to final size will only accommodate 30 character spaces. Wider graphics and fonts can be used but only where it is unavoidable. Pictures can be trimmed to size more effectively with PICASSO in the ZOOM mode, rather than with TI-ARTIST as the former readily displays the picture elements in terms of character width.

(ii) Try to avoid pictures that have excessive areas of unbroken black. There are few dot matrix printers that can faithfully translate on screen graphics to paper printout. The offending effect is a series of regularly occurring thin horizontal stripes devoid of print. The result can be rather disappointing with a well chosen picture originally derived from a digitized photograph. The occurrence of this effect is brought about by the printer not responding effectively to the printer drive file. With some graphics programs there are separate drive files for different types of printers. Unfortunately this choice does not exist with PICASSO.

Method

The graphics on the sheet are printed in two operations, the graphics that are intended for the inside of the card are printed in an inverted form at the top of the page. This was Ashley's suggestion and certainly made production a lot easier. In retrospect, Ashley's solution was extremely simple, and I should have thought of it myself, but in the process of trying to solve many small unrelated problems simultaneously, lateral thinking took a holiday.

Using PICASSO the two sets of graphics are stored in two separate 85 sector DV80 files. This method does not utilise the text importation properties of the program but rather uses the file as a paste-up area or graphic spreadsheet where the chosen pictures and fonts can be manipulated to various locations on to what eventually will print out to an area the width and just under half the height of an A4 sheet.

You can choose for yourself which of the files is to contain the inverted graphics. I prefer to invert the graphics for the inside of the card.

Before you start locating graphics on the file, it is most important that you plan out on a sheet of paper, the same size that you will use for the card, the approximate location of your chosen graphics.

Placement of Graphics

For those of you not familiar with the PICASSO 85 sector file it represents an area of 80 screen character spaces in width by 42 screen character spaces in height. Access to this area is through a window or port 32 characters wide and 24 characters high, the normal screen size. This window accepts TI-ARTIST format graphics as a picture or as a transparent overlay. The window can be positioned over any part of the 60x42 character space by the use of joysticks. Identifying the location of the window, except on initial booting, can be a bit tricky as there are no horizontal or vertical reference coordinates that will serve to identify the horizontal or vertical boundaries. This problem is overcome by first creating a marked up template file with suitable corner and vertical mid page reference indicators. For corner markers turn on pixels in a diagonal arrow shape starting from each corner and extending one character width towards the centre of the area. The vertical middle reference, representing the the vertical folding line of the card, is produced by turning on all the pixels of the two middlemost pixel columns for the full height of the area. Once this marked up template is created it should be saved away as a utility file and used repeatedly for graphic placement. It should be saved under the name of 'TEMPLATE' so it's not confused with the card files that are going to be created. The 85 sector files are saved by pressing 'S' followed by the path and filename, they are retrieved by pressing 'G' followed by the path and filename.

We are now ready to place our graphics on the two 85 sector files. We will first deal with the inside part of the card.

Inside Part of the Card

- 1) Load the TEMPLATE file. On initial loading the access port exposes the top LHS of the 60x42 screen character area.
- 2) Using option 3 (OVERLAY GRAPHIC) from the FILE UTILITY MENU, load the graphics planned for the bottom of the inside back cover of the card.
- 3) Invert and mirror reverse the graphics.
- 4) Position the access port so that the top RHS corner symbol is visible at the top RHS corner of the screen.
- 5) Use option 3 to load the graphics planned for the bottom of the inside front cover.
- 6) Invert and mirror reverse the graphics.
- 7) Position the access port so that the bottom RHS corner symbol is visible at the bottom RHS corner of the screen.
- 8) Load the graphics planned for the top of the inside front cover with option 3.
- 9) Invert and mirror reverse the graphics.
- 10) Position the access port so that the bottom LHS corner symbol is visible at the bottom LHS corner of the screen.
- 11) Use option 3 to load the graphics planned for the top of the inside back cover.
- 12) Invert and mirror reverse the graphics.
- 13) Save this file under the name of 'CARDIN' for card inside.

Outside Part of the Card

- 1) Load the TEMPLATE file.
- 2) Position the access port so that the top RHS corner symbol is visible at the top RHS corner of the screen.
- 3) Using option 3, load the graphics planned for the top of the front cover. Remember this is the front of the card so it is not to be inverted. (a sigh of relief!)
- 4) Position the access port so that the bottom RHS corner symbol is visible at the bottom RHS corner of the screen.
- 5) Load the card's graphic font message with option 3.
- 6) Position the access port so that the bottom LHS corner symbol is visible at the bottom LHS corner of the screen, and load in, if you are inclined to a little self indulgence, a signature graphic, in this case my logo.
- 7) Save this file under the name of 'CARDOUT' for card outside.

The use of option 3 (OVERLAY GRAPHICS) rather than option 2 (LOAD GRAPHIC FILE) was necessary to avoid overprint, and therefore corruption of any existing picture or font graphics, visible within the access port window, but also to preserve the corner and vertical reference indicators, during the graphics loading procedure.

Printout Consideration

As mentioned previously, the height of the 85 sector TEMPLATE area when printed is less than half the length of either the fanfold page or the A4 sheet, so to ensure that both template printed areas are symmetrical about the middle height of the sheet, that is, its first fold, it is necessary to advance the sheet in the printer through a short distance, between printing of the first and second 85 sector files. The advancement is carried out via a simple print program in BASIC. No attempt has been made to incorporate specific line advance control characters as these vary between different brands of printers. The program is simply as follows:

```
10 OPEN #1:"PIO"
20 PRINT #1:
30 PRINT #1:
   etc
160 CLOSE #1
```

continued on page 27

Direct I/O Interface

Lou Amadio (July 1989, p5)

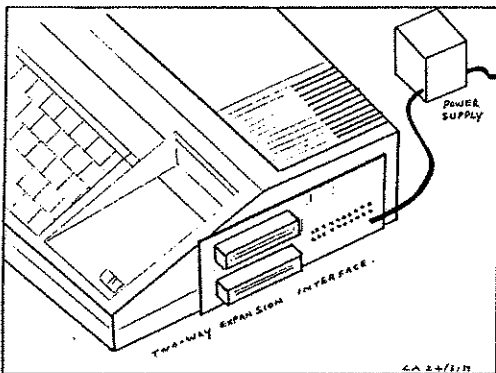
(This project is based on a prototype unit built by Geoff Trott about 3 years ago.)

With the limited availability (and high price) of second hand TI99/4A Peripheral Expansion Boxes and the fact that the Peter Schubert Mini Expansion System is no longer readily available, the time is right to outline yet another way to expand your TI99/4A. The device that I am about to describe will allow you to connect up to two TI (or equivalent) PEB cards directly to the console.

A typical configuration would be a 32K RAM and a disk controller, while an all out system could have a Peter Schubert Multi-function Card (32K, RS232, PIO and double density disk controller) with a RAMdisk. Alternatively, you could add a modem or even a PGRAM card! Note that some configurations may require a separate (internal) 32K memory expansion to work properly.

The interface has its own power supply, which is independent of the console power supply.

Because of the proximity of the PEB cards to the console, no buffer chips are required, simplifying the construction considerably.



Parts Required

- 1 x 12V, CT, 1.2A transformer
- 1 x 25V, CT, 1.2A transformer
- 2 x 3A power diodes
- 4 x 1A power diodes
- 1 x 7805 +5 volt IC regulator
- 5 x 2200 uF 25V chassis caps
- 2 x 1uF 25V tantalum capacitors
- 3 x 10uF 25V tantalum capacitors
- 3 x .1uF 25V capacitors
- 7 x 47 ohm 0.5 watt resistors
- 3 x 1000 ohm 0.5 watt resistors
- 1 x 44-way 0.1" edge connector
- 2 x 60-way 0.1" edge connector
- 2 x 60 by 17 hole veroboard
- 8 x rubber feet (7mm thick)
- 0.5A fuse and holder
- Power cable, grommet and clamp
- Aluminium box

The Power Supply

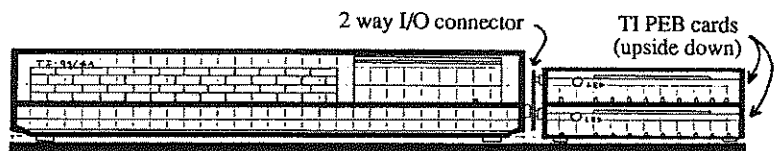
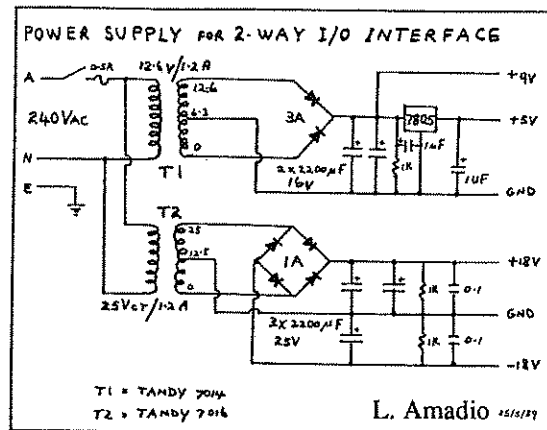
Power requirements for two PEB cards is approximately:

- +18 volts at 0.4 amps
- 18 volts at 0.1 amps
- +9 volts at 0.9 amps

Non TI cards are likely to require lower currents, attributable to the usage of modern low-power chips. In

any case, the unregulated voltages should not exceed the levels indicated above otherwise the card voltage regulators may run too hot. For this reason, metal cased TI PEB cards are preferred for this application as the clamshell provides a heatsink for the voltage regulators. Refer to the notes on how to adapt higher voltages to the task.

The circuit diagram for the power supply is shown below. It is fairly conventional and consists of two low cost transformers and a number of rectifiers and filter capacitors (two transformers were used as they were found to be cheaper than a multiple winding unit).



Front View of Console and I/O interface L. Amadio, TIsHUG 23/5/89

If you use chassis mounted types for the main filter capacitors, you can mount the diodes directly onto the capacitor lugs and use point to point wiring for input and output. This greatly simplifies construction. Use the circuit diagram to ensure correct polarity of the diodes and capacitors. The bleeder resistors across the capacitors ensure that the power supply will discharge on power off even if no cards are connected.

The 5 volt regulator is included to supply a constant voltage for a number of "pull up" resistors.

All the components are mounted in an aluminium box which is suitably earthed. The box also makes an excellent heat sink for the 5 volt regulator. Bolt the regulator directly to the box, close to the filter capacitors. There is no need to insulate the metal tab as it is at earth potential. Solder 1uF tantalum capacitors between the input and earth and the output and earth (negative side of the capacitor goes to earth).

Parallel the primary windings of the power transformers and connect them to the mains via a 3 pin plug, cable and 0.5 A fuse. A power switch is optional as most computer systems are switched off at the wall when not in use. Ensure that the mains cable is safely secured to the box using a grommet and anchor lug and that the earth lead (green) is firmly connected to the metal of the box. A cable with 6 wires (rated at 1 amp each) is required between the power supply and the I/O interface.

The Interface

The console I/O has 44 contacts (2x22) and the PEB cards have 60 contacts (2x30) each. I used wire wrap edge connectors for this project as these have longer pins which are suitable for stand-off soldering. The

connectors were cut down from longer units, as these were cheaper than buying the correct size. If you decide to make your own, cut the edge connectors for a neat fit with both the console and TI99/4A expansion cards. Remove unwanted pins by pulling them out from the wire wrap end with a pair of fine nose pliers. Chamfer the top and bottom edges of the 44-way connector to aid insertion into the console I/O port. Thoroughly clean the connector to remove any particles of plastic which may be lodged between the metal contacts.

The connectors were mounted on 2 strips of 150 mm wide (0.1 inch hole spacing) veroboard. Cut two pieces of veroboard 60 holes by 17 holes as indicated. Note the direction of the copper tracks. Place the boards back to back (copper side out) and mark the location of the edge connectors using the diagram for assistance (the boards are used back to back for added strength).

It is important that the edge connectors are spaced sufficiently apart to fit two standard PEB cards. With a hole spacing of 9 this will be a very neat fit. Use a spacing of 10 holes (that is 1 inch) if you want a little more clearance.

The vertical clearance between the 44-way and the lower 60-way connector was designed to allow for 6 mm thick rubber feet beneath the PEB card. The card is mounted upside down for convenient peripheral connection at the rear. Cards not housed in a metal (or plastic) clamshell must be supported by at least two standoffs. In this case, a cover should be made from a piece of aluminium with suitable holes drilled for the activity LEDs.

The points marked with an "x" on the "veroboard" diagram indicate where the tracks must be cut on both the top and bottom boards. Do this carefully with a 3mm bit and a hand operated drill. Run the sharp edge of screwdriver blade across the cut tracks to ensure that there are no thin copper slivers still shorting the tracks together. Place the two strips of veroboard together and fit the 3 edge connectors in place. Note that the console (44-way) edge connector is located on the opposite side to the PEB card connectors. Ensure that the tracks cut previously align with the edge connector pins. Allow approximately 5 mm gap between the back of the edge connector and the veroboard and solder only 2 pins (opposite corners) of each connector to the board on one side only. Carefully check for alignment using a set square and ruler in both the horizontal and vertical planes and realign if necessary. Correct alignment here will ensure trouble free operation later. Check for correct spacing between the 60-way connectors by plugging in two TI PEB cards. If all is well solder the rest of the pins in place on both

sides (see footnotes below). Ensure that the veroboards are kept firmly together during this operation. When finished, check with a resistance meter that no two pins are connected together.

Mount the 47 ohm pull-up resistors first to the points indicated on the table below, and then to a common point above the board which can be connected directly to the regulated +5 volt supply.

The next stage is to make the direct links between the 44-way and the bottom 60-way connectors. The 44-way connector has pin 1 on the bottom closest to the front of the console, and the 60-way connector has pin 1 on the top closest to the rear of the console. Follow the instructions in the table below and complete the connections one at the time. As there are a large number of connections to be made, it is best to use fine multistrand coloured wire. Some of the connections will be fairly close so leave a little extra length on the wire to allow for soldering. For each link in turn, measure, cut, strip and pre-tin the ends of the wire and also pre-tin the start and finish points on the board prior to soldering the link. Make the connections to the veroboard tracks adjacent to the pins rather than to the pins themselves. Taken one step at the time you will soon have finished the most difficult part of the project. When completed, double check everything with a resistance meter. It helps to have a friend check your connections while you call the start and finish points.

Connect the rest of the power supply leads to the appropriate pins as per above table. Use a suitable common point for all of the ground connections. Decouple the power connections by soldering 10uF capacitors to earth. Take care with supply polarity.

The following relate to the PEB 60-way connector:

Pins 1 and 2 to +9V unregulated.

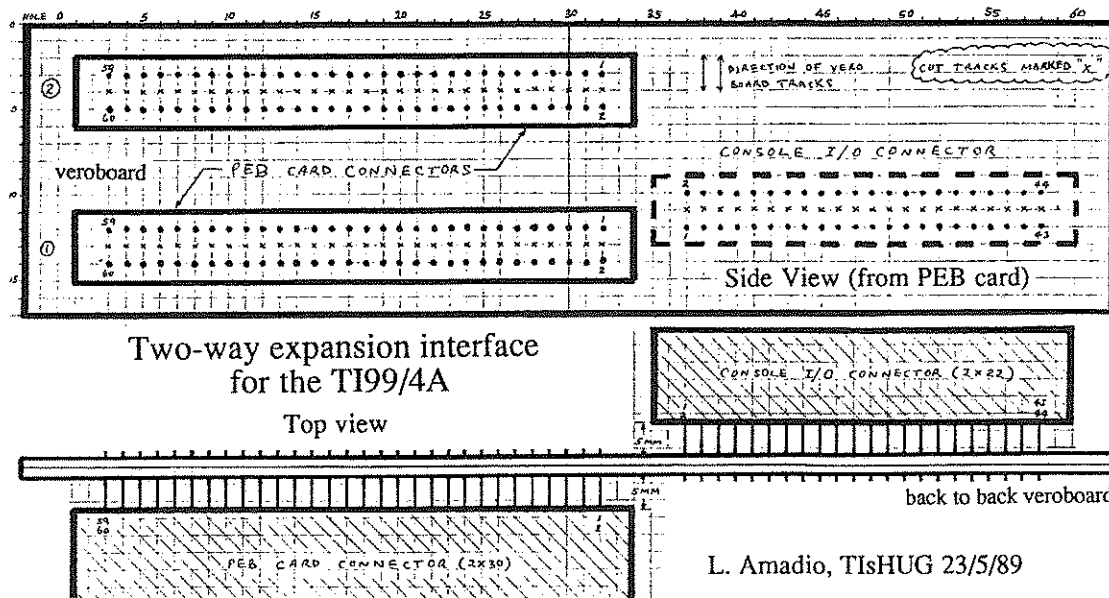
Pins 3, 5, 7, 20, 27, 47, 49, 53 connected to ground.

Pins 12, 13, 15, 16, 45, 46 and 48 have 47 ohm (0.5 watt) resistors to +5 volts regulated.

Pins 57 and 58 to -18V unregulated.

Pins 59 and 60 to +18V unregulated.

Pins 8, 9, 14, 18, 11, 12, 13, 15, 16, 45, 46 and 48 are not connected.



L. Amadio, TIsHUG 23/5/89

I/O Port to PEB Connections

I/O #	Function	PEB #	I/O #	Function	PEB #
1	+5V(speech)	NC	23	GND	7,49
2	SBE(H)	NC	24	CLKOUT(L)	50
3	RESET(L)	6	25	GND	20,53
4	EXTIN(L)	17	26	WE(L)	54
5	A5(H)	40	27	GND	3,27
6	A10(H)	33	28	MBE(L)	NC
7	A4(H)	39	29	A6(H)	37
8	A11(H)	34	30	A1(H)	44
9	DBIN(H)	52	31	A0(H)	43
10	A3(H)	42	32	MEMEN(L)	56
11	A12(H)	31	33	CRUIN(H)	55
12	READY(H)	4	34	D7(H)	19
13	LOAD(L)	18	35	D4(H)	24
14	AB(H)	35	36	D8(H)	22
15	A13(H)	32	37	D0(H)	28
16	A14(H)	29	38	D5(H)	21
17	A7(H)	38	39	D2(H)	26
18	A9(H)	36	40	D1(H)	25
19	A15/CRUOUT(H)	30	41	IAQ(H)	NC
20	A2(H)	41	42	D3(H)	23
21	GND	5,47	43	-5V(speech)	NC
22	CRUCLK(L)	51	44	AUDIOIN	10

Power Up

Note: Under no circumstances should you plug or unplug PEB cards with the power on. Allow at least one minute after switching the power off.

With the interface detached from the console, and without plugging in any PEB cards, switch on the power and check that the correct voltages are present where you expect to find them on the edge connector contacts.

- 60-Way: +9V on pins 1 and 2
- 60-Way: -18V on pins 57 and 58
- 60-Way: +18V on pins 59 and 60
- 60-Way: Ground pins 3, 5, 7, 20, 27, 47, 49 and 53
- 44-Way: No voltages present on any of the contacts when disconnected from the console

If all is well, switch off the power, connect one PEB card and plug the interface into the console I/O port. Support the card with self adhesive rubber feet. Switch on the power to the console only. The console should operate normally. If not, unplug the interface and check your wiring. (Note: Make sure that the I/O contacts are clean). If the console operates normally, switch on the interface power and check the voltages present on the pins indicated above.

Once the bottom 60-way connector is working satisfactorily, it is time to connect the two 60-way connectors in parallel. Wiring is done with direct links between the pins, ie from pin 1 of the bottom connector to pin 1 of the top connector, and so forth. Recheck as above first without, then with the PEB cards. Measure the unregulated voltages going into the cards. These should be no lower than 16 volts and 8 volts for the supplies in question.

Veroboard

60-way Veroboard may be hard to get, so a limited supply, cut to size, will be available from the shop.

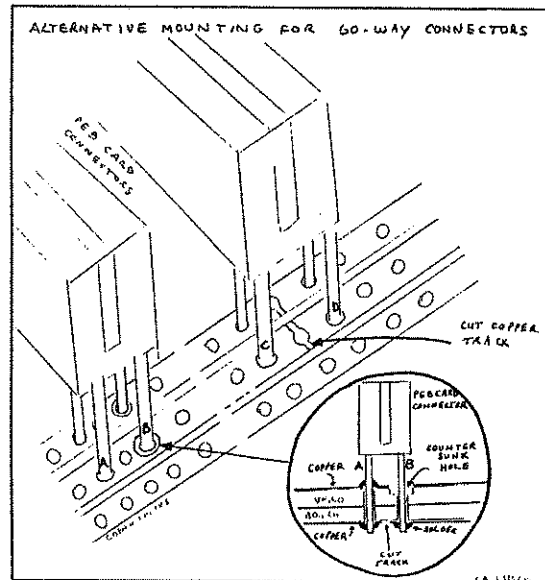
Excessive Voltages

If the unregulated voltages from your power supply exceed those indicated above, they may be reduced by inserting a 5 watt power resistor between the 2200 uF filter capacitors in either the +9 volt and/or the +18 volt supplies. Use the table below as guide:

Measured	Required	Resistor
+12V	+9V	3.3 ohm
+21V	+18V	6.8 ohm

Alternative Construction Method

During the construction stages of this project, it occurred to me that the back-to-back veroboards could be used to better advantage by utilising the available tracks to connect the two 60-way connectors in parallel. The idea is illustrated in the diagram below. What we need is a way of connecting point "A" to point "C" while at the same time bypassing pin "B". If we countersink the hole for pin "B" on the top board (without drilling all the way through) we ensure that the pin can pass through the board without making contact with the upper track. This same track is cut between pin "C" and "D"



Similarly the hole for pin "C" is countersunk on the bottom board and the track between pin "A" and pin "B" is cut. If this process is repeated for all 60 tracks, then the two edge connectors will be in parallel simply by soldering all pins except the countersunk ones. If you use this method to connect the 60-way connectors, then you must make a small change to the way that you connect the direct wire links from the 44-way connector. The difference should be obvious if you have followed the above instructions.

Final Note

As a word of encouragement, it took a lot longer to prepare and write this article than to actually build the I/O interface!

(You may also need to remove some jumpers in TI Disk controller cards and TI RS232 cards. These are located close to the edge connector and are wire links which can be cut easily. Their purpose is unclear as they are associated with interrupts which are not used by the TI99/4A, but they cause the console to lock up when used without buffers. GWT)

continued from page 13

After the Vagon Captain has tortured you with the first verse, grit your teeth and enjoy the poetry. He will then, to your dismay, read you the next verse.

While you could easily live without hearing it, in this case, you do need to listen so you know the word to type in. Fortunately, after the second verse, you do not have to enjoy the poetry. Unfortunately, since you survived both verses, the Captain is going to have you and Ford shoved out the airlock (you have now found something worse than appreciating Vagon poetry).

While Ford tries to talk the guard out of spacing the two of you, type in the word from the poem. You must put quotes around the word, or it will not go through. Then get the plotter when the case opens. Now just wait awhile, and you and Ford will soon be in the airlock, with very little time left. continued on page 23

RGB Interface Power Supply

Eric Ockenden (July 1990, p7)

Those enterprising persons who have embarked upon the task of putting together the interface unit required to provide the correct signals from the T199/4A console to the Wang colour monitor may find that when they finally get a good image on the screen it is marred by broad darker bands with soft edges slowly moving down the screen.

These bands are caused by the interference effects produced by the mixing of the small ripple voltage getting through the power supply filter and the field scanning frequency. The fact that they move so slowly down the screen means that the two frequencies involved are very nearly the same, but not quite (50Hz).

The strength of tone of these bands relative to the main screen depends on the amplitude of the ripple voltage. It seems that the screen image is extremely sensitive to this ripple interference. A scope measurement showed that only a few millivolts of ripple produces an unacceptable pattern.

Normally one would expect that the 1000uf/25v filter capacitor across the +12v bus would be enough to smooth the supply adequately, for other purposes this would be so, but in this case it is not. The steady current of this unit with a video signal only (no audio) being processed is approximately 100ma. This figure varies according to the volume of audio output, with peaks approaching 200ma. Therefore the amplitude of the audio tends to modulate the strength of the bars due to the poor regulation of the power supply.

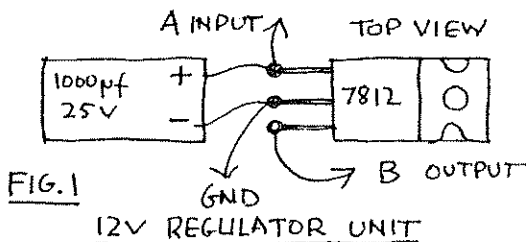
The seriousness of the effect will depend greatly on the type and quality of the plug pack or transformer used to power the unit. Most ordinary DC plug packs have very poor regulation with exaggerated current specifications. Using a discrete transformer of 2amp rating produced some banding when used without a regulator even when the 1000uf cap was increased to 3300uf.

The solution to this filtering problem is based on the use of an adequate transformer or plug pack. For safety reasons the inexperienced constructor would be advised to use a plug pack so that handling the mains connections to the transformer would be avoided, but for those who would prefer to use the transformer (which would be mounted inside the case of the unit) have the mains connections checked before plugging into a power outlet.

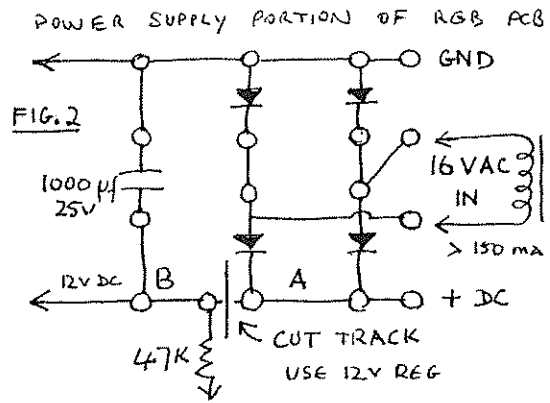
Checking the current suppliers' catalogues, only one suitable plug pack is available. Dick Smith's catalogue no. is M-9567 at \$16.95. This gives 18VAC at 900ma. The 900ma is overgenerous, but there is nothing having 16VAC with a lesser rating (300ma).

With this pack the modification is simple and results certain. Two extra components are required.

1. A 7812 voltage regulator TO220 case.
2. A 1000mf/25v electrolytic capacitor.



See Figure 1 as to how to combine these two items into a single unit with 3 leads attached. (INPUT, GND, OUTPUT). There is one change to the PCB. One track to cut. Figure 2 indicates where to make this cut.



Mounting the regulator is not difficult. The metal tab of the 7812 is as one with the centre common pin and can be bolted directly to the metal housing or other convenient location, and is at GND potential. The three leads can then be soldered to the points indicated in Figure 2.

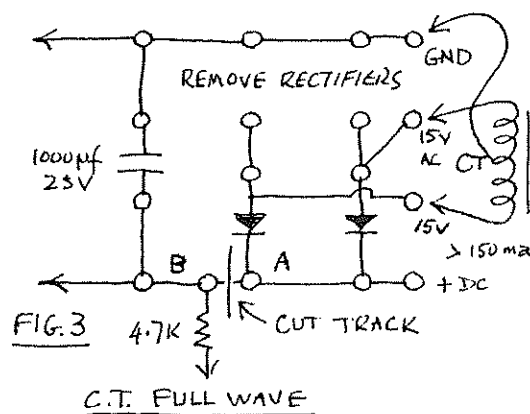
A heat sink is not required as the 7812 is operated at a very low current level.

Voltages less than 15v AC or DC and current ratings less than approximately 300ma will not allow the regulator to keep the ripple to the very low level required. This rules out all of the other advertised plug packs.

A second way to fix the problem is to use a discrete transformer. The smallest transformer suitable for this job is a Dick Smith or JayCar unit DSE 2155 at \$9.95 Catalogue M-2155, rated at 1 amp. with a tapped secondary at 0 to 8.3, 7.5, 8.5, 9.5, 12, 15v. Use the 0 to 15 terminals. This transformer is larger than needed but there is nothing smaller listed.

Note:- 12.6v AC or DC is not suitable to power the 7812.

The simplest and most reliable way to go is as outlined above, that is: Use the 16VAC 900ma plug pack with the 7812 regulator. However, for those who might like to explore other approaches the following information is presented.



Jaycar catalogue a small 240vac transformer with suitable ratings, if used according to the full wave centre tapped circuit together with the 12V 7812 regulator. See Figure 3 for details. This transformer gives the required 15VAC input at 150ma and is small enough so that together with the 12v regulator will mount easily inside a suitable box. It is listed at \$8.95 Catalogue MM-2007. Note that the two upper diodes are removed and the centre tap from the transformer is taken to a convenient GND point.

continued on page 28

Trials of a Disk Fixer

Robert Peverill (September 1988, p9)

Some time ago, when SSSD full height drives for the TI99/4A were still quite expensive, I was able to obtain a DSDD 80 TPI full height drive at a good price. Unfortunately the "Disk Manager II" module could only do 40 TPI DSDD. Not to worry, at the price 40 track DSDD was better than nothing. Happiness followed, as now I had twice as much storage capacity for storing and developing my programs, but, Murphy's Law was lurking not far away in the shadows, waiting for the most inopportune time to strike and create disaster. Then it happened, "BRRR" "BRRR", "BRRR" "BRRR", the new drive crashed, taking with it a program from the "HOME COMPUTER MAGAZINE" that I had been debugging and other programs I had also been developing at the same time. Why had I not backed them up you ask? Simple, my differing disk formats made it slow to back up, plus it would require twice as many disks for the backing up of the double sided drive, and as it was at a time when disks were costing around \$40-\$50 per box, I say no more.

As I said this was quite some time ago, so long ago in fact I had forgotten what programs had been lost. In the mean time the drive had been repaired, new compatible DSDD 40 TPI drives purchased and all odd format disks transferred over. But still the old damaged disk lingered on. I obtained a "NAVARONE DISK FIXER MODULE" and after a long period of "fits and starts" I had dumped various sectors of "good" and "bad" disk Directories, Bit Maps, Directory Link Maps, Directory Entries and various sectors of program data.

I read the Navarone booklet but found it a little confusing in parts, then read the series of articles by Bruce Caron and found these cleared up a majority of the hazy areas.

So with listings, articles, pencils and "Oh my aching head, what am I in for?" set about the task of finding out what happened to the bad disk and how to fix it.

The symptoms of the disk failure were as follows:-

When using Disk Manager II to catalog the disk, it would start cataloguing normally then produce garbage for a number of lines, missing some programs before completing the catalog. When trying to back up the disk using DM II only some of the programs catalogued could be copied, many could not.

The Fix

Start by reading sector 1 (Directory Link Map).
R 0001,D (R 1,D)

```
NAVARONE IND. *** DISK FIXER V2.0 ** SECTOR DUMP SECTOR ADDRESS 0001
ADDR = 01 23 45 67 89 AB CD EF INTERPRETED
```

```
0000 = 0013 0014 0010 000B 0012 000A 000F 0011 *****
0010 = 0009 000E 0008 0007 0006 0003 0004 0005 *****
0020 = 0002 000C 000D 0000 0000 0000 0000 0000 *****
0030 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0040 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0050 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0060 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0070 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0080 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0090 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00A0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00B0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00C0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00D0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00E0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00F0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
```

Count the number of 1 word (2 bytes) Directory Entry sector addresses, add 2 (1 for Link Map and 1 to be sure) and convert to hexadecimal. This gives the number of sectors to print. Now print out all used directory sectors for evaluation.

P 0002,D,N (P 2,D,N)

The last sector printed should contain ">E5"s (intepreted as "e") in every byte, if not, print some more sectors. Do not exceed sector >21 (33) as this is the last standard Directory Entry sector. If your Link Map contains more than 30 Directory Entry addresses

then you will have to read the Link Map and print all the additional sectors indicated.

Now the good part begins. With print out and pencil in hand do the following. Using the Directory Link Map, create a listing of all file names in the order in which they appear. This is done by taking the first sector address, going to that sector and writing down the interpreted file name found at that sector. Return to the Link Map for the next sequential address to check, and so on until all file names have been written down. Where a scrambled entry occurs just put a line or mark in the listing to show a fault occurred. Check that all files are in some sort of alphabetical order.

Having done that the next thing to do is get some sector details from the entries. Go to sector 2.

```
NAVARONE IND. *** DISK FIXER V2.0 ** SECTOR DUMP SECTOR ADDRESS 0002
ADDR = 01 23 45 67 89 AB CD EF INTERPRETED
```

```
0000 = 542F 4D41 4E20 2020 2020 0000 0100 000E T/MAN *****
0010 = 6300 0000 0000 0000 0000 0000 2200 0000 C*****"p"p
0020 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0030 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0040 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0050 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0060 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0070 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0080 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
0090 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00A0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00B0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00C0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00D0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00E0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00F0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
```

- Bytes 0-9 = File name.
- Byte >C = Indicates the file type. Use information to decode the file details and write it down next to the directory entry sector (in my case 01, a BASIC program).
- Bytes >E->F = The total numbers of sectors used to store the file.
- Byte >10 = Indicates where the end of file ((EOF) marker (>AA program, >FF variable files) occurs in the last sector used for the file.
- Bytes >1C->1E = When decoded using information mentioned previously, gives the first sector address used for the file and the number of consecutive sectors used in the file block (not counting the first sector). Write these values down.

Continue this process for all remaining Directory Entry sectors.

For the file indicated in sector 2, the starting sector should be >22. Now take the details of bytes >1C->1E (sector 2), decode the start sector and file length, add them together and write it down. This is the last sector address of the file. Do the same for the remaining good sector entries. Now take sector 2 start and finish sector addresses and add 1 to the finish sector address (another way is to add the value at byte >F to the start sector of entry). Compare it with the start sector address of the next file shown in sector 3. They should be the same. Continue doing this for all good entries. Now go back to any bad entries found previously. Look at the entry prior to the bad one, take the finishing sector of that entry, add 1 to it and write the number down as the start sector address next to the bad entry. Look at the entry following the bad one, take the starting sector, subtract 1 and write it down as the finish sector of the bad entry. Subtract the new start address from the new finish address, add 1 and write it down as the new total file length. Print or read the last sector as indicated. Search for the EOF indicator (>AA or >FF) and write the byte address down next to the bad entry. Now we have a probable START, FINISH and LENGTH of the file, also where the file finishes in the last sector. If the damaged entry is not too bad then read it in, otherwise read a good entry of a similar file type. Use the Alter command to change the hexadecimal values at bytes 0-9 to the appropriate file name. Ensure byte >C has the correct bits set for the file type. Alter bytes >E->F to show the total number of sectors used for the file. Alter byte >10 to show the new EOF

marker position. Encode the new start and finish sector addresses and Alter bytes >1C->1E with the new values. Now write this sector back to the disk. This concludes the reconstruction of the File Directory Entry.

Now to fix a directory that would not print out. Simple, read in a Directory Entry similar to the missing entry, alter it to suit the new values found using the steps previously described, then write that sector back onto the unreadable sector address and WULLAH! (Well it worked for me).

Quit Disk Fixer and insert Disk Manager II and catalog the disk. In my case the above procedures worked and I have now recovered all my lost files from the bad disk.

And now for some interesting details. In attempting to recover my disk I looked at both good and bad disk listings. My files were mainly BASIC programs which did not always fill the last sector, and because of this I found some interesting things.

They are as follows.

All BASIC program files end with >00 >AA >3F >FF >11 >03. >00 is the program end of line "symbol", >AA is the EOF marker, >3F >FF >11 >03 is unknown but is always there.

The 8th byte after the EOF marker contains the sector address of the Directory Entry for the file.

The 10th byte after the EOF marker contains the file type details.

Immediately after the file type byte or the 11th byte after the EOF marker, the file name begins and continues for 10 bytes including blanks, however, even though the interpreted ASCII shows the correct spelling of the file name the hexadecimal value of the first byte is always offset by >80. Eg. "A"=>41 is shown as >C1. Why? I do not know.

Bytes 6 and 7 after the filename give the total number of sectors used by the file.

Bytes 39 to 41 after the EOF contain the encoded start and finish sector address.

These details only appear on a less than fully used program file final sector but can help simplify reconstructing a Director Entry.

```
NAVAKONE IND. *** DISK FIXER V2.0 ** SECTOR DUMP SECTOR ADDRESS 002F
ADDR = 01 23 45 67 89 A B C D E F INTERPRETED
```

```
0000 = 54BE C801 3000 084D 4554 BEC8 0130 0008 T>H*O**MET>H*O**
0010 = 5358 BEC8 0233 3100 0946 4946 BEC8 0233 SX>H*31**FIF>H*3
0020 = 3100 0846 54BE C802 3331 0007 54BE C802 1**FT>H*31**T>H*
0030 = 3331 0007 53BE C802 3331 0007 46BE C802 31**S>H*31**F>H*
0040 = 3331 0009 9DC8 0543 4C45 4152 0015 9A20 31**H*CLEAR**
0050 = 494E 4954 4941 4C49 5A45 2056 414C 5545 INITIALIZE VALUE
0060 = 5320 00AA 3FFF 1103 0000 0002 0001 D42F S **?*****T/
0070 = 4041 4E20 2020 2020 0000 0000 000E 0000 MAN *****
0080 = 0000 0000 0000 0000 0000 22D0 0000 0000 *****"p****
0090 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00A0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00B0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00C0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00D0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00E0 = 0000 0000 0000 0000 0000 0000 0000 0000 *****
00F0 = 0000 0000 0060 0000 0000 0000 0000 0000 *****
```

Why these details occur and how accurate they are, I am not sure, but they do. Only the person or persons at TI who created the TI99/4A Disk Operating System (DOS) know for sure. Maybe they could be compelled to feel obliged to disclose the full details of the "DOS", after all, it has been 4 or 5 years since TI pulled the plug on the 99/4A. Besides, with Corcomp and Myarc running their own "DOS", it could not hurt to give it up now. Failing TI giving up the details, there may be others who do know the exact "ins and outs" of the TI99/4A "DOS" and maybe it is time they let the "cat out of the bag".

The above details and procedures have been for non-fractured BASIC/Extended BASIC program files. o

continued from page 2

I find the machine to be just what I want and I am encouraged by reports of new hardware and software being continually developed. I do not plan on abandoning the machine and will use it throughout the 90's. I hope this gives you some idea as to how I take advantage of the TI's many fine features. o

Teachers use the TI99/4A

Bob Relyea (June 1990, p19)

I was asked last year to do an article explaining how I use the TI as a teacher. Obviously it took me quite a while to get around to it. Surviving the first few months of the new editing job played a role in the delay.

First of all, I use the TI because it is an easy system to get acquainted with. Apple pretty much has their way as far as the school market is concerned. Do not ask me why as the Apple IIe that we have in our Science department is an archaic device that does not have many of the nice programming features that the TI does. For example, it will not do automatic line numbering (NUM) and it will not resequence (RES). Editing is very clumsy as well, but with the TI it is very simple.

I did not write this article simply to rubbish the Apple but to explain why I use the TI instead of other machines in my teaching. It has all the features that I need. I am basically a word processing, data base and spreadsheet person. These features are well developed with the TI and I get out of them all that I need at a price that leaves other computers for dead! Cost-wise they do not even compare. This is all to say nothing of the fun I get out of the club and the things that we all learn together. I like the fellowship of the Main and Regional Meetings to boot. People have been very friendly and very willing to help in every way.

Here are some of the things that I have done with the TI the past couple of years:

1. Grades for the HSC students have to be 'standardised'. This is a word which basically means the average deviation from the mean of a set of grades. It is a very time consuming procedure to do manually. I adapted a program written for the Apple to run on the TI and everytime I need a set of grades standardised which is about 20 times a year, I just boot up the Grade Standardiser program (see page 27 of the May, 1989 issue), bung them in the old TI and let 'er rip. In a short period of time it is all done and I get an instant print-out with the scores.

2. SORTING is a feature of both Multiplan and TI Base that I find extremely useful. For instance, every year my year 11 and 12 students purchase a book containing the last half dozen HSC papers in Physics together with suggested solutions. I type these all in TI Base under the titles:

CORETOPIC SUBTOPIC YEAR QUESTNUM

For example,

Waves Diffraction 1984 B 18

Once this is all typed in I usually sort on subtopic which ends with all questions on a particular subtopic being listed together. This means that everytime I finish teaching a topic, for practice we turn to the HSC questions booklet and know where to find all questions on that topic without having to look all over for them. I find this very useful.

While on the topic of sorting, I use a similar procedure for producing class lists. Last year I took on the job of Year Patron for the incoming year 7's. Year Patrons are responsible producing classlists for each of the nine year seven classes that we have. I originally typed these all in manually on Multiplan which involved the Surname, First name, and Roll Class. I stored each one under its separate name, such as 7Green. Print-outs were then produced from these lists. I also used the external merge feature of Multiplan to get all of these lists together and then I used the SORT feature to get an alphabetised list of every student in year 7 (216 students). This list is extremely useful

continued on page 2

Games Information

Robert Brown and Stephen Judd (March 1988, p7)

This is the first article on GAMES INFO to help people to solve games, adventures and any program that you dont understand or is giving you the hassles.

Just say that you were stuck on Return to Pirates Isle and you could not get out of bed. You could dial up TEXPAC and send some mail to username 'GAMES', and within a week you could be on your way to finishing it. Whatever your question may be, we can either solve it, or guide you in the right direction.

Now we had better get started on this month's file. This month we will be talking about test modes in games. You may not know this but quite a few of the old TI modules have the hidden test modes. This allows users to "CHEAT" in several ways, for example, change level or lives or speed depending on the capabilities of the game.

The modules we know that do this are...

Alpiner, Chisolm Trail, TI Invaders (disk tape version only), MunchMan, Munch Mobile, Moon-Mine, and TI Runner.

"How do we do this sir?", I hear you ask. Well my son, it is simple. All the above games except TI Runner and Chisolm Trail you can type '***' which is 'SHIFT-8-3-8' at the title screen. With Chisolm Trail you must wait until it gives you the prompt LEVEL (1-9) then quick as a flash type *** and you are in. With TI Runner you can skip up a level by typing FCTN-5 before you have started your present level.

If you find any more of these little surprises then tell us so we can tell everybody else.

Here we go with the solution to Hitch Hikers Guide to the Galaxy, Part 1

Here you are, mild mannered Arthur Dent, about to start the worst day of your life, although you do not know that...yet! Actually, the day is already getting off to a bad start, since you have just woken up in the dark, with a really bad headache (and it is all downhill from here).

The first thing you need to do is stand up and turn on the light. That is a little better, anyway! Or maybe not, since you are having a hard time getting coordinated. Grab the dressing gown and put it on, then look in the pocket. Ah, an analgesic! Take that, then get the screwdriver and the toothbrush, and head South to the porch (did you hear a tree fall? Rather ominous, is it not?).

Here you find something no modern home should be without: junk mail. Take the mail, and go on outside. Uh-oh! There is a very big bulldozer on its way to level your home, and there is Prosser standing by, watching it all. Are you going to take this lying down?

You bet you are! That is the only way to stop it: lie down in front of the bulldozer. No matter how close the thing gets, do not panic; it will not run you over (of course, in a short time, it really will not matter what happens to the house, but you do not know that yet). Just wait awhile until Ford Prefect shows up (read the junk mail while you wait).

Ford seems a trifle preoccupied with the sky, but he is aware enough of you to try and give you back your towel. Do not take it, or he will leave and you will be a lot worse off than you ever imagined (can things be worse than this? They sure can!).

Instead of taking the towel, ask Ford about your home. He will eventually come to his senses, and realize what is going on. When that happens, he will be able to persuade Prosser to take your place in front of the bulldozer, while the two of you head off to the pub to hoist a few.

As soon as Prosser takes your place, go South and West to the pub. Buy a cheese (?) sandwich while you wait for him to arrive (when you read the description, you will understand about the "?"). When Ford gets there, he will buy you a few beers. Drink only three of them.

Around about the time you have finished the third one, there will be a loud crash. In fact, it is the

sound of your home being demolished by the bulldozer (that will teach you to trust anyone who wears a digital watch!). Do not take that sitting down, leave the pub and return to where your house used to be. Along the way, you will see a starving dog.

While you may wonder if anything could eat that sandwich and survive, give it to the dog, who will (amazingly!) enjoy it immensely, ignoring a microscopic space fleet that whizzes past (remember that fleet). Then continue on to the ruins of your home (Ford will be right behind you).

And just about now, to put a perfect ending to a perfect day (which has just barely begun), the Vogon construction ships appear, to demolish the Earth to make way for a new Hyper-space Bypass (hmmm, maybe Ford was not kidding when he said he was from another planet, or that Earth would be destroyed in a short time).

Still, do not panic...wait until Ford drops the Sub-etha signalling device. There will not be much time after that, so pick up the device, push the green button (if you dropped the Aunt's thing, have no fear: it will turn up again later), and you will be in...the dark.

Get used to that, you will be spending a lot of time there before this adventure is over. Notice that, at first, you cannot do much. All your five senses seem to be out of order. However, if you wait, and read the descriptions very carefully, you will see that eventually, it mentions only 4 of your senses. The one that is missing is the one you can use. Keep this in mind, it will come in handy later.

Right now, your nose seems to be working again, so smell. Sniff, sniff. Ugh! Whatever it is, it sure is strong! You are also now dimly aware of a shadow, so look at it. Well, well, it turns out to be Ford Prefect! And, looking around, you find yourself in the hold of a Vogon ship. Certainly better than being on Earth (or where Earth used to be).

There is a glass case with an Atomic Vector Plotter inside, but do not bother with it yet. You have something else to do first, namely, obtain a Babel Fish. That should not be hard, right? All you need to do is push the button on the dispensing machine, and you will have one, right? Hehehehehe!

Those Babel Fish are pretty slippery characters (but, you may have found that out already for yourself!). And the cleaning robots are certainly no help, they seem to have only one mission in life; grabbing your Fish away from you. Well, we really cannot let that happen!

So, first thing to do is remove your gown and hang it on the hook. Now, wait for Ford to curl up, then get the towel and the satchel. Put the towel over the drain, and the satchel in front of the robot panel. Now comes the part that drives most people crazy: they do not know how to stop the upper-half of the room cleaning robot. But, it is so simple; just put the junk mail on top of the satchel.

Now you can push the button! Then step back and watch the Rube Goldberg shenanigans, which end with the Babel Fish stuck solidly in your ear (squish!). Bet you never thought outer space would be like this! However, now that you have the fish, you will be able to understand anyone who talks to you.

By the way, somewhere along the line, you will get a message that one of the phrases you have used was instrumental in starting a war that wiped out most of a small galaxy. There is nothing you can do about this; no matter how you try, it will come to pass. Rather unfortunate, is it not? Even more unfortunate, sooner or later, the survivors will figure out how that happened, and they will be looking for revenge...but more about that charming prospect later.

Right now, press the switch on the case. This will tell you what the code word is that will open the case so you can snatch the plotter. Make careful note of what word is required; it is chosen randomly each time. Too bad you have to listen to some pretty rotten poetry to get the word.

Speaking of poetry, in a short while, you and Ford will be hustled into the Captain's quarters, and strapped into Poetry Appreciation Chairs (worse things could happen, but right now, you probably cannot think of any).

continued on page 9

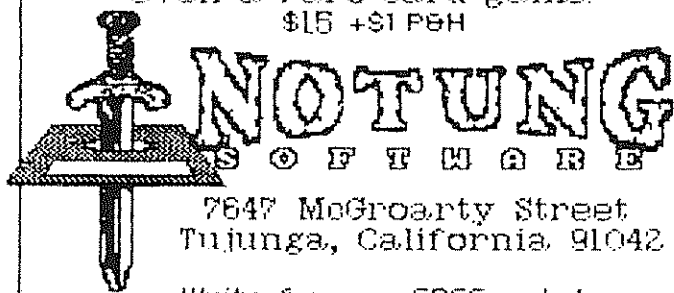
WHAT DOES PLAYING FARO, FINDING OUT HOW 'WILD' WILD BILL REALLY WAS, AND SINGING 'HOME ON THE RANGE' HAVE IN COMMON?



DISK of the OLD WEST

By Ken Dillard

Venture into those thrilling days of yesteryear with Notung's "Disk of the Old West". This well-researched 4-diskette package is filled with over 55 TI-Artist Pictures and Instances, 3 Fonts, 5 period songs, text essays on the some of the legends and even a Faro card game!
\$15 +\$1 P&H



7647 McGroarty Street
Tujunga, California 91042

Write for our FREE catalog

TI CASINO Supplement

While not an update for TI CASINO, the TI Casino Supplement provides a variety of add-on features for TI Casino. There's four new joke routines for the comedian at Club Notung that are considerable more racy than the original one. Part of the TI Casino's lobby has been remodeled to add in a second floor for more Casino action. There's a new "Old-West" Faro table you can put on the second floor and there's an alternate new higher paying slot machine with some new graphics. Just \$5. (Package Deal: TI CASINO and SUPPLEMENT disks for \$18) +\$1 P&H

Here's some other outstanding NOTUNG products:

\$1 P&H for 1st item, 50¢ for each additional item

Disk of Dinosaurs.....\$7
Son of the Disk of Dino...\$10
Bride of the Dsk of Dino..\$12
Ken's MIDI FAVORITES (Volumes 1, 2, 3 & 4).....\$5 each
Disk of Horrors.....\$12
Disk of Pyrates.....\$10

Book: "How to Use the Printer's Apprentice... And not go INSANE!.....\$19.95

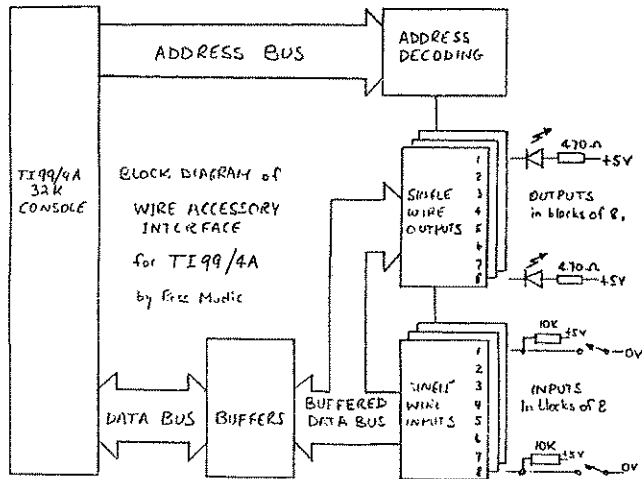
A new game for 9640 ABASIC!
Darrin Andrade's "WINGQUEST"
.....\$12

Wire Accessory Interface for the TI99/4A

Ross Mudie (July 1988, p5)

Have you ever thought "I would like to control the kids' train set with the computer?", or may be, "How can I connect a full music keyboard into the TI99/4A?", or "How would I control a robotic unit with the TI99/4A?". Other possible applications include: a security system; a fancy sign; or the Christmas tree lights.

Have I attracted your interest? The prototype of the Wire Accessory Interface Unit was first shown on the TIISHUG tutorial day at Burwood on 4th June 1988.



This article will show how such things are possible on a 32K memory expanded TI99/4A console. I have developed a prototype of a circuit which plugs directly into the expansion port of a 32K console which can have up to 128 single wire inputs or outputs, in blocks of 8 of either type, that is, inputs or outputs. The unit provides and accepts 5 volt logic signals on the output and input wires. Additional circuitry will be required to drive circuits requiring other conditions. My first application is for the control of a model train set, to perform the tasks of changing the points, operating the signal lamps, controlling the 12V power to the rails to stop trains and to use the information from train movement detectors to prevent train crashes when more than one train is operating on the same track. After due consultation with younger son, Peter, it was decided to go for 32 inputs and 96 outputs for the train set, used as follows:

- Inputs: 29 track sections to be monitored.
3 spare.
- Outputs: 32 for the 16 points on the layout.
29 for controlling 12 volt power to the track.
32 for signals, 1 per track section + 3 amber.
3 for miscellaneous building lights.

1. Construction of the demonstration prototype.

The development unit was constructed on a small piece of veroboard connected directly to a 44 way plug which plugs into the TI99/4A's expansion port. The 5 volt DC power rail from the console was used for the PCB power. Constructors should check the current drawn from the console 5V power rail and provide an external 5V power supply if the current drawn exceeds 50mA. The current drawn by the prototype was 100mA with no LEDs on and 170mA with the LEDs on.

The circuit is memory mapped from hexadecimal 8680 to 868F, which means that it uses 16 bytes. This memory area is allocated to the sound chip which is not fully decoded. This means that the sound chip will respond to values written to any memory location between hexadecimal 8400 and 87FF. Whilst this can cause some false operation of the sound chip, no other problems occur and the sound chip can be easily turned off again. It is also placed in memory immediately above John Paine's Time of Day clock.

The prototype unit was set up with 8 outputs driving 8 Light Emitting Diodes (LEDs) and 8 inputs connected from a PCB mounted switch. The outputs are at hexadecimal 8680 whilst the inputs are at hexadecimal 868C.

2. How it works.

The inputs and outputs are interfaced to the computer via 74LS373 integrated circuits which are "Tri-State octal latches". This means that they handle 8 circuits, (that is what "octal" means), and they provide a memory element for each input line, (that is the latch).

The remainder of the circuitry provides the decoding of the address bus of the computer. What this means is that when the computer addresses the appropriate memory location then the addressed input or output chip is allowed to look at the data bus and to store the value present on the data bus, if it is an output, or to place a value on the data bus if it is an input. When the extended basic statement (or command) CALL LOAD is used in the format CALL LOAD(-31104,V) the computer will place the value in the variable V on the data bus and the value hexadecimal 8680 on the address bus. After a short delay there is a pulse on the WE* line which "strokes" the output interface chip. IC25 buffers the data bus, since up to 16 74LS373 chips may need to be driven from the data bus.

This is probably a good time to look at the circuit diagram. Integrated circuits (1A), (2), (3) and (4A) decode the address bus of the computer. When the value hexadecimal 8680 (decimal -31104) is present on the address bus and the "not Write Enable" (WE*) line is active (low), then the "Latch Enable" (LE) input of IC6 receives a pulse of logic 1 and at this point IC6 stores what ever is present on the data bus. The stored value is then maintained on the output of IC6 until a new value is written into IC6 or the computer is turned off.

To perform a read from an input with Extended BASIC, a CALL PEEK is used as follows: CALL PEEK(-31092,A). This will read the 8 switches by placing a pulse of logic 0 on the control pin 1 of IC7 for the duration of the Data Bus IN (DBIN) signal whilst the address hexadecimal 868C is decoded by the PCB logic. Whilst the control input of IC18 is at logic 0, its outputs become low impedance and the conditions on the inputs are passed to the outputs which are connected to the data bus of the TI99/4A.

3. Increasing the unit from 16 to 128 lines.

The prototype unit was constructed for just 16 lines to try out the idea. Each 8 inputs require another 74LS373 chip and a NOR gate for each 8 outputs and an OR gate for each 8 inputs.

The chip requirement for the 128 line unit with 96 outputs and 32 inputs is as follows.

IC no.	Qty.	Type	Function.
1	1	74HC20	Address bus decoding logic 1's.
2	1	CD4078	Address bus decoding logic 0's.
3	1	74LS154	Address bus decoding 4 LSB's.
4	1	74HC32	Control logic.
5	1	74HC32	Final address decoding for input.
6-21	16	74LS373	Output buffers and input selector.
22-24	3	74LS02	Final address decoding for output
25	1	74LS245	Bi-directional data bus buffer
26	1	74HC04	Inverters.

The prototype unit contains ICs 1, 2, 3, 4, 5, 6, 18, 22, 24 and 25.

4. How to build your own unit.

There is a lot of work to design a printed circuit board layout for 128 inputs and outputs and the cost of a small number of double sided plated through boards may not be worthwhile. A problem with committing the circuit to a PCB layout is that the PCB does not allow outputs to be converted to inputs. There may however be interest in a smaller PCB with, say, 16 inputs and 16 outputs.

If anyone is interested in getting a PCB for this design then please let me know how many I/O's you would like. If there is enough interest steps will be taken to design a PCB. Of course the simplest way is to

"roll your own" on veroboard, this way you can make a unit with just the required I/O quantity and combination.

5. Assembly language programming.

The design lends itself to easy assembly language programming. Access is simple and should be quick, making processing of the inputs by an interrupt routine most probably quite feasible. As programs are developed they will be published if any interest is shown in the project.

6. Controlling a train set or other device.

The inputs and outputs are not suitable to directly control the train set. As the rest of the interface is developed copies of circuits will be available to be published if there is interest from the membership. The I/O is 5 volt logic only and is NOT protected against destruction from connecting the wrong things to it, nor is it protected against static electric discharge, apart from the normal protection provided within the integrated circuits.

The author will take no responsibility for any damage that any person may do to their computer as a result of information contained in this article.

7. Summary.

This development represents a simple way of using a "spare" TI99/4A console to perform a control task. The circuit developed so far is unsuitable for use in the expansion box since there are minor interfacing differences to operating via the side port of the console.

Additional details of hardware and software development will be made available if interest is shown in the project.

```

100 ! SAVE DSK6.CONTROL1
110 CALL INIT
120 DISPLAY AT(4,2)ERASE ALL: "DEMONSTRATION PROGRAM
FOR": TAB(10); "PROTOTYPE": " ACCESSORY
CONTROLLER": : TAB(8); "by Ross Mudie"
130 DISPLAY AT(10,1): "To control lamps,": "PRESS": :
"1. Switches on PCB": "2. Binary count": : "3.
Keys 1 to 8 control lamps"
140 DISPLAY AT(22,1): "Use BACK (Fctn 9) to return to
this menu from any part"
150 CALL KEY(3,K,S):: IF K<49 OR K>51 THEN 150 ELSE
K=K-48
160 ON K GOTO 190,290,410
170 !
180 !
190 ! SWITCHES TO DIRECT CONTROL LAMPS
200 DISPLAY AT(4,1)ERASE ALL: "OPERATE SWITCHES ON
PCB.": : "Note value on screen & lamps"
210 CALL ESC
220 CALL PEEK(-31092,A)
230 DISPLAY AT(20,1):A
240 CALL LOAD(-31104,A)
250 CALL KEY(3,K,S):: IF K=15 THEN 120
260 GOTO 220
270 !
280 !
290 ! STEP THROUGH LAMPS IN BINARY SEQUENCE
300 DISPLAY AT(4,1)ERASE ALL: "Note value on screen &
lamps"
310 CALL ESC
320 FOR V=255 TO 0 STEP -1
330 DISPLAY AT(20,1):V
340 CALL LOAD(-31104,V)
350 CALL KEY(3,K,S):: IF K=15 THEN 120
360 NEXT V
370 GOTO 120
380 !
390 !
400 ! KEYBOARD TO CONTROL LAMPS
410 CALL CLEAR :: CALL ESC
420 DISPLAY AT(4,1): "USE KEYS 1 to 8,": "note value on
screen & lamps": : "1 2 3 4 5 6 7 8" :: GOTO
460
430 CALL KEY(3,K,S):: IF K=15 THEN 120 ELSE IF K<49 OR
K>56 THEN 430
440 K=K-48
450 IF KM(K)=1 THEN KM(K)=0 ELSE KM(K)=1

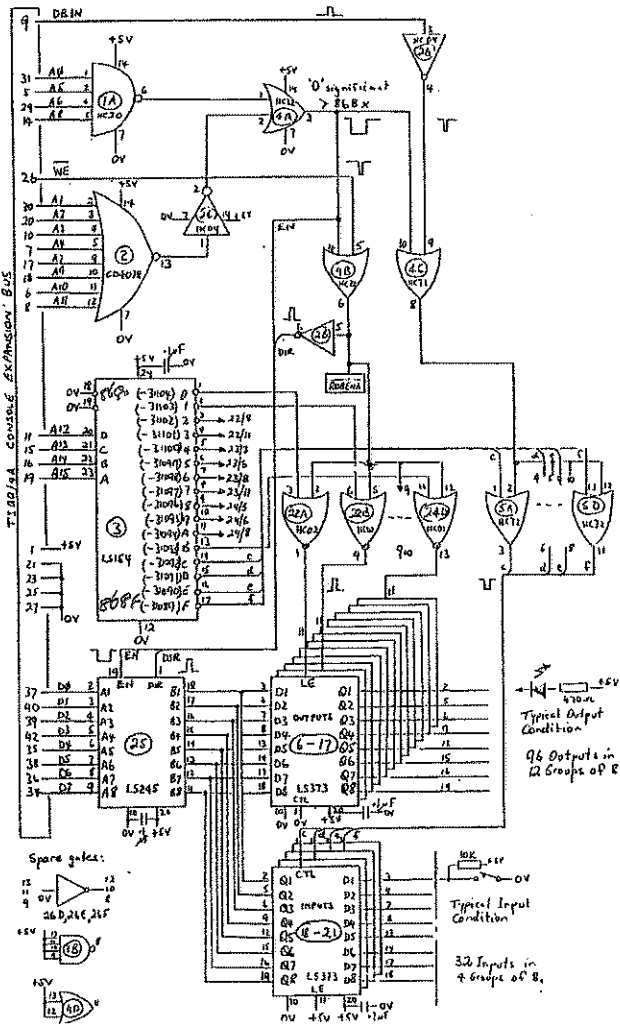
```

```

460 DISPLAY AT(10,1): KM(1); KM(2); KM(3); KM(4);
KM(5); KM(6); KM(7); KM(8)
470 V=0
480 FOR T=0 TO 7
490 IF KM(T+1)=1 THEN V=V+(2^T)
500 NEXT T
510 DISPLAY AT(14,1):V
520 CALL LOAD(-31104,255-V)
530 CALL KEY(3,K,S):: IF S<>0 THEN 530
540 GOTO 430
550 SUB ESC
560 DISPLAY AT(24,1): "Use BACK (fctn 9) to escape"
570 SUBEND

```

WIRE ACCESSORY INTERFACE
for TI 99/4A, 128 I/O.
Ross Mudie 30/5/88



Address Decoding

A	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
>868x	1	0	0	0	0	1	1	1	0	1	0	0	0	x	x	x

See Ross Mudie's I/O Interface at the Faire.

The Parable of the TI

Dick Warburton (June 1991, p2)

Once upon a time there were some little computers who lived with their parents in America. One of them became an orphan, because his parents did not want him any more. Lots of people tried to look after him, but because they did not get much help, they had to work things out for themselves, and he had a hard time growing up. His computer friends however, had such a good time, because they did not have to do much for themselves, and people did not ever expect them to grow up. Their parent companies had lots of other new computers whom they trained to take the places of the older computers. The little computers were very smart, but the old ones grew lazier and lazier, because their parent companies did lots of things to keep them happy. They just kept on doing the same old things, and because they had slick advertising to support them, they just kept on going as they were. People who looked after them, paid lots of money for the privilege, and bought them all sorts of expensive toys to play with. Some of these toys were very old, and were hard to operate.

People kept on looking after them, and these machines paid for their keep by performing for their masters. Some of them could only really play games and amuse people, but one of them was very good at school, and came to have lots of friends. Unfortunately his parents died too, and he became an orphan. Then almost nobody wanted him, because he was only good at schoolwork. The other one went to work in an office, and he learned lots of things to do. Unfortunately, he was not very versatile, and even though people spent lots of money writing simple instructions for him, he did not always do what he was told, and some people wished that he would grow up. He was also hard to look at, and did not have nice colours. He was very slow, and some people thought that he was stupid. Most of his owners, got fed up with him and locked him away in a cupboard, and bought a shining new model, with lots of new functions and shiny new colour monitors and things. These new machines were called DOS machines. DOS stands for Dammed Orrible System. DOS machines are very good at making their owners feel really helpless and frustrated, and are often used as paperweights or typewriters.

The little orphan had a hard time, and had to work hard to survive. He was able to do lots of things. He was not very big, but he was very intelligent and learned quickly. He was very efficient for his age and size, and because he was only small, he had to learn lots of tricks to get the jobs done properly. People who looked after him, liked him, and helped him to grow up into a very sophisticated computer. He could do almost everything that the other three computers could do, and was much more obedient. His instructions were much easier to give, and he had users who liked him. They did lots of things to make his job easier. Some of his users gave him a beautiful colour monitor, while others liked to see how fast he could work by providing a ramdisk. He kept on growing stronger and stronger, and better. The other computers almost stopped growing. Because he is made so well he can be fixed up if he gets sick, and hospital bills are small. His friends like to help each other, so they are able to do things to expand him much more cheaply than other computers. This makes them happy in a time when costs are high, and there is not much money. This little orphan expects to live for a long, long time. He has friends all over the world who care for him, and who help to keep him growing. They keep making things for him which give him bigger muscles, and let him run much faster. His owners are very proud of what he can do, when all the costs and comparisons are made.

A simple story, but with much truth. The TI can do so many things well. With ramdisks and eproms to help it along, it is efficient, user friendly, and will do

almost everything I could want of my computer. It is cheap to run, cheap to expand, and has a great bunch of users world wide who help each other. It is a fun machine for real computer users. Through club membership, owners can become proficient in a number of ways. They can get help with programming, using a wide range of software, buying up to date software, and getting help for hardware problems. Costs for club members are kept to a minimum. Members can learn to about computer hardware, add accessories cheaply, expand their machines at minimum cost. All for a miserly \$30 per year, let alone receive an excellent magazine and have the use of our own Bulletin Board. If your membership falls due, renew now, so our orphan can continue to grow, and we can continue to provide the range of services to club members.



'They've called their first stop-work meeting.'

Speeding up Memory Image Loader

Phillip Hayes (July 1986, p26)

After purchasing the memory image files tape release, I found the loader to be too slow in loading it from Extended BASIC. The loader takes 2 minutes 16 seconds to load, far too long when it is considered that the modified version of mine loads in exactly 32 seconds, yet contains the same code. This article will explain how the loading time was reduced by 75%.

The BASIC pointers at -31952 and -31950 refer to the start and end of the line number table. After Extended BASIC is selected, these pointers indicate the first free address to store a BASIC program. More on this later.

The TI99/4A, in saving BASIC programs to cassette or disk has a minor flaw. It will save from the top of memory to the address pointed to by -31950, not looking at the actual location of the tokenised program. Combining these features, it is evident that it is possible to reserve space at the top of the memory map from BASIC programs and have this memory saved with your BASIC program.

This is how the loader was "sped-up". Sufficient space was allocated at the top of memory to contain the Memory Image Loader code. A small BASIC program was written to define the pointers for low memory at 8194, 8196, add the program names to the DEF table and "poke" a small machine language routine into low memory which would move the code from the allocated space to its original location so the Memory Image Loader was then able to be LINKed to from BASIC.

The next step was to move the code from low memory into the allocated space at the top of memory and SAVE to CS1. The process is complete!

When loaded, the BASIC program can be RUN. After invoking the machine language routine through a LINK, the Memory Image Loader is LINKed also. The Memory Image Loader code is contained at the top of memory, so the move routine can simply transfer it to low memory. In effect, the Memory Image Loader code is saved to cassette in memory image form!

The above also applies to disk. Furthermore, cassette owners can use machine language routines in their programs and have them loader with their programs. Another application is saving data with a program, among other possibilities.

Please enjoy your fast loading loader, as I have found the time saving considerable using this method rather than the conventional!

CALL LOAD(address, direct data) method.

Jenny's Younger Set

Joshua Rust (October 1987, p8)
and Vincent Maker (June 1990, p20)

Dear Jenny,

This is another quiz program on DOCTOR WHO. I hope it is OK for THE YOUNGER SET.

VINCENT MAKER

```
100 CALL CLEAR
110 REM *****
120 REM *DOCTOR WHO QUIZ*
130 REM *
140 REM *      BY      *
150 REM *      *      *
160 REM * VINCENT MAKER *
170 REM *      *      *
180 REM *      FOR      *
190 REM *      *      *
200 REM * MELANIE LEWIS *
210 REM *****
220 DISPLAY AT(5,7):"QUESTION ONE:NAME THE MAN WHO WAS
    BEHIND THE CURSE OF PELADON?"
230 DISPLAY AT(8,7):" A) ARCTURUS  B) IZLYR
    C) HEPESH  D) KING PELADON"
240 PRINT "PRESS THE CORRECT LETTER."
250 CALL KEY(O,J,I)
260 IF I=0 THEN 250
270 IF J=67 THEN RIGHT=RIGHT+1 ELSE WRONG=WRONG+1
280 CALL CLEAR
290 DISPLAY AT(5,7):"QUESTION TWO : WHO LED THE DOCTOR
    AND ROMANA TO SAFETY IN THE 'WARRIORS GATE'?"
300 DISPLAY AT(10,7):" A) BIROC  B) RORVIK
    C) DAVROS  D) CONGRESSMAN BROOK"
310 PRINT "PRESS THE APPROPRIATE KEY."
320 CALL KEY(O,J,I)
330 IF I=0 THEN 320
340 IF J<>65 THEN WRONG=WRONG+1 ELSE RIGHT=RIGHT+1
350 CALL CLEAR
360 DISPLAY AT(5,7):"QUESTION THREE : WHO KILLED DAVROS
    IN 'RESURRECTION OF THE DALEKS'?"
370 DISPLAY AT(10,7):" A) THE DOCTOR  B) TEGAN
    C) STEIN  D) THE DALEKS"
380 PRINT "PRESS THE RIGHT KEY."
390 CALL KEY(O,Y,U)
400 IF U=0 THEN 390
410 IF Y=67 THEN RIGHT=RIGHT+1 ELSE WRONG=WRONG+1
420 CALL CLEAR
430 DISPLAY AT(5,7):"QUESTION 4 : WHO KILLED LYTTON IN
    'ATTACK OF THE CYBERMEN'?"
440 DISPLAY AT(10,7):" A) THE DALEKS  B) THE DOCTOR
    C) THE ICE WARRIORS  D) THE CYBERMEN"
450 PRINT "PRESS THE ANSWER."
460 CALL KEY(O,B,A):: IF A=0 THEN 450
470 IF B<>68 THEN WRONG=WRONG+1 ELSE RIGHT=RIGHT+1
490 IF RIGHT=0 THEN A$="NEVER GO FOR A JOB AS A
    'DOCTOR WHO' EXPERT. ZERO OUT OF FOUR."
500 IF RIGHT=1 THEN A$="ARE YOU SURE IT WAS THIS QUIZ
    YOU WERE DOING? ONE OUT OF FOUR."
510 IF RIGHT=2 THEN A$="DON'T RATE YOURSELF A PRO JUST
    YET. TWO OUT OF FOUR."
520 IF RIGHT=3 THEN A$="YOU ARE NEARLY AN EXPERT...
    BUT NOT QUITE..."
530 IF RIGHT=4 THEN A$="IT'S A SHAME THAT THEY'VE
    TAKEN THE PROGRAM OFF THE AIR. YOU COULD'VE GIVEN
    SYLVESTOR MCCOY A RUN FOR HIS MONEY."
540 CALL CLEAR
550 PRINT A$
560 PRINT
570 PRINT
580 PRINT
590 PRINT "YOU GOT ";RIGHT;" RIGHT AND ";WRONG;" WRONG."
600 END
```

Dear Crocodile Jones,

I have a problem with ADVENTURE 7. I cannot get off the merry-go-round with out getting killed.

How do I do it?

V. MAKER

Dear Vincent

Look at the the controls of the ride. There is a button there. Press it and the ride will stop.

Then it will be safe to jump from it. CROCODILE JONES

Dear Jenny,

Here is a program that will make a sprite move in a circle by plotting the sprit around the circle's circumference. By altering the variable R (radius), you can increase or decrease the size of the circle. Be careful not make the radius too big - 32 is as large as I would go - as bad values can occur. A small circle, say of radius 3 or 4, will produce a good "hovering effect with a larger sprite pattern than that provided, and that might be useful for a ghost etc in a game.

```
100 CALL CLEAR :: CALL SCREEN(2)
:: CALL CHAR(128,"60FOFO60")
110 FOR D=1 TO 99999 STEP 8
120 A=D*(22/7)/180
130 R=18
140 X=R*COS(A)+138
150 Y=R*SIN(A)+96
160 CALL SPRITE(#1,128,6,Y,X)
170 NEXT D :: GOTO 110
```

You will notice that the circle is not perfectly smooth. You can obtain a smoother circle by decreasing the step rate in line 110. This however will slow down the program. Similarly, by increasing the step rate, the program will speed up, but the circle will be less smooth. This can be partially improved by then decreasing the circle's radius. The co-ordinates of the centre of the circle are 96 down by 138 across. This of course can be altered to suit the direction of rotation and can be changed by swapping over COS and SIN in lines 140 and 150.

By making various alterations to the program, you can produce some quite interesting results. Here are some examples of the experiments that I have tried:

To get a figure eight, try the following:

```
130 X=R*SIN(A)+138
140 Y=R*SIN(2*A)+96
```

To obtain a vertical ellipse, alter line 130 to read:

```
X=2*R*COS(A)+138
```

The value "2" can be altered to vary the height and width of the ellipse. For a horizontal ellipse, change line 140 to read:

```
Y=2*R*SIN(A)+96
```

Again alter the value "2" to vary the shape of the ellipse. A three leaf clover can be obtained with the following:

```
130 X=20*COS(A)*COS(3*A)+138
140 Y=20*SIN(A)*COS(3*A)+96
```

More leaves can be added by increasing "3" in both lines to a higher number. NB: a value of "4" doesn't give 4 leaves, neither does "5" etc.

Try these:

```
130 X=R*COS(2*A)+138 or
```

```
130 X=R*COS(3*A)+138 or
```

```
130 X=R*COS(A)+138
140 Y=R*COS(2*A)+96 or
```

```
130 X=R*COS(.5*A)+138
140 Y=R*COS(2*A)+96
```

continued on page 25

Clock Programs for Hard Disk

Ben Takach (August 1989, p15)

A number of cards designed for the TI99/4A feature battery backed real time clocks. Many TI99/4A users have purchased or assembled one of these cards from kit sets, yet one finds hardly any programs, which have utilised the date and time features of the inbuilt real time clock. I have to confess that many of my own programs have ignored the clock ticking away in my CorComp Triple Tech card for over half a decade. The arrival of the Myarc hard disk card changed it all. I soon got sick of setting the clock every time the system was booted up. A number of time related programs were incorporated in various programs which required time or date information. My programs naturally have been written for the Triple Tech and Myarc combination. The programs will have to be edited accordingly to suit John Paine's or Peter Schubert's cards.

Here is the basic, stand alone program, which was used in part in other programs to obtain the time or the date or both.

```

100 !SAVE WDS1.UT11.TIME
101 OPEN #1:"CLOCK" :: INPUT #1:A$,B$,C$ :: CLOSE #1
102 SEC$=SEG$(C$,7,2):: MIN$=SEG$(C$,4,2)::
   HR$=SEG$(C$,1,2):: DAY$=SEG$(B$,4,2) ::
   MON$=SEG$(B$,1,2):: YR$=SEG$(B$,7,2)
103 OPEN #1:"TIME",INTERNAL,FIXED :: PRINT
   #1:SEC$,MIN$,HR$,DAY$,MON$,YR$ :: CLOSE #1
110 CALL CLEAR
120 DISPLAY AT(10,1):"SYSTEM CLOCK SET/DISP. PROG."
130 PF=0 :: IF P$="" THEN P$="PIO"
140 DISPLAY AT(12,1):"Set/Display/Print clock or End
   the program? (s/d/p/e)"
150 CALL KEY(3,R,S):: IF S<>1 THEN 150
160 IF R=83 THEN 170 ELSE IF R=80 THEN 340 ELSE IF R=68
   THEN 350 ELSE IF R=69 THEN 640 ELSE 150
170 CALL CLEAR :: IF SEC$="" THEN SEC$="55"
180 DISPLAY AT(4,5):"YEAR ? ";YR$
190 DISPLAY AT(5,5):"MONTH ? ";MON$
200 DISPLAY AT(6,5):"DATE ? ";DAY$
210 DISPLAY AT(7,5):"HOUR ? ";HR$
220 DISPLAY AT(8,5):"MIN. ? ";MIN$
230 ACCEPT AT(4,12)SIZE(-2):YR$
240 ACCEPT AT(5,12)SIZE(-2):MON$
250 ACCEPT AT(6,12)SIZE(-2):DAY$
260 ACCEPT AT(7,12)SIZE(-2):HR$
270 ACCEPT AT(8,12)SIZE(-2):MIN$
280 DISPLAY AT(10,1):"Any correction ? (y/n)"
290 CALL KEY(3,R,S):: IF S<>1 THEN 290
300 IF R=89 THEN DISPLAY AT(10,1):RPT$(CHR$(32),28)::
   GOTO 230 ELSE IF R<>78 THEN 290
310 OPEN #1:"TIME",INTERNAL,FIXED
320 PRINT #1:SEC$,MIN$,HR$,DAY$,MON$,YR$
330 CLOSE #1 :: GOTO 110
340 DISPLAY AT(18,1):"PRINT DEVICE ? ";P$ :: ACCEPT
   AT(18,16)SIZE(-3):P$ :: PF=1
350 CALL CLEAR
360 OPEN #1:"TIME",INTERNAL,FIXED
370 INPUT #1:SEC$,MIN$,HR$,DAY$,MON$,YR$
380 CLOSE #1
390 IF MON$="01" THEN MO$="JAN"
400 IF MON$="02" THEN MO$="FEB"
410 IF MON$="03" THEN MO$="MAR"
420 IF MON$="04" THEN MO$="APR"
430 IF MON$="05" THEN MO$="MAY"
440 IF MON$="06" THEN MO$="JUN"
450 IF MON$="07" THEN MO$="JUL"
460 IF MON$="08" THEN MO$="AUG"
470 IF MON$="09" THEN MO$="SEP"
480 IF MON$="10" THEN MO$="OCT"
490 IF MON$="11" THEN MO$="NOV"
500 IF MON$="12" THEN MO$="DEC"
510 IF PF=1 THEN 580
520 DISPLAY AT(10,9):DAY$;" ";MO$;" .19"&YR$
530 DISPLAY AT(11,9):HR$;" H.";" ";MIN$;" MIN."

```

```

540 FOR PAUSE=1 TO 4000 :: NEXT PAUSE ! ** you may
   shorten the loop delay at will**
550 DISPLAY AT(20,1):"Hold down key B to exit
   fromdisplay loop until minutes are incremented."
560 CALL KEY(3,R,S):: IF R=66 THEN 110 !** or END at
   your option **
570 GOTO 360
580 DISPLAY AT(12,1):"Print 1 Date only, 2 Time only, 3
   Date & Time." :: OPEN #1:P$,OUTPUT !*insert spaces
   between options 1,2 and 3 according to desired
   screen display image **
590 CALL KEY(3,R,S):: IF S<>1 THEN 590 600 IF R=51 THEN
   610 ELSE IF R=50 THEN 620 ELSE IF R=49 THEN DF=1 ::
   GOTO 610 ELSE 590
610 PRINT #1:DAY$;" ";MO$;" .19"&YR$ :: IF DF=1 THEN 630
620 PRINT #1:HR$;" H.";" ";MIN$;" MIN."
630 CLOSE #1 :: DF=0 :: GOTO 110
640 END

```

How does it work?

Line 101 inputs the time and date string from the Triple Tech card.

Line 102 converts the string to a form, which the Myarc card can understand.

Line 103 will set the clock of the Myarc Hard Disk Card.

Lines 120 to 160 is the menu segment displayed on the screen. The SET option will enable the user to change the setting of the Myarc clock, if for any reason a different date or time setting is to be used.

Lines 170 to 300 deal with the entry data of the time and date. The entry routine uses an endless loop to correct any of the entered data. I use this routine in many of my programs where a number of questions have to be answered. The default data is already displayed, one may accept it or change it at will.

Lines 310 to 330 will change the setting of the Myarc clock.

Lines 360 to 380 will input the time and date information from the Myarc clock.

Lines 390 to 500 are used to change the numerals 01 to 12 into the name of the months. This is the best way to avoid the frustrating confusion of date display conventions. Most of the clock cards of USA origin use the mm.dd.yy convention (including Myarc).

Lines 520 to 570 are used to display the date and time at the centre of the screen. It is a continuous display, which is incremented every minute. Seconds are not displayed. Displaying the seconds is an overkill, the second count can hardly be read, thus confusing, besides the display of the seconds is almost always inaccurate. The program moves on when the letter "B" is pressed at the appropriate time. Impatient users, who get frustrated by the long delay may shorten the pause in line 540. Key unit 3 (as we all know), returns the upper case ASCII code regardless of the keyboard status set by the ALPHA LOCK key.

Finally lines 580 to 630 execute the print-out options.

A segment from the above program was incorporated in the LOAD file of the hard disk manager program, and also saved as an independent program by the title TIMESET. Timeset is incorporated in most of the LOAD programs I am using.

```

100 ! SAVE WDS1.UT11.TIMESET
110 OPEN #1:"CLOCK" :: INPUT #1:A$,B$,C$ :: CLOSE #1
120 SEC$=SEG$(C$,7,2):: MIN$=SEG$(C$,4,2)::
   HR$=SEG$(C$,1,2):: DAY$=SEG$(B$,4,2) ::
   MON$=SEG$(B$,1,2):: YR$=SEG$(B$,7,2)
130 OPEN #1:"TIME",INTERNAL,FIXED :: PRINT
   #1:SEC$,MIN$,HR$,DAY$,MON$,YR$ :: CLOSE #1

```

Other programs, which require the input of date, such as an invoice writing program or the Keating's folly company car log book program may use another segment of the time program.

```

330 OPEN #1:"CLOCK" :: INPUT #1:A$,B$,C$ :: CLOSE #1
340 SEC$=SEG$(C$,7,2):: MIN$=SEG$(C$,4,2)::
   HR$=SEG$(C$,1,2):: DAY$=SEG$(B$,4,2) ::
   MON$=SEG$(B$,1,2):: YR$=SEG$(B$,7,2)

```

```

350 OPEN #1:"TIME",INTERNAL,FIXED :: PRINT
      #1:SEC$,MIN$,HR$,DAY$,MON$,YR$ :: CLOSE #1
360 GOSUB 1690

460 DISPLAY AT(7,1):"DATE ? ";DA$

490 ACCEPT AT(7,8)SIZE(-10):DA$

```

```

1690 REM DATE INPUT ROUTINE
1700 OPEN #1:"TIME",INTERNAL,FIXED
1710 INPUT #1:SEC$,MIN$,HR$,DAY$,MON$,YR$
1720 CLOSE #1
1730 IF MON$="01" THEN MON$="JAN"
1740 IF MON$="02" THEN MON$="FEB"
1750 IF MON$="03" THEN MON$="MAR"
1760 IF MON$="04" THEN MON$="APR"
1770 IF MON$="05" THEN MON$="MAY"
1780 IF MON$="06" THEN MON$="JUN"
1785 IF MON$="07" THEN MON$="JUL"
1790 IF MON$="08" THEN MON$="AUG"
1800 IF MON$="09" THEN MON$="SEP"
1810 IF MON$="10" THEN MON$="OCT"
1820 IF MON$="11" THEN MON$="NOV"
1830 IF MON$="12" THEN MON$="DEC"
1840 DA$=DAY$&"."&MON$&".19"&YR$
1850 RETURN

```

The above program lines were extracted from my invoice writing program. Note line number 490. This line gives a choice to accept or change the date, if for any reason the invoice date is not the actual date it was prepared.

Lines 1730 to 1830 may be considered by some observers to be superfluous embellishments. I included these, because of the number of conventions used worldwide to write the date. This way the day and the month is clearly defined.

Compare the lines 390 to 520 of the time program and lines 1730 to 1840 of the above program segment. If lines 390 to 500 are omitted from the former, then MO\$ in line 520 have to be changed to MON\$. Lines 1730 to 1830 may be omitted without any further change to express the months in arabic numerals. Some European countries use roman numbers to write the months. If this convention is adopted then MON\$ will be made equal to "I" to "XII".

In conclusion, I use another finger and time saving routine to ensure that my Myarc RAMdisk is ready and waiting for me at DSK2 every time the system is turned on. This program unfortunately will not run from Extended BASIC, thus it has to be loaded and run from BASIC in two moves.

```

100 CALL PART(32,400,80)
110 CALL EMDK(2)
120 CALL CLEAR
130 PRINT "PUSH ALPHA LOCK DOWN, THEN TYPE; BYE & PRESS
RETURN": : : : : : : : :
140 END
150 REM PROGRAM NAME: START

```

The purpose of line 130 may need some explanation. This program is very useful if the computer is used by non computer minded office staff. My system is also used to send and receive telex messages. The outgoing telexes and the incoming messages are stored on the RAMdisk to save time. This program prepares the RAMdisk and ensures the use of the mandatory use of capital letters. The next program is started by the command: RUN "DSK1.TERMINAL".

```

100 !SAVE DSK1.TERMINAL
110 OPEN #1:"CLOCK" :: INPUT #1:A$,B$,C$ :: CLOSE #1
120 SEC$=SEG$(C$,7,2):: MIN$=SEG$(C$,4,2)::
      HR$=SEG$(C$,1,2):: DAY$=SEG$(B$,4,2) ::
      MON$=SEG$(B$,1,2):: YR$=SEG$(B$,7,2)
130 OPEN #1:"TIME",INTERNAL,FIXED :: PRINT
      #1:SEC$,MIN$,HR$,DAY$,MON$,YR$ :: CLOSE #1
140 CALL CLEAR :: DISPLAY AT(1,1):"CHECK SYSTEM
CONFIGURATION: MODEM SET TO: 300 orig
switch NOT ON BELL (up)"

```

```

150 DISPLAY AT(5,1):"auto/manual switch off
PRINTERSELECT:
switched to MP"
160 DISPLAY AT(10,1):"PRINTER: paper
roll fed in, platen press lever engaged,
power & on line lights on"
170 DISPLAY AT(14,1):"pitch set to 12 char/inch (dip
sw. 1 off, 2 on)"
180 DISPLAY AT(18,1):"PRESS ANY KEY WHEN READY TO
PROCEED" :: CALL KEY(0,R,S):: IF S<>I THEN 180
190 CALL CLEAR :: DISPLAY AT(12,1):"the name of the
parameter file is: LOGON
200 FOR PAUSE=1 TO 500 :: NEXT PAUSE
210 RUN "DSK1.FASTTRMIXB"

```

The time/date routine is again incorporated in this program besides its other functions. It will remind the operator to prepare the printer, then it will run the terminal program.

The examples show the numerous applications of the real time clock in a TI99/4A environment. Not to speak of the enjoyment of the astonished looks and the inevitable question from the onlooking fellow 99'ers "how did it do it?"

One Liners

Dennis Hodgson (April 1986, p7)

Like the majority of TI99 members, as identified by a survey last year (1985), I have had my TI99/4A for a couple of years and have tinkered with programming with varying success. I, like most, have only the basic computer and cassette recorder (plus Extended BASIC, a necessity!). I have spent a good deal of time concentrating on writing one liners because I became horrified with the slow cassette loading speed, yet could not justify a disk system.

One liners load in less than 10 seconds: a poor man's disk drive, if you like. I found that I could achieve quite powerful programs in one line. Also, I really had to learn precisely how every programming command/statement operated. It is a great challenge to write One Liners and perhaps the best way to learn our Extended BASIC language.

there is no perfect program. Thus programs can always be improved. Perhaps that is why we do not get around to submitting our efforts. Let us get off our tails and make some program submissions, be they good or bad, in case they help someone else or, even give someone an idea and everyone benefits!!!

Here is a One Liner for starters. See if you work out how it functions. It will give the day of the week for any day, month and year since 1905. At the prompt enter, for example, 25,12,1947. Answer = "TH" for Thursday.

Thanks to Alan swales for the excellent information in the September 1985 issue of the SND (TND) regarding the Georgian calendar.

Programming hint

Using Extended BASIC, you are able to enter a line that exceeds 5 lines on the screen. This is how: Type in the line to the point where the computer beeps, int the case of this program, PRINT SEG (the end of the line) and press enter. Now type 100 (the line number) and then press FCTN[X] for the edit mode. The line will re-appear. Move the cursor to the end of the line and continue with the rest of the entry. Press enter when done. If you LIST the program, you will find a line longer than normally accepted by the computer. Now try it!

One Liner Program

```

100 A=1 :: INPUT D,M,Y :: FOR T=A TO M-A ::
      H=H+29+VAL(SEG$("20212122121",T,A)):: NEXT T ::
      J=H+(Y/4<>INT(Y/4)AND M>2)+INT(365.25*(Y-A))+D ::
      PRINT SEG$("SASUMOTUWETHFR",2*(J-7*INT(J/7))+A,2)::
      RUN

```

Real Time Clock

John Paine (March 1988, p5)

I have recently been spoiled by using the John Johnson Menu program in conjunction with a RAMdisk installed in my PE box. The menu program does so much that it is hard to believe that with extra hardware in the system, it could do more.

The software includes the necessary routines to access the three known Real Time Clock peripherals that have been on the market for some time. The purpose of this note is to describe a fourth clock that is compatible with the menu program as well as being a stand alone device that can be used by members who may not have a RAMdisk.

Before starting, let us look at the existing clocks available for the TI.

1) CorComp Triple Tech Card

This is a PE box card that uses an OKI clock chip and can be easily read, altered and used under BASIC. Its major disadvantage is that it is controlled by a DSR and is mapped into memory location >4000. The OKI chip is relatively small in size but difficult to use because of the need for external latches and support chips. CorComp also released a stand alone unit that plugs into the side of the console.

2) MBP Clock Card

Once again, this is a PE box card, but uses a National clock chip. The MBP card is addressed to operate in the memory mapped block starting at address 8640. The National chip is much larger in size, but requires less support in terms of extra chips. Once again simple BASIC programs can be used to set and read the clock. Because of the memory address location of the clock chip, it is necessary to incorporate a CALL SOUND statement in any BASIC program immediately following the CALL LOAD statements used to 'poke' the values into the clock registers. The CALL SOUND statement will turn off the sound generator, otherwise the noise will drive you crazy. Like the Triple Tech Card mentioned above, the MBP card has a battery back up.

3) John Clulow Clock

I have only seen some sketchy documentation on the Clulow clock, but it seems to use the same chip as the MBP card and is mapped in almost the same area as the MBP clock. I will assume that the same comments will apply as with the MBP clock.

As stated earlier, all these clocks are built around PE box cards, so with this in mind, the following circuit and ideas are based on a stand alone system, which can be built into an existing external peripheral, such as speech synthesizer, thermal printer, stand alone RS232, stand alone disk controller, or even the console itself. The minimum requirement for operation is possession of Extended BASIC to set the time. Time can be read with BASIC and E/A or Minmem, but you will have to write your own software. I will provide a disk to any user group with some utilities and set clock programmes written in Extended BASIC.

Now down to business. The clock that I am about to describe is based on the same National clock chip used by MBP, and will be mapped into the same memory address. The prototype clock was built on a small Tandy prototype card, which cost \$1.67 and the total project including clock chip, crystal, resistors, diodes, dry cell batteries (in this case), and chips cost less than \$30.00.

I built the total package, including batteries, into the top compartment of the Speech Synthesizer, which allowed for complete stand alone performance, and offered the added advantage of portability to show other members etc.

It is not my intention to describe a wire by wire method of assembly, because of the many possible locations for mounting. Instead refer to the circuit diagram below for connection of the clock to the major bus lines. You will also note that the circuit shows different options for battery back up. The prototype initially used dry cells, but was upgraded to nickel cadmium cells, which can be recharged when power is supplied to the host peripheral. Although a lithium battery option is shown, I consider the cost to be too high for such a project.

The parts needed are:

- A suitable board for mounting e.g. Tandy Proto board
- One only MM58167 clock chip from NSD Melbourne
- One 32,768KHz crystal from NSD
- One 30pf variable capacitor Geoff Woods
- Note: The crystal and trimmer can be salvaged from an old LCD digital watch.
- Two only 74LS138 decoders
- One 74LS04 hex inverter
- Two or three 1N914/1N4148 diodes
- One 22pf capacitor
- One 1K resistor
- One 220K resistor
- One 100 ohm resistor (if using Ni-Cads)

Batteries of your choice. I used half length AA cells from Tandy, although I later used a PC board mount Ni-Cad pack from Radio Spares (\$9.00 approx.). A heap of ribbon cable to connect the bus to the clock. There are 29 wires to be soldered to the expansion bus. This includes the 5 volt power and ground.

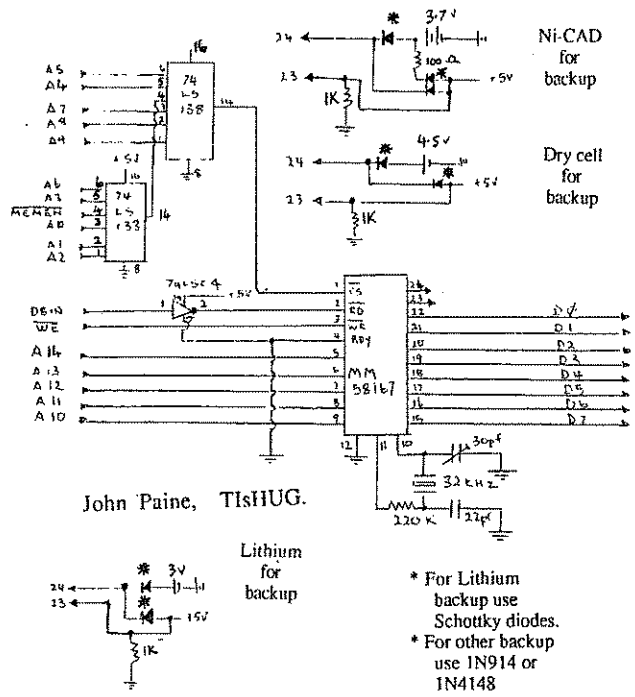
The actual wiring on the board is simple and is easily accommodated with point to point wiring. The external connections to the bus are the most tiresome.

To calibrate the clock, National Semiconductor recommends that a digital frequency meter be connected to pin 19 of the clock chip, and the trimmer be adjusted until a frequency of 500Hz is achieved. I just powered up and set the clock. After 3 weeks, the clock lost 5 seconds compared to my watch. I can live with that as it is quicker to load the SETCLOCK routine than tuck around with calibration.

Load the program JUSTCLOCK from the utility disk (XB). Immediately power is connected, the clock should start and count from 00:00:00. If all is OK, then the seconds will tick upwards. Obviously after 59 seconds, the minutes will increment. If the seconds do not change, you will have to check the crystal, 220K resistor and capacitors connected to pins 10 and 11 of the clock chip. If nothing displays, then check the address and data lines.

The disk that I mentioned above will have a number of utilities and sample programmes that can be used to set the clock and also to read the clock. I have also included a couple of cataloguing programmes which also make use of the clock and will date and time stamp the printed listing.

Those of you with a RAMdisk and Menu 7.1 will see immediately the benefit of the clock, but those without should not despair as the clock is very usable. In time I will release more utilities for the clock and perhaps some of the brighter XB programmers can come up with other applications.



Real Time Clock circuit diagram

Implanting Machine Code into BASIC Programs

George Meldrum (November 1987, p9)

This is to be a topic for the tutorial day. These notes are meant to supplement the talk and hopefully supply an adequate introduction to the topic.

To explain what we are trying to achieve imagine your BASIC program is a suitcase. Next think of fitting a false bottom to the case inside which you hide your secret code. Open the case (LIST) and all looks normal. Carry the case (OLD and SAVE) and the false bottom goes with it.

BASIC programs which have machine code programs embedded behind them provide a quick and easy form for the computer user. From a users view the advantages are:

- (1) Program loads and runs like any other XB program
- (2) No special loaders are required
- (3) Programs can be easily copied between tape / disk

The hardware required for this form of production technique is : Console + Extended BASIC + 32K memory then either a cassette recorder, or disk drive. The software required is a Debugger type program such as that supplied on the Editor/Assembler disk. If you do not have access to a Debugger program then you can copy a slightly modified version available on the tutorial day. A copy will also go to the software library and probably make its way to the club shop.

So that no complications are introduced select a single, Editor/Assembler option 5 file, for implanting. Keep a note pad handy to record the values displayed while using the Debugger. If you make an incorrect entry in the Debugger then type fctn-x to terminate the command. Be careful to include the "V" where indicated. Now down to it :-

- (1) Load the Debugger program :
CALL INIT :: CALL LOAD("DSK1.DEBUGX") -disk users
RUN "CS1" -for cassette users
- (2) Load the E/A opt 5 machine code file :
OLD DSK1.filename
OLD CS1
After loading an * I/O ERROR 50 message must be displayed.
- (3) Link to the Debugger program :
CALL LINK("DEBUG")
- (4) Cassette users ignore this step.
Type :
.M 969V <enter>
0969 = kk <enter>
- (5) Cassette users ignore this step.
Type :
.H 966,kk <enter> (use the value kk from step 4)
look for H1+H2=tttt
- (6) Disk users use the value tttt from step 5, and calculate the value uuuu as tttt + 5.
Type :
.M ttttV,uuuV <enter> -disk users
.M 969V,96EV <enter> -cassette users
Output :
0969 = 00 00 ee ee dd dd *****
- (7) Find the length of code.
Type :
.H eeee,6 <enter> (eeee from step 6)
look for H1-H2=nnnn
- (8) Calculate where to put the code in high-memory.
Type :
.H FF9C,nnnn <enter> (nnnn from step 7)
look for H1-H2=ssss

- (9) Cassette users ignore this step.
Work out where the code starts in vram.
Type :
.H tttt,6 <enter> (tttt from step 6)
look for H1+H2=cccc
- (A) Transfer code from vram buffer to high-memory.
Use the value cccc from step 9
value ssss from step 8
value nnnn from step 7
Type :
.N ccccV,ssss,nnnn <enter> -disk users
.N 096FV,ssss,nnnn <enter> -cassette users
- (B) Calculate new last free address in high-memory.
Type :
.H ssss,1
look for H1-H2=aaaa
- (C) Set BASIC's start and end of line number table pointers.
Type :
.M 8330 <enter>
8330 = FFE7 aaaa <space bar>
8332 = FFE7 aaaa <enter>
- (D) ** Full system users see note at end of step 10.
Poke in the following code (note that XXXX displayed is an unknown and unimportant value).
Type :
.M FF9C <enter>
FF9C = XXXX 02E0 <space bar>
FF9E = XXXX 3FC0 <space bar>
FFA0 = XXXX 0200 <space bar>
FFA2 = XXXX 03F0 <space bar>
FFA4 = XXXX 0201 <space bar>
FFA6 = XXXX A000 <space bar>
FFA8 = XXXX 0202 <space bar>
FFAA = XXXX 0310 <space bar>
FFAC = XXXX 0420 <space bar>
FFAE = XXXX 202C <space bar>
FFB0 = XXXX 0200 <space bar>
FFB2 = XXXX 08F0 <space bar>
FFB4 = XXXX 0420 <space bar>
FFB6 = XXXX 2024 <space bar>
FFB8 = XXXX 0200 <space bar>
FFBA = XXXX 030E <space bar>
FFBC = XXXX 0420 <space bar>
FFBE = XXXX 2030 <space bar>
FFC0 = XXXX 0200 <space bar>
FFC2 = XXXX 0401 <space bar>
FFC4 = XXXX 0420 <space bar>
FFC6 = XXXX 2030 <space bar>
FFC8 = XXXX 0200 <space bar>
FFCA = XXXX 07F5 <space bar>
FFCC = XXXX 0420 <space bar>
FFCE = XXXX 2030 <space bar>
FFD0 = XXXX 0200 <space bar>
FFD2 = XXXX ssss <space bar>
(take value ssss from step 8)
FFD4 = XXXX 0201 <space bar>
FFD6 = XXXX dddd <space bar>
(take value dddd from step 6)
FFD8 = XXXX 0202 <space bar>
FFDA = XXXX nnnn <space bar>
(take value nnnn from step 7)
FFDC = XXXX CC70 <space bar>
FFDE = XXXX 0642 <space bar>
FFE0 = XXXX 16FD <space bar>
FFE2 = XXXX 0460 <space bar>
FFE4 = XXXX dddd <space bar>
(take value dddd from step 6)
FFE6 = XXXX 0000 <enter>
- (E) Exit the Debugger.
Type :
.Q <enter>
- (F) You are now back to BASIC. Type the following:
100 REM PROGRAM NAME
110 REM
120 CALL INIT

continued on page 4

Programming c99

Craig Sheehan (August 1989, p14)

Many members have expressed the sentiments that BASIC is too slow, and assembly language is too time consuming both to learn and write. An easier way to obtain the speed of assembly and the ease of a language like BASIC is to use a compiled language. With a compiled language, like 'c99', you write your program using an editor, like Funlwriter, and then pass the text through a program called a compiler, that produces a machine language output.

This series of articles will concentrate on one such compiled language, 'c99', and will assume no prior knowledge. 'C' is a universal language used widely on all computers from home machines to large mainframes. Being so widely used, it offers the advantage that provided you have a 'C' compiler for a new computer, you can transport all your programs to this new computer.

Before you start using 'C', you must have a compiler. In the case of the TI99/4A, a compiler that supports a subset of the language has been written by Clint Pulley, and is available as a fairware program from the club shop as a floppy for the usual media fee.

We shall now launch into our first program, which like all first programs, simply prints a message on the screen. Whilst it does nothing useful, it serves as good exercise on using the compiler and showing the basic syntax of a 'C' program. The program is shown in Figure 1.

```
-----  
/* First 'c' program */  
  
extern printf();  
  
main()  
{ printf("Your first 'C' program\n");  
  exit(0);  
}
```

Figure 1 - Your first 'C' program

The program may seem simple, which it is, but you may think at first that actually getting into a runnable program is a little complicated. However, after doing it a few times, it will become second nature.

You will need two disks: one with the 'C' compiler (available from the club shop), and the other a blank initialised disk. From side A of the 'C' disk, copy the following files onto the blank disk: CFIO, CSUP, PRINTF and STDIO. Whilst we will not use all these files in this part, it will be handy, and save a lot of disk swapping, in later parts.

Now type in the program. This done using the editor of Funlwriter's Editor Assembler mode (option 1 on the Assembler menu rather than the Formatter menu; space toggles between them). You should use this editor as any carriage return (cr) characters in the program will ensure that will not work.

Do not be too concerned with the number of spaces between commands since 'C' is a free format language. This means that so long as the program has at least one space where I have put at least one, it does not matter how the program is set out. In fact, you could condense it into a single, unreadable paragraph, and it will still compile successfully. Once entered, save it to the blank disk. I will use the filename "DSK1.PRINT;C". The last two characters of this filename are used to make it easy to find 'C' programs on disk catalogs.

Having typed in the program and saving it, we now compile it using the following procedure:

- (1) Load the 'C' compiler from the 'C' disk using a program loader such as Funlwriter (option 3 - Loaders followed by option 3 - Program). The filename to enter is "DSK1.C99C" assuming the 'C' disk (side A) is in drive one.
- (2) Answer "N" to both of the two prompts "Include c-text" and "Inline push code".
- (3) Place the disk with the 'C' program into drive one. For the input filename, type in the filename of the 'C' program that you wish to compile. In this case it is "DSK1.PRINT;C"
- (4) For the output filename, enter the name you wish to call the assembly language output. For this example, I will use "DSK1.PRINT;S". The suffix ";S" signifies that this file is an assembly language file.
- (5) Once the compiler is finished, quit the program and select the assembler from Funlwriter (option 2 - Assembler).
- (6) Type in the name of file output by the 'C' compiler for the "Source file name". In this case it is "DSK1.PRINT;S". Then enter a filename for the "Object file name". I will use "DSK1.PRINT;O".
- (7) Leave the print device name blank and delete the "R" from the options, so that only the "C" remains. Press FCTN[6] and your 'C' program will be assembled. The resulting file "PRINT;O" is the machine language file that can be run.

Having finally compiled this short program, it can now be loaded. From Funlwriter, select loaders (option 3) followed by load/run (option 4). Place your previously blank disk in drive one and enter the following filenames in sequence: "DSK1.CSUP", "DSK1.PRINTF" and "DSK1.PRINT;O". Then press <ENTER> without a filename. Finally, enter "START", press <ENTER>, followed by FCTN[6], and if all goes well, your efforts will result in the message "Your first 'C' program" on the screen.

Having seen the result of the program in Figure 1, we will examine its operation. The first line is simply a comment. Any text that appears between the "/*" and "*/" characters is ignored by the compiler. The second line tells the compiler that the PRINTF function is required, and that it will be loaded separately at a later time. The rest of the program contains its body. "main()" defines a subprogram called main, whose statements are enclosed by curly brackets ({ ... }). Each statement is ended with a semicolon (";"). In this case there are two statements: the first is the printf command, that prints the message on the screen, whilst the second gives you the option of either returning the title screen or rerunning the program.

I can not stress enough that if you want to learn 'C', or any other skill for that matter, you must practice what is shown here and then experiment. For example, you may be wondering what the two characters "\n" do, since that are not printed on the screen. Place these two characters in the middle of a message and see what happens (do not forget to recompile it). You can also find out what error messages the compiler gives: leave out a semicolon, or a bracket and see what happens. Only by experimenting (it cannot damage the computer) can you learn and understand. ○

continued from page 9

In fact, time has just run out, and there you are in the depths of space. Lucky for you, the Guide explained how to survive all of 30 seconds out there! Well, perhaps not so lucky, since, considering the vastness of space, it is quite improbable that another ship will come by to pick you up before the 30 seconds run out. So naturally, 29 seconds later, the Heart of Gold (the HOG) comes past and picks you up. ○

From the Trenches

Werner Kanitz (October 1989, p30)

It did it again!!!

Half way through a page of typing the RAMdisk died.
Oh woe is me!

"Well there y'go complain enuff an' some one'l come t' the rescue (or the devil will come to get his dues)."

Remember that most wonderful of devices, that circuit board with all those fancy chips stuck onto it gingerly inserted into a vacant slot in that most useful of add-ons the PEBox, backed up as it were with a number of batteries in case there is a lack of power. I referred to it above, that devil's own device, the RAMdisk.

For years I managed to avoid constructing one due in no small way to the fact that I am more into spitting chips than soldering them. I could however not elude that silver tongued fox Herr Amadio. I was persuaded to buy a ready made model. Once installed the boy thought she was a ripper, and so it was, when it was running. Blessed are those without for they shall not suffer the aggravations of blips and surges, and require to reinstall their ROS every few weeks.

"It is the price of technology they reckoned yer tryen t' make the thing do things it wozn't meant t' do." That is not much consolation.

Never mind, Lou to the rescue. "Ve can fix det midt ein chip preprogrammed wif der operating system burnt into it for ever (EPROM)." Great news, no more worries, get me one. Done. Thinks I, get it home pop out the old, slip in the new and you are cooking with gas so to speak. "B!**sh!t!!"

You need different software to drive it, no problem, I will install it. You need to make hardware changes to your RAMdisk. (*@!*, is nothing simple!)

I do not know anything about oxy-cutting and welding even on this minute scale. It is getting to the point where I am tempted to cut my losses and go PC. At least everything is plug in. G%, I hate electronics!

The Rhyme to an Ancient Computer.
by Werner "Wordsmith" Kanitz

I don't know why I bought the thing
Some said it was a bargain
The cost was cheap free games rolled in
But then we reassessed again
For nothing there was serious

Chips and cards were everywhere
And the number of free slots did shrink
Chips and cards were everywhere
Still not enough I think.

Its chips were there its slots were free
Its keys cold and inviting
Its skin was white as leprosy
My nightmare TI PC was she
Who drains man's purse expanding

For joy for joy this glorious toy
The men would play and fiddle
Expanding up was all the go.
First box and then the extra memory
The RAMdisk followed close behind
To my everlasting misery.

I can't believe I am here today
Dad I but said, Mazz go away,
When first my mind was tempted
But then again as you can see
My soul could once again be free
If this association would be ended.

(my apologies to Mr Collieridge)

Debugs in de Print

Shane Ferrett (November 1987, p10)

LINES SHOULD READ AS FOLLOWS

```
32300 CALL LOAD(9726,4,192,2
16,0,131,124,2,224,131,224,4
,96,0,112):: CALL LOAD(8194,
39,4)::SUBEND
```

NB. 9726 INSTEAD OF 3726

```
32320 FOR B=1 TO 16 STEP 2 :
: C=ASC(SEG$(A$,B,1)):: D=AS
C(SEG$(A$,B+1,1)):: C=((C>57
)*7-48+C)*16+(D>57)*7-48+D
```

NB. the plus sign after 16 before bracket

```
32330 CALL LINK("POKEV",767+
8*A+(B+1)/2,C):: NEXT B :: S
UBEND
```

NB. SUBEND ADDED

```
32340 SUB COLOR(A,B,C) :: CA
LL LINK("POKEV",2063+A,(B-1
)*16+C-1):: SUBEND
```

NB. REMOVAL OF SUBEND

ONCE THE PROGRAM IS OPERATIVE MODS CAN BE MADE SUCH AS
THE MULTIPLE CALL LOAD STATEMENT CALL
LOAD(ADDR1,D1,D2,D3,D4,"",ADDR2,D5,D6,D7) TO SHORTEN
IT SLIGHTLY

AS A MATTER OF INTEREST FOLLOWING IS THE SOURCE FOR
THE CALL LOADS

```
DEF POKEV
FAC EQU >834A
NUMREF EQU >200C
XMLLNK EQU >2018
VMBW EQU >2024
PAD EQU >8300
GPLST EQU >837C
GPLWS EQU >83E0
NEXT EQU >0070
AORG 9472 >2500
POKADD BSS 2
INFOBF BSS 16
TEMP1 DATA 100
POKEWS BSS 32
AORG 9636 >25A4
* START OF PROGRAM
POKEV LWPI POKEWS GET OWN WORKSPACE
LIMI 0 DISABLE INTERRUPTS
* GET FIRST PARAMETER FROM BASIC
CLR RO
LI R1,>1
BLWP @NUMREF
BLWP @XMLLNK CONVERT TO INTEGER
DATA >12B8
MOV @FAC,@POKADD STORE AT COUNTER
AB @PAD+18,@TEMP1+1
* GET REMAINING PARAMETERS FROM BASIC
LI R3,>2
MORINP CLR RO
MOV R3,R1
BLWP @NUMREF
BLWP @XMLLNK
DATA >12B8
MOVB @FAC+1,@POKADD(R3) STORE IN BUFFER
INC R3
C R3,@TEMP1 LAST PARAMETER?
JNE MORINP NO LOOP FOR ANOTHER
MOV @POKADD,RO GET POKE ADDR IN VDP
LI R1,INFOBF POINT TO INFO TO POKE
MOV R3,R2 GET INFO BYTE COUNT
AI R2,-2 ADJUST COUNT BY 2-
BLWP @VMBW POKE VALUES IN VDP
* EXIT ROUTINE
CLR RO RESET STATUS
MOVB RO,@GPLST i.e. no error
LWPI GPLWS RESTORE GPL WORKSPACE
B @NEXT GOTO NEXT STATEMENT
'RETURN TO BASIC'
```

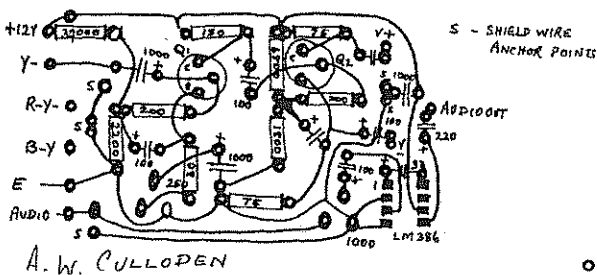
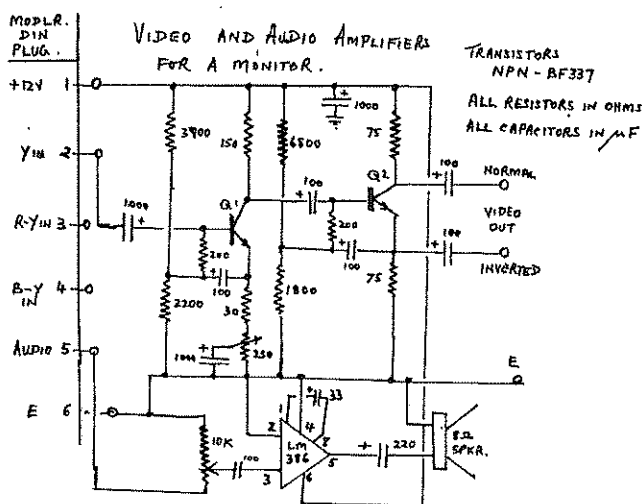
Adding Sound to a Monitor

Alf Cullogen (February 1988, p28)

I bought a monitor at the Adler warehouse sale on the last weekend in November. Ross was good enough to put it on the BBS that weekend, so if you did not see it, you would have missed a good deal. There were Model 312 Cicada Modems for \$270 brand new which, I considered a good buy but unfortunately I did not have the ready cash. The monitor tested 100% when fed a signal from a VCR, but no sound. The signals from the modulator socket would drive a speaker and the monitor but I felt there was a need to control the volume. As well, it would be handy to be able to control the video level too.

Being of Scottish ancestry and armed with Ben's copy of a circuit kindly provided in the September 1986 TND, it was necessary to get a couple of transistors that would work, since my junk box did not include BD139s. A couple of BF337s, which have a better frequency response but not quite the power capability, were substituted and the bias resistors adjusted so that 100KHz square waves at the input were bigger and just as square at the output. An LM386 was at hand so that it came in handy for the sound.

A carbon from my bank deposit book, which is hardly used for deposits these days, together with a cleaned bit of circuit board, were used with a Dalo pen to trace the circuit on the board and etch it the same afternoon. The circuit board is housed with the speaker in the same box with an RCA plug on a screened lead to take the video to the monitor. A DIN plug identical to that used for the modulator, was used to bring the inputs to the circuit board, using the 12V provided on pin 1 and the earth on pin 6. Changing to the TV is just a matter of swapping plugs to the one from the modulator. The speaker box sits behind the console, unseen but not unheard. A copy of the schematic and circuit board are included. Not a bad result for a beginner!



continued from page 30

The loss of so many members has been somewhat of a blessing in disguise for the remaining and new members, as there have been some amazing equipment bargains to be had and the majority of members now boast expanded systems with multiple disk drives (and in some cases, multiple RAMdisks).

With Texcomp's recent Multiplan sell-off, many members took the opportunity to purchase this terrific piece of spread-sheet software, so we may soon see a challenger to the current local favourite use for the TI99/4A, that of graphics. (I think it is a fair accomplishment that TI-Writer or Funnelweb is the most used software package for the TI99/4A, so please do not take me to task on the preceding statement).

Apart from the 'Volunteer', or casual Committee Member position, the Committee of TIUP has remained unchanged since 1988 when a bloodless coup prevented a takeover by the IBM compatible brigade. The group produces its newsletter, TIUP It BITS, every two months and has an active Newsletter exchange programme with the remaining Australian User Groups and a dozen or so groups in the USA.

TIUP contacts are as follows:

President:

Merv Trowbridge,
90 Williamson Avenue,
BELMONT, Western Australia 6104
Telephone (09) 277 1091

Vice President:

Frank Graham,
20 Hudson Street,
BAYSWATER, Western Australia 6053
Telephone (09) 271 5972

Secretary:

Geoff Warner,
3 Maru Way,
LESMURDIE, Western Australia 6078
Telephone (09) 291 7310

Treasurer:

Steve Wilkinson,
5 Carey Place,
COOLOONGUP, Western Australia 6188
Telephone (09) 528 3808

Committee Member:

Colin Bingham,
88 Rupert Street,
SUBIACO, Western Australia 6008
Telephone (09) 381 3781

The Group Address:

TI-99 Users of Perth (Inc),
PO Box 8083,
PERTH, Stirling Street, Western Australia 6849

continued from page 18

Please note that another change can be made to this program which involves putting the values X and Y into a CALL MOTION statement. The alterations are as follows:

105 CALL SPRITE(#1,188,6,96,96)
160 CALL MOTION(#1,Y,X)

I will leave it to you to decide which method produces the best results, but I should mention that for a sprite to maintain circular motion (actually moving, not just plotted) it has to undergo continual changes in direction, and thus the result is a bit jerky. Anyhow, you can decide which method you would prefer.

Happy Computing Joshua Rust

To Charge or not to Charge

John Ryan (July 1991, p8)

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

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

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

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

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

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

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

TMM saw an article in a Popular Science magazine many years ago which gave a simple transistor circuit to allow humans to recharge us about five or six times before we are finally discarded. The secret is not to let our single cell voltage fall below 1.0 Volt ie. a 9 Volt Battery which has six of us in series, should not be allowed to fall below 6 Volts on load. This means that we have to be removed from circuit and checked and never allow us to have our potential fall below 1.0 Volt. With a heavy current drawing dump truck toy you would have to check the potential quite shortly in spite of Junior's protests, whereas with a TV remote control it would not be necessary more than once in every six months. If you look at the curves for typical cells of different types, you will see that when we reach 1 Volt on load, we do not have far to go. Much like you humans when you reach 100 years of age. This means that we have to be removed from circuit and checked more often so that our 'On Load' voltage is never allowed to fall below 1.0 Volt. With a heavy current drawing toy dump

truck our potential will have to be checked quite shortly, in spite of Junior's protests whereas with a TV remote control it would not be necessary more than once every six months. If you look at curves for typical cells for the different types, you will see that when our potential falls to 1.0 Volt we are down to 10% of our energy level. Incidentally our mercury cousins should not be recharged because they can explode - just like you humans that have a mercurial temperament. If you have a charger you use to charge our Nickel Cadmium cousins and if the current is adjustable you can use it to charge us as well as long as you remember that our bigger brothers will stand a heavier current as shown in the table below. A very neat little charger can be found at Paddy's Market for about \$17 (maybe \$15 with a little bargaining) which allows four cells to be charged individually at one time, with a LED in series with each and with a built in battery checker. In the meantime, if you really wish to save money on us, keep your eye on our prices and you will find that they vary quite a lot from store to store. Make a point to avoid those of us that have been advertised on TV as the prices are loaded with the costs of the advertisements while they are really no better than some of us that are never brought into the limelight. We will try and encourage one of our Nicad cousins to write a further article on tricks and tips for using and caring for them.

Percent of energy remaining in carbon-zincs and alkalines for various ON LOAD voltages. Recommended Charging currents for our brethren of various sizes.

Volts	Current mA
1.5 = 100%	AAA 20
1.4 = 85%	AA 40
1.3 = 65%	C 60
1.2 = 35%	D 100
1.1 = 20%	9 Volt 10
1.0 = 10%	Lantern 140
0.9 = 0%	

continued from page 4

```

LWPI >3FCO      my workspace
LI RO,>03FO
LI R1,>A000
LI R2,>0310
BLWP @VMBR      read character definitions
LI RO,>08FO
BLWP @VMBW      relocate to E/A position
LI RO,>030E
BLWP @VWTR      E/A default color table
LI RO,>0401
BLWP @VWTR      E/A default pattern table
LI RO,>07F5
BLWP @VWTR      E/A default screen color
*
LI RO, FROM
LI R1, LOAD
LI R2, LEN
LOOP MOV *RO+,*R1+ relocate code
DECT R2
JNE LOOP
*
B @START      execute program
DATA 0
END
    
```

This same source code can be used each time with the only adjustment required being the four EQUate directives at the start. Now reload the BASIC code saved in step 10 and load the newly assembled code above.

```

OLD DSK1.filename
CALL INIT
CALL LOAD("DSK1.object_filename")
SAVE DSK1.filename
    
```

Now the code is embedded the program can be run at any time by: RUN"DSK1.filename".

Letter to the Editor

Michael Ball (September 1990, p3)

Dear Bob,

I have been reading the news digest and trying to think of ways that I could contribute to it. I have been a member of the TISHUG for about seven years now, and although I would not class myself as an avid computer user, I do use it in spurts!

I first purchased the TI in 1982 primarily as a games machine and to help my two boys (pre schoolers then) learn the basics of education. The TI is still the best for this in my opinion! I also used it in my employment to work out some electronic problems. Then, it would end up sitting there gathering dust, not being used by anybody.

All this time I maintained my membership in TISHUG. Why? I enjoyed reading the magazine. Even if I did not use the computer from one month to the next, it was still interesting to read what others were doing. I did not get into programming, and I do not see that I ever will other than in simple basic. I have not attended many meetings, mainly because I live out of Sydney and my week ends are taken up by my own and my childrens sporting time.

So what am I leading to; well I would not like to see the club fold! How can I help? Having read in the Digest over the last year or so that you (the committee) would like to know what it is that we, the members, want, I thought that I would come forward with a suggestion. There is no doubt that there are members out there like myself who would like to put their TI's to use but do not know how to go about getting the best program to work in the most efficient way. I feel a little embarrassed to ask for help because the ones I would ask are into "higher" things to do with the computer. Are there others out there that feel as I do? Please believe me, this is not a complaint or bitch against the club or any member. If I was unhappy with TISHUG I would not have remained a member for so long.

Now, having said that, I would like to suggest that members be invited to write in with their own problems and that the members of TISHUG could, via the TND, be able to give their solutions. This would help that person, and at the same time maybe get some renewed interest in the TI and the club. I could like to start the ball rolling.

Here is a use I would like to put my TI to. I am in the local sport fishing club and I would like to use the computer to keep a record of the fish captures for the month, calculate the points value of each capture, sort them into their various categories and place the highest point scoring fish to the top of the list. Then compare the monthly captures to the club record for that fish and if it beats the current record to update the record list.

The formula for working out the points value of a capture is:

$$\frac{\text{weight} \times 100 \times \text{fighting factor}}{\text{line class}}$$

For example, weight = 2.9kg, f/factor = 1.0,
line class = 2kg.

$$\frac{2.9 \times 100 \times 1.0}{2.0} = 145$$

The information is supplied in the following format:

SPECIES	WEIGHT	LINE	DIV(ISION)	*ANGLER	POINTS
Aust salmon	1.9	2	1	J.McEWAN (L)	114
Tiger shark	200.0	15	4	P.SHARLAND	1066

*L:lady, J:junior

Of course the points would not be worked out when the capture form is handed in to me. The above format

is the way that the results need to be printed out for the club news letter.

I have tried using Multi Plan for this task, and it did work. However, I could not get it to do all the things I wanted. This is where the more adept members of the club may be able to come to my assistance.

I hope my ideas are of use to the club, and I look forward to reading of any way that I can utilise my venerable old TI99/4A for this task.

Kind regards Michael Ball

O

continued from page 6

The print program for the fanfold page consists of ten PRINT #1: steps and 14 are required for the A4 sheet as its height is slightly greater. For the sake of convenience this print program should be saved under the name of 'ADVANCE'.

Trial Printout

- 1) Load the CARDIN file.
- 2) Press the 'P' key and type in 1 to the number of overstrikes prompt. Type in PIO.CR to the printer device prompt.
- 3) When the file has been printed, exit the PICASSO program but do not turn the printer off.
- 4) Go to BASIC and load ADVANCE.
- 5) Run the program, exit BASIC and reboot PICASSO.
- 6) Load in the CARDOUT file and repeat step 2.

Remove the sheet from the printer and fold it over twice as described previously to check that the location of graphics is where you had planned them to be. If you prefer to relocate them slightly, mark on the folded sheet, their new location. The reason why the template indicator graphics were left intact earlier should now be evident, and that is, for printout to screen orientation.

Final Printout

- 1) Load PICASSO.
- 2) Load the CARDIN file.
- 3) Relocate the graphics if required and remove all template indicator graphics.
- 4) Press the 'P' key and type in the number of overstrikes you require. This is for the finished product so you will want the graphics printed as crisp as possible. The number of overstrikes range is 1 to 7, the number you select will depend on the condition of your printer ribbon.
- 5) Type in PIO.CR for the printer device prompt.
- 6) As per the TRIAL PRINTOUT, after the CARDIN file has been printed:
 - (i) Exit to BASIC.
 - (ii) Run the ADVANCE program.
 - (iii) Reboot PICASSO.
- 7) Load the CARDOUT file and repeat steps 3 to 5.

Concluding Comment

There is really no comparison in card production ease, between my technique using PICASSO and Larry using PAGE PRO 99. However Larry's copy of PAGE PRO and its utilities are to my knowledge the only one purchased by a member of our TISHUG community at the present time. There may be other members that have the programs on order of which I am naturally not aware. Until such time as we can order more, my method, although a little cumbersome, should bridge the gap, at least for the coming festive season.

It is very easy to be biased about card making programs considering that fairly sophisticated ones have existed for some time for the IBM. I have seen some cards produced with the IBM PRINT SHOP and quite objectively the cards that I produce using PICASSO and those produced with PAGE PRO 99 are comparable in quality, but ours for that over used cliché, a fraction of the cost. I will not pretend to compare our programs with the VENTURA PUBLISHER or ARTS AND LETTERS, they are for professional use and are simply not in the same league, but once again we are not paying over \$2000 to produce very presentable artwork.

O

PRK and PRG

Daniel Harris (September 1988, p10)

I suppose most users have tried to use Personal Record Keeper to do sorts, and I have more or less succeeded in doing that and all the correspondence is out and generating interest, so now is the time for a little conservative skiting about how the monster was created.

Firstly, and worstly, if you give the PRK a lot of files to crunch it will take a long time, decide two files are not worth bothering about and move them to the end of the list. Once this happens, one is apt to swear at it, maybe use the system as a football or something, or otherwise do one's block. The worst thing one can do is tell the system to re-sort. It will burble away for a day or so! Instead, sort on another key, this will only take half to three-quarters of an hour. Then tell it to do the sort you want. The wise move is to save the other-key sort as well, as it took half an hour or so to generate, and might be useful. The re-sort will take only half an hour. It is a little less strenuous than watching shards of plastic carom off the walls and ceilings, takes a bit longer, but will get the information the way you told it to go. Those bodgy files will now be where they should have been in the first place. Now your printer will be enough for any subsequent work, and you will not be using scissors and glue to patch up a bodgy listing.

The next thing that can happen with the PRK is that you try too many keys, or make the fields too big. Try the number 12,000 as a rough guide. The number of files times the characters (or numeric fields) in each should be a co-factor of an integer to give the product 12,000. This means that if you have 15 fields of 15 characters, you are limited to 12,000 divided by 15, then again divided by 15. If you want to use the Personal Report Generator it may be a good thing to keep to about 60% of that, or 12,000 divided by 4, times 3, or 9,000. So for a file subsequently to be manipulated by PRG, you count all the characters in all the fields (including numeric fields) and divide THAT total into 9,000 instead of 12,000. This will save DATA ERROR when you try to put it into PRG to chop files out, after the sorting. The PRK will chew through most sorts in 35 minutes, UNLESS you try to re-sort something that is already nearly sorted. For some reason that takes longer. For a list of 60 files or less it only takes a minute or two. I have successfully got it to chew up, swallow, and digest 363 files, which is what took it the 35 minutes, doing alpha suburb, then adding the postcodes which are a snap once the thing is in alpha suburb order like the postcode book.

That reminds me of another thing; PRK does not like blank fields when sorting. It is apt to move files with blank fields to the end of the list. Back to the sort; I could then do numeric postcode sort, although the files I was using put it in as characters, otherwise the PRG would add the postcodes and insist on letting me know their totals, likewise telephone numbers have to be characters not integers or you get total telephone numbers and digital overflow messages and whatever befalls of trying to add lists of telephone numbers!

The files which were postcode-sorted could then be chased through a street directory as easily as the alpha-suburbs could be chased through the postcode book. Having found all the bodgy suburbs, access-wise, it was simple enough to do the deletions, then resave the purged files. These could then be category-sorted, printed, and chopped for saving as separate categories using PRG. One has to use a printout to plan this, the PRG makes it all too easy to obliterate files! Separately-saved category files can then be augmented to take all the addressing details. It is then simple to put envelopes into a printer, having got the format as a report in PRG, and press the button for each envelope. One learns to count the fields and take the printer off-line at the end of each addressing. This will not be a major problem I am sure. The category-sorted addresses then get the message for each category plus the address as per the envelope. At 180

plus envelopes, plus four printed foolscap pages each, that was 180 times 5 equals 900 plus bits of printing to do. I suppose I might have been a teensy bit balked if I had worked that out before doing it!

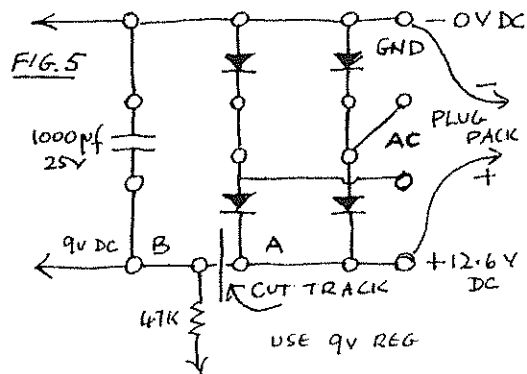
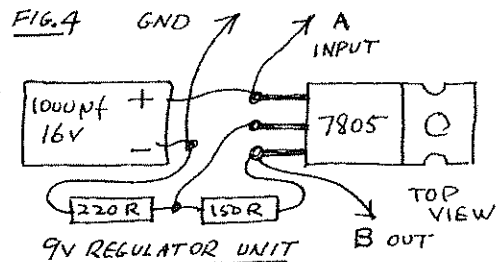
So there we have the way to use PRK for really massive chunks of data. Firstly, use very conservative little files to sort the material and enable you to delete what will be too inconvenient to work with, get the supplementary data in (such as postcodes) re-sort, edit, save, and re-sort to categories, and use the PRG to split the files before augmenting them to take addressing or other data using the PRG to add fields, and to delete files when not part of the list being worked on. The data needed can then be added out of PRK and printed as address labels using PRG. (PRK equals Personal Record Keeper and PRG equals Personal Report Generator.)

The other topic is; DIABOLICAL RIBBONS. I wonder if any other users have a Diablo Printer? It lives up to its name if you try non-genuine ribbons, the point of essence being a little slot in the right of the cartridge with a brass or copper leaf spring in it which takes up the torque during printing and changing directions. Without it, and the substitutes tend to lack it, the ribbon slips up or is thrown up, and only the tops of letters are printed! Genuine Diablo Ribbons cost \$18 whereas the substitutes cost only \$9, but it would seem as if the "pact" must be taken seriously! Watch it if a dealer tries to sell you a cheapo ribbon. It might just work if it has that slot on the right with the leaf spring.

continued from page 10

Another approach suggested by Lou Amadio aimed at using the more common 12.6VDC 300ma plug packs by running the interface at 9V regulated DC, and on the assumption that members may already have such packs available. The new price of 12V 300ma packs is around \$15 to \$20 odd dollars, which is more than the 16V 900ma ac pack as recommended above. If you have to purchase a pack, this is not the way for you to go.

There are no 9V positive regulators available so a 7805 in a TO-220 case can be doctored by using two resistors to lift the output to 9V. The input voltage to the regulator must be greater than 12V DC. Details are given in Figure 4.



I have built the RGB Interface and powered it by both the 16VAC plug pack and the full wave centre tap circuit and both gave a bright clean screen. However, I have not been able to check the 9V version in regard to its performance on the screen, but the regulator circuit does output a steady 9V as predicted.

Regional Group Reports

Banana Coast Regional Group Coffs Harbour

The Banana Coast Users Group started in 1985, when one of our current members moved to this area from Wollongong to live. Not knowing any other TI99/4A members in this area, he sought approval from TISHUG to form a users group and advertised in the local paper. From that single advertisement, the heart of the group was formed into a strong niche of mainly cassette users who share their problems as well as programs. Being a long distance from Sydney, it is difficult to attend the monthly meetings. With the best News Digest (TISHUG News Digest) as a backup and other Magazines such as MICROpendium, it all made computing a little easier in our remote locality.

With the availability of cheaper disk drives and the invention of the Mini-PE system by a fellow TISHUG member, the group soon graduated to disk systems, which made using the TI99/4A a pleasure. While some of our members are complete game freaks, others dabble in programming with some limited success, being able to choose from a wide variety of languages such as BASIC, Pascal, Forth, Logo and, of course, the sometimes dreaded Assembler.

Some of our members delight in re-writing programs intended for other machines, mainly games, so they can be used on the TI99/4A. This is a real learning process in programming in BASIC. While most of us have moved into a Blue Machine Area, the interest in the TI99/4A is still very strong and we eagerly await the monthly arrival of our TND magazine, which is always filled with interesting articles.

Having attended the TI-Faire in Brisbane some years ago, we wish TISHUG all the best in the coming event and hope it can stimulate some more interest in the TI99/4A. This machine, which was years ahead of the rest in technology and programs, still has not been surpassed by the newer machines in the programming area. The availability of many types of upgrades, such as RAMdisks, EPROMs added to RAMdisks and Hard disks has kept the TI99/4A in the forefront of the computing world.

The Glebe Regional Group from Mike Slattery

The Glebe Regional Group was founded from the Marrickville group originally run by Shane Andersen. At the time of formation of the Glebe group, Shane was the editor of the club magazine, the bulletin board sysop and the convener of the Marrickville group. He used to hold the meetings in his flat which was usually covered in articles in preparation for assembling the next issue of the magazine. All this was proving to be a problem to Shane and he expressed an interest in dropping the Marrickville group altogether.

He enquired of the then members of his group if anyone would be interested in taking over the running of the group, as he did not want it to simply fold. Although I had only been a member of TISHUG for about six months, I agreed to give it a try. The meetings were then transferred to my current address and were originally held on the Wednesday evening following the monthly meeting. After a while, they were transferred to a Friday evening and finally to the Thursday evening.

At first there were only a few of the Marrickville group people who attended the meetings, but eventually the group expanded to about 20 or so members who attended on a fairly regular basis.

Over the years, the group has followed the fortunes of the parent group. As TISHUG expanded, so did the

Glebe group, but in recent years, the group has declined to about half a dozen regulars, following the lead of the parent group.

In its heyday, the Glebe Regional Group was a very active group, with members like Peter Schubert, perhaps the club's most successful and prolific hardware developer, being very active. Other prominent members were Shane Ferrett, John Paine and of course, the current co-ordinator, Dick Warburton.

The Glebe Regional Group meetings were never structured like some of the other group meetings. We usually started off with someone showing off a new program, or perhaps wanting to solve a programming or hardware problem. Over the years this has proved very useful to the group members in need of assistance in any of these areas.

Due to the declining numbers, the meetings have evolved to mainly discussion evenings. Also as most members over the years have purchased other computers, we sometimes discuss and compare the different computers available to the members. On several occasions, the members have had the opportunity to compare identical programs on two different computers, namely an Apple Macintosh and an IBM.

I would like to take the opportunity to thank those members who have attended the Glebe Regional Group meetings over the years and have helped to give me the enjoyment that I have had in running these meetings. I would like to think that the group would last another 12 years but I doubt that it will in its present format, although I am sure we will get at least a further 4 or 5 years before the wheels fall off completely. Not a bad effort for an orphan!

Good luck and best wishes for the TI-Faire and the future.

Illawarra Regional Group

The Illawarra Regional Group is based in the city of Wollongong (third largest city in NSW) which is a very picturesque area, situated on the coast and bounded by an escarpment to the west. Along the escarpment there is rainforest and so the people of Wollongong are close to both the bush and the sea. The largest local industry is the Port Kembla Steelworks and this is a very large integrated steelworks which takes the raw materials (local coal, imported iron ore) and produces a great variety of finished steel products. The area also boasts the University of Wollongong which is rapidly gaining a solid reputation as a first class university, both in teaching and research. Members of the Illawarra Regional Group come from as far north as Helensburgh (35 km), as far south as Nowra (80 km) and as far west as The Oaks (45 km).

The Illawarra Regional Group started in 1983 as a group in its own right, meeting at the premises of Shop 4 in Corrimal, a suburb just north of the Wollongong city centre. Shop 4 sold computer equipment, including the TI99/4A and were keen to foster user groups for all types of computers. In about the middle of 1984, the proprietors of Shop 4 moved into smaller premises in the centre of Wollongong and it was no longer possible to meet in their shop. For a few months we met in the home of Bob Montgomery and then formed into a proper group to raise the money to hire the hall of St. Matthews Church in Mangerton, south of the city centre.

At about this time we became a regional group of TISHUG as there were a number of us who were also members of TISHUG. Our numbers peaked at about 30 paid members and were about 20 for quite a while. In 1987 we changed our meeting place to the Keiraville Public School, west of the city centre and met there from 1987 until 1991. We had some very memorable Christmas Parties over the years with whole families coming along.

User Group Updates

TI99/4A Brisbane User Group

The user group in Brisbane has about 40 members on the books, many of whom live outside of Brisbane. The meetings are held on the last Friday night of the month and attract around 15 members. The topics of discussion vary. There is usually a lot of interest in the table where we sell software. When a new shipment arrives it is like Christmas and everyone gathers around to see what is available.

It has been in this area that much of our effort, and finances, have been dedicated in the past year or so. We now have a reasonable range of the most popular software from the US. Some pieces of software have hit the mark more than others. There are many members who have found Page Pro from Asgard Software quite useful. The software table now boasts a wide range of Page Pro support disks: fonts, pictures etc.

When it comes to hardware projects, the Brisbane group have had some successes. Most obvious at the Faire will be the Cadet Console Expansion system. This device was designed by Col Christensen and is aimed at giving the console-only user a little more 'horse-power'.

Two other projects have been prominent. The first was an RGB interface for the console. This device replaces the RF modulator and allows the console to be connected directly to a high-resolution monitor. Its creator, Hillary Giffory, seems to have been able to produce one of the best RGB interfaces around. The other project is a speech interface for the PEB. Chas Bagley orchestrated this one. The card accepts the speech synthesizer and allows it to be placed in the Expansion Box. The concept has been around for some time but this is probably the first to be produced in Australia.

On the software side, Col Christensen has certainly made his mark. Hardmaster, distributed commercially by Asgard Software, is one of the most popular hard disk sector editors around. His cassette tape orientated free-ware releases, Tapemaster and Tape-It, have certainly found wide acceptance.

In a time when the future for many user groups looks uncertain, the Brisbane group seems to be faring well. Like most groups, a lot of the work has been done by a few members. It is pleasing to see now though, that some of the brand new members of a few years ago are beginning to take on some of the responsibilities.

In the history of the Brisbane User Group, one of the most significant events was the TI-Faire in 1988. TISHUG is to be congratulated for their efforts to join the 'select' group of faire hosts and all the members of this user group wish the organizers of the 1992 TI-Faire every success.

TIUP - THE TI-99 USERS of PERTH (Inc)

TIUP, which was founded in 1982, celebrated its 10th Birthday in March of this year. Considering Perth's distance from the rest of the TI99/4A world and the departure of the original members for bluer pastures, this is really quite an achievement.

The group currently boasts around 20 odd members (a far cry from our peak of well over 100 in the early 1980's), of whom approximately half regularly attend the monthly meetings. We have one member who has recently returned from a contract in Pakistan and another who lives in Derby, the regional centre for Western Australia's vast Kimberly Region, so the TI99/4A influence is felt over a broad geographical span.

After the initial fall-off of members, our numbers have remained pretty much constant for the past few years, with the new members balancing out those who do not renew.

continued on page 25

We have always had a good mix of members interested in programming and in hardware as well as those who just want to learn about computers. I can remember many fine lessons on programming in BASIC by Bob Montgomery and George Meldrum and there were even attempts at learning assembler language with the help of Geoff Trott. If anyone wanted a program, Rolf Shreiber was usually able to provide it as well as suggesting some interesting ways of doing tricky things, like loading assembler programs from tape, and of course Lou Amadio was always handy with a soldering iron.

We have accumulated very extensive libraries of books, software and modules for our members to borrow. There was quite a deal of interest in these for a number of years, but not much lately. Now we meet at the home of Geoff Trott, in the dungeon, through the garage. There we discuss advanced topics of how to use Funnelweb, Multiplan, TI-Artist and all the other great software that is now available for the TI99/4A. Geoff has an 80 column card for good pictures, so a number of us are waiting for the TIM cards to arrive (sigh!). We hope that TISHUG will carry on in its present form for the foreseeable future and hope the Faire is a great success and just reward for all the work put into it by the organising committee.

SUTHERLAND Regional Group

The Sutherland Regional Group has now been conducting monthly meetings for a total of six years, since its inception in 1986.

A fixed meeting place has been established at the home of Peter Young at Jannali.

The group is based in the Sutherland Shire of Sydney, which could be described as the birthplace of modern Australia, as it includes the Kurnell Peninsular on Botany Bay where Captain James Cook first anchored the Endeavour in 1770. Other features of the Sutherland Shire include the Cronulla beachfront and Royal National Park, the second oldest national park in the world, next to Yellowstone in the USA. The Shire is bounded by the Georges River in the North, the Holdsworth Army Reserve in the West and the Illawarra Escarpment in the South.

The group possesses a wide range of expertise ranging from Technical Officers in communications to Administrative and Accounting qualifications. The numbers attending meetings are in the range of six to eight persons. Most of the systems owned by these people are fully expanded, with members collectively possessing a multitude of RAMdisks of varying sizes, two hard disks and a real assortment of floppy drives ranging from 80 track, half height to 3.50 inch configurations. Derek Wilkinson has adapted an IBM keyboard to interface with his TI99/4A console, which sits somewhat in the background. Spare parts are the order of the day, with most members having more than one machine.

The group also has its own Diagnostics equipment to assist with fault finding, as and when they occur in either the PEB or console. The format of all meetings is quite informal, with a wide range of other topics being discussed:

- Automotive engineering
- Aviation, including photographs of all the local air shows
- Australia's two carrier telecommunications system

Many personal friendships have developed and continued over the years as a result of our involvement with the TI99/4A computer. Interstate and overseas visitors are more than welcome to make contact with us at any time, with a view to increasing the interchange of information.

Peter Young

o

**HOT
STUFF**

GIF MANIA

**Breathtaking Graphics are Yours
Free for the Asking!**

Imagine having the world's largest collection of clip-art and scanned images at your fingertips? Sounds like a dream come true, right? But what if dreams did come true... if they did, and they sometimes do, this dream would be called GIF Mania! Using GIF Mania industry standard GIF files can be viewed on an ordinary TI-99/4a. Not only can you view GIFs you can also convert them into a regular TI Artist picture... and from there the possibilities are endless. More than 1 million GIF files currently exist and are available free from online services, local bulletin boards and user group libraries. A small collection of GIF files is included. *Requires 32k, disk system and either an XB or E/A cartridge. Compatible with the Geneve in GPL mode but will not take advantage of it's advanced display capabilities.* **Only \$14.95**

TI ARTIST PLUS!

The Champion in Graphics Design!

More than just an ordinary drawing package, TI Artist Plus! is a complete drawing system that consists of 6 dynamic design modules. With these powerful modules virtually anyone can create, edit, transform, scale, print and present the most dazzling of graphics. Animated sequences can be even developed using it's unique movie editor. The innovative point-and-shoot menu system makes using TI Artist Plus! a breeze. Not only is TI Artist Plus! a graphics powerhouse, it's the #1 best-selling and supported graphics program ever developed for the TI-99/4a. *Requires 32k, disk system and either an XB, MM or E/A cartridge. Compatible with the Geneve in GPL mode but will not take advantage of it's advanced display capabilities. Supports most printers (call for printer compatibility information).* **Only \$24.95**

SOUND FX

**Make Your Computer Sing, Talk and Scream Without
Any New Hardware! Now That's Incredible!**

The most incredible thing happened in our labs... we discovered how to make a standard TI-99/4a and Geneve 9640 play true digital sound through your existing monitor/television, without requiring you to purchase any new hardware. We call our discovery Sound F/X -- you'll call it amazing. But we didn't stop there! Taking our breakthrough one step further, we developed a vital conversion utility that links Sound F/X to the outside world. This sophisticated utility (included free) actually converts industry standard sound files recorded using an IBM, Macintosh or Amiga into native Sound F/X files. Features include a convenient disk cataloger, a variable playback speed control and memory buffer analyzer. *Requires 32k, disk system and either an XB, TIW or E/A cartridge. Compatible with Geneve in GPL mode.* **Only \$14.95**

SUPER DEALS

Hurry... These Super Deals Won't Last Long!

Super Deal #1: Get all three best-sellers, GIF Mania, TI Artist and Sound F/X, for the incredibly low price of **\$49.95**

Super Deal #2: Get Sound F/X and 6 disks packed with sound bytes (sold separately for \$9.95) for only **\$21.95**

Super Deal #3: Get TI Artist Plus!, GIF Mania and a two-disk Artist's Companion (selected randomly) for only **\$39.95**

TEXAMENTS

Serving the TI Community Since 1984

Office: 516-475-3480

53 Center Street, Patchogue, NY 11772

BBS: 516-475-6463

Please add \$3.25 for domestic first class and Canadian delivery, \$8.50 for foreign air mail delivery. Orders are typically shipped within 48 hours of receipt. C.O.D. orders must be placed by phone. We do not accept credit cards. For more information please call or write.