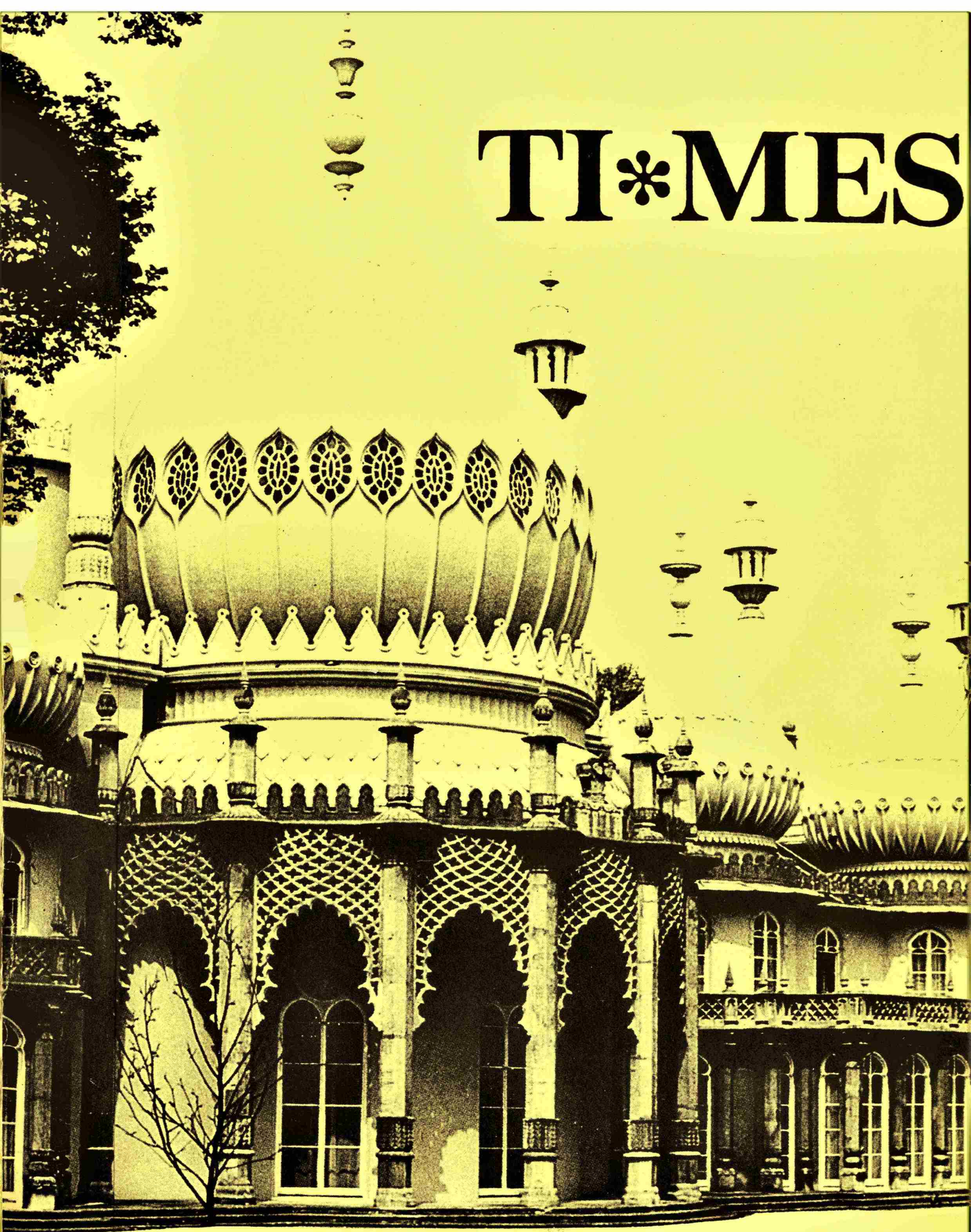


TI * MIES

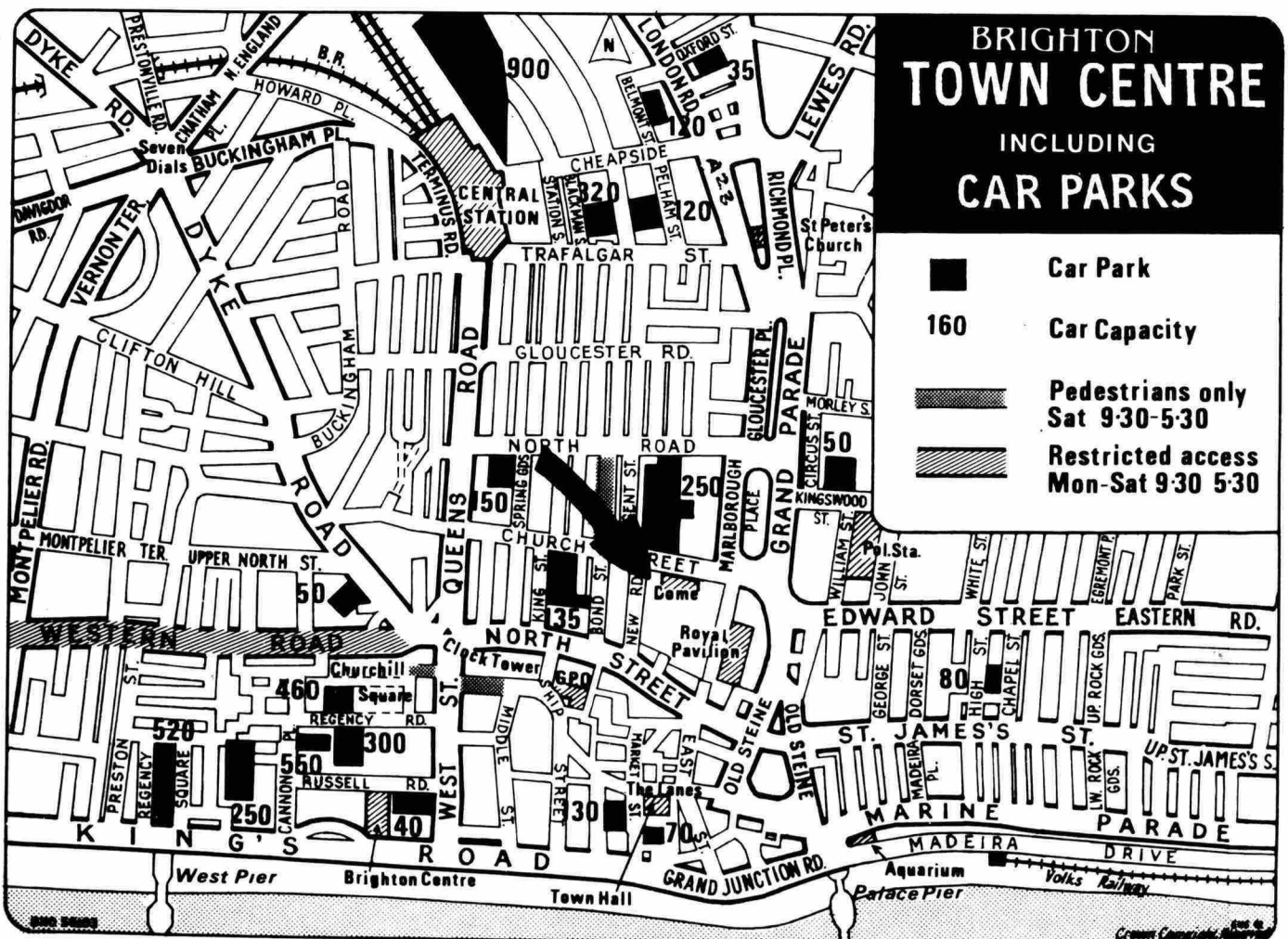


TI Users Show Edition
SPRING ISSUE NUMBER EIGHT

INSIDE THIS ISSUE

2	Editorial.	
3	RAMBLES REVIEWS	- Stephen Shaw.
10	MINIMEM demonstration	- " "
13	MANCHESTER MUSINGS	- John Rice.
19	POKEING with MINIMEM	- John Roe
22	BASIC Character identification	- J. Hawkins.
23	BASIC Beginners feedback	- J. Hawkins.
24	TI SOFTWARE REVIEWS	
25	DIY HARDWARE	- Derek Ford.
27	Review of Library programs.	- Graham Hilton.
28	Your letters	
33	Classified adverts.	
34	RANDOM EYES	- Graham Baldwin.
37	Getting the most out of TI WRITER	- Allen Burt.
40	BASIC PROGRAM Wordbug Bunny	- Sam Nash.
44	RAMBLES ON FORTH start here	- S. Shaw.
48	FORTH disc formatting	- Philip Marsden.
52	Area Contacts.	
52	HINTS ON CALL LOADS	
53	EXTENDED TUTOR	- Tony McGovern.
56	TIPS FROM TIGERCUB	
59	TECHNICAL TALK Rest button	- Viv Comley
61	STAR GEMINI 10X Review.	

British TI99/4a Users Show Dome Corn Exchange The Royal Pavillion Estate



Brighton Sunday 28th April

Cover: Royal Pavillion Brighton.

Welcome ----- to TI99/4a EXCHANGE

TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES
SPRING 1985 NUMBER EIGHT

40, Barrhill, Patcham, BRIGHTON, East Sussex, BN18UF. Tel: 0273 503968



SECOND U.K. TI USERS SHOW AT THE PLEASURE DOME...

YES! the show is on and what a setting too. One of Britains most well known resorts and its famous Palace The Royal Pavillion will be playing host to the Nationwide TI Users group get together. TI99/4a Exchange will be at the DOME Corn Exchange on Sunday 28th April.

This time the event is being well supported by many well known personalities. You will also have the opportunity to see all the main supporting traders of TI hardware and software under one roof.

Those of you who attended the Manchester event will know how well supported the TI99/4a is. This time even more interest is being shown by the media. Not only the press but LIVE broadcast and interviews by the BBC. Calling all local TI99/4a group leaders, we will want to hear from you out there. If you wish to represent your local area TI User Group for the broadcast get in touch with us soon.

The program for the event will be published as soon as we have your response as regards representation for the Discussion group corner being set up. The TI Users Show will be an everything event, packing as much as possible into the day. Bring the family. Mum can have a day off too as we have even arranged full catering facilities at very reasonable prices. By the way would you like to participate in a swap shop? If so then bring along those unwanted cartridges, books etc to exchange with others.

How to get there.....

If you go by Coach the terminal is almost on the doorstep. The mainline

railway station is 7 minute walk and if you choose to go by car then directions are clearly marked on major roads out of town. Just follow signs "The Royal Pavillion".

TIHOME Club closes...

Ray Hodges Assocs Ltd, better known as TI Home Computer Users Club or TIHCUC, have announced in their Winter/Spring publication that members subscriptions will not be renewed. Over 2000 TI Users have been dumped. In a statement the public relations company appointed by Texas Instruments to represent TI Users said "its just not economical long term. We have of course, honoured our obligations.." What hope have Ray Hodges Associates when they say "...there just aren't enough of you to keep the Club viable." Mr Dicks sums up in his column "Well, heres the end of another load of rubbish" Hmmm!

*** Thankyou ***

We can only be glad that with your loyal support your TI*MES will carry on. In this issue we have had to make room for new editorial. You will note some more new names. Alas Howard Greenberg who has been over to the USA and Peter Brooks who is up to his eyes in work are not in this issue. However Peter is doing a great job representing TI99ers in the computer press. Many long hours are spent sorting and editing information so that TI*MES keeps abreast. It is always hard when some writers are not included. We will always try to keep a broad readership. Our special thanks to all who continue to maintain very high standards. The TI99/4a lives on as an interest everyone can share. ED C&A.

TI99/4a Exchange TI*MES newsletter is supported only by its subscribers. This TI users Group is INDEPENDANT of Texas Instruments and is completely non profit making. TI*MES is published quarterly, JANUARY, APRIL, JULY, and OCTOBER months. Editorial etc is provided by group members, other user-groups and other related sources. Views expressed are those of the writer and not necessarily those of TI99/4a Exchange. Whilst efforts are made to ensure accuracy no responsibility can be accepted by TI 99/4a Exchange as a result of the applying of such information found within the pages of TI*MES. You are invited to contribute copy for publication in TI*MES. If you would like to make a contribution please submit copy on A4 only this MUST be typed with a disk or tape if a program is included. Unaccepted material will be returned ONLY if accompanied by a S.A.E. The editors reserve the right to refuse advertising.

RAMBLES

REVIEW SECTION

There is a remarkable amount of material available for TI owners, but with supply being so limited, there is often a choice of buying blind or not buying... and at the November show, a number of owners did impress upon me their dislike of buying without seeing.

So, lots of reviews for you to see what effect some items have had on ME. These are very personal views of course but I shall try to make them as helpful as possible!!

And at this early stage, an APOLOGY to readers expecting reviews of two LOGO books in the last issue: these WERE written and submitted, but cut out by Clive, who ran out of room. This submission of RAMBLES is exceptionally long, so I will cross my fingers and hope Clive leaves them in.... to be found somewhere in the following pages!!

SXB (Super Extended Basic) by J & KH SOFTWARE

Media: DISK & Manual. Requires disk system & 32k & Extended Basic.

Price: about US\$85 plus UK duty and vat

Source: No UK supplier.

Available from Unisource, Lubbock or Tenex

J&KH Software, 2820 S. Abingdon St., Arlington, Va, USA, 22206

SXB is a collection of 100 utilities in machine code, which you can call up in your own programs using CALL LINK("NAME", INPUT/OUTPUT LIST)

The main use of SXB (& the one which makes it worth the price tag) is to build your own data base program: if you wish, a Personal Record Keeping Program which does what the module doesn't, faster, and uses the 24k area of the 32k ram. You can design a general purpose data base or a specialised one, the choice is yours.

The manual is divided into various sections, the first of course dealing with setting up the DATABASE: the actual data is held in a single string array, and various utilities are provided for handling the data in the array. The array is filled in the normal manner, by inputting data, formatting it to fixed length fields (utility provided) and concatenating each "page" of data into one element of the array. There is a pretty fast sort utility which knocks spots off the PRK module (not difficult!) and utilities to insert, delete, and replace individual items. The general purpose "update" is also available. A utility is provided to build lots of specialised data bases from one general base... all very quickly.

To assist you, there are STRING ARRAY ROUTINES, including encryption routines, and the facility to view on screen the contents of a string array: even after a BREAKpoint. The facility is provided to change characters, eg all dashes to spaces, all lower case As to upper case As and so on.

There is a group of STRING ROUTINES, very useful in setting up the fixed length fields, or encoding strings of spaces or zeros. There is a more powerful SEG\$, and a utility to translate ASCII characters to their HEX values ("A"="41").

If memory space is short, there are INTEGER routines which allow you to pack four integers where normally the TI stores one number. This is useful for data packing, but does NOT give the speed increase associated with integer work.

Those CALL LINKs take some time after all!

Possibly of greatest interest are the VDP routines which are applicable to most ExBas programs. These include windowing commands, the ability to read the screen text 256 characters at a time (a giant ACCEPT AT!), Various display utilities. You can define (and get to store) up to 31 characters in one fell swoop for really fast game startups. Change colours VERY QUICKLY. And there is a nice true lower case available.

continued.....>

SXB, continued...

Incidentally, once SXB has been loaded, all these routines are available for any subsequent program you load, OR can be used in direct command mode, so you can type in a program with true lower case on screen.

→ The SXB disk is heavily fortified with copy protection devices!

If you worry about crashing the disk, J&KH will sell you a back up disk for \$15.

The price does not just include the above. You also receive, after registering your purchase, the first six issues of SXBrief (total 24 pages) which mainly gives a disk database program which is not only useful but shows how SXB works.

I did not mention above, but there is 256 bytes free for extensions, which are loaded by short and simple ExBas subprograms (sing CALL LOADs). Although only one can be resident, it takes very little time to load by using for instance CALL MYROUTINE etc.

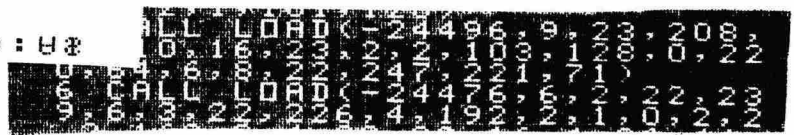
SXBrief gives some extensions. There are extra display and data handling routines. My favorite routine in this set is a screen dump utility... yes, another screen dump, and not entirely in machine code, but still pretty fast.

And with some interesting variations. The three dumps shown below were all prepared with this routine... one normal, one upside down and reversed, and one inverse. It is the first utility I have seen which can do this. The dump does not have to be text: it could be any user defined graphics.

```

00 OPEN #1:"PID.CR"
01 CALL GRAPH
02 NEXT T
03 ACCEPT AT(T,2)SIZE(-28):A$

```



Hope they copy well. Thats not all... you are also entitled, as a purchaser, to subscribe to further issues of SXBrief, at a cost (to the UK) of US\$15 for 12 issues. Examples of what is in issues 7 on: double size bit image screen dump and extra display and data handling routines. The routines are short, but if you wish you can also buy a disk (US\$15) which contains the routines in the first 15 issues of SXBrief, in CALL LOAD format (MERGE files), in Source code, and in Object code.

An expensive item yes, but taken as a whole package, well worth the money. You should however be prepared to do your own programming, and to look very closely at the routines offered.

UNISOURCE take Access (called Master Charge) and Barclaycard (called VISA) and you MAY be able to phone your order in:

Texas 1-806-745-8834 or write
P O Box 64240 Lubbock Texas USA 79464 (Catalogue US\$5.00)

JOYSTICK: PROSTICK II Price: Around 22.00
Supplier: Arcade Hardware

A very neat palm-held joystick with a light firing button operated by the INDEX FINGER (the strongest finger you have, though the button is very light!) and suitable for left or right handed operation.

This joystick is the first I have seen which allows you to easily and deliberately obtain diagonal movement. This is by means of a simple mechanism called a "switchable gateplate"(tm), which allows you the option of permitting only the four cardinal directions, switching out diagonals (useful for Munch Man &c). Complete with TWO WAY adaptor which permits you to use it as joystick one or two (or add a second joystick for two player games). Metal shaft construction and a joy to use. Highly recommended.

SUPER SKETCH drawing tablet.
Medium: Cartridge Required: Console only
About £65.00 from Arcade Hardware or Parco Electric



When graphics tablets can easily cost over £300, you don't expect too much for £65... in fact SUPERSKETCH is a remarkable product at the price, and the most significant drawback is a feature of the VDP in our consoles:

When in bit map mode (hi res graphics), each character on screen is divided into 8 rows of 8 pixels. For each row of 8 pixels, only two colours are permitted. When using SuperSketch, this can cause "bleeding" problems when you try to place three colours in a row! With great care you can reduce the visual effects of this but it is a significant drawback.

SUPERSKETCH uses a movable arm which can be used to trace a drawing, or merely used 'freeform'. A full palette of colours is available, and a good choice of 'brushes'. As drawing straight lines can be difficult, special facilities are provided to assist you. There is a "fill" command also, but if there is even a one pixel gap in the shape, the ink will run out and fill the screen!

Once created, pictures can be saved to cassette ONLY, not to disk, not to printer, although owners with 32k ram may be able to patch in a screen dump program of their own via an interrupt switch.

Essentially a toy, Supersketch is a very interesting peripheral for the 99/4A, and seems strongly enough built to be used by the younger members of the family, who could find it to be a very expressive medium. A colour tv should be considered essential for this item! although it can be used with black & white.

...LATE EXTRA:

News just reaches me of two disk based programs which supply the missing Super Sketch utilities:

DATA FLEX SOFTWARE (USA) have one disk which enables you to dump a Super Sketch Picture to an Epson/Gemini/Okidata type printer and an extension to this which lets you store the pictures on disk. Examples have been produced which are very acceptable.

These disks require 32k ram and will load with either MiniMem, Ed/As, or Extended Basic. An interrupt switch has to be wired across the console side socket -

connecting LOAD and SYSTEM GROUND makes the console jump out of the module into the machine code dump routine: which can also be used with basic programs, or modules (such as video chess, car wars, munch man &c.)... what a way to record your high scores!

The main disk will work without the interrupt switch to dump Basic programs, and can also on its own dump Basic program screens to disk.

Not yet available in the UK..... but I have passed details on to Howard, so contact him for latest news!!

~~~~~

ROMAG CEAF SCREENS. About £20

Standard screens available from ARGOS catalogue stores  
or supply is available from ROMAG direct.

Standard Screens: 9" 12" and 14".

If your tv has a strange screen shape (& is still current) Romag will produce a screen to fit at not extra cost. Larger screens to order, higher prices!

FREEPOST ROMAG CEAF DEPT Blaydon on Tyne Tyne & Wear NE21 5SG  
continued.....>

**Romag "CEAF" T.V./Monitor  
Screen Filter. Easily and quickly  
fitted, using velcro fastenings, to T.V.  
and computer monitor screens.  
Significantly reduces reflections and  
glare whilst improving contrast,  
colours and sharpness of image.**

**Shatterproof laminated glass,  
curved to fit close to the screen.  
Normal T.V. pictures as well as com-  
puter displays can be improved.  
Suitable for both colour and black  
and white T.V's with a 14" screen.**





REVIEWS continued...

LOGO MICRO SCENES: BEACH MICRO SCENES

Cassette or Disk. Requires LOGO or LOGO 2 plus 32k ram.  
by American Software Design & Distribution Co

Availability: USA only unfortunately.

Try: TENEX, P O Box 6578 South Bend Indiana USA 46660

Cassette: US\$18 plus post \$10.75

Disk: US\$20 plus post \$10.75

(Order three copies and the post is still only \$12.75)

VISA (Barclaycard) preferred method of payment:

Quote card number, expiry date and cardholder name as on card.

As this is the ONLY commercial offering I have seen written in LOGO for ANY computer, it has to be worth a look...

The concept involves introducing a child to computing by manipulating a small limited world. BEACH MICRO SCENES was clearly intended to be the first of a series, but no others have surfaced.

Type TREE to place a tree on the screen. Similarly with other nouns such as Jet, Car, Sun, Boy and so on.

Then introduce verbs as in JET ZOOM or BOY WALK.

And for really advanced use try TREE PAINT :RED SLOW to create a slowly moving red tree!!!

In the best educational tradition, cards are provided with the words and a picture, which can be used to remind young users of the words available, and can be used outside the program to prepare in advance a sequence of commands.

This offering is true to the initial Logo ideas, and is flexible and suitable for experiment (eg boy sink, house zoom....DO they work... should they work?). Exploration and Fun are encouraged.

Procedures can be amended and suggestions are made - eg change BOY to GIRL! The weakest procedure I could find was RISE...SET which really should have a SH command in there... easily adjusted.

This program may encourage LOGO owners with dusty unused modules to prepare their own micro worlds....

TI/COLLINS BOOKS: STARTER PACKS 1&2, GAMES WRITERS PACKS 1 & 2

Media: Four paperback books (total 400 pages) with four cassettes.

Availability: At time of writing 12.50 for the set from PARCO.

What an amazing set of books, quite incredible value at this price. EVERYTHING you wanted to know about TI BASIC, with some interesting programs thrown in (and on tape to save you keying them in!).

I did have a problem or two: The first tape was reverse wound, which meant taking the cassette housing apart and rewinding the tape in reverse. One of the books was badly glued and very nearly loose leaf. STILL VERY GOOD VALUE.

The books start right at the beginning and take you through TI Basic:

SP1...setting up. Introduces flow charts. Intro to sounds.

Excellent program CHARLIES (study the program carefully!)

and a really FUN program with the misleading name of KEYS.

SP2...Planning your program. More sound & graphics. Arrays. DEBUGGING.

Programs include a character generator (there was one in the original 99/4A manual, dropped from later editions).

GWP1..Some simple games. Moving characters. Using joysticks. The dense pack theory....! TI BASIC ACCEPT AT routine. DISPLAY AT routine.

continued.....>



GWP2..Dealing and Sorting. Blackjack(Pontoon). War Games. Simulations.

\* BUG in program COMMANDO: line 5050 differs from book and has an error.  
Should read: 5050 W\$=" FINAL SCORE "&STR\$(TN\*M\*((TS\*10)+(GN\*5))/(4-UN))

THESE BOOKS ARE EXTREMELY WELL WRITTEN and assuming no prior knowledge move on to very deep water. Some very good programming practices are well illustrated. Even if you have had your console 3 years and use only ExBas, you can still benefit from these books. I cannot recommend them more strongly: BUY THESE BOOKS. They will make a better programmer out of you if you read them carefully from cover to cover! and will even improve your ex bas programming.

The weakest part deals with cassette loading, but that can be forgiven!

~~~~~  
SAMS BOOKS: The following books are published by SAMS and imported by PITMANN'S, the short hand people. They are available from TI*MES, contact Clive. Very little stock is held in the UK, and there may be delivery delays. Clive will advise you of the costs.

51 FUN & EDUCATIONAL PROGRAMS: Gil M Schechter

90 page paperback book and cassette. With 51 programs on 90 pages, with the first program on page 11, and a few blank pages in between.... you will appreciate that we are looking at short programs with very little in the way of explanation. These are the sort of programs which keep cropping up in these huge collections, short simple and without decoration.

These are just the sort of programs which every brand new computer owner needs to cut his teeth on. Adding the decorations in the form of graphics and sounds, can be quite educational. The programs are in a clear format (not always the most efficient!) and can help a beginner learn some of the basics. If you want 51 short programs and utilities, try this one.

ENTERTAINMENT GAMES IN TI BASIC & EXTENDED BASIC

by Khoa Ton and Quyen Ton. 168 page book and cassette.

20 games in this package, and what a difference to the previous offering:

My favorite game is the TI BASIC version of TI's primitive module "ZERO ZAP", with all three screen layouts! Scoring system modified slightly, and made more suitable for two players. See how TI BASIC can compete with a module (in this case, very well!!!). The Biorhythm program even has a printer dump which dumps the graphics. And the use of the speech synthesiser in HOME BOUND.... this is a good game collection, and if you look at the listings, you are bound to learn something. Program explanations are limited to details of what blocks of line numbers do. Well worth having this one. Recommended.

TI99/4A BASIC PROGRAMS by T O Knight & D LaBatt.

119 page book and cassette. 29 programs or so.

The music section is weak, but this book serves as a good general introduction to TI Basic programming. Each program has a line-block explanation, and one or two modification suggestions. The obligatory 'character generator' program is in there, and the equally common 'blackjack'. Hangman seems to be missing... This comes in between the first two Sams books reviewed above. It would be an interesting addition to the pack of Collins books.

TI99/4A TRIVIA DATA BASE by J F Hunter and G L Guntle

110 page book plus 17 fold out flow charts plus cassette.

The title comes from a game popular in the States, and VERY expensive here, Trivial Pursuit. The PROGRAM uses this idea to demonstrate how to create a DATABASE program. Versions for cassette and disk are included. NB: These programs are in EXTENDED BASIC. The disk files are random access, so you can really learn quite a bit about data access from this book.

The programs do work, but the value is in explaining, in some detail, the actual form and creation of the programs. If you have difficulty with data, have a look at this offering. Advanced programming: assumes some knowledge of ExBas. A detailed look at the listing could teach you quite a bit!

.....>

THE TOOLKIT SERIES: TI99/4A EDITION

By D Dusthimer and T Buchholz. 214page book only: no tape!

Aha...here is hangman! This book starts at the beginning and introduces some very simple TI BASIC programming techniques. There is quite a bit of repetition of the TI Manual.

The SOUND section has some useful sounds: police siren, bell, buzzer, rocket, munching sound!

The section on animation subroutines is likely to be popular: I receive a number of enquiries on this subject. The final program in the section is a 'lander' type program. Programs in this book have 'rem' type statements on the right hand side of the page, alongside the relevant program line, making it quite easy to see what is happening. Not a classic book, but a good book for a new owner.

FORTH PROGRAMMING. Leo J Scanlon. 246page paperback.

This is the next book to buy after you have saved up for STARTING FORTH by Brodie. This book takes you beyond the multiplicity of Forth primers, and digs a little deeper. Describes both Forth 79 and fig-Forth. TI FORTH is in the category of fig-Forth, and a chapter in the TI Manual describes the differences.

A super book which I can only very strongly recommend. Tells you how to avoid FORGETting your definitions by moving the FENCE! and all sorts of things. There are numerous examples of Forth usage-eg wait until key X has been pressed. Useful and helpful.

The Best Book of Multiplan. Alan Simpson.

150 page paperback.

Now that we have a faster Multiplan to hand, it is nice to see so many Multiplan books in the bookshops. Here is SAMS version. In common with the others, it works through Multiplan by means of worked examples. NB: TI MULTIPLAN uses standard Multiplan entry: you can enter from these books with your TI just as you can from an Apple or IBM. Good eh!

Chapter titles in this book include:

Creating a spreadsheet, Functions, Formatting, Ranges and Copying, Editing, Windows and Locked Cells [good title heh!], Sorting, Naming, Printing, and Linking.

I cannot honestly select any one Multiplan book as being notably different from any other! This SAMS book is one you may wish to consider.

~~~~~  
OK LOGO FANS...

LOGO PROGRAMMING: A practical guide for parents and teachers

by Anne Moller, paperback by CENTURY. £6.95. 145 pages.

An impressive title. The text is a fair attempt to discuss Logo from the point of view of a 'classical' teacher. Many of the points Papert felt so strongly about have been utterly missed. As example: the example program "TO MAN" in this book is remarkably similar to one Papert thought was 'inefficient'. Papert showed how 'top down programming' was NOT a difficult concept for youngsters, and if applied to everyday learning could be of benefit: his example was juggling (see MINDSTORMS).

TI LOGO is mentioned!!! This is not a primer nor a program book, but rather a discussion document. You should ideally read MINDSTORMS first! There are however procedures listings in the book, and the section on list handling may be of help. A worthy book, but beware of the basic attitude, which discretely undermines what LOGO could be. If you should see a teacher with this, ensure he/she ALSO reads MINDSTORMS.



# MINI MEMORY DEMONSTRATION

Everything but the kitchen sink in this one!

Requires Mini Mem and Speech Synth.

See notes at end.

```
1 CALL CLEAR
2 PRINT "BEFORE LOADING THIS
PROGRAM DID YOU REMEMBER TO
RESERVE MEMORY BY KEYING IN
:"
3 PRINT "CALL LOAD(-31888,57
,108)": "IF YOU DID,PRESS ENT
ER": "OTHERWISE BREAK AND DO
IT!"
4 CALL SOUND(-200,-7,0)
5 INPUT A$
100 CALL CLEAR

109 REM (C) 1985 S.SHAW!!
110 PRINT "IN TI BASIC YOU C
ANNOT HAVE-"
120 PRINT "SPRITES": "SPEECH"
:::"THE TI99/4A DOES NOT PE
RMIT-"
130 PRINT "SIMULTANEOUS SPEE
CH AND MUSIC": "CONTINUOU
S AUTOMATIC MUSIC"

139 REM A COUPLE OF SPRITES
:
140 CALL POKEV(768,62,15,138
,15,150,200,138,15,208)

149 REM MOVING SLOWLY:
150 CALL POKEV(1920,0,5,0,0,
0,5,0,0)

159 REM LET'S MOVE THEM!:
160 CALL LOAD(-31878,2)

169 REM AND GIVE THEM AN IN
TERESTING SHAPE:
170 A$="1030327FFB2"
180 B$="000032FF7070301"

189 REM SET SPEECH ADDRESS
190 @=-27648

199 REM LOAD SOUND TABLE:
200 CALL POKEV(14700,3,142,1
5,144,30)
210 CALL POKEV(14705,3,133,1
3,144,60)
220 CALL POKEV(14710,3,128,1
5,146,30)
230 CALL POKEV(14715,3,142,1
5,144,15)
240 CALL POKEV(14720,3,141,1
7,145,15)
250 CALL POKEV(14725,3,142,1
5,146,30)

260 CALL POKEV(14730,3,129,2
0,146,30)
270 CALL POKEV(14735,3,141,1
7,144,15)
280 CALL POKEV(14740,3,129,2
0,146,15)
290 CALL POKEV(14745,3,131,2
1,146,30)
300 CALL POKEV(14750,3,140,2
3,146,30)
310 CALL POKEV(14755,3,139,2
6,144,60)
320 CALL POKEV(14760,3,141,1
7,146,30)
330 CALL POKEV(14765,3,141,1
7,144,60)
340 CALL POKEV(14770,3,142,1
5,146,30)
350 CALL POKEV(14775,3,133,1
3,144,60)
360 CALL POKEV(14780,3,128,1
5,146,30)
370 CALL POKEV(14785,3,142,1
5,144,15)
380 CALL POKEV(14790,3,141,1
7,145,15)
390 CALL POKEV(14795,3,142,1
5,146,30)
400 CALL POKEV(14800,3,129,2
0,146,30)
410 CALL POKEV(14805,3,141,1
7,144,15)
420 CALL POKEV(14810,3,129,2
0,145,15)
430 CALL POKEV(14815,3,131,2
1,146,30)
440 CALL POKEV(14820,3,140,2
3,146,30)
450 CALL POKEV(14825,3,129,2
0,144,60)
460 CALL POKEV(14830,3,134,0
0,159,30)

469 REM AND TELL THE CONSOL
E "PLAY IT AGAIN TEX":
470 CALL POKEV(14835,0,57,10
8)

479 REM TELL IT WHERE TO ST
ART FROM:
480 CALL LOAD(-31796,57,108)

489 REM TELL IT TABLE IS IN
VDP RAM AREA:
490 CALL PEEK(-31747,A)
500 IF A/2<>INT(A/2)THEN 520
510 CALL LOAD(-31747,A+1)
```

```

511 REM LETS HAVE A DISPLAY
:
520 PRINT " J": " JJJ":
" J"::::

529 REM DEFINE CHR$(58) AS
A ZERO:
530 CALL PEEKV(1152,A,B,C,D,
E,F,G,H)
540 CALL POKEV(1232,A,B,C,D,
E,F,G,H)

549 REM TELL CONSOLE TO STA
RT PLAYING MUSIC:
550 CALL LOAD(-31794,1)

559 REM WHILE MUSIC IS PLAY
ING LETS CHANGE THE DISPLAY
A LITTLE:
560 FOR T=1152 TO 1231
570 CALL PEEKV(T,A,B,C,D,E,F
,G,H)
580 CALL CHAR(42,B$)

589 REM INSERT TIME DELAYS
FOR SPEECH:
590 IF T/7<>INT(T/7)THEN 610

```

```

599 REM MAKE IT SPEEK! (SPE
ECH SYNTH REQUIRED):
600 CALL LOAD(@,70,"",@,65,"
",@,72,"",@,70,"",@,64,"",@,
80)

609 REM REDEFINE DISPLAYED
CHARACTER:
610 CALL POKEV(1512,A,B,C,D,
E,F,G,H)
620 CALL CHAR(42,A$)
630 NEXT T
640 GOTO 560
650 END

660 REM TI BASIC REQUIRES M
INI MEMORY MODULE PLUS SPEEC
H SYNTH.
670 REM MAKE LINE 600 A REM
IF YOU DON'T HAVE THE SPEEC
H SYNTH.

```

The last time you saw this list of CALL POKEV's was in Issue 6 of TI\*MES. It has had added to it a few other things Mini Mem makes possible, and also shows a brand new discovery.

The last full listing would have shown (line numbers from THIS listing):

```

469 REM SWITCH SOUND OFF
470 CALL POKEV(14835,4,159,191,223,255,0)
.....
624 REM IS MUSIC STILL PLAYING?
625 CALL PEEK(-31796,A,B)
626 IF B<243 THEN 630 ELSE 700
.....
700 CALL LOAD(-31796,57,108)
710 CALL PEEK(-31747,A)
720 IF A/2<>INT(A/2) THEN 740
730 CALL LOAD(-31747,A+1)
740 CALL LOAD(-31794,1)
750 GOTO 560

```

Sorry Stan there wasn't room for LOGO this time... maybe next issue??? Mr Burt: use REPLACE STRING with the TI Writer Editor to transliterate! Take another look at lines 110 to 120 on page 47 of TI\*MES No7. EXCELLENT!!!! When writing, please remember that 'ole SAE for a reply.

WHAT A LOT OF CODE WE HAVE LOST HERE!!!

At the end of the article in issue 6, the suggestion was made that instead of the somewhat empiricle method of seeing which address was being scanned ( line 625 immediately above), you could instead have a look to see if the SOUND was still turned on... eg:

```

625 CALL PEEK(-31794,A)
626 IF A=0 THEN 700 ELSE 630

```

UNFORTUNATELY this doesn't seem to work with speech added.

The shorter alternative, shown in the main listing, is to tell the computer: "When you reach the end of the music, PLAY IT AGAIN!"

This instruction is found in line 470, where after a zero, we give it the next address to start playing from, in this case, back to the beginning.

more.....>



If you really want to be impressed, change line 640 to 640 STOP. Now, when READY appears, does the music stop? It can be stopped by ANY CALL SOUND or by the INPUT and ERROR tones.

To clear the screen of the remnants of the sprites, key in, in direct command mode, CALL LOAD(-31878,0)

You won't find THIS in the Editor/Assembler manual. It is hidden in the TE2 Protocol Manual (another rarity) and thanks to Russell for passing this info on.

MEMORY RESERVATION: If you have a disk controller attached, the controller will reserve memory (Call Files(3) ) and prevent the Basic program crashing into the sound table.

If you do not have a disk controller attached, before you load this program, you MUST protect the sound table by keying in in direct mode:

CALL LOAD(-31887,57,108) [ENTER]

then key in NEW [enter] then load this program.

### CALL SOUND:

Look at lines 4 and 5 of the preceding program.

They are relevant even if you don't have a mini memory.

Found in Pete Brook's TI LINES....

If you ran this program, did you like the sound of the input prompt at the start?

It appears that if you precede the INPUT statement with a negative duration CALL SOUND, the CALL SOUND definition will replace the usual INPUT tone. The minimum duration for proper working seems to be about -70. If you specify -4000, you get a very long input tone!

You can vary the time duration, the frequency (& use sounds!) and the volume of the INPUT command.

By setting Volume to 30, you can even have a SILENT INPUT.

*Stephen Shaw*

~~~~~  
HELLO THERE!

A gentle word about these fairly complex discoveries....

even if you do not have the mini memory, please read these notes!

We are trying to obtain a public domain program to allow you to use these several mini mem items with ExBas and 32k ram.

Also, there are some programming tips which are applicable to the bare console only... see for instance the use of a negative call sound to modify the INPUT tone in the above program!!

Learning more about your console may or may not be immediately pertinent, but the more you know the better you will program.

If you cannot take it all in in one go, don't worry, you can always come back to it.

REMEMBER: We all started from the same position, with NO knowledge of computing whatsoever. Some of us have owned our consoles longer, some of us have access to more rare information and some of us just plain have more time. The only restriction to learning is a belief that it is not possible: refer to MINDSTORMS for some superb examples!!

If you have ANY queries (except machine code and hardware!) drop me a line, with an SAE for a direct reply if required. Tell me what YOU want to see in RAMBLES. If an article makes you ask a question I am delighted: it shows someone has been reading it! Please allow ample time for any reply, especially in early April, when I expect to be working many hours of overtime!

~~~~~  
RAMBLES: I regret that due to lack of space in this issue of TI+MES, there is no room for the usual Rambles gossip, nor for the written articles on VDP Registers and Disk Sector Access. It is hoped that these can be found room in the next issue, but this will mean the next Forth article by me will be in SIX months time.

Apologies to all for any inconvenience. STEPHEN SHAW.

# MANCHESTER MUSINGS

## TANDY DMP-110 PRINTER AND CORCOMP RS232 INTERFACE

Since I last wrote, things have expanded a bit on the hardware front, with the acquisition of a printer. I saw the Tandy DMP-110 dot matrix printer at £199 (£100 off the list price) in Tandy's summer sale and succumbed to my love of bargains! The only problem was that it jammed up with a sticky label buried deep in the works almost as soon as I got it, resulting in a 2 week return to Tandy. But all's been fine since then - now I know never to wind back a sheet of labels. Once they're in the printer it's forward winding only. That advice probably applies to all printers when they're being used with labels. Investigation "under the bonnet" after the jam revealed that the mechanism's a Seikosha 550B.

The trouble is, you can't just buy a printer for the TI99, can you? It's got to be connected somehow. The choice is either a free-standing interface (e.g. Axiom parallel or Boxcar or Myarc serial - available from Arcade Hardware) or one which fits in the Peripheral Expansion Box. Since I'd already got the latter, I decided to buy an interface card to go inside it. The printer has both serial (RS232) and parallel (Centronics) interfaces, so the standard TI RS232 card looked ideal, as I could either use one of its two RS232 serial interfaces or its parallel one. I couldn't get the Texas card, but Parco were able to supply Corcomp's equivalent. Incidentally, I hear that Corcomp's future is a bit uncertain, but I've been assured by Parco that they'll fix faults, should the card develop any.

If I'd read Howard Greenberg's tale of woe in the last but one issue of TI\*MES, I doubt if I'd have bought a Tandy printer! However, the connection is really simple, provided you don't try and use the parallel interface like Howard did! The DMP-110 prints at a maximum speed (in compressed character mode) of 83 character a second. Its serial interface operates at the equivalent of either 60 or 120 characters a second. So by switching the printer's interface to 120 characters per second, and the Corcomp RS232 interface to the same speed (1200 baud), the printer can be driven flat out (in theory : this assumes that the TI99 can generate data that quickly. The connection is such that if the computer does outpace the printer, the printer will hold up the computer till it's ready to print again).

The 3-wire cable you need to connect the two serial interfaces is as follows:

|           |                  |               |           |
|-----------|------------------|---------------|-----------|
| TANDY     | 4-pin DIN plug   | 25-pin D-type | CORCOMP   |
| PRINTER   | Pin 2 ---        | DTR -- Pin 20 | RS232     |
| SERIAL    | Pin 3 - Ground - | Pin 7         | INTERFACE |
| INTERFACE | Pin 4 -- Data -- | Pin 3         |           |

(That's what I needed, but check your printer manual and TI interface manual to be sure!).

Apart from a range of fonts and print modes (bold, elongated, underline, superscript and subscript):

|                       |             |                   |                  |
|-----------------------|-------------|-------------------|------------------|
| Standard Normal       | <b>Bold</b> | <u>Underlined</u> | Elongated        |
| Standard Elite        | <b>Bold</b> | <u>Underlined</u> | Elongated        |
| Condensed             | <b>Bold</b> | <u>Underlined</u> | Elongated        |
| Correspondence Normal | <b>Bold</b> | <u>Underlined</u> | Elongated        |
| Correspondence Elite  | <b>Bold</b> | <u>Underlined</u> | Elongated        |
| Proportionally Spaced | <b>Bold</b> | <u>Underlined</u> | Elongated        |
| <i>Italic Cursive</i> | <b>Bold</b> | <u>Underlined</u> | <i>Elongated</i> |
| Microfont             | <b>Bold</b> | <u>Underlined</u> | Elongated        |

the nicest feature of the printer is its graphic modes. Its high resolution mode is really easy to use, and I've adapted the screen\dump routine in "99er Magazine" Volume 2 Number 1 (November 1982) to use in Mini-Memory from TI Console BASIC (written using the Editor/Assembler) to provide a "double size" dump of the screen by printing 4 dots for each dot on the screen. It gives screen dumps like this (full size):

```

                THE GRAPHICS SET.
    128 ■      129 ■      130 ■      131 ■
    132 ■      133 ■      134 ■      135 ■
    136 ■      137 ■      138 ■      139 ■
    140 ▲      141 ▲      142 ▲      143 ▼
    144 ■      145 ●      146 ▲      147 ▲
    148 ▼      149 ▼      150 □      151 ◆
    152 ▲      153 ➔      154 ➔      155 ➔
    156 ➔      157 ⤴      158 ⤴      159 ⤴

    ** DONE **
    >CALL LINK("DUMP")
    
```

If you've got this Tandy printer and want a copy, let me know. It takes about 8 minutes to dump a whole screen. (I've also written a routine in Extended BASIC which takes an hour to do the same thing!).

DISKS

The Post Office publishes a pamphlet entitled "Wrap Up Well" which includes the advice for "Magnetic tapes,etc" (which must include cassettes and floppy disks) that "where it is essential that no erasure occurs, it is recommended that a box large enough to allow a minimum of 100mm (4 in) of soft packing all round the item is used". Most people don't bother, and my experience to date is that this has been unnecessary for mail within the UK - but some precautions are necessary even here - as some recent experiences have shown.

I've had two examples in the last few months where the advice should have been taken and wasn't. I ordered some programs from the International Users Group (IUG) on floppy disk, and received two disks from the USA, which were quite unreadable on my disk drive - or anywhere else! I sent them back and, sure enough, the catalog sectors had been wiped out in the post on both of them - presumably through some X-ray equipment used to scan the mail, here or in the USA, for dangerous items. The replacement disks were wrapped in aluminium foil - but fared only slightly better. I could sometimes read one of the disk catalogs, but not any of the programs on the disk. But the BBC micro at work came to the rescue - it was able to read all the data on both disks - so I was able to produce duplicate disks that I could read on my TI99! I'm not really 100% sure why this should be the case - presumably the YE-Data disk drive on the BBC is a lot more tolerant and sensitive than my standard Texas drive. But the moral of this particular story would seem to be that anyone sending or receiving disks with the USA ought to make sure that the Post Office's advice is taken.

However, I wasn't prepared for what happened the other week. I ordered the TI Advanced Assembler Debugger (if you buy it you'll find out its real name!) from Paul Dicks of TIHOME following his note about it in the TI Home Computer



Users Club News (do we have Stephen Shaw to thank for its publication?!). It was posted to me in a large A4 envelope with a single piece of packing material around the disk - its paper sleeve! I wasn't in when the postman called, so he folded the envelope (and contents) in half to get it through my letterbox. I can report that Texas disk drives do not read creased disks - but, and I hardly dared to believe it - YE-Data drives do! - so the Beeb at work came to the rescue again - but really .....! Looking back through past issues of the late lamented TIDINGS, I see that Stephen Shaw once had the same trouble - perhaps his postman's now on my round!

DISK FILES

Talking of disks, I was writing a little program to process TI-Writer formatted output files (VARIABLE length records, DISPLAY type) for output on the Tandy printer. (Actually this was to have been a preliminary for a project suggested by a comment of Howard Greenberg's to print proportionally from TI-Writer's non-proportionally spaced output - of which there may be more in a future issue). I forgot to read the manual, as usual, and got in a bit of a mess. So, to save anyone the hassle I had, here's a short tutorial on disk files.

Assume that a disk file has 2 records, each containing 2 characters.

- Record 1 (length 2 characters) : "space linefeed"
- Record 2 (length 2 characters) : "pagefeed linefeed"

If you read the file with an INPUT #1:A\$ statement, you get the following:

|          |                                            |  |                       |
|----------|--------------------------------------------|--|-----------------------|
|          | Console BASIC:                             |  | but inExtended BASIC: |
| Record 1 | [LEN(A\$) is 1] A\$ is "linefeed"          |  | [LEN(A\$) is 0]       |
| Record 2 | [LEN(A\$) is 2] A\$ is "pagefeed linefeed" |  | [LEN(A\$) is 0]       |

Now the User's Reference Guide (page 69) and Extended BASIC Manual (page 102) tell you that when you read data with an INPUT statement, BASIC throws away any leading spaces. I can't find anywhere that says that BASIC gives you all the other Control characters but Extended BASIC gives you none of them!

By the way, I was doing the wrong thing anyway! If you want to read all the data in the file, then you need to use LINPUT #1:A\$, but this is only available in Extended BASIC.

So there we are : three ways of reading the same file - the one you choose depends on what you want to see!

MAGAZINES

I'm glad that the rumoured end of COMPUTE!'s TI99/4A coverage seems to be ill-founded. The October issue contains a pledge to continue support of our computer (and includes a fairly useful disassembler for Extended BASIC) and recent issues all contain TI programs. Two of my local newsagents stock this magazine, which provide games and educational programs with considerably more supporting explanation than the UK glossy games mags (they come with a "COMAG" sticker on them, which might be a clue to their UK wholesaler's identity).

The specialist 99 mags seem to have dried up. The last copy of Home Computer Magazine (ex-99er) which I received from my annual subscription through Paul Dicks of TIHOME was the Volume 4 Number 2 issue. (2 magazines since February 1984 hardly seems good value for a year's subscription of £32 to a "monthly" magazine!). However, I've found that both Parco Electrics of Honiton and ABC Computers of St. Austell have more recent copies (Vol. 4 Nos. 3 and 4) in

stock - so at least the mail seems to get from the USA as far as Devon!! Paul's had so much hassle with TI, his ordering intermediary I understand, that he's ceasing to handle subscriptions and refunding the remainder of current subscriptions for those who write to him - thanks Paul; sorry it had to end that way.

And the IUG seems to have gone into suspension - no "Enthusiast 99" received since Volume 2 Number 3 (May/June 1984) - and it's meant to be bi-monthly. So I gave them a call in the middle of February. I gathered that no more issues are planned at this time, and that all outstanding advertising was placed in a supplement to the Software Catalog which was "mailed to members in December". When I said I'd not yet received this, the lady I spoke to promised to check and, sure enough, a Catalog update arrived soon afterwards by air mail (postmarked 16th February 1985)! So ... if you're a member of IUG and have not received this - write to IUG and ask for your copy. The advertisements are interesting - including a Scott, Foresman educational software "close out" sale with 24 modules (many NOT PREVIOUSLY ISSUED) for sale at 4.95 dollars each. (I've ordered a few of these for my young nieces, so watch out for a progress report in the future). A firm called Dragonslayer advertises an Auto Spell-Check program for TI-Writer (U.S. spelling of course!) at 50 dollars; and Tenex advertise a Quality 99 Software utility QS-WRITER which allows loading the TI-Writer disk from Extended BASIC, Editor/Assembler or Mini-Memory modules - sounds like another in the "piracy aids" catalogue!

Galaxy's "TI-User" magazine seems to be getting thinner - and the latest issue (Number 6 November 1984) was photocopied, unlike the previous printed issues. Still, the clearance prices on Galaxy's remaining TI modules are attractive!

TI Users Club News No. 3 appeared eventually. The most annoying feature of this magazine is that every useful piece of information contributed to it seems to be categorised as "too long" or "too technical" and available only on request. Perhaps contributors who've been placed in these categories should write for "TI\*MES" instead - in fact, I see that at least one already does!

#### TAPES

After my disk troubles, I suppose I should have expected to run into tape problems as well! Anyway, bouquets to Games Machine Ltd who replaced a tape without quibbling when I returned it to them as unreadable. But no flowers to the TI Club of Acton (run by J.A. Murphy). I saw an advert for a tape magazine for the TI in Home Computing Weekly in July and so I sent for the August issue on 31st July 1984. I received a tape on 25th September 1984 indicating that it had been posted on 1st August, but returned by the Post Office on 21st September! As I couldn't read it, I sent it straight back - and that was the last I heard. But I see from a letter in HCW that someone received no tape at all and another had similar difficulties to myself. How many other members said goodbye to £1.60? Did anyone get a readable tape? and did anyone take up the offer of the next issue (October 1984) at a further £1.60?

#### TI-FORTH

I picked up a copy of TI-FORTH from Parco when I dropped in to see their "TI Aladdin's Cave" in Honiton on the way to Cornwall in November for a belated summer holiday. The staff are really keen, and dedicated to the continuing support of the TI99 - long may they continue!

I've not had a chance to learn too much of the language yet - but already it seems to have its uses in providing facilities to read disks without the constraints of the Texas disk filing system. The real beauty of the language is that its powerful yet simple - reaching parts of the TI99/4A that other

languages don't reach - or reach only with difficulty.

The problem is, for most users, that you need an Editor/Assembler module, 32K RAM expansion, a disk controller and at least one disk drive to run it!

How about publishing a page of useful FORTH routines in each issue?

Incidentally, "Starting FORTH" by Leo Brodie is really useful for the newcomer to FORTH, particularly as the TI FORTH manual refers to it and gives a page by page indication of the differences between the two implementations described - with definitions of routines to provide most of the "missing commands" from TI-FORTH.

#### MORE PURCHASES

The first part of this letter was written just too late for the last issue, so here's an update on recent activity.

Unable to resist a bargain, I was pleased to see Multiplan reduced in price in Parco's Winter Sale. Now I can go back and work through the extensive series of tutorials that was published in "99'er". Peter Brooks told me he's just bought Multiplan as well - it would be interesting if any users who've had Multiplan for some time could write in and tell us the sort of things they use this "spreadsheet" package for: I've bought it partly because I think it'll help me with some financial planning, and partly just to get experience of this widely used "management tool" (it's great attraction is that Multiplan is available on most small computers with exactly the same user interface).

Promptly, on cue, exactly as my console warranty expired, the console began to experience hang-ups after removing modules! I'm sure this is due to the contacts getting dirty in the cartridge connector. It only used to happen when the console was warm. I'm a bit scared of taking things to pieces, though I'm sure I could have simply taken Stephen Shaw's advice and removed the offending dirty "cleaning" foam. However, I adopted the alternative approach of buying a Widget (alias Navarone Cartridge Expander) from Howard Greenberg (alias Arcade Hardware). Now I can plug in the three cartridges I use most: Disk Manager II, TI-Writer and Extended BASIC, simultaneously and switch between them with this wonder gadget - no console should be without one! It does keep the cartridges cooler, too, though the console itself seems to get a bit hotter!

#### FORTH REVISITED

I set myself a little project of writing a disassembler in FORTH firstly to learn the language but with the main objective of decoding how FORTH reads disks. It's got "sector read and write" words which let you read any part of a disk as you want and not just in the way that the TI file system lets you. My next intention was to decode the disk controller software to see whether there are any useful undocumented features there.

Three events somewhat diminished my enthusiasm: (1) I talked to Richard Blenden - and he's done it all already (working on decoding the disk controller from Pascal). Much of the material I thought I might discover, he's already uncovered (watch out for Peter Brooks' report on this - he's doing a Boswell to Richard's Johnson!). (2) I bought "Thinking FORTH" by Leo Brodie, which made me look at my disassembler program (translated from and published in COMPUTE! in Extended BASIC) with disgust - so it's going to have to be re-written before I can think of distributing it! (3) The source of TI FORTH is in the public domain! I got my copy from TIHOME (now run by Peter Brooks) and listed it all (it occupies nearly 2 disks)! Apart from listing the assembler calls to read and write disk sectors and format disks, it includes a very useful, commented source of an extract from the Editor/Assembler utilities, which FORTH uses but in a different part of memory from the E/A.



For someone who wants to get into advanced assembler programming and has the E/A system, I recommend getting hold of the source of FORTH. After all, we improve our English writing ability by reading, and the same principle applies to computer languages as well.

If you have FORTH and you want to see the code in your disk controller, try this:

```

HEX 0 VARIABLE MYBUF 2000 ALLOT ( set up buffer )
-CRU ( call in CRU words )
1100 1 SRL DUP SBO ( set disk controller CRU bit )
4000 MYBUF CMOVE ( copy out disk controller software )
SBZ ( clear disk controller CRU bit )

```

Now you can DUMP or disassemble the code at leisure!

The byte at MYBUF+1 will give you the hardware version number of the disk controller - mine's 02, what's everyone else's?

I wrote to the address that Stanley Dixon gave in the last issue for the FORTH INTEREST GROUP(UK) but had my letter returned "Gone Away". Has anyone got their new address?

### SURPRISE, SURPRISE!

Now I've seen it all: a user's club that doesn't want members! Just when I thought I'd finished this piece, what should drop through the letter box this morning but issue No. 4 of TI Home Computer Users Club News - with the alarming information that "existing loyal Members" will receive copies of the magazine "albeit in a revised format"(?) for as long as their subscriptions last. Now, as a "Founder Member" my subscription's just run out after 1 year! Since mailing lists were closed in December 1984 and as former TIHOME members are promised membership till the end of 1985, it follows that something's going to be produced this year! But you can only get a copy if you (a) joined before it started (TIHOME members) or (b) joined late!! So why not take my money for another year? I suspect it might be because it would alter the profit/loss (and therefore tax) position of the "Club" (company?) before it's wound up. But they're still willing to part me from my money for small ads (read by a decreasing number of readers!), or to take an article - which I'll never be able to read! It all makes you wonder what the "revised format" will be like - a single duplicated sheet, perhaps? The most annoying thing was that they just wished Members with expired subscriptions "all the best for the future" - no mention of an alternative source of support like TI\*MES! Ah well, what should I have expected for £5? (I've written along the above lines to TIHCUC - I'll keep you posted of any reply).

Well, I suppose that just about wraps it up for now. There must be something about the atmosphere in Manchester which makes its inhabitants enthuse about the TI99 at such great length! Which reminds me - it was great to meet so many people at the Users Convention at the Ritz in November - everyone seemed to agree it was really useful - so I hope we can have another one soon.

Best wishes,

*John Rice*

John Rice

7 Lincoln Road, SWINTON, Manchester M27 3WR

## POKEING WITH MINIMEM

I bought my Minimem some two months after I bought the TI.99/4A, and at that time delivery of Ex-Bas modules was subject to long delays. At that stage in my knowledge of the TI.99/4A the Minimem handbook might just as well have been written in Sanskrit for all the good it was to me.

After a few false starts and some debugging I did eventually manage to key in the sample assembly program to simulate Ex-Bas' DISPLAY AT command, and got it to work OK. When I did eventually get my Ex-Bas I neglected the Minimem for several months. My interest in the Minimem was revived by Stephen Shaws articles in TI\*MES and also S. Michel's machine code programs. By this time with my wider knowledge of the TI.99/4A the handbook began to make more sense, and I began to experiment by PEEKING into the VDP RAM; easily done with the Minimem, but not so easy with Ex-Bas. I also tried using POKEV to alter a program while it was running, specifically to alter the DIM values in an array declaration. My experiments were successful but the results were of limited practical use, and further research seems to be indicated.

However, it suddenly dawned on me that the CALL POKEV command could be used as a DISPLAY AT command, thus doing away with the necessity to have the assembly program in the Minimem RAM. For instance, try the following short program:-

```
100 CALL CLEAR
110 CALL POKEV(301,168,165,172,172,175)
120 GOTO 120
```

Run it, and HELLO suddenly appears in Row 10 and (Graphics) Column 14

How does it work? Well the first number in a CALL POKEV is a memory location in VDP RAM, and subsequent numbers are the values to be poked into that and successive memory locations. In the TI.99/4A the screen is memory mapped which means that every character location on the screen has one unique memory location in VDP RAM assigned to it. There are 768 character locations on the screen (32\*24) and the corresponding memory locations are numbered from 0 to 767. Thus Row 1, Column 1 on the screen is controlled by memory location 0. In the example above memory location 301 determines what character will appear on the screen at row 10 and column 14. The number of the memory location can be calculated by using the formula  $M = (Row-1)*32 + Column - 1$ .

What do the remaining numbers mean? They are not ASCII codes are they? No, but they are related to them by a simple arithmetical relationship. For some reason when using POKEV to put values on the screen there is a 'screen bias' of 96. This means that to put a character on the screen you poke not its ASCII code but its ASCII code plus 96. So if you subtract 96 from the numbers in the POKEV command (except for the first number which is the starting memory location number) you get 72,69,76,76,79. Effectively we have put the code for H in location 301, for E in location 302 and so on, to display the word HELLO where we wanted it.

The POKEV command works just as fast as the DISPLAY AT command and a lot faster than the usual subroutine using CALL HCHAR with a FOR-NEXT loop. As a trade off it is slower to program especially if you have a lot of text to display, since you have to look up the ASCII number first (if you can't remember them all) and then add 96. Also you have to calculate the memory location although you could I suppose use the row and column numbers in the conversion formula as the first number in the POKEV command. If you have a lot of text it might slow up the display a little however. If you mean to use this a lot I suggest you add an extra column to your table of ASCII codes to give the ASCII code plus 96.

Using the POKE command you are limited to string expressions and cannot use string variables, but the same limitation also applies to the assembly language program. A further limitation is that you cannot fill a screen row with one POKEV statement, you can only get 24 characters at most into one program line. A program line can only be 4 rows long and can only have 112 characters therefore. Whereas in DISPLAY AT you enter the character itself, in POKEV you enter a 3 digit number, and a comma as a separator which occupy 4 character spaces in a program line. You can at most therefore get only 28 characters in a program line, but allowing for the CALL POKEV tokens and the brackets this reduces to 24. You should note also that while DISPLAY AT uses the Text screen of 28 columns POKEV uses the Graphics screen of 32 columns.

CALL POKEV can also be used very effectively in producing complicated graphic displays, using many different characters, as you can get up to 24 different chars in one statement which would otherwise use 24 CALL HCHAR statements. Where a uniform colour and character is needed over a large part of the screen, CALL HCHAR is clearly better with its ability to handle repetitions of one character in one short statement. Using CALL POKEV in these circumstances you would either have to repeat the one code up to 24 times, or use a FOR-NEXT loop. A judicious mix of CALL POKEV and CALL HCHAR can ease the programming of graphics and speed the execution.

The obvious next step is to find a similar way to simulate ACCEPT AT. If there is I have been unable to find it, although I may be again overlooking the blindingly obvious. It is easy enough to work out a hard core routine in TIBASIC using a loop with CALL HCHAR. Using MINIMEM you can use CALL POKEV instead of CALL HCHAR but this is of no advantage. The minimum program needed is as follows:-

```
1000 I=1
1010 A=""
1020 CALL KEY(O,K,S)
1030 IF S=0 THEN 1020
1040 IF K=13 THEN 1090
1050 CALL HCHAR(R, C+I,K) (or CALL POKEV(RC+I,K+96) where RC=(R-1)*32+C-1)
1060 A$=A$&CHR$(K)
1070 I=I+1
1080 GOTO 1020
1090 RETURN
```

This works alright, but don't make any mistakes. Try backtracking to correct an error. You don't know whether anything is happening because there's no cursor, but press another key and you will see that the print position has not backtracked.

"The moving finger writes and having writ, moves on,  
Not all your piety nor wit can lure it back to cancel half a line"

(With apologies to Edward Fitzgerald and Omar Khayam)

The routine as it stands has serious deficiencies. No cursor and no editing facilities. In special cases this may not matter much. I have a program to set problems in arithmetic to my grandchildren, and I use this routine for the answers. These answers are never more than 3 digits and are displayed immediately following the display of ANSWER = so a cursor is hardly necessary. To correct keying errors (not arithmetic errors) the children are instructed to press E and start again. I use a 'start again' routine to cancel the display and start afresh from the beginning, by adding the following line:-

```
1035 IF K=69 THEN 1000
```

If the answers include alpha characters then I suggest using the REDO key instead of E. (REDO is FNC & 8). Line 1035 then tests whether K=6. This is a simple and effective method where only a few characters will be entered at one go, but it is not really suitable where long lines of text have to be input.

If you are going to have editing facilities you do need a cursor. You can't access the cursor routine from basic other than by using INPUT, so it is necessary to devise some substitute. A difficulty at once arises if you just use character number 30. It won't blink, but worse it erases the character it passes over, so that you have to add further programming to reinstate it. The whole routine is beginning to get unwieldy and over complex and we still haven't added the editing facilities.

Then an idea occurred to me. MINIMEM can produce sprites as Stephen Shaw demonstrates in TI\*MES number 7. Sprites pass over other graphic characters without erasing them, and what is more the parts of the sprite character not blocked in are transparent. So after some experimenting I produced the following extra lines for the routine:-

```
1015 E=-1
1025 CALL POKEV(768,R*8-7,(C+I)*8-7,127+E,15,208)
1026 E=-E.
```

This works fine, even to the blinking.



To code a sprite with Minimem you need 6 numbers in the CALL POKEV statement. The first number is 768 which is the first memory location in the Sprite Attribute Table. The next two numbers are the pixel row and pixel column numbers (sometimes referred to as dot row and dot column). As there are 8 rows of 8 pixels in each character, there are 192 pixel rows and 256 pixel columns on the screen. Those of you with the Smart Programmers Guide to Sprites will recognise the formulae to convert graphic row and column to pixel row and column used in the CALL POKEV statement. The next number, 127+E is the character number (including the bias of 96). Each time round the loop this number changes from 126 to 128 and then back to 126. Subtract the 96 and these numbers are 30 and 32, cursor and space. The blinking is created by the rapid alternation between these two characters. The next number 15 is the colour number, in this case it is the number for white, as the Color Table in Minimem uses the numbers 0 to 15 instead of Basics 1 to 16. The last number is something of a mystery to me. The sprites work alright without it, but there tends to be a random selection of garbage displayed on the screen as well as the sprites. Include the 208 as the last number and the garbage vanishes.

I have never been keen on TI's blinking cursor, and prefer an underline myself. So I tried an underline as an alternative, using character number 95. Add 96 and you get 191. So line 1025 became CALL POKEV(768,R\*8-7,(C+I)\*8-7,191,15,208). Since the underline as placed here does not obscure any part of the character it indicates there is no real need for any blinking so we can dispense with lines 1015 and 1026.

Next editing. After some experimenting I decided against full editing with INS and DEL. The addition of further lines within the loop disclosed another problem. The keyboard response was becoming more sluggish with each extra line. As I am a fast typist myself this caused errors of omission, as the keyboard failed to respond before I passed on to the next key. It became a matter of some importance therefore to keep the number of lines within the loop to a minimum. The only lines I added therefore were those necessary to move the cursor backwards or forwards. If it was merely a wrong character to be corrected this was OK but a character omitted had to be corrected by overwriting from that point. Not a very onerous demand in practice as the error was usually spotted almost straightaway.

Now a further problem arose. Suppose you wished to enter HELLO but hit the wrong key and got HWLLO. OK, backtrack the cursor to the W and type E, then forward with the cursor and ENTER. The display on the screen looks OK but if at a different point in your program you have occasion to reproduce the string as stored by the computer you get HWLLOE. This won't do. Line 1050 has merely added the E to the end of the string instead of it replacing W. I could think of several ways of sorting this problem out by slicing the string up with SEG% but they all add further complex lines to the loop and slow down the keyboard response even further. My solution was to remove the building up of the string from the main loop, thus incidentally helping speed up the keyboard response, and adding the following loop to the end of the routine.

```
1090 FOR J=1 TO I-1
1100 CALL GCHAR(R,C+J,L) (or CALL POKEV etc).
1110 AS=AS&&CHR$(L)
1120 NEXT J
1130 RETURN
```

This loop reads the string only after correction.

Finally I tidied up the routine a bit and removed all unnecessary calculations from inside the main loop to speed it up as much as possible, and to help in ensuring that a key has registered added a CALL SOUND line to give a very short BEEP. The final result is given below.

I am not altogether happy with the performance of this routine. Response is slow but just about acceptable. From my little knowledge of Assembly Language it ought to be possible to write a routine to grab hold of the cursor and editing routines, so that you could use them with a CALL LINK statement. But if you are going to have an assembly language routine sitting in your Minimem RAM you might just as well have a routine to cover the whole ACCEPT AT statement rather than just the cursor and editing. Can any assembly enthusiasts oblige?

7 Harbury Close,  
Matchborough West,  
Redditch,  
Wores.  
B98 OEF.

MAIN PROGRAM

```
200 R=10 (or whatever row you like)
210 C=6 (or whatever column you like)
220 GOSUB 1000
```

ACCEPT AT SUBROUTINE

```
1000 I=1
1010 A$=""
1020 RC=(R-1)*32+C-1
1030 X=R*8-7
1040 Y=C*8-7
1050 CALL KEY(O,K,S)
1060 IF S=0 THEN 1050
1070 CALL POKEV(768,X,Y+I*8,191,15,208)
1080 IF K=13 THEN 1160
1090 IF K=8 THEN 1140
1100 IF K=9 THEN 1120
1110 CALL POKEV(RC+I,K+96)
1120 I=I+1
1130 GOTO 1050
1140 I=I-1
1150 GOTO 1050
1160 FOR J=1 TO I-1
1170 CALL GCHAR(R,C+J,L)
1180 A$ = A$ & CHR$(L)
1190 NEXT J
1200 RETURN
```

} These lines are to remove unnecessary  
calculation from the loop

*John Roe*  
John Roe.

~~~~~  
TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES
TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES
TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES
~~~~~

CHARACTER IDENTIFICATION.  
BY J.HAWKINS.

Very often when I have to de-bug a programme which I have typed in, I need to identify the graphics to trace exactly what is what. You can of course get a bit of squared paper and plot them out. But what does "010F3FFCFFF3FEFO" really look like on the screen? Therefore I devised the following little programme to help me. I save the dodgy programme onto cassette and load this into the computer. Then I type in the HEX character codes of each CALL CHAR and roughly sketch them on to the list. Elementary dear Watson, but it does help and is quicker than pen and paper plotting.

The I= 1 to 30 business prints six consecutive characters you input across the screen. When the question mark appears just bung in the "01010303F880FFF3" and press ENTER.

It should not be too difficult to reverse the process enlarged 8 times so that you could draw your character with black squares and spaces on a 8x8 grid-then press enter to get the HEX string. NOTE:A CHAR CONSTRUCTOR program is available in the Group's software library - U16 that will do this job for you.

```
100 CALL CLEAR
110 PRINT "HEX CHARACTER IDENTIFIER": : : :
120 PRINT "TYPE IN THE CHARACTER STRING WHEN THE QUESTION MARK APPEARS": : : :
130 INPUT "PRESS ENTER":NUL$
140 CALL CLEAR
150 FOR I=5 TO 30 STEP 5
160 CALL HCHAR(10,I,33)
170 CALL CHAR(33,A$)
180 INPUT A$
190 NEXT I
200 CALL CLEAR
210 GOTO 150
```

# BEGINNERS FEEDBACK

By J.G.Hawkins

I wonder how many hopeful programmers like myself have been caught out and perplexed when trying to use a PRINT AT subroutine with graphics.

If you have the time it is quite interesting to try the following programme; the object being to plot a small graphic and add annotations which is where things can go wrong.

As usual the Texas manual does give the technical data but very real down-to-earth instruction to the novice.

Lines 100 to 300 and 600 to 640 Print the words.

Lines 320 to 430 define the characters 84,73,65,53,57,92,70,79,72,78,76.

These character identification numbers have been carefully chosen to show the effect of bad number choice.

Lines 440 to 460 define the colours.

Lines 470 to 580 place the characters in position on the screen.

Type it in and run it.

First the words appear printing out perfectly but when the little graphic is printed the words turn into elements of the graphic and become completely obliterated.

Do you understand why?

Three graphic identifier strings are identical and this is deliberate.

The reason for the pollution of the printing is because I have used numbers to define the graphics which are the same as the ASCII code numbers of the letters in the printing, which is a thing to be avoided.

Now to put things right we have to chose numbers outside the letter ASCII code, ones which are above 127, which are the colour sets 13,14,15 and 16.

So EDIT as follows:-

Lines 320 and 470 change 69 to 128.

Lines 330 and 480 change 84 to 129

Lines 340 and 490 change 73 to 130.

Lines 350 and 500 change 65 to 131.

Lines 360 and 510 change 53 to 132.

Lines 370 and 520 change 57 to 133.

Lines 380 and 530 change 92 to 134.

Lines 390 and 540 change 70 to 135.

Lines 400 and 550 change 79 to 136.

Lines 410 and 560 change 72 to 137.

Lines 420 and 570 change 78 to 138.

Lines 430 and 580 change 76 to 139.

Line 440 change to FOR C=13 TO 14 (because the new numbers are in these sets.)

Now the story is different. The little graphic appears and the printing remains unadulterated.

```
100 CALL CLEAR
110 Y=10
120 X=6
130 C$="THE GRAPHIC ABOVE"
140 GOSUB 600
150 Y=12
160 X=6
170 C$="ONLY SERVES TO ILLUS
TRATE"
180 GOSUB 600
190 Y=14
200 X=6
210 C$="THE EFFECTS OF GRAPH
IC"
220 GOSUB 600
230 Y=16
240 X=6
250 C$="SET NUMBERS"
260 GOSUB 600
270 Y=18
280 X=6
290 C$="ON TEXAS TI99/4A"
300 GOSUB 600
310 REM SET OUT GRAPHIC
320 CALL CHAR(69,"E7FFFFC3C3
FFFFFF")
330 CALL CHAR(84,"0000F0F0C
FCFFFF")
340 CALL CHAR(73,"0000F0F03
3FFFFFF")
350 CALL CHAR(65,"E7FFFFC3C3
FFFFFF")
360 CALL CHAR(53,"FFC3C3C3C3
C3FFFF")
370 CALL CHAR(57,"FFF0F0F0FF
FFFFFF")
380 CALL CHAR(92,"FF0F0F0FFF
FFFFFF")
390 CALL CHAR(70,"FFC3C3C3C3
C3FFFF")
400 CALL CHAR(79,"FFC3C3C3C3
C3FFFF")
410 CALL CHAR(72,"F0F0F0F0F0
F0F0F0")
420 CALL CHAR(78,"0F0F0F0F0F
0F0F0F")
430 CALL CHAR(76,"FFC3C3C3C3
C3FFFF")
440 FOR C=3 TO 8
450 CALL COLOR(C,9,4)
460 NEXT C
470 CALL HCHAR(4,15,69)
480 CALL HCHAR(4,16,84)
490 CALL HCHAR(4,17,73)
500 CALL HCHAR(4,18,65)
510 CALL HCHAR(5,15,53)
520 CALL HCHAR(5,16,57)
530 CALL HCHAR(5,17,92)
540 CALL HCHAR(5,18,70)
550 CALL HCHAR(6,15,79)
560 CALL HCHAR(6,16,72)
570 CALL HCHAR(6,17,78)
580 CALL HCHAR(6,18,76)
590 GOTO 310
600 REM PRINT AT SUB
610 FOR B=1 TO LEN(C$)
620 CALL HCHAR(Y,X+B,ASC(SEG
$(C$,B,1)))
630 NEXT B
640 RETURN
```



## MORE TI REVIEWS

PENTATHLON by INTRIGUE SOFTWARE.

Extended basic - £5.95.

This game simulates 5 Track and field events for your TI. Once loaded, you input your name, choose the colour of your 'gear' and select your event. There are 5 events which are 100m, 1500m, discus, long jump and high jump. I found the running events rather slow. Perhaps I'm wrong but the key response is so bad that I suspect a fall through here and there.

However the field events surpassed my expectations and I found the long jump particularly good. All in all a good game and well worth the £5.95. (\*\*\*\*)SW

TASK FORCE by PROSOFT.

Extended basic -£5.95.

The object is to destroy red and black planes that fly overhead. The instructions are nicely presented and my hopes were high that it would be a good game. I was disappointed. Your "ship" is crudely drawn in block graphics as are the planes. The only bit I enjoyed were the explosions. Colourful as they were they did not save the game. All in all a lousy game which in no way justifies the price. (\*)SW

RAGING RIVER plus SUPER JACKPOT.

STAINLESS SOFTWARE

Basic - £6.95

I have seen games like Raging River on other computers and so was a little sceptical about a TI basic version. However I was not disappointed. Although simple in concept it is extremely frustrating to play. The object is to steer your boat along a winding river and to avoid the crocodiles and other obstacles. Apparently the record is 82 miles. Well whoever set that is pretty good, or else he wrote it!

Super Jackpot is quite a realistic copy of the arcade scrooges, with nudge, hold and gamble. I found it held my interest a bit longer than some games. Congratulations to the author. If you like arcade machines you'll like this. (\*\*\*\*)SW

CRAZY CLIFF by STAINLESS SOFTWARE.

Extended basic - £6

Cliff has to climb to the top of a skyscraper avoiding flowerpots, opening windows, aircraft etc. Increasing number of hazards and height on advanced screens and suprise guest appearances at the top of some buildings. Uses joystick. The key to success is to keep moving. An addictive game and good value for money. (\*\*\*\*)PAS.

ATLANTIS by INTRIGUE.

Basic - £6.95.

This game loads in two parts. The second part has a loading screen (something to look at, a good job, the program takes 10 minutes to load.)The object is to steer your diver round the caves in search of the lost city of Atlantis. Control is either keys or joysticks. On your way you avoid various nasties in the shape of sharks and giant squid. You also have to pick up pearls and diamonds to increase your score. I found this really enjoyable and strongly recommend it. (\*\*\*\*).SW.

Our thanks to Simon Waley for his program reviews.

# D. I. Y. HARDWARE

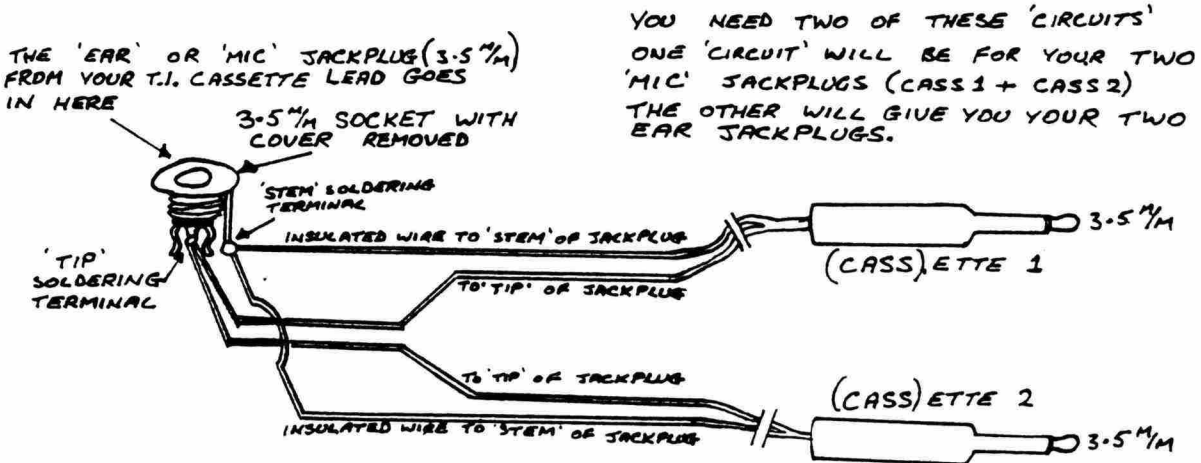
We all know that the T.I. will save AND old to C.S.1 but will only save to C.S.2, and even then only if you have a dual cassette lead. Data saved to C.S.2. cannot be checked except by inserting tape into C.S.1. and fiddling the instructions.

There is another way. To do this you need a standard single cassette lead (or the C.S.1. side of your dual lead). You also need several jackplugs, jack sockets, flexible wire, and .. a single pole double throw switch (S.P.D.T.) commonly called a two way switch.

All you have to do is to 'branch' the single EAR, MIC, and REM jackplug outlets into dual jackplug outlets. The MIC and EAR 'branches' are the same just follow the diagram below for both. The REM is more complicated and will be dealt with later.

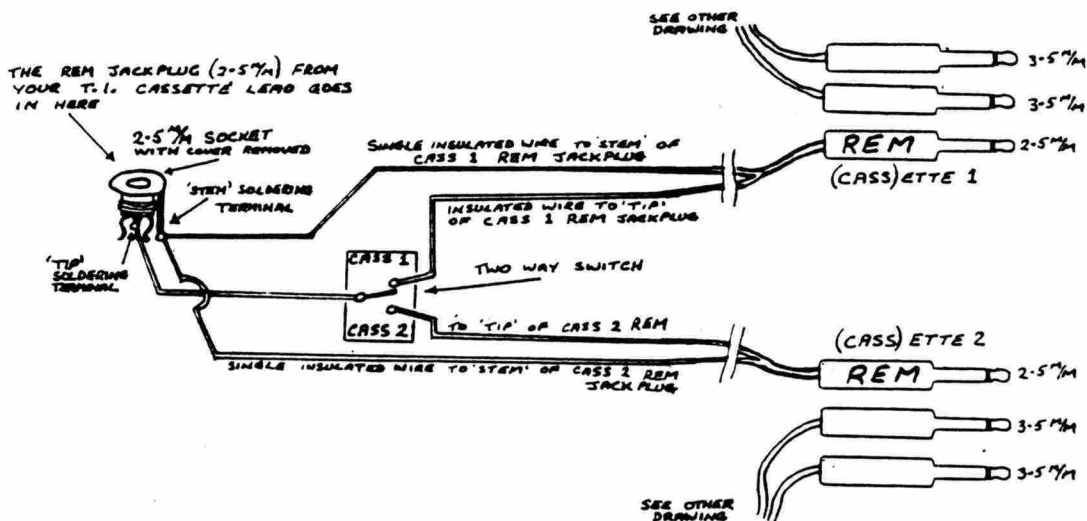
Two points to note here

- (1) If you make this modification you will only need to use the (C.S.1.) command to your computer ignoring the (C.S.2.) command.
- (2) If you haven't even got a T.I. cassette lead, you can combine the following instructions with my notes on how to make a single cassette lead, to make the lead of your choice. (( I.E. you could leave out the intermediate jackplugs/sockets.))

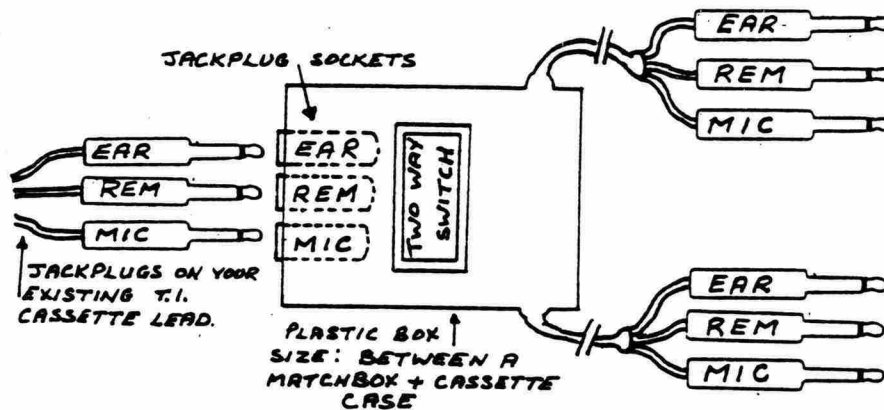


The above diagram is for 'MIC' and 'EAR' circuits, so you need two off

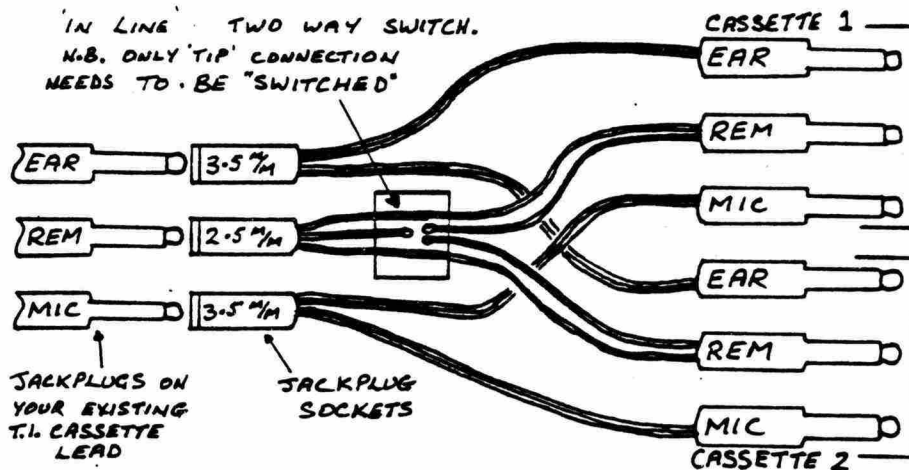
The circuit for the REMote branch is similar but includes the two way switch. see the diagram below.



The drawings supplied here are diagrammatical only, the final choice in design and layout is up to you. You may want long or short leads, you may want to base it all in a small (plastic) box. see diagram below,



Or you may want to use an "in line" switch, on the appropriate part of the REMote wiring. see example below.



I hope this will be of some help.

DEREK FORD.

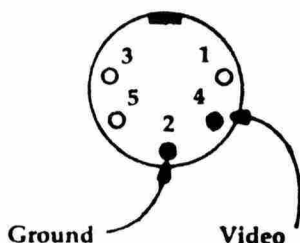
### TI Monitor Connection

Can you help me? Our school has recently purchased 8 Texas Instruments TI-99/4A microcomputers. Among the 8 donated black and white television sets was a Hitachi monitor used with an Apple computer.

Can I hook up the Hitachi monitor to the TI? If so, how?

George S. Ruff

#### TI-99/4A Video Connector



Although the TI video output is a color signal, an acceptable black and white monitor picture can be obtained by taking the two signals shown in the figure below to the monitor input. The figure shows the video connector as you would see it facing the back panel of the TI-99/4A, or the back side of an easily-obtainable five-pin DIN plug which plugs into the connector. Use shielded cable with the shielding braid connected to the GROUND pin. For the other end of the cable use whatever type of plug mates with your monitor's input jack.



## LIBRARY REVIEW

REVIEW: M11 "BANANAS". A Musical Program in EXTENDED Basic. This Extended Basic prog plays a delightful rendition of that old favourite "YES WE HAVE NO BANANAS." Mums'n Dads will probably remember this tune which was written more years ago than I can recall! This is a definite program to show friends who have other makes of computer, as it makes full use of the TI. sound and graphics. The musical arrangement is what I would term "whacky" It plays the tune, prints the words and displays quite good pictures of the articles mentioned in the lyrics. I found it all quite funny and well recommended.

REVIEW: D3 "PANO" A Kiddies story requires TE.II and SPEECH SYNTHESIZER. This is a "Storybook" type prog for very young children. It is a tale of PANO THE ENGINE from a book by Sharon Holaves, and is told by "ROCKY THE ROBOT" whose figure appears on the right hand side of the screen and his mouth opens as he starts to speak. Briefly, the story is printed on the screen and spoken a couple of lines at a time, while underneath the printing is displayed a picture of the train. There are one or two slight spelling errors, but all in all it should interest the very young and may even help them to improve their reading.

REVIEW: U1 AUTO LOAD and disk directory utility. EXTENDED BASIC. This is a first class utility prog for Disk users. There have been several programs published that auto load extended basic progs, (I even wrote one) but this has got to be the best! It can only be described as PROFESSIONAL. If this is stored on all your Disks with the name "LOAD", every time EX.BASIC is selected it will automatically load and run. Its options include listing all progs etc on any Disc drive, it will load and run them and will even catalogue them to a printer like the DISK MANAGER module does. Another extremely useful function is that you can even delete progs and files from your Disk! This can save time and wear and tear on the cartridge port. Two very minor points, if you ask it to catalogue a Disk that has not been formatted it will stop with an error, also very large progs which require CALL FILES(1) will not load, however these are very minor inconveniences. I will certainly use this on all my Discs.

REVIEW: B1 MODEM UTILITY for TERMINAL EMULATOR PROG. Full system required and EDITOR ASSEMBLER or MINIMEM. This extremely useful Utility is available on Disk only, needs the full system and a Modem. It will in fact replace the TERMINAL EMULATOR II module (except for speech). It is designed for Bulletin Board use and is very versatile. Just about any current BAUD rate can be selected, in fact the only real limitation as far as I can see is the actual design of the TI RS232 card and of course your modem. Files can of course be transferred to other computers. One benefit over TEII is that you can control screen and character colours, I find the colours on TEII a little bit hard to read in certain circumstances. I believe, (I have not had time to try it), that this prog will also spool off text to a printer at the same time as receiving, previously we had to interrupt the data flow and then save a screenful to Disk or Printer. A very handy prog for the Communications enthusiast.

REVIEW: U32 A Printer Utility for TI WRITER type Files. EXTENDED BASIC This Utility prog, written in EX.BASIC allows the user to print TI WRITER type files on Disk to a printer. It displays quite a large Menu to allow you to send the correct Control Codes to your printer ie. Emphasised, or Condensed print modes etc etc. I found it a great time saver as usually, I am not very organised, I have to look up any special printer codes in the handbook if I wish to use any different print styles to enhance my literary wotsername 'er prose cough cough, (blushes deeply). A very useful program. One little warning, this utility, like several in the group Library is of American origin, and on reading the REMS the Author advises any users to make back up copies of files first. All I can say is that I have found no problems at all, and it seems Bug Free to me.

Well thats all for now, I hope some of this short witty will give you an idea what programs are yours for the asking from the TI\*MES library.

Graham Hilton.

(TI99/4a Exchange group contact)

8 Sandwich Close

Saint Ives

Cambridge

PE17 6DQ

\*\*\*\*\*  
\*\*\*\*\* YOUR LETTERS \*\*\*\*\*  
\*\*\*\*\*

PARCO ELECTRICS of Honiton write: Since you took the time, trouble and space to answer questions that were posed in the problem section of our magazine before we had a chance (S.S) short of subject matter for once?), we thought you would be pleased to include the following statement from Issue 3 of '99/4a'. "PARCO ELECTRICS are now the ONLY TI99/4a APPOINTED DEALER, and claims by anyone else about supposed unavailability of products must be treated as purely unofficial. It has been intimated for instance, that PARSEC, among others, cannot be obtained. Well we want to put it on record that we not only carry adequate stocks of this (and most other) items, but could order more tomorrow.

Yes, we admit we have let some of you down at times - it hurts us too - and we don't pretend that there aren't the odd 'hot properties' that sell faster than we can get them; but anyone who has seen our shop (better still our warehouse) will confirm that we are a LONG LONG way from waving the white flag that others are starting to grope for. Whatever odd items appear and disappear in the meantime, please consult us about stocks in general. Consult PARCO rather than other commentators, as we share neither their restricted view nor their morbid readiness to press the self-destruct button prematurely. Above all, DON'T PANIC.

ED:- TI\*MES is controversial at last! It is the function of the User group to be a common meeting point for the EXCHANGE of news and ideas It's good news that there's plenty of goodies left for our computer. We need the dealers in products for our computer just as much as they need you as customers. Without this interaction the 99 would surely fade away so keep on supporting them.

CLIFF SPENCE OF 33 NEATH ROAD, WHITEHALL, BRISTOL wrote to tell us of a meeting he arranged in BRISTOL in January. They had 3 TIs running with demos and a visit from a representative from PARCO. It was attended by 40+ users and as a result regular meetings will be held.

ED:- We are interested in any local groups being set up and will do whatever we can to promote and support them. If anyone else wants to take the same initiative as CLIFF let us know.

MAURICE RYMILL of Bournville, BIRMINGHAM writes:- The pin connections for the COR-COMP RS232 interface to the TANDY CGP 115 printer are as follows. (You require a 25 pin D plug and a 4 pin DIN plug and a length of 4 core cable)

Pin 3 on 25 pin TO pin 4 on DIN plug. Pin 7.....TO pin 3  
Pin 1 .....TO pin 1..... Pin 20 TO Pin 2

Entry code for file is - "RS 232.BA=600.DA=7.PA=N" but to list a program to the printer only requires - LIST"RS232.BA=600" The quote marks are essential.

Tandy occasionally put this 4 colour printer on offer at around £130.

R.TRUEMAN writes:- Thank you to all the people who bought my games, Flooraway, Flip Flap, Noteworthy, Crazy Cliff etc and thank you to Mr.S.Shaw for publishing them.

Unfortunately Mr Trueman and his TI have parted company but you can still buy this excellent programs from STAINLESS SOFTWARE. Crazy Cliff is reviewed in this issue of TI\*MES.

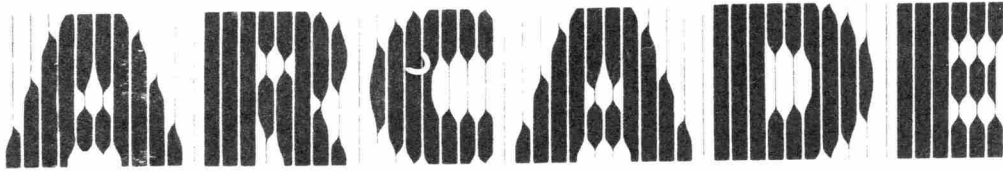
ROSS SARGENT of FOLKESTONE writes re Graham Baldwin's Parsec bug of crashing at the top of the screen. "I too get this but only on level 2. Still on bugs in Superhod from Parco I push the joystick down whilst pressing the fire button and 'superhod' becomes immune to danger with the bulldozers passing right through him."

Mr B. SHOLANKE of WELWYN GARDEN CITY, HERTS is interested in contacting TI owners in his area possibly with a view to holding meetings. Please ring him on Welwyn 27232.

JOHN BINGHAM of Esher, Surrey writes:- I came across a little quirk of Ext. Basic. Not really something I suggest be used usefully, but something to be avoided. GOSUB should not be used before ELSE which is followed by a multiple statement, otherwise only the first statement of the multiple statement is skipped. eg IF I=1 THEN J=1::GOSUB 200 ELSE K=1;;J=2. Whether I=1 or not this statement will result in J=2. If instead - I=1 THEN GOSUB 200::J=1 ELSE K=1::J=2. THEN I=1 WILL RESULT IN J=1 AND IF I =1 will result in J=2.

RAY FEARN of NOTTS writes " Any info on IC inserts for speech synthesiser. Did THORN EMI ever release any cartridges.

ED:- There is no word on the synthesizer. Another brilliant conception that never made it. Could they have been similar to the extra vocabulary modules available for Speak and Spell. The THORN EMI games came within a whisker of release but never made in onto the market. Great pity as we have seen the games and they were excellent.



# HARDWARE

211 Horton Road Fallowfield Manchester M14 7QE Tel 061 225 2248

## PERIPHERALS

T.I. Boxes - Call, there may be some secondhand units. No more new T.I. Boxes.  
T.I. Disc Control cards. As above.  
T.I. 32k R.A.M. cards £95.00. T.I. Internal Disc Drives - £125.00  
Myarc Disc Drive Control Card - 4xds/dd or ss/sd Disc Drives. + additional  
CALLS. (LINK,DIRectory,LOAD)  
Myarc RS232 card £125.00 - Baud up to 19200 - True Centronics.  
Myarc 128k card £249.95 - To order only - - 32k R.A.M. + 96k R.A.M. disc/Print  
Spooler.  
Myarc Mini Peripheral Box - Compact box, containing 32k R.A.M., 1 x RS232, 1 x  
Centronics + Disc Controller. Does not contain drive, but has space and power  
supply for 2 x half height ds/dd.  
Disc Drives - With the number of different types the Myarc Control card makes  
available, please write or call for prices.  
Boxcar stand alone 32k R.A.M. £125.00  
Boxcar stand alone RS232 + Parallel - £119.95.  
Axiom Centronics stand alone £109.95  
Alphacom 42 Thermal Printer - £145.00  
Navarone Widgit £39.95 - Sorry about the increase, but it's overdue.  
Super Sketch £65.00. See software for screen dumps.  
Personal Peripherals Joysticks (pair) £24.95  
Newport Prostick £22.00 (single) As strong as my original and as good.  
Super Champ (single) joystick £14.95. Not many left.

## SERIOUS SOFTWARE

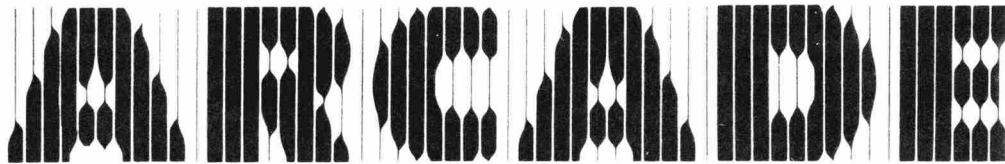
Navarone Console Writer - Requires just Console and printer £49.95 Now available  
Navarone Database - Requires Disc system + 32k R.A.M. £65.00 Now available  
Household Budget management £ 9.95  
Home Financial Decisions £ 9.95  
T.I. WRITER Requires disc, 32k, RS232 + Printer £74.95  
MULTI-PLAN Requires disc 32k, The baffling, but brilliant spreadsheet. £74.95  
T.I. LOGO II Requires 32k. The Child Appropriate Language £74.95  
DFX Screen Dump - Dumps anything (including Super Sketch, modules etc. to  
Epson control printer. (Shinwa, Star, Microline etc.) Requires load interrupt  
switch to be fitted. Call for details. (ON DISC) £24.95  
DISC REPAIR KIT. Replacement for the Navarone Disc Fixer, this program is  
BRITISH WRITTEN!!! It does more than Navarones, is more user friendly and has a  
better documentation. DISC £29.95

EXTENDED BASIC - £74.95. Newly manufactured under licence from T.I. with  
improved manual.

SST COMPILER - Available around June.

*Look forward to seeing you all in Brighton on 28th.  
(I'll be better prepared too!)*





# HARDWARE

211 Horton Road, Fallowfield, Manchester M14 7QE Tel. 061-225 2248

## SOFTWARE

### GAMES (Arcade)

|               |        |                                         |        |               |        |
|---------------|--------|-----------------------------------------|--------|---------------|--------|
| Frogger       | £24.95 | Q*Bert                                  | £24.95 | Midnite Mason | £24.95 |
| Micro Surgeon | £19.95 | Fathom                                  | £19.95 | Moonsweeper   | £19.95 |
| Demon Attack  | £19.95 | Star Trek                               | £19.95 | Buck Rogers   | £19.95 |
| Hopper        | £14.95 | Munchman                                | £ 9.95 | Indoor Soccer | £14.95 |
| Miner 2949'er | £23.95 | (Not many left and no more being made.) |        |               |        |
| TI Invaders   | £ 9.95 | Connect 4                               | £ 9.95 | TombstoneCity | £ 9.95 |
| Sewermania    | £19.95 | Superfly                                | £ 9.95 | Meteor Belt   | £ 9.95 |
| Space Bandits | £19.95 | Bigfoot                                 | £ 9.95 | Arcturus      | £48.00 |

**ATARI SOFT ARE BACK FOR THE TI.  
AND AT GREAT PRICES TOO.**

|               |        |             |        |
|---------------|--------|-------------|--------|
| DONKEY KONG   | £14.95 | JUNGLE HUNT | £ 9.95 |
| DEFENDER      | £14.95 | PAC MAN     | £ 9.95 |
| Ms. PAC MAN   | £14.95 |             |        |
| POLE POSITION | £14.95 |             |        |
| MOON PATROL   | £14.95 |             |        |
| ROBOTRON      | £14.95 |             |        |

## SOFTWARE

### GAMES (Adventures)

Scott Adams.

|                                                 |        |
|-------------------------------------------------|--------|
| Return to Pirates Isle - Tough sequel to Pirate | £19.95 |
| Adventure/Pirate - The one that started it all  | £19.95 |

INFOCOM. - Infocom are the worlds no.1 for Adventure games. These are very complex adventures (text only) and supplied on Disc only. They are expensive, but you do get a great deal for the money, both in terms of quality of packaging and the standard of game leaves nothing to be desired. At the time of placing this advert, the games have been ordered, but there is no guaranteed delivery date or availability. Expected E.T.A. is hopefully in time for the show (28th April)

|           |        |             |        |            |        |
|-----------|--------|-------------|--------|------------|--------|
| ZORK I    | £39.95 | ZORK II     | £44.95 | ZORK III   | £44.95 |
| ENCHANTER | £39.95 | DEADLINE    | £49.95 | WITNESS    | £39.95 |
| STARCROSS | £49.95 | INFIDEL     | £49.95 | PLANETFALL | £39.95 |
| SORCERER  | £44.95 | CUT THROATS | £39.95 |            |        |

AND ANNOUNCING - HITCHHIKERS GUIDE TO THE GALAXY £39.95

*Sorry about the lack of fancy artwork, but this is the best way of giving you the most accurate picture.*

*Howard Greenberg*

# TEXAS INSTRUMENTS 99/4A HOME COMPUTER SYSTEM

<sup>TM</sup>  
**SUPER  
SKETCH**

**99/4A**

MAGAZINE



**APPOINTED**

WHOLESALE

# DEALER



# PARCO ELECTRICS



*for a free price list and further information, send s.a.e. to:*

PARCO ELECTRICS, 4 DORSET PLACE,  
NEW STREET, HONINGTON, DEVON,  
EX14 8QS. TELEPHONE (0404) 44425

*European office:*

PARCO ELECTRICS, AM GASTHAUS 2, 2971. HINTE 2,  
LOPPERSUM, WEST GERMANY.  
TELEPHONE (04925) 1773.

CLASSIFIED ADVERTS

**COMPUTER HOME SERVICE**

\*\*\*\*\*  
**TI99/4A DUST COVERS.** Best selling standard cover in semi-clear durable PVC bound and stitched edges. £3.25+35p P+P.

**De luxe cover** in lined leathercloth with clear keyboard panel. £4.00+35pP+P.

**DEMAGNETISER.** Essential for clean programs. Use with CSI Only £2.30+25pP+P

**TV AERIAL SPLITTER (SWITCHED)** will prevent wear on TV socket and cable. Simply flick a switch to £1.99 +25pP+P

**BOOK AND SOFTWARE COMBINATIONS.**

These unique new releases offer excellent value for money. The book lists all taped programs and helps you understand the program structure. The programs range from arcade games to utilities, educational, file management, finance and planning aids. Each package contains much detail and is complete with handsome storage case.

**TI CALC:** Ex basic Spreadsheet £15.50

**TI TRIVIA DATA BASE:** Ex Basic. £15.50

**TI99/4A: 24 BASIC PROGRAMS** £14.99

**TI99/4A: 51 FUN AND EDUCATIONAL PROG's** £9.49

ADD £1.00 POSTAGE ON COMBINATIONS

**OTHER BOOKS FROM SAMS PUBLISHERS**

**FORTH PROGRAMMING** by Scanlon. £13.50

**BEST BOOK OF MULTIPLAN.** -£9.50

**SPECIAL OFFER.**

**TI99/4a Users Handbook.** Published by Weber. This book is written in a clear, concise manner that allows a "first time" computer user to operate and program the TI99/4a computer. It also contains 350 PAGES of information that will be of interest to the experienced computer user. Information on the use of Disk drives, DOS commands, advanced graphics and sound commands, printer usage, or file handling. There is a reference guide to all TI Standard and Extended commands for the TI99/4a. This book is an absolute bargain **LESS than HALF PRICE: -MEMBERS ONLY** £5.95

\*\*\*\*\*

**GETTING STARTED WITH THE TI99/4A** by Stephen Shaw....£5.50 Easy to read.

**MASTERING THE TI99/4A** by Peter Brooks. £5.50 or **both** these books **for only £10.00** These prices include postage and packing.

\*\*\*\*\*

Send Order to **COMPUTER HOME SERVICE, 40 BARRHILL PATCHAM BRIGHTON BN1 8UF.**

PLEASE ALLOW 28 DAYS TO PROCESS

\*\*\*\*\*

**TI88ES back issues** available are ONLY issues 4 and 5. They cost only £1 each to members. **TI99/4A EXCHANGE, BRIGHTON.**

**BRAND NEW** boxed modules. Parsec £8 Tombstone City, Attack, Chisholm Trail, Blast £5 each. Teach yourself basic(cass) £3.

Phone 021-471-2391 evenings.

**FOR SALE** - TI99/4A + Ext.Basic + Joyst + approx.£100 of books. -£100 the lot (cannot split) Phone DERBY 833225 Thursdays to Sundays(evenings after 7pm)

**FOR SALE.** The fantastic SUPER SKETCH graphics tablet for the TI99/4a! Absolutely brand new!(still boxed). Paid £65. Bargain at just £50(ono). D.Prince, 18 Hallw\od Ave, Salford, M6 8WW (loads of modules and peripherals too!!)

**WANTED.A** TI99/4a computer Peripheral Box, 32k ram card, Disk drive(internal), Disk controller card, TI Writer module and TI printer. Ring Chris 0884 258272.

**FOR SALE.** Offers accepted. Multiplan £70. Minimemory £40. Editor/assembler £35. Parsec £8. Speech synthesiser £20. Terminal emulator II £25. Peripheral expansion Box £80. Tandon Disc drive + controller card £220. 128K Memory expansion card £100. RS232 card £90. Fundamentals of TI 99/4a Assembly Language £14. Mike Eydenberg, 4 Hintlesham Close, Stowmarket, Suff. Stowmarket 676093.

**FOR SALE.** Disc controller card and disc manager. £50 ono. Tel 01-316-0244.

**EXCHANGES.** Will swap Statistics, Personal Record Keeping, Tombstone City, Indoor Soccer, Music maker.(or combination) **WANTED-** Moonsweeper, Popeye or similar. Contact USER GROUP - 0273 503968.

**FOR SALE.** TI99/4A, 32K expansion, disc controller, disk drive. £300. CARTRIDGES - Extended Basic £40, Parsec £9, Hunt the Wumpus £9, Munchman £9, Soccer £9, Speech synthesizer £25, some cassettes. Phone 01-845-9725 after 6.30pm.

**ADVERTISE YOUR SWAPS HERE NEXT ISSUE FREE. REACH TI99/4a USERS FAST. ADVERTISE ON THIS PAGE FOR 5P A WORD. (MEMBERS ONLY) Trade Advertisements Rates on application.**

**WORDBUG BUNNY cont'd  
from page 43**

2350 PRINT "DO YOU WANT TO R  
LAY AGAIN

PRESS Y FOR

YES N FOR NO"

2360 CALL KEY(3,KY,S)

2370 IF KY=78 THEN 2400

2380 IF KY=89 THEN 220

2390 GOTO 2360

2400 END

2410 IF RND>.2 THEN 2460

2420 IF RND>.5 THEN 2450

2430 CALL CHAR(158,"82828254  
38384444")

2440 GOTO 2460

2450 CALL CHAR(158,"28448254  
38384444")

2460 RETURN





## RANDOM EYES

Ho hum. I hope you enjoyed the debugging practice in the previous adventure article. (Did anyone notice the problem? Did anyone read the article...?)

In listing 2 lines 190 and 200 should read:

```
190 FOR I=1 TO 5
200 IF POS(Q$,V$(I),1)=0 THEN 230
```

Right. On with the words.

In many adventures the most intriguing part for the player is the manipulation of the objects scattered around the scenario to solve the various puzzles and problems set by the programmer. Let's start by looking at ways of deciding where the objects will be placed in the adventure. In the last article I suggested a method of assigning a reference number to every noun (or object). This routine uses a similar technique to assign a location number to each object. A simple array, OB\$( ), will contain a location number in each element.

```
100 REM LISTING 3
110 FOR I=1 TO 5
120 READ OB$(I)
130 NEXT I
140 DATA 2,5,99,4,1
```

I hope you can see that object 1, ie OB\$(1), will now appear in location 2, object 2 in location 5 and so on. The number 99 in DATA is used to signify that the object is being 'carried' by the player, ie the player is treated as location 99. (This may all seem A about F but bear with me for a while and I hope the reasons for working this way round will become clear.)

In practice a 2-dimensional array can be used to store both object descriptions and their locations. Using the nouns from listing 2 and the locations from listing 3 the table would look like this:

|      |      |       |       |      |                          |
|------|------|-------|-------|------|--------------------------|
| RING | DOOR | STONE | CHAIR | BOOK | (Equivalent of N\$( ).)  |
| 2    | 5    | 99    | 4     | 1    | (Equivalent of OB\$( ).) |

Obviously this is a more elegant method, but tends to make the listings and explanations rather more complex so I'll stick to single arrays for the moment, if you don't mind.

Now that we have our objects and their original locations sorted out we can think about telling the player where they will be found in the adventure.

When a player moves to a new location he needs to be given not only a description of his surroundings but also a list of which objects can be found there. This routine prints that list.

```
100 REM LISTING 4
110 PRINT "I CAN SEE"
120 FOR I=1 TO 5 (Or number of objects in the adventure.)
130 IF OB$(I)<>STR$(LOC) THEN 150
140 PRINT N$(I)
150 NEXT I
```

This listing compares the object location list with the current location variable (LOC) and prints out any noun that matches. If the location contains no objects at all we'll have a blank space on the screen after printing "I CAN SEE", which would look silly. Therefore we could add a simple counter at line 135 and print "NOTHING OF INTEREST" if the counter has not been incremented.

Now that the player has arrived at a location and been told what he can find there he may wish to TAKE or DROP an object. Dropping an object is fairly straightforward and involves only two checks - does the object exist and is the player carrying it? The first check is covered by the verb/noun recognition routine and the second simply involves checking the objects' location. If it is not "99" the player is not carrying it and we can chastise him with a message to that effect. If he is carrying it we can change its location from "99" to that of the current location.

You may wish to play dirty and cause any object dropped in a particular location, say, a swamp, to disappear for ever. We could do this by assigning any object dropped in the swamp the location of "88", for instance, to signify that it can no longer be used by the player. A null ("") could be used instead if your program won't crash when it sees one.

If the player wishes to TAKE something then things get a little more complicated. Again we have to check if the object is in the vocabulary then check if the player is already carrying it. Next we must check if the object is in the player's current location and print "CAN'T SEE IT HERE" if it isn't. Now, to back-track a little, when setting up our list of objects and their locations we may decide that there will be 30 objects, of which only the first 20 can be taken, the others being too large or immovably fixed ( a mountain or a tree, for example). Therefore, if the noun number found by the word recognition routine is greater than 20 the object cannot be taken and a message "CAN'T TAKE THAT" is printed.

Please note that the checks should be carried out in this order or logical inconsistencies may result. When all these checks are satisfied the player can TAKE the object, altering its location to "99".

To be realistic (and to prevent the player solving the adventure too easily) we cannot allow too many objects to be carried, so we could restrict his carrying capacity to, say, five objects, and print "CARRYING TOO MUCH" if he attempts to take any more. A simple 'counter' variable, incremented every time he takes an object and decremented when he drops it will take care of that. If we could find room in memory we could even assign an arbitrary weight to each object, because it's rather silly to allow a player to carry five objects, each one of which, in reality, might require all his strength to lift.

So, to recap, our player can now wander around the adventure, taking and dropping objects to his heart's content. What we must do now is build in some logic to detect when he has solved a particular problem. For example, if one problem involves using a bucket of water to put out a fire we must check that the player is carrying the bucket, that the bucket contains water, that the player is in the right location to solve the problem and that the fire has not been put out already. If all these checks are passed, then on the command, "POUR WATER", we can print "OK THE FIRE'S OUT" and set a variable, say, F, to -1. Should he try to put out the fire again at any time, we can test the status of the variable and tell him, "YOU'VE ALREADY DONE THAT".

These logical checks can be tedious to write but are obviously essential to the plot and must be made bomb-proof, or the player may be able to solve the puzzles in ways you never thought of. Even worse, he may not be able to solve them at all!

The EXAMINE command can give a lot of fun to the programmer, as he can tailor the responses from useful, via non-committal, to downright obscure. Provided you play fair and don't actually lie in the responses you can tease a player into a trap (not necessarily fatal) that he might well have avoided if only he'd taken the time to think about his next move.

I think that puts the bare bones of a simple adventure together, so let's tidy it up and make it a little friendlier. Here are some extra commands you may like to include to this end.

SAVE can be used to store the players' current status on a cassette file so that he can break off from the game and play again some other time without having to type his way through the whole adventure for the umpteenth time. This involves opening a file and printing to tape every variable and array element that could be altered in the course of the game. The most economical way to do this is by converting all the relevent information into one string that can be saved on a single segment of tape. It's not as tricky as it sounds, honest... Conversely, we need some more code to get the information back off tape and into the program, all of which takes up precious memory. I must admit I tend to ignore SAVE and use the available memory purely for the adventure.

INVENTORY (or INV) is both useful to the player and simple to program. As the player gets stuck at various points in the adventure his efforts to solve the problem confronting him will eventually cause the location description etc to scroll off the screen, leaving him to wonder what the original problem was. INV branches the program back to print the current location description and hence the objects that can be seen there and the objects carried by the player.

HELP requires some careful thought from the programmer. Ideally it should give the player an extra clue at a tricky spot rather than give away the brilliantly logical answer you've been so careful to conceal. A simple ON LOC GOTO.....(Help responses) is probably the easiest way to program this command, with a laconic "YOU'RE ON YOUR OWN HERE" for locations where no help is available.

Looking back over this article I can see that I've left out an awful lot of detail (after all, whole books have been written on the subject), but I hope it has given you an idea or two about the pleasures and pitfalls of adventure programming.

END

\*\*\*\*\*

Being slightly richer than I thought (er, I forgot about the gas bill) I treated myself to some new modules recently. The first was Protector 2, from Atarisoft, (for the kids, of course) and on plugging in and selecting the game I was entranced by some of the nicest music my TI has ever produced. If it wasn't written by J.S. Bach then it should have been; I was greatly impressed. The game itself I found less impressive, but it keeps the children amused, even if they do spend their time blasting away at the little men they should be rescuing...

The other modules were PRK and Household Budget Management, one of which I hoped would solve a budget record problem at work. Alas, the HBM won't do it but the PRK is showing possibilities, if only I can devise a formula to make it perform all my calculations at one pass. Sigh of relief - the PRK module is compatible with my Alphacom printer (or should that be the other way round?).

Total Inconsequence Dept. The 'repeat' argument in CALL HCHAR and VCHAR can have a value of up to 32767, which, if used, takes about 35 seconds to execute. The only use I can think of for the phenomenon is as a delay. Any other offers?

TRACE is a very useful command when debugging your latest (or even worse, someone elses) masterpiece but it does have the disadvantage of disrupting any screen display. (Just like the cassette operating instructions, darn it!) Don't forget that TRACE and UNTRACE can be used as program statements, so you can turn TRACE on and off from within the program and, hopefully, keep your screen display intact.

Happy computing,

*Orlando Baldwin*

32, Ellesmere Drive, SOUTH CROYDON, Surrey. CR2 9EJ

# TI WRITER TIPS

by Allen Burt.

This time I would like to start with page 98 of the TI-WRITER Manual --"SPECIAL CHARACTER MODE" . There are two potentially misleading statements on this page.

- 1) The third paragraph implies that you have to use the TRANSLITERATE command if you wish to use a combination of two or more special codes. This we now know to be untrue. In the last article I demonstrated how to input three character combinations.
- 2) The second error is the page reference at the end of paragraph three(...see example on page 121....should read 107).

## REMINDER

SPECIAL CODES are accessed by means of CONTROL key and KEY 'U' -the cursor then appears as CHR(95) the underline character. You can then enter the special code by pressing either SHIFT or FUNCTION together with the KEY as shown on page 146 of manual. (This is only necessary to obtain ASCII codes 0 - 31).

## MORE TIPS FOR THE "EDITOR".

- a) Space can be saved on TI-WRITER files by removing all the blank lines within the layout. Once you have decided how many blank lines are needed for each part of the document you can place a number of LINE FEEDS at the end of the line preceeding the blank line area with the aid of CONTROL 'U' & SHIFT 'J' -ie CHR(10), (this document is produced without any blank lines on the screen display). This means that one record in the file can be used for one line of characters together with a number of blank lines.

Each line of writing on the screen display requires one record in the file (Files are Display Variable 80 which means that each record can contain up to 80 Characters -each blank line occupies one record).

- b) Sometimes when finishing a line there is not enough space to enter the Carriage Return(CR) without word-wrap carrying the last word and the CR over to the next line. This can be avoided by using CONTROL 'U' & Shift 'M' [ CHR(13) ] -but you will have to ensure that you are not on the line above...."\*End of file Version"..... because this method of inputting a CR will not put you on the next line. You have to 'Insert' a line first and then use the 'arrow keys' to move between lines.

This method of using the CR is particularly useful when editing data tables. It does not produce an extra line that has to be deleted afterwards.

- c) Most printers can be prevented from stopping when the 'paper out' signal is activated. The Gemini 10 requires CHR(27);"8" codes. This is useful when printing on single sheets because it allows you to print to the bottom of the page. Make sure that you do not have too many lines otherwise you will print on the platten. This is especially important if you are using tip (a) -the line numbers at the side of the screen will not correspond with the actual number the printer will print.







# BACK TO BASICS

THIS IS A GAME FOR YOUNG CHILDREN, 4-8 YEARS OLD! THEY ARE SHOWN A PICTURE, AND MUST SPELL IT CORRECTLY BECAUSE THERE ARE PICTURES, LITTLE CHILDREN FIND IT MORE ENJOYABLE AND EASIER TO PLAY THAN "HANGMAN"

WORBUG BUNNY; BY SAM NASH, LEEDS.

DONT TYPE THE REMS. - THERE IS NOT ENOUGH MEMORY

LINE 100-240 INITIALIZE AND COLOR SETS

250-300 DEFINE CHARS FOR RABBIT, CARROTS AND BUGS

310-380 TITLE SCREEN, LOOP TO READ AND PLAY MUSIC

DATA FOR MUSIC, READ IN LINE 350. X=TONE, Y=DURATION. Y IS MULTIPLIED BY 2 IN LINE 360 TO GIVE PRESENT DURATION, ALTER THIS FOR YOUR OWN TASTE

470-490 SCATTERS 30 "BUGS"

510-540 READ CHARS FOR 1ST 7 PICTURES

560-600 SUBROUTINE TO READ NAMES AND CHAR NUMBERS FOR EACH PICTURE

610-840 DATA FOR 1ST 7 PICTURES, READ AT LINES 520 AND 570

870-1740 MAIN ROUTINE: BRIEFLY-RANDOMLY CHOOSES THE 7 PICTURS AND CHECKS THE PLAYERS SPELLING, THE RABBIT HOPS ABOUT IF A LETTER IS CORRECT, EATS THE CARROTS AND SQUASHES THE BUG AFTER A WORD IS CORRECT

SPELT IT IS "MARKED-OFF", SO IT IS NOT CHOSEN AGAIN

1750-1780 SUBROUTINE FOR A NEW PAGE

1790-2050 READ, AND DATA FOR 2ND 7 PICTURES

2060-2340 READ, AND DATA FOR 3RD 7 PICTURES

2350-2410 END OF GAME

SUBROUTINE FOR "BUGS PINCHING"

```

10 REM SAM NASH,
    43 UPLAND GROVE
    LEEDS LS8 25X
    -----
20 REM WORD BUG BUNNY
    TI BASIC
    -----
30 REM
100 DIM WD$(6), A(6), B(6), C(6)
    , D(6), E(6), F(6), G(6), H(6), I
    (6), J(6), K(6), L(6), M(6),
    N(6), O(6), P(6)
110 RR=20
120 RC=15
130 RD=1
140 CD=1
150 RANDOMIZE
160 FOR Q=1 TO 13
170 CALL COLOR(Q, 2, 16)
180 NEXT Q
190 CALL COLOR(14, 9, 1)
200 CALL COLOR(15, 16, 1)
210 CALL COLOR(16, 2, 1)
220 CALL CLEAR
230 RESTORE
240 CALL SCREEN(13)
250 CALL CHAR(144, "2628306EF
E1E270C")
260 CALL CHAR(146, "24140C767
F78E430")
270 CALL CHAR(140, "E0F0F8783
C0E0601")
280 CALL CHAR(154, "FFFFFFFFF
FFFFFFF")
290 CALL CHAR(158, "828282543
8384444")
300 CALL HCHAR(1, 1, 31, 768)

```



```

310 PRINT "          WORDBUG B
UNNY          "
320 CALL HCHAR(24, 1, 31, 32)
330 PRINT ::::::"SPELL THE P
ICTURES NAME TO LET THE BUN
NY RABBIT EAT THE CARROTS
AND SQUASH THE BUGS"
340 FOR R=1 TO 93
350 READ X, Y
360 CALL SOUND(200*Y, X, 9)
370 NEXT R
380 CALL HCHAR(1, 1, 31, 768)
390 DATA 392, 2, 392, 2, 349, 2, 3
30, 2, 349, 1, 392, 2, 262, 4, 294, 1
, 330, 2, 392, 2, 392, 2, 494, 2
, 494, 1, 523, 2
400 DATA 523, 4, 523, 1, 523, 2, 6
59, 2, 349, 2, 329, 2, 349, 1, 330, 1
, 349, 6, 392, 1, 392, 1, 392, 1
, 349, 2, 392, 2
410 DATA 294, 7, 392, 1, 392, 3, 3
92, 1, 349, 2, 330, 2, 349, 1, 392, 2
, 262, 4, 294, 1, 330, 2, 392, 2
, 392, 2
420 DATA 494, 2, 494, 1, 523, 2, 5
23, 4, 523, 1, 523, 2, 659, 2, 349, 2
, 392, 2, 349, 1, 330, 1, 349, 6
, 392, 1, 392, 1
430 DATA 392, 1, 349, 2, 392, 2, 2
94, 8, 659, 1, 659, 1, 659, 1, 587, 3
, 659, 1, 698, 1, 659, 2, 392, 4
, 523, 1, 698, 1
440 DATA 659, 2, 392, 4, 523, 1, 6
98, 1, 659, 2, 392, 4, 440, 1, 494, 3
, 523, 1, 494, 2, 440, 2, 440, 2
, 392, 3, 262, 1, 262, 2
450 DATA 392, 2, 392, 1, 349, 1, 3
30, 2, 262, 2, 262, 8, 330, 1, 294, 1
, 262, 8
460 RESTORE 610

```



```

470 FOR R=1 TO 30
480 CALL HCHAR(INT(RND*13)+1
2,INT(RND*26)+4,158)
490 NEXT R
500 GOSUB 1750
510 FOR R=1 TO 72
520 READ Q,H$
530 CALL CHAR(Q,H$)
540 NEXT R
550 GOTO 770
560 FOR W=0 TO 6
570 READ WD$(W),A(W),B(W),C(
W),D(W),E(W),F(W),G(W),H(W),
I(W),J(W),K(W),L(W),M(W)
,N(W),O(W),P(W)
580 GOSUB 2410
590 NEXT W
600 RETURN
610 DATA 33,0000000001030301
,34,2060FFFFFFFFCFFFFFFF,35,4060
F0F0F83CFCF8,36,00000000
00010307,37,FF7F1F3FFFFFFF
620 DATA 38,F0E080C0F0F8FCFE
,44,FFFFFFFFFFFFFFFFFC,50,4282
82623213111,56,00001020C
08000FF,62,02020303030FFFFFFF
630 DATA 39,0F0F1F1F3F3F3F3F
,41,000C8C8CCCCCCCCC,42,3F3F
1F1F0F0F0703
640 DATA 45,CCCC8C8C1CF8F,46
,0000000000030707,47,001F3F2
06CC00202,48,0080E030101
80808,49,030301
650 DATA 51,181010302060DE03
,52,30206040C0C0808,53,C6663
63616161616,54,010103030
2060D0F,55,86264A4B509326C7
660 DATA 57,161E1E1E0C0C38F8
,58,0000000000000F18,59,0000
0000007CE682,60,00000000
0001070E,61,1030202020E0BF3F
670 DATA 63,000000F03C070303
,64,1830606060301F03,91,3F3F
000000000FFFF,92,FC00000
0033FF,93,060C38F0C
680 DATA 94,0F1F101411100B0C
,100,121212121212FEFE,106,85
850505FD
690 DATA 95,E0F010501010A040
,96,077CFEFFFFFFEFFFF,97,C07C
FEFEFEFEFEFE,98,EEEEFEFF
90533212,99,EEEEEEFE1294989
700 DATA 101,909090909090FEF
E,102,000000000003060C,103,0
00000007FC0007D,104,0000
0000F01C04F6,105,18302161414
0404
710 DATA 107,130B0908F8,108,
00008080F80C0603,109,474F5F7
C1C1F0F07,110,80C0E0FFE0
E0C080,111,03070FFE0E0F0703
720 DATA 112,C3E3F77E70F0E0C
0,113,00000000C0FFD5DA,114,0
00000000C060B0,115,D5DA
D5DAD5DAD5DA

```



```

730 DATA 116,1F8302810281028
1,117,00F0B050B050B05,123,FF
810000000081FF
740 DATA 118,D5FACFC1C1C0C0C
,119,02810281C2417F01,120,B0
50B050B050B0F,121,C0C0C0
C0C0C0C0C,122,011F78C0C0F89F
81
750 DATA 124,80F81E03031FF98
1,125,0000003078D8CC8C,126,8
08080808080808,128,01010
10101010101,129,8C0C0C0C0C8C
8CD8
760 DATA 130,80808080E0781F0
1,131,00000000000081FF,132,0
1010101071EF880,133,F87
770 GOSUB 560
780 DATA CAT,33,34,35,32,36,
37,38,32,39,154,154,41,42,15
4,44,45
790 DATA DOG,46,47,48,32,49,
50,51,32,32,52,32,53,54,55,5
6,57
800 DATA HAT,32,58,59,32,60,
61,62,63,64,91,92,93,32,32,3
2,32
810 DATA MAN,32,94,95,32,32,
96,97,32,32,98,99,32,32,100,
101,32
820 DATA CAR,32,32,32,32,102
,103,104,32,105,106,107,108,
109,110,111,112
830 DATA FLAG,32,113,114,32,
32,115,116,117,32,118,119,12
0,32,121,32,32
840 DATA CUP,32,32,32,32,122
,123,124,125,126,32,128,129,
130,131,132,133
850 GOSUB 870
860 GOTO 1980
870 FOR T=1 TO 7
880 GOSUB 2410
890 Y=0
900 W=INT(RND*7)
910 IF WD$(W)="" THEN 900
920 CALL HCHAR(5,7,A(W))
930 CALL HCHAR(5,8,B(W))
940 CALL HCHAR(5,9,C(W))
950 CALL HCHAR(5,10,D(W))
960 CALL HCHAR(6,7,E(W))
970 CALL HCHAR(6,8,F(W))
980 CALL HCHAR(6,9,G(W))
990 CALL HCHAR(6,10,H(W)),
1000 CALL HCHAR(7,7,I(W))
1010 CALL HCHAR(7,8,J(W))
1020 CALL HCHAR(7,9,K(W))
1030 CALL HCHAR(7,10,L(W))
1040 CALL HCHAR(8,7,M(W))
1050 CALL HCHAR(8,8,N(W))
1060 CALL HCHAR(8,9,O(W))
1070 CALL HCHAR(8,10,P(W))
1080 X=LEN(WD$(W))
1090 FOR R=14 TO (X*2)+12 ST
EP 2
1100 CALL HCHAR(7,R,32)

```

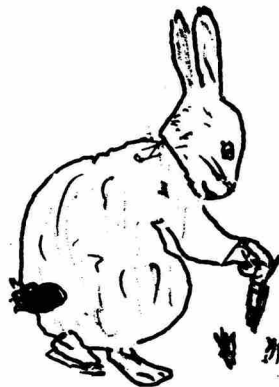




```

1890 DATA 112,000000000000F8F8
0C,113,C0E0E0E0E0E0E0E,114,0
000000000070505,115,FC86
82FF03,116,00000080E0F0303
1900 DATA 117,80808CFFF3331E
0C,118,050505FFFF,119,303030
F0F,120,0000000000000000
1910 DATA EGG,33,34,35,32,36
,32,37,32,38,39,40,32,41,42,
43,32
1920 DATA STEPS,44,45,46,120
,154,47,47,46,154,154,47,48,
154,154,154,32
1930 DATA LADDER,32,49,50,32
,32,49,50,32,32,49,50,32,32,
49,50,32
1940 DATA SNOWMAN,51,52,53,3
2,54,55,56,32,57,58,59,32,60
,61,62,32
1950 DATA SHIP,32,32,32,32,6
3,64,91,92,93,94,95,96,97,15
4,154,98
1960 DATA FRAM,32,32,32,32,9
9,32,100,101,102,103,104,105
,106,107,108,109
1970 DATA VAN,32,32,32,32,11
0,111,112,32,113,114,115,116
,117,118,117,119
1980 RESTORE 1790
1990 FOR R=1 TO 62
2000 READ Q,H#
2010 CALL CHAR(Q,H#)
2020 NEXT R
2030 GOSUB 560
2040 RESTORE 1910
2050 GOSUB 870
2060 RESTORE 2130
2070 FOR R=1 TO 77
2080 READ Q,H#
2090 CALL CHAR(Q,H#)
2100 NEXT R
2110 GOSUB 560
2120 GOSUB 870
2130 DATA 33,000103060C18306
,34,FF8080808080808,35,FF030
70D193161C1,36,FF8080808
080808,37,FF010101010101,3
8
2140 DATA 8101010101010103,3
9,80808080808080FF,40,010101
01010101FF,41,060C183060
C08,42,00000001070E183,43,00
1F20FF,44,00F80810E060380C,4
5
2150 DATA 00001E3466448684,4
6,246440C080F87C,47,00000111
100807,48,05070202808081
01,49,060C081010080C04,50
2160 DATA 6030180C0701,51,00
00000000F13E,52,0205041C24C4
7E,53,060486C44664361E,5
4,073F608080E0DF87,55
2170 DATA FF003C42000000FF,5
6,E0FC06010107FDE1,57,40705F
434040404,58,000000FF,59
,020EFAC202020202,60,6060606
06060606

```



```

2180 DATA 61,060606060606060
4,62,2020202030180F01,63,000
00000000081FF,64,0404040
40C18F08,91,00010101015D5454
,92
2190 DATA 0042424242420001,9
3,00E020E080E0008,94,0080808
0808E020E,95,545C0000007
0507,96,01010101010101,97
2200 DATA 808080808080808,98
,080E0000000E0206,99,1010000
00070507,100,01,101,FF,1
02,F20E00000008080E,103
2210 DATA 50700701030202,104
,00000202030203,105,00000302
838083,106,0404800080808
,107,010306070E0D1B15,108
2220 DATA FFDEAD5AA55AA55,1
09,C0F0B85CAC54AE56,110,3A35
6AD5EBD7EED5,111,AA55AA5
5AA55AA55,112,AB77AB57ABF7BB
57
2230 DATA 113,EAD56A753F1D06
03,115,ABD7ABD7AE7CF08,114,A
A57AA55AFFDBEFF,116,3C3C
3C3C7C7E7EFF,117
2240 DATA 0000030F1F3F7FFF,1
18,3FFFFFFFFFFFFFFFFF,119,FCFF
FFFFFFFFFFFFFF,120,0000C0F
0F8FCFEFF,121,FFEE44,122
2250 DATA FFEF470303030303,1
23,0303030303030303,124,0303
0303333333E1C,125,FFFEE0D
0C8C4C2C1,126
2260 DATA FFFF04081020408,12
7,008080C04060203,128,C1C2C4
C8D0E0C0C,129,8040205048
444221,130,1018080C04060203,
131
2270 DATA 701C0701,132,20201
0D0703C2721,133,814120100804
02C1,134,100804040808080
8,135,905828DC7C1E0601
2280 DATA BOX,32,32,32,32,33
,34,35,32,36,37,38,32,39,40,
41,32
2290 DATA FISH,32,32,32,32,4
2,43,44,45,46,47,48,49,50,51
,52,53
2300 DATA DUSTBIN,54,55,56,3
2,57,58,59,32,60,32,61,32,62
,63,64,32
2310 DATA CLOCK,91,92,93,94,
95,96,97,98,99,100,101,102,1
03,104,105,106
2320 DATA TREE,107,108,109,3
2,110,111,112,32,113,114,115
,32,32,116,32,32
2330 DATA UMBRELLA,117,118,1
19,120,121,122,121,121,32,12
3,32,32,32,124,32,32
2340 DATA KITE,125,126,127,3
2,128,129,130,32,131,132,133
,127,32,134,131,135

```



```

1110 CALL SOUND(200,(RND*100
0)+400,3)
1120 NEXT R
1130 V=0
1140 CALL KEY(3,KY,S)
1150 GOSUB 2410
1160 IF (S=0)+(KY<65)+(KY>90
) THEN 1130
1170 FOR R=1 TO X
1180 IF KY=ASC(SEG$(WD$(W),R
,1)) THEN 1270
1190 NEXT R
1200 IF V THEN 1130
1210 BR=INT(RND*12)+13
1220 BC=INT(RND*26)+4
1230 IF (BR=RR)*(BC=RC) THEN
1210
1240 CALL HCHAR(BR,BC,KY)
1250 CALL HCHAR(BR-1,BC,158)
1260 GOTO 1130
1270 CALL HCHAR(7,R*2+12,KY)
1280 Y=Y+1
1290 S=POS(WD$(W),CHR$(KY),1
)
1300 LE#=SEG$(WD$(W),1,S-1)
1310 RI#=SEG$(WD$(W),S+1,X-S
)
1320 WD$(W)=LE#&" "
1330 WD$(W)=WD$(W)&RI#
1340 V=1
1350 FOR Z=1 TO 4
1360 BR=INT(RND*13)+12
1370 BC=INT(RND*26)+4
1380 IF (BR=RR)*(BC=RC) THEN
1360
1390 CALL HCHAR(BR,BC,140)
1400 NEXT Z
1410 IF Y=X THEN 1440
1420 IF R=X THEN 1130 ELSE 1
170
1430 GOTO 1680
1440 FOR HF=1 TO X*10
1450 GOSUB 2410
1460 CALL HCHAR(RR,RC,31)
1470 IF (RR>12)*(RR<24) THEN
1490
1480 RD=-RD
1490 IF (RC>4)*(RC<29) THEN 1
510
1500 CD=-CD
1510 RR=RR+RD
1520 RC=RC+CD
1530 CALL GCHAR(RR,RC,RG)
1540 CALL HCHAR(RR,RC,145+CD
)
1550 CALL SOUND(-1,RND*330+1
0,12)
1560 CALL GCHAR(RR,RC+CD,RF)
1570 CALL HCHAR(RR,RC+CD,31)
1580 IF (RG=31)*(RF=31) THEN
1670
1590 IF (RG>90)+(RF>90) THEN
1630
1600 CALL SOUND(1,262,0)
1610 CALL SOUND(1,523,0)
1620 CALL SOUND(1,1047,0)

```



43

```

1630 IF (RG>140)*(RF>140) T
HEN 1660
1640 CALL SOUND(-600,-8,0)
1650 IF (RG>158)*(RF>158) T
HEN 1670
1660 CALL SOUND(-100,-6,0,11
0,0)
1670 NEXT HF
1680 CALL HCHAR(7,14,31,18)
1690 GOSUB 1750
1700 WD$(W)=" "
1710 NEXT T
1720 CALL HCHAR(7,14,31,18)
1730 GOSUB 1750
1740 RETURN
1750 FOR R=4 TO 9
1760 CALL HCHAR(R,6,32,6)
1770 NEXT R
1780 RETURN
1790 DATA 33,000003070C08181
,34,3CFFC3,35,0000C060301018
08,36,1020202020201010,3
7,0804040404040808
1800 DATA 38,10183F3A351A1D1
A,39,0000FFAA55AA55AA,40,081
8FAC5CAB58B0,41,0D06030
102050F0F,42,55AA55AABE55FFF
F
1810 DATA 43,70E0C080C060F0F
0,44,FF00000000000001,45,FF0
60C183060C081,46,8080808
0808080FF,47,03060C183060C08
1
1820 DATA 48,03070D193161C18
1,49,90909F9F9090909,50,0909
FDFD090909,51,00000001
01030202,52,7E7EFFFF000024
1830 DATA 53,0000008080C0404
,54,0203010003060C18,55,423C
00FF,56,40408000C0603018
,57,306040C8888888C8,58,1000
00001
1840 DATA 59,0C06021311111111
3,60,484830101010180F,61,001
000000000000FF,62,12120C0
8080818F0,63,000000001092543
8
1850 DATA 64,0000000000000001
F,91,070C003F3F3F2121,92,18C
3,93,10FE1010101010FF,94
,3F3171C1C0C9C0FF,95,212121F
F001100FF,96,000000F0303030F
F
1860 DATA 97,7F3F1F0F070301,
98,FFFFFFEFEFFCFCFC,99,0080C
0E070381C0E,100,1F181818
18191A1C,101,C0703C468603030
3
1870 DATA 102,0F0F08080C0C06
07,103,FFFF804020103C42,104,
F8F000000000071A,105,030
303030303864E,106,0301010101
1880 DATA 107,91FF1FF808894A
3C,108,12FFA3BFA1110907,109,
3CF8F0101020C080,110,000
00000007FFFC0,111,0000000000
FFFF

```

# RAMBLES ON FORTH

## FORTH

Start here....

Now that TI FORTH is available to anyone with a disk system, 32k, and EITHER Editor/Assembler OR Extended Basic....(separate disk for ExBas version) how do you use it?

You receive your disk with TI FORTH on it.

FIRST: Copy the disk, using your Disk Manager module.

If you have one of those new fangled double sided drives, initialise the copy disk as SINGLE sided: the double sided header is larger and makes a mess of the Forth screens! Later on maybe we'll go over formatting JUST the second side for your program screens... clever is Forth!

Now, put the original disk in a safe place!

The SYSTEM DISK carries most of the FORTH language on its files FORTH SCREENS. It is a feature of FORTH that it rewrites the disk files, and it is easy to accidentally overwrite your system disk! So don't do ANYTHING until you have a nice safe copy!

Now...

Editor Assembler: Select "LOAD AND RUN"

File name is: DSK1.FORTH

EXTENDED BASIC: Put the disk in the drive and select ExBas. Autoload takes care of the rest.

You will see the message 'Booting Forth', followed by a menu of options, and after a short delay, a cursor will appear.

A menu will appear: only the 'kernel' of Forth has been loaded. To use Forth, you must select portions from the menu: the more you load the less memory is free for your program.

We will use the menu to create a 'custom' Forth which loads pretty quickly... note as you use these Menu options just how slowly they function!

Fortunately, we can load our desired selection, then BSAVE the whole lot, to be BLOADED when required, in one piece, quickly!

As a beginner we will need to boot most of Forth...

This is MY selection. First type EMPTY-BUFFERS [ENTER]

Now, the SCREEN DUMP is located on SCREEN 72, and may not be suitable for your printer. Lets have a look....

In order to EDIT screen 72, we need to load an editor. Choose EITHER the 40 column editor (-EDITOR) or the 64 column editor (-64SUPPORT). To load these just key in the name, with the dash in front, and press ENTER.

Now when the cursor reappears, type in 72 EDIT [enter]

Screen 72 will be loaded and will appear on screen.

As supplied, screen 72 is set up for RS232. If you use PIO, make the following changes:

Line 2: Change >RS232 to >PIO

Line 3: Change >RS232 to >PIO

Line 4: Set the string to the actual file name used by your printer, that is " PIO" or " PIO.CR" and so on.

Remember the space after the first quotes!!!!

Check the screen: some versions have a typing error! If you see the word PAB\_ADDR, you must change it to PAB-ADDR.

continued.....>

Now press FCTN 9 (BACK) to return the cursor to the bottom of the screen.  
Now we write the amended screen to disk by typing FLUSH [enter]

Lets make sure everything really is been cleared out: type in COLD [enter]  
and when the cursor reappears, EMPTY-BUFFERS [ENTER]

And we are ready to customise FORTH.

Type EMPTY-BUFFERS [enter]

Now type in, on one line:

-PRINT -COPY -VDPMODES -BSAVE [enter]

Thats the essential FORTH.

When the disk stops at last and the cursor reappears, let's put a marker flag  
in there: type in:

: FLAG1 ; [enter]

(colon, space, FLAG1, semicolon.... then [enter])

A bit more Forth: type in, on one line:

-GRAPH -DUMP -FLOAT [enter]

Now another flag. Type in

: FLAG2 ; [enter]

and finally, your choice of editor, either -64SUPPORT or -EDITOR [enter]

Thats almost everything you are likely to need.

Lets save it! To save EVERYTHING in memory, and to locate it on the disk from  
screen 51 onwards (deleting what was previously on those screens!!!!):

Type in:

' TASK 51 BSAVE [enter] The first character there is a FCTN 0, a single  
quote.

When the BSAVE has ended, you need to amend the "BOOT" screen, screen 3, to  
BLOAD your binary image.

Type in 3 EDIT [enter] and adjust your screen 3 to look something like this:

```
SCR #3
 0 ( WELCOME SCREEN )
 1 BASE->R HEX 10 SYSTEM ( Clears Screen )
 2 0 0 GOTOXY ." Loading TI Forth " CR 10 83C2 C! ( Quit Off )
 3 DECIMAL 51 BLOAD 16 SYSTEM MENU
 4 1 VDPME !
 5 0 DISK_LO !
 6
 7 180 DISK_HI !
 8
 9 : FREE SP@ HERE - . ;
10
11 : PAGE 0 0 GOTOXY CLS ;
12
13 -6392 FENCE !
14
15 R->BASE
```

When copying, omit for the time being the FENCE line! See below...

Thats my WELCOME screen.

180 DISK\_HI ! on this screen sets the system up for two single sided drives

180 DISK\_SIZE ! on this screen sets up for one or more double sided drives

360 DISK\_HI ! on this screen sets up for two double sided drives.

The EXCLAMATION MARK (!) is important here!

DISK\_HI is the highest numbered screen, at the rate of 90 per single side.

DISK\_SIZE is the number of screens per DISK.

continued.....>



FREE will allow you to obtain the free memory at any time just by keying FREE.

PAGE will at any time clear the screen and home the cursor.

Once you have set up your screen three, as above or adjusted for your disk system, FLUSH it for the time being.....

Press FCTN 9 (BACK) then enter FLUSH [enter]

One more step:

In Forth you can FORGET the basic Forth words, such as +... and in the course of doing so, FORGET everything above it in memory, which can be quite drastic. FORTH allows you to place a FENCE which prevents you FORGETting important things.

Let's see where the top word in the COMMAND STACK is:

Type in ' PAGE . [enter]

(single quote, space, PAGE, space, full stop, [enter] )

The number printed is where we want to put the FENCE, to again type 3 EDIT

Add to the end of the screen (as above)

[number] FENCE !

and FLUSH the screen to disk as before.

Now... cover the write protect tab!!! And your disk is all ready.

To PRINT a single screen, type in:

SWCH 3 LIST UNSWCH to list screen 3 in this case.

To see how quickly FORTH now loads, type in COLD again.

To see a list of words available to you, type in VLIST [enter]

Hold SPACE to halt the scrolling!

[Thanks to Craig Miller for a lot of the above details]

Now, entering in FORTH:

SPACES are IMPORTANT. If I show a space, put one in!

First, mark the beginning of the memory area you are about to use:

ENTER:

: ME ; (that is: COLON,SPACE,ME,SPACE,SEMI COLON, then ENTER)

The word ok will appear, then the cursor will come back.

This is quite important. Every word you define in FORTH is entered into memory. If you use FORGET ME, the definition of ME is removed AND EVERY WORD DEFINED AFTERWARDS. This can avoid annoying warning messages!

Putting that blank definition of ME at the start of your work is a useful way of scrubbing definitions.

Now that ME is in the library, lets experiment with multiple definitions!:

ENTER : PRINT 4 . ; (Type in the COLON)

then ENTER : PRINT 5 . ;

The computer obtained a warning message from the disk: note it and continue:

ENTER : PRINT 6 . ;

this time the warning message is brought from memory!

Now what does the word PRINT do? Let's try it: ENTER PRINT

Hmmm. It has the LAST definition we put in!

ENTER FORGET PRINT

ENTER PRINT now it has the 2nd to last definition!

You are warned if you redefine a word, but the new definition is the one that will be used. If you FORGET the word, the previous definition now becomes current, and so on. You can have a full stack of definitions of PRINT, and then delete them backwards!

You can even redefine basic FORTH words such as +, so be very careful! To check to see if a word is already in FORTH, key in:

' WORD .

and if an address is printed, the word is in there already. continued.....>

Redefining words is considered bad practice!!!!

Using ME as the first defined word, (we could have used AUDREY or anything else), we can still use ME to place another marker on the stack, and delete first the second section with FORGET ME, then delete the first section with another FORGET ME!

FORTH is a useful language, as you are able to experiment with the stacks without having to actually enter and run a program. It is very like LOGO in this respect.

These simple tests have been in 'direct mode', we have not stored a program yet, merely tried defining some simple words. That is the strength of FORTH: If a command you want does not exist, you can define it in terms of existing commands, and build up your own powerful vocabulary.

If you load -64SUPPORT, ensure that -TEXT is loaded before you use it (-VDPMODES includes -TEXT). If -TEXT is not resident, you will not be able to leave the 64 column split screen. To change from the 64 column screen to normal, just back out with FCTN 9, then at the bottom of the screen enter the word TEXT [enter].

Thats more than enough for now, no room for sample screens sorry. Let Clive and me know if you've enjoyed this forthright ramble!! Forth is Fun.

Two items:

1. A volunteer to operate/coordinate a TI Forth Interest Group???? Please?? Then we won't take over TI\*MES!!!
2. New address for the FORTH INTEREST GROUP UK, not as quoted in issue 7:  
write: D J Neale Esq., 58 Woodland Way, MORDEN, Surrey  
Sub is seven pounds a year for 6 issues of FORTHWRITE p.a.

*Stephen Shaw*

#### SPECIAL MEMBERS OFFERS

from Stephen Shaw (on behalf of TI\*MES).  
10 Alstone Road STOCKPORT Cheshire SK4 5AH

#### TI FORTH:

DISK A: TI FORTH TO LOAD FROM EDITOR/ASSEMBLER  
DISK B: TI FORTH TO LOAD FROM EXTENDED BASIC  
DISKS C & D: TI FORTH SOURCE CODE (2 disks)

DISK E: MULTIPLAN REVISION: Faster and with auto-repeat

DISK F: TI WRITER REVISION: True lower case on screen with slashed zeroes.  
No page feed at start of FORMATTER.

NB: Please indicate EXACT printer name you use with FORMATTER and that name will be inserted as a default (eg PIO.LF or PIO.CR etc)

DISK G: NAVARONE SUPER BUGGER

We are seeking more public domain programs for you... watch next issue.

#### PRICES (UK ONLY):

One disk: £4.00 Two disks: £7.00, each extra disk £3.00

If you supply the disk(s): One recorded for £2, each extra £1.

These are public domain programs and you may copy them for your friends.

TI FORTH MANUAL, Xeroxed and spiral bound with laminated covers: £34.00

OR ask for special LOAN COPY: £34 deposit, £32 refunded if loan copy returned in seven days. (Only applies if you specify LOAN when ordering).

PRK BOOKLET: Details of extra calls available when PRK or STATS modules are inserted. CALL A, D, G etc. £1.50

\* OVERSEAS: Send details of your requirements with an International Reply Coupon (sold at your post office) for a quotation.

## ADVANCE TO FORTH

Before I start on the Forth, I would like to pass a few comments on other topics related to the Texas computer.

First, I was pleasantly surprised by the response to my article in the last issue. The article was written specifically to stimulate discussion about our computer, and to get people to communicate with each other instead of using the computer in isolation. Unless we ourselves can keep the interest going, the Texas will gradually die, as we cannot expect the manufacturers of software and peripherals to carry on indefinitely in a market that is not going to expand. We should thank Clive and Audrey for taking the plunge and starting the User Group, for at least we now have a common point of reference as regards the Texas computer. I think we should all now make a greater effort to participate in our chosen activity instead of just being passive observers. In this way we can be sure that even if external interest wanes, we shall be able to carry on finding new things to do with our computer, due to the ideas being floated round amongst ourselves. If my articles do nothing else but stimulate discussion, then at least they will have done something.

I received several letters which commented on the points that I made, and several that asked for help. The letters came from far and wide, some from as far away as Sweden, which shows that interest is widespread and far from dead. I still have several replies to make, notably those to Sweden, and to John, whose letter arrived before my copy of TI+MES! (No offence meant C&A).

Some people wrote asking for technical advice, and I do hope that the replies that they received were satisfactory. Should anyone else feel that I can be of assistance, please get in touch by all means. The advice is free, but please enclose a SAE for your reply, unless of course you use the telephone. Please bear in mind that I cannot give advice on the Basics (extended or otherwise), as I have hardly used them. From the efforts that I have seen in the user group newsletters and the magazines, there are people much more skilled than myself in Basic. I have every admiration for those people who have attempted to unravel the inner workings of Basic with very few software tools.

I can give advice of sorts on the hardware side of the computer, as I have a full set of circuit diagrams for the computer, and its peripheral cards, along with technical data on the chips used in the computer. These are now available from Texas in Bedford I am told, whereas mine had to be imported from the States.

We all know by now that Forth is available to run under Editor-Assembler, but it is now available to run under Extended Basic and Mini-memory. At the risk of placing Peter Brooks under even more stress, he has the relevant items. I have tried the ExBas version, and it appears to run just the same way as E-A Forth, but it is much easier to start up. Just place the disk in the drive, start ExBas and Forth will auto-boot.

It seems, from reading Howard Greenberg's article in the THUC newsletter, that we are not losing all backing from hardware manufacturers, as he holds stocks of several Myarc peripherals, the most interesting being the disk controller card, which can handle up to four double-sided, double-density drives, presumably 80 tracks per side also. This would give a storage capacity, on-line of 2.88 megabytes. This should be enough even for Pascal users.

Just in case anyone should feel that our cpu, the 9900, is a somewhat inferior device due to the relatively slow operating speed of our computer, and the lack of press coverage, it might come as a surprise to find that the 9900 is used quite extensively in industrial process control, where fast reaction to real-time situations is necessary. Last week I was on a course, courtesy of my employers, on multiplexing equipment for data transmission over the telephone network. The equipment we were shown bore very little resemblance to the single-user modems that are at present being marketed for several hundred pounds; in fact it was like comparing a wooden club to a machine-gun. At the heart of this complex equipment was the good old 9900! Each circuit board, of which there could be up to sixty-eight, was controlled by a 9900 cpu, along with lots of peripheral chips from TI of just the same type as those used in our own home computer, and one board did in fact contain the new lightning-fast 99000 cpu from TI. All this goes to show that our home computer has some very speedy and sophisticated relatives.

Now for an apology. Due to some error in the transmission of my Diskformat program to the magazine pages, I have been told that there is a bug! The program will not allow a repeat format without restarting the program. This is due to an error in the main calling routine, named "DISKFORMAT". The last line of this routine should be:

```
FORMAT BASIC FINISH IF NAME 0A 20 FILL MYSELF ENDIF ;
```

The FORGET DISKF cannot now be used, as the system says that the routine DISKF is in protected memory. I have not yet been able to find a way round this, and so you will have to remember to FORGET DISKF yourself if you wish to conserve memory.

The routine "ZX" should of course be deleted, as this is a left-over from the development period.

In one of the letters I received, I was roundly criticised for the terrible style of the program. This was quite justified in that it was badly laid out, and not at all easy to follow, but there was a reason. This was that the program was meant to fit in as few screens as possible, so that users of one disk drive, who have few spare screens to use on a standard Forth disk, would have to delete as few screens as possible in order to fit this program on. It would be possible to load the Editor and then change the system disk for a disk onto which the new screens could be "flushed" after typing in with the editor. I should have thought of this earlier, but using two drives all the time, one tends to forget the obvious methods used by single drive users. Future programs will be laid out in proper style.

As a late addition to this article, I have typed out the DISKFORMAT program again, and this time included comments to explain the various sections. I don't pretend that this is the best Forth style that can be achieved, but it may help those that are even less experienced than myself in Forth programming.

Now onto the Forth subject for this issue, accessing disk sectors via standard Forth commands. Disk sector access is particularly easy with Forth, as the system has been designed to use virtual memory, which is jargon for using disk memory as an extension of the computer's memory. In the case of microcomputers with floppy disks, it is a very slow method of working, but used in mini and mainframe computers where the hard disks move at an incredible speed (at an incredible price), and where there are separate processors to deal with the disk input/output and memory management, virtual memory is a very good way of extending computer memory on a relatively cheap basis. But to return to Forth; because of the need for frequent disk accesses, the kernel of the system has routines built into it which give easy disk access to the end user (you and me).

Simply by placing the number of the block (screen) on the stack, and giving the command BLOCK, that block of four sectors is fetched in from the disk and placed in one of Forth's five disk buffers. To enable us to locate the buffer used, the start address of the buffer is left on the stack. This is the start of the data in the block, but there is a word (2 bytes) immediately before that address which gives the number of that block. This word also contains a flag (an indicator) which tells the system if the block has been altered in any way, thus requiring re-writing to the disk, should that buffer be required for another block at a later time. Both the block number and the update flag (because it is altered by the UPDATE command) can be altered by you, so that you can cause the block to be copied to another location on the disk by FLUSHing the block. The most significant bit should be set to 1 if you wish the next FLUSH operation to write the block out to disk. The least significant byte contains the block number and should be altered to the desired block number. Great care should be taken when doing this, as it is easy to overwrite other blocks which you may want to keep. At this low level of operation, there are no safeguards built into the system to prevent overwriting blocks of data. (Come to think of it, Basic provides no safeguards either).

Perhaps I should say at this point that a disk sector on a TI-formatted disk is 256 bytes long, and as there are four sectors to a block, this gives 1024 bytes per block, which is the standard size of Forth block or screen. This is the smallest unit of disk access available to Forth, without using assembly language routines, but as we can manipulate data in these 1k blocks quite easily, we do not need to resort to assembly language.

An example of block data manipulation is shown in my disk formatting program in the section DISKSETUP, where the 'header' for a Basic disk is formed in the disk buffer before being FLUSHed to the disk.

The contents of a disk block can be inspected by using the editor, but this can deal only with ascii characters, non-printable characters showing on the screen as rather strange symbols, or garbage. To inspect the non-printable characters, we must use the BLOCK command and display the contents of the buffer by using a command such as the DUMP command. This will display any area of memory, in hexadecimal form, with the ascii equivalents (if any) alongside, thus giving an excellent method of seeing any printable messages built into machine code.

When you have decided what parts of the block you wish to alter, the alterations must be made by storing the new value using "!" , which is the Forth equivalent of Poke. The best number base to use is hexadecimal, as this is the base used by DUMP. As can be seen by inspection of DISKSETUP, values which are non-printable can be entered, thus allowing the repair of damaged disk sectors; damaged from a data point of view, not physically damaged. I have in fact used this method to repair several disks which had suddenly been reported as 'not initialised'. On inspection with Forth and the DUMP command it was apparent that the disk system had spread garbage throughout the directory sectors. It took about half-an-hour to rectify the fault, but this was better than losing a disk of data. As long as the sector formatting information remains intact so that the disk controller hardware can make sense of the disk, then Forth does not require that there be any sort of disk information on the first few sectors of the disk, before it is accessible to block reading.

This method of disk access cuts out the need to use a special device such as the Navarone Disk Fixer, excellent though that product is, and this should save some people some money.

While writing this article, it seems that writing a special program to alter the contents of a sector would perhaps not be worthwhile, as the Forth commands are so simple to use on a manual basis that to alter bytes would not be difficult. Perhaps it might be nice to have the sectors scrolling up and down the screen at will, without having to use DUMP all the time. This could possibly be achieved by using the VMBW command to write the portion of the block currently pointed at to the screen with the pointer incremented by depressions of the space bar, thus altering the section of the block currently in the 'window'. There would have to be some translation of the block contents into displayable characters, as in the DUMP command, and perhaps another buffer to store these characters in before copying to the screen. I'll throw it open to other Forth enthusiasts—is it worth doing? How should it be implemented? Could we collaborate on a project like this, or are there any other software projects which would be more interesting that a group of people could join in and share the load? Please let me know what you think.

Some paragraphs ago I mentioned the update flag and the block number byte. These can be used to help in a disk copy program. While people like myself can use the SMOVE command to copy disks easily from one drive to another, for single drive users it is a more difficult task, involving many changes of disk with the Disk Manager. I thought that a program which would cut down the number of changes to 3 would be ideal, and would have been relatively easy to do, but the system would not allow me to use as much VDP ram as I wanted to use, so the program will now take 4 changes of disk. This is achieved by using expansion and VDP ram to store the information, whilst the disk is changed from the master to the target disk.

As the program is eight screens long (nicely set out) I don't think that C&A would accept that on top of the pages already written, so it will have to wait for next time. I can tell you that it will perform the same function as SMOVE, but 15 seconds slower due to storing the buffers in memory instead of writing them straight out to disk. In comparison with the BACKUP facility of Disk Manager, my program is 30 seconds faster. The times are:

|            |                      |
|------------|----------------------|
| SMOVE      | 3 minutes            |
| MY PROGRAM | 3 minutes 15 seconds |
| BACKUP     | 3 minutes 45 seconds |



FORMAT DISK routines in a more understandable format.

The -SYNONYMS option must be loaded before loading this program from disk.

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>: PAGE CLS @ @ GOTOXY ; @ VARIABLE NAME @ ALLOT NAME 10 32 FILL  @ VARIABLE DRIVES @ VARIABLE TRACKS @ VARIABLE SIDES @ VARIABLE THIS-ONE  @ VARIABLE BASICFLAG  : INPUT QUERY 32 WORD HERE NUMBER DROP ;  : NAME? CR ." DISK NAME ? " QUERY 32 WORD HERE COUNT NAME SWAP CMOVE ;  : DRIVES? CR ." NUMBER OF DRIVES ? " INPUT DRIVES ! ; : TRACKS? CR ." NUMBER OF TRACKS ? " INPUT TRACKS ! ; : SIDES? CR ." NUMBER OF SIDES ? " INPUT SIDES ! ; : WHICH? CR ." FORMAT WHICH DRIVE ? " INPUT THIS-ONE ! ;  --&gt;  : READY? CR CR ." PLACE YOUR BLANK DISK IN DRIVE " THIS-ONE @ . CR ." AND PRESS ANY KEY " DROP ;  : MODDISK_SIZE 225 TRACKS @ * 100 /  : MODDISK_HI DISK_SIZE @ DRIVES @ * DISK_HI ! ;  : NEWTRACKS TRACKS @ 13706 ! ;  : NEWSIDES SIDES @ 33616 ! ;  : FORMAT THIS-ONE @ FORMAT-DISK ;  : TITLE ." FORTH DISK FORMATTER"  : Y/N KEY DUP 89 = IF DROP 1 ELSE 78 = IF @ ELSE MYSELF ENDIF ENDIF ;</pre> | <p>This command clears the screen and homes the cursor to the top left hand corner.</p> <p>This sequence sets up a variable to contain the disk name if used with Basic. ALLOT reserves space for the name, and the NAME 10 32 FILL sequence fills the name with blanks.</p> <p>These lines set up variables to contain various values that I found easier to use than trying to juggle with the stack. I personally feel that it can be counter-productive to try to use the stack all the time.</p> <p>The name THIS-ONE seemed to fit nicely into the program flow in the calling routine.</p> <p>BASICFLAG contains the indicator which says whether or not to write a Basic header.</p> <p>A word to allow input of numbers as characters. NUMBER then converts the string to a double number, hence the need for DROP, which removes the most significant 16 bits of the number, converting it back to a single-length number.</p> <p>Allows entry of a character string for the disk name ended with a space or carriage return. The string is then moved by COUNT NAME CMOVE into the array variable reserved for the disk name.</p> <p>These lines prompt for and input the values required by the program.</p> <p>This little character means "carry on loading with the next screen".</p> <p>Allows you to change your disk, and waits for you to press a key. THIS-ONE @ . fetches the disk number that you have input and displays for checking by you.</p> <p>The value of the key pressed is not needed and so is dropped off the stack.</p> <p>This is a calculation for modifying the disk size. Disk size is 2.25 times the sides number of tracks, times the number of sides, in sectors. Forth does not easily do floating point maths, hence the fiddle.</p> <p>Similar to above, except that it is for DISK_HI .</p> <p>The memory location 13706 decimal was found by much hard work in the assembler support routines section of Forth.</p> <p>Storing the number of sides in CPU scratchpad ram. Again found by diligent searching and help from a friend.</p> <p>Using the standard Forth commands wherever possible. (Why do it the hard way?)</p> <p>Title prompt. Insert spaces in it to centre it on the screen.</p> <p>A series of checks to make sure that either a Y or an N is pressed. If not, MYSELF calls the Y/N routine again.</p> |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre> : BASIC? CR ." WILL THIS DISK BE USED WITH BASIC ? " Y/N DUP BASICFLAG ! IF NAME? ENDIF ;  : FINISH CR CR ." FORMATTING FINISHED " CR ." FORMAT ANOTHER ? " Y/N  : BASIC BASICFLAG @ IF DISKSETUP ENDIF ;  HEX  : DISK-SETUP THIS-ONE @ DISK_SIZE @ * BUFFER  DUP NAME SWAP @A CMOVE DUP @A + DISK_SIZE @ 4 * SWAP !  DUP @C + @944 SWAP ! DUP @E + 534B SWAP ! DUP 1@ + 2000 SWAP ! DUP 12 + 26 DISK_SIZE @ 2 / + @ FILL  DUP 38 + @300 SWAP ! DUP DISK_SIZE @ 2 / 38 + DUP ROT + SWAP 100 SWAP - FF FILL  DUP 100 + 100 @ FILL 200 + 200 E5 FILL  UPDATE  FLUSH </pre> | <pre> Asks if Basic header is to be written onto the disk.If the answer is yes,then a 1 is written to BASICFLAG,else if the answer is no,a 0 is written to BASICFLAG. Before storage of the flag,the flag is duplicated and used to test if the NAME? entry routine will be called.  Gives the chance to format another disk.  Checks to see if you wanted a Basic header,and writes the the header if you said yes. I have moved this from the third screen because it keeps all the routines together.  Change the base.Makes it easier to work the next section out.  Reserves a buffer in memory for the data the program is about to put on the first 4 disk sectors.The correct block number for the drive that you are using worked out automatically. The disk name is written to the buffer. Number of sectors on the disk written to the buffer.  Writing various pieces of initialisation data to the buffer.  Filling in the bit-map field to indicate to the disk routines how many sectors have been used.  Filling in the remainder of the 3rd and 4th sectors with the data that the disk routines expect to see.  Mark the buffer as updated so that it will be written to disk.  Physically write the buffer to disk. </pre> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

The routines that do the work have now been written.All that remains is to write a routine that will call the 'working' routines.This routine will be called DISKFORMAT.

|                                                                                                                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre> : DISKFORMAT PAGE TITLE CR CR TRACKS? SIDES?  DRIVES? WHICH? BASIC? READY?  MODDISK_SIZE MODDISK_HI NEWTRACKS  NEWSIDES FORMAT </pre> | <pre> Clear the screen. Print the title. Create a few empty lines for neatness. Input number of tracks. Input number of sides per disk.  Input number of drives on the system. Input the drive number to be used for formatting. Are we to use it for Basic? Is the disk in the correct drive? All the user inputs are now finished.We are now on automatic pilot. Tough luck if you leave your system disk in the wrong drive. ALWAYS TAKE IT OUT WHEN FORMATTING !!!  Modify DISK_SIZE. Modify DISK_HI. Store the number of tracks in the right place so that the standard Forth disk formatter routine can find it. As above for the number of sides. Call the Forth disk formatting routine. </pre> |
|---------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

```

FORMAT          Call the Forth disk formatting routine.
BASIC           Write the disk header if necessary.
FINISH         Give chance to do another. Leaves a flag on the stack to indicate
              the answer you gave.
IF             Test the flag.
  NAME @A 20 FILL You do want to format another so clear out the name array ready
              for the next formatting.
  MYSELF       Now call DISKFORMAT again.
ENDIF
;
DECIMAL        Change the base back to decimal.

```

I hope that the above description has helped to explain what was probably very confusing in the last issue. You will never get a listing like this in Forth, as the screen is only sixty-four characters wide, and this was written in Wordstar with sideways scrolling on an 80 column screen to give 132 columns. It is very difficult to write a Forth program and get a decent amount of comments on the same screen. It becomes imperative that you choose the names of your routines to reflect the operations being carried out, so that you will know what the routine does when you come back to it six months later.

I'll sign off now, but I hope to see many people at the show in Brighton.

Phillip Marsden.

6, Kennerleigh Grove,  
Leeds 15.  
LS15 8NQ.

Please excuse the slightly blobby printing. I have tried re-inking my ribbon, and this is the result.

PLEASE REMEMBER IF YOU ARE WRITING TO ANY OTHER MEMBER OR A CONTRIBUTOR TO THE MAGAZINE TO INCLUDE A STAMPED ADDRESSED ENVELOPE FOR YOUR REPLY.

#### AREA CONTACTS.

Henry Clark, 60 St Pauls Road, New England, PETERBOROUGH, Cambs. 0733 42642  
 Harry Fridmore, 17 Jerrards Close, HONITON, Devon. 0404  
 John Carter, 16 Sherwood Ave, NORTHAMPTON. 0604 842760.  
 Simon Pryce, 48 Mount Street, SHREWSBURY, Salop. 0743 67799. Interested in amateur radio  
 John Bingham, Rygghagen 7B, 4070 Randaberg, Stavanger, NORWAY. 04-599228  
 FORTH INTEREST. Stanley Dixon, 28 Grange Park Road, LEEDS, LS8 3BB.  
 PASCAL INTEREST. Stanley Dixon, 28 Grange Park Road, LEEDS, LS8 3BB.  
 Graham Hilton, 8 sandwich Close, Saint Ives, CAMBRIDGE. 0480 65228.

## HELPFUL HINTS

### CALL LOADS.....FROM ORANGE COUNTY USERS GROUP

For those of you with Extended BASIC and memory expansion, here are a couple of addresses that might prove to be useful.

CALL LOAD(-32116,4)

Allows you to enter TI BASIC from Extended BASIC.

CALL LOAD(-31748,X) {X=0 to 255}

Changes the speed of the cursor and sound chip.

CALL LOAD(-31962,255)

Resets systems, and boots "load" program.

CALL LOAD(-31888,55,215)

Turns off disk drives and frees the buffer space.

CALL LOAD(-31888,55,215)

Turns drives back on and opens buffer.

Extended

Tutor



with  
Tony  
McGovern.

A bug crept into the comment in the first tutorial on CALL-ing nonexistent subprograms. TI Basic, when given a name in a CALL for a nonexistent subprogram, completes the prescan and then crashes with BAD NAME when it comes to the line. TI Basic will load and prescan XB programs with unrecognized CALLs. If the program is such that it is otherwise acceptable to TI Basic, then it can be run under TI Basic if it never encounters the lines with the 'bad names'. I have been using for some time this very same property in an auto LOAD program on all our Basic and XB program disks which also will catalog disks from Basic.

Tony McGovern

Funnelweb Software

(C)

.....

SUBPROGRAM PARAMETER LISTS

In the last chapter we saw how subprograms fitted into the overall workings of Extended Basic. In this chapter we are going to go into the details of writing subprograms. Most of the fiddly detail here concerns the construction of the parameter lists attached to CALL and SUB statements, and some of the little traps you can fall into.

Any information can be transmitted from the CALLing program to the CALLED subprogram via the parameter list, and anything not transmitted this way remains private for each program, with the exception of the DATA pool which is equally accessible to all. If something is mentioned in the parameter list then it is a two-way channel unless special precautions, provided for in XB, are taken. In this case the CALLing program can inform the subprogram of the value of a variable, but not allow the CALLED program to change the value of the variable as it exists in the CALLing program. Arrays however, numeric or string, can't be protected

from the follies of subprograms once their existence has been made known to the subprogram through the parameter list.

Let's for starters take a very simple but useful example, where a program needs to invoke a delay at various points. Now some BASICs (and TI LOGO) have a built-in function called WAIT. XB doesn't have this command so you have to program it. It can be done by a couple of CALL SOUNDS or with a FOR-NEXT loop. Let's use an empty loop to generate the delay, about 4 millisec. each time around the loop, and place the loop in a subprogram.

```
230 CALL DELAY(200)
```

```
670 CALL DELAY(200/D)
```

```
990 CALL DELAY(T)
```

```
3000 SUB DELAY(A):: FOR I=1 TO A :: NEXT I ::SUBEND
```

This is easier to follow when editing your program then using a GOSUB, and you would need to enter the subroutine in every subprogram since GOSUBbing or GOTOing out of a subprogram is verboten. Also it's less messy than writing the delay loop every time. The example shows several different CALLs to DELAY. The first supplies a number, and when DELAY is CALLED, the corresponding variable in the SUB list, A, is set to 200. This is a particular example of the kind of CALL from line 670 where the expression 200/D is first evaluated before being passed to DELAY to be assigned to A. Variable D might for instance represent the level of difficulty in a game. The CALL from line 990 invokes a numeric variable T, and A in the subprogram is set to the value of T in the CALLing program at the time when the CALL is executed.

Nothing untoward happens to T in this example, as the DELAY subprogram does nothing to change A. Now it may not matter in this instance if T did not retain its value after the subprogram CALL. Suppose instead the delay was to be called out in seconds. Then a subprogram on the same lines DELAYSEC might go

```
230 CALL DELAYSEC(2)
```

```
990 CALL DELAYSEC(T)
```

```
4000 SUB DELAYSEC(A):: A=A*250 4010 FOR I= 1 TO A :: NEXT I :: SUBEND
```

Now after DELAYSEC has been executed with the CALL from

990, T will have value 250 times its value before the CALL. This won't be a bother if you don't use T again for its previous value. If the CALLing program specifies a numeric constant as in line 230, or a numeric expression, the change in A in the subprogram has no effect on the main program. Suppose you can't tolerate T being changed in line 990 (and this kind of thing can be a source of program bugs). You will find that XB allows for forcing T to be treated as though it were an expression, thus isolating T from alteration by the subprogram, if T is enclosed in brackets in the CALL (not SUB) list. Suppose DELAYSEC is also called from line

```
970 CALL DELAYSEC((T))
```

If this CALL in line 970 is followed by the CALL from line 990, T not having been altered in the meanwhile, the same delay will be obtained, but if the order of CALLs were reversed the second delay would be 250 times the first. In the language of XB this is known as "passing by value" as distinct from "passing by reference". This can only be done for single variables or particular array elements, which behave like simple variables in CALL lists. Whole arrays cannot be passed by value, but only by reference. Expressions and constants can only be passed by value, and its hard to see what else could be done with them. In the example as written, a different variable name was used in the SUB, but if you remember the little experiment in the last chapter you'll see that it wouldn't make any difference if T had been used in the SUB list instead of A.

Now let's complicate things a little by flashing up a message on the bottom line of the screen during the delay interval.

```
200 CALL MESSAGE(300," YOUR TURN NOW")
```

```
270 CALL MESSAGE(T,A$)
```

```
3000 SUB MESSAGE(A,A$):: DISPLAY AT(24,1):A$ 3010 FOR I=1 TO A :: NEXT I :: DISPLAY AT(24,1):"" 3020 SUBEND
```

The SUB parameter list now contains a numeric variable and a string variable in that order. Any CALL to this subprogram must supply a numeric value or numeric variable reference, and a string value or string variable reference, in





precisely the same order as they occur in the SUB list. In the little program segment above, line 200 passes constants by value and line 270 passes variable references. There is no reason why one cannot be by value and one by reference if so desired.

This process can be extended to any number of entries in the parameter list, provided the corresponding entries in the SUB and CALL lists match up entry by entry, numeric for numeric, string for string. The XB manual does not say so explicitly, but it appears that there is no limit apart from the usual line length problems, on the number of entries in the list. This is the only apparent difference between the parameter list in XB subprograms and the argument lists for CALL LINK("xxxxxx", , ... ) to machine code routines in XB, and Minimemory and E/A Basics.

One little freedom associated with built-in subprograms is not available with user defined subprograms. Some built-ins, such as CALL SPRITE permit a variable number of items in the CALLing list. Parameter lists in user defined subprograms must match exactly the list established by the SUB list or an error "INCORRECT ARGUMENT LIST in ..." will be issued. To compensate for this inflexibility user defined CALLs allow whole arrays, numeric or string, to be passed to a subprogram. Complete arrays may be passed by reference only. Individual array elements may be used as if they were simple variables and may be protected from alteration by bracketing in the CALL list. An array is indicated in the parameter list by the presence of brackets around the array index positions. Only the presence of each index need be indicated as in A(). MATCH(,,) indicates a three-dimensional array MATCH previously dimensioned as such, explicitly or implicitly. Don't leave spaces in the list. If the subprogram needs to know the dimensions of the array these must be passed separately (or as predetermined elements of the array). TI Basics are weaker than some others in that they do not permit implicit operations on an array as a whole, a very annoying deficiency.

Arrays may be DIMensioned within subprograms. This will introduce a new array name to the program, and an array or variable name from the SUB parameter list can't be used or an error message will result. In the following code the main program passes, among other things, an array SC to subprogram BOARD (perhaps a scoreboard writing routine in a game).

```
100 DIM SC(2,5) :: ....
450 CALL BOARD(P,A$( ),SC( ))
4000 SUB BOARD(P,A$( ),S( ))::
DIM AY(5):: ....
.... :: CALL REF(P,AY( ),S( ))
4080 SUBEND
5000 SUB REF(V,A( ),B( ))::
.... :: SUBEND
```

BOARD generates internally an array AY() which is passed to another subprogram REF (maybe this resolves ties) along with SC( ), which BOARD knows as S( ), and REF in its turn as B( ). There is however no way that the main program or any subprogram whose chain of CALLs doesn't come from BOARD can know about AY. Equally well REF, if CALLED only from BOARD and not directly from the main program, could not have found out about P or SC( ) except through BOARD, even if BOARD did nothing with them except mention them in its parameter list.

By following this line of reasoning you can see that there is no way for a subprogram whose chain of CALLs does not come through BOARD to know about array AY(). The only way around this is for AY() to be DIMensioned in the main program (even if this is its only appearance there) and the message passed down all necessary CALL-SUB chains.

This idea of DIMensioning an array only within a subprogram is particularly useful if the array is to READ its values from DATA statements and to be used in the subprogram. This could be done again from any other subprogram needing the same data, without having to pass its name up and down CALL-SUB chains. Remember that DATA statements act as a common pool from which all subprograms can READ. If the array values are the results of computations then these values must be passed through the CALL parameter lists.

For completeness note that , although the XB manual has nothing to say about it, IMAGE statements for formatting PRINT output are accessible from any part of a program in the same way as DATA

statements and not confined to the subprograms in which they occur as are DEF entries.

It is not necessary to have any parameters in the list at all. Subprograms used this way can be very helpful in breaking up a long program into more manageable hunks for ease of editing. We shall also see in later chapters that there can be other benefits as well.

One more XB statement for subprograms remains, the SUBEXIT. This is not strictly necessary as it is always possible to write SUBEND on a separate line and to GOTO that line if a condition calling for an abrupt exit is satisfied. Like a lot of the little luxuries of life however, it is very nice to have and makes programs much easier to read and edit. It does not replace SUBEND which is a signal to the XB pre-scan to mark the end of a subprogram. SUBEXIT merely provides a gracious and obvious exit from a subprogram (awkward in some Pascals for instance). The next chapter will demonstrate typical examples of its use.

#### IV. USEFUL SUBPROGRAM EXAMPLES

In the previous chapter we used as an example a DELAY subprogram which could, with a little refinement, be used to substitute for the WAIT command available in some other languages. You can extend this idea to build up for yourself a library of handy-dandy subprograms which you can use in programs to provide your own extension of the collection of subprograms that XB offers.

For our first example let's take one of the more frustrating things that TI did in choosing the set of built-in subprograms. If you have Minimemory or E/A you know that the keyscan routine, KSCAN, returns keyboard and joystick information simultaneously, while XB forces you to make separate subprogram CALLs, KEY and JOYST, to dig it out. Since these GPL routines are slow it is difficult to write a fast paced game in XB that treats keyboard and joysticks on an equal footing as is done by many cartridge games. On the other hand in games where planning and not arcade reaction is of the essence there is no reason why the player(s) should be forced to make a once-and-for-all choice and not be able to use either at any stage of the game.

```

1 REM ++++++
2 REM +++++TITILE SCREEN+++++
3 REM +++HUG LIBRARY 4/84+++
4 REM +++BY MARK CHANCE+++
5 REM +++++ABOUT 2K LONG+++
6 REM FROM HOUSTON USERS GRO
UP
7 CALL CLEAR :: FOR A=65 TO
90 :: CALL CHARPAT(A,A$):: C
ALL CHAR(A,SEG$(A$,3,10)
&SEG$(A$,11,2)&SEG$(A$,13,6)
):: NEXT A
8 FOR A=48 TO 57 :: CALL CHA
RPAT(A,A$):: CALL CHAR(A,SEG
$(A$,3,4)&SEG$(A$,5,2)&S
EG$(A$,7,10)):: NEXT A
9 REM CC=SCREEN COLOR
10 CC=8
11 DIM R(15),Q(8):: CALL CHA
RPAT(45,ZZ$):: CALL CHAR(94,
ZZ$):: CALL CHAR(137,"0"
):: CALL CHAR(64,"3C4299A1A1
99423C"):: CALL CHAR(40,"0")
12 DATA 2,9,10,11,12,13,14
13 RESTORE 12 :: FOR A=1 TO
7 :: READ B :: Q(A)=B :: NEX
T A :: RESTORE 14 :: FOR
A=1 TO 14 :: READ B :: R(A)
=B :: NEXT A
14 DATA 7,4,2,12,13,14,16,5,
3,14,9,15,10,11
15 IF CC<>8 THEN GOSUB 30
16 DATA 40,96,104,112,120,12
8,136
17 FOR A=1 TO 14 :: CALL COL
OR(A,1,1):: NEXT A :: RESTOR
E 16 :: FOR A=1 TO 6 ::
READ B :: CALL CHAR(B+1,"0")
:: NEXT A
18 RESTORE 16 :: U=2 :: FOR
A=1 TO 7 :: READ B :: FOR C=
1 TO 2 :: CALL VCHAR(1,U
,B,3):: U=U+1 :: NEXT C :: F
OR C=3 TO 4 :: CALL VCHAR(1,
U,B+1,3):: U=U+1 :: NEXT
C
19 NEXT A :: CALL VCHAR(1,30
,40,3):: CALL VCHAR(1,31,40,
3)
20 RESTORE 16 :: U=2 :: FOR
A=1 TO 7 :: READ B :: FOR C=
1 TO 2 :: CALL VCHAR(19,
U,B,3):: U=U+1 :: NEXT C ::
FOR C=3 TO 4 :: CALL VCHAR(1
9,U,B+1,3):: U=U+1 :: NE
XT C
21 NEXT A :: CALL VCHAR(19,3
0,40,3):: CALL VCHAR(19,31,4
0,3)
22 FOR A=3 TO 8 :: CALL COLO
R(A,2,1):: NEXT A :: CALL CO
LOR(1,2,1)
23 RESTORE 16 :: FOR A=1 TO
7 :: READ B :: CALL CHAR(B,"
FFFFFFFFFFFFFFFF"):: NEX
T A
24 F=1 :: FOR A=1 TO 14 STEP
2 :: CALL COLOR(Q(F),R(A),R
(A+1)):: F=F+1 :: NEXT A
25 DISPLAY AT(10,7):"TI S H
U G AUST"
26 DISPLAY AT(17,2)BEEP:"REA
DY^HIT ANY KEY TO BEGIN" ::
DISPLAY AT(23,6):"@1983
MARK CHANCE"
27 CALL KEY(3,X,Y):: IF Y=0
THEN 27
28 CALL CLEAR :: CALL CHARSE
T
29 GOTO 32
30 FOR A=1 TO 14 :: IF R(A)=
CC THEN R(A)=8 :: CALL SCREE
N(CC)
31 NEXT A :: RETURN
32 REM +++++END OF HEADER++++

```



## EXTENDED BASIC TUTORIAL

The subprogrammers approach to this problem, once it realized that it can be done (and we have commercial XB games where the writers haven't) is to write the game using joysticks, but replacing JOYST by a user defined sub-program JOY which returns the same values as JOYST even when keys are used.

The first step in telling whether keys or joysticks are being used is to check the keys, and if none have been pressed then to check the joysticks. If a key has been pressed then its return, K, has to be processed so that the direction pads embedded in the keyboard split-scan return the corresponding JOYST value. A subprogram along the lines of the one used in TEX-BOUNCE\* does just this.

```

900 SUB JOY(PL,X,Y):: CALL
KEY(PL,K,ST):: IF ST=0
THEN CALL JOYST(PL,X,Y)::
SUBEXIT
910 X=4*((K=4 OR K=2 OR
K=15)-(K=6 OR K=3 OR K=14))
920 Y=4*((K=15 OR K=14 OR
K=0)-(K=4 OR K=5 OR K=6))
930 SUBEND

```

PL is the player (left or right joystick or side of the split keyboard) number and is unaltered by the procedure. The simple-minded approach for converting K to (X,Y) values by using the XB logic operators (one of the more annoying omissions from console Basic) seems to work as well as any. The subprogram as written checks the keys first but balances this out by putting the processing load on the key return.

This is as good a time as any to sharpen your own skills by working out alternative versions of this procedure, and also by writing one for mocking up a substitute CALL KEY routine to return direction pad values even if a joystick is used.



**More  
next  
ISSUE**

FROM ED YORK  
CIN-DAY  
EXTENDED BASIC QUIRKS

Many of you who have Extended Basic may not know that you can load a program from cassette and then execute or RUN it automatically by typing in RUN "CS1", or for those with disk...RUN"DSK1.nnnnnnnnnn" n=Filename.

The first one being an undocumented command. Now, doesn't that kill two birds with one stone?

Also many of you have been trying to make the computer speak phrases using Extended Basic and the Speech Synthesizer but are disappointed with the results!

Well again it is not documented that you need the # symbol before and after phrases that are listed (see the List Of Speech Words located in Appendix L of the Extended Basic Manual).

Phrases such as "WHAT WAS THAT", "READY TO START" and "THAT IS RIGHT" must be entered as CALL SAY("#WHAT WAS THAT#"), CALL SAY("#READY TO START#") and CALL SAY("#THAT IS RIGHT#") in order to hear the computer speak them correctly.

### DEBUGGING HINTS FROM DAYTON

Write out a rough outline of the section of your program before you start and begin each important section with a major line number (I.E. 1000, 2000, 3000, etc.), then you won't have much trouble finding certain types of bugs since you will then know in which section they occurred.

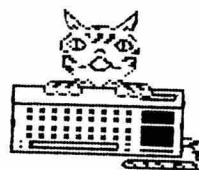
All line numbers take two bytes of memory in a program regardless of their length (I.E. 10, 1000, 32555 all take the same space).

An array variable list may now be made at a break in a program in X-Basic using a multi-line loop (I.E. FOR R=1 TO 10 :: PRINT A\$(R); :: NEXT R). This will list A\$(1 to 10) versus take something like: PRINT A\$(1);A\$(2);...A\$(10).



## TIPS FROM TIGERCUB

Copyright JIM PETERSON.  
 TIGERCUB SOFTWARE  
 156 Collingwood Ave.,  
 Columbus OH 43213



My new catalog #5 is now available for \$1.00, which is deductible from your first order. It contains over 130 programs in Basic and Extended Basic at only \$3.00 each (plus \$1.50 per order for cassette, packing and postage, or \$3.00 for diskette, PP&M).

The entire contents of Tips from the Tigercub Nos. 1 through 14, with more added, are now available as a full disk of 50 programs, routines and files for only \$15.00 postpaid.

Nuts & Bolts is a diskful of 100 (that's right, 100!) XBasic utility subprograms in MERGE format, ready for you to merge into your own programs. Contents include 13 type fonts, 14 text display routines, 12 sorts and shuffles, 9 data saving and reading routines, 9 wipes, 8 pauses, 6 music, 2 protection, etc., etc., all for just \$19.95 postpaid!

My 28-Column Converter, published in last issue of TI\*MES has a bug which causes a line to disappear if the wrap-around causes it to begin with a period and you are using the formatter option. Here is the fix -

Change line 300 to read: 300  
 FOR W=1 TO 5 :: READ CH\$,R\$

Change line 280 to read:

280 DATA @, (, &, ^, ~, \*, !, ., \

In other words, your DATA items will be the "at" sign above the 2, the left brace on the front of the F key, the ampersand on the 7 key, the right brace on the front of the G, the carat sign above the 6, the tilde on the front of the W, the asterisk above the 8, the whatsit? on the front of the A, the period, and the backslash on the front of the Z.

A couple of other changes will automatically turn off the automatic fill and adjust, and turn it back on. At the end of line 180, add :: PRINT #2:".NF" and change line 270 to NEXT J :: PRINT #2:".FI;AD;"

:: CLOSE #2 :: CLOSE #1 :: END

Now, as long as the text strings in your program don't contain those oddball characters, all should be well. However, the program has one more bug which is common to all 28-column converter programs, and for which I can find no really good fix. If a program line is exactly 80 characters long, the next program line will follow immediately after it instead of starting on the next line. So, load the file in the Editor mode and scan it before you print it. If any of you whiz kids (or whiz grandpas) can figure out a way to program around that problem, please let me know!

Some time ago I set up a challenge for someone to write a 1-line XBasic program which would take only 70 seconds to scramble the numbers from 1 to 255 into a completely random sequence without duplication. Richard Mitchell, the editor of Super 99 Monthly, came up with an program which is shorter than mine and runs about 10 seconds faster - but it sure does chew up a lot of memory!

```
1 DIM A(255),C(254):: RANDOM
  IZE :: CALL PEEK(-31808,B)::
  IF B=0 OR A(B)=B THEN 1 ELSE
  E C(D)=B :: A(B)=B :: D=D+1
  :: IF D=255 THEN END ELSE 1
```

And if you're not subscribing to Super 99 Monthly, you should be! It's only \$12 a year, and full of very useful programs, routines and tips. The address is Bytemaster Computer Services, 171 Mustang Street, Sulphur LA 70663.

Also be sure to get the National Ninety-Niner from the 99ers Users Group Association (3535 So. H St. #93, Bakersfield CA 93304), also only \$12 a year. Their roster of writers is beginning to look like the Who's Who of the TI world.

Alphabet Song with routines for the speech synthesizer. If you can type the alphabet without a mistake, you get an encore.

```

100 CALL CLEAR
110 PRINT "          ALPHABET S
ONG"
120 FOR J=1 TO 20
130 PRINT
140 NEXT J
150 PRINT "          by Ji
m Peterson": "Wait, please"
:
160 OPEN #1:"SPEECH".OUTPUT
170 DIM T$(26,2)
180 DATA 12,12,4,4,1,1,4,7,7
,8,8,10,10,10,10,12,4,4,7,8,
8,10,4,8,8,10
190 FOR J=1 TO 26
200 READ X
210 T$(J,1)="//"&STR$(X)&" "
&STR$(X/10*32)
220 T$(J,2)=CHR$(J+64)
230 NEXT J
240 T$(23,2)="DOUBLE"&"!"&"!
"&"U"
250 CALL CLEAR
260 PRINT "READY - TYPE THE
ALPHABET"
270 T=0
280 K2=64
290 CALL KEY(3,K,ST)
300 IF (ST<1)+(K<65)+(K>90)T
HEN 290
310 IF K<>K2+1 THEN 330
320 T=T+1
330 PRINT #1:T$(K-64,1):T$(K
-64,2)
340 CALL HCHAR(12,17,K)
350 K2=K
360 IF K<>90 THEN 290
370 IF T=26 THEN 390
380 GOTO 270
390 FOR K=65 TO 90
400 CALL HCHAR(12,17,K)
410 PRINT #1:T$(K-64,1):T$(K
-64,2)
420 NEXT K
430 PRINT #1:T$(1,1):"NOW IV
E":T$(3,1):"SAID MY":T$(5,1)
:"A B":T$(3,1):"SEEZ"
440 PRINT #1:T$(8,1):"WONT Y
OU":T$(10,1):"COME AND":T$(1
2,1):"PLAY WITH":T$(1,1):"ME
"
450 GOTO 270

```

Terry Atkinson's routine to redefine the cursor has aroused some interest, so I fiddled around and came up with this version to change the cursor automatically to whatever character, normal or redefined, that you input.

## TIPS FROM TIGERCUB

```

100 !CURSOR CHANGER by Jim P
eterson
110 INPUT A$ :: A=ASC(A$)::
CALL CHARPAT(A,A$):: FOR J=1
TO 16 STEP 2 :: H#=SEG$(A$,
J,2):: CALL HEX_DEC(H$,D)::
T=T+1 :: H(T)=D :: NEXT J ::
120 CALL INIT :: CALL LOAD(8
196,63,248)
130 CALL LOAD(16376,67,85,82
,83,79,82,48,8)
140 CALL LOAD(12288,H(1),H(2
),H(3),H(4),H(5),H(6),H(7),H
(8))
150 CALL LOAD(12296,2,0,3,24
0,2,1,48,0,2,2,0,8,4,32,32,3
6,4,91)
160 CALL LINK("CURSOR")!THAN
KS TO TERRY ATKINSON
170 SUB HEX_DEC(H$,D):: N=1
:: DEC=0
180 FOR J=1 TO LEN(H$):: A#=
SEG$(H$,LEN(H$)-J+1,1):: IF
ASC(A#)>58 THEN HT=ASC(A#)-5
5 ELSE HT=VAL(A#)
190 DEC=DEC+N*HT :: N=N*16 :
: NEXT J
200 IF DEC<>32768 THEN D=DEC
ELSE D=-(65536-DEC)
210 SUBEND

```

And of course you can always color the cursor with CALL COLOR(0,5,11) or whatever colors you like.

Most folks don't seem to know, and some folks refuse to believe, that the Memory Expansion can't store strings. If you are one of the disbelievers, plug in your Memory Expansion and try this -

```

100 FOR J=1 TO 255 :: M#=M#&
CHR$(J):: NEXT J
110 DIM A$(100):: X=X+1 :: A
$(X)=M# :: PRINT X :: GOTO 1
10

```

Now RUN that. On my console, I get MEMORY FULL when X=43 although the SIZE command shows I have 24399 bytes of program space free (in the Expansion) - but only 204 bytes of free stack (in the console). Without the Memory Expansion I can get X up to 51, and in Basic to 53.

This can be a serious handicap if you are running a program which reads in a large number of strings from DATA statements, or generates strings while running.



Of course, when the Memory Expansion is attached, the program and the numeric variables are stored in the Expansion, leaving all the console memory available for strings - but if you do not generate strings, the console memory remains unused, because numeric data cannot overflow into it!

If your program generates more numeric variables than the Memory Expansion can hold, you can however store them in the console by converting them to strings, using STR\$, and convert them back to numbers with VAL. This will allow you store an additional 700 to 900 or more numbers. Try this -

```
100 DIM A(3040),A$(1000):: F
OR X=1 TO 3000 :: A(X)=99 ::
PRINT X :: NEXT X
110 Y=Y+1 :: A$(Y)=STR$(99)
:: PRINT Y :: GOTO 110
```

When you get MEMORY FULL, type SIZE.

Dave Renkenberger sent me a neat little routine, and I played around with it a bit. For you who are not football fans, I'd better explain that the Wave is performed at football stadiums when the cheerleaders get the fans to stand and cheer, one seating section at a time, across the stadium - and those drunks on the roof are usually out of sequence.



```
90 !THE WAVE by David Renken
berger/modified by Jim Peter
son
100 CALL CLEAR :: CALL SCREE
N(4)
110 A$="**the wave**"
120 DISPLAY AT(4,14-LEN(A$)/
2):A$
130 B$="press any key to sto
p"
140 DISPLAY AT(22,14-LEN(B$)
/2):B$
150 B$="995A3C3C3C3C2466"
160 A$="000018187EBD3C3C"
170 FOR CH=91 TO 118 :: CALL
CHAR(CH,A$):: M$=M$&CHR$(CH
):: NEXT CH :: FOR R=8 TO 12
:: DISPLAY AT(R,1):M$ :: NE
XT R
175 FOR T=1 TO 26 STEP 5 ::
DISPLAY AT(22,T):SEG$(M$,T,1
):: NEXT T
180 FOR CH=91 TO 123 :: CALL
CHAR(CH,B$):: CALL CHAR(CH-
5,A$):: CALL SOUND(-999,-7,5
*RNDD):: CALL KEY(3,K,ST):: I
F ST<>0 THEN STOP
190 NEXT CH :: GOTO 180
```

MEMORY FULL

Happy hackin'

Jim Peterson



### TIGERCUB

Remember a full page ad a few issues ago... and TIPS FROM TIGERCUB ever since...

Jim Peterson writes TIPS... exclusively for worldwide user groups, and also runs a small software house supplying simple programs for a mere US\$3 each. We don't have the full set of Tips here... so why not buy them on disk? Jim offers TIPS FROM THE TIGERCUB on disk for US\$17 (to the UK). This has a lot of text in dis/var 80 format, and the routines from the first 15 Tigercub Tips. Somewhere around 50 files on the disk!!! And the LOAD program, if you look at it closely, will teach you how to redefine the ExBas cursor (32k ram required). The LOAD program alone is excellent. And Jim's tips are worth having!

He also has available at US\$22, "NUTS AND BOLTS" a disk of 100 utility subprograms in MERGE format. My favorite is the 13 typefonts! For just one pound note Jim will send you a copy of his catalogue of \$3 programs... not complex programs, quite simple ones, but nicely written. Jim also has a nice habit of adding some unlisted programs to the tape or disk, if you send in a reasonable order! For overseas postage, Jim asks for \$5 on top of the cost of the \$3 programs, however many you order. The two disks listed above include UK postage.

Tigercub Software 156 Collingwood Avenue  
Columbus Ohio USA 43213

Stephen 

TI99/4a Hardware reset...Part one.

by Viv. Comley.

The power-up reset on the TI99/4a is achieved by taking pin 5 of the TMS 9904 4-phase clock driver to a low voltage. This is achieved automatically at switch-on, and any time that a module is inserted into the GROM port, by the circuit of figure 1.

The operation of the circuit is as follows. Firstly, assume that there is not a module inserted into the GROM port. Capacitor C606 is initially discharged, and at switch-on when +5 volts appears at the top end of resistor R605, C606 will charge up to the voltage at the junction of R605 and R606. This is around +4.6 volts, but depends upon the input resistance to pin 5 of the TMS 9904.

This sequence of operations means that the computer is reset at switch on, and the reset becomes disabled after a time given roughly by the product of C606 and R605, (i.e. 0.26 seconds). This is a necessary function to clear the random garbage that will appear in the computer's memory every time that it is switched on.

A similar process occurs when a module is inserted into the GROM port. With the computer switched on and pin 5 of the TMS 9904 around +4.6 volts, inserting a module puts -5 volts onto pin 1 of the GROM port. With C506 initially discharged this again pulls the voltage on pin 5 of the TMS 9904 to around zero volts as the charge stored in C606 is shared across to C506. The charging up process is repeated as C606 regains its voltage, but this time it only reaches to about +3 volts because of the addition of R514 and the -5 volts on pin 1 of the GROM port. This process now ensures that every time a module is inserted into the GROM port, the TI99/4a is reset.

To operate the power-up reset process at any time it is therefore only necessary to discharge C606. If the system locks up, TI recommend that the console is switched off, and then switched on again after about two minutes. The time delay is necessary to ensure that capacitor C606, and if necessary C506, become discharged sufficiently to trigger off the power-up reset when the console is switched on again. However, C606 (and C605) may be discharged by connecting a momentary push-to-make switch in series with a 560 ohm resistor across C606 (figure 2). Operation of the push button discharges C606 through the 560 ohm resistor and brings the voltage on pin 5 of the TMS 9904 down to approximately 0.2 volts. This will operate the power-up reset process without the requirement for switching the console off and on.

The above details are from the TI99/4a console and peripheral expansion technical data. For the following information I am indebted to Mike Kingham of Sunbury on Thames. The voltage on pin 5 of the TMS 9904 must be taken low (between 0 to 0.5 volts) for at least three clock cycles for the TMS 9900 to execute its power-up reset routine. With a clock frequency of about 3 MHz, three cycles will last for one microsecond, so therefore the time constants of the RC circuit more than cover the time needed for the reset operation.

The main advantage in adding the push button is that turning the console on and off is not too good for reliability. Switch-on in any electronic or electrical circuit is the time when faults can appear due to the stresses occurring, (as a poor example, lamp bulbs mainly fail when they are first turned on). Also there will be no need to wait for two minutes whilst the capacitors discharge before the power-up reset will operate. A further advantage that I have found is that a machine code program, (for example, the Display Enhancement Package), is often found to be still resident in memory and uncorrupted after a hardware reset following a crash.

*V. Comley.*

Mr. V.E.Comley.

Dormer Cottage,  
7 St. Vincents Hill,  
Redland,  
Bristol BS6 6UP.,  
Avon.

**Next issue of TI\*MES**

We will publish step by step instructions needed to install the push button into the console.

We welcome and thank Viv. Comley who has put alot of time and effort in production of this article. ED CA.

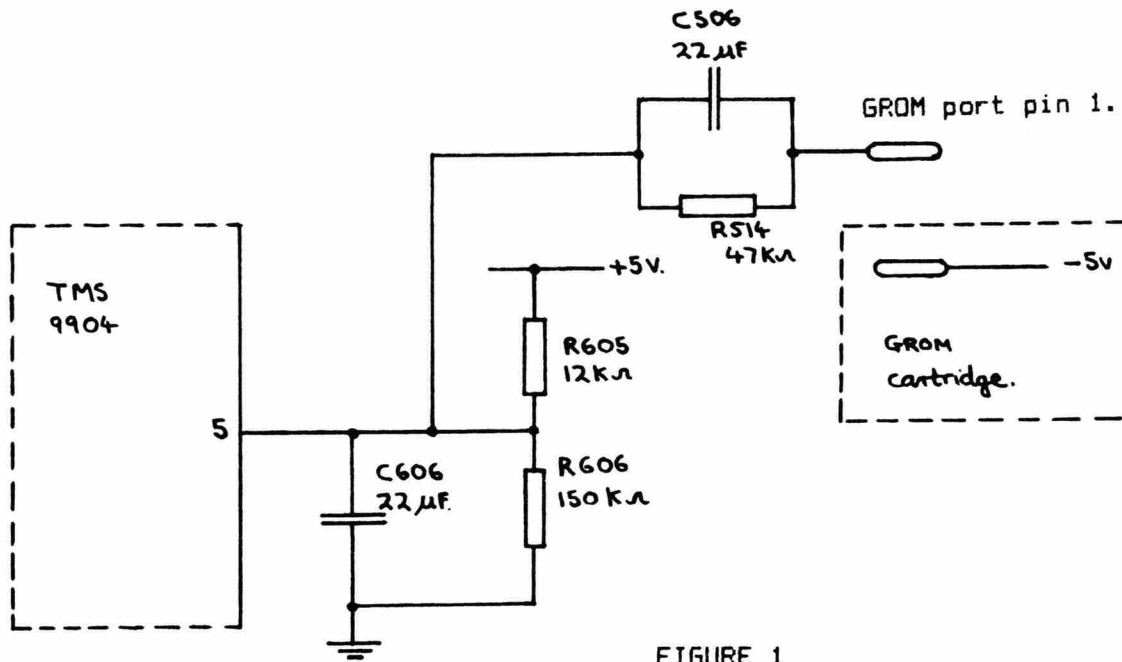


FIGURE 1  
Power-up reset circuit.

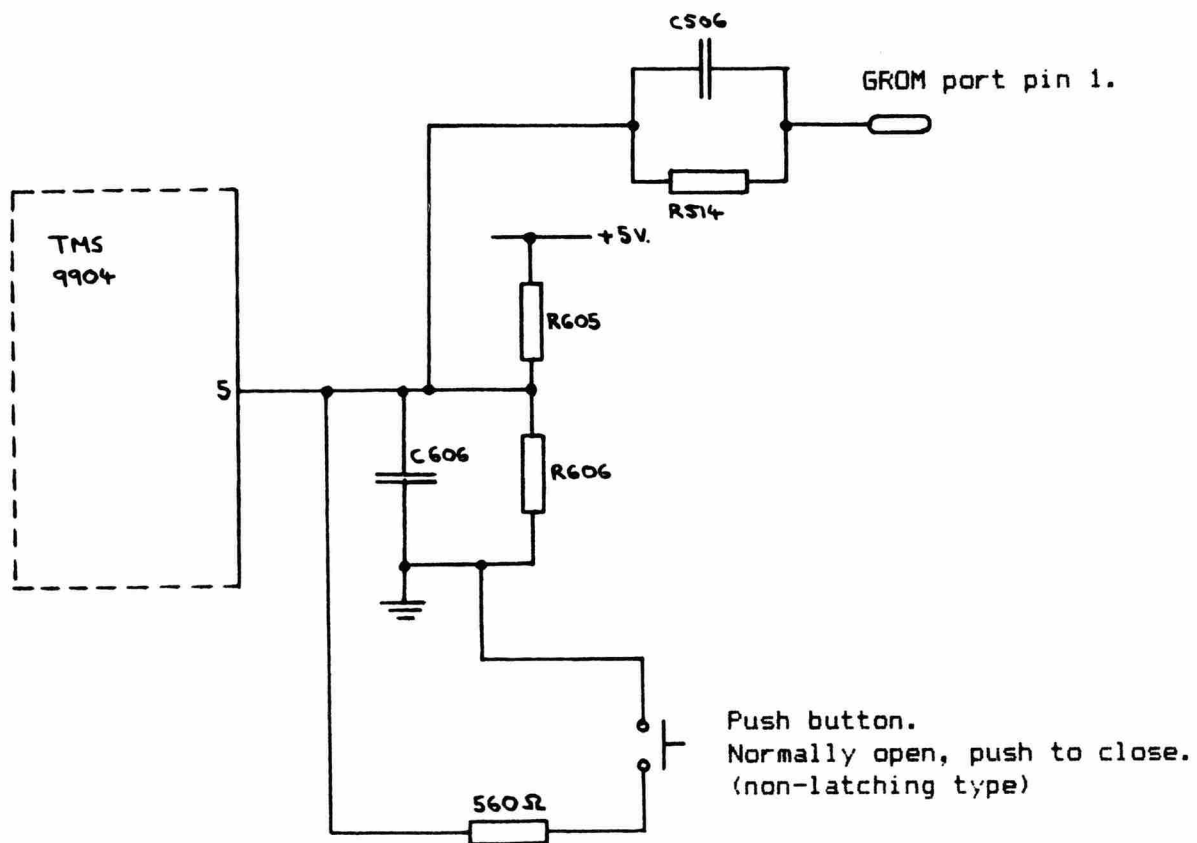


FIGURE 2

Modification to add  
reset push button.

(Full details of fitting reset button next issue of TI\*MES)

# PRINTER REVIEW

The Star Gemini 10X is one of the cheapest dot-matrix printers in its class selling at just under £200 (shopping around advisable as prices can vary enormously.)

It measures 15"x12"x5" high and weighs 15lbs. It comes complete with tractor feed and fittings for paper rolls. It will also take single sheets.

There are four LED lights for power on, ready, on line and paper out plus controls for form feed and line feed.

The printer has to be set for your computer by the setting of DIP switches but this procedure is explained in the accompanying manual.

Loading paper is straightforward. Up to 3 carbon copies can be made as the distance between the print head and roller can be adjusted. The tractor feed sprockets are adjustable for widths of up to 10". The ribbon is reversible which extends its life considerably. These are also economic to buy.



## Mickey Mouse

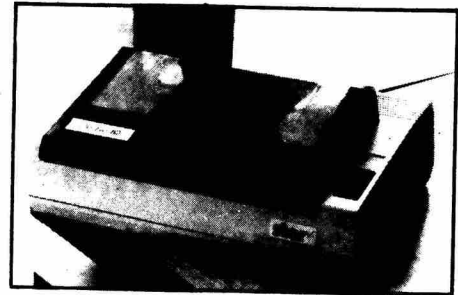
THIS IS AN EXAMPLE

THIS IS STANDARD PRINT  
 ABCDEFGHIJKLMNOPQRSTUVWXYZ  
 abcdefghijklmnopqrstuvwxyz  
 1234567890!@£\$%^&\*()+=/;-:;<.>  
 ~[]\_?'|{} \ ' "

Here are Italics

*ABCDEFGHIJKLMNPOURSTUVWXYZ*  
*abcdefghijklmnopqrstuvmxyz*  
*1234567890!@£\$%^&\*()+=/;-:;<.>*  
 ~[]\_?'|{} \ ' "

XXX) ENLARGED MODE  
 XXX:  
 ABCDEFGHIJKLMNOPQ  
 abcdefghijklmnopq  
 1234567890!@£\$%^&  
 ~[]\_?'|{} \ ' "



**STAR GEMINI-10X**  
 120CPS ● BI-DIRECTIONAL LOGIC SEEKING ●  
 FRICTION TRACTOR AND ROLL HOLDER  
 STANDARD ● DOWN LOADABLE CHARACTERS  
 ● ULTRA HIGH RESOLUTION ● 80 COLS

Print can be made in normal, elite, condensed, enlarged, emphasized, italics, double strike, super and subscript which should be enough for anyone. It can also produce high-res graphics.

The User manual is well produced and readable. Each of the 72 control code is described in detail. There is also information on wiring. The User Group has obtained from the States a TI99/4a addendum to the manual which they could photocopy for anyone interested. (12 sides) This includes cable connection instructions and a number of short programs. £1 should cover the cost.

The printer is fast- 120 characters per second and bi-directional. There is a 12 month guarantee.

All in all an extremely economical printer which performs well. With the use of emphasized printing it can produce a near letter quality print, and it fulfills all the needs required by a small business or the interested computer fanatic.



# STAINLESS SOFTWARE

10 Alstone Road STOCKPORT Cheshire SK4 5AH

CATALOGUE: From 28th April 1985, only catalogue issue G will be valid. Due to very heavily declining sales, a number of programs will cease to be available from that date. Also due to small sales, I no longer have the finance to provide free catalogues. For a copy of the catalogue (still on 5 A4 sheets) please send a donation of 50p towards costs. From April 25th, an SAE will bring a price list of titles and prices only: no details.

OVERSEAS: Please send 4 International Reply Coupons for catalogue by airmail, or two coupons for list by airmail.

## NEW PROGRAMS:

TXB...TEX BOUNCE all the way from Australia, a masterpiece by Tony McGovern (read his article in this issue). Really fast sprite action with no sprite detections missed! TXB is a sort of bagatelle/pin ball game, for one or two players, with so many variations I make around 300 games possible? Shoot a ball from one of two firers and it bounces around, being deflected by bats. When it hits a launcher, an optional bonus cup or goes off the playing field, your turn ends and a score is taken. Fascinating program, quite different to anything else. NOT a game that is over in two minutes! £8.00 in EXTENDED BASIC.

CO LIST. Listing utility from Funnelweb Farm, Australia. Used to produce the listing of the MINI MEM DEMO in this issue, and the gossip page of RAMBLES cut from this issue! Prints programs in columns 28 characters wide, in 1 2 3 or 4 columns per page (using Epson compatible printer codes). Output to disk available. Many options in this 22k+ program which includes a little machine code for speed. EXTENDED BASIC plus 32k ram required. DISK ONLY. £12.00

SECOND FLOOR in TI BASIC by R Trueman. JOYSTICK REQUIRED. £6.00

The sequel to FLOORAWAY. If you don't have FLOORAWAY yet, you must play it first! It costs £7.00 on its own, or as a special package with SECOND FLOOR, the two programs for £12.00

TI BASIC you say! Ray Kazmer, an ExBas fanatic in California played FLOORAWAY for four hours when he first saw it. Tony McGovern thinks it is the best TI BASIC program he has seen, and one of the best in any language. If you must, the program DOES run in ExBas (faster). A SUPER program, and now a worthy sequel with more perils.

The latest from PEWTERWARE, and FIVE STAR reviews from PB! in HCW!

TRACTOR FOLLIES (5 stars but review not printed at date this copy was written!) EXTENDED BASIC, Joystick required, £6.00 Drive your tractor past the oncoming obstacles. It gets harder at night time!

FAST FROG plus NORBERT. Joystick Required. Extended Basic. £7.00 the two.

Two familiar programs, neatly written. Frogger you know! Norbert is the cube hopper. PB's comment: "Excellent". True praise!

SPECIAL OFFER: All three new Pewterware programs for £12.00

Q BOND. Extended Basic by Mike Curtis... another Cube hopper! £5.00

My favorite cube hopping program. Quite different in concept to Norbert, even though the idea is the same.

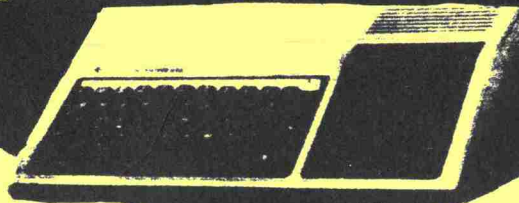
Support for Stainless Software has now dropped to a level which will not pay for advertising! Please tell your TI owning friends about our programs! (And please ask them to send an SAE when writing... funds are VERY low).

Apologies for lack of Rambles and so many promised articles. I just couldn't squeeze everything into the 20 pages allowed. Stephen Shaw.

# TI-99/4A

Users Convention

**THE  
EVERYTHING  
SHOW**  
For The  
**TEXAS INSTRUMENTS  
99/4A  
HOME COMPUTER**



Sunday 28th April 1985

The Dome

Corn Exchange

BRIGHTON

*Members are allowed  
access at 10-30am.*

*Don't miss this Show  
- IT'S MADE FOR YOU!*