





### EDITORIAL

Another bumper fun issue for you! Normally the large issues of your magazine will be the autumn and winter ones, to give you the longest reads during the winter months, but as the autumn issue was the normal length last year we are making this issue a long one. I hope you will find every page worth reading. You will find details of the arrangements for the 1991 A.G.M. in Shrewsbury. Doors open 10.30 am., formal A.G.M. starting at 1 pm., as usual.

### AGENDA

The Agenda for the formal meeting will be as follows:

1. Minutes of 1990 AGM.
2. Reports from Officers.
3. Nominations and Elections of Officers for the year.
4. Second Annual Show.
5. Any Other Business.

### DISCLAIMER

Views expressed in this publication are those of the contributor and not necessarily supported by the Committee. We would like to acknowledge any items from other publications which are not specifically attributed in the text.

### NEXT COPY DATE

All copy for the magazine should reach the Editor by 1st June. Since reproduction is by photocopier it is essential that the print should be as black as possible, and it should be arranged to make full use of the space on an A4 sheet, 15mm from the top and side edges, and 25mm from the bottom edge.

### DETAILS OF THE 1991 ANNUAL GENERAL MEETING TO BE HELD AT THE MUSIC HALL SHREWSBURY IN SHROPSHIRE ON SATURDAY MAY 11TH 1991

Yes folks its that tme of year again!! The 91 AGM is coming around again and its fallen to me this this year to organise it!! (qulp!)

The Venue is The Music Hall in Shrewsbury town centre, it is about 10 mins walk from the train station (assuming you dont get lost!!) and is quite easy to find. Should you get lost either on foot or in a car then just about any local should be able to direct you to the correct place - Shrewsbury Town is not that large and just about everyone knows where the music hall is. The maps included show where Shrewsbury is in relation to the country, and where the music hall is in the town centre.

Entry to the AGM is free and there will be lots to see and hear. The committee meeting will take place as well as election of committee members and questions to committee members from the floor. There will be many TI experts there to answer any questions you may have, and the librarians will be there (module/cassette/disk) so you can update your software collection if you so wish, and there will plenty of eager TI users there (I hope!) to share TI anecdotes with!!

The music hall itself has a cafeteria on the ground floor should you get hungry, there is a public house literally around the corner, and there is also a Public Information Centre inside the Music Hall that can tell you all the places you should visit while you are in the Town. (Like the towns 700 year old castle!!)

The town is well served both rail and road wise, it has a main station as mentioned before, and is easy to get to from the road. When you approach Shrewsbury, you want to get on the A5 or the A49 depending on which direction you are coming in from. (see the map)

Do please try to come along. The group is what you make it and if you are not there to make any suggestions then how can we expect to improve the services of the group. Enough said!!

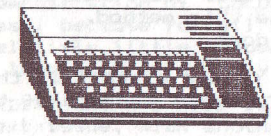
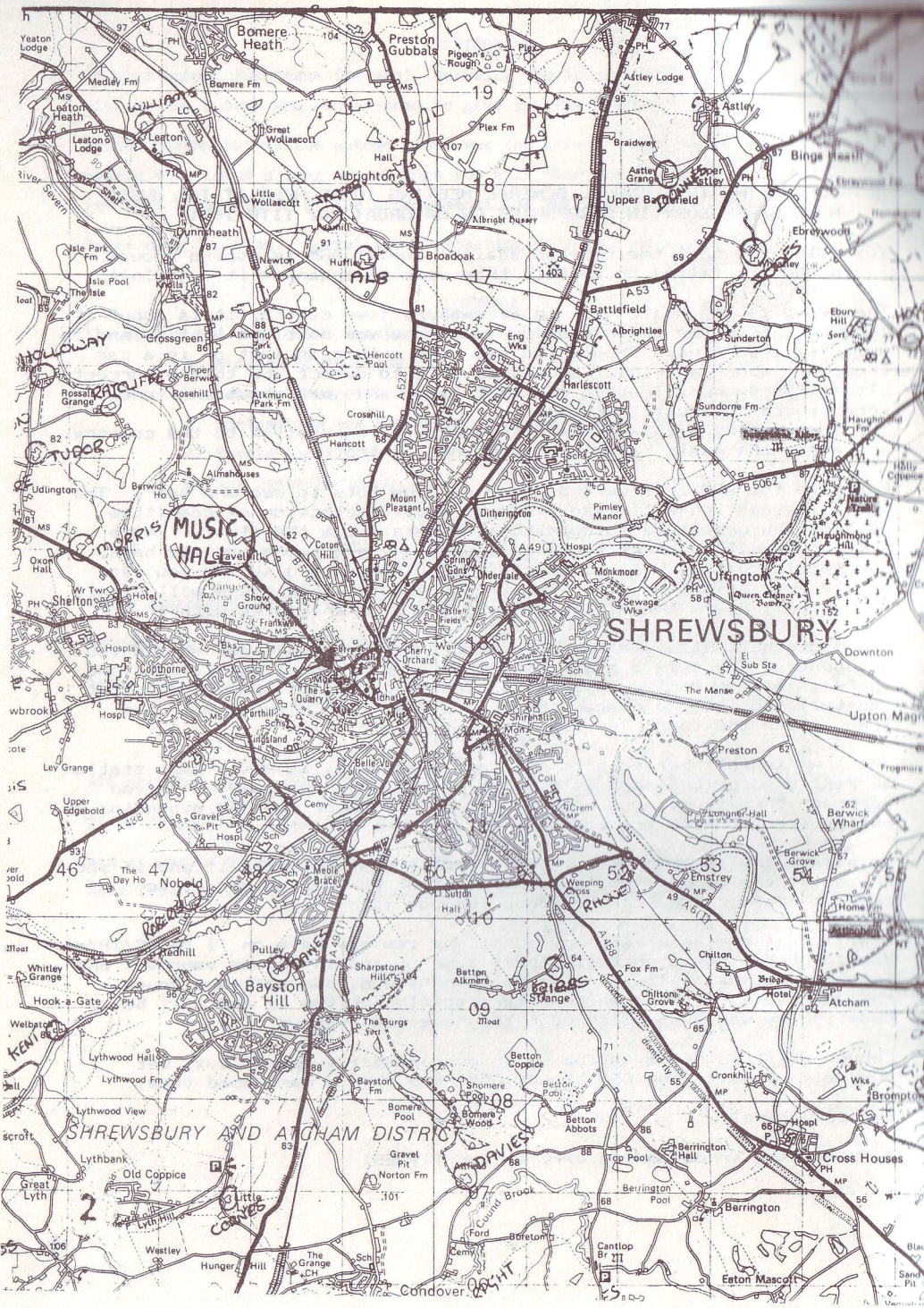
I will be there to answer any questions you may have on TI BAISC/XB/or Machine Code. Don't just sit there come up and introduce yourself and lets have a chat!! I will also be launching my own software house called Abbots Software which caters solely for the TI and will be pedaling my wares so watch out! You have been warned!!

Enough waffling! I am taking up to much magazine space! Just get yourself along to the AGM at ANY COST and have a darn good time!!

See you there!!

Mark Wills. (Programming officer for TI\*MES)





# CONSOLE ONLY CORNER

BY MARK WILLS

FROM THE PROGRAMMING OFFICERS VERY OWN FINGERS(?!)  
-----

Greetings from the planet cranium! I bet you didnt know that the group even had a "Programming Officer" well Ha Ha! it has. So what do you do? I hear you ask (no snide comments please Mike G.!!) well up until now, not a lot - but all that is about to change. It has been difficult writing articles for the mag until now because although I had a printer and interface from MGCS (plug plug!) I didn't have an expanded system, and so couldn't use any word processing software to write on. That has all changed also thanks to the PEB that steven and MGCS (plug plug!) arranged for me. Now I shall be able to write reams and reams of pointless drivel about things TI!

My job is really to stimulate you programming buffs out there and (more importantly) help the more inexperienced members (unexpanded impartial) to actually learn something from their TI. I shall submit an article or two every issue and hopefully a program of some sort, this will unexpanded wherever possible.

I thought i'd start off this issue with something simple but effective, and unexpanded. (I don't think the unexpanded owner is very well catered for - i'm going to TRY and change that.)

Have any of you ever experimented with mixing color DOOPS! sorry I mean colour on your TI. We all know that in the CALL COLOR() subprogram we can specify a foreground (the colour of the writing on the screen for eq.) and the background (the colour of the screen BEHIND the writing) colour for the character sets.

By defining a character with CALL CHAR() that has a shape with quite a lot of dots (pixels) on, and quite a lot off, if you give the foreground and background colours, a different colour, then the two colours will appear to be mixed on the screen into one colour. Confused? I thought you would be! Let me elaborate...

If you took a shape like this :    % % % %  
wich has the code AA55AA55AA55AA55    % % % %  
and give the foreground (the %s)    % % % %  
the colour red, and the background    % % % %  
(the spaces) the colour white, you    % % % %  
would find that you had rather a    % % % %  
nice shade of orange. This rather    % % % %  
simple concept can be expanded to    % % % %  
give our humble TI a range of 256  
colours - and by combining different combinations even more can be  
obtained. The drawback is that the TI (Tiny as I affectionately call  
it) can still only show 16 of those 256 colours on the screen at the  
same time. This is because Tiny has only sixteen character sets.



If you type in the program below (TI basic, XB if you use the VDP util from disk library) you will see for yourself the range of beautiful colours available to you if you use this method. I will go through what the program actually does

First it clears the screen and sets it to the colour black. It then initiates a loop which defines every second character in each of the sixteen character sets to shape shown above. The space character is set to a solid block which is coloured black, it stays coloured black throughout the program so that the screen itself is coloured black if you see what I mean(?!)

The variable CH is set up and given a value of 33. (The second character in character set 1. (Chr\$(33)="!"))

The nested loops Y and X are used to display a large box of characters in conjunction with CALL HCHAR(). At each pass through the loop, CH is incremented by eight, so that it points to the second character of the next character set. It is then tested to make sure that it isn't trying to point to character higher than 152. (There are only 159 characters available), and if it is, it jumps to line 190 to reset CH back to 33. Notice that although we are jumping out of the loop, we are re-entering it at the NEXT statement in line 160. If we were to jump to line 120 each time, the computer would think that we are creating a new loop called X each time, and the FOR/NEXT stack would soon fill up, causing an OUT OF MEMORY error.

Then comes the loop to cycle through all 256 colours. This is done by colouring the foreground of each character set with each of the available 16 colours, but colouring the background of all sixteen character sets with the same colour, (which is different from the foreground colour!). Then, we cycle through each of the sixteen background colours, which creates the illusion of shades of colours. (I said it was confusing. Have a ganders at the program, it explains it better than I!!)

```
10 REM 256 COLOURS DEMO
20 REM M.WILLS 1990
30 REM
40 CALL CLEAR
50 CALL SCREEN(2)
60 FOR I=33 TO 152 STEP 8
70 CALL CHAR(I,"AA55AA55AA55AA55")
80 NEXT I
90 CALL CHAR(32,"FFFFFFFFFFFFFF")
100 CH=33
110 FOR Y=5 TO 20
120 FOR X=10 TO 25
130 CALL HCHAR(Y,X,CH,1)
140 CH=CH+8
150 IF CH>152 THEN 190
160 NEXT X
170 NEXT Y
180 GOTO 210
190 CH=33
200 GOTO 160
210 B=2
220 FOR I=1 TO 16
230 CALL COLOR(I,I,B)
240 NEXT I
250 B=B+1
260 IF B>16 THEN 210
270 GOTO 220
```

4

A short but incredibly effective program, giving a rather nice demonstration of "COLOUR" or "PALLETE" mixing. It also runs very fast, because for all the screen activity taking place, there is relatively little happening in the program, it is just cycling through a loop which effectively writes a different CALL COLOR each time.

Well I think I will leave it there for now, rest assured I will write next issue, with another article, until then I hope you liked this article, if it seems a little oddly put together its probably because it's my first, and has been written "out of my head" as it occurred to me, I should get better at it as time progresses!

It was great to see so many people at CUFFLEY this month, I was there demoing my MINI MEM extension to TI BASIC which will shortly be available for sale through DATABASE and MGCS (plug plug!!). More details next issue.

I have submitted the game Creepy Crawlies (hopefully in this issue) for publication, whatever you do, TYPE IT IN, it shows you just what can be done in TI BASIC. It will also run in EXTENDED BASIC unmodified, for extra speed. It is very long I know, but do please type it in because the effort is worth it. (I would say that because I wrote it!) or if you don't want to type it in then send me a disk or cassette plus stamped addressed envelope and I will send you a free COPY. (Could not get it in this issue. Please ask Mark. Ed.)

Bye for now and may your machine never get Tilt!!

Mark Wills. T.I.U.G(UK)  
Programming Officer for TI\*MES  
Committee Member. (Address on front cover)

#### RAMBLES POST SCRIPT UP DATE

In the text of RAMBLES I have mentioned that Database has the Waterworks disk at ten pounds. In mid February I purchased this one from Martin for TWELVE pounds plus the usual pound postage. Check all prices direct with Martin before ordering! The game is ~~in~~ of the manual is ~~in~~ **\*\* The copy of WATERWORKS supplied was fatally bugged and unplayable. \*\***

~~from ordering- it looks good.~~

Apologies to Martin too, for not spelling his surname correctly, which is Blyth WITHOUT a final E (Welsh spelling?).

A little too close to press date Martin sent me a sample of a games disk with three games, entitled BINDOFF GAMES 1. These are XB games, and a quick play shows them to be original and playable. I do not know the price! but they look worth consideration- more next issue after I have had adequate time to play them properly. Purchasers will prefer to print out the docs with TI Writer rather than use the rather primitive XB viewer/printer on the disk!

There are areas of difficulty out there which I do not know about... what would you like TI Writer to do and dont know how to? Need an article on mail merge, transliterate, formatting? Any problems with anything (except hardware-see Mike- or Machine Code) drop me a line, and if you would like a reply direct, an SAE is always a great help!

Problem answered: to load a long cassette program when you have a disk drive which gets in the way due to buffer reservation, send to the disk library for the two disk offering of REBEL disks, which will be of help. If you send two disks the cost is only three pounds.

\*\* The disk library welcomes contributions of new material either written by yourself or obtained from elsewhere!! Thank you! \*\*

5



MODULES MODULES MODULES MODULES

APPEAL TO ALL DISK DRIVE OWNERS.....

Do you have any modules that you would consider selling or donating to the module library. Reasonable prices paid. For more information please contact me at the address given below.

The latest list of modules available for purchase follows! please note that cheques should be made payable to "E.H.SHAW". Also members are advised to contact me about the modules that they are seeking as the stock is constantly changing.

ADDITION AND SUBTRACTION 1	2.00	PARSEC	3.50
ADDITION AND SUBTRACTION 2	3.00	NUMBER MAGIC	4.00
ADVENTURE and PIRATE TAPE	5.00	HOUSEHOLD BUDGET MAN.	3.00
BLACKJACK + POKER	2.50	PROTECTOR	4.50
HUNT THE WUMPUS	3.00	SHAMUS	3.50
BEGINNING GRAMMAR	3.00	DEFENDER	4.00
HOME FINANCIAL DECISIONS	2.50	OTHELLO	4.00
CAR WARS	5.00	ALPINER	4.00
CONNECT 4	3.50	THE ATTACK	3.50
MUSIC MAKER	5.00	TI INVADERS	3.50
* DISK MANAGER	2.00	STATISTICS (NO DOCS)	3.00
EARLY READING	2.50	SPEECH EDITOR	4.00
EARLY LEARNING FUN	2.50	SUPER DEMON ATTACK	4.50
* EXTENDED BASIC AND MANUAL	22.50	* TI MULTIPLAN/DISK/MANUAL	25.00
PERSONAL RECORD KEEPING	3.50	TOMBSTONE CITY	2.50
PERSONAL REPORT GENERATOR	3.50	TERMINAL EMULATOR II	5.00
NUMERATION 1	3.00	VIDEO GAMES 1	3.50
MULTIPLICATION 1	3.00	VIDEO CHESS	4.50
MUNCHMAN	4.00	YAHTZEE	4.00
MOONMINE	5.00	ZERO ZAP	3.00
DATABIOTICS WORDWRITER+ (includes PIO cable + manual).....	25.00		

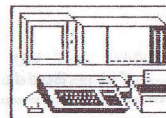
\* MODULES MARKED WITH AN ASTERISK REQUIRE DISKS OR 32K RAM OR BOTH. ALSO PLEASE NOTE THAT EARLY READING NEEDS A SPEECH SYNTH TO RUN.

PURCHASING MODULES FROM THE LIBRARY

You may return any module purchased within four weeks and be refunded the purchase price less postage which will be charged at the rate of 20 pence per module.

Application to loan/purchase modules:

NAME: .....	Module required:.....
ADDRESS: .....	.....
.....	.....
.....	.....
.....	.....
.....	.....
I enclose cheque/PO for £..... (as indicated on the list) & post to	
PLEASE MAKE CHEQUES PAYABLE TO E.H.SHAW.	MR. E.H. SHAW
Foreign orders can only be accepted if a	CROW HOLT FARM
BANKERS DRAFT is enclosed drawn in STERLING	BASFORD
OR a LONDON bank. It also helps if a little	LEEK
extra is added on for postage overseas.	STAFFS, ST13 7DU



# MEMBERSHIP NEWS

from Peter Walker

Membership Secretary

Since issue 31 we welcome the following new members: David Pollicott, Walter Allum, James Smith, Richard Barrett, Gene Bohut, R.E. Metz, Stephen Crabb, Nicandro Peluso, Mrs M.E. Mathers, David Bent, Mark Windram, John Senter, Paul Saunders, Sarah & Rosemary English, James Johnson, David Rogers, Nathan Sanuel, A.Thomas, André Cornélis and Henry Weatherburn. The welcome influx of new members is a result of increased publicity for the group, both here and in the USA. Many thanks to Phil and Stephen for this. It keeps me busy answering the many enquiries from prospective members.

The Cuffley Show was a quiet success, some 38 members attended and a wide variety of software and hardware was demonstrated. In fact I never got round to see everything on show: that is the problem when you yourself are demonstrating. Over the day I demo'ed TE2 protocols, TI-Artist with Mouse, Mouse driven Telephone Tones demo, Telephone Dialler, Membership Database, TI-BASE, Extended Display Package, LOGO Database, LOGO Doctor & Animal programs, 80 column editor & diskreview, Page Pro 99 and Multiplan. Well that's all I can remember anyway! In between times, my disks and RamDisks were playing up. Many thanks to all who turned up with their systems, to DATABASE and Edward for their sales tables and indeed all who attended.

BACK ISSUES: Still available (but some in small numbers): 4, 8-14,16,18-20,22-27,29-31. New special deal: 1-2 issues £2 each, 3-4 £1.50 each, 5 and over £1 each. Postage extra!! For those wishing to order multiple copies, I am prepared to try a trust system. I will send your order and when you see how much the order cost to post you can send a cheque back to me for the full amount (payable to the Group, not me please).

Many of you will this issue receive your renewal reminder. For those not aware, for historical reasons (well 1987 to be exact) a large proportion of our members have subscriptions running from July through to June. I sincerely hope that as many of you as possible will renew. Even if you are thinking of moving to another machine, don't sell up your TI99/4A!! Stay with us! You would be surprised how many people have regretted their loss of the TI99/4A afterwards. I have a PC compatible at home, but still use the TI more!

May the bytes be with you!

PW

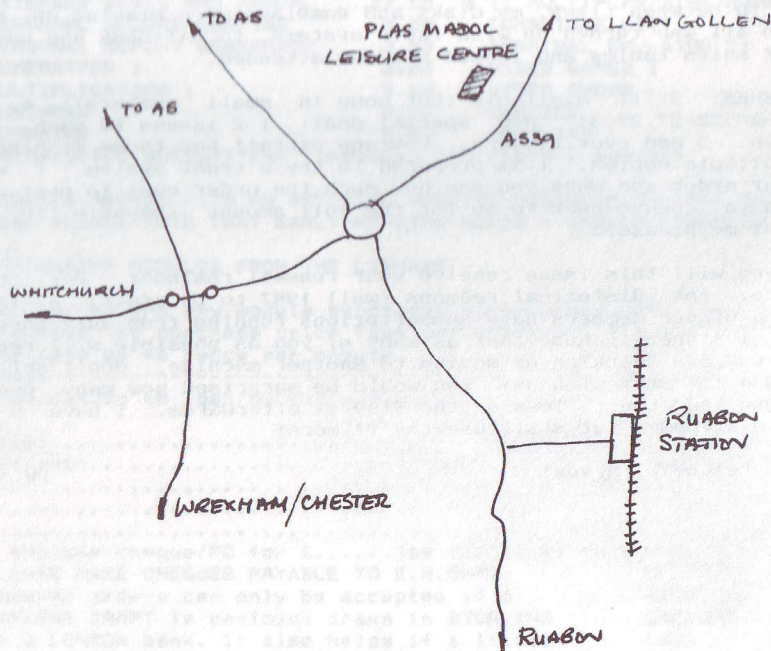


The next CHESTER/WREXHAM workshop will take place on Saturday 22nd JUNE at the PLAS MADOC LEISURE CENTRE in the AQUA LOUNGE this centre boasts an indoor heated pool sport facilities and a full cafeteria it is at Acrefair on the A539 Llangollen road south of Wrexham the nearest train station is Ruabon situated on the Wrexham/Shrewsbury line and is about 5mins walk away. By bus take any Llangollen bus from either Wrexham or Chester.

We will try to cover whatever the majority of members are interested in suggestions to me please (Mike). I hope by then to have a Quest RAM disk to demonstrate. Also Mark Wills will be demonstrating the latest from Abbotsoft TI software some of which really shows what the TI can do with properly written software in Basic.

This meeting will be open to anybody with an interest in the TI-99/4A wether a member of the group or not and we hope as many of you as possible try to get along.

MIKE



News at last on the Dutch 80 column card. I've been in contact with Berry Harmsen of "TI-GEBRIUKERSGROEP" and he has told me there have been unexpected development problems but these are now resolved and they have some prototypes working and will send me full details as soon as they can.

Better news from Aussie land who have produced the Quest RAM disk this was demonstrated by Mike Poskitt and John Murphy at the Cuffley meeting although neither had completed their RAM disks the benefits were fairly obvious the layout was neat and tidy and by using 32k 62256 memory chips only 16 chips are required and a 17th can replace the 32k memory card thereby freeing another slot in the box. Full details can be found in the June 90 Micropendium.

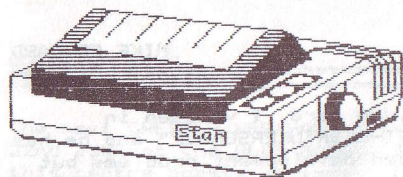
Hardware continues to change hands at a good rate with supply struggling to keep up with demand. It is nice to see people who sold up a few years ago coming back and requesting full expansion systems as well. The January sale was sold out within the first week or two with a few members acquiring some never to be repeated bargains and members from as far away as France and Sweden showing interest. For those of you that were disappointed I'm sorry better luck next time.

As promised below is a list of Radio Rallies that M.G.C.S hope to attend in the coming year if you're about come and see us for a chat and you never know maybe a bargain.

- 31 MARCH : CENTRE OF ENGLAND RALLY. (1)  
NATIONAL MOTORCYCLE MUSEUM BIRMINGHAM.
- 6 MAY : MID CHESHIRE RALLY.  
CIVIC HALL WINSFORD CHESHIRE.
- 18 AUG : RED ROSE RALLY (1)  
BOLTON SPORTS AND EXHIBITION CENTRE.
- 22 SEPT : CENTRE OF ENGLAND RALLY. (2)  
NATIONAL MOTORCYCLE MUSEUM BIRMINGHAM.
- 2-3 NOV : NORTH WALES RADIO RALLY ( 2 DAY EVENT )  
ABERCONWY CENTRE LLANDUDNO.
- 24 NOV : RED ROSE RALLY (2)  
BOLTON SPORTS AND EXHIBITION CENTRE.

Mike





## PRINTER SURVEY

FEEDBACK AND ADVICE

ON PRINTERS FOR THE

TI99/4A

by Peter Walker

Many thanks to the following members who responded to the printer survey as requested in issue 30, in no particular order: David Duncan, Peter Hutchison, Chris Christian, Robert Bates, Phil Trotter, John Seager, Stephen Shaw, Ashley Tilling, Graham Steward, Alain Couegnat, John Harris, Peter Jackson, Bill Moran, John Dunning, Jim Ballinger, Gary Smith and Barrie Clark. With myself, this gives a population of 18, not large enough to be truly representative but enough to suggest some general advice.

Few members have actually had problems with printers, which is encouraging. The printers used include: Epson MX80, RX80, FX80, LX80, LX800 and LQ500; Star SG10 and LC10; Panasonic KXP1081 and 1091; Shinwa CP80 and CP80+; Citizen 120D; Brother M1009 and EP44 (typewriter); and some miscellaneous daisywheels: Triumph Adler Royal Office Master 2000, JUKI 2200 (typewriter) and Smith Corona L1000. In terms of recommending printers, it would appear that you should consider Epson, Star and Panasonic. Probably best to avoid IBM and some Tandy printers. I have decided against printing here all the details of all the printers as volunteered by members as it would get very repetitive, but if any specific details are wanted, please phone me. If purchasing another type or acquiring second-hand, the following advice has been drawn from the responses and my own experiences over the years:

Daisywheel printers with quality ribbons give the best quality print, but are slow, noisy, can't do bit-map graphics and are generally less flexible.

The TI99/4A requires very specific and non-standard wiring for both its PIO and RS232 ports, so don't expect a shop bought lead or one used on another computer to work. The correct wiring is given at the end of this article.

If you have problems with the PIO port even with the correct wiring, it is possible that your printer requires a "true" Centronics interface. I know at least one OKI printer that has this problem, and I think Barrie Clark may have experienced the same thing with the Colour version of the Star LC-10. Our PIO is not strictly Centronics, in that it expects the printer to acknowledge each character with a short burst of "printer busy" on the busy lead. (Don't confuse this with the acknowledge pulse which has its own lead and is not used by the TI.) Some printers only send "printer busy" when the printer is truly busy and wants no more text to be sent; ie the various

conditions: Buffer Full, Manual Busy, Paper Out etc. A revised DSR ROM chip has in the past been available to convert the TI RS232 card to true Centronics which overcomes this problem. Contact Mike Goddard for availability of this chip.

Sometimes when printing bit-map graphics, you must set beforehand an appropriate line spacing to make the dots contiguous. Sometimes the auto LF has to be set too.

If using a printer on the RS232, don't use the very fastest speeds. Printing speed is overall determined by the printer, so using 9600 or 4800 doesn't speed anything up and our machine is sometimes unreliable at these fastest speeds.

If buying an old printer, avoid those that use DIP switches for changing settings that you want to alter frequently, especially if these can't be changed by received Control Codes.

CR and LF control. This is the source of frequent problems. The TI generally issues its own CR and LFs when printing display records, so avoid setting your printer to modes that can generally be described as:

"CR forces LF", "LF forces CR" or "80 chars forces LF"

The first and last of these will tend to give you problems, such as double spacing between lines when not wanted. The filename extensions .LF and .CR used for printer filenames work as following:

.LF causes the TI not to add a LF after each line.

.CR causes the TI not to add a LF or CR after each line.

You use these when the LF or CR is explicitly included in the text string you are printing. For example, text from the TI Writer Formatter includes explicit LFs, so the printer filename extension .LF should always be used. This .LF can also be used on regular text to overcome the setting "CR forces LF", but there is no similar way to overcome "LF forces CR".

I have seen many instances of TI software using the .CRLF filename extension - there is no such extension! It is interpreted as .CR which on its own will remove both the auto CR and LF. Worse still is the use of both .LF and .CR together. The MacFlix package uses a default printer name of PIO.LF.CR which causes incorrect Dot Graphics printing on my system. PIO.CR is the correct name! Programmers worldwide please note!!

Do ensure that your printer is Epson compatible, as many TI packages assume this, though some are now featuring printer profile files that will allow wider compatibility, such as TI-BASE, TI-Artist, and Page-Pro 99. Some printers are switchable between Epson and IBM emulation, which gives you the best of both worlds.

Try to get a printer with a wide set of fonts & styles - most modern ones do, but my old MX80 is nowhere near as good as more recent models. It's nice to have NLQ (Near Letter Quality), several type faces, Italics, Pica, Elite, Condensed and Double, proportional spacing, multiple national alphabets, downloadable character sets, amongst other goodies.



For most computer applications you will be using tractor fed fanfold paper, so do get a tractor feed. Friction feed is needed as well for cut sheet work or continuous paper rolls. A cut-sheet feeder is a nice accessory to have. With cut sheets, you must be able to disable the end of paper alarm; usually there is a Control Code for this.

If exchanging text files with others or depositing README files with programs in libraries, please don't include printer control codes in your text. Even if most people are Epson compatible, there are several variants of Epson codes around. The worst offender for this is the TI-BASE tutorial set which includes codes for reversing the direction of the printer, a feature few of us have. The net effect is to produce a printout which is nearly unreadable. Both Ed/Ass and TI-Writer have features for stripping out Control Codes (0-31 in ASCII), though this will still leave a few rogue characters which were Escape modifiers. In general, the only characters which are universally recognised are CR, LF, PA and BEL.

If anyone else has any words of wisdom on printers, please write in and tell us. Finally, here are the PIO and RS232 port pin configurations. Please note that the configuration shown for the printer is not universal, so do check the exact pin-outs of your printer before constructing a lead.

PW

#### RS232 PRINTER CONNECTIONS

TI99/4A	PRINTER
RD 2 (14) <-----	2 Data Out (Note 2)
TD 3 (16) ----->	3 Data In
(Note 3) DSR 20 (19) <-----	20 Buffer Full/Busy (Note 4)
Earth 7 (7) -----	7 Earth

#### Notes:

1. TI99 leads in brackets refer to RS232 port 2
2. Connection only needed for printers using X-ON/OFF protocol, rather than pin 20 handshake.
3. The lead 20 is called DTR in the manual, but is in reality a DSR, see issue 20 for a full discussion of RS232 protocol.
4. Some printers use pin 11 instead of pin 20. Equally, the TI99 card can be re-strapped to put the DSR on pin 11.

#### PIO PRINTER CONNECTIONS

TI99/4A	PRINTER
Handshake Out 1 ----->	1 Strobe
8 bit Data 2-9 ----->	2-9 Data In
Handshake In 10 <-----	11 Busy
Earth 11 -----	16 Logic Ground

Note: The printer lead numbers are those used by my Epson MX80, others may vary. Do not use the acknowledge signal on the printer.

#### CASSETTE LIBRARY REPORT.....

NICKY GODDARD

The cassette library is being used by more and more members although I am still having problems listing the library contents as it was in quite a jumble when I took it over so for the time being the last list published in TI\*MES is still valid although there have been very few additions recently. I would like to reintroduce the original idea of library cassettes for donations that is if you send an original program or programs in you will receive the same amount of programs of your choice from the Library in return.

I thought that the meeting at Cufflev was a success quite a few people turned up but not many cassette owners as I found out during the raffle(which had prizes for both cassette and disk owners).

#### SURPLUS STOCK CASSETTES SURPLUS STOCK CASSETTES SURPLUS STOCK CASSET

SPECIAL OFFER SURPLUS STOCK CASSETTE PRICE - 50p EACH  
POSTAGE 35p FOR THE FIRST ONE AND THEN 15p EACH THEREAFTER

#### GAMES--:

BLAST IT, OLDIES BUT GOODIES GAMES 2, PHAROAH'S CURSE/3D NOUGHTS AND CROSSES, CORE, MARKET SIMULATION, SORCERER'S CASTLE/LUNAR LANDER, PARCO HOP ON, BATTLESHIPS, ATTACKMAN, MISSION 99, BATTLESTAR ATTACK, HUCHBACK HAVOC, OLDIES BUT GOODIES GAMES 1, ARENA III, SENGOKU JIDAI, FORBIDDEN CITY/MASTERBRAIN, DEVILS ISLAND/RUSSIAN ROULETTE, KONG, FUNPACK 1, SPACE INVADERS/STAR TREK, WAR GAME, SNAKE, PARCO GOLF, LAZER LANK.

#### EDUCATIONAL GAMES--:

HAPPYMATH.

#### UTILITIES--:

STARTER PACK 1, GAMES WRITERS PACK 2, GRAPHIC PAIRS, PERSONAL FINANCIAL AIDS, TI LOGO SAMPLER, BEGINNERS BASIC TUTOR, GAMES WRITERS PACK 2, TEACH YOURSELF BASIC, STARTER PACK 2, GRAPHING PACKAGE, TEACH YOURSELF EXTENDED BASIC, GAMES WRITERS PACK 1, VIDEO TITLES.

#### BEST OF 99ER ON TAPE--:

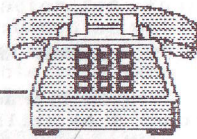
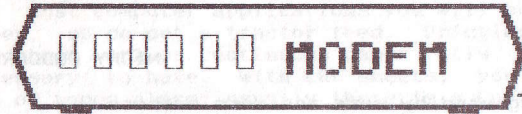
TAPE 1, TAPE 2, TAPE 3, TAPE 4, TAPE 5.

NB ANY TITLE WITH A STROKE BEFORE THE FIRST LETTER ARE ON THE

SAME TAPE AS THE TITLE BEFORE IT.

NICKY GODDARD, TI USER GROUP UK, CASSETTE LIBRARIAN, "SARNIA", CEMETERY ROAD, RHOSLLANERCHRUGOG, WREXHAM, CLWYD, LL14 2BY.





## TELECOM TIPS -TE2

by Peter Walker \*\* TE2 PROTOCOLS Part 2

In this issue we will start looking at the more complicated parts of the TE2 protocol, namely the Extended Writes which allow you to change a wide range of TI99/4A attributes in the remote system, such as playing sound, speaking words, changing the colour of the screen and characters sets, redefining characters etc. Those of you who attended the Cuffley Show would have seen my TE2 demo that showed all of these in action.

In the last issue I discussed the use of Escape Codes, ie sequences starting with ASCII 27 or <ESC>. All Extended Writes use the Escape sequence <ESC>G<DEL><ESC>( at the beginning and <ESC>) at the end. <DEL> = ASCII 127. In the middle is a variable length sequence which starts with a one byte Operation Code and is followed by data relevant to the particular operation.

Now there are some important issues which arise from this structure and its use over telecommunication systems. Some packet networks use the 8th bit as an active parity bit, which means that only 7 bit data can be used, so characters 128-255 cannot be used. We must also avoid any byte values which can cause misoperation of the data network such as Control Codes which initiate breaking down the session. Thus the host should avoid sending ASCII values in the control range 1-26.

TE2 Extended Write data must also avoid any data values which can incorrectly simulate the <ESC> code which signals the end of the data string. Thus the data, which can normally be any value from 0 to 255 should not cause values below 32 to be sent.

Lastly, it is easier to create TE2 files if most of the values (aside from <ESC>) correspond to normal ASCII letter characters, since not all file editors can enter ASCII characters below 32. TI Writer can, using the control u mode, but others cannot.

For all these reasons, the data field within Extended Writes is often coded in a special way which ensures that all transmitted bytes are in the range 33-127. Further, other TE2 codes will often add 32 to a value to ensure that only 'editable' characters are produced. A good example of this is the escape Y command used for positioning the cursor on the screen, which was discussed last time.

But to return to the issue of Extended Write command data. The coding system used is to code all bytes in the form 01xxxxxx. The string of 8 bit bytes is spread over the 6 bit fields in bit sequence, and any remaining bits are set to 0. Thus if we wish to encode the sequence of

bytes 234, 53, 21 we would proceed as follows. Convert 234, 53, 21 to binary:

11101010 00110101 00010101 then break into strings of 6 bits:

111010 100011 010100 010101 then prefix with 01:

01111010 01100011 01010100 01010101 then convert to byte numbers again:

122, 99, 84, 85; this sequence corresponds to zcTU in character form. All this conversion is somewhat tedious, so I have developed an ExBas program which does the conversion for you. It is listed at the end of this article. It converts numbers in DATA statements and displays the result on the screen, though it would be very easy to modify the program to other methods eg reading in binary or output to DV80 file.

By now I suspect that I may have lost a good deal of the people who started reading this article who probably hoped to find some easy Extended Write they could try straightaway. Lets look at the Extended Write needed to make the remote computer speak. This uses the Operation Code 39 (Hex >27) or in character form '. All you do is to write the words to speak in ASCII upper case in the data field. The coding is therefore:

<ESC>G<DEL><ESC>('HELLO FOLKS<ESC>) or in character values:-

27,71,127,27,40,39,72,69,76,76,79,32,70,79,76,75,83,27,41

If you want the remote TI99 to display the words as well as speak them use Op Code 38 (&) instead.

Now lets look at a more complex Extended Write. Suppose you want the remote computer to display ASCII 35 as a pound sign £ instead of a hash #. We must redefine character 35, just as one would do with CALL CHAR() in Basic. We use Op Code 32 which is the <Space> character. The data is the usual 8 byte character definition string as used in CALL CHAR. The £ sign is defined by the string 00 1C 22 20 78 20 20 7E. However, in this case the special encoding discussed above must be used which results in the string @ApbHG`HGx. The Op Code is followed by two bytes used to code the value 35, again, in a way which keeps the coded values in the range above ASCII 32. The code 35 is >23 in Hex. The two parts >2 and >3 are separately added to >20 to produce the two bytes >22 and >23, which is ASCII 34 and 35; ie "#

Thus the complete sequence would be:

<ESC>G<DEL><ESC>("#@ApbHG`HGx<ESC>)

After TE2 has received this, all subsequent ASCII 35s will be displayed as £. Should you want to redefine a contiguous block of characters eg 35,36,37 all you have to do is to add the definition string(s) to that of 35 in the data field and then encode. TE2 will interpret the data as sequential character definitions until the <ESC>) is received.



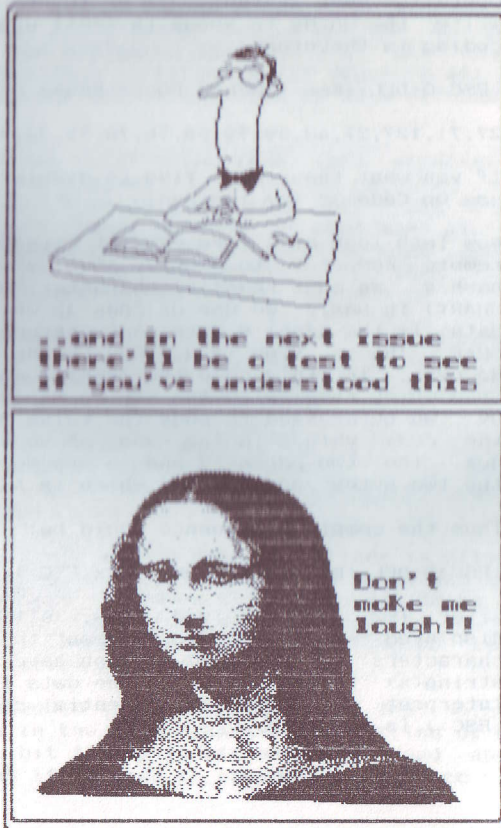
The complete list of Op Codes is:

- 32 Define Character(s)
- 33 Load Sound Tables
- 34 Play Sound Tables
- 35 Stop Sound
- 36 Select Character Bank (0-127 or 128-255)
- 37 Define Colour Sets
- 38 Speak & Display Text
- 39 Speak Text without display
- 40 Speak Allophones
- 41 Look up words (associate a number with word in speech dictionary)
- 42 Speak words associated with numbers (from Op 41)
- 43 Change Screen Colour (graphics mode)  
Change Foreground & Background Colours (text mode)
- 44 Return Status (causes TE2 to return certain status information such as Screen width, Display Mode, and Wrap)

There isn't space here to describe the structure of all these Extended Writes. Enthusiasts are recommended to get the full TE2 Protocol Manual. Otherwise I might well return to some of these in later issues.

Peter Walker

```
1  REM CONVERTS 8 BIT DATA TO
2  TE2 ENCODED FORM.
3  REM ENTER DATA IN LINE 260
100 DIM A(30),B(40)
110 FOR J=1 TO 100
120 READ A(J)
130 IF A(J)>256 THEN A(J)=0
14 GOTO 150
140 NEXT J
150 B=256*A(1)+A(2):: K=3
160 L=1 :: D=1024
170 F=B/D
180 IF F>32768 THEN F=F-32768
B
190 B(L)=64 OR(INT(F)AND 63)
200 PRINT CHR$(B(L));
210 IF ST=1 THEN STOP
220 IF D<64 THEN D=INT(D*256)
):: S=256*A(K-1)+A(K):: K=K+
1 :: IF K-1=J THEN ST=1
230 D=D/64
240 L=L+1
250 GOTO 170
260 DATA 6,148,180,136,17,16
7,15,20,2,159,191,10,2,148,1
80,20,2,159,191,100,0,17,0,9
99
```



### PIRATE ISLAND ADVENTURE

Take Rum, Take Sneakers, Take Crackers, Climb Stairs, Pull Bookcase, Enter Passage, Go East, Take Torch, Take Bag, Say Yoho, Say Yoho, Open Bag, Drop Bag, Get Matches, Go East, Enter Shack, Drop Rum (Pirate SHOULD take rum and 'Scuttle off chortling', although sometimes he doesn't. If that is the case, then pick the rum back up, drink some (read the screen comment!), then drop it again until the pirate takes it...). Get Chest, Drop Crackers, Go West, Go East, Climb Hill, Drop Chest, Light Torch, Drop Book, Enter Crack, Enter Shed, Get Hammer, Go North, Enter Crack, Unlight Torch, Drop Torch, Get Book, Go Down, Go West, Go West, Say Yoho, Enter Window, Go Down, Get Nails (they are not shown on screen), Get Rug, Drop Rug, Get Keys, Climb Stairs, Enter Passage, Go East, Say Yoho, Say Yoho.

Drop Nails, Drop Hammer, Drop Keys, Say Yoho, Enter Window, Enter Passage, Go East, Get Bottle, Say Yoho, Say Yoho, Drop Book, Drop Bottle, Go East, Go East, Climb Hill, Get Torch, Light Torch, Enter Crack, Enter Shed, Get Wings, Go North, Enter Crack, Unlight Torch, Get Chest, Go Down, Go West, Go West, Drop Torch, Get Keys, Drop Chest.

Unlock Chest, Look Chest, Get Plans, Look Chest, Get Map, Drop Plans, Drop Map, Drop Keys, Get Bottle, Enter Lagoon, Go North, Get Water, Get Fish, Go South, Go East, Drop Wings, Get Torch, Get Keys, Go East, Go East, Light Torch, Enter Cave, Go Down, Drop Fish, Drop Bottle.

Unlock Door, Enter Hall, Go East, Drop Matches, Enter Shed, Get Shovel, Go North, Get Sails, Get Lumber, Go West, Enter Pit, Go Up, Go West, Go West, Go West, Unlight Torch, Drop Torch, Drop Lumber, Drop Sails, Enter Lagoon, Dig Anchor (you must do this when the tide is OUT, otherwise you will drown. I suggest you wear your water wings just in case you enter the lagoon when the tide is in. If you happen to enter when the tide is in just type WAIT several times....), Get Anchor, Go East, Drop Anchor.

Build Boat, Get Book, Say Yoho, Enter Window, Enter Passage, Go East, Wake Pirate, Say Yoho, Say Yoho, Drop Book, Get Map, Go East, Enter Shack, Get Crackers, Get Parrot, Go West, Go West, Enter Ship, Set Sail (this time the tide must be IN. So if it isn't, just type WAIT several times until the tide does come in).

Enter Shore, Go South, Go East, Enter Monastery, Drop Parrot, Drop Crackers, Get DOUBLOONS, Go West, Go 30, Dig, Get Box, Go West, Go North, Drop Map, Dig, Get Bottles, Drop Bottles (again, the pirate MUST take the bottles.) Get Map, Go South, Wake Pirate, Go North, Enter Ship, Set Sail, Enter Shore, Drop Keys, Drop Shovel, Get Book, Get Hammer, Open Box, Drop Box, Get STAMPS, Say Yoho, Enter Window, Go Down, Drop Stamps, Drop Doubloons, SCORE and YOU WIN !!!!

Corrected solution by David Duncan 15 Inglewood Close, Darlington, Co. Durham.  
Postcode: DL1 2TX.



## EXTENDED BASIC TUTOR 3

by Tony McGovern. PART THREE.

Newcastle TI99ers Sydney Australia.

Our next example will be a good start on a non-trivial utility program for printing out TI BASIC or XB listings on a 80 column printer in two side by side columns which preserve the normal screen listing format. If you just LIST "RS232.BA=..." then the computer sends it out in DISPLAY/VARIABLE 80 format and it is up to you to tell the printer how to handle it. Something approaching screen image format is only obtained (with extra paper consumption) with the printer margins set way in. 80-col printout beats none at all by miles but let's try to be fancier. If you don't have disk or printer then this lesson won't be of immediate use, but will still be a good example to work through as a programming exercise. We might as well do something useful.

First we figure out what needs to be done, and work out a set of procedures that can be CALLED as needed. The program will do only the minimum necessary to do the job properly. Bells and whistles can be added later. In one or two places we shall make provision for adding extras (bells and whistles have nothing on speech) by dummy subprograms which can be filled in later. For a good discussion of the use of such "stubs" see the excellent book by R. Mateosian, "Inside Basic Games". The detailed coding examples in this book are in Apple or Trash-80 Basics, but Mateosian develops ideas in a form much more in tune with a TI XB subprogram realisation than with these less capable Basics.

So let's start designing our program by deciding what we want it to do. We want the output nicely formatted on the page with top and bottom margins, in 2 columns each in screen image (28 char/line) format. More columns (assuming the output device will handle them) are no problem -- once you can count to 2 then 3 is easy. Lines of Basic are not to be split from from one column to the next or from one page to the next.

Some things commonly encountered in printed listings, such as indenting of FOR-NEXT loops don't fit at all well with the multi-statement lines of XB (but might with TI Basic listings) so will not even be thought about here. On the other hand insertion of spaces before REM or SUB statements greatly improves the readability of XB listings, without doing violence to the idea of being screen list compatible. Page numbering is no big deal to add (a console only XB program can fill 6 pages).

At the other end of the business the LISTING to be printed is assumed taken from a disk file such as DSK1.LIST where it has been written by LIST "DSK1.LIST". A trivial difficulty easily taken care of is the blank first record written by LIST. The real problem is that LIST doesn't care about preserving XB lines as distinct entities. Each XB line starts out as a separate print record and if it is less than 80 characters long stays in one piece.

XB lines can easily extend into 2 print records and more (Basic lines much less frequently), but LIST places no markers to show which print records contain the start of XB lines. So if we are going to meet our specification that XB lines be treated exactly as in a screen list then something more subtle than a simple LINPUT is needed. There's one of our most important building blocks identified --- SUB BASICLINE(...).

Any utility program needs title and advice screens so there's SUB TITLES to keep all the details from cluttering the main program. The program will also need SUB OPTIONS(...) to handle file and device name entry and print options which might be offered.

Now the real core of the program is the way in which it must assemble a whole page before printing anything because line feed moves ever on. So we need SUB PAGEBUFFER(...) to take the output of BASICLINES, chop it into screen format hunks and decide where these are to be located on the page. Then we need SUB PRINTPAGE(...) to massage the completed pages and ship them off to the printer. That about sums up the sub-programs that are called directly from the main program, and all that is necessary is to figure out the initialisation -- DIMs, default filenames etc etc, and to write the logic for program flow.

Before we start writing any code we should decide what utility sub-programs are to be used by those already defined. As the list is written into columns SUB WRITECOL(...) is a good candidate for repeated use, and SUB WRITEPAR(...) to take a line of BASIC and return it chopped up into 28 character lines to WRITECOL. Since BASICLINE fetches the input records it is the appropriate place to detect End Of File. We might as well use PRINTPAGE to wipe the slate clean before writing a new page.

Let's dress up the input of filenames and Yes/No responses a little as SUB FILENAME(...) and SUB YN(...), with SUB MORE(...) to end it all. Other useful utility sub-programs which will be included are SUB TXTCOL(..) to change display colors in one CALL, SUB KEYCON to carry the burden of "press any key to continue", and SUB DELAY(..) is always handy.

That about finishes the roster of procedures necessary to make up the listing program, and now the detailed coding can start after some thought on the necessary chains of parameter passing. The principle that you should plan your programs from the top down and code them from the bottom up is just as valid in Extended Basic as it is in TI-LOGO or TI-FORTH where the form of the language makes it difficult to do otherwise. Sub-programs make it possible to go the same way in XB with ease. Less capable dialects of Basic make it a lot harder to keep your thoughts organised and your code on the rails.

The actual program will now be listed piece by piece and commented on in detail. The listing has been transferred into this TI-Writer file from a working copy of the program using a more elaborate version. The present program is actually a simplified version of the one originally written, but is powerful enough to do a useful job.

```
100 REM ** SIMPLIST **
110 REM * PRINTER LIST *
120 REM ** FROM DISK **
130 REM -FUNNELWEB FARM-
140 OPTION BASE 1 :: DIM PRLN$(66,2)
150 REM * DEFAULT VALUES *
160 CALL TITLES :: SFIL$="DSK1.LIST" :: PDEV$="RS232.BA=
4800"
170 CALL KEYCON
```

The first part of the main program shown here sets default values and DIMensions the string array PRLN\$ for two columns of 66 lines each. The top and bottom few lines will be left blank so that page format is obtained without sending printer control codes. A 66 line/page, 80 col. printer is assumed.



```

180 REM * NEW FILE ENTRY *
190 CALL OPTIONS(SFIL$,PDEV$):: ENDFILE=0 :: LINPUT #1:N
EWS
200 REM * NEW PAGE ENTRY *
210 CALL PAGEBUFFER(PRLN$(,),ENDFILE)
220 CALL PRINTPAGE(PRLN$(,),PDEV$):: IF ENDFILE=0 THEN 2
10
230 REM * END OR NEXT *
240 CLOSE #1 :: CLOSE #2 :: CALL MORE(NM):: IF NM THEN C
ALL SPEAK("GOODBYE"):: GOTO 250 ELSE 190
250 STOP

```

OPTIONS returns file and device names as entered there, and the remainder of line 190 resets the End of File flag, and throws away the first line of the list-file. At new page entry the page buffer is filled and then printed out repeatedly until it runs out of listing, and then it asks if you are finished. That's all there is to the main program folks. And now to the sub-programs that do all the work.

```

260 SUB TITLES
270 CALL CLEAR :: CALL SCREEN(1):: DISPLAY AT(12,6)BEEP
:"PRINTER LISTING"
280 SUBEND
290 SUB OPTIONS(S$,P$):: DISPLAY ERASE ALL :: CALL TXTCO
L(16,5)
300 CALL FILENAME(1,2,"Edit as needed and ENTER","N?")
310 CALL FILENAME(4,4,"Source file for listing",S$)
320 CALL FILENAME(8,4,"Printer devicename",P$)
330 CALL YN(" Change mind ?","N",22,5,I):: IF NOT(I)THEN
CALL HCHAR(22,1,32,64):: GOTO 300
340 DISPLAY ERASE ALL :: IF S$="" OR P$="" THEN DISPLAY
AT(1,2)BEEP:"NO INPUT/OUTPUT POSSIBLE" :: CALL DELAY(500
):: GOTO 300
350 OPEN #1:S$,DISPLAY ,INPUT ,VARIABLE 80 :: OPEN #2:P$
,DISPLAY ,OUTPUT,VARIABLE 80
360 SUBEND

```

TITLES here is little more than the barest stub, but you can fill that out to your own fancy. OPTIONS takes down the file names, does some checking, and opens the files.

```

370 SUB PAGEBUFFER(PRLN$(,),EFL)
380 REM * NEW COL ENTRY *
390 PLN=6 :: COL=COL+1 :: IF COL>2 THEN COL=0 :: SUBEXIT
ELSE PRINT "":"** Reading column #";COL:"":""
400 REM * NEW PARA INPUT *
410 IF EFL THEN PRINT "":" *":"*** END of FILE ***":" *
":" :: SUBEXIT ELSE CALL BASICLINE(NEW$,EFL):: PRINT NEW
$:""
420 CALL WRITECOL(PLN,COL,PRLN$(,),NEW$)
430 IF NEW$="END of COL" THEN 390 ELSE 410
440 SUBEND

```

The new column entry in PAGEBUFFER resets the line counter PLN to top of page with a margin, increments the column count, and exits back to the main program if the page is full. If not it tells BASICLINE to fetch a new program line and WRITECOL to enter it in the page buffer. If BASICLINE says it has read the last line it exits and lets the main program worry about that, otherwise it gets another Basic line or starts a new column. A stub here, CALL SKIPLINE(NEW\$,SK), could have uses.

```

450 SUB BASICLINE(N$,E)
460 N$="" :: IF NX$="" THEN LINPUT #1:NX$
470 N$=N$&NX$ :: IF LEN(NX$)<80 OR EOF(1)THEN NX$="" ::
E=EOF(1):: SUBEXIT ELSE LINPUT #1:NX$
480 PX=POS(NX$," ",1):: IF PX<2 OR PX>6 THEN 470
490 P=POS(N$," ",1):: IF PX<P THEN 470
500 NR=-1 :: FOR I=1 TO PX-1 :: C=ASC(SEG$(NX$,I,1)):: N
R=NR AND C>47 AND C<58 :: NEXT I :: IF NOT(NR)THEN 470
510 IF SEG$(N$,LEN(N$),1)=" " THEN 470
520 IF VAL(SEG$(NX$,1,PX-1))<VAL(SEG$(N$,1,P-1))THEN 470
530 REM ** CHECK QUOTES
540 NQ,I=0
550 I=POS(N$,CHR$(34),I+1):: IF I THEN NQ=NQ+1 :: GOTO 5
50 ELSE IF NQ<>2*INT(NQ/2)THEN 470
560 SUBEND

```

The procedure BASICLINE which retrieves complete lines of Basic code from the LIST-file is the only part of the program with decision flow complex enough to warrant drawing out a flow diagram beforehand. I am not going to reproduce this here, but you can work out your own and see if it leads to similar code. The problem comes when the procedure has read in a line exactly 80 characters long. Does the next LIST record then represent a continuation of the same line of Basic or is it the start of a new Basic line?

This difficulty can't be ignored if screen list format is to be preserved since 28 into 80 does not go exactly. The procedure provides a cascade of tests each of which checks whether the record being scrutinised should be appended as a continuation of the previous Basic line.

A few more rare cases could be tested for along the lines of 540-550. There is one (that I know of) unlikely case which BASICLINE cannot resolve even in principle. Can you spot it? It does seem to work well already though. The intricate input code is needed since a VARIABLE file can only be read sequentially, and if the battery of tests says that the last record LINPUTed does start a new Basic line, then this must be saved till BASICLINE is called the next time.

Just be thankful for static variables in XB subprograms! You also have to take care not to set off the End of File alarm prematurely.

```

570 SUB WRITECOL(P,C,P$(,),N$):: IF NC THEN P=6 :: NC=0
580 IF P>=57 THEN N$="END of COL" :: NC=-1 :: SUBEXIT
590 CALL WRITEPAR(P,C,P$(,),N$)
600 SUBEND

```

Now that WRITECOL has the line of Basic it sends it off to be formed into a paragraph. This simplified program handles coming to the end of a column in a slightly wasteful way that is very simple to program. A normal XB program line lists at most on 5 screen lines, and no matter how tricky you are in entering longer lines the program has already limited it to a string variable (max length 255 or 10 screen lines) or has crashed with an error.



The simple minded solution is to exit with End of Col message if the proposed starting line for the new paragraph is past a fixed place somewhat short of the end of the column. The value entered, line #57, is a compromise between making the program totally bulletproof or wasting space. A better approach is to print as far as possible, testing each new paragraph to see if it fits, and if not, holding it over for the next column. If you wondered why the string was called NEW\$, then spare a thought for OLD\$ which which vanished without trace during program simplification for tutorial purposes.

```
610 SUB WRITEPAR(P,C,P*(,),N$)
620 P=P+1 :: IF LEN(N$)>28 THEN P*(P,C)=SEG$(N$,1,28)::
N$=SEG$(N$,29,LEN(N$)-28):: GOTO 620 ELSE P*(P,C)=N$ ::
N$=""
630 SUBEND
```

Sub-program WRITEPAR almost was called SALAMI as it slices up NEW\$ and assigns the slices to successive printlines. Once entered line 620 loops on itself recursively until the remaining piece fits on a screen line. It assumes range checking has been done before entry.

```
640 SUB PRINTPAGE(P*(,),D$):: PRINT "":** Page print st
arted"
650 PRINT "":** Assembling printlines:" and printin
g to" :: PRINT "":" " ;D$
660 FOR I=1 TO 65 :: PRINT #2:TAB(9);P*(I,1);TAB(45);P*(
I,2):: P*(I,1),P*(I,2)="" :: NEXT I
670 SUBEND
```

Not much needs be said about PRINTPAGE beyond noting that line 660 formats a single print record from the two column entries and erases the page buffer as it goes.

```
680 SUB YN(A$,B$,R,C,X)
690 DISPLAY AT(R,C)BEEP:A$ (Y/N) "&B$ :: ACCEPT AT(R,C
+LEN(A$)+7)VALIDATE("YN")SIZE(-1)BEEP:A$ :: X=A$=B$ :: R
=R+2 :: SUBEND
700 SUB KEYCON :: DISPLAY AT(24,6)BEEP:"ANY KEY TO PROCE
ED"
710 CALL KEY(3,I,ST):: IF ST=0 THEN 710 ELSE DISPLAY ERA
SE ALL
720 SUBEND
730 SUB FILENAME(R,C,M$,D$)
740 DISPLAY AT(R+1,C):RPT$("- ",LEN(M$)):: DISPLAY AT(R,C
):M$ :: IF D$<>"N?" THEN DISPLAY AT(R+2,C):D$ ELSE SUBEX
IT
750 ACCEPT AT(R+2,C)SIZE(-15)BEEP:D$ :: SUBEND
760 SUB MORE(NM):: DISPLAY ERASE ALL :: CALL TXTCOL(3,12
):: CALL YN("More listings","N",16,2,NM):: SUBEND
770 SUB DELAY(A):: FOR A=1 TO A :: NEXT A :: SUBEND
780 SUB TXTCOL(A,B):: CALL SCREEN(B):: FOR I=0 TO 12 ::
CALL COLOR(I,A,B):: NEXT I :: SUBEND
```

The FILENAME routine writes an underlined heading, DISPLAYs the default response, and ACCEPTs the reply. If it is asked no question, "N?", it expects no answer. The other SUBs just do their job when called. YN acts like input routines familiar in other TI modules.

```
790 SUB SPEAK(A$):: CALL PEEK(-28672,SP):: IF SP=96 TH
EN THEN CALL SAY(A$) ELSE CALL DELAY(5*LEN(A$))
800 SUBEND
```

This is a last little goodie tagged on so that you may add speech prompts to your program where desired. A bald CALL SAY has the annoying behaviour that it seems to take forever in giving up the attempt if no speech synthesizer is attached. Line 800 checks that speech is connected and line 820 substitutes a controlled delay if not. CALL SPEAK("...") can then be inserted anywhere it is wanted in the program.

So there we have it, a worked out example of a non-trivial and useful program that makes essential use of the sub-program facility of XB. It shows that the XB programmer can, with a style that finds natural expression in the language without undue contortions, follow the general principles of "structured programming" without getting hung up in the Swiss straight-jacket so beloved by some proponents.

The program as presented is a cut-down version of the all-singing, all-dancing model, COLIST, which has now grown to >22K and uses 48 subprograms. In all the versions, subprograms have been an essential tool for program development. Now it's time to take retrospective look at what at what we have done and chase a few more subtleties

[Co List, now replete with some machine code, is on sale from your user group disk library.]



## JOY PAINT

(C) Copyright 1986  
Sophisticated. Easy to use.

These graphics and hundreds more **new** available.

Requires: TI 99/4a, 32k, Disk Drive, Joystick, and one of the following: Ed/Asm, Ex-Basic, Mini-Memory, or TI-Writer. \$49.95 Postpaid.

~~DD NOT~~ **Make for FREE Diskette**

PRINTED WITH JOYPAINT

NOW ONLY FROM \$7.00 COMP-ROBINE DO NOT WRITE TO GREAT LAKES!

RAMBLES by Stephen Shaw  
For TI\*MES April 1991

Greetings and welcome once more. Your enquiries are welcome, on any aspect of TI computing (except assembly on which I am an illiterate!), comments on and requests for Rambles are very welcome too. An SAE always helps for a direct response!  
My address, which is the address for the group's disk library, is:  
10 Alstone Road, STOCKPORT, Cheshire, SK4 5AH.



**EXTENDED BASIC BUG:**

I am grateful to Bruce Harrison of Harrison Software for pointing this one out in the December issue of Micropendium- if you have a disk system, and have a disk of XB programs which has a program called LOAD, your system will automatically load the LOAD program from DSK1 when you select Extended Basic. It also kicks RANDOMIZE into touch, not only for the LOAD program but also for Any program that the LOAD program loads and runs. If the computer manages to find a LOAD program, then your RND will awlays provided the same sequence every time you boot up, even if your program has a RANDOMIZE.

```
100 RANDOMIZE
110 FOR T=1 TO 6
120 PRINT INT(RND*100)
130 NEXT T
140 RUN "DSK1.LOAD"
```

and watch the six numbers repeat again and again, in defiance of the RANDOMIZE command.

Thus if you use a LOAD menu program to put in XB programs such as games or graphics or what have you, you may as well forget about RANDOMIZE... except...

```
70 CALL INIT
80 CALL PEEK(-31880,[,])
90 CALL LOAD(-31808,[,])
```



and now try it... see, different numbers....  
( Yes Virginia you can use [ and ] as variable names!)

NOTE that you must omit the CALL INIT if it has already been used for example to load an assembly utility such as The Missing Link else you will destroy the utility! Call Init is also not required if you use the Triton version of XB titled Super Extended Basic on the module label and (c)1987 T PC.

Bruce tells us this fix can be attributed to Harry Wilhelm and dates back a couple of years.

From January MICROpendium, the sad news of the death of JOHN BIRDWELL, age 41. John contributed DSKU (Disk Utilities) to the TI community.

In January I received an ASGARD catalogue, and listed below are the current offerings. Airmail postage is quoted as \$7.50 per order, and credit cards are accepted subject to a 7% surcharge. Quote Mastercard for Access and Visa for the others! Quote credit card expiry date, and item code of item required. --->

This listing is not a recommendation and orders are sent at your risk, but please contact me if you don't receive an order within six weeks, and especially if you don't receive an order and your credit card has been charged!

You are strongly recommended to order through DATABASE, who will of course charge for their services. As an example, ROCK RUNNER at \$12.95 plus \$7.50 airmail plus possible vat at 15% say \$3 =say \$23 plus exchange costs at say three pounds.... Database lists at a mere Ten Pounds, which is not a bad price.

The issue number of a TI\*MES review and the page numbers are quoted as appropriate!

- TRIS MODULE. ITEM E01A. \$19.95 REVIEW #30 p46
  - ROCK RUNNER. Item No: E05. DISK
  - requires EdAs and j/s. \$12.95. Review #31 p47
  - Tournament Solitaire. Item No: E06. Disk.
  - BRAND NEW. 7 solitaire games. \$14.95
  - Waterworks. Item No E07. Disk. Req j/s. BRAND NEW. \$12.95
  - Disk of Dinosaurs. 2 disks. It No G10. \$9.95
  - Disk of Pyrates. 4 disks. It No G!! \$9.95
  - Yet Another Paint Program.
  - For 80 column card owners. Disk. It No G15. \$29.95
  - Page Pro 99. Disk. It No:P01. \$24.95. Review #26 p11.
  - Music Pro. Disk. It No:P15. \$17.95. Review #28 p48
  - QUICK RUN. Disk. It No: U03. \$9.95 Review #24 p61
  - Beyond Video Chess. It No U04. \$9.95 Review #28 p45
  - Pix Pro. It No U06. \$14.95. Review #28 p46
  - Orphan Survival Handbook. Book. It No T03. \$9.95 Review #17 p12
  - Asgard Mouse. Plugs into RS232. Req Disk drive. It No H04. \$49.95
  - Midi Master. Module+disk, req RS232. Item P01. \$44.95
  - High Gravity. Disk. It No E20. \$4.95. Review #20 p23
  - Balloon Wars. Disk. It No E21. \$3.95 Review #30 p46
  - Column Attack. Disk. It No E22. \$3.95
  - Font Writer 2. Disk(specify DISK). It No G16. \$9.95 Review #20 p2
  - Total Filer. Disk. It No P22. \$6.95 Review #22 p50
- This lengthy list is only a selection from a very large range!  
ASGARD SOFTWARE P O BOX 10306 ROCKVILLE MD USA 20849  
(Contact MARTIN BLYTHE at DATABASE for these-  
he recently listed both Rock Runner and Waterworks at just Ten Pounds each).



- I have also received a catalogue from NOTUNG SOFTWARE which lists:
  - THE RING COMPANION. 2 disks. Text & instances (Rackham) re The Ring. \$8
  - SON OF THE DISK OF DINOSAURS. An animation, text & instances. 2 disks. \$12.
  - Ye Cock Ale. Beer bottle labels and recipes. \$7
  - Filib. A database command set for TI Base for video collectors. \$7
  - Fonts and Borders. Vol 1. \$7. Vol 2 \$7. Vol 3:\$7.
  - 1991 Star Trek TNG Calendar. \$10
  - 1991 KBGB Girlie Calendar. \$5.
- Postage on these is \$2.50 per order. Payment must be by US\$ notes or Money Order.  
Notung Software. 7647 McGroarty Street. Tujunga. CA. USA. 91042.

- And another catalogue -must be the season for them- from COMPRODINE, at 1949 Evergreen Avenue, FULLERTON, Ca, USA, 92635.
- Postage is \$4 per order and payment must be by US\$ money order or draft.
- All products are on disk, and where appropriate the TI\*MES issue in which a review appeared is indicated in (brackets).
- Backsteine, a new breakthrough fame (32) at \$10. Living Tomb \$15.
- Warzone(28)..\$10.
- Artist Printshop(28)..\$25. Giant Art Posters(28)..\$15
- Certificate 99 \$7, Cert 99 Companion 1 and 2 at \$7 each.
- Joypaint \$7, Joypaint Pal \$7. Extended Business Graphs \$7
- Border Maker for Artist Printshop \$10. and also
- 30 (thirty) SSSD disks of small graphics (3/4" square) which print out with TI Writer EDITOR.



From New Scientist Christmas issue...

Santa delivers to a very long road. He notices that at one house, when he removes the first digit and then squares the remaining digit he is back to the original number. Here, let's do this one for you:

125.....25 x 25 = 125

What is really quite amazing is that when he gets to the 25th house number which satisfies this rule, he finds it is his own house! What number is his house?

I have taken out a sub to a pretty expensive magazine called RECREATIONAL AND EDUCATIONAL COMPUTING which at 30p per page isn't too cheap- issues are bimonthly or so and you can receive some sample copies by sending US\$15 to:

REC, 909 Violet Terrace, Clarks Summit, PA, USA, 18411.

Here is a puzzle from one issue:

In the land of make believe there is an eccentric jailer (gaoller?) who takes the following action on New Years Day:

At minute 0 all 100 doors are locked shut.

At minute 1 he goes to every cell and opens the door.

At minute 2 he goes to door 2 and if it is open, closes it. If it is shut he opens it. The he goes to door 4 and does the same thing and so on to every SECOND cell.

At minute 3 he begins at cell 3, and reverses the state of its door, carrying on with every third cell (Reversing 3,6,9,12 etc) and so on and so on and so on.

At minute 100 he just reverses cell 100.

Now then... how many prisoners can walk free- eg which cell doors are open?

With reference to the puzzle set on p53 of issue #30, regarding the three hands of a clock...

For ease of reference, refer to the position of ALL three hands in terms of the clock face reading in hours- that is if the second hand is pointing to the hour number 3, we refer to the position of that hand as 3.

The easiest way to approach this problem is to consider firstly just two hands- the hour hand and the minute hand. Instinctively we can feel that these two hands for an angle of 120 degrees (or 4 hours!) about 24 times in 12 hours and then repeat.

When the hour hand moves from 12 to 1, a passage of one hour, the minute hand moves from 12 to 12, moving through 12 hours (hour numbers!). Thus the difference in movement between the two hands is  $12-1=11n$ . Our problem requires that the difference between the hour and minute hands is 4 hours (on the clock face) so we must solve for  $11n=4$ . Because a clock face is circular (you noticed!) there is more than one value of  $n$  we can use.

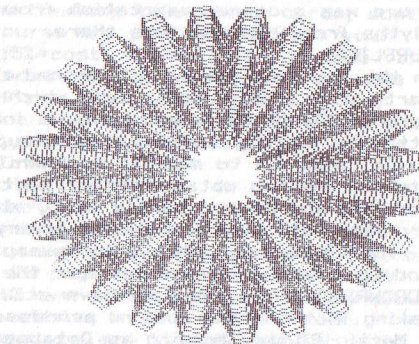
We can now write a program for values of  $n$  from one upwards until 12 hours have elapsed and the numbers repeat. It is necessary to continually reduce the number to keep it within the 12 hour period.

Then, having our 22 values when the hour and minute hands are exactly 4 hours apart we can examine the position of the second hand to see where it lies at these very limited number of times. If it is precisely four hours from each of the hour and minute hand, our problem is solved!

```
100 FOR N=4 TO 96 STEP 4
110 IF N/12=INT(N/12) THEN 180 ! hands coincide
120 HOURS=N/11
130 MINS=HOURS*12
140 IF MINS>12 THEN MINS=MINS-12 :: GOTO 140
150 PRINT HOURS ! note 0.09' is 1/11th
160 PRINT MINS
170 PRINT "====="
180 NEXT N
```

~~~~ and then ~~~~

```
1 REM THREE HANDED CLOCK
2 REM DATA=TIME ELAPSED FOR HOUR AND MINUTE HAND
3 REM S SHAW JAN 1991
4 REM TI99/4A EXTENDED BASIC
5 REM
100 DATA 0.36363636
110 DATA 0.72727272
120 DATA 1.45454545
130 DATA 1.81818181
140 DATA 2.54545454
150 DATA 2.90909090
160 DATA 3.63636363
170 DATA 4.0
180 DATA 4.72727272
190 DATA 5.09090909
200 DATA 5.81818181
210 DATA 6.18181818
220 DATA 6.90909090
230 DATA 7.27272727
240 DATA 8.0
250 DATA 8.36363636
260 DATA 9.09090909
270 DATA 9.45454545
280 DATA 10.18181818
290 DATA 10.54545454
300 DATA 11.27272727
310 DATA 11.63636363
320 DATA 99,99
330 READ HOURS :: PRINT "====="
340 IF HOURS=99 THEN STOP
350 MINS=HOURS*12
360 IF MINS>12 THEN MINS=MINS-12 :: GOTO 360
370 PRINT SEG$(STR$(HOURS),1,6)&"---"&SEG$(STR$(MINS),1,6)
380 SECS=HOURS*60*60
390 IF SECS>1200 THEN SECS=SECS-1200 :: GOTO 390
400 IF SECS>60 THEN SECS=SECS-60 :: GOTO 400
410 PRINT SECS/5 ! CLOCK FACE READOUT IN HRS
420 COUNT=COUNT+1
430 IF INT(COUNT/8)=COUNT/8 THEN 440 ELSE 450
440 FOR PAUSE=1 TO 4000 :: NEXT PAUSE
450 GOTO 330
460 END
```



This is a pretty fast way of sorting this problem out. The ticking second hand? Well expressed in terms of the hour numbers, a second is 0.2 hr, so if one of the 22 solutions comes within 0.2 of satisfying our needs, maybe ticking does have some significance....?

Much harder query: If the perfect solution is not possible, can we find the time that the hands are closest to our requirement? For this purpose, differences are added regardless of their sign-

```
We require: 120      120      120
We get say: 119      118      123
Error       1      + 2      + 3      = error 6.
```

I do not have an answer to this one and look forward to hearing from you.

My thanks to Walter Allan for correspondence received on this one!



CUFFLEY...

The meeting at CUFFLEY went well in January, although not as well attended by SOUTHERN users as could have been hoped. It was mentioned that Cuffley being actually North of London was to be considered a Northern show... hmmm. Those who did attend had a good chat amongst themselves and there were some real bargains to be picked up for those who wanted them.

Cuffley was attended by about 38 TI users - (The Sydney group in Australia, now starting its tenth year, also had 38 attending its AGM in December 90...!).

There was good representation from the DORTIG Dorset group, and Martin Blythe from Database was there.

SUPPLIES...

In the last few months I have had requests for software items which Martin can supply from stock, which puzzles me a little. I am not in any form of TI business, and have no intentions of semi-commercial supply. Attempts to help members with group purchases have met with far too little response to make it worthwhile. Martin Blythe however has good stocks, and can obtain current software from Databiotics (latest modules) and Asgard (mainly disk), although he suffers the same supply problems with these people as everyone else.

Martin had at Cuffley, for example, BEYOND WORDWRITER, the latest module based wordprocessor for the cassette user - after our review of WORDWRITER + recently we have still had an enquiry from a cassette user asking IF there was a word processor for cassette users!

Martin Blythe, trading as Database, is at:

Bronfa, Llanybydder, Dyfed, SA40 9UB. Tel 0570 481079.

And Edward Shaw (see cover) has a good stock of modules available second hand. Edward or Martin can (between them) obtain virtually anything you may want!

In extremis- and it really shouldn't be too often! - if you cannot find what you want, drop me a line, but note I do not have stocks, and you must expect severe delays! I am not competing with Edward and Martin- merely supplementing them, as I do have US contacts who can assist with odd items.

I have also been taken to task for not pointing out in very blunt language that you really can still obtain a very very large number of items of software on module or disk, and almost every module ever made is still available from somewhere. Before you are tempted to look at another brand of computer, consider what software you need. There is a very good chance you can obtain an adequate working system for your TI, probably at a much lower cost!

MICROpendium has been mentioned in almost every issue. They review new items and tell you where to get them. There is also regular advertising of older material especially TI modules at low low prices.

The cost of modules in the States can be very low, and this does make the overhead costs of such long distance purchases seem very high. For example, if you buy a US\$ International Money Order from the Royal Bank of Scotland there is a commission charge (as at Feb 1991) of Three Pounds. -----> more

NOT a lot for a \$200 order but rather steep for a US\$5 order! It is the same piece of paper no matter what value is written on it, with all the same work to do! There is a similar problem with postage- a single disk MAY cost only \$5 by airmail, but you can usually expect to pay \$8/\$10 postage on most orders. A large package by SEAMAIL may be \$30 or more... so don't forget MARTIN, who does all the work for you as well as covering overhead costs - which include storage and buying his food!

If you wish to order directly, it is not difficult, you buy an IMO or you quote your credit card number (TexComp take credit cards, most other suppliers are too small). In due course the postman delivers your package, and MAY ask you to pay 15%+costs on the total invoice value before handing the package over (most packages for low values are allowed through without charge). If you have difficulties with any supplier, please write and tell me and we can spread the word and maybe even offer some assistance. Note you may have to wait two months for a package!

Some time ago I printed a membership application form for the Sydney group, which was utterly ignored by everyone. They now have a second UK member to join me! They publish a 24 page (size A4 but still small print!) magazine 11 times a year and overseas subs are A\$45 by slow seamail or A\$60 by airmail. Their articles include machine code, hardware, and Forth as well as some you are already familiar with. Their address is:

TISHUG (Australia) Ltd., P O Box 214, Redfern, NSW, AUSTRALIA, 2016 and another plug for

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

Seamail US\$30, Airmail US\$42, and MICROpendium take credit cards! Quote card expiry date and say MASTERCARD for Access and VISA for members of that group as well as the card number of course. Monthly, 40/48 pages, now in 8th year.

MICROpendium gave TI\*MES a good review in their January 1991 issue!

Also in their January issue a note that COMPRODINE was not only still bringing out new goodies but had on board programs from Great Lakes Software such as JoyPaint and Certificate 99 at very low prices.

Some welcome comments at last on the graphics routines- after a long silence it appears there are enough of you out there to keep some graphics programs coming! Thanks for writing everyone, it is appreciated.

TI WRITER et al...

Our friends in the East Anglia group have passed on some good TI Writer tips, culled from other newsletters, notably...

... CTRL 8 places a carriage return and adds a line. It is faster to type ctrl 8 then fctn 3 than it is to type say (ctrl u, fctn r, ctrl u) should you need to insert carriage returns in already written text.

... when you enter LF Funnlweb will place your cursor after DSK- that is, on the disk number. If you type LF SPACE SPACE enter, the cursor will appear after the dot eg after DSK1. or whatever. Backspace has a similar effect to space- wherever the cursor is when you press enter, it will be in a similar position on the device/filename.



## BETTER BASIC PROGRAMS...

A simple puzzle:

I somehow manage to find a pound (100 pence) and wish to purchase items which are priced at 14p, 17p, 22p, and 39p. In what combinations may I purchase these items so that I exactly spend my pound?

We can quickly get our calculators out, work out the maximum number of each I can buy for a pound, and come up with code like this...

```
100 FOR A=0 TO 7
110 FOR B=0 TO 5
120 FOR C=0 TO 4
130 FOR D=0 TO 2
140 IF 14*A+17*B+22*C+39*D=100 THEN PRINT A;B;C;D
150 NEXT D
160 NEXT C
170 NEXT B
180 NEXT A
```

Can you follow the reasoning behind this code? This little program takes about 25.4 seconds to find the four possible answers.

We know our console is not the fastest in the world, so can we speed up this work? The first improvement is interesting- curious too. This simple amendment:

```
100 FOR D=0 TO 2
110 FOR C=0 TO 4
120 FOR B=0 TO 5
130 FOR A=0 TO 7
140 IF 14*A+17*B+22*C+39*D=100 THEN PRINT A;B;C;D
150 NEXT A
160 NEXT B
170 NEXT C
180 NEXT D
```

produces nearly a 10% speed-up, and runs in about 23.2 seconds. Most odd.

We can however speed things up a great deal more by applying a little thought to the problem. If we buy a 39p item we have only 61p left which at most allows us to buy 3 items at 22p- in this case the C loop does not need to test for a value of 4. If we buy two 39p items we have only 22p left and can only buy one 22p item. The following code reduces the number of steps in the loop by taking account of this sort of limitation. The first line is slightly wasteful but will make the flow a little clearer:

```
100 FOR D=0 TO 100/39 ! think about it
110 FOR C=0 TO (100-39*D)/22
120 FOR B=0 TO (100-39*D-22*C)/17
130 A=INT((100-39*D-22*C-17*B)/14)
140 IF 14*A+17*B+22*C+39*D=100 THEN PRINT A;B;C;D
150 NEXT B
160 NEXT C
170 NEXT D
```

----->MORE

This code would you believe gives us the same four answers in just 2.82 seconds or thereabouts.

However we can still speed things up a little. Look at lines 130 and 140. Provided A does have an integer value our problem has a solution, and we can simplify this part like this:

```
130 A=(100-39*D-22*C-17*B)/14
140 IF A=INT(A) THEN PRINT A;B;C;D
```

This improves the speed to around 2.24 seconds- about the time we saved with our first improvement. Just a little thought before coding a problem can save much un-needed work by the computer and speed things up to a degree which is greater than the difference between a 4Mhz computer and a 32MHz computer -all other things being equal, which they never are!

Thanks to Recreational & Educational Computing Newsletter Vol 1 #4. Sample copies \$15, from 909 Violet Terrace, Clarks Summit, PA, USA, 18411

=====

A WARM WELCOME to our new members STATESIDE who are very welcome to drop me a line should they wish to share their knowledge and experience and goodies.... and if there is ANY U.K. product you want but cannot get let me know! (eg Red Dwarf paperback!). Every member is of course welcome to write and say what they like or dislike about Rambles, to ask questions and so on. If a personal reply is needed an SAE (or dollar bill) is always useful.

=====

TI\*MES #31 update... re page 26...

NO response from Paul Scheidemantle, despite sending a couple of IRCs. Could be a postal error, or I got his address wrong, or he don't want to know...???

MICROpendium Jan 91 issue indicates at least some CorComp owners are not finding it easy to get IDT to repair failed CorComp cards.

## TENTH ANNIVERSARY...

Easter 1981 I keyed in my first program on a NTSC TI99/4 borrowed from Texas Instruments (those were the days. Try cadging an Amiga from ukw!) and subsequently received an invoice from TI dated June 1981 for a fully expanded TI99/4 side-car system -the UK PAL version was then available, but with sound from a little loudspeaker in the console as TI had not then worked out how to rejig the modulator, made for European PAL, to work properly on UK PAL. They never did get it ENTIRELY right! despite many changes of modulator design.

I became a founder member of the UK User Group, and by January 1982 had a program published in Computer and Video Games Magazine (still going but no listings these days). I was also a founder subscriber to 99er Magazine, and was soon making my first purchase of modules from the USA, they were so much cheaper there than TI was selling them for here!

As the year progressed I wished to share the software I was buying, and did the proper thing by establishing licensing arrangements and sold programs as Stainless Software, and so we went on... upgrading to a PEB and 99/4A in due course, continuing to write for anyone who wanted my material.

----> continued ---->



In due course, the collected writings were assembled into a book- I was asked to write it by a commercial publisher, who made the first payment before a word was written. WRITTEN ENTIRELY WITH TI WRITER, it brought in a little money, all of which was spent on new programs and books!

I love games, especially strategic games. I enjoy programming. And the TI has so many superb programs, so many languages to explore (still discovering new things about Basic!). I have a STACK of computer books full of programming ideas, and hundreds of programs to write, if I can ever find the time (anyone need programming ideas, mainly strategy or puzzle games?).

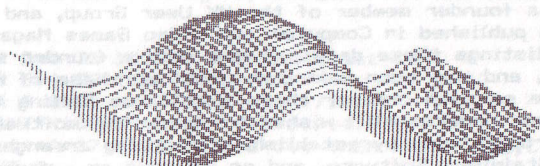
My TI has at this moment clocked up 11,044 hours, and has outlived the tv I bought to use with it! I am quite happy to go on using it for another ten years, even longer maybe, only an irreparable breakdown is going to separate us!

REVIEW: THE RING COMPANION. 2 disks. NOTUNG SOFTWARE. US\$8+2.50 POST. This two disk set is a companion to the music disks by Ken Gilliland, "RICHARD WAGNERS GREATEST HITS" and "RICHARD WAGNERS DER RING DES NIBELUNGEN". We managed no less than three versions of The Ring on TV over Christmas/New Year, so it may be a little more familiar to you than last year!

This disk set contains:  
A huge picture of The Ring, with a dwarf and Rheinmaiden, composed of eight TI artist pics-formatted for printing with TI Artist Plus! but any other \_P printer will do. Plus a pic of Richie Wagner. Plus 12 pictures and instances of Ring cartoons by Arthur Rackham. Plus some text about The Ring (roots, characters, Wagner). Plus a program to demonstrate 30 "leitmotifs". Quote a package for lovers of Wagner.

REVIEW: SON OF THE DISK OF DINOSAURS. 2 disks. NOTUNG SOFTWARE, \$12+2.50 POST.

A follow up to the Disk of Dinosaurs published by Asgard, this disk set contains a further landscape to place dinosaurs into, and 25 TI artist instances of individual dinosaurs, which Notung Software used to very good effect in a very unusual Christmas card! There is a further 62 frame cartoon adventure with Thug, but this time using Comic Show 4.0, so you cannot slow down the rather rapid movement! Plus text on the dinosaurs, and a quiz which allows you to build a dinosaur by answering correctly ten questions. THE source for a TI Artist instance of a Pachycephalosaurus!



DISK LIBRARY NEW ADDITIONS

Stephen Shaw. 10 Alstone Road, STOCKPORT, Cheshire, SK4 5AH

To receive a printed copy of "NEW/DISKS" file , please send an SAE. The disk library catalogue comprises well over a thousand disks, and the text extends over three disks- sorry, print outs not available, please send three disks and return postage for an up to date copy!!!

NEW DISKS ADDED since December 26th 1990:

-TI BASE TUTOR by Martin Smoley is now updated to 21 articles spread over nine disks. These disks contain articles on TI Base written by Marty on TI Base since July 1988 up to late 1990, and are complete with database files and command files. For your convenience the disk set is available split up, but please note that Marty developed particular uses over several months and you may need older disks to fully make sense of the later ones!

>>TI BASE TUTOR A+B (TWO DISKS): July 88 to Dec 88. Version 1 to 1.02. Getting started, SETUP, CREATE, Mailing label, using two databaes together, convert IF40 to DV80, changing field size, using old TI-Mail data.

>>TI BASE TUTOR C+D (TWO DISKS): Jan 89 to May 89. Up to Vn 2.01. Club type record system. X type fields. Graphics database! Part one of printing labels with graphics and text.

>>TI BASE TUTOR E+F (TWO DISKS): June 89 to Jan 90. TI Artist instances to TI Base data (with commented assembly source code); DATE type; TI Base to TI Writer mailmerge format; chequebook database; ;FOR clause;

>>TI BASE TUTOR G+H (TWO DISKS): From Feb 90 to May 90. To Version 3.0. Printing labels across and printing a label and a letter at the same time to two different printers(!). A stock program showing use of several databases together. Macros.

>>TI BASE TUTOR I. (One disk only). June and July 1990. (Last). INSTALL. Inventory control.

>MANIPULATOR 4 Vn 1.1 by John Seager is a utility for ExBas users, which allows you to extract portions from an XB program; to delete or move blocks of lines, to resequence a block of lines, to check that all GOTDs actually go somewhere, and to locate all lines referring to a specific variable. This is a machine code utility, quickly acting on a program in memory, NOT using merge format disk files. Resequene SEEMS to be faster than the on-board console utility?!!!

>TIGERCUB GOSPEL DISK from Jim Peterson. Absolutely lots of gospel songs, with words on screen, and quite a variety of XB programming techniques used to produce different effects, well worth listing these programs as well as listening to them.

>RLE28 is now full and has: Baby#1, Baby#2, Basket(Easter), Battle, Cairofont(little pics), Curve, Goggins1, Hedgehog, Hunter, Mermaid, Pets01, Pets02, Phone1, Postman Pat.

>RLE29 (202 sectors used): Deathhead, Flower, House, Middle, and two extraordinary pictures in colour, ported from Myart, Dreamhouse and City, with only minimal colour-bleed. Pretty good.

TIGERCUB SOFTWARE COLLECTIONS- as from Jim, the Tigercub software is only on these disks in the library, but some of the other files can be found duplicated on other library disk...



>TIGERCUB BRAIN GAMES. 3 coin weighing puzzles; Reverso, Bassackwards, Pick Up Sticks (Nim), Vega, Match a Patch, 3d tic tac to, Acmehotel (find the bomb), colorsq, deliver the cake adventure, L-Game, Mastermind and Othello.

>TIGERCUB BRAIN TEASERS. Missionaries and Cannibals, Election, 4x4 Puzzle, Tower of Hanoi, 3 Bucket Puzzle, Old Timer Puzzle, Preachers Lawyers and Used Car Salesmen, 15 Puzzle, Hexapawn, Lastrobot, Mousemaze, Queen, Rotate, Shootstars.

>TIGERCUB BRAIN BUSTERS. Can of Worms (Nim), Rithmatik, Division Cryptogram, Nimbo, Glunk, 100% (from Belgium), Addition Magic, Arithmagraph (Mike O Reagan), Bagels, Digitron, Fourinrow, Goinghome, Gomoku, IQMath, Math Puzzle, Mawari, Multiplication Madness, One Check, One to Five, Othello (different version), Sphinx (from Belgium).

>TIGERCUBS BEST. Alley Craps, Whitewater Run, Scrum, Haunted Graveyard, Mechanical Aptitude Test, Fourinrow, Highjump (from Italy), Kroaker, Leaper, Left/Right, Mazzo (from the author of Diablo, very much easier this one!), and three machine code games, the Mad Bomber, I'm Lost, and Cat and Mouse.

>TIGERCUB KALEIDOSCOPIES AND DISPLAYS. Million Mirages, Keleido vision, Jewels on Velvet, Multivision, Optical Illusion, 10000 Sights, Andrew, Aurora, BoxArt, Colorburst, Colorsquare, Colour Vision, Columbia, Escher, Eternity, Fascination, Hypnosis, Kalsquares, Kalvision, Patches, QuickKal, Snow, Spritedemo.

>NEW TESTAMENT. Three double sided ARCHIVED disks! Counts as six disks. Don't be afraid to ask for other formats (eg unarchived or SSSD!

>JEUX 4. More games from France! Bowling, Chasseur, CuiCui, Jackpot, Mission Impossible, Peche (excellent fishing game), Reussite, Solitaire.

>TIMES TEXT DISK 4 now added to collection. Double sided disk, archived. Text from TI\*MES Sept and Dec 90.

>MCH2... 135 sectors full of Milton Bradley module HONEY HUNT with an unusual XB loader that's worth listing!

>added to UTIL26 is ARTIST ENLARGER Vn 2.5(1991) by Howard Uman, which can make instances AND fonts half or double size, even, if you wish, acting in one direction only, for super high or super wide fonts. Large XB program.

That's about it for now. Members not using the disk library, do write and tell me why! We have oodles of good things to share with you.

Stephen.  
=====

REVIEW- JIFFY FLYER -DISK - US\$10 from Comproline + \$4 post.

Rather an old program but I don't seem to have reviewed it yet. It is a remarkably simple program!  
It enables you to quickly prepare a single sheet poster, using a border (several are part of the program); a large type heading, a small type announcement over several lines (several fonts are part of the program), and either an internal border or four corners made up of the same graphic, which you load from disk- the small pictures are to be in CSGD format, several are supplied, and the Disk Library has many more.

And that really is that!  
The format is pretty fixed, which makes for a fairly straight forward and easy to use program- the more powerful a program the less easy it is to use (The Printers Apprentice is about ten times harder to use than this program!). A very similar utility can be found in ARTIST PRINTSHOP from the very same publisher, with extra facilities, but priced at \$25. Artist Printshop is the more versatile but if your budget is tight, this little utility, at less than half the price, may well suit.

-----  
EDUCATIONAL LEVELS from Unisource Catalogue.

American Grade One has been taken to be 5 yrs old. Assume age range may be -1 to +2 years from those stated as target ranges:

- |                               |                       |                               |                   |
|-------------------------------|-----------------------|-------------------------------|-------------------|
| 1. EARLY READING.             | 2. READING RAINBOWS.  | 3. READING FUN.               | 4. READING ON.    |
| 5. READING ROUNDUP.           | 6. READING TRAIL.     | 7. READING RALLY.             | 8. READING POWER. |
| 9. READING FLIGHT.            | 10. READING WONDERS.  | 11. ADDITION & SUBTRACTION 1. |                   |
| 12. ADDITION & SUBTRACTION 2. |                       | 13. ADDITION & SUBTRACTION 3. |                   |
| 14. DIVISION 1.               | 15. MULTIPLICATION 1. | 16. FRACTIONS 1.              | 17. NUMERATION 1  |
| 18. NUMERATION 2              | 19. NUMBER BOWLING    | 20. FROG JUMP.                | 21. SPACE JOURNEY |
| 22. PYRAMID PUZZLER           | 23. STAR MAZE         | 24. PICTURE PARTS.            |                   |

The following modules are by Milliken and are not to be confused with the very similarly titled Scott Foresman titles above:

- |                                  |                                  |                         |              |
|----------------------------------|----------------------------------|-------------------------|--------------|
| 25. ADDITION.                    | 26. SUBTRACTION.                 | 27. MULTIPLICATION      | 28. DIVISION |
| 29. INTEGERS.                    | 30. FRACTIONAL NUMBERS.          | 31. DECIMALS            |              |
| 32. PERCENTS                     | 33. LAWS OF ARITHMETIC.          | 34. EQUATIONS.          |              |
| 35. MEASUREMENT FORMULAS.        | 36. NUMBER READINESS.            |                         |              |
| 37. ALLIGATOR MIX.               | 38. ALIEN ADDITION               | 39. DEMOLITION DIVISION |              |
| 40. DRAGON MIX                   | 41. MINUS MISSION                | 42. MINUS MISSION       |              |
| 43. METEOR MULTIPLICATION.       | 44. WORD RADAR                   |                         |              |
| 45. WORD INVASION                | 46. COMPUTER MATH GAMES 2.       |                         |              |
| 47. COMPUTER MATH GAMES 6.       |                                  |                         |              |
| 48. SCHOLASTIC SPELLING LEVEL 3. | 49. SCHOLASTIC SPELLING LEVEL 4  |                         |              |
| 50. SCHOLASTIC SPELLING LEVEL 5. | 51. SCHOLASTIC SPELLING LEVEL 6  |                         |              |
| 52. EARLY LEARNING FUN.          | 53. BEGINNING GRAMMAR            |                         |              |
| 54. NUMBER MAGIC                 | 55. TI LOGO 2 (32K RAM REQUIRED) |                         |              |
| 56. EARLY LOGO LEARNING FUN.     | 57. STORY MACHINE.               |                         |              |
| 58. FACEMAKER                    |                                  |                         |              |

Yes, 58 educational modules to choose from! Match that!  
Item numbers 37 to 45 and No 58 carry no age recommendations. 58 is probably suitable for 4 up while 37 to 45 require an initial interest in video game playing and are probably for 6 up.

- to 5: 1,2,3,...11,...17,24,25,26,27,...36,46,52,53,55,56,57  
6:.....3,4,...11,12,13,17,18,24,26,27,...46,47,52,54,55,56,57  
7:.....3,4,5,6,12,13,14,15,16,18,24,27,28,46,47,48,54,55,56,57  
8:.....4,5,6,7,8,13,14,15,16,18,22,23,24,27,28,30,33,46,47,49,54,55,56,57  
9:.....5,6,7,8,9,10,14,16,20,22,23,27,28,29,30,31,32,33,34,35,46,47,50,55  
10:.....7,8,9,10.....20,22,23,27,28,29,30,31,32,33,34,35,46,47,51  
11 and up:.....9,10...19,20,21,22,23,27,28,29,30,31,32,33,34,35,46,47.

=====

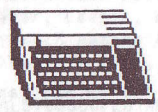




# ROLL OF HONOUR

THE 1999/4A HOME COMPUTER  
FROM TEXAS INSTRUMENTS  
UK MODEX 1981-1991  
STILL GOING STRONG

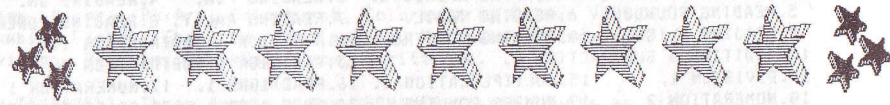
SUPPORT THE UK USER GROUP



TJUGUK

READ T\*MES!

Margaret Thatcher  
Margaret Thatcher






QUARTERLY MAGAZINE  
 SOFTWARE LIBRARIES  
 MODULE DISK TAPES  
 ASSISTANCE  
 UK SUB 12 50 PA  
 EUROPE 15 00 PA

HAPPY  
 T I M E S

UK USER GROUP  
 1983 1991  
 STILL GOING STRONG

**T I 9 9 4 A**

R E A D  
 T I M E S




REVIEW: BOOK: THE TURING OMNIBUS: 61 Excursions in Computer Science.  
 By A K Dewdney. Hardback.  
 415 pages. W H FREEMAN AND CO. ISBN 0 7167 8154 9

We have previously reviewed the earlier book, The Armchair Universe, and this one follows on from that, but this time the articles are a little heavier and we do not have the simple algorithms to give us fast graphics output. Rather a set of articles loosely connected to computing, the reading and understanding of which just may help you to resolve computing problems faster or easier. Something of a recreational math book. Some of the chapter headings include: Error correcting codes; karnaugh maps; the fastest sorts of sorts; the chaitin-kolmogoroff theory.... This book is intended as a "semi popular compendium of computer science" rather like a touring bus (pun intended) visiting selected landmarks of computer science. You have been warned!

REVIEW: BOOK: THE MAGIC MACHINE: A HANDBOOK OF COMPUTER SORcery.  
 BY A K DEWDNEY. paperback. 357 pages.  
 W H Freeman. ISBN 0 7167 2144 9

This 1990 book is a little easier to take, it has a few easy to translate algorithms to turn into programs, and in theory 16 out of 28 chapters could give rise to a program of some sort, although some could be hard to sort out! Once more lots of general reading to improve your approach to problems! Including even a mention of transactional analysis; other headings include: balls in boxes; trains of thought (some classic recreational math puzzles); palmiters protozo; catching biomorphs. THIS book is a collection of recreational material that was previously published in Scientific American under the heading "computer recreations".

REVIEW- DISK-GAME- BACKSTEINE by Quinton Tormanen. US\$10 plus \$4 post from COMPRODINE.  
 1949 Evergreen Avenue, Fullerton, Ca, USA, 92635.

Well that it the worst thing about this offering out of the way anyway, with a high score of zero out of ten for the awful name (unless you are German, or speak the language, when you will know the title is BRICKS). This program was written in an English speaking country (sort of) for sale in same. No excuse.

It is a Breakthrough type program- where you move a bat at screen bottom and a bouncing ball batters away at a wall of bricks, and if the ball hits the floor it is lost.

Not original. However the IMPLEMENTATION is absolutely superb, well up to modern arcade standards, with the ball and some bricks throwing shadows on a patterned background. Bricks take one two or three hits to knock out, and may drop one of six different bonus prizes, which make your bat bigger or affect the ball in some way. There are also some bricks which cannot be destroyed.

The game comes with an assortment of fifty screens (made up of different patterns of bricks) and there is an editor to make your own up. Game play is with joystick one for one player. Editing can be with joystick one, or with more control (eg slower) with joystick 2 or even ESDX keys.



The graphics and implementation of this game are exceptional. You may not be playing it none stop for the next year, but it is an excellent game to while away an hour or so every now and then, and contemplate just what our ancient machine is capable of... and was capable of in 1980. Why didn't TI release modules of this quality! Highly recommended.

REVIEW- EXTENDED BUSINESS GRAPHS from Comproline \$7 plus \$4 post. This program originated with Great Lakes Software. It is quite old, and rather shows its age. Basically you key in your data, and have a few graphing options. Graphs can be printed out in a single format, and CANNOT BE SAVED TO DISK! in any format.

A little restricted perhaps, but there are not many graphing programs available, and this new price isnt bad. I will supply illustrations of ALL the types of graph it is capable of! which illustrate what it can do better than mere words.

REVIEW- PICTURE IT -disk from COMPRODINE for US\$10.

A graphics utility program with several options, none of them very fast, but quite useful for all that.

Dating back to 1987, this program is by Rodger Merritt, who runs Comproline. The program disk has several utilities...

BANNERS... to print instances up to 12 lines high and/or text in a specially supplied 8 inch high font, sideways, much enlarged. You get to choose which ASCII character represent a "pixel" so there is plenty of room to vary contrast. Banners seen at Romley and Chester AGMs were prepared with this. Instances can be text prepared with any TI Artist font of course!

INSTANCES... can transfer a TI Artist instance into an XB program, either as redefined characters or as defined sprites. The output file is a merge format DV163 file.

TI WRITER... and you can save instances to disk in a format which prints out via the TI Writer FORMATTER (uses a lot of >TL's).

If you need the program, you need it!

REVIEW- CERTIFICATE 99- disk from Comproline, \$7 plus \$4 post.

This is again a rather old program, which allows you to fill an A4 sheet of paper with a computer produced certificate, with a choice of borders, a choice of graphics, a choice of fonts, and a choice of signatures (perhaps Mrs Thatcher is no longer appropriate?). There are support disks available from both Comproline AND Notung with extra fonts etc.

I shall try to reduce a certificate for illustration purposes, but we need rather a large reduction so it may not work out too well...

If you need to produce certificates, this is the most directly applicable TI program, and pretty inexpensive too!

100gsm natural parchment looking paper IS available in clean edge tractor feed paper (tear off the sprockets if you do not have tractor feed).

100 sheets £6.39 plus carriage

250 sheets £12.29 plus carriage

2000 sheets £76.99 plus carriage

100 sheets with ribbons and sticky seals (most authentic!) £10.09

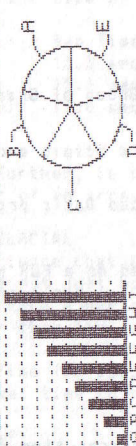
from: Print Rite, 18 Woburn Close, Northburn Park, CRAMLINGTON,

Northumberland, NE23 9QP. Tel 0670 737704. Credit cards ok.

carriage £2.30 per item or telephone for quote.

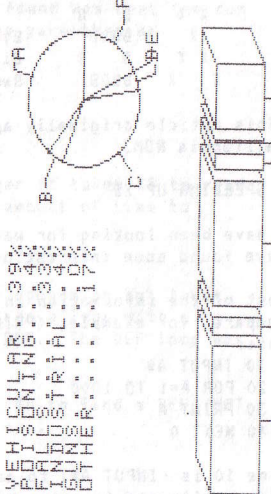
Paper details extracted from ad in Computer Shopper, March 1991.

**EXTENDED BUSINESS GRAPHS**

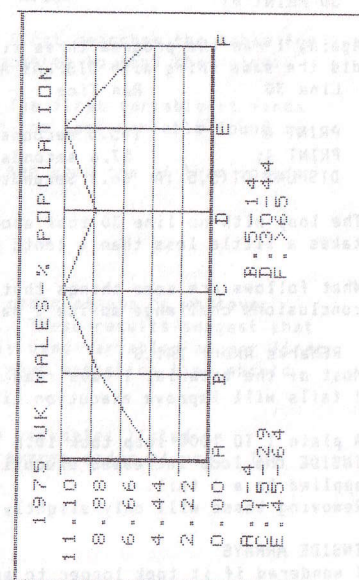
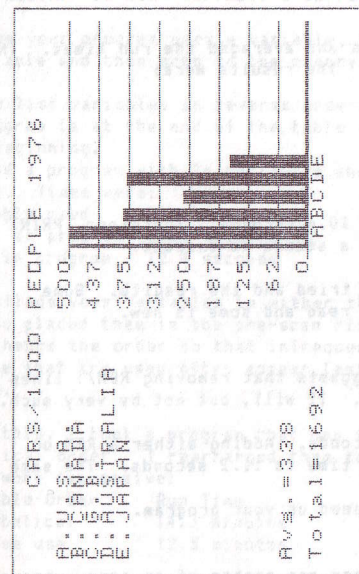


**GREAT LAKES SOFTWARE**

1974 ACCIDENTS UK



REGULAR...  
 MONTHLY...  
 QUARTERLY...  
 YEARLY...  
 ...  
 ...





```

X X BBBB # 17
X X B B
X BBBB By
X X B B Jim
X X BBBB Swedlow

```

[This article originally appeared in the User Group of Orange County, California ROM]

#### SPEEDING UP XB

I have been looking for ways to speed up Extended Basic programs and have found some interesting things.

Most of the information in this article is based on a FOR NEXT loop. To compare, for example, <PRINT A> with <PRINT A;>, I ran a program like this:

```

10 INPUT A$
20 FOR A=1 TO 1000
30 PRINT A
40 NEXT A

```

Line 10 is <INPUT A\$> to make sure that the pre-scan time did not skew the results and to give me a marker to start my stopwatch.

I ran the program three times and then averaged the run times. Next I changed line 30 to:

```
30 PRINT A;
```

Again, I ran the program three times and averaged the run times. Then I did the same thing with DISPLAY AT. The results were:

| Line 30           | Run Time      |
|-------------------|---------------|
| PRINT A           | 105.5 seconds |
| PRINT A;          | 57.6 seconds  |
| DISPLAY AT(5,5):A | 66.3 Seconds  |

The loop with no line 30 took about 10.6 seconds. Therefore, PRINT A takes a little less than a tenth of a second to execute.

What follows are some things that I tried and the results. Some conclusions challenge advice I have read and some is new.

#### REMARKS AND ! TAILS

Most of the material I have read suggests that removing REM/! lines and ! tails will improve execution time. It will, but not by very much.

A plain 1 TO 1000 loop took 10.6 seconds. Adding either a REM or ! line INSIDE the loop increased execution time to 11.2 seconds. The same applied to a ! tail.

Removing these will only slightly speed up your program.

#### INSIDE ARRAYS

I wondered if it took longer to access one member of an array versus another. I DIMentioned an array at 200 and then looked at different members.

I expected to find some relationship between the number and the time (faster to access the beginning or end). What I found was that the run time depended on the size of the number inside the parenthesis:

| Array Members    | Run Time     |
|------------------|--------------|
| B(1) to B(9)     | 15.3 seconds |
| B(10) to B(99)   | 15.6 seconds |
| B(100) to B(200) | 16.0 seconds |

Apparently the more digits a number has, the longer it takes XB to read and digest it. Further, it takes about the same amount of time to access any member of an array.

#### DEFINITIONS ARE GLACIAL

I knew from experience that DEFINITIONS were time consuming but I was surprised to find out just how slow they are. I ran a loop with a calculation done thru a DEF and then without the DEF. The DEF loop was 55 seconds slower than the non-DEF loop.

It takes a long time (in computer terms) for your 4A to find a DEF (NOT counting the actual time to execute the calculation).

Avoid DEFINITIONS!

#### ORDER OF VARIABLES

One of the things your TI does during pre-scan is to build a variable table. This is a table of all of the variables in the program and the memory location of each variable's current value.

Each time your program uses a variable, it first searches the table for the variable and then goes to the memory location to find the value.

Your TI's list variables in reverse order. The first variable it finds in a program is at the end of the table and the last variable found is at the beginning.

I created a program with 26 variables and then used one of them inside the loop. Times were:

| Variable used    | Run Time     |
|------------------|--------------|
| First in program | 19.0 seconds |
| Last in program  | 12.4 seconds |

Your TI finds your variables in either the order of use or whatever order you placed them in the pre-scan list. These results suggest that if you change the order so that infrequently used variables appear first and those that are used often appear last, your execution time should decrease.

To test this, I took a program that had the variables listed in alphabetical order. I rearranged them to reverse order of use. The results were impressive:

| Variable Order | Run Time     |
|----------------|--------------|
| Alphabetical   | 14.3 minutes |
| Reverse use    | 12.5 minutes |

This step reduced run time 1.8 minutes or 12% percent. NOT BAD!



TIPS FROM THE TIGERCUB  
#58.1  
Tigercub Software 156 Collingwood  
Ave. Columbus OH 43213

LINE NUMBERS  
Your TI also uses a line number table. This table is similarly in reverse order (first line last, last line first, etc). When you use a line number in your program (GOTO, etc), your 4A must search the line number table to find the memory location of the line contents. Then it reads the line instructions, crunches them and executes.

I constructed a program that looks something like this:

```
10 INPUT A$
20 FOR I=1 TO 1000
30 ??????
40 NEXT I
50 RETURN
```

-----  
Lines 60 thru 2050  
- 200 lines -  
are all REM lines  
<60 REM>, <70 REM>, etc  
-----

```
2060 RETURN
2070 SUB A :: SUBEND
```

Run times depended on line 30:

| Line 30    | Run Time     |
|------------|--------------|
| GOSUB 50   | 26.9 seconds |
| GOSUB 2060 | 12.6 seconds |
| CALL A     | 15.6 seconds |

GOSUB 50 took longer than GOSUB 2060 as the computer had to wade thru 202 lines to find line 50 in the line number table versus one line to find line 2060.

Clearly, in long programs, put your frequently used subroutines and groups of code at the end of your program.

SUMMARY

For other ideas read the September, 1984 issue of Millers Graphics' "The Smart Programmer" and a November, 1983 article in "99'er Home Computer Magazine" by John Dow, "Squeezing the most out of TI Basic".

Experiment. You will find other time savers. If you run some of the programs in this article you will probably get slightly different times because you may be more accurate with your stopwatch and your programs may differ slightly from mine.

I hope that these ideas will help you write and refine your XB programs.

Enjoy!

=====



I am still offering over 120 original programs at \$1 each, or on collection disks at \$5 each. The five Tips From The Tigercub disks are reduced to \$5 and the three Nuts & Bolts disks are now just \$10 each.

My catalog is available for \$1, deductible from your first order (specify TIGERCUB catalog).

\*\*\*\*\*  
TI-PD LIBRARY

I have selected public domain programs, by category, to fill over 300 disks, as full as possible if I had enough programs of the category, with all the Basic-only programs converted to XBasic, with an E/A loader provided for assembly programs if possible, instructions added and any obvious bugs corrected, and with an auto loader by full program name on each disk. These are available as a copying service for just \$1.50 post paid in U.S. and Canada. No fairware will be offered without the author's permission. Send SASE for list or \$1, refundable for 11page catalog listing all titles and authors. Be sure to specify TIDP catalog.

\*\*\*\*\*

In Tips #55 I published a CHARSUB routine to convert character patterns into assembly source code, and in Tips #55 and #56 I published several routines to manipulate hex codes into new character sets. Those patterns locked fine on my old TV, but when I demo'd them on a high resolution monitor I could see too many missing pixels.

So I wrote this CHARFIX program which, when MERGED into a program and CALLED after any character redefinition is completed, will permit any normal or re-identified character to be viewed on screen and edited and will then write the hex codes of any range of printable characters into an

assembly source file which can be assembled, loaded and linked to instantly change character sets. This routine also reidentifies the common punctuation into the same character sets as the letters, as described in Tips #55. If you do not want this feature, delete lines 29001-29003.

```
29000 SUB CHARFIX
29001 DATA 32,33,34,44,46
29002 RESTORE 29001 :: FOR J
=1 TO 5 :: READ CH :: CALL C
HARPAT(CH,CH#):: CALL CHAR(J
+90,CH#):: CALL CHAR(J+122,C
H#):: NEXT J
29003 CALL CHARPAT(63,CH#)::
CALL CHAR(64,CH#):: CALL CH
AR(96,CH#)
29004 DISPLAY AT(1,1)ERASE A
LL:"1 2 3 4 5 6 7 8 9 0 : ;"
:" " : " @ A B C D E F G H I J
K L M : " : " N O P Q R S T U
V W X Y Z [ : " : " \ ] ^ _ a
b c d e f g h i j"
29005 DISPLAY AT(9,1):"k l m
n o p q r s t u v w x : " :
"y z { | } ~"
29006 CALL CHAR(128,"FF"&RPT
$("81",6)&RPT$("FF",9)&"FFFF
"&RPT$("C3",4)&"FFFF"):: CAL
L COLOR(13,2,16)
29007 CALL CHARVIEW
29008 SUBEND
29009 SUB CHARVIEW
29010 DISPLAY AT(13,14):"CTR
L V TO VIEW" :: DISPLAY AT(1
4,14):" " :: DISPLAY AT(15,1
4):"CTRL E TO EDIT" :: DISPL
AY AT(17,14):"CTRL S TO SAVE
"
29011 DISPLAY AT(19,14):" "
:: DISPLAY AT(20,14):" "
29012 CALL KEY(0,@,S):: IF S
=0 THEN 29012 ELSE IF @=150
THEN 29015 ELSE IF @=133 THE
N 29014 ELSE IF @=147 THEN 2
9013 ELSE 29012
29013 CALL DELSPRITE(#1):: C
ALL CHARSUB(HX#( )):: DISPLAY
BEEP :: STOP
29014 CALL EDIT(K):: GOTO 29
010
29015 DISPLAY AT(24,1)BEEP:"
"
29016 DISPLAY AT(24,1):"PRES
S A KEY" :: CALL KEY(0,K,S):
: IF S<1 OR K<32 OR K>143 TH
EN 29016
29017 DISPLAY AT(24,1):" " ::
CALL CHARPAT(K,CH#)
```



```

29018 R=13 :: FOR J=1 TO 15
STEP 2
29019 H#=SEG$(CH$,J,1):: CALL
L HEX_BIN(H$,B$)
29020 H#=SEG$(CH$,J+1,1):: C
ALL HEX_BIN(H$,B$):: FOR L=
1 TO 8 :: C$=C$&CHR$(ASC(SEG
$(B$&B$,L,1))+80):: NEXT L
29021 DISPLAY AT(R,1):C$::
DISPLAY AT(R,10):SEG$(CH$,J,
2):: R=R+1 :: C$="" :: NEXT
J :: DISPLAY AT(22,1):CH$::
: GOTO 29012
29022 SUBEND
29023 SUB HEX_BIN(H$,B$):: H
X$="0123456789ABCDEF" :: BN$
="0000X0001X0010X0011X0100X0
101X0110X0111X1000X1001X1010
X1011X1100X1101X1110X1111"
29024 FOR J=LEN(H$) TO 1 STEP
-1 :: X$=SEG$(H$,J,1)
29025 X=POS(HX$,X$,1)-1 :: T
$=SEG$(BN$,X$+1,4)&T$ :: NE
XT J :: B$=T$ :: T$="" :: SU
BEND
29026 SUB CHARSUB(HX$( ))
29027 DISPLAY AT(12,1)ERASE
ALL:"Source code filename?":
"DSK" :: ACCEPT AT(13,4)SIZE
(12)BEEP:F$ :: OPEN #1:"DSK"
&F$,OUTPUT
29028 DISPLAY AT(15,1):"LINK
ABLE program name?" :: ACCEP
T AT(16,1)SIZE(6):P$
29029 DISPLAY AT(18,1):"Rede
fine characters from ASCII
I to ASCII"
29030 ACCEPT AT(19,7)VALIDAT
E(DIGIT)SIZE(3):F
29031 ACCEPT AT(19,21)VALIDA
TE(DIGIT)SIZE(3):T
29032 PRINT #1:TAB(8);"DEF";
TAB(13);P$ :: PRINT #1:"VMBW
EQU >2024" :: PRINT #1:"
STATUS EQU >837C"
29033 NB=(T-F)*8 :: CALL DEC
_HEX(NB,H$):: A=768+F*8 :: C
ALL DEC_HEX(A,A$)
29034 FOR CH=F TO T :: IF CH
<144 THEN CALL CHARPAT(CH,CH
$)ELSE CH$=HX$(CH)
29035 IF FLAG=0 THEN PRINT #
1:"FONT";:: FLAG=1
29036 FOR J=1 TO 13 STEP 4 :
M$=M$&">"&SEG$(CH$,J,4)&"
" :: NEXT J :: M$=SEG$(M$,1,
23)&" *"&CHR$(CH)
29037 PRINT #1:TAB(8);"DATA
"&M$ :: M$="" :: NEXT CH

```

```

29038 PRINT #1:P$;TAB(8);"LI
R,Font" :: PRINT #1:TAB(
8);"LI RO,>"&A$ :: PRINT #
1:TAB(8);"LI R2,>"&H$
29039 PRINT #1:TAB(8);"BLWP
@VMBW":TAB(8);"CLR @STATUS"
:TAB(8);"RT":TAB(8);"END" ::
CLOSE #1
29040 SUBEND
29041 SUB DEC_HEX(D,H$)
29042 X$="0123456789ABCDEF"
:: A=D+65536*(D>32767)
29043 H$=SEG$(X$, (INT(A/4096
)AND 15)+1,1)&SEG$(X$, (INT(A
/256)AND 15)+1,1)&SEG$(X$, (I
NT(A/16)AND 15)+1,1)&SEG$(X$
,(A AND 15)+1,1):: SUBEND
29044 SUB EDIT(CH)
29045 DISPLAY AT(13,14):"1 T
OGGLE" :: DISPLAY AT(14,1
5):"CURSOR" :: DISPLAY AT(15
,14):"E S D X TO MOVE" :: DI
SPLAY AT(17,14):"CTRL A TO A
BORT"
29046 DISPLAY AT(19,14):"CTR
L R TO" :: DISPLAY AT(20,15)
:"REIDENTIFY"
29047 R=13 :: C=3 :: X=128 :
: CALL SPRITE(#1,130,11,R*8-
7,C*8-7):: X$=CHR$(129)&CHR$
(146)
29048 CALL KEY(0,K,S):: IF S
<1 THEN 29048 ELSE ON POS("1
EeSsDdXx"&X$,CHR$(K,1))+1 GO
TO 29048,29049,29050,29050,2
9051,29051,29052,29052,29053
,29053,29055,29056
29049 X=X+1+(X=129)*2 :: GOT
O 29054
29050 R=R-1-(R=13):: GOTO 29
054
29051 C=C-1-(C=3):: GOTO 290
54
29052 C=C+1+(C=10):: GOTO 29
054
29053 R=R+1+(R=20)
29054 CALL LOCATE(#1,R*8-7,C
*8-7):: CALL HCHAR(R,C,X)::
GOTO 29048
29055 CALL DELSPRITE(#1):: S
UBEXIT
29056 FOR R=13 TO 20 :: FOR
C=3 TO 10 :: CALL GCHAR(R,C,
GH):: CALL LOCATE(#1,R*8-7,C
*8-7):: B$=B$&CHR$(GH80)::
NEXT C
29057 CALL BIN_HEX(B$,H$)::
DISPLAY AT(R,10):H$:: B$=""
:: HEX$=HEX$&H$ :: NEXT R :
: DISPLAY AT(22,1):HEX$:: C
ALL CHAR(CH,HEX$):: HEX$=""

```

```

29058 CALL DELSPRITE(#1):: F
OR R=13 TO 20 :: DISPLAY AT(
R,14):" :: NEXT R :: SUBEND
29059 SUB BIN_HEX(B$,H$):: H
X$="0123456789ABCDEF" :: BN$
="0000X0001X0010X0011X0100X0
101X0110X0111X1000X1001X1010
X1011X1100X1101X1110X1111"
29060 L=LEN(B$):: IF L/4<>IN
T(L/4)THEN B$="0"&B$ :: GOTO
29060
29061 FOR J=L-3 TO 1 STEP -4
:: X$=SEG$(B$,J,4)
29062 X=(POS(BN$,X$,1)-1)/5
:: T$=SEG$(HX$,X+1,1)&T$ ::
NEXT J :: H$=T$ :: T$="" ::
SUBEND
I think that programs, at least
noncommercial ones, should be
open for anyone to modify for
their own use. For that reason, I
would not normally publish the
following routine. How ever, I
recently received a large number
of programs, originally in the IUG
library, and found that the
author's name had been erased from
the title screen or REM of every
one of them. I know, because I
already had many of the original
versions, including some that I
wrote myself.
Now, that is inexcusable. If a
programmer is willing to share
his work, he does deserve credit
for it. And if people are going
to play that dirty, maybe there is
good reason for protecting
programs.
So here is how to do it. Ken
Woodcock wrote this in genius
routine and published it in the
Tidewater newsletter. I have
modified it so that it can be
deleted after it has done its
work. It is to be MERGED into any
XBasic program(32k required) and
RUN, and will change the line
length byte of each line to zero,
so that the program cannot be
LISTED, although it can be loaded
and run.
1 CALL INIT :: CALL PEEK(-31
752,A,B,C,D):: SL=C*256+D-65
539 :: EL=A*256+B-65536 :: F
OR X=SL TO EL STEP -4
2 CALL PEEK(X,E,F,G,H):: ADD
0*256+H-65536 :: J=J+1 :: I
F J<4 THEN 3 :: CALL LOAD(AD
D1,0)
3 NEXT X :: STOP :: !@P-

```

Save that as FIX in MERGE format. Merge it into any program (RESequence first if it has line numbers less than 4) and RUN. Then type 1, FCTN X and FCTN 3 to delete line 1. Delete lines 2 and 3 in the same way. Then SAVE. Now try LISTing it and watch the fireworks.

Ken wrote an even more ingenious UNFIX routine to unprotect the program, but I'm not passing that on!

Now, suppose you have a party game program that you don't want the kids playing with. So, RESequence it to some odd number, such as RES 797. Put in a line just before that 796 STOP. Then merge in FIX, run it, and delete those first 3 lines.

I hope you remember what line number you resequenced it to start from, because now you can only run it by RUN 797!

In Tips #57 I reported the discovery that printing to the disk from the TI Writer Formatter, with the C option, really converted the carriage returns to trailing blank ASCII 32's, and I published a routine to strip them. I have found an easier way. First PF and C DSK... to convert the CRs to blanks. LF DSK... and SF DSK... to strip out those blanks, but that leaves the pestiferous tab line, so LF DSK... and PF DSK... again!

=====

First key this in

```

1 DISPLAY AT(12,1)ERASE ALL:
"SKIP INSTRUCTIONS? Y" :: AC
CEPT AT(12,20)SIZE(-1)VALIDA
TE("YNyn");@Q$ :: IF @Q$="Y"
OR @Q$="y" THEN 8
2 DISPLAY AT(24,5)ERASE ALL:
"PRESS ANY KEY"
3 RESTORE 30721
4 REM
5 FOR J@=1 TO T@ :: READ @#
:: DISPLAY AT(J@,1):@#:" "
6 CALL KEY(0,K,@S):: IF S@=
0 THEN 6
7 NEXT J@
8 DATA 0
9 RESTORE 8 :: READ N
10 REM

```



Save it by  
SAVE DSK1.D/MERGE, MERGE  
Then key this in

```
100 OPEN #1:"DSK1.D/MERGE",V
ARIABLE 163,INPUT :: OPEN #2
:"DSK1.D/MERGE2",VARIABLE 16
3,OUTPUT :: L=129 :: FOR J=1
TO 10
110 LINPUT #1:M# :: PRINT #2
:CHR$(0)&CHR$(L+J)&CHR$(156)
&CHR$(253)&CHR$(200)&CHR$(1)
&"!"&CHR$(181)&CHR$(199)&CHR
$(LEN(M#))&M#&CHR$(0):: NEXT
J
120 CLOSE #1 :: PRINT #2:CHR
$(255)&CHR$(255):: CLOSE #2
```

Run it to convert D/MERGE into a merge format file D/MERGE2 on DSK1. Then key this in. Don't change line numbers.

```
100 CALL CLEAR :: OPEN #1:"D
SK1.@DATA",VARIABLE 163,OUTP
UT :: DEF L$(X)=CHR$(120)&CH
R$(X)
105 PRINT #1:L$(X)&CHR$(161)
&CHR$(200)&CHR$(6)&"@DUMMY"&
CHR$(0)
110 L=L+1 :: X=X+1 :: ACCEPT
AT(L,0):M# :: IF L=24 THEN
CALL CLEAR :: L=0
120 IF M#<>"END" AND M#<>"en
d" THEN PRINT #1:L$(X)&CHR$(
147)&CHR$(199)&CHR$(LEN(M#))
&M#&CHR$(0):: GOTO 110
130 REM
140 PRINT #1:CHR$(0)&CHR$(4)
&"T"&CHR$(190)&CHR$(200)&CH
R$(LEN(STR$(X1)))&STR$(X1)
&CHR$(0)
141 PRINT #1:L$(X)&CHR$(168)
&CHR$(0)
150 PRINT #1:CHR$(255)&CHR$(
255):: CLOSE #1
```

Enter MERGE DSK1.D/MERGE2 to merge in that file. SAVE the program as DATAWRITER. Then RUN it and try it out by using it to write itself some instructions. Answer the prompts with

DATAWRITER V1.2  
by Jim Peterson

To be used to add instruc- tions to programs.

Type the instructions and format them, centered or hyphenated or rightadjusted just as you want them to appear on screen, and enter each line. They will be written to a D/V163 file named @DATA. When finished, enter END.

Then enter NEW, then MERGE DSK1.@DATA, and RUN to see if everything is OK. If so, load the program needing instructions, make sure its lowest line number is more than 10 and the highest is less than 30721, and enter MERGE DSK1.@DATA.

And enter END, then OLD DSK1.DATAWRITER, then MERGE DSK1.@DATA.

=====



## MACHINE CODE TUTORIAL PART THREE

by Mack McCormick

### KEYBOARD AND JOYSTICK ACCESS

OR

### INTERACTING WITH YOUR COMPUTER

This assembly language tutorial describes how to use the keyscan (KSCAN) utility contained in the computer for keyboard and joystick input. After you have studied this tutorial you should understand the key addresses required, how to set up for the routine, and how to invoke the utility.

A somewhat different approach has been taken with this tutorial. I have included a few advanced programming concepts to reduce the learning curve required to go from the beginning books on the market to the advanced techniques used in commercial software.

If you don't understand some of these concepts then don't worry, they will come to you later. Simply disregard them for now. Concentrate on the objective of learning the keyscan routine. I would appreciate any feedback on whether you like or dislike the approach of introducing advanced techniques gradually. Several folks have asked questions about addressing modes for opcodes. I have attempted to use all of the modes and will explain them as we go through the program. Let's get started!

If you think about it Extended BASIC uses the KSCAN routine a lot. It's used when you are in the command mode and when in the program execution mode it is used for INPUT, CALL KEY, CALL JOYST, and ACCEPT AT. In fact it's the only way we have to interface to the computer. The KSCAN routine itself is contained in the console in the ROM chip beginning at address >20. It's a large routine and it talks to the keyboard using the Communications Register Unit (CRU) thru a TMS9901 Peripheral Systems Interface chip (just background stuff). It is not difficult to use.

Just REFERENCE KSCAN in your program to tell the computer you will be using it's KSCAN routine. The X/B equate is >201C. Just do a BLWP @KSCAN to use the routine. There are four single byte addresses you need to know in order to use KSCAN. >8374 tells the computer how to map out the keyboard for scanning. Place one of the following bytes at this address to change the keyboard mapping.

```
>00 Standard TI Keyboard.
>01 Left Side of keyboard and JOYST 1
>02 Right side and JOYST 2
>04 PASCAL keyboard
```

See your BASIC reference guide for detailed information on the keycodes returned in each mode for each key.

>8375 contains the ASCII hex value of the last key pressed or >FF if no key was pressed. We use the >FF feature to allow us to detect if a key is held down for auto-repeating.

>8376 contains the value of the Y (up/down) position of the JOYST and >8377 the X (right/left) position. There are three values returned >00 for no movement, >04, and >FC (-4). These values actually are looked up in a GROM chip by the KSCAN routine based on the position of the controller. There is another way to check if a key has been pressed. The GPL status byte at >837C is bit mapped (each bit in the byte provides different information). Here's the way the STATUS byte is mapped.

| 0  | 1 | 2  | 3 | 4   | 5  | 6 | 7 |
|----|---|----|---|-----|----|---|---|
| IL | A | IE | C | IOV | IO | X | - |



This status byte is set as a result of the execution of an opcode. For a detailed explanation see you E/A manual. The bit we are interested in for the KSCAN routine is the EQUAL or bit number 2. This bit is set whenever the result from an opcode is equal.

Many opcodes affect this bit and other opcodes use it to make a branching decision. For example, jump equal (JEQ) and jump not equal (JNE) check this bit to see if it is on when deciding if a jump is to be taken. There is a somewhat unique feature that is frequently used in programming. If you execute MOV R1,R1 or move any byte or word to itself the equal status bit will be set if the value there is equal to zero. We use this feature to check for a key press.

By MOV @STATUS,@STATUS we can see if the equal status bit is set. If set by the KSCAN routine no key has been pressed. If reset, there has been a key pressed. If a key was pressed we can get it's ASCII value from >B375. OK, there's most of the background let's go to the program and look at how KSCAN is used in practice.

Remember that the computer is actually dumb but very fast. Consider for a moment what is required to have a full function cursor in BASIC. You even have to tell the computer to put the cursor on the screen.

Note we REFERENCED KSCAN as a system utility we will be using. Next we establish the EQUates needed for the program. All these do is equate a label with a value and do not occupy any memory. The label may now be used instead of that value.

We EQUated MYHS with >B300 which is the bottom of high speed RAM which is actually contained in the computer. I say high speed because the CPU can address 16 bits at a time instead of 8 bits if we placed our workspace in expansion RAM.

This increases the speed of program execution as long as we keep the most frequently used values in our registers. This is generally a good place for your workspace even when LINKING from BASIC or X/B but there are exceptions so consult your manual first if you are using system ROM utilities.

Next we have the five addresses >B374 - >B37C needed for the KSCAN utility. GPLWS EQU >B3E0 is the address for the Graphics Programming Utility (GPL) workspace which is used in BASIC and elsewhere as we will see in a moment. The next address is >BC02 which is the address where the write address for the VDP chip is memory mapped in CPU RAM. More in a moment. VWD EQU >BC00-VDPWA will be used as an addressing offset later in the program. This EQUates the label VWD to >FFFE or (-2).

Next are the values we will use for comparison with the ASCII value returned by KSCAN at >B375 in the program.

The statement LI VDP,VDPWA uses the EQUates we established previously to load R15 (the R is not required to be present) with the value of >BC02 (the VDP write address).

Next we set up the screen by setting the background to white and then using a subroutine to set a block of VDP RAM (the color table in this case) equal to a value (dark blue on white characters in this case). Look at the BLKVDP subroutine for a moment. The BL branch and link opcode branches to a subroutine which uses the same workspace registers as the main program. The computer knows where to return because the return address (next statement after the BL) is placed in R11.

We can pass data to a subroutine by following the BL with a DATA statement. In this case we are passing three pieces of data. The VDP RAM location to write to, The data value to write, and the number of times to write it. We use an addressing mode called indirect auto incrementing to help us out. The statement MOV \*R11+,R0 does the following.

R11 is pointing to the address of the word of memory following the BL at this time (>3B0 in this case).

This statement says "Move the value of the data at the address contained in R11 to R0 and then increment the address in R11 to the address of the next word of memory (>4F00 in this case). We just passed >3B0 to R0 for use by the subroutine.

ORI R0,>4000 sets the first two bits of >3B0 to binary 01 or >43B0 without changing the other bits.

This is necessary because the first two bits set to 01 tell the VDP chip that we are going to alter the address where it writes or reads data in VDP RAM. Once the new address is set the VDP chip will automatically increment the address to the next byte for consecutive byte writes or reads. This really saves time because a VSBW or VSBR would alter the address every time the routine is used.

Data is then passed to R1 and R2 using the same technique. Now we are ready to alter the VDP address. We \*always\* write the least significant byte first so we execute a SWAP Byte command and MOVE >80 in this case to the VDPWriteAddress. Now we swap the bytes back and write the most significant byte (>43 in this case).

We can now move consecutive data to or from VDP RAM very quickly using the statement MOV B R1,@VWD(VDP). This moves >4F in this case to the VDP write data address at >8C00. How did we come up with >8C00? By using indexed memory addressing of course. Remember VDP is really R15 which we loaded previously with the VDPWA (>BC02).

The parenthesis always contains a register. Remember that VWD contains >FFFE or -2. This statement adds the value contained in VWD to the value contained in VDP (R15 ,>BC02) and that becomes the address the data to move to or >8C00. Easy huh? <grin>.

Because we incremented the address in R11 to the next word in the previous statement MOV \*R11+,R2 it is now pointing to the next instruction in the program or the following BL @BLKVDP. Next we clear the screen by writing spaces (>20) to all positions and put up the prompts. Now for the first keyscan.

We previously cleared the word of memory at KEYADR or >B374 to tell the computer to scan the entire standard keyboard. Next we MOVEByte @STATUS,@STATUS to see if the equal bit is set (on). We check with a JumpEqual opcode and if the STATUS bit is set we loop back to the KSCAN routine until a key is pressed.

When a key is pressed we fall out of the loop and compare the value at KEYVAL (>B375) to a constant to see which key was pressed. We only are checking for one, two, or quit being pressed in this case. If there had been multiple keys we were checking (say more than 5) then the most efficient means would have been to set up a table and use indexed memory addressing to branch to the correct routine. When the correct key is pressed we Jump to that routine and execute.

Note the EOJQ routine. This causes the computer to return to the power up title screen. First interrupts are enabled using a LoadInterruptMaskImmediate command because the GPL interpreter (system monitor) runs with interrupts enabled. Next we tell the computer to use the GPL Workspace. Finally, we Branch and Link Workspace Pointer to the power up reset vector contained in console ROM at address >0000.

Let's look now at the VDWTR routine. First we BranchandLink to another subroutine to clear the screen and place the MSG3 prompt at the bottom of the screen.

No data (parameters) are passed to this subroutine. Note that the return address in R11 is saved in R10 because we BL to the BLKVDP routine and the original return address would be lost. Rule: When nesting subroutines always save the return address. This is one method when you only are going one or two levels deep.

In some programs I write I go up to six levels deep. In that case I implement a stack in CPU RAM and PUSH and POP values to the stack. Finally we B \*R10. We Branch to the address contained in R10 which is our original return address. This is known as indirect addressing. Note in the next keyscan we do not check the STATUS byte to detect a keypress but rather check KEYVAL (>B375) for >FF (no key pressed).



This enables the key to auto-repeat. We use some code to slow it down. You should try experimenting with these values. Next we see if Clear, Redo, or the Left Arrow was pressed. If not we print the character on the screen. This routine is fairly straight forward and well documented so you should have no problem following it.

Next is the JOYST routine. Again we clear the screen. Next we scan the entire keyboard. One note on addressing here. The statement CLR @KEYADR tells the computer to clear "whats at (@)" the address of the KEYADR label (>8374 in this case). This is called symbolic memory addressing. MOVB @JOYVAL,@KEYVAL tells the computer to scan the left half of the keyboard and JOYST one by moving a value of 1 to >8374.

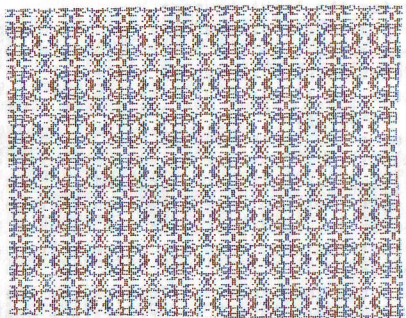
Then we execute a KSCAN but don't wait for a key. We then divide the JOYST routine to check for three basic conditions; up, down, and if neither of these then left/right.

We only check for left/right movement in the up and down routines if the JOYST was moved either up or down. This routine is sufficiently documented for you to follow from here.

If you are really interested in learning assembler there is only one way. Hit the books and practice programming. Please feel free to ask any questions you may have on anything I did not make clear. The next tutorial will cover File Handling. Until then "Assembler Executing". \*Your\* SYSOP. Mack McCormick

The following code is available on disk from the disk library.

```
*****
***** As of: 8 May 1985
*****
***
*** TUTORIAL 3 (NOVICE)
*** Keyboard and Joystick Access
*** By Mack McCormick 74206,1522
*** Use Load and Run from
*** M/M or E/A
*** Entry point is START
*** Active Fctns:Clear,Redo,Quit
*** and Left Arrow
***
*****
*****
```



```
DEF START
REF KSCAN,VSBW,VMBW,VWTR
```

TITL 'JOYSTICK AND KEYSKAN DEMO PROGRAM 10 MAY 1985'

```
*****
* CPU RAM EQUATES
*****
MYWS EQU >8300 DEFINE MY WORKSPACE (HIGH SPEED RAM)
KEYADR EQU >8374 KEYBOARD TO SCAN (>00 TO >04)
KEYVAL EQU >8375 ADDRESS WHERE THE KEY VALUE (ASCII) IS RETURNED
YPOS EQU >8376 Y POSITION FOR THE JOYSTICK
XPOS EQU >8377 X POSITION FOR THE JOYSTICK
STATUS EQU >837C GPL STATUS BYTE
GPLWS EQU >83E0 ADDRESS OF GPL WORKSPACE
VDPWA EQU >8C02 VDP WRITE ADDRESS
VWD EQU >8C00-VDPWA WRITE DATA ADDRESS OFFSET
VDP EQU 15 REGISTER FOR THE VDP WRITE ADDRESS
```

```
*****
* KEYSTROKE VALUES STANDARD KEYBOARD
*****
ONE BYTE >31 ASCII VALUE FOR 1
TWO BYTE >32 ASCII VALUE FOR 2
CLEAR BYTE 2 FCTN 4
QUIT BYTE 5 FCTN =
REDO BYTE 6 FCTN B
LARRROW BYTE 8 FCTN S
H04 BYTE 4 JOYST UP
H00 BYTE 0 JOYST NO MOVEMENT
HFC BYTE >FC JOYST DOWN (-4)
HFF BYTE >FF VALUE FOR NO KEY PRESSED
```



```
*****
* MISC VALUES
*****
JOYVAL BYTE 1 VALUE FOR JOYST 1
MSG TEXT 'PLEASE SELECT'
MSG1 TEXT '1 - VIDWRITER'
MSG2 TEXT '2 - JOYST FUN'
MSG3 TEXT 'FCTN REDO-RETURN TO MAIN MENU'
EVEN FORCE THE PROGRAM COUNTER TO AN EVEN ADDRESS
```

```
*** PROGRAM STARTS HERE ***
START LWPI MYWS POINT TO OUR WORKSPACE
LI VDP,VDPWA LOAD THE VDP WRITE ADDRESS IN R15

*** SET UP THE SCREEN ***
LI R0,>070F SET THE BACKGROUND COLOR TO WHITE (>F). VDP REG 7.
BLWP @VWTR WRITE IT TO VDP REGISTER 7

BL @BLKVDP SET THE CHARACTERS TO DK BLU ON WHITE
DATA >380,>4F00,>1E COLOR TABLE, DK BLU ON WHT, 30 BYTES

ENTER BL @BLKVDP NOW CLEAR THE SCREEN
DATA 0,>2000,767 PLACE >20 IN ALL POSITIONS OF THE SCREEN IMAGE TABLE

CLR @KEYADR SET TO STD TI KEYBOARD

LI R0,201 PUT UP 1ST PROMPT
LI R1,MSG ADDRESS OF DATA TO WRITE
LI R2,13 NUMBER OF BYTES TO WRITE
BLWP @VMBW ON VDP ROUTINES eg. VMBW,VMBR, ETC.
LI R0,265 R0 IS ALWAYS THE VDP RAM ADDRESS
LI R1,MSG1 R1 IS ALWAYS THE CPU RAM ADDRESS OR CHAR TO WRITE

BLWP @VMBW IF SINGLE BYTE ROUTINE (VSBW,VSBR)
LI R0,297 R2 IS ALWAYS THE NUMBER OF BYTES TO MOVE
LI R1,MSG2
BLWP @VMBW
```

```
*** TITLE SCREEN KEYSKAN ***
SCAN1 BLWP @KSCAN SCAN THE KEYBOARD
MOVB @STATUS,@STATUS IS THE EQUAL STATUS BIT SET?
JEQ SCAN1 YEP. KEEP WAITING FOR A KEY PRESS
CB @KEYVAL,@ONE WAS ONE PRESSED?
JEQ VDWR YEP. GO EXECUTE THAT SUBPROGRAM
CB @KEYVAL,@TWO WAS TWO PRESSED?
JEQ JOYST YEP. GO EXECUTE THAT SUBPROGRAM
CB @KEYVAL,@QUIT WAS QUIT PRESSED?
JEQ EOJQ YEP. RETURN TO TITLE SCREEN
JMP SCAN1 ONLY ACCEPT 1,2, OR QUIT. KEEP SCANNING.
```



```

*-- RETURN TO THE POWER UP TITLE SCREEN --*
EOJQ  LIM1 2      ENABLE INTERRUPTS
      LWPI GPLWS  LOAD GPL WORKSPACE
      BLWP @>0000 BRANCH TO THE RESET VECTOR (TITLE SCREEN)

*-- VIDEO TYPEWRITER ROUTINE --*
VDWTR BL  @PRMT  CLEAR THE SCREEN AND PUT UP THE PROMPT

      CLR R3      SCREEN LOCATION COUNTER
SCAN2  BLWP @KSCAN SCAN THE KEYBOARD
      CB @KEYVAL,@HFF HAS A KEY BEEN PRESSED?
      JEQ SCAN2   KEEP CHECKING

      CB @KEYVAL,R1 SAME KEY AS LAST TIME?
      JEQ YEP     SAME KEY

* EXPERIMENT WITH THE VALUES FOR R4 UNTIL YOU OPTIMIZE THE SPEED
      LI R4,>3000  DELAY TO SLOW DOWN PRINTING FOR USER REACTION (DIFF KEY)
      JMP LOOP
YEP    LI R4,>5000  DELAY FOR AUTO REPEAT CURSOR
LOOP   DEC R4      SUBTRACT 1 20,480 TIMES (THATS FAST!)
      JNE LOOP    PLACE AN * IN FRONT OF THIS LINE TO SEE THE SPEED

NOPE   CB @KEYVAL,@CLEAR CLEAR THE SCREEN?
      JEQ VDWTR   RESTART THE PROGRAM AND CLEAR THE SCREEN
BYPASS CB @KEYVAL,@REDO EXECUTE A WARM START TO TITLE SCREEN
      JEQ ENTER   RETURN TO TITLE SCREEN
      CB @KEYVAL,@LARRW LEFT ARROW PRESSED?
      JEQ ARROW   YEP

      MOV R3,R0    MOVE THE COUNT TO R0
      MOVB @KEYVAL,R1 MOVE THE KEYVALUE TO R1 MSBYTE
      BLWP @VSBW  WRITE IT TO THE SCREEN

      INC R3      NEXT SCREEN POSITION
      CI R3,735  ONLY ALLOW TO WRITE TO SCREEN POSN 735
      JLT SCAN2  CONTINUE TO NEXT CHARACTER
      JMP VDWTR  CLEAR THE SCREEN AND RESET THE CHAR COUNTER

ARROW  DEC R3
      CI R3,0    CHECK FOR LOWER SCREEN LIMIT
      JLT VDWTR CLR R0 AND START OVER
      MOV R3,R0  MOVE THE COUNT TO R0
      LI R1,>2000 ACSII SPACE CHAR IN MSBYTE OF R1
      BLWP @VSBW BLANK OUT THE CHAR
      LI R1,>0800 LOAD FOR AUTO REPEAT DELAY
      JMP SCAN2  GET THE NEXT CHAR

*-- JOYSTICK ROUTINE --*
JOYST  BL  @PRMT  CLEAR THE SCREEN AND PUT UP THE PROMPT
      LI R0,400  START PATTERN IN CENTER OF THE SCREEN

JOYLP  CLR @KEYADR SET FOR KEYBOARD SCAN
SCAN3  BLWP @KSCAN SCAN FOR FCTN REDD
      CB @KEYVAL,@REDO REDD PRESSED?
      JEQ SCAN3  RETURN TO TITLE SCREEN
      CB @KEYVAL,@CLEAR CLEAR PRESSED?
      JEQ JOYST  START OVER

      MOVB @JOYVAL,@KEYADR SCAN JOYST 1
      BLWP @KSCAN SCAN THE JOYSTICK
      MOV R0,R14
      LI R0,16  * ADDED TO TEST FIRE BUTTON

```

```

MOVB @KEYVAL,R1
AI R1,>2000
BLWP @VSBW
MOV R14,R0

      CB @YPOS,@H04 MOVED UP?
      JNE CKDWN     NOPE. MUST BE DOWN

      CB @XPOS,@H00 MOVED IN THE X DIRECTION?
      JNE XDET     YEP GO FIND OUT IF LEFT OR RIGHT
      AI R0,-32    MOVED UP ONLY SO -32 TO GET TO ROW ABOVE
      JMP DRAW     DRAW THE PATTERN

XDET   CB @XPOS,@H04 MOVED UP AND RIGHT?
      JNE XDET1    NOPE
      AI R0,-31    PRINT POS UP AND TO RIGHT ONE POSITION
      JMP DRAW     DRAW IT

XDET1  AI R0,-33   UP AND LEFT IS ALL THAT IS LEFT
      JMP DRAW     DRAW IT

CKDWN  CB @YPOS,@HFC WAS JOYST MOVED DOWN?
      JNE LAT      NOPE

      CB @XPOS,@H00 MOVED STRAIGHT DOWN?
      JNE XDET2    NOPE
      AI R0,32     NEXT ROW DOWN
      JMP DRAW     DRAW IT

XDET2  CB @XPOS,@HFC DOWN AND LEFT?
      JNE XDET3    NOPE
      AI R0,31     ONE LINE DOWN AND ONE SPACE LEFT
      JMP DRAW     DRAW IT

XDET3  AI R0,33    ONE LINE DOWN AND ONE POSITION RIGHT

LAT    CB @XPOS,@HFC NOT MOVED UP OR DOWN. WAS IT LEFT?
      JNE LAT1     NOPE
      DEC R0       IT WAS SO MOSVE LEFT ONE POSITION
      JMP DRAW     DRAW IT

LAT1   CB @XPOS,@H04 WAS IT MOVED RIGHT?
      JNE JOYLP    START ALL OVER AT BEGINNING
      INC R0       YEP MOVED RIGHT.

DRAW   CI R0,0     CHECK FOR LESS THAN 0
      JLT OUTBDS  YEP OUT OF BOUNDS
      CI R0,735   CHECK FOR GREATER THAN 735
      JGT OUTBDS  YEP OUT OF BOUNDS
      JMP INBDS   JUMP TO INBOUNDS
OUTBDS MOV R2,R0   RESTORE OLD SCREEN LOCATION IN R0

INBDS  LI R1,>4000 LOAD THE @ SYMBOL TO DISPLAY
      MOV R0,R2   SAVE THE OLD VALUE IN CASE NEW IS OUT OF BOUNDS
      BLWP @VSBW  WRITE IT TO THE SCREEN

      LI R4,>1000  DELAY FOR A FEW MSECS.
DELAY  DEC R4
      JNE DELAY  PLACE * IN FRONT TO SEE REAL SPEED
      JMP JOYLP  START ALL OVER

```



```

*****
* This subroutine clears the screen *
* and places the MSG3 on the screen *
* No inputs required. *
* Alters R0,R1,R2,R10. *
* No reserved registers on exit *
* Levels 2 *
*****

```

```

PRMT MOV R11,R10 SAVE THE RT ADDRESS SINCE WE WILL NEST ANOTHER SUB

```

```

BL @BLKVDP CLEAR 732 SCREEN POSITIONS
DATA 0,>2000,737 PLACE >20 IN THE SIT FOR 732 POSITIONS

```

```

LI R0,73B LINE 23 OF SCREEN (BASE 0)
LI R1,MSG3 ADDRESS OF MSG3
LI R2,29
BLWP @VMBW

```

```

B *R10 RETURN TO MAIN PROGRAM

```

```

*****

```

```

* This subroutine writes data to a *
* block of VDP RAM *
* by directly accessing the VDP *
* Advantage: Very Fast and efficient *
* Inputs: ADDRESS,DATA,# OF BYTES *
* Alters:R0,R1,R2,VDPWA *
* Levels 1 *
*****

```

```

BLKVDP MOV *R11+,R0 VDP ADDRESS TO WRITE TO
ORI R0,>4000 OR ADDRESS WITH BINARY 0100 (>4) TO INDICATE A WRITE
MOV *R11+,R1 DATA TO WRITE
MOV *R11+,R2 NUMBER OF BYTES TO WRITE

```

```

* R11 (RT ADDR) NOW POINTS TO THE INST FOLLOWING THE BL IN THE MAIN PROGRAM

```

```

SWPB R0 ALWAYS WRITE THE LSBYTE FIRST
MOVB R0,@VDPWA MOVE THE LSBYTE TO THE VDP ADDRESS REGISTER
SWPB R0 SWAP THE BYTES BACK
MOVB R0,@VDPWA WRITE THE MSBYTE NOW

```

```

* THE VDP CHIP ADDRESS REGISTER IS NOW POINTING TO THE ADDRESS IN R0

```

```

BLKLP MOVB R1,@VWD(VDP) MOVES DATA TO EACH SCREEN POSITION
* VDP CHIP ADDRESS REGISTER AUTOMATICALLY INCREMENTS TO NEXT WRITE POSITION
DEC R2 DECREMENT THE COUNT
JNE BLKLP FINISHED? (GPL STATUS EQ BIT SET?)={0?} NOPE...CLRLP

```

```

RT

```

```

END

```

```

TI BITS * Number 20
By Jim Swedlow

```

[This article originally appeared in the User Group of Orange County, California ROM]

#### HOW YOUR TI SAVES TEXT FILES

In our TI world, most text is saved in Display Variable 80 (DV80) files. This is what the TI Writer Editor uses. What is DV80?

Display means that the file is saved using ASCII characters. If you use a disk sector editor to look at a DV80 file, you will see that the text looks just about the same as it was written. Internal files are not always as easy to read (but that is another column).

A file is made up of records. In a DV80 file, each record contains one line of text as it appears in the Editor. Variable means that each record is only as long as the text line.

Consider these two lines of text:

```

TI
99/4A

```

When this is saved on disk, there will be two records in the file. The first record is "TI" and the second is "99/4A". Each record is preceded by the number of characters in the record in Hex. Hex FF is used to mark the end of the file. The file would look like this:

```

Hex 02 54 49 05 39 39 2F 34 41 FF
ASC T I 9 9 / 4 A

```

In a Display Fixed 80 (DF80) file, each record still contains one text line but is exactly 80 characters long. Your 4A pads each record by adding the required number of spaces to the end of each text line.

#### WHY YOU NEED TO KNOW HOW THE REST OF THE WORLD SAVES TEXT FILES (IF YOU HAVE A MODEM):

As a loyal TI user you may not think that you need to know how others (MS-DOS, CPM, etc) save text files. If you do any work with modems, however, you do.

The reason is that you may download a text file and find that it is Display Fixed 128 (DF128). Why? There is a standard protocol in the TI world for transferring files using XMODEM. It was designed by Paul Charlton (creator of FAST TERM). The first record is NOT the first line of text. Instead, it is the disk header sector (which describes the file in a manner than can be read by the disk controller). If the first record is not the header, however, your modem program (TELCO, FAST TERM, MASS TRANSFER, etc) assumes that you are talking to a non TI system and will save the file as DF128. The reason, then, that you need to know how other systems save text files is that you may get one.

#### HOW THE REST OF THE WORLD SAVES TEXT FILES

The short answer is DF128. But there is more. Unlike a DF80 file, there is no padding. Instead, all of the text lines are run together. The end of each text line is marked with a carriage return <CR or CHR\$(13)> and a line feed <LF or CHR\$(10)>.

One record may have one, two or more text lines, each ending with a CR and LF. If there is not enough room left in a record for the next text line, enough of the line is added to the record to bring it to 128 characters. The rest of the text line starts the next record - followed by a CR and LF. The end of the file is marked with CHR\$(26), which in the IBM and CPM worlds is <CTRL Z>.

Remember our sample text?

```

TI
99/4A

```



Since it well under 128 characters, the file will only contain one record:

```
Hex 54 49 0D 0A 39 39 2F 34 41 0D 0A 1A
ASC T I          9 9 / 4 A
```

Hex 0D 0A is a CR and LF. Hex 1A is the end of file marker CHR\$(26).

#### CONVERTING FILES

There are a number of programs that convert files from DF128 to DV80 or from DV80 to DF128. Some of the assembly ones are quite fast. There should be a program in this issue called CONVERT. It does those conversions and two others.

A little background. Sometimes you may look at a file and notice that there are no CR's. If you reformat such a document, everything will be jumbled into one big paragraph. TI Writer stops reformatting when it hits a CR. FUNNELWEB stops when it hits a CR or a blank line. Either way, the document is a mess.

CONVERT, when converting a DF128 file to DV80, can add a CR to blank lines, to the end of paragraphs and to lines that start with a period (Formatter commands). This takes a little longer but it makes the file much easier to edit. Also, CONVERT can add CR's to DV80 files that lack them.

#### NOTE TO OTHER USER GROUPS

I have now written 20 XB Columns and 20 in the TI BITS series. From time to time, other user groups have published some of my work in their news letters. If anyone wants a complete set, please send me two (2) DSSD disks or four (4) SSSD disks, a return mailer and return postage. My address is 7301 Kirby Way, Stanton, CA 90680.

Enjoy.

```
100 ! CONVERT
110 ! Version 1.0
120 ! 09 Aug 88
130 ! By Jim Swedlow
140 ! Based on XPREP
    ! By Carl Walters
150 !
160 DISPLAY ERASE ALL :: CAL
L SCREEN(5):: FOR A=0 TO 14
:: CALL COLOR(A,16,1):: NEXT
A
170 FOR A=1 TO 4 :: READ T$(
A):: NEXT A
180 N$=CHR$(13)&CHR$(10):: Z
$=CHR$(26):: C$=CHR$(13):: G
OTO 300
190 DATA DF128 -> DV80 add
CR's,DF128 -> DV80 no CR's,
DV80 -> DV80 add CR's,DV80
-> DF128
200 CALL KEY :: Q$,S,P,K,I$,
W$ :: !@P-
210 !
220 ! STRING CHECK SUB
230 !
240 P=1 :: IF I$=" " OR I$="
" THEN I$="" :: RETURN ELSE
IF ASC(I$)=46 THEN RETURN EL
SE P=0 :: RETURN
250 !
260 ! CLOSE FILES AND END
270 !
280 CLOSE #1 :: CLOSE #2 ::
DISPLAY AT(19,1)BEEP:"DONE"
290 FOR P=1 TO 100 :: NEXT P
300 !
310 ! TITLE SCREEN
320 !
330 DISPLAY AT(5,5):"CONVERT
Version 1.0" :: : : : :
:"Press For"
340 FOR S=1 TO 4 :: DISPLAY
AT(14+S,1):STR$(S); " ";T$(S)
:: NEXT S :: DISPLAY AT(19,1
)BEEP:"5 End Program"
```

```
350 !
360 ! PICK FUNCTION
370 !
380 CALL KEY(0,K,S):: IF K<4
9 OR K>53 THEN 380 ELSE K=K-
40 :: IF K=5 THEN DISPLAY ER
ASE ALL :: STOP
390 DISPLAY AT(13,1):T$(K):
!"Input File: DSK":: "Outp
ut File: DSK":: : :
400 ACCEPT AT(15,18)BEEP:I$
410 ACCEPT AT(17,18)BEEP:W$
420 !
430 ! OPEN FILES & INIT
440 !
450 DISPLAY AT(19,1):"Workin
g . . ."
460 IF K>2 THEN OPEN #1:"DSK
"&I$,INPUT ELSE OPEN #1:"DSK
"&I$,INPUT ,FIXED 128
470 IF K=4 THEN OPEN #2:"DSK
"&W$,OUTPUT,FIXED 128 ELSE O
PEN #2:"DSK"&W$,OUTPUT
480 A=1 :: W$="" :: ON K GOT
O 720,570,490,650
490 !
500 ! DV80 -> DV80 ADD CR's
510 !
520 LINPUT #1:I$ :: GOSUB 21
0 :: IF EOF(1)THEN 550
530 IF A THEN IF P THEN PRIN
T #2:I$;C$ :: GOTO 520 ELSE
Q$=I$ :: A=0 :: GOTO 520
540 IF P THEN PRINT #2:Q$;C$
:I$;C$ :: A=1 :: GOTO 520 EL
SE PRINT #2:Q$ :: Q$=I$ :: G
OTO 520
550 IF A=0 THEN IF P THEN PR
INT #2:Q$;C$ ELSE PRINT #2:Q
$
560 PRINT #2:I$;C$;C$ :: GOT
O 250
570 !
580 ! DF128 -> DV80 NO CR'S
590 !
600 LINPUT #1:I$ :: W$=W$&I$
:: K=1 :: S=LEN(W$)
610 IF SEG$(W$,K,1)=Z$ THEN
250 ELSE IF K>S THEN IF EOF(
1)THEN 250 ELSE W$="" :: GOT
O 600
620 P=POS(W$,N$,K):: IF P TH
EN PRINT #2:SEG$(W$,K,P-K)::
K=P+2 :: GOTO 610
630 P=POS(W$,Z$,K):: IF P TH
EN PRINT #2:SEG$(W$,K,P-K)::
GOTO 250
640 W$=SEG$(W$,K,255):: IF E
OF(1)THEN PRINT #2:W$ :: GOT
O 250 ELSE 600
650 !
660 ! DV80 -> DF128
670 !
680 LINPUT #1:I$ :: IF ASC(I
$)=128 THEN I$=""
690 W$=W$&I$&N$ :: P=LEN(W$)
700 IF P>128 THEN PRINT #2:S
EG$(W$,1,128):: W$=SEG$(W$,1
29,255)
710 IF EOF(1)THEN PRINT #2:W
$&Z$ :: GOTO 250 ELSE 680
720 !
730 ! DF128 -> DV80 ADD CR'S
740 !
750 LINPUT #1:I$ :: W$=W$&I$
:: K=1 :: S=LEN(W$)
760 IF SEG$(W$,K,1)=Z$ THEN
820 ELSE IF K>S THEN IF EOF(
1)THEN 820 ELSE W$="" :: GOT
O 750
770 P=POS(W$,N$,K):: IF P TH
EN I$=SEG$(W$,K,P-K):: K=P+2
ELSE 800
780 GOSUB 210 :: IF A THEN I
F P THEN PRINT #2:I$;C$ :: G
OTO 760 ELSE Q$=I$ :: A=0 ::
GOTO 760
790 IF P THEN PRINT #2:Q$;C$
:I$;C$ :: A=1 :: GOTO 760 EL
SE PRINT #2:Q$ :: Q$=I$ :: G
OTO 760
800 P=POS(W$,Z$,K):: IF P TH
EN I$=SEG$(W$,K,P-K):: GOTO
820
810 W$=SEG$(W$,K,255):: IF E
OF(1)THEN I$=W$ ELSE 750
820 IF A=0 THEN GOSUB 210 ::
IF P THEN PRINT #2:Q$;C$ EL
SE PRINT #2:Q$
830 PRINT #2:I$;C$;C$ :: GOT
O 250
```



## WHY SHOULD YOU LEARN TO PROGRAM?

by Jim Peterson

Why should you learn to program? To make money? No way! If you could write a program to guarantee world peace, eliminate hunger and cure AIDS, you couldn't make money selling it to the TI world!

Why should you learn to program? To contribute something to the TI world? OK, but don't expect any thanks! Contributing a program to the TI public domain is like dropping a pebble into a bottomless dry well - you will never hear a splash, not even a thud.

Why should you learn to program? Because no one has written the program you need? Well, now you have a good reason! Since there is neither money nor recognition in programming, the programmers tend to write what they feel like writing, not what you want them to write.

Why should you learn to program? For one reason, because I know that you would like to make some changes in the programs that you use frequently. I know that, because the only feedback I ever get is from people who wish that I would change this or that! You really wouldn't have to learn very much to change colors, add or silence a beep or a burp, output to disk instead of printer, etc., etc.

Beyond that, unraveling someone else's code can be tricky and frustrating (and I pity anyone who tries to unravel my code!) Often I find it easier to just rewrite the basic idea in my own way.

If you do modify someone's program, please put a note on the title screen, or at least in a REM, that you did so - and unless you are very sure that you have not introduced a bug, don't distribute your version! Programmers do not like to be blamed for other people's mistakes, and the sales of good programs have been ruined by the bad reputation resulting from pirated, modified and bugged copies.

But, the real reason for learning to program - it's fun, it's challenging, it's creative! There is something very satisfying about getting an idea to make the computer do something it has never done (as far as you know!) and then succeeding in making it do what you want. There is a thrill in pushing the limits of that obsolete tiny TI pea brain just a little bit farther.

There are those who prefer to exercise their creativity with the soldering iron, those who can plug and soup up a Model T computer to run like a Ferrari. I regard them with awe and wonderment, and I'm glad they are around. Without them, I wouldn't have my RamDisk, and my equipment wouldn't get repaired.

Personally, I am the ultimate klutz. If I approach my car with a screwdriver, all four tires go flat. My one feeble attempt to repair my P-Box resulted in failure, expense and embarrassment. But, without having more than a faint idea what goes on beneath that keyboard, I have learned to punch the keys (two right fingers and a left thumb) and create hundreds of programs and routines which have given me a great deal of satisfaction.

It's been fun! You should try it sometime.

INFOCOM was a company that made text-only adventures of mind boggling complexity, and many were released for the TI99/4A. Later larger adventures required a double sided disk, and were not officially released for the TI but can be found. Some of these were so large they required also extra RAM at >6000. Here is a full list of Infocom adventures for the TI.

CODE: YEAR: AUTHOR: LEVEL: REQ: NAME:

```
A1.....83.....15.....3.....S.....INFIDEL
A3.....84.....15+12...2.....S.....CUTTHROATS
A4.....86.....7.....2.....D.....HOLLYWOOD HIJINX (Treasure hunt)
C1.....86.....17.....2.....R.....LEATHER GODDESSES OF PHOBOSS
H1.....87.....5.....-.....D.....LURKING HORROR (Login 99. Password T
I. Clue)
K1.....84.....16+10...1.....R.....SEASTALKER
M1.....82.....14.....4.....S.....DEADLINE
M2.....83.....16.....2.....S.....WITNESS
M3.....84.....5.....3.....S.....SUSPECT
M4.....86.....11.....2.....D.....BALLYHOOD (at the circus!)
M5.....86.....16+10...1.....R.....MOONMIST
R1.....87.....1.....-.....D.....PLUNDERED HEARTS
B1.....82.....5.....4.....S.....STARCROSS
B2.....83.....15.....4.....S.....SUSPENDED
B3.....83.....17.....2.....S.....PLANETFALL
B4.....84.....17+4...2.....S.....HITCHHIKERS GUIDE TO THE GALAXY
B6.....87.....17.....-.....D.....STATIONFALL
Z0.....85.....3.....1.....D.....WISHBRINGER
Z1.....80.....14+5...2.....S.....ZORK 1
Z2.....81.....14+5...3.....S.....ZORK 2
Z3.....82.....14+5...3.....S.....ZORK 3
Z4.....83.....14+5...2.....S.....ENCHANTER
Z5.....84.....17.....3.....S.....SORCEROR
Z6.....85.....5.....4.....D.....SPELLBREAKER
-----S.....SAMPLER
```

Authors: 1. Amy Briggs 3. Brian Moriarty. 4. Douglas Adams  
5. Dave Lebling  
7. Dave Anderson. 10. Jim Lawrence. 11. Jeff O'Neill.  
12. Jerry Wolper. 14. Marc Blank. 15. Michael S Berlyn.  
16. Stu Galley. 17. Steve Meretzky.

Level: Some games were released without level indication.  
1=Easiest 4=Expert.

Req: S-Standard TI disk system. D-Double sided disk drive.

R-Double sided disk system plus ram at >6000 .

Note that Code A2 was not used, and some codes are not available on TI format.

In 1986 Infocom was bought by Activision, now calling itself Mediagenic. They have now formally written off their purchase for nine million dollars, and while they still own the name, there will be no more Infocom adventures as they are now known. (Information taken from Oct/Nov issue of New Atari User).

This next program is definitely fractal in nature but also fairly short, indeed I will first present it as a one liner - this requires THE MISSING LINK but can be translated easily to any other language giving pixel graphics:

```
10 FOR X=1 TO 170 :: FOR Y=1 TO 180 :: T=((X/32)^2)*Y/32 :: IF T AND
1 THEN CALL LINK("PIXEL",X,Y) :: NEXT Y :: NEXT X :: GOTO 10
```







```

1540 CALL HCHAR(21,18,32)
1560 CALL KEY(0,A,B)
1580 IF B=0 THEN 1520
1600 IF A=ASC("A") THEN 1660
1620 IF A=ASC("B") THEN 2220
1640 GOTO 1500
1660 CALL HCHAR(22,31,30)
1680 CALL HCHAR(22,31,32)
1700 CALL HCHAR(23,31,30)
1720 CALL HCHAR(23,31,32)
1740 CALL KEY(0,A,B)
1760 IF B=0 THEN 1660

1780 REM MOVE SQUARE A
1800 IF A=ASC("1") THEN 1860
1820 IF A=ASC("2") THEN 2040
1840 GOTO 1660

1860 REM MOVE A CLOCKWISE
1880 CALL GCHAR(RA(1),CA(1),
TEMPA)
1900 FOR C=1 TO 19
1920 CALL GCHAR(RA(C+1),CA(C
+1),TEMPB)
1940 CALL HCHAR(RA(C+1),CA(C
+1),TEMPA)
1960 TEMPB=TEMPB
1980 NEXT C
2000 CALL HCHAR(RA(1),CA(1),
TEMPA)
2020 GOTO 1500

2040 REM MOVE A ANTICLOCKWIS
E
2060 CALL GCHAR(RA(20),CA(20
),TEMPA)
2080 FOR C=19 TO 1 STEP -1
2100 CALL GCHAR(RA(C),CA(C),
TEMPB)
2120 CALL HCHAR(RA(C),CA(C),
TEMPA)
2140 TEMPB=TEMPB
2160 NEXT C
2180 CALL HCHAR(RA(20),CA(20
),TEMPA)
2200 GOTO 1500
2220 CALL HCHAR(22,31,30)
2240 CALL HCHAR(22,31,32)
2260 CALL HCHAR(23,31,30)
2280 CALL HCHAR(23,31,32)
2300 CALL KEY(0,A,B)
2320 IF B=0 THEN 2220
2340 IF A=ASC("1") THEN 2400
2360 IF A=ASC("2") THEN 2580
2380 GOTO 2220

2400 REM MOVE B CLOCKWISE

```

```

2420 CALL GCHAR(RB(1),CB(1),
TEMPA)
2440 FOR C=1 TO 19
2460 CALL GCHAR(RB(C+1),CB(C
+1),TEMPB)
2480 CALL HCHAR(RB(C+1),CB(C
+1),TEMPA)
2500 TEMPB=TEMPB
2520 NEXT C
2540 CALL HCHAR(RB(1),CB(1),
TEMPA)
2560 GOTO 1500

2580 REM MOVE B ANTI-
CLOCKWISE
2600 CALL GCHAR(RB(20),CB(20
),TEMPA)
2620 FOR C=19 TO 1 STEP -1
2640 CALL GCHAR(RB(C),CB(C),
TEMPB)
2660 CALL HCHAR(RB(C),CB(C),
TEMPA)
2680 TEMPB=TEMPB
2700 NEXT C
2720 CALL HCHAR(RB(20),CB(20
),TEMPA)
2740 GOTO 1500
2760 END

2780 REM (C)1982
2800 REM BY STEPHEN SHAW
2820 REM 10 ALSTONE ROAD
2840 REM STOCKPORT CHESHIRE

2860 REM SK4 5AH
2880 REM
2900 REM *****

2920 END

```

## DESIGNS FOR FUN

Originally written for TI Basic by Peter Brookes and published in TIDINGS in June 1982. Rewritten for Extended Basic plus The Missing Link by Stephen Shaw 1990.

### Main Menu:

0. Set Defaults... resets all parameters to the original default values.
1. Check Settings.. displays current parameter values
2. Select Pattern Area.. each block of pattern may be set to an area of 1, 4, 9, 16 or 25 characters- from 8x8 pixels to 40x40 pixels.
3. Select Colours.. Only operational if The Missing Link is loaded and set up for 16 colour use. Sets screen, foreground and background colours.
4. Select Bias.. Allows selection from 16 predefined biases-for more use option 5. See description below.. (Default 01234789CEF)
5. Create Bias.. Allows input of a bias string from 2 to 112 characters, composed of the numbers 0 to 9 and the letters A to F.

The program is designed so that the selection of combinations at random is made from the bias string. If the string contains the entire range of combinations (0 to F) the program will randomly select any of the 16 possible combinations of 4 pixels. If however the bias string contained only 1248 the program would only be able to select from those patterns, thus biasing the result.

6. Reset Randomiser.. It is the nature of the beast that it may become cyclic-probably with a large cycle! Using this option will break the cycle and shift the pattern generation to another range.

7. Select Mode.. either continuous or pattern will stay on screen until you press a key.

8. Continue.. get the patterns going.

### During pattern generation:

Press and hold key S to return to main menu.

User prompted pattern change: any key except S & A for next pattern.

To print: Hold CTRL and FCTN together.

To save in TI Artist format: Once the pattern is on screen and before the definitions at screen bottom start to change, press and hold key A. Indicate drive number, then file name (up to 8 characters, omit \_P). Easier to use if user prompt mode is specified.

Patterns saved in TI Artist format can then be cut out into Instances and used for texture or repeated over the full screen as required before printing. Can be used for book jackets or fly leaves or even wallpaper! Enlarge and use as a crosswork design. etc etc.

As listed the equivalent character definitions are given at screen bottom- if you would like to save your TI Artist files without these then amend the following line:

2730 FOR @=1 TO 24 :: etc etc

Or to view a full screen pattern after the new "characters" have been displayed, make the following amendment:

2630 FOR L=1 TO 24

The characters are numbered from left to right first then from top to bottom, the number pattern at right indicates how the 5x5 grid is numbered!

13 14 15 16

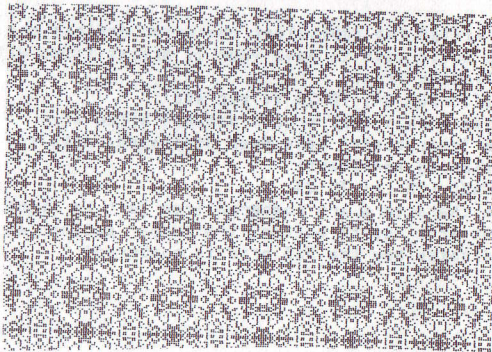
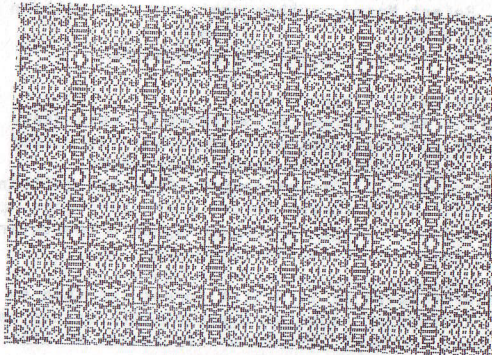
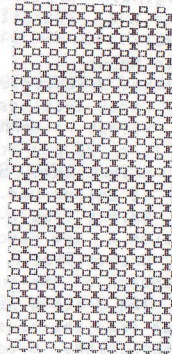


```

100 REM DESIGNS FOR FUN
110 REM TO TML S SHAW 1990
120 CALL LINK("CLEAR")
130 CALL LINK("CHSIZE",8,8)
140 OPTION BASE 1
150 DIM C$(24),D$(40),M$(2)
160 H$="0123456789ABCDEF"
170 I$="084C2A6E195D3B7F"
180 L$="54433322"
190 M$(1)="CONTINUOUS"
200 M$(2)="USER-CUED"
210 DEF R(X)=INT(RND*X)+1
220 FOR L=96 TO 127
230 CALL LINK("CHAR",L,"00013B3323033F7F")
240 NEXT L
250 GOTO 2270
260 FOR L=1 TO Z*8
270 D$(L)=""
280 NEXT L
290 S=LEN(S$)
300 FOR L=1 TO Z*4
310 FOR M=1 TO Z
320 A$=SEG$(S$,R(S),1)
330 D$(L)=A$&D$(L)&SEG$(I$,POS(H$,A$,1),1)
340 CALL KEY(1,K,T)
350 IF K=2 THEN 550
360 IF K=1 THEN GOSUB 2670
370 NEXT M
380 D$(Z*8-L+1)=D$(L)
390 NEXT L
400 Q=96
410 FOR L=1 TO Z*8 STEP 8
420 FOR M=1 TO 2*Z-1 STEP 2
430 A$=""
440 FOR N=L TO L+7
450 A$=A$&SEG$(D$(N),M,2)
460 CALL KEY(1,K,T)
470 IF K=2 THEN 550
480 NEXT N
490 CALL LINK("CHAR",Q,A$)
500 GOSUB 2540
510 Q=Q+1
520 NEXT M
530 NEXT L
540 GOSUB 2610
550 RETURN
560 FOR L=1 TO 24
570 C$(L)=""
580 NEXT L
590 FOR L=1 TO Z
600 FOR M=1 TO Z
610 C$(L)=C$(L)&CHR$(95-Z+L*Z+M)
620 NEXT M
630 NEXT L
640 FOR L=1 TO Z
650 FOR M=1 TO VAL(SEG$(L$,Z,1))
660 C$(L)=C$(L)&C$(L)
670 NEXT M

```

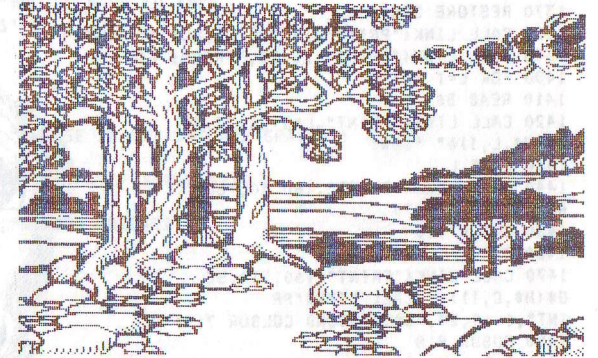
(C) PETE BROOKS 1982 V1.0



```

680 C$(L)=SEG$(C$(L),1,28)
690 NEXT L
700 FOR L=Z+1 TO 24
710 C$(L)=C$(L-Z)
720 NEXT L
730 RETURN
740 FOR L=1 TO 10
750 CALL LINK("PRINT",L*8-7,1,C$(L))
760 NEXT L
770 RETURN
780 Z=3
790 GOSUB 560
800 A=1
810 S$="01234789CEF"
820 C=15
830 B=15
840 F=2
850 CALL SCREEN(C)
860 REM
870 CALL LINK("COLOR",F,B)
880 REM
890 CALL SOUND(100,1220,0)
900 RETURN
910 CALL SOUND(100,1220,0)
920 CALL KEY(0,K,T)
930 IF T<1 THEN 920
940 P=POS(H$,CHR$(K),1)
950 IF P=0 THEN 920
960 CALL SOUND(100,1220,0)
970 RETURN
980 CALL LINK("CLEAR")
990 CALL LINK("PRINT",2,20,"CURRENT SETTINGS")
1000 RESTORE 2480
1010 FOR L=1 TO C
1020 READ B$
1030 NEXT L
1040 CALL LINK("PRINT",12,20,"SCREEN IS "&B$)
1050 RESTORE 2480
1060 FOR L=1 TO B
1070 READ B$
1080 NEXT L
1090 CALL LINK("PRINT",22,20,"BACKGROUND IS "&B$)
1100 RESTORE 2480
1110 FOR L=1 TO F
1120 READ B$
1130 NEXT L
1140 CALL LINK("PRINT",32,20,"FOREGROUND IS "&B$)
1150 CALL LINK("PRINT",42,20,"BIAS IS "&S$)::
CALL LINK("PRINT",52,20,"PATTERN REA IS "&STR$(Z*Z))
1160 CALL LINK("PRINT",62,20,"MODE IS "&M$(A))
1170 CALL LINK("PRINT",72,20,"PRESS 'S' TO CONTINUE")
1180 CALL KEY(1,K,T)
1190 IF K<>2 THEN 1180
1200 CALL SOUND(100,1220,0)
1210 RETURN
1220 CALL LINK("CLEAR")
1230 CALL LINK("PRINT",20,20,"PATTERN AREA")::
CALL LINK("PRINT",30,20,"PRESS: OR:")
1240 FOR L=1 TO 5
1250 CALL LINK("PRINT",30+L*9,20,STR$(L)&" "&
STR$(L)&" "&("&STR$(L)&" X "&STR$(L)&")")

```





```

1260 NEXT L
1270 CALL LINK("PRINT",120,20,"YOUR SELECTION ?:")
1280 CALL SOUND(100,1220,0)
1290 CALL KEY(0,K,T)
1300 IF (K<49)+(K>53) THEN 1290
1310 CALL SOUND(100,1220,0)
1320 Z=K-48
1330 CALL LINK("PRINT",140,90,Z)
1340 GOSUB 560
1350 RETURN
1360 CALL LINK("CLEAR")
1370 RESTORE 2480
1380 CALL LINK("PRINT",12,20,"SELECT COLOURS")
1390 CALL LINK("PRINT",22,20,"PRESS: FOR:")
1400 FOR L=1 TO 16
1410 READ B#
1420 CALL LINK("PRINT",L*8+26,23,SE
G$(H$,L,1)&" "&B#)
1430 NEXT L
1440 CALL LINK("PRINT",160,40,"SC
REEN COLOUR ?:")
1450 GOSUB 910
1460 C=P
1470 CALL LINK("PRINT",160,220,SE
G$(H$,C,1)): CALL LINK("PR
INT",169,20,"BACKGROUND COLOUR ?:")
1480 GOSUB 910
1490 B=P
1500 CALL LINK("PRINT",169,220,SEG$(H$,B,1)):
CALL LINK("PRINT",178,20,"DIFFERE NT FOREG
ROUND COLOUR ?:")
1510 GOSUB 910
1520 IF B=P THEN 1510
1530 F=P
1540 CALL LINK("PRINT",178,220,SEG$(H$,F,1))
1550 FOR L=1 TO 200
1560 NEXT L
1570 RETURN
1580 CALL LINK("CLEAR")
1590 RESTORE 2520
1600 CALL LINK("PRINT",10,20,"BIAS SELECTION")
:: CALL LINK("PRINT",20,20,"PRESS: FOR:")
1610 FOR L=1 TO 16
1620 READ S#
1630 CALL LINK("PRINT",L*8+22,30,SE
G$(H$,L,1)&" "&S#)
1640 NEXT L
1650 CALL LINK("PRINT",170,20,"YOUR SELECTION ?:")
1660 GOSUB 910
1670 CALL LINK("PRINT",170,220,SEG$(H$,P,1))
1680 FOR L=1 TO 200
1690 NEXT L
1700 RETURN
1710 CALL LINK("CLEAR")
1720 CALL LINK("PRINT",10,10,"BIAS CREATION")
1730 REM
1740 CALL LINK("PRINT",22,20,"USE 0 TO 9 AND A TO F")
1750 REM

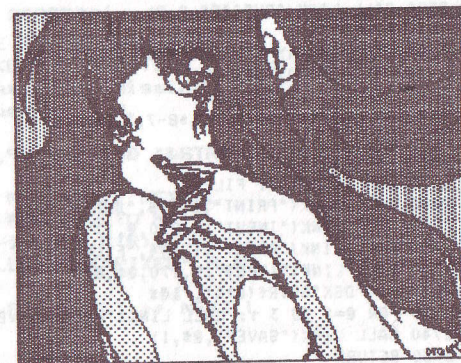
```



```

1760 CALL LINK("PRINT",32,20,"INPUT BETWEEN 2 & 112 DI
GITB LONG")
1770 CALL LINK("PRINT",62,20,"PLEASE ENTER YOUR ST
RING I- ")
1780 CALL LINK("INPUT",70,20,S$,112)
1790 CALL LINK("CLEAR")
1800 CALL LINK("PRINT",100,20,"CHECKING...")
1810 IF LEN(S#)>1 THEN 1850
1820 CALL LINK("PRINT",109,20,"YOU'VE MADE A MISTAKE")
1830 CALL LINK("PRINT",119,1,"YOU'LL HAVE TO INCR
PAGE THE LENGTH OF YOUR STRING ")
1840 GOTO 1760
1850 FOR L=1 TO LEN(S#)
1860 A# =SEG$(S#,L,1)
1870 IF A#<>" " THEN 1900
1880 CALL LINK("PRINT",109,20,"PLEASE DO NOT INCLUDE SPACES")
1890 GOTO 1750
1900 IF (A#<"0")+(A#>"9")*(A#<"A")+(A#>"F") THEN 1750
1910 NEXT L
1920 CALL LINK("PRINT",150,50,"STRING CHECKED")
1930 FOR L=1 TO 200
1940 NEXT L
1950 RETURN
1960 CALL LINK("CLEAR")
1970 RANDOMIZE
1980 CALL SOUND(100,1220,0)
1990 RETURN
2000 CALL LINK("CLEAR")
2010 CALL LINK("PRINT",10,20,"SELECT MODE"):
CALL LINK("PRINT",20,20,"PRESS: OR:")
2020 FOR L=1 TO 2
2030 CALL LINK("PRINT",40+L*10,40,STR$(L)&" "&M$(L))
2040 NEXT L
2050 CALL LINK("PRINT",80,30,"YOUR SELECTION ?:")
2060 CALL KEY(0,K,T)
2070 IF (K<49)+(K>50) THEN 2060
2080 A=K-48
2090 CALL LINK("PRINT",90,120,A)
2100 CALL SOUND(100,1220,0)
2110 FOR L=1 TO 200
2120 NEXT L
2130 RETURN
2140 CALL LINK("CLEAR")
2150 GOSUB 850
2160 GOSUB 740
2170 GOSUB 260
2180 CALL KEY(1,K,T)
2190 IF K=2 THEN 2260
2200 IF K=1 THEN GOSUB 2670
2210 ON A GOTO 2170,2220
2220 CALL KEY(1,K,T)
2230 IF T<1 THEN 2220
2240 CALL SOUND(100,1220,0)
2250 GOTO 2170
2260 RETURN
2270 GOSUB 780
2280 GOSUB 560

```





```

2290 CALL LINK("CLEAR")
2300 CALL SCREEN(15)
2310 RESTORE 2500
2320 CALL LINK("PRINT",10,20,"MA
IN MENU"):: CALL LINK("PRI
NT",20,20,"PRESS: TO: ")
2330 FOR L=0 TO 8
2340 READ B#
2350 CALL LINK("PRINT",L*9+28,30,STR$(L)&" "&B#)
2360 NEXT L
2370 CALL LINK("PRINT",168,30,"YOUR SELECTION ?:")
2380 CALL SOUND(100,1220,0)
2390 CALL KEY(0,K,T)
2400 IF (K<48)+(K>56) THEN 2390
2410 K=K-47
2420 CALL LINK("PRINT",176,140,K-1)
2430 CALL SOUND(100,1220,0)
2440 FOR L=1 TO 200
2450 NEXT L
2460 ON K GOSUB 780,980,1220,1360,1580,
1710,1960,2000,2140
2470 GOTO 2290
2480 DATA TRANSPARENT, BLACK, MEDIUM GREEN, LI
GHT GREEN, DARK BLUE, LIGHT BLUE, DARK ED, CYAN, MEDIUM RED
2490 DATA LIGHT RED, DARK YELLOW, LIGHT YELLOW, DARK GRE
EN, MAGENTA, GREY, WHITE
2500 DATA SET DEFAULTS, CHECK SETTINGS, SELECT PATTERN A
REA, SELECT COLOURS, SELECT BIAS, CREATE BIAS
2510 DATA RESET RANDOMISER, SELECT MODE, CONTINUE
2520 DATA F2481F4B12F8124F124B,0246BACE13579BDF,FFEEC
C00,FECEB,01234789CEF,1248,8 CEF137F,11333311333311CB
2530 DATA B421B421B421,124812481248,9AC953F,1122448B,
07E07E07E,DB00DB00,969696F0 8181,1032547698BADCFE
2540 REM PUT CHARDEFS ON SCREEN
2550 CALL LINK("CHSIZE",6,7):: CHARNO=Q-95 ! 1 TO 25
2560 NEG=8*INT(Z*2/2+2) ! 1 TO 13 *8
2570 ROW=193-NEG+INT((CHARNO-1)/2)*8
2580 COL=1-((Q/2)<>INT(Q/2))*114 :: CALL LINK("PRI
NT",ROW,COL,STR$(CHARNO)&":"&A $)
2590 CALL LINK("CHSIZE",8,8)
2600 RETURN
2610 REM DISPLAY NEW PATTERN
2620 BOTTOM=(192-NEG)/8
2630 FOR L=1 TO BOTTOM
2640 CALL LINK("PRINT",L*8-7,1,C$(L))
2650 NEXT L
2660 RETURN
2670 REM TO ARTIST FILE
2680 CALL LINK("PRINT",1,196,"DSK")
2690 CALL LINK("INPUT",1,220,@,1)
2700 CALL LINK("PRINT",1,194,"FILE ")
2710 CALL LINK("INPUT",9,170,@$,8)
2720 @#="DSK"&STR$(@)&". "&@#
2730 FOR @=1 TO 3 :: CALL LINK("PRINT",@*8-7,1,C$(@)):: NEXT @
2740 CALL LINK("SAVEP",@$,1)
2750 RETURN

```

In Dr Pickover's excellent book COMPUTERS PATTERN CHAOS AND BEAUTY he gave a short routine for producing a graphic result from pascals triangle, which did not work for me so I left it. I found the original article in his Journal of Chaos and Graphics for August 1988 and this encouraged me to have another look. The program is based on plotting pixels when an array element takes a zero value, but as the program started off initialising the array to zero values, and as MOD(zero,N) is always zero, I kept getting a solid black triangle... then I thought of initialising to a value of 1, which is appropriate to the triangle... what is pascals triangle? It is formed of numbers like this:

|  |  |   |   |    |   |      |       |
|--|--|---|---|----|---|------|-------|
|  |  | 1 |   | OR |   | 1    |       |
|  |  | 1 | 1 |    |   | 11   |       |
|  |  | 1 | 2 | 1  |   | 121  |       |
|  |  | 1 | 3 | 3  | 1 | 1331 |       |
|  |  | 1 | 4 | 6  | 4 | 1    | 14641 |

The right hand form is merely a reformat which makes computer graphics easier! Note that each number is the sum of each of the two numbers above it, assuming a value of 1 along each side.

The triangle has a number of fascinating mathematical applications, the one I first met with in school was the result of (A+B)^N. If N is two, that is we multiply (a+b) by (a+b) then we get a^2+2ab+b^2, a 1-2-1 pattern.

Similarly, (a+b)^3 is a^3+3a^2b+3ab^2+b^3, a 1-3-3-1 pattern, and so on down the triangle.

There are many other applications, but here is a graphic one for the computer. Lets examine each number in turn, and if it is exactly divisible by a number of our choosing, turn a pixel on in that position or leave it off.

Thus looking to the right hand format, and using a divisor of 2, we would have a pattern, using 0 for pixel off, X for pixel on, of:

```

0
00
0X0
0000
0XXX0 and so on.

```

Let's put that into a program!

```

1 REM PASCALS TRIANGLE BY MODULAR MEANS
2 ! FROM CLIFFORD PICKOVER. Modified S Shaw Dec 90
3 ! for TI99/4A + ExBas + The Missing Link.
4 ! try different values if DIV.
90 DIV=4
100 CALL LINK("PRINT",90,15,"MOD "&STR$(DIV))
110 DIM P(190),C(190)
120 FOR L=0 TO 190 :: P(L)=1 :: NEXT L
130 FOR N=2 TO 190 :: FOR R=2 TO N
140 CALL MOD(P(R)+P(R-1),DIV,C(R))
150 IF C(R)=0 THEN CALL LINK("PIXEL",R,N)
160 NEXT R
170 FOR L=1 TO 190 :: P(L)=C(L) :: NEXT L
180 NEXT N
190 GOTO 190
200 SUB MOD(A,B,OP) :: OP=INT(A-B*INT(A/B))
210 SUBEND

```

or if you wish to look at a much larger triangle we can output straight to our epson compatible printer like this:



```

100 DIM P(470),C(470)
110 G$=CHR$(27)&"K"&CHR$(470-256)&CHR$(1)
120 FOR L=0 TO 470 :: P(L)=1 :: NEXT L
130 OPEN #1:"PID.CR"
140 PRINT #1:CHR$(27);"A";CHR$(1);CHR$(10);CHR$(13)
150 DIV=15
160 FOR N=2 TO 470 :: PRINT #1:G$&CHR$(0)&CHR$(0)
170 FOR R=2 TO N :: CALL MOD(P(R)+P(R-1),DIV,C(R))
180 IF C(R)=0 THEN PRINT #1:CHR$(2); ELSE PRINT #1:CHR$(0);
190 NEXT R
200 FOR L=2 TO 470-N :: PRINT #1:CHR$(0); :: NEXT L
210 FOR L=1 TO 470 :: P(L)=C(L) :: NEXT L
220 PRINT #1:CHR$(10);CHR$(13)
230 NEXT N
240 CLOSE #1 :: STOP
250 SUB MOD(A,B,OP)
260 OP=INT(A-B*INT(A/B))
270 SUBEND

```

```

=====
1 REM WHAT IS NEXT NUMBER...
2 ! 3 5 7 9 ?
3 ! 9 16 25 36 ?
4 ! 8 27 64 125 ?
5 ! 4 7 11 16 22 29 ?
6 !
7 ! write a program to find the number ? in each case.
8 !
9 ! Ecker.86. Shaw.91.
100 CALL CLEAR
110 DIM A(30,30)
120 PRINT "INPUT A SERIES OF NUMBERS"
130 PRINT "INPUT -99 TO TERMINATE- THIS
WILL NOT FORM PART OF THE SEQUENCE
140 J=1
150 INPUT A(1,J)
160 IF A(1,J)=-99 THEN A(1,J)=0 :: J=J-1 :: GOTO 190
170 J=J+1
180 GOTO 150
190 N=J
200 FOR K=2 TO N
210 FOR J=1 TO N+1-K
220 A(K,J)=A(K-1,J+1)-A(K-1,J)
230 NEXT J
240 NEXT K
250 A(N,2)=A(N,1)
260 FOR L=N-1 TO 1 STEP -1
270 A(L,N-L+2)=A(L,N-L+1)+A(L+1,N-L+1)
280 NEXT L
290 PRINT "NEXT NUMBER IS ";A(1,N+1)
300 GOTO 120
301 ! not all series will be solved with this program.
=====

```

The disk library has recently received the text of the New Testament on disk. I thought it would be nice to have the computer read it to me... hence the following program. It has been put together specifically to deal with the Bible text, and only takes account of the letters A to Z, in upper OR lower case.

It is written to be used with the Terminal Emulator 2 (TE2) module and of course requires the speech synthesiser peripheral.

Using TE2 we program in Basic, and cannot use the XB form of LINPUT A\$ to read the text from disk. As some punctuation commonly found in text represents "end of record" when using Display format files, we have added a comma to the INPUT line, which ensures that we miss no text. In XB LINPUT ignores the commas and so on (=LINE INPUT).

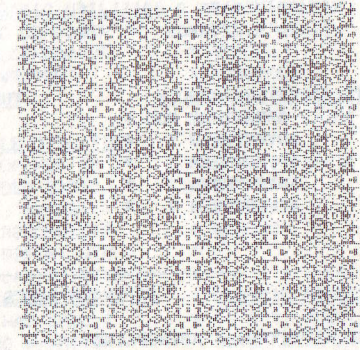
Apart from the need for the comma after A\$ there is nothing too strange about this little program. As TE2 only "speaks" upper case we have to go through the string to alter it all to upper case, and for the purpose of the intended usage, certain characters are excluded from the string to be spoken. The chapter and verse numbers used in the Bible are very out of place if you insert them as you read!

Hope you find this of use!

```

100 REM READ DV80 FILE
110 REM WITH EMBEDDED
120 REM AND LOWER CASE
130 REM AND USE TE2 TO READ
140 REM IT OUT LOUD.
150 REM THIS PROGRAM ONLY
160 REM SPEAKS A-Z and a-z
170 REM NUMBERS ARE REMOVED
180 REM RECOMMEND THAT TEXT
IF AT ALL POSSIBLE FOR
SMALLER PAUSES.
190 REM
200 REM S SHAW FEB 91
210 REM
220 OPEN #1:"DSK3.LUKE1",INPUT
230 OPEN #2:"SPEECH",OUTPUT
240 INPUT #1:A$,
250 IF LEN(A$)<1 THEN 240
260 PRINT A$
270 A$=" "%A$
280 FOR T=1 TO LEN(A$)
290 A=ASC(SEG$(A$,T,1))
300 IF A>57 THEN 320
310 IF A<>32 THEN 360
320 IF A<95 THEN 350
330 A=A-32
340 IF A=61 THEN 360
350 B$=B$&CHR$(A)
360 NEXT T
370 A$=B$
380 B$=""
390 PRINT #2:A$
400 IF EOF(1) THEN 420
410 GOTO 240
420 CLOSE #1
430 CLOSE #2
440 END
=====

```



**Back Issues of TI\* MES For Sale**

Nos. 4,8,9,10,11,12,13,14,15,16,17,18,19,20. All for £12.50 inc. P&P or offers. Les Watson, 0245-72572 After 18.00 hrs.



PHONE FOR PRICES AND POSTAGE.  
UNLESS OTHERWISE STATED, ALL NEED EXTENDED BASIC AND 32K EXPANSION.

| ENTERTAINMENT DISKS  | MODULE BACK-UP DISKS           | GENERAL ITEMS      |
|----------------------|--------------------------------|--------------------|
| AIRLINE *            | ARCTURUS *                     | BRIDGE BIDDING 1 * |
| BACKGAMMON *         | BIGFOOT *                      | BRIDGE BIDDING 2 * |
| BARRAGE *            | BUCK ROGERS *                  | BRIDGE BIDDING 3 * |
| BEANSTALK ADV. *     | BURGER BUILDER *               | DISC DOCTOR *      |
| BIG TEXAS SPY ADV. * | DEFENDER *                     | DISC REPAIR KIT *  |
| CAVERNS *            | ESPIAL *                       | EDITOR/ASSEMBLER * |
| COLUMN ATTACK *      | INVADERS *                     | ELECTRICAL ENG. *  |
| DRAW POKER *         | JUMPY *                        | GRAPHING PACK *    |
| ESCAPE FROM BRONT. * | LISSO *                        | GRAPHX *           |
| GAMES 1 *            | MICRO-PINBALL II *             | INVENTORY MAN'NT * |
| GAMES 2 *            | MICRO-TENNIS *                 | INVOICE MAN'NT *   |
| GAMES 3 *            | MIDNIGHT MASON *               | LOGO *             |
| GAMES 4 *            | POLE POSITION *                | LOGO II *          |
| GAMES 5 *            | POPEYE *                       | MAILING LIST *     |
| GAMES 6 *            | TI-TOAD *                      | MATH ROUTINE *     |
| HIGH GRAVITY *       |                                | MINI DISC SAVER *  |
| JOTTO *              | <u>INFOCOM BACK-UP DISKS</u> * | PAGE PRO 99 *      |
| KARATE CHALLENGE *   | CUTTHROATS *                   | PP SIDWAYS *       |
| LEGENDS I *          | DEADLINE *                     | PP FONTS *         |
| LEGENDS II *         | ENCHANTER *                    | PROG. AIDS II *    |
| MINER 2049ER *       | HITCHHIKER *                   | PROG. AIDS III *   |
| MISSION DESTRICT *   | INFIDEL *                      | STRUCTURAL ENG *   |
| ROCK RUNNER *        | PLANETFALL *                   | SUPERBUG II *      |
| WATERWORKS *         | SORCERER *                     | TEXT TO SPEECH *   |
| WIZARD'S DOMINION *  | STARCROSS *                    | TI-EXAM *          |
| WIZARD'S LAIR *      | SUSPENDED *                    | TI-FORTH *         |
| WIZARD'S REVENGE *   | WITNESS *                      |                    |
|                      | ZORK 1,2 or 3 *                |                    |

TI ADVENTURE MODULE

- ADVENTURE MOD. \*
- ADVENTURES 1 \*
- ADVENTURES 2 \*
- ADVENTURES 3 \*
- ADVENTURES 4 \*

TUNNELS OF DOOM MODULE

- T.O.D. MODULE \*
- T.O.D. EDITOR \*
- DOOM GAMES 1 \*
- DOOM GAMES 2 \*
- DOOM GAMES 3 \*

FAIRWARE DISCS

|                         |                       |                         |                         |
|-------------------------|-----------------------|-------------------------|-------------------------|
| ASSAULT THE CITY(TOD) * | ASTROBLITZ *          | AUSSIE GAMES 1 *        | BEST OF UK 1 *          |
| BEST OF UK 2 *          | CERBERUS *            | CHECKERS & BACKGAMMON * | COLOSSAL CAVE(TI MOD) * |
| DEMON DESTROYER *       | FREDDY *              | GHOSTMAN *              | GREAT 4A GAMES 1 *      |
| GREAT 4A GAMES 2 *      | GREAT 4A GAMES 3 *    | GREAT 4A GAMES 4 *      | GREAT 4A GAMES 5 *      |
| GREAT 4A GAMES 6 *      | KINGS CASTLE *        | MAJOR TOM *             | MAZE OF GROG *          |
| MAZOGZ *                | OH MUMMY *            | PERFECT PUSH *          | SARGON CHESS 1 *        |
| SOLITAIRE/SCRABBLE *    | SPACE STATION PHETA * | STRIP POKER *           | SUPER YAHTZEE/WHEEL I * |
| TETRIS *                | THE MINE *            | THE QUEST *             | TI OPOLY *              |
| TI RUNNER II *          | X-RATED GAME *        |                         |                         |
| DIRECTOR *              | DISK MANAGER 1000 *   | DISK+ AID *             | DRAGONSLAYER *          |
| DUMPIT *                | FAMILY ACCOUNT. *     | FORTH TUTOR *           | FUNNELWEB *             |
| INFOCOM RAPID LOADER *  | LOADERS & CATS. *     | MEMORY MANIPULATOR *    | MICRODEX 99 *           |
| PANORAMA *              | SPREAD SHEET *        | TI WRITER MANUAL *      | UNIVERSAL DISASSEMBLE * |

T199/4A LIST: JANUARY 1991  
PHONE FOR PRICES AND POSTAGE

| ENTERTAINMENT MODULES | EDUCATION MODULES            | EXTENDED BASIC CASS.         | TI-BASIC CASSETTES.         |
|-----------------------|------------------------------|------------------------------|-----------------------------|
| 1 AIRLINE *           | * ADD/SUBTRACT1 *            | * A.B.M.CONTROL *            | * 3D RACE *                 |
| 1 BARRAGE *           | * ADD/SUBTRACT2 *            | * ARENA3 *                   | * ADV.MANIA *               |
| 1 BLACK HOLE *        | * ALIEN ADDITION *           | * AIRLINE *                  | * ALIEN ATTACK *            |
| 1 BLASTO *            | * ALLIGATOR MIX *            | * BACKGAMMON *               | * BENEATH/STARS *           |
| 1 CAR WARD *          | * BEGIN. GRAMMAR *           | * BLASTEROIDS *              | * BLACK TOWER *             |
| 1 CHICKEN TRAIL *     | * DECIMALS *                 | * BOUNCER *                  | * BLOOD BANK *              |
| 1 CORRECT FOUR *      | * DEMO DIVISION *            | * DRAW POKER *               | * BOMBER *                  |
| 1 DEFENDER *          | * DIVISION 1 *               | * FAIRWARE 1 *               | * CHALICE *                 |
| 1 DRIBBLE KONG *      | * DRAGON MIX *               | * FAIRWARE 2 *               | * CRAZY GOLF *              |
| 1 DRAGON FLYER *      | * EARLY LEARNING *           | * FAIRWARE 3 *               | * DEVIL'S ISLAND *          |
| 1 FATHOM *            | * EARLY READING *            | * FROGLET *                  | * DRAGON COMBAT *           |
| 1 GARDIAN *           | * MULT'ICATION 1 *           | * GLOBAL RESCUE *            | * DRONE *                   |
| 1 HUNT THE WUMPU *    | * MINUS MISSION *            | * HANG GLIDER *              | * FORBIDDEN CITY *          |
| 1 JUMBLE *            | * NUMBER MAGIC *             | * KONG *                     | * FUNPAC 1&2 *              |
| 1 JUMBLEBREAKER 2 *   | * PRO-TYPER 3 *              | * MANIC MINER *              | * FUNPAC 3 *                |
| 1 JUMBLE HUNT *       | * READING RALLY *            | * MINER '49er *              | * GOLF *                    |
| 1 KARTING *           |                              | * PARCO GOLF *               | * GRAND PRIX *              |
| 1 KICK-STARTER *      |                              | * ROMEO *                    | * HAUNTED HOUSE *           |
| 1 MICRO-TENNIS *      |                              | * SCRAMBLER USA *            | * HUNCHBACK HAVOC *         |
| 1 MINE PACHMAN *      | <u>EDUCATION CASSETTES</u> * | * SUPERHOD *                 | * KAT TRAXX *               |
| 1 MINE PACHMAN *      | * AC CIRCUIT ANF *           | * WALDOBALL *                | * LUNAR LANDER *            |
| 1 MINE PACHMAN 2 *    | * BASIC TUTOR *              | * WIZARD'S LAIR *            | * MANIA *                   |
| 1 MINE PACHMAN 3 *    | * CHILDSPLAY *               | * WIZARDS REVENGE *          | * MARKET SIM'TION *         |
| 1 MINE PACHMAN 4 *    | * ELECT.ENGIN. *             |                              | * MOONBASE 5 *              |
| 1 MINE PACHMAN 5 *    |                              |                              | * OLDIES 1&2 *              |
| 1 MINE PACHMAN 6 *    |                              |                              | * PENGUIN *                 |
| 1 MINE PACHMAN 7 *    | * GAMEWRITER 1 *             | <u>EXT.BASIC UTILITIES</u> * | * PILOT *                   |
| 1 MINE PACHMAN 8 *    | * GAMEWRITER 2 *             | * BASIC COMPILER *           | * RESCUE *                  |
| 1 MINE PACHMAN 9 *    | * HOP-ON *                   | * CHAR.GENERATOR *           | * ROBOPODS *                |
| 1 MINE PACHMAN 10 *   | * HATH ROUTINE *             | * TYPEWRITER *               | * SANTA & GOBLINS *         |
| 1 MINE PACHMAN 11 *   | * MUSIC SKILLS *             |                              | * SENGOKU JIDAI *           |
| 1 MINE PACHMAN 12 *   | * PROGRAM. AIDS *            |                              | * SKI *                     |
| 1 MINE PACHMAN 13 *   | * STARTER PACK 1 *           | <u>MODULE UTILITIES</u> *    | * SORC'S CASTLE *           |
| 1 MINE PACHMAN 14 *   | * STARTER PACK 2 *           | * DESKTOP PUB'ER *           | * TI-TREK *                 |
| 1 MINE PACHMAN 15 *   | * STRUCT.ENG. *              | * EXTENDED BASIC *           | * WARGAME *                 |
| 1 MINE PACHMAN 16 *   | * TINY LOGO *                | * HOUSE. BUDGET *            | * WUMPUS *                  |
| 1 MINE PACHMAN 17 *   |                              | * PERS.RECORDKEEP *          |                             |
| 1 MINE PACHMAN 18 *   |                              | * PERS.REPORT GEN *          |                             |
| 1 MINE PACHMAN 19 *   |                              | * PHYSICAL FITNESS *         |                             |
| 1 MINE PACHMAN 20 *   |                              | * SPEECH EDITOR *            | <u>CLASSIC ADVENTURES</u> * |
| 1 MINE PACHMAN 21 *   | <u>TUNNELS OF DOOM.</u> *    | * TERMINAL EMUL.2 *          | * TI ADV. MODULE *          |
| 1 MINE PACHMAN 22 *   | * T.O.D.MODULE *             | * BEYOND W/WRITER *          | * ADVENTURELAND *           |
| 1 MINE PACHMAN 23 *   | * ASSAULT/CITY *             |                              | * BUCKAROO BANZAI *         |
| 1 MINE PACHMAN 24 *   | * THE DOCTOR *               |                              | * THE COUNT *               |
| 1 MINE PACHMAN 25 *   | * ORBS *                     | <u>GENERAL ITEMS</u> *       | * GHOST TOWN *              |
| 1 MINE PACHMAN 26 *   | * SPACE MINE *               | * CASSETTE CABLES *          | * GOLDEN VOYAGE *           |
| 1 MINE PACHMAN 27 *   | * TRUE KING *                | * ELECT.S'SHEET *            | * THE HULK *                |
| 1 MINE PACHMAN 28 *   | * DIAMOND QUEST *            | * GRAPHING PACK. *           | * KNIGHT IRONHEAR' *        |
| 1 MINE PACHMAN 29 *   |                              | * PERS.FINANCIAL *           | * MISSION IMPOSS. *         |
| 1 MINE PACHMAN 30 *   |                              | * PRINT INT'FACE *           | * MYSTERY FUNH'SE *         |
| 1 MINE PACHMAN 31 *   |                              | * ATARI ADAPTOR *            | * PYRAMID OF DOOM *         |
| 1 MINE PACHMAN 32 *   |                              | * ADAPTOR TO 2XTII *         | * SAVAGE ISLAND *           |
| 1 MINE PACHMAN 33 *   |                              | * 1XTII JOYSTICK *           | * SORC'R CLAYMORG *         |
| 1 MINE PACHMAN 34 *   |                              | * 2XTII & ADAPTER *          | * SPIDERMAN *               |
| 1 MINE PACHMAN 35 *   |                              | * DUST COVER *               | * STRANGE ODYSSEY *         |
| 1 MINE PACHMAN 36 *   |                              |                              | * VOODOO CASTLE *           |
| 1 MINE PACHMAN 37 *   |                              |                              | * WITCH'S BREW *            |
| 1 MINE PACHMAN 38 *   |                              |                              | * ADULT ADVENTURE *         |
| 1 MINE PACHMAN 39 *   |                              |                              | * HINT BOOK *               |