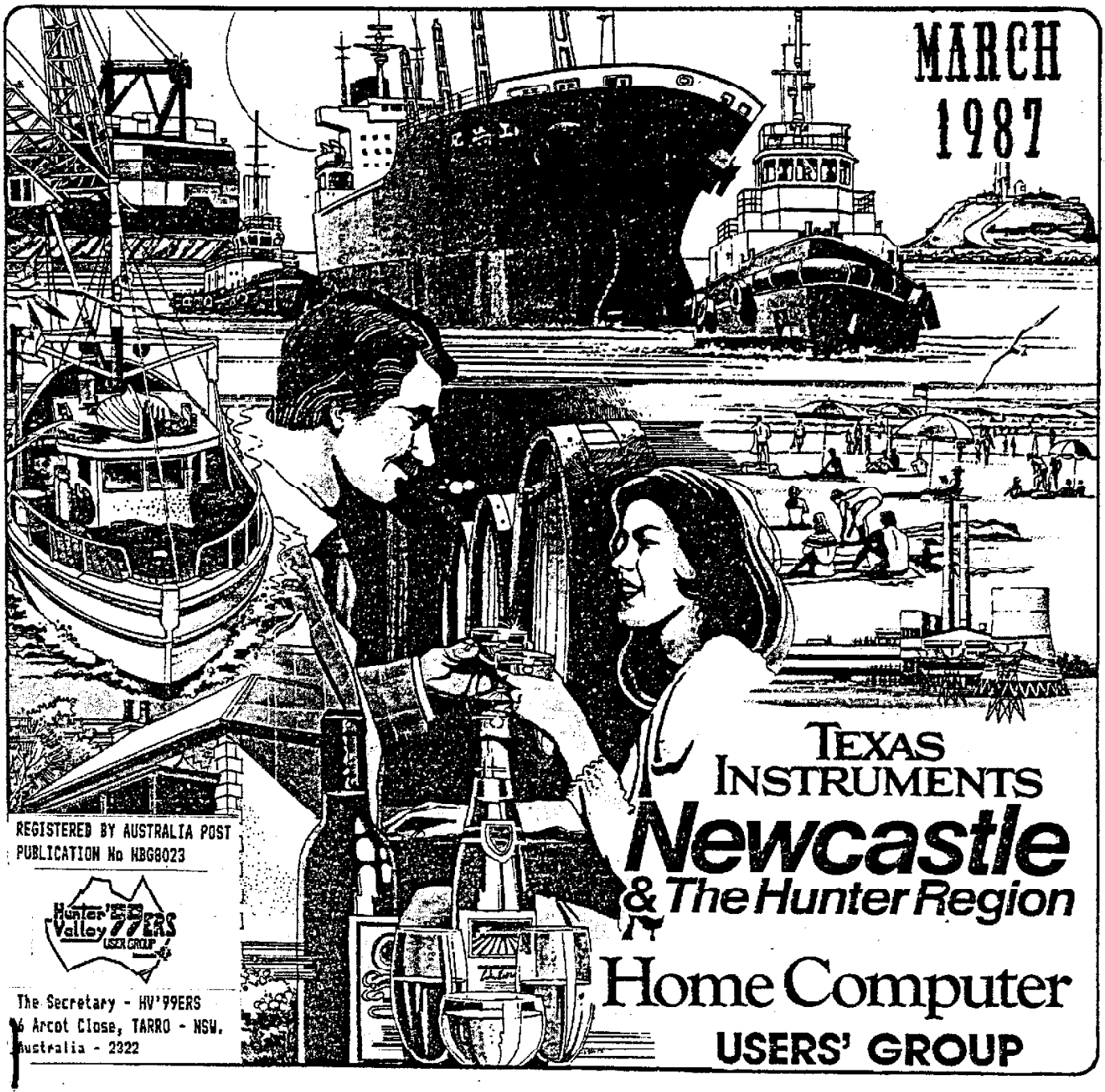
HUNTER VALLEY 99'ERS NEWS

TI 99/4A

HOME COMPUTER NEWSLETTER



REGISTERED BY AUSTRALIA POST
PUBLICATION No NBG8023

The Secretary - HV'99ERS
6 Arcot Close, TARRO - NSW.
Australia - 2322

YOUR COMMITTEE

PRESIDENT
Allen Wright
77 Andrew Rd.,
VALENTINE 2280
Ph. 468120

VICE PRESIDENT
Jim Grimmond
31 Jarrett St.,
TORONTO 2283
Ph. 595751

SECRETARY
Albert Anderson
6 Arcot Close,
TARRO 2322
Ph. 662602
Viatel 496626020

TREASURER
Brian Rutherford
9 Bombala St.,
DUDLEY 2290
Ph. 498184

SOFTWARE LIBRARIAN
Alan Lawrence
35 Bayview St.,
WARNERS BAY 2282
Ph. 486509

PUBLICATIONS LIBRARIAN
Paul Mulvaney
26 Marmong St.,
MARMONG POINT 2284
Ph. 583623

EDITOR
Brian Woods
9 Thirlmere Pde.,
TARRO 2322
Ph. 662307

COMMITTEE
Bob MacClure
75 Deborah St.,
KOTARA SOUTH 2288
Ph. 437431

Gary Jones
53 Janet St.,
JESMOND 2299
Ph. 573744

Tony McGovern
215 Grinsell St.,
KOTARA 2285
Ph. 523162

Peter Coxon
25 Reserve Rd.,
WANGI WANGI 2267
Ph. 751930

CONTRIBUTIONS

Members and non members are invited to contribute articles for publication in HV99 NEWS.
Any copy intended for publication may be typed, hand written, or submited on tape/disc media as files suitable for use with TI Writer (ie. DIS/FIX 80 or DIS/VAR 80). A suitable Public Domain word processor program will be supplied if required by the club librarian Al Lawrence.
Please include along with your article sufficient information to enable the file to be read by the EDITOR eg. File Name etc.
The preferred format is 35 columns and page length 66 lines, right justified.
All articles printed in HV99 NEWS (unless notified otherwise) are considered to be PUBLIC DOMAIN. Other user groups wishing to reproduce material from HV99 NEWS may feel free to do so as long as the source and author are recognised.
Articles for publication can be submitted to.

THE EDITOR
HV99 NEWS
9 THIRLEMERE PDE.,
TARRO 2322

General address for ALL other club related correspondence.

THE SECRETARY
HV99 USER GROUP
6 ARCOT CLOSE,
TARRO 2322

DISCLAIMER

The HV99 NEWS is the official newsletter of the HUNTER VALLEY NINETY NINE USER GROUP. Whilst every effort is made to ensure the correctness and accuracy of the information contained therin, be it of general, technical, or programming nature, no responsibility can be accepted by HV99 NEWS as a result of applying such information.
TEXAS INSTRUMENTS trademarks, names and logos are all copyright to TEXAS INSTRUMENTS.
HV99 is a non profit group of TI99/4A computer users, not affiliated in any way with TEXAS INSTRUMENTS.

SECRETARYS' REPORT

Welcome once again from us here at HV99 and to our new members, both locally and abroad, thank you for your support.

A bright spot for us here is the return of Vice Pres. Jim Grimmond from his rather concerning illness. We said that we were going to take it easy on him for a while but alas that unfortunately is not going to be the case. I hate following good news with bad, but in some ways this next item is one which is difficult for us to actually face up to however it goes as follows - one of the founders of HV99 (which goes back to early 1984), and our only President, Allen(Joe)Wright has formally resigned as President of the HV99 group. His reasoning for this move from office is simple and is as sound as his moral fibre; **to allow for change.** I say simple because in my own experience, I have learned that *if you don't allow for change, change won't allow for you* - ask Charles Darwin!

I would like to take this opportunity to both personally and on behalf of ALL HV99ers publicly thank Joe for his irreproachable guidance that undoubtedly has helped put HV99 in the respected position that it now holds in the worldwide TI99/4A User Group community.

The news isn't all that bad because Joe now will have more time to devote to the learning and teaching of the ASSEMBLER which I think has got to the stage where he dare not look back.

Now onto the good stuff!

I would like to congratulate the new office bearers of the TIsHUG group in Sydney and wish them well in what looks to be a promising and busy year. I would also like to thank the previous officers of TIsHUG for allowing our small unofficial party to sit in at the AGM as this enabled us to actually put faces to the voices we speak to over the phone.

Whilst I am on that subject(AGM), the HV99 AGM is in JUNE of this year and believe it or not that is only 3 months away, so if you are interested in helping in the running of the group please start to consider your nomination or nomination of others NOW. The paperwork will be sent to you in the next couple of issues of the HV99 newsletter.

As usual the newsletters from other groups continue to roll in and still they continue to get better and better. Thank you to you all and particularly our exchange groups. With this in mind a letter from Tasmanian member Steve Taylor points out that our non-local members are at a disadvantage when it comes to access to the publications library. This is unfortunately very true, and as a suggestion Steve proposes that a newsletter 'chain' be set up to service this need. Ideas and suggestions on this, from YOU the disadvantaged are needed so that the group can disperse this valuable information to as many members as possible.

On the overseas front, news on the IBM compatibility system has finally arrived from TRITON and the whole thing looks very smart indeed. An AD elsewhere in this issue will give an insight to the system. Many more add-ons, conversions and amazing peripherals are popping up every day from the US.,Canada,West Germany and yes, AUSTRALIA so if you ever get the urge to yuk! trade it in, I seriously put it to you to look very carefully as to what your new system will offer for your precious dollar.

With that in mind i'll leave it at that so you can get into the rest of the mag. Back again next month.

Albert Anderson
4a4me

RANDOM BYTES

By Bob Carmany

Since this is the first article that I have submitted to the HV99 UG, I thought that we had better get acquainted. Since I am based here in the USA, there are going to be some minor spelling differences in the articles (ie. program instead of programme, color instead of colour, etc.). They are not mistakes so don't go blaming Brian for them.

This column, "RANDOM BYTES" is going to be a potpourri of material for a wide range of TI Users. I hope to cover the spectrum from the novice BASIC programmer to the more advanced Forth or A/L programmer. With that "lofty" goal in mind, let's go ahead and get started!

Albert Anderson has told me that there is not much of an interest so far in the UG for the "Wycove" variety of Forth. Well, I am going to take this opportunity to try to change a few minds. Tim MacEachern of Wycove Systems has recently come out with a new version (3.0) of Wycove Forth. It has several advantages over the "beast" that TI created! First of all, the system is debugged (better than TI). It also comes complete with access screens already written for speech and sound utilities. The best part, though, is in screen loading time. The advertisements say that it will load screens 50% faster than Wycove Version 2.1 and 68% faster than TI Forth. How very true that is --- my "benchmark" load experience (loading a minimum of 4 screens) indicates that those claims are absolutely true. In fact, mine did just a bit better!!

Now, for the "bad" part! The release is a limited release -- without any documentation. If you have or have access to the documentation for either Version 2.0 or 2.1, you are all set. Everything is the same except that cassette access has been removed and there are a few minor changes (most are in the short doc file that comes with it). The system (2 disks) can be ordered from Tim MacEachern, P.O. Box 1105, Dartmouth, Nova Scotia, B2Y 4B8, CANADA. The price : $25.00 (US).

I recently wrote a little article for MICROpendium (March) about programming with a word processor. It discussed programming from "scratch" but didn't mention how to dump an existing program so that it could be edited or altered using something like FUNNELWEB. The easiest way to change a program from program format to a D/V 80 format is simply to load it into the computer and then, in the immediate (command) mode type in: LIST "DSKx.filename" where "x" is the drive number and "filename" is the name you want the D/V 80 file to have. Just remember to make it a different name than the original file if you are going to save it on the same disk drive.

Speaking of programming on a word processor, the biggest advantage is to be able to get rid of all of those "typos" without having to go through the program on a line-by-line basis. Use a template file of line numbers, write your program, and run it through DRAGONSLAYER Spellcheck. All you have to do is make up a user dictionary that contains all of the BASIC and XBASIC commands (ie. HCHAR, VCHAR, PRINT, etc.). The Spellcheck program will "ferret" out all of the mis-spelled words for you and allow you to make the corrections. Then, run XLATE on the D/V 80 file and it will create a RUNnable program.

Here's one that Tony McGovern reminded me of for those of you with FUNNELWEB (or FUNLWRITER). It is ironic that I had the information in front of me all the time from another source (MG GRAM KRACKER docs). If you are loading a series of program image files and find that one of them --- not the last one --- overwrites the loader and the whole program will not run, here is a quick fix:

First, find out which of the files is the culprit. That is easy,

the computer locks up while the "Loading DSKx.filename" is on the screen.

Since there are files to be loaded after that one, the first word of the file will be FFFF (more to follow). Using a disk sector editor, change it to 0000 (last file to be loaded). Then, using DM1000, change the file name to be the last in the load sequence.

Here is the whole process from the beginning. Suppose you had a series of program image files to load that were designated as: DUMP1, DUMP2, DUMP3, and DUMP4. You find that when FUNNELWEB starts loading the files sequentially, it loads the first two without any trouble. However, when it gets to DUMP3, it begins loading the file and then "locks up" and will not load the last file. You have identified DUMP3 as the culprit!

Go into the file with a sector editor and change the first word (it will be FFFF) to 0000. Go into DUMP4 and change the first word (it will be 0000) to FFFF.

Now, using DM1000, give DUMP3 a temporary name (ex. DUMP5). Rename DUMP4 by changing the name to DUMP3 and then rename DUMP5 (temporary) to DUMP4. The program image files should now all run without incident.

All you have really done is to swap DUMP3 and DUMP4 and change the first word of the file header to allow the offending file to be loaded last. The procedure is quite simple but also quite effective. Remember, though, do all of these changes to a COPY in case you make a mistake or the file just doesn't load like it should -- there is always an exception to every rule.

Here is one more "quick" tip before I get a BUFFER FULL message. Have you ever got the rather enigmatic DATA READ ERROR message when you tried to run a program? Have you tried to track down the source of the error message? Often times, even using TRACE can lead to nothing! If you know what the variable name is (ex. READ A$), you can easily locate the error in your data statement. Go ahead and RUN the program and when the error message appears, type in PRINT A$ in the command mode. The last correct data value (or string) will be printed on the screen. It is a simple matter to go into the program and find that value and look at the next one which is the one that gave you the error message in the first place.

Well, I'm rapidly running out of BUFFER space for this month, we will have some more BASIC tips and some Forth material (Wycove) in the next column. I will have a little program that will allow you to see what those character definitions actually look like -- you know, the ones that are contained in data statements.

Any questions or suggestions for future topics can be sent to: Bob Carmany, 1304 Larson St., Greensboro, North Carolina, USA, 27407.

TI-WRITER.. UNDER THE HOOD

by tony mcgovern

Have you ever wondered what was going on inside your TI-99/4a while you are sitting there with TI-Writer stoked up, your magnum opus up there on the screen, and you are pondering what key to press next. We'll have a look at it now from a programmer's point of view, without actually getting into the gory details of code. Instead of worrying about the sort of details that you need to know to write a loader like Funnelweb we'll take an inside out view starting right there with the blinking cursor on the screen. This view won't go into all the possible details because the Editor is a tightly written program in assembly language and it is quite complex. So the discussion will be generally correct but will not dive off into all details along the way, or make all the caveats for particular cases.

So you are going to press a key sometime. What is the Editor doing ? Waiting for you to press the key of course ! It does this by sitting there in a tight little loop repeatedly calling the console keyscan routine while waiting for it to indicate that a key has been pressed. There's more to it than that too and it's obvious to the eye - the cursor is flashing. What makes the cursor ? The Editor is in text mode so it can't be a sprite and has to be another character which the key loop periodically substitutes for the character under the cursor position. This means there has to be a counter to time the flash and maybe some extra delay to set the basic loop timing. What else is happening ? If you wait long enough to make up your mind what key to press the whole screen blanks out. No big deal here because it is built into the console as a system function. Not all home computers are so considerate of your TV. All that the key loop needs to do is enable interrupts briefly every time the key routine is executed. This is also the routine that senses the <fctn => Quit key but this has been disabled previously on the way in. The original TI-Writer Editor re-enables Quit when it goes back to GROM for SD, something I found out the hard way early in the piece.

Now you press a key. The key loop notices this and jumps out into the main key processing routine. Actually it's a little more complicated than this because it has to check the key against the last one pressed to see if it should do anything in the autorepeat line. We won't go into the details of this because writing key routines with autorepeat and flashing cursor is a devilishly tricky business. The Editor does not use a buffer to store keystrokes. This is both a blessing and a curse. The curse is that no keys can be detected while the processor is off executing the consequences of the last keystroke. The keyboard is then dead. Normally for text input this is over in a short enough time that it is not noticed. The blessing is that you don't suffer the drawbacks of simple minded key buffers such as multiple execution of commands if you hold the key down while waiting for some slow action to work.

First the key processor has to decide whether the key pressed is a control function of some sort as defined in the manual and if so take appropriate action. For the moment let's assume it is a plain old letter key that should put up a displayed symbol on the screen.

Hold on a moment there - where did the screen come from - how does the Editor know what to write up ? It maintains in CPU RAM in low memory a 24 line set of 80 column lines, and writes up the appropriate part of this to the VDP screen display memory according to which horizontal screen window is in force and whether line numbers are to be displayed. What is more, it does this in between every keystroke. Remember this isn't a

Basic program placing characters on the screen one at a time. It isn't even the GPL/Basic line editor. This is the raw speed of the machine at work. What it does show is just how much TI Basics slug down the speed of the 9900 with multiple layers of interpretation. This sets the basic timing cycle of the Editor. If it were running on a much faster processor, say a 9995, then it would be sitting there wasting an even bigger proportion of its time once it had done all its chores after each keystroke, even for a very fast typist. The quick screen updating by TI-Writer rather spoils your attitudes to the type of word processor that has to go out to disk for just about anything, and/or writes up the screen as a terminal.

While we are still in pursuit of the big picture, the next question to be answered is where do the CPU screen buffer contents come from. The mass of text file itself is kept in high memory along with some working buffer areas and the code for managing the buffer, about 1K in length, leaving about 23K for text buffer. How is the text stored ? In the text buffer it is a little like storage of Basic programs in memory. A line number table is stored with pointers to the actual line itself. Thus whenever a line is added the line itself is merely added to the end of the buffer and only the line number table need be updated. Deleting a line is a more serious business, as not only must the line number table be adjusted but the line itself has to be excised from the buffer and the rest of the text moved up in bulk to fill the gap. You may have noticed that deleting a range of lines is a slow process. This is because the lines are removed and the buffer adjusted one line at a time. This is, when you think about it not a high priority area for speeding up the code or writing faster code in the first place. Faster code always takes more room. Still a smarter Delete Line would have been nice.

A more important goal is to squeeze as much text as possible into the buffer. Unlike some word processors which keep the text file on disk, only retrieving various parts of it as necessary, TI-Writer is of the type that keeps the entire document in memory. The trade-off is between total size of document and speed of response in moving around the text. It is then important to squeeze in as large a document as possible, as the TI-99 has half of its 64K memory map assigned to various ROMs. This is done by storing the lines in the text buffer in run length encoded form. You have heard of that in connection with all those RLE bit-mapped pictures that have been around recently. Well, if you hadn't thought of it in connection with your TI-Writer then you are now like the character in the play who discovered he had been speaking prose all his life. Every time a line is placed in the text buffer it is encoded, and every time a line is fetched from the buffer, whether for the screen buffer or to be written to an external device such as disk or printer, it is expanded out into its normal display form. This extra overhead of code and processing time is the price paid for squeezing more text into the buffer. It does slow I/O to disk somewhat, and the sector interlace on disks which TI optimized for program file loading is not necessarily the fastest for TI-Writer file handling.

The difference from the picture example is that it is strings of characters that are encoded. Since lines are only 80 characters long a length count may be distinguished by setting its most significant bit, allowing a count of up to 127. The tagged length byte is followed by the character that is to be repeated. This all means that there is no easy direct relation between the size of a text file as it is stored on disk and the amount of text buffer it occupies. The only external indication of the state of the buffer by TI-Writer is when it is full so this is the only state you can really experiment with. So here's a little experiment. First take a freshly initialized or swept disk to the largest capacity your drives will handle. Fill a few lines with asterisks or any other non blank character. Then use the

Copy function to make as large a file of these lines as will fit. Then save the file to the empty disk.

Now back to the main story. Suppose instead of a printing character you had typed <fctn-9> or <ctrl-c> which is a lot easier on the hand. The text on the screen drops a couple of lines, and a command line appears and typed entries appear on the second line. That you know well. A couple of minor details are interesting here. Everything you type comes out in upper case. This is done by filtering the keystrokes through a little routine that changes lower case to upper case and not by fiddling with the key-unit. Also it doesn't matter where on the command line you start the entry because another little routine chugs along the line until it finds the first non-blank character. To this stage there haven't been great changes going on, but once a command has been decided on a major shuffle takes place. You will recall that I said the screen buffer of 80 column lines is in low memory. Isn 't that where the bulk of the Editor's code resides ? Yes it is, and when the Editor starts up it executes some initialization code and then lays down the screen buffer over this code and a whole lot more that is needed only in command mode. That doesn't sound all that good an idea, but you have to remember that there is still a copy of that code in the machine in VDP memory. Recall that the program file loading process as defined in the disk DSR first dumps the the program file into VDP and then it is up to the program that invoked the LOAD to do with it what it wants. The convention with E/A program files is that the first word is a tag, null to indicate the last file, the next the length of the file, and the third the starting address where it belongs in CPU RAM. This transfer is done nondestructively so EDITA1 is still there in VDP. So far so good but now EDITA2 is loaded and overwrites EDITA1 in VDP. No problem though because it is much shorter and doesn't overwrite the parts of EDITA1 needed later. So every time the command mode code is needed it is copied from its ghost image in VDP.

Having the command code overlayed on the screen buffer means that the visible screen image must remain frozen because the buffer is now full of code. Equally well the program cannot be allowed to write any text into the buffer area or the code would be corrupted. When the command function is over the program refreshes the CPU buffer and then the regular loop writes the appropriate part of this to the screen and life goes on as normal. So you can see that a whole lot of shuffling is going on behind the scenes.

I can recall sorting all this code overlay process out for the first time for the first pass at what is now Funnelweb by writing a program file loader for Minimem and then using our trusty old DIS/ASS program from Basic and printing out the disassembly in 4 columns with COLIST. I still use that same printout for figuring out patches to the Editor. It took a whole weekend and more of poring over it, marking up the routines and branch tables. One little confusion was that the VDP pointer in the disk file didn't seem to fit the overlay process, but was written to the correct value during program initialization. Just recently I found that was because the program was loaded with E/A during the development process and those were the values frozen in when it was saved off.

It has been noted that the DV/80 file loader is quite robust, for instance not being upset by being presented with records longer than 80 characters, merely truncating them to 80 chars. It does this by filling a buffer area in VDP with blanks and then having the file system write the record on top of that. This way a record of a blank line saved as a single blank character is expanded out to a full line of blanks in the input buffer, and then compacted again for the text buffer. This robustness has been compromized in Funnelweb. because at one stage the file buffers were moved up towards the pattern table during the efforts to fit in all of SD and its fully paged directory. It turned out not to be necessary but somehow

or other they never were moved back again. Issues dated Mar 87 and later will have the original buffer position restored. It is not that it was ever in any danger of crashing, just that screen patterns for control characters could be

corrupted by records of excess length.

At this stage I suspect that this is is probably enough of a dip into the deep waters of the Editor. If it gets you thinking about how your word processor is going about its business while you are actually at the keyboard, then this article will have served it's purpose, and hopefully won't have upset your typing rhythm too much.

--- Tony McGovern, Funnelweb Farm

TI WRITER TIP

This FUNNELWEB tip comes from Jane Laflamme, writing in the January 1987 issue if the Newsletter of the Ottawa TI99/4a Users Group.

Using the Formatter can use a lot of paper to check and recheck errors while learning how to use it. You can save on paper by 'printing' your file to disk rather than the printer. Be sure to use another filename so you don't overwrite your original file. Ms Corker, in the booklet, TI-Writer Tips and Tricks, recommends adding P to the original filename, eg If DSK1.MYFILE is the file created by the Editor, then printing to disk through the Formatter could be called DSK1.P_MYFILE. By loading the 'printed' file back into the Editor, you can see how the Formatter works. You can also delete any of those first three line feeds if you don't want them. If you then wish to print the file from the Editor rather than the Formatter, you will have to use 'PIO.LF' rather than 'PIO'.

*** PRESS RELEASE ***

ORGANIC DATA TRANSMISSION SYSTEM

CHAOS MANOR has reported the results of several years of intensive study into the feasability of transmission of digital data by organic media. Considerable success has been achieved and further study and experiments will be conducted to improve the system.

According to a senior official at CHAOS MANOR the system to date only handles data serially but other studies show that massive data transfers by means of parallel methods will enable unlimited amounts of digital data to be transferred instantly.

The official also stated that due to the ever increasing need for better and faster methods of data transmission the military and other government bodies will be approached for financial assistance because it would be of great benefit to those departments.

The system is extremely flexible in operation and utilizes an ultra high speed miniature white ink jet, using what is called "fast life terminating bio-degradable ink". This is coupled to the sending terminal by a very simple digital to analogue interface, the ink jet is trained over a trail of Formicidae and with each passing Formicidae the ink jet is fired in the correct sequence giving black and white Formicidae which represents the binary code. At the receiving end a simple photo cell reads the passing Formicidae and converts it back to digital data, since the ink is so fast in its disintegration the Formicidae turnaround is fast and can be used over again without any waste.

Genetic engineers have been called in to give the Formicidae the speed of light and this will perfect the system.

REPORTER Ron Kleinschafer HV99.

```
SCR #28
  0  ( MARCH-87/01)
  1  ------------------------------------------------------------
  2  -                                                          -
  3  -                                                          -
  4  -   STRUGGLING FORTH                                       -
  5  -                                                          -
  6  -   HV99ERS  MARCH  1987                                   -
  7  -                                                          -
  8  -   SIMPLE WINDOW ROUTINE                                  -
  9  -   DISCUSSION OF USE                                      -
 10  -   CORRECTION TO FEBRUARY                                 -
 11  -   CRUNCHING SOURCE CODE                                  -
 12  -   VECTORED EXECUTION                                     -
 13  -                                                          -
 14  -                                                          -
 15  ------------------------------------------------------------
```

I still havn't heard from anyone about the decompiler, so I'm living in hope.

When I walked into the club meeting last month, (I rarely find the time to go), I found some source code for a different version to the autorepeat editor in a magazine, though I hav'nt tried it yet: also Bruce mentioned to me that the 64 column editor was auto-repeat!!!!, with similar structure to the one I was trying to decipher, so perhaps he will do it for me?

Our Forth group for '87 is up and running, we had a long session last week, at the end of which I'm sure I had managed to confuse everyone present. Hopefully all those who promised to materialize and didn't will show up next time.

I noticed I'd left a word out of last months mini-program: CONTINUE, the definition of which is included his week as well.

SIMPLE WINDOWS.

Windows seem to be all the rage nowadays, so I thought I'd have a go at writing a routine to do them, forgetting for the time being about what one wants in them.

There are really only a couple of requirements:
1.Define the screen area you want to have a window on
2.Save the data underlying this area to a temporary buffer
3.Use your window as you see fit
4.Be able to re-write what was underneath when one exits the window.

WHAT USE ARE THEY

Well, I suppose you don't really need them, but they do jazz up the program a bit, and make it more user friendly.Anyone familiar with PC programs will see endless uses. They are good for things such as popping up listings when the user wants them, such as help screens, storing data whilst programming, or even running a concurrent program using an ISR routine which our little machine is easily capable of. On that I definitly won't elaborate right now.

THIS MONTHS SOURCE CODE.

At the risk of being tedious I've included several almost identical versions of the same code. I will later discuss these in terms of legibility, compactness, and the amount of memory they occupy.

We have covered Frame/box drawing before, so include these screens for completeness. First run the program and see what it does:

TYPE 62 LOAD, then when it has compiled type TEST.

Sure this seems simple. The screen will fill with, for the want of something better, A's, a window will appear with a message. Pressing the space bar will re-write the window and return you to Forth.

Lets examine the key words:

BLANK-WINDOW:

Simply uses HCHAR to sequentially blank each row of your designated box area. Note the stack order is different from the others, no particular reason, I just wrote it that way first and could'nt be bothered changing it. Like the other words it is general, in the sense it can be used anywhere on the screen, BUT: there is no validation of what you put on the stack for it, as this is totally unnessessary, ie you must ensure the parameters are within range for the screen size. If you dump wrong data for it on the stack and your program crashes you deserve the inconvenient results. NOTE: if you want to re-word an application using all three words with a single input of co-ordinates eg:

10 20 1 1 WINDOW for instance, you will have to modify this to expect the same order as with the others, and have each word leaving copies of the indices on the stack for the next. I didn't do this because Its not really necessary.

This routine is a good general routine for many programs where you want to clear part of the screen, therefore perhaps we should rename it PARTCLS or PCLS.

SAVE-WINDOW WRITE-WINDOW.

These were an interesting evolution of some pretty horrific code involving 2 nested DO...LOOPs and the VSBW routine, which worked but was messy. When my scribbles finally ended up in code, it was obvious that there is only a single difference between the two definitions, a SWAP on line 11 of SCR# 32, to place in correct order the from/to addresses.

These two definitions expect the height of your window, its width, and the starting column and row. They simply read from/write to the screen one row at a time and either take the data and deposit it in an area you have designated, in our case SAVED-BUFFER within the dictionary, or take data from there, and back to the screen

Again this type of loop illustrates a repetitive thing in Forth, using the loop index to increment an address you either want to store data to or take data from. Here, SAVED-DATA is incremented by the Width of the window multiplied by the loop index, ensuring each new line of the window read is deposited another Width bytes along the data array, so as to not write over the last lot. Similarly where to either read from or deposit to on the screen, is calculated as an offset from the top left hand corner or 1VDP-ADR.

We can in fact combine these two definitions, or rather make a primitive or precursor definition which can be used in both, which brings us to vectored execution.

VECTORED EXECUTION.

Before we get to this we'll have to take a brief look at what happens when Forth executes a word. Power up your machine and interact! Let's define a word:

```
: HOUSE ." dwelling we live in" ;
```

now type HOUSE, to which Forth replies:

```
dwelling we live in ok
```

When HOUSE is placed in the dictionary, Forth gives in an address, just as you have a house number, so it can find it again. You may not normally need or want to know where it is. Lets look anyway.

The Forth '(tick) finds a words address within the dictionary as we saw last month. Type:

' HOUSE . and Forth replies with a number eg -6789. This is equivalent to the number of your house in your street. Actually its the start of several addresses pertaining to HOUSE, ' leaving what's called the Parameter Field address. Forget about why, in a later article (promises promises) I'll do the dictionary structure.
The code to run or execute this word is not actually at this exact address. Typing ' CFA converts parameter->code field address.

Now type:

' HOUSE CFA EXECUTE <enter>

to which once again you will get:

dwelling we live in

So what use is it? On the surface it seems like a long hand way of getting Forth to do something. There are occasions when It's useful, which brings me to vectored execution.

We can execute the definition indirectly by finding out its address, storing it in a variable, and fetching it from the variable to execute it later on in the program when its needed.

Refer now to SCR# 57,58,59. I previously mentioned how similar the two definitions of SAVE-WINDOW and WRITE-WINDOW were. These screens remove the words which are different in these two definitions and re-define them on screen 57, replacing them with the common definition RD/WRT. This fetches the address stored in the variable R-W which will either point to the words to save or rewrite the data.

IE in vectored execution a single word, in this case RD/WRT can be made to perform different things at different times. My example here is not a particularly useful one, it actually takes up more bytes. On it other occasions it may make the code more useable and readable.

CONDENSING SOURCE CODE.

Finally for those of you who don't want to understand what your doing, or, if you do and you want it all on one screen I include scr# 62. The variable SAVED-DATA I have used throughout can be tailored to the size of window you want to save. If dictionary space is a problem, you can dump the data into either the buffer area (see previous article) or the VDP chip, a topic I'll start covering next month.

Next look at scr#63. This is a further condensed version, with reduced and illegible names, which on the surface of it should save more memory.

SIZE COMPARISIONS.

Excluding the routine for drawing the box, which would be universally used in your program for other things and takes up 230 bytes, and the area taken by SAVED-DATA which need not take up dictionary space, it is interesting to compare the space taken by the three versions. This also excludes the words FILL-4,MSG1,TEST and CONTINUE,or their equivalents as they only demonstrate the routine, but are not part of it.

1.SCR#31-34(longest) -216bytes
2.SCR#62-short/vectored -290bytes
3.SCR#63-short/ illegible -238bytes

I've got to admit I was quite surprised as well so I re-did the calculations and got the same answers. I guess the few additional definitions in the one with vectored execution make the difference.

Anyway, bye for now. Next month we will take a journey into the VDP chip, FOR SURE, as I've already written the article (if I can find what disk I left it on.....)

ADDRESS FOR CORRESPONDENCE :

RICHARD TERRY
141 DUDLEY RD
WHITEBRIDGE NSW 2290
AUSTRALIA
(049) WK 436861 H 22450.

VALUABLE MATERIAL LOST!

Richard Terry lent his Forth book "Forth Fundamentals" by C. Kevin McCabe to someone some time ago and to date has not been returned! If you happen to have this book could you please return it to Richard as soon as possible (he needs it to keep one step ahead of the SIGS group).

```
SCR #29
  0 ( WINDOW - Frame Characters      10Feb87)
  1 HEX
  2
  3 00FF 0000 0000 0000 83 CHAR ( 131 SINGLE UNDERLINE             )
  4 8080 8080 8080 8080 85 CHAR ( 133 SINGLE LINE RIGHT VERTICAL   )
  5 0404 0404 0404 0404 86 CHAR ( 134 SINGLE LINE LEFT  VERTICAL   )
  6 0000 0000 0000 00FF 87 CHAR ( 135 SINGLE LINE BASELINE         )
  7 8080 8080 8080 80FF 88 CHAR ( 136 SINGLE LEFT BOTTOM  CORNER   )
  8 0404 0404 0404 04FF 89 CHAR ( 137 SINGLE RIGHT BOTTOM CORNER   )
  9 FF00 0000 0000 0000 80 CHAR ( 128 SINGLE LINE TOP              )
 10 FF80 8080 8080 8080 81 CHAR ( 129 SINGLE TOP LEFT CORNER       )
 11 FF04 0404 0404 0404 82 CHAR ( 130 SINGLE TOP RIGHT CORNER      )
 12
 13 DECIMAL
 14
 15

SCR #30
  0 ( WINDOWS  - Single frame box   11Feb87)      ( CHECKED CLEAN)
  1
  2 : P    2 * SP@ + @ ;  ( put copy of nth number to top of stack)
  3
  4 : SBOX ( Leftcol/up row/Rghtcol/lower row single box routine  )
  5        4 P  4 P                 1 129 HCHAR   ( upper left  cnr )
  6        4 P  OVER                1 136 HCHAR   ( lower left  cnr )
  7        OVER 4 P                 1 130 HCHAR   ( upper right cnr )
  8        OVER OVER                1 137 HCHAR   ( lower right cnr )
  9        4 P 1+ OVER 4 P 1- 7 P - 135 HCHAR   ( lower horizontal)
 10        4 P  4 P 1+ 3 P 1- 6 P - 133 VCHAR   ( right vertical  )
 11        OVER 4 P 1+ 3 P 1- 6 P - 134 VCHAR   ( left  vertical  )
 12        4 P 1+ 4 P  4 P 1- 7 P - 128 HCHAR   ( upper horizontal)
 13                                  3DROP DROP ;
 14
 15

SCR #31
  0 ( WINDOWS - blanking window     11Feb87)
  1
  2 0 VARIABLE SAVED-DATA 1000 ALLOT
  3
  4 : CONTINUE      BEGIN ?KEY 32 = UNTIL ;
  5 : 1VDP-ADR                      ( expect col,row--start vdpad)
  6                 40 * + ;        ( leaves start vdp of window )
  7
  8 : BLANK-WINDOW                  ( Expect col,row,width,height)
  9                 0 DO            ( blank out height lines loop)
 10                   3DUP           ( copies of col,row cnt-width)
 11                   SWAP I + SWAP  ( increment each row          )
 12                   32 HCHAR       ( use col/row+I,width->blanks)
 13                  LOOP
 14                   DROP DROP DROP ; ( leaves nothing on stack    )
 15
```

```
SCR #32
  0 ( WINDOWS -Save window data      11Feb87)
  1
  2 : SAVE-WINDOW                          ( exp height/width/col/row  )
  3                 1VDP-ADR               ( upper left start vdpaddr  )
  4                 ROT                    ( height to top for loop    )
  5                 0 DO                   ( read box by row hght times)
  6                    2DUP                ( dup width/1vdppos         )
  7                    OVER I *            ( offset from start save adr)
  8                    SAVED-DATA +        ( start addr to deposit to  )
  9                    SWAP                ( 1vdp address to top stack )
 10                    I 40 * +            ( vdp adr of start each row )
 11                    SWAP ROT            ( now vadr/adr/cnt for VMBR )
 12                    VMBR                ( write line to saved-data  )
 13                  LOOP
 14                    2DROP               ( drop copy of 1vdp/width   )
 15                    ;                   ( leaves nothing on stack   )

SCR #33
  0 ( WINDOWS -Rewrite window        11Feb87)
  1
  2 : WRITE-WINDOW                         ( exp height/width/col/row  )
  3                 1VDP-ADR               ( upper left start vdpaddr  )
  4                 ROT                    ( height to top for loop    )
  5                 0 DO                   ( read box by row hght times)
  6                    2DUP                ( dup width/1vdppos         )
  7                    OVER I *            ( offset from start save adr)
  8                    SAVED-DATA +        ( start addr to read from   )
  9                    SWAP                ( 1vdp address to top stack )
 10                    I 40 * +            ( vdp adr of start each row )
 11                    ROT                 ( now adr/vadr/cnt for VMBW )
 12                    VMBW                ( write line of saved data  )
 13                  LOOP
 14                    2DROP               ( drop copy of 1vdp/width   )
 15                    ;                   ( leaves nothing on stack   )


SCR #34
  0 ( WINDOWS - sample program)
  1
  2 : FILL-4  PAGE 959 0 DO 32 EMIT LOOP ;
  3
  4 : MSG1    11 11 AT ." MESSAGE/OR"
  5           11 13 AT ."  PROGRAM " ;
  6
  7 : TEST    FILL-4
  8           15 20  5  5  SAVE-WINDOW
  9            5  5 20 15  BLANK-WINDOW
 10            5  5 24 19  SBOX
 11           MSG1
 12           CONTINUE
 13           15 20   5 5  WRITE-WINDOW
 14           CONTINUE
 15           ;
```

```
SCR #35
  0 ( WINDOWS - Load screen          10Feb87)
  1
  2  29 LOAD 30 LOAD 31 LOAD
  3  32 LOAD 33 LOAD 34 LOAD
  4
  5
  6
  7
  8
  9
 10
 11
 12
 13
 14
 15

SCR #57
  0 ( WINDOWS -VECTORED)
  1 0 VARIABLE R-W
  2
  3 : (SAVE)       SWAP ROT VMBR  ;
  4
  5 : (WRITE)      ROT  VMBW      ;
  6
  7 : RD/WRT       R-W @ CFA EXECUTE ;
  8
  9
 10
 11
 12
 13
 14
 15

SCR #58
  0 ( WINDOWS -Save window data     11Feb87)
  1
  2 : (WINDOW)                          ( exp height/width/col/row  )
  3               1VDP-ADR              ( upper left start vdpaddr  )
  4               ROT                   ( height to top for loop    )
  5               0 DO                  ( read box by row hght times)
  6                  2DUP               ( dup width/1vdppos         )
  7                  OVER I *           ( offset from start save adr)
  8                  SAVED-DATA +       ( start addr to deposit to  )
  9                  SWAP               ( 1vdp address to top stack )
 10                  I 40 * +           ( vdp adr of start each row )
 11                  RD/WRT             ( arrge stack for read/write)
 12                LOOP
 13                  2DROP              ( drop copy of 1vdp/width   )
 14                  ;                  ( leaves nothing on stack   )
 15
```

```
SCR #59
  0
  1
  2 : SAVE-WINDOW                          ( exp height/width/col/row  )
  3                ' (SAVE)  R-W !         ( adr of save changes ->R+W )
  4                (WINDOW)                ( draw window               )
  5                ;                       ( stack empty               )
  6
  7
  8 : WRITE-WINDOW                         ( exp height/width/col/row  )
  9                ' (WRITE) R-W !         ( adr of save changes ->R+W )
 10                (WINDOW)                ( draw window               )
 11                ;                       ( stack empty               )
 12
 13
 14
 15

SCR #62
  0 ( WINDOWS - entire code          11Feb87)        BASE->R DECIMAL
  1 0 VARIABLE SAVED-DATA 1000 ALLOT   0 VARIABLE R-W
  2 : 1VDP-ADR   40 * + ;  : CONT  BEGIN ?KEY 32 = UNTIL ;
  3 : BL-WINDOW  0 DO 3DUP SWAP I + SWAP
  4              32 HCHAR LOOP DROP DROP DROP ;
  5 : (SAVE)     SWAP ROT VMBR  ;  : (WRITE) ROT VMBW ;
  6 : RD/WRT     R-W @ CFA EXECUTE ;
  7 : (WINDOW)   1VDP-ADR ROT 0 DO 2DUP OVER I * SAVED-DATA +
  8              SWAP I 40 * + RD/WRT LOOP 2DROP ;
  9 : SV-WINDOW  ' (SAVE)  R-W ! (WINDOW) ;
 10 : WR-WINDOW  ' (WRITE) R-W ! (WINDOW) ;
 11 : FILL-4     PAGE 959 0 DO 52 EMIT LOOP ;
 12 : MSG1       11 11 AT ." MESSAGE/OR" 11 13 AT ."  PROGRAM" ;
 13 : TEST       FILL-4 15 20 5 5 SV-WINDOW 5 5 20 15 BL-WINDOW
 14              5 5 24 19 SBOX MSG1 CONT 15 20 5 5 WR-WINDOW CONT ;
 15                                              R->BASE

SCR #63
  0 ( WINDOWS - illedgible code eg  11Feb87)        BASE->R DECIMAL
  1 0 VARIABLE SD 1000 ALLOT   0 VARIABLE R-W
  2 : 1A          40 * + ;  : CONT  BEGIN ?KEY 32 = UNTIL ;
  3 : BW          0 DO 3DUP SWAP I + SWAP
  4               32 HCHAR LOOP DROP DROP DROP ;
  5 : (S)         SWAP ROT VMBR  ;  : (W) ROT VMBW ;
  6 : RW          R-W @ CFA EXECUTE ;
  7 : (WD)        1A ROT 0 DO 2DUP OVER I * SD +
  8               SWAP I 40 * + RW LOOP 2DROP ;
  9 : SW          ' (S) R-W ! (WD) ;
 10 : WR          ' (W) R-W ! (WD) ;
 11 : F4          PAGE 959 0 DO 52 EMIT LOOP ;
 12 : MSG1        11 11 AT ." MESSAGE/OR" 11 13 AT ."  PROGRAM" ;
 13 : TEST        F4 15 20 5 5 SW 5 5 20 15 BW
 14               5 5 24 19 SBOX MSG1 CONT 15 20 5 5 WR CONT ;
 15                                              R->BASE
```

FORTH LEARNERS GROUP

Report by Brian Rutherford

Richard Terry's Forth group met for the first time this year on Thursday 19th. February, and although only four (four for forth) turned up, there were a few others who wanted to come but could not make it, plus some confusion about which Thursday it was supposed to be.

Still, much was learned by the few in attendance, in fact the meeting went till after 11pm. First we newcomers were taught how to use the EDITOR and how to save lines to the PAD and place them on other screens where needed and how to pick up line and move them else where on the screen using FCTN 3 and FCTN 8. The functions of the stack and the dictionary were explained by Richard plus the elements of inputing from the keyboard when prompted for by a programme. All this as I said before took us through to 11pm, when we all staggered of home to bed.

Our second meeting was held last Thusday night, 5th March, and again the 6 people in attendance were treated to another informative tutorial by Richard (with some help from Joe!). The operation of the Stack, especially in regard to numerical operations was explained in some detail, until finally we all saw the light (I think!). These sessions with Richard are certainly the way to learn FORTH, so if YOU are interested, the FORTH SIGS group intend meeting every Thursday evening at 6.30pm. If you are attending one night, please ring Richard (436861) during the Wednesday to make sure that the meeting will be held.

BASIC GROUP

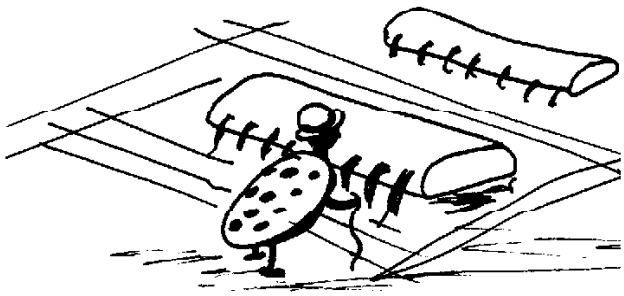
Any people wishing to form a BASIC group to learn programming from the start are invited to contact Paul Mulvaney (phone 583623), as he has volunteered to act as co-ordinator for the group. Instruction will be held on Tuesday nights at Warners Bay High two or three nights a month (dependant on Committee and General Meetings). No previous knowledge is necessary, as the instruction will start at the beginning and progress to a level that enables you to join Gary's Extended BASIC group.

The first get together is planned for the Tuesday after the APRIL General Meeting, ie Tuesday, 21st April.

ELECTRIC CIRCUITS II

Due to unforeseen technical problems (Paul did not get all the bugs out of the program in time), Part II does not appear this month. The weather in the Hunter area over the last month has been more conductive to swimming, sailing and drinking than to programming. See you next month!

Paul Mulvaney



I'VE BEEN IN HERE FOR AGES

ODYSSEY OF A TI TRAVELLER

In recent years I have travelled to several countries for my holidays. During which I have endeavoured to ascertain the popularity of the TI computer during these visits. Below is an short account of my recent trip to the U.S.A. There I found the TI alive and well. As you can gather from below, fellow users are a very friendly bunch of people who go out of their way to make one feel at home. If anyone is contemplating make a trip there, I would be happy to help in the way of contacts, &c. Please feel free to contact me at the following address: 8 Teesdale Crescent, Plympton Park 5038, South Australia.

I left on the 26th November, 9.30 a.m. from Adelaide to Melbourne. Four hour wait. To Sydney by United Air. One hour wait then to U.S.A., arriving Seattle late on 26th November (gaining a day).
Thursday 27th was a holiday (Thanksgiving). I was invited by Barbara Weiderhold to see a video of the Seattle Fair which had been held on September 26-27, 1986. Barbara was Chairperson and sponsor of the fair.
Barbara, with whom I had Thanksgiving, has a shop called the Queen Anne Computer Shoppe dealing wholly with TI.
Friday met Chuck Wynne and his son, Tom. Spent some time with them at their home.
Saturday morning back to Chuck's, who with Tom, drove me in the afternoon to Bits & Chips for a visit and then on to Barbara's shop An Aladdin's Cave for TI users. There met many fellow users.
Sunday was spent around Seattle with other members and ended the day by being invited out to dinner.
Monday arrived in Portland in the afternoon. That night went with Ron to his home. Ron runs a B.B.S.
Tuesday was free to sightsee and that night went to the A.G.M. of the Portland group. I spoke of the TI activities in Australia.
Wednesday a member, who had business in Eugine, offered to drive me there. At the hotel I made contact with Laurel and Melvin, local group members, and had dinner with them.
Thursday travelled all day by bus to San Francisco, arriving early morning. Phoned the president, Charles Peterson, who I had met last visit in 1984. Early afternoon went to Dr. Guy Romano, the operator of Amnion Helpline, and that night met Charles and the club librarian, Jim. All day Saturday was spent with club members.
Sunday went sightseeing around San Francisco.
Monday more sightseeing. This time, a trip to Carmel, the place were Clint Eastwood is Mayor.
Tuesday a local bus to Sacramento (the capital of California) arriving late afternoon. I then phone Wovely, a contact I was given in San Francisco, had dinner and back to the hotel. Wovely runs a B.B.S. He introduced me to one of his friends, Jim.
Wednesday was more sightseeing and in the afternoon again met Jim, who also runs a B.B.S., who invited me to stay for dinner.
Thursday I left for a day trip to Los Angeles by Greyhound, arriving early evening. Phoned Terrie Masters, the President, and went with Terrie to see Tom Freeman (a committee member).
Friday sightseeing around Los Angeles, and then to the suburb of Anaheim for overnight stay. On Saturday a local member, Rodger, took me sightseeing and to his place for dinner.
Sunday, early morning, a trip to a Computer Trash and Treasure where all types of items are sold. Then at 10 a.m. to Disneyland. This being the first Disneyland built (the one in Florida which I saw in 1984 is the largest, followed by Tokyo, which I visited in 1985).
Monday went by bus to Hollywood, a short trip away. The afternoon spent walking around Hollywood. Saw it again at night. Has a very interesting night life.
Tuesday spent with Los Angeles group again till late. That night went to dinner with Ken.
Wednesday left Los Angeles and flew to Hawaii and made contact with Phil (President of the local group).
Thursday was a day of relaxation on the beach.
Friday, sightseeing trip around the island.
Saturday I was taken by Phil to Paul's home, another member, and spent the day with them.
Sunday caught *"The Bus"* to Paul's home and met his son, Shaun. Spent all day with them and arrived back at the hotel at 1 a.m.
Monday, spent with Paul until early evening, then to the airport. Left Hawaii Airport at 1 a.m. on Tuesday, arriving back in Sydney on the morning of 24th December (lost a day), then on to Melbourne and Adelaide, arriving at 5.45 p.m.
The user groups in the U.S. are very friendly and go out of their way to help each other and visitors. There are many B.B.S. Boards in America.
I left Australia with 6DS/DD disks and returned with 190 disks of various types. All types of programmes from XB/E.A/TI Writer/ /Pascal/C/ Forth/G.P.L./Gram Kracker/Max-RLE.
Yes, I did see a lot of the West Coast of the U.S.A. and I made many good friends.
The TI is not dead — see following article.

GEOFF SHIPTON

Addendum — In July/August of this year I am contemplating a trip around the world. If anyone can help me with further contacts it would be most appreciated. Please reply to address given earlier in article.

Reprinted from The Seattle Times published Tuesday, May 6, 1986

It's a model of loyalty

by Richard Buck

Times business reporter

Last fall, I thought I owned a dead home computer, a machine that once seemed full of promise, then doomed to collect dust in my basement.
But the newspaper story I wrote then about the Texas Instruments model 99/4A home computer turned into a premature obituary, to my initial embarrassment and my later delight.
For all I know, there could be hundreds of other TI home computers collecting dust in closets, basements and bottom drawers, mostly because their owners don't realise that parts, software and advice about the machine are available in abundance in Seattle.

After all, it has been years since the last model 99/4A came off the TI assembly line.

In the early 1980s, the TI home computer was one of the biggest-selling consumer electronics products. More than 3 million were sold from 1979 (the initial price was $1200) to 1983, when TI dropped the price to $50 and stopped making them.

And the model 99/4A, not to be confused with the high-power professional models that Texas Instruments still makes, has inspired an intense, passionate loyalty among many owners.

Much of that passion is centered in Seattle, as I learned last fall after I wrote a personal account of how the computer my son and I bought had become useless to us.

I was amazed when, almost immediately after the story was published, my telephone started ringing and my yellow "in" box filled up with mail from owners of the TI computer.

On top of Queen Anne Hill, the Queen Anne Computer Shoppe is devoted exclusively to supporting that very computer. It even has a limited quantity of brand new ones in stock, priced at $85, complete with a one-year warranty. Used consoles sell for $50.

The store is owned by Barbara Wiederhold, a woman with an international network of contacts through which she can get just about any hardware, software or information relating to the Model 99/4A.

Roy Scully/Seattle Times

Barbara Wiederhold runs a Queen Anne shop that deals exclusively with the Texas Instruments 99/4A computer.

Wiederhold, who says she has about 3000 customers, spent several hours in person and over the phone helping my son and me revive our TI 99/4A. She and Phil Jordan, her technical expert and consultant, lavish time and attention on other TI owners and don't hesitate to refer them elsewherefor products if their shop can't offer the best deals.

Wiederhold says she can set someone up with a complete TI computer system, including color monitor, disk drives, printer and lots of software for under $1000. For $150 you could get a modest starter system, including at least a half-dozen software programs, to hook up with any television set.

Although Texas Instruments stopping making the 99/4A in 1983, the company supported it through a network of service centres, including one in Redmond, until about a month ago, when the centres were closed. The company still has a nationwide toll-free telephone number (800-TI-CARES) to help owners get parts, service or accessories and to answer technical questions about the machine.

If TI wanted a continuing fan club, it wouldn't have to look beyond Wiederhold.

"I got into this as a hobby and I saw a need," she said. "I didn't open this shop to sell people things. I opened it to get things for people if they needed them. We use this shop as a recycling centre, a learning centre and a training centre."

Her tiny, cluttered, second-floor shop is filled with books, magazines, newsletters, software, hardware and accessories for the TI computer.

"I get calls from all over the United States because I'm on bulletin boards," says Wiederhold.

Those electronic bulletin boards are run by users' groups that share information, tips, software and advice. Chuck Wynne, president of the Puget Sound 99ers group, claims 150 paying members. Four other users' groups in this state are focused on the TI machine.

Wynne frequents Wiederhold's store and another one, Bits & Chips in Edmonds, which carries a large collection of TI software. The Edmonds store, run by Janie Lawrence, also sells Sanyo computers, which are compatible with IBMs.

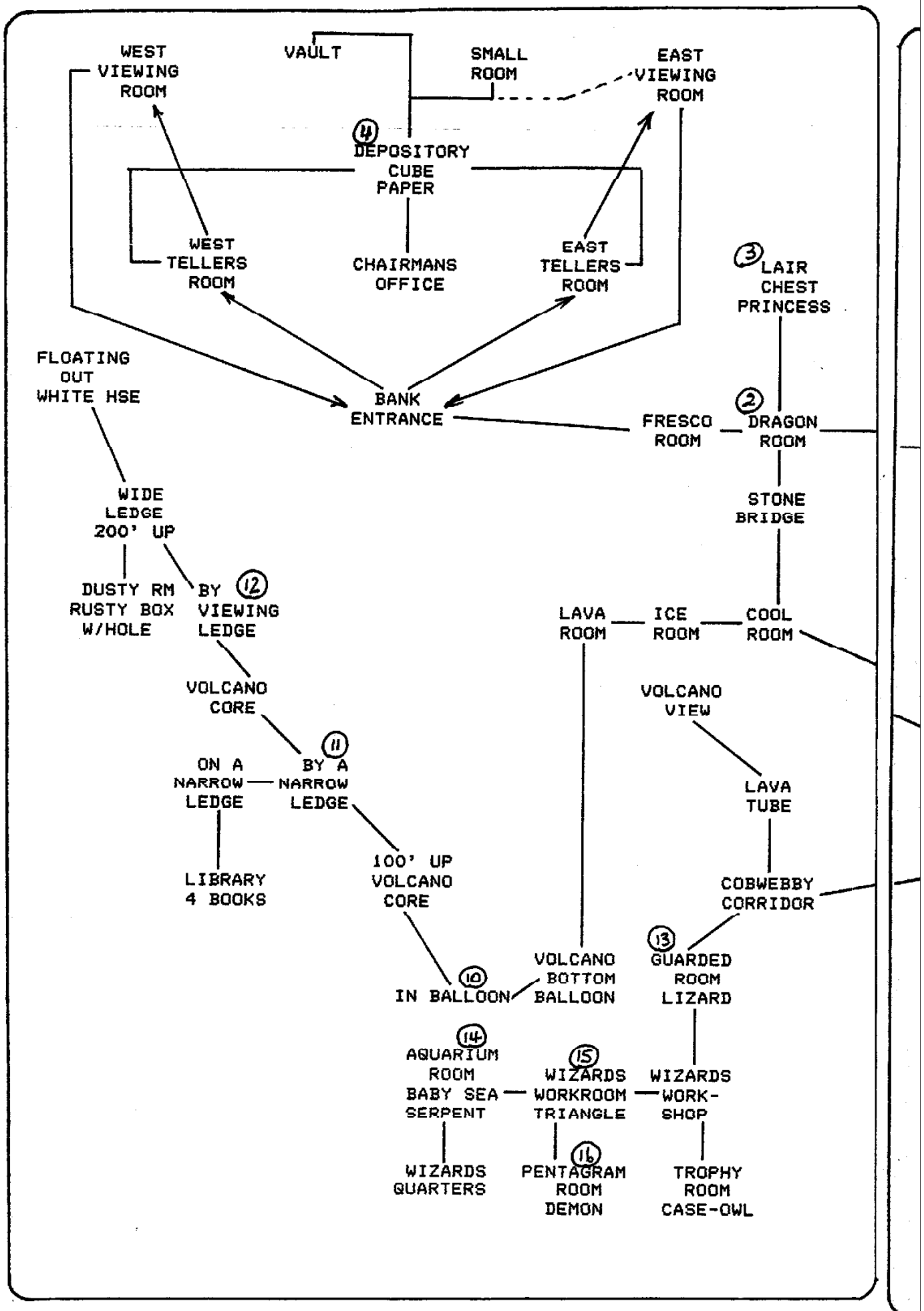
Wynne says he owns an IBM-compatible computer, "but it won't hold a candle to the TI." Wynne says his tiny TI can be programmed in any computer language and will work with any modem and any disk drive that's compatible with IBM models. "You can do virtually anything with that machine," he says.

That may be stretching a point. The TI's internal memory is 16 kilobytes, usually written 16K, the equivalent of about eight typewritten pages. By today's standards, that is next to nothing. The smallest computer IBM makes has 256K (about 125 typewritten pages), and even that is too limited to run some of the most popular business software programs.

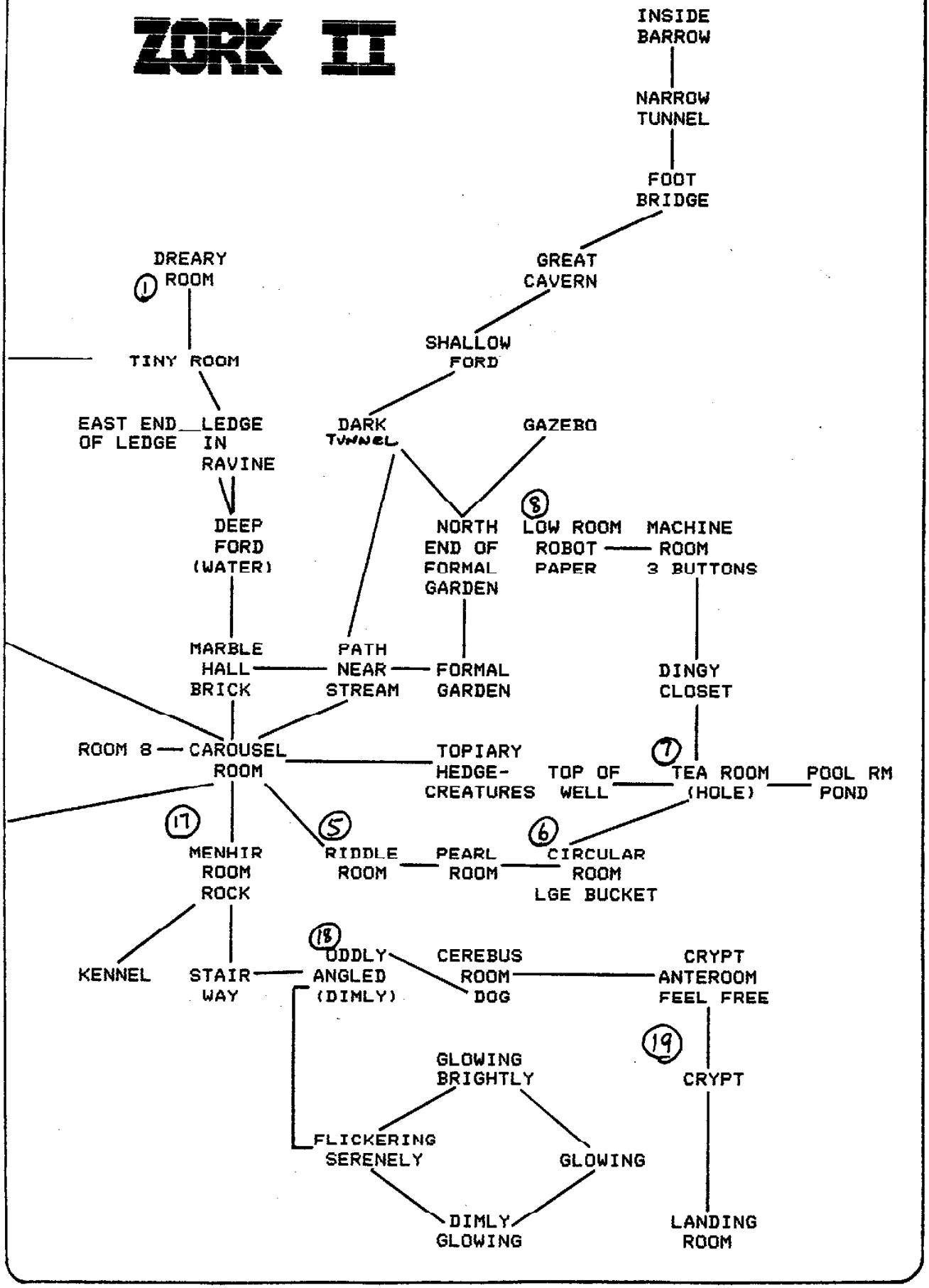
Wiederhold and Jordan are working with the State of Washington TI 99/4A Users Group to organise a convention of TI home-computer owners and vendors next September at a hotel near Sea-Tac Airport.

"We are inviting thousands of people from all over the United States and Europe, and we are scared to death that we will have more people than we can accommodate," said Wiederhold.

That remains to be seen. But at my house, this little computer is no longer collecting dust. Once again it's a busy machine, doing math, teaching touch-typing and programming, playing games and churning out nice-looking home-work papers, personal letters and memos.


WEST VIEWING ROOM
VAULT
SMALL ROOM
EAST VIEWING ROOM
4
DEPOSITORY CUBE PAPER
WEST TELLERS ROOM
CHAIRMANS OFFICE
EAST TELLERS ROOM
3
LAIR CHEST PRINCESS
FLOATING OUT WHITE HSE
BANK ENTRANCE
FRESCO ROOM
2
DRAGON ROOM
WIDE LEDGE 200' UP
STONE BRIDGE
DUSTY RM RUSTY BOX W/HOLE
BY 12 VIEWING LEDGE
LAVA ROOM
ICE ROOM
COOL ROOM
VOLCANO CORE
VOLCANO VIEW
ON A NARROW LEDGE
11
BY A NARROW LEDGE
LAVA TUBE
LIBRARY 4 BOOKS
100' UP VOLCANO CORE
COBWEBBY CORRIDOR
13
GUARDED ROOM LIZARD
VOLCANO BOTTOM BALLOON
10
IN BALLOON
14
AQUARIUM ROOM BABY SEA SERPENT
15
WIZARDS WORKROOM TRIANGLE
WIZARDS WORK-SHOP
WIZARDS QUARTERS
16
PENTAGRAM ROOM DEMON
TROPHY ROOM CASE-OWL

ZORK II


INSIDE BARROW
NARROW TUNNEL
FOOT BRIDGE
GREAT CAVERN
SHALLOW FORD
DREARY ROOM
1
TINY ROOM
EAST END OF LEDGE
LEDGE IN RAVINE
DARK TUNNEL
GAZEBO
DEEP FORD (WATER)
NORTH END OF FORMAL GARDEN
8
LOW ROOM ROBOT PAPER
MACHINE ROOM 3 BUTTONS
MARBLE HALL BRICK
PATH NEAR STREAM
FORMAL GARDEN
DINGY CLOSET
ROOM 8
CAROUSEL ROOM
TOPIARY HEDGE-CREATURES
TOP OF WELL
7
TEA ROOM (HOLE)
POOL RM POND
17
MENHIR ROOM ROCK
5
RIDDLE ROOM
PEARL ROOM
6
CIRCULAR ROOM LGE BUCKET
18
KENNEL
STAIR WAY
ODDLY ANGLED (DIMLY)
CEREBUS ROOM DOG
CRYPT ANTEROOM FEEL FREE
19
GLOWING BRIGHTLY
CRYPT
FLICKERING SERENELY
GLOWING
DIMLY GLOWING
LANDING ROOM

IBM 4A

SPECIFICATIONS

TRITON *TURBO XT* Personal Computer

MICROPROCESSOR: Intel 8088, 8/4.77 MHz clock speed (software selectable).

OPERATING SYSTEM: MicroSoft® Disk Operating System (MS-DOS).

MEMORY: 256K RAM (Expandable to 640K).

DISK DRIVE: One 5¼" double-sided, double-density, 360K thin-line mini-floppy, 48 tracks per inch.

VIDEO: RGB/composite color graphics display adapter.

INTERNAL EXPANSION: Eight standard user-accessible IBM PC card slots.

EXTERNAL CONNECTIONS: Standard parallel printer port, composite video, RGB, AC outlet.

POWER: 120VAC, 50-60 Hz.

BRIDGE BOX

4A MODE: Common video out, 99/4A video in.

XT MODE: Keyboard out, video in, XT Video in. Selectable power-up mode callable from TI BASIC or Extended BASIC. Concurrent processing. Five LED status display.

PRODUCT COMPARISON TABLE

FEATURE	TRITON TURBO XT PC	IBM PC-XT	TANDY 1000 EX
Microprocessor	8088 8 MHz/4.77 MHz	8088 4.77 MHz only	8088 7.16 MHz/4.77 MHz
Operating System	MS-DOS 2.11/GW-BASIC	PC-DOS 3.2/BASIC	MS-DOS 2.11/GW-BASIC
Memory	256K/incl. diagnostics	256K/incl. diagnostics	256K/incl. diagnostics
Keyboard	TI-99/4A provides complete key set	101-key detached	90-key non-standard attached
Video Card	Color composite video & RGB	Color composite video & RGB	Color composite video & RGB
Disk Drive	One 5¼" double-sided, double-density, 360K thin-line floppy	One 5¼" double-sided, double-density, 360K thin-line floppy	One 5¼" double-sided, double-density, 360K thin-line floppy
Internal Expansion	Eight standard 11" slots	Eight standard 11" slots	One non-standard "PLUS" slot
Printer Connection	One parallel	One parallel	One parallel
IBM Software Compatibility	True Compatible	IBM	True Compatible
Software Included with DOS	Text edit, filer, calendar, telecomm, phone dir. & dialer, calculator, help system, menu system, DOS & BASIC tutor, ramdisk, print spooler. Includes files to automatically install all features.	None	Text edit, worksheet, filer, calendar, telecomm, paint, phone dir. & dialer, calculator.
Warranty	One year	90 days	90 days
Toll-free Support	Yes	No	No
Price (incl. DOS)	$569	$2559	$829

ASSEMBLY LANGUAGE

FOR THE LAYMAN

WITH ALLAN WRIGHT, MYARS

Well I have finally dragged myself away from other distractions and back to the trusty T.I. keyboard. Since my last article I have had time to think about the articles and where they should be headed. Much to my delight, at long last we have an Assembly Language group started and meeting regularly. It has become obvious that each and every one of us has his own direction that he wants to go in Assembly. This has lead me to believe that after several introductory articles, tutorial type articles are not what is needed. Instead it seems to me that articles giving useful hints for the novice and some useful routines would be more advantageous. This also will allow the articles to cover a broader range of information.

For anybody who is just starting into Assembly, might I suggest that you read my first three Assembly articles, the first of which appeared in the August 1986 News Letter. One final word of warning!! Assembly is not easy to learn. On the other hand it is not hard either. It is just like all other things in life, if you are prepared to put in time and effort, then the rewards will be there for you.

REGISTERS. The C.P.U. in our little computer has three internal registers.

* THE PROGRAMME COUNTER.

* THE WORKSPACE POINTER.

* THE STATUS REGISTER.

You will recall that when Super Bug was run in a previous article we had to load some information into the PROGRAMME COUNTER and the WORKSPACE POINTER. This was done so that our programme could be run by SUPER BUG under controlled conditions. This allowed us to then look inside the computer during execution of our programme. What did we load??? The PROGRAMME COUNTER was loaded with the address of the first instruction of our programme. The WORKSPACE POINTER was loaded with the address of REGISTER 0.

When a break point was reached after our programme was started by SUPER BUG some information was shown on the screen. This was in order,

* The WORK SPACE POINTER

* The PROGRAMME COUNTER.

* The STATUS REGISTER.

The point here is that each time we hit a break point the WORKSPACE POINTER HAD NOT ALTERED it still contained the address >A004. This is the location of REGISTER 0 in our programme. Where as the PROGRAMME POINTER had changed and was pointing to the next instruction to be executed. The STATUS REGISTER showing the record of the results of the last instruction executed (more on this REGISTER in a later article).

WORKSPACE

Knowing the above information about the WORKSPACE POINTER. Would it be possible to change that address in the WORK SPACE POINTER during a programme?? If we could, then maybe we could have more than one WORKSPACE. What does this mean?. Well if we can do that then we could have one WORKSPACE for the main body of our programme and another WORKSPACE for our subroutines. This would allow us to store information in our WORKSPACE without fear of it being lost during the execution of a sub routine. Can we do that on our little down trodden T.I./994A? YES!

When this is done it is called a CONTEXT SWITCH.

BLWP There are two instructions which can be used to cause a CONTEXT SWITCH.

BLWP - BRANCH AND LOAD WORKSPACE POINTER.

XOP - EXTENDED OPERATION.

This discussion will be confined to BLWP. To return from a CONTEXT SWITCH the instruction RTWP is used. RTWP (RETURN WITH WORKSPACE POINTER).

One operand is required for BLWP, it can use any one of the 5 general addresssing modes. The operand indicates the address of a two word vector that is used to perform the CONTEXT SWITCH.

The contents of these two words are:

FIRST WORD:
ADDRESS OF THE SUBROUTINE WORKSPACE.

SECOND WORD.
ADDRESS OF THE SUBROUTINE ENTRY POINT.

An example of the first two instructions of a subroutine with the LABEL SBRT, which is to be called by BLWP are as follows:

```
   BLWP  @SBRT        BRANCH TO SBRT
      ----
      ----
    PROGRAMME
      ----
      ----

   SBRT  DATA SBRWS   ROUTINE W/SP
         DATA ENTRY   ROUTINE ADDR
```

This of course means that the labels SBRWS and ENTRY have been equated at the start of the programme. SBRWS to the address of R0 of the WORKSPACE for the sub routine and ENTRY to the starting address of the sub routine.

A more commonly used method of indicating the subroutine entry address is to use the following code:

```
   SBRT  DATA SBRWS,$+2
```

The code $+2 in the DATA loads into the second word of the vector the current location (the address in the programme counter) plus 2. That is the first word of the vector will hold the address of SBRWS. The second will indicate the entry point for the sub routine which in this case begins immediately following the second word of the transfer vector. Using this method saves on the use of a label and also allows us to write subroutines separately to the main programme without having to be concerned about using an already used LABEL. A common WORKSPACE LABEL eg. SBRWS is used in all the sub routines.

Are you wondering how the computer returns from a CONTEXT SWITCH?

When the BLWP instruction is executed ie. a CONTEXT SWITCH is performed. The contents of the WORKSPACE POINTER, PROGRAMME COUNTER and STATUS REGISTER are saved. Saving this allows programme execution to continue at the next instruction following the BLWP. After the sub routine is completed. REGISTERS R13,R14,R15 of the WORKSPACE pointed to by the first word of the transfer vector are used to save this data.

R13 CONTENTS OF W/SPACE POINTER.

R14 CONTENTS OF PROGRAMME COUNTER.

R15 CONTENTS OF STATUS REGISTER.

If you intend to used R13,14 or 15 in your subroutine then you MUST save their contents before they are used. Then restore them before attempting to return from the subroutine. Otherwise you could end up anywhere.

RTWP - RETURN WORKSPACE POINTER.
This instruction reverses a CONTEXT SWITCH. It requires no operand. When executed it returns the contents of R13,14 and 15 to the WORKSPACE POINTER, the PROGRAMME COUNTER and the STATUS REGISTER. RTWP is normally the last instruction performed in a subroutine.

SOURCE LISTING
The listing below contains two User written subroutines which are BRANCHED to buy using BLWP.

They are CONT1 and CLEAR1. CONT1 is used to display the text message on line 0007 of the programme and to accept any key stroke to allow programme flow to continue. The second subroutine CLEAR1 clears the screen by filling the screen with the blank character >20.

CONT1 is at address >00A8. Look now at line 0028 of the source listing.

```
BLWP @CONT1
```

This instruction is at address >008C, the address immediately following contains the address >00A8 (The address of the vector for CONT1). Address >00A8 (the first word of the TRANSFER VECTOR) contains the address of the sub routine WORKSPACE >8300 which was EQUATED to SBRWS at the top of the programme. The next address >00AA (the second word of the TRANSFER VECTOR) contains the ENTRY point for the sub routine >00AC.

The second subroutine, CLEAR1 is at address >00CC. The BLWP for this sub routine is on line 0030 of the source listing at address >0096. As above, the next memory location contains the address of the transfer VECTOR >00CC.

TAKING DATA INTO A SUBROUTINE.

The ability to take specific pieces of data into a subroutine allows the programmer to write more universal subroutines. To illustrate this point and to show how it is done I have taken data into both CONT1 and CLEAR1. In the first case CONT1 the data which is taken into the routine allows the screen location, which message and the length of message to be changed each time the routine is called if so desired.

The data taken into CONT1 is on line 0029 of the source listing. The first data item >02E6 is the screen location start of MESS2. MESS2 is the text which is to be written to the screen and the next data item 22 is the length of the text to be written.

When a CONTEXT SWITCH is performed the address of the next instruction after BLWP >0090 is saved in R14 of the new WORKSPACE. In the case of CONT1 what the computer thinks is our next instruction is in actual fact data which we want in the sub routine. The next instruction that the computer should execute is at memory location >0096 NOT >0090. We have at our disposal an addressing mode called REGISTER INDIRECT AUTOINCREMENT. This mode is used to get the data we specified into the subroutine. Line 0037 of the source listing uses this mode.

```
MOV  *R14+,R0
```

This instruction will move the contents for the address pointed to by R14 into R0, then add two to the address in R14. In the above example the contents of >0090 will be moved into R0, ie >2E6 the start screen address for the text will be placed into R0. Then R14 will autoincrement to >0092, it's contents, the address of the text MESS2 will be moved into R1 on line 0038 of the source listing. R14 will autoincrement to point to >0094 the length of the text to be written which is moved into R2 on line 0039. R14 is autoincremented again and now points to >0096 the next instruction to be executed when the programme returns from the subroutine.

This same proceedure is used to get data into CLEAR1.

CONCLUSION

I hope that I have been able to express myself clearly enough for you to understand the procedures described. I also hope that you will find this article useful. I am going to continue writting them on themes that I think will be useful to the novice programmer. Hopefully other members will be inspired to contribute articles also.

Before Christmas I received a very pleasant letter from Garry Christensen of the Brisbane User Group. Garry had enclosed a disc of useful sub routines in Assembly language. Garry has done an excellent job with this and has included on the disk first class documentation. If you have not seen this disc then I suggest that you approach Al Lawrence ASAP and get one. The information in this article should allow you to start using Garry's routines within your own programmes and give you a reasonable chance of understanding

what is happening. Using them will help with your learning curve.

Next month a discussion on BL. In the meantime however some suggested reading.

The Assembly Group has been set the task of reading chapters 1 through to and including 5. Also to type in and get running the programme example in chapter 5 of that book. Any questions which come out of the Group in regard to this will be printed and answered in the next article. I will keep noting in these articles what the Group is doing. If you can't attend the Group then this should allow you to keep with us. Any questions will be welcomed.

From the E/A Manual
Predefined symbols page 53.
Context switch page 45.
Data page 225.

From Ira McCormack "Assm. Lang Prog"
Addressing Formats pages 59-68.
Branch and Sub routine instructions chapter 16.

Until next month,
Allen Wright

LISTING BEGINS OVER PAGE.....

FOR SALE
PE.SYSTEM

TI.- CORCOMP 9900 MINI SYSTEM
System includes:-
* TI-99/4A console with XB.
* Full Corcomp 9900 mini PE.
  - DSDD disk controller
  - RS232 card (RS232/PIO)
  - 32K memory expansion
* 1 only AMUST DT100 Printer(PIO)
* Speech Synthesizer
* Assorted other TI.Modules

ALL BOXED AND FREIGHTED

$900o.n.o.

Prepared to break the system up for reasonable offers.

CONTACT:-
VERN TAYLOR
c/o Radio Australia
DARWIN - N.T.
5790
Phone > 089-270730

or Contact - The Secretary
HV99ers
6 Arcot Cl.
TARRO - NSW.
2322

for further details.

WANTED

TI-RS232 card
PE-Box only

CONTACT:-
ALBERT ANDERSON
6 Arcot Cl.
TARRO - NSW.
2322
Phone > 049-662602

```
99/4 ASSEMBLER                                                   PAGE 0001
VERSION 1.2
 0001            * FIFTH TUTORIAL PROGRAMME
 0002            * 1-03-87 FILENAME SOURCE=EA/PROG/5D
 0003            *                 OBJECT=EA/P/5DOBJ
 0004                    DEF  START                      PROGRAMME NAME
 0005                    REF  VSBW,VMBW,KSCAN            REFERENCES
 0006 0000   48  MESS1   TEXT 'HUNTER VALLEY 99'ERS'     FIRST MESSAGE
 0007 0014   50  MESS2   TEXT 'PRESS ANY KEY TO CONT.'   SECOND MESSAGE
 0008 002A   20  MESS3   TEXT '                  '
 0009      837C  STATUS  EQU  >837C                      STATUS REG EQUATE
 0010      8374  KBDSTAT EQU  >8374                     KEYBOARD DEVICE
 0011      8375  KEYVAL  EQU  >8375                     KEY VALUE RETURNED
 0012      8300  SBRWS   EQU  >8300
 0013 003E   20  ANYKEY  BYTE >20                       ANY KEY PRESS MASK
 0014 0040 0000  RETURN  DATA >0000                     SPACE TO SAVE R11
 0015 0042       WSREG   BSS  >20                       SPACE FOR MY WORKSPA
 0016 0062 C808  START   MOV  R11,@RETURN               SAVE RETURN ADDRESS
      0064 0040'
 0017 0066 02E0          LWPI WSREG                     CREATE MY W/SPACE
      0068 0042'
 0018 006A 0204          LI   R4,>2C6                   LAST SCN POSITION
      006C 02C6
 0019 006E 0206          LI   R6,>04                    R6=NUMBER OF TIMES
      0070 0004
 0020 0072 0205  SCRN1   LI   R5,>06                    FIRST SCN POSITION
      0074 0006
 0021 0076 0201  SCRN2   LI   R1,MESS1                  ADDRESS OF TEXT
      0078 0000'
 0022 007A 0202          LI   R2,20                     NUMB CHARS TO SCREEN
      007C 0014
 0023 007E C005          MOV  R5,R0                     LOAD ADDR INTO R0
 0024 0080 0420          BLWP @VMBW                     WRITE THEM!
      0082 0000
 0025 0084 0225          AI   R5,>0020                  NEW SCN POS FOR MESS
      0086 0020
 0026 0088 8105          C    R5,R4                     ADDRESS OF MESS1
 0027 008A 16F5          JNE  SCRN2                     NUMB CHAR TO SCREEN
 0028 008C 0420          BLWP @CONT1                    BRANCH TO CONT1
      008E 00AB'
 0029 0090 02E6          DATA >2E6,MESS2,22
      0092 0014'
      0094 0016
 0030 0096 0420          BLWP @CLEAR1                   BRANCH TO CLEAR1
      0098 00CC'
 0031 009A 0000          DATA >0,>2FF                   SCREEN POS TO CLEAR
      009C 02FF
 0032 009E 0606          DEC  R6                        R6-1
 0033 00A0 16EB          JNE  SCRN1                     NOT ZERO? GO AGAIN
 0034 00A2 C2E0          MOV  @RETURN,R11               RETURN ADDR TO MY R1
      00A4 0040'
 0035 00A6 045B          RT                             BACK TO EA.
 0036 00A8 8300  CONT1   DATA SBRWS,$+2                 SCREEN ADDRESS
      00AA 00AC'
 0037 00AC C03E          MOV  *R14+,R0                  ADDRESS OF TEXT
 0038 00AE C07E          MOV  *R14+,R1                  NUMBER OF CHARS TO S
 0039 00B0 C0BE          MOV  *R14+,R2
 0040 00B2 0420          BLWP @VMBW                     WRITE THEM!
      00B4 0082'
 0041 00B6 04E0          CLR  @STATUS                   CLEAR STATUS REGISTE
      00B8 837C
```

```
 99/4 ASSEMBLER
VERSION 1.2                                          PAGE 0002
 0042 00BA 0420  CONT2  BLWP @KSCAN                  BRANCH TO KSCAN
      00BC 0000
 0043 00BE 9820         CB   @ANYKEY,@STATUS         bIT 2 SET?
      00C0 003E'
      00C2 837C
 0044 00C4 16FA         JNE  CONT2                   NO? BACK TO KSCAN
 0045 00C6 04E0         CLR  @STATUS                 CLEAR STATUS AFTER U
      00C8 837C
 0046 00CA 0380         RTWP                         BACK TO MAIN PROGRAM
 0047 00CC B300  CLEAR1 DATA SBRNS,$+2
      00CE 0000'
 0048 00D0 C13E         MOV  *R14+,R4
 0049 00D2 C03E         MOV  *R14+,R0
 0050 00D4 0201         LI   R1,>20                  BLANK CHAR DATA
      00D6 0020
 0051 00D8 0420  CLEAR2 BLWP @VSBW                   WRITE BLANK!
      00DA 0000
 0052 00DC 0600         DEC  R0                      DECREMENT SCREEN POS
 0053 00DE 0280         CI   R0,R4
      00E0 0004
 0054 00E2 16FA         JNE  CLEAR2                  NO? BACK TO CLEAR2
 0055 00E4 0380         RTWP                         RETURN TO MAIN PROGR
 0056                   END
 0000 ERRORS
```

FOR SALE

4A PE SYSTEM(complete)

* TI. 99/4A Console
* TI. PE-Box
* TI. RS232 Card
* TI. DSSD Disk Controller
* TI. 32k Memory Expansion
* Twin Slimline DS Drives
  (mounted in PE-Box)
* Heaps of modules,XB,ED/ASS,
  DM2 etc,etc.

$600.00

CONTACT:-
VIV BRUNNER
61 Broughton Rd.
KEDRON - QLD.
4031
Phone > 07-8574829
VIATEL> 785748290

or Contact - The Secretary
HV99ers
6 Arcot Ct.
TARRO - NSW.
2322

for further details.

Peter Smith has the following items for sale. Any inquiries contact Peter on 049-336164.

FOUNDATION 128K CARD

Many uses - this card is surplus to my needs. This card can be updated with an EPROM from Myarc to be set up to use 128K CPU and XB11.1 (Myarc) - Great gear. Price negotiable.

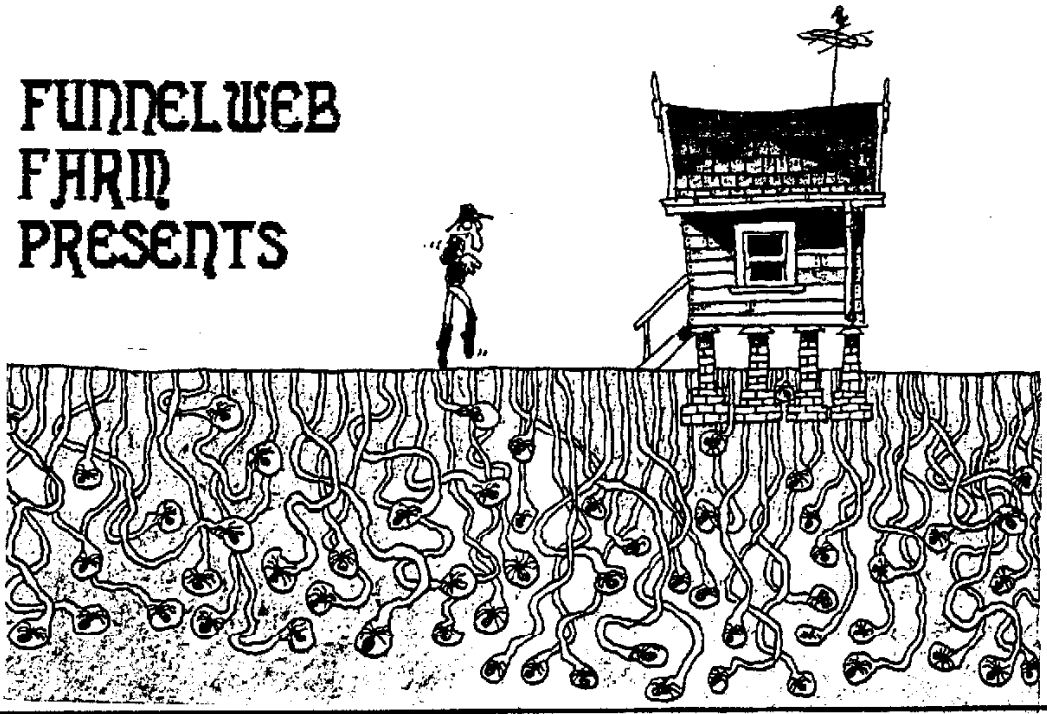
MODEM

Peter Schubert's card in a box 300/300 baud modem. This is a cheap way to enter the exciting and useful world of communications. Price $80

BROTHER M1009 PRINTER

Small, but big on ability and heart (sounds like the Editor). I am updating to its big BROTHER (get it?), so wont need this little beauty. Sacrificed at $200.

FUNNELWEB FARM PRESENTS



ENTOMOLOGY CORNER 11

It seems a long while since I wrote the last article in this series, Nov 86 in fact, and the due date is rapidly approaching for the Mar 87 HV99 Newsletter. Our Northern Hemisphere readers will just have to remember that this is not the season when much computer hobbying gets done in Australia. There isn't really all that much to report so most of this month's ravings are in the accompanying article on the workings of TI-Writer's Editor. The pressures of HSC have closed in on Will, so don't expect too much from him for a while.

Despite its having been summer and surf season there have been some developments in the FUNNELWEB program package. The most notable of these was just before Xmas. I had decided that it was time to have another closer look at the internals of the Editor and the net result is that it has been speeded up enough that the old problem of loss of keystrokes during automatic wordwrap at the end of a line has been largely eliminated. Not entirely as a fast typist can still beat it. I don't come in that category and the only way I can beat it is by fluttering my finger on the same key without lifting it. Some perverse law of computing almost always makes double letters split over the ends of lines and these are the only ones I ever get any sped on. I think the most interesting part of this is that the old TI-99/4a hardware can do the job after all. No fancy new computers like the 99/8 or 9640 are needed to solve that particular problem. All it takes is for the computer to be programmed properly. TI's code was in fact faulty, a slight blunder in program design rather than a detail coding mistake, but more of the FWB modifications previously made to the Editor were involved in the speedup process. The same problem still exists in the later European issue of TI-Writer so TI had never bothered trying to fix this most obvious deficiency or else couldn't figure out how to do it. I suspect they weren't really trying any more, first because the 99/8 was coming with speed to burn, and once

that was canned no-one cared any more.

The most recent update was with Vn 3.5 of DM-1000 when the source code finally arrived last week. This has been worked over to the same effect as with Vn 3.3 earlier and will find Horizon RAMdisks at CRU bases >1200 and higher. I have sent the details back to Ottawa and I firmly hope that they incorporate it for once and for all in DM-1000. The Vn 3.5 DM-1000 itself has some new features which will be welcome, and some bugs have been fixed. A prospect for the future is to add Will's Myarc formatting routine to eliminate the problems with Myarc's DD tracks being different from everyone else's. The trouble with that is that no doubt it will not be compatible with Myarc's FDC80 upgrade. I'm not even sure whether the boot disk tracking will continue to work with that new ROM. It's about time that Myarc gave up the TI type secrecy on the internals of their products. Many people won't want to go through with the 9640 what they did with the 99/4.

An evil, insidious bug lurking in the central menu screen loader has also finally been exposed and squashed. This was most noticeably causing problems with the loading of the c-Compiler from the menu screen - sometimes it would and sometimes it wouldn't. What the bug was doing was to interfere with the flag that called for the loading of the E/A utilities, and the compiler would load only if these were already present. While I was at it I revised CO slightly so that it now meshes in completely with REL3 (compiler Vn 2.1) of c99. Also if CO,CP,.. are loaded by name from the loader screen, it now knows not to try returning to FWB which is no longer there, but still saves the filename in the mailbox.

Another bug that turned up very recently crept in during the change to Vn 3.4. Nobody has reported it yet, and it only makes it presence felt when SD is done on a disk with 100 or so files on it. It doesn't hurt the Editor but crashes the machine on return to FWB. Just a pair of pointers involved in juggling memory that had escaped updating. Issues of FWB carrying dates of Mar 87 or later will have this corrected.

We have also loosened the criteria by which FWB decides whether there is a valid name in the mailbox to be passed on to utility programs. Now it inspects only the first 3 characters for "DSK" or "RD." which allows disk volume name specification, or the universal name for the Myarc RAMdisk, the one it always responds to no matter what drive it is emulating. Also filenames for the utility loaders may be up to 23 characters long. With a bit more foresight this should have been 25 but that will have to do until the next major revision if that ever occurs. This means that volume name access may be used for utility files. The longer names may also be used in the LOAD program for the XB level of user list. If volume name specification of assembly language utilities is used here then the auto boot-disk tracking must be disabled by the K value for that entry. This is because boot disk tracking and volume name specification are incompatible in principle and practice. The User List accessed from the central menu screen and defined with ULINSTL still only accepts 15 character names, and this cannot be increased because there just aren't enough bytes available.

The 'fairware' system struggles along, maybe as well as one could expect in the TI market. Maybe us smalltown folk are right in thinking that big cities are bad places. Not a single response from the whole of Sydney, and just one from the whole of the Greater Los Angeles area, other than someone making a fairware survey. Nothing is black and white here, but the TI-99 market is one where even commercial software on the whole has been very reasonably priced. Not like for other machines where absolutely exorbitant prices of software subsidise massive advertising and brainwashing campaigns, battalions of lawyers, and layers of high margin resellers. Fairware for a machine out of production for over 3 years now is inevitably a much less

portentous business, but even here there are some ripe double standards operating. For instance a recent issue of TI*MES from the UK carried a reprint of an article on piracy, a veritable cri de coeur from the author of DISkASSEMBLER, followed in the next issue by a review by the same author of Grampacker, a utility for the MG device. Guess what was the first entry on the personal menu shown. Need I say more ? While on that subject we received a genuine original of DkA as a Xmas present. Will promptly rewrote his backup utility for the Myarc as he couldn't be bothered searching for his previous effort and swapping disk controllers. He is going to release it as 'fairware' with a send $$ for source code rider, to our FWB friends anyway. He has been extremely disappointed with the response to Disk-hacker, from a few established friends and the odd previous stranger or two. This is despite evidence of frequent downloads from USA networks. That is the reality of the 'fairware' system. Readers of this series have probably noticed that I have a distaste, frequently expressed for hype. There is still too much of it around, but there are fortunately solid and well designed programs out there that will help prolong the life of our computer investment. I would be quite disappointed to find that anyone competent to use Funnelweb didn't find it as least as good as I have ever made it out to be.

The name 'Funnelweb' and derivatives have always been our private joke which occasionally causes some comment. Just as well names of programs don't bite. Of course if you are thinking along those lines what if a typical macho male "power user" of IBM PC/ATs all loaded down with hard disks and oodles of extra little segments of memory were to catch the dread "microsoft" affliction.

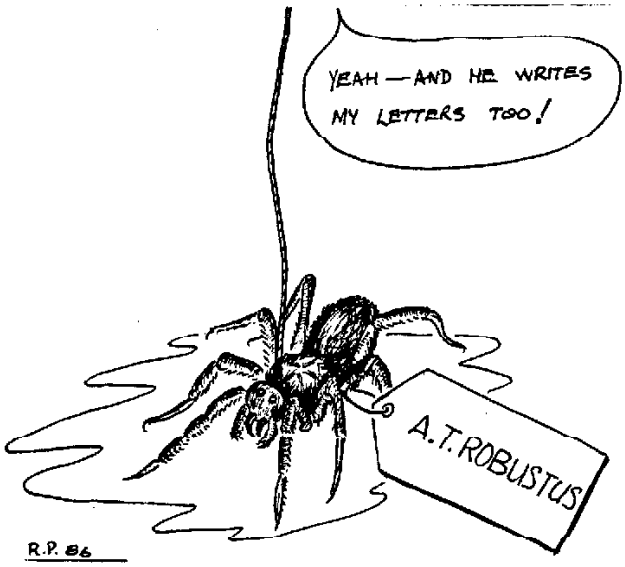
From news received from places as far afield as Sweden it would appear that Myarc 9640's are now being delivered. What is not yet clear is if the operating system and software are up to scratch. At the moment we are saving up for one, influenced perhaps as much as anything by the investment in time spent on and liking for 9900 family assembly language. I am also aware that the computer board is only a small part of what is involved, because if you are not going to throw away most of the graphics capabilities of the 9640 you are going to be in for the cost of a color monitor with far better resolution than the television receivers which are adequate for the 99/4a. William still wants an Amiga, but is realistic enough to wait for new models of that to appear. My own assessment is that if we were to get an Amiga first, we would get so involved with 68K coding that we wouldn't bother with the 9640 and the 99/4a would be relegated to being a backup machine for routine work with no further program development being of interest. The 9640 and 99/4a would have a lot more in common for development purposes if we did have the new machine.

In fact the updated 99/4a equipped with 2x360K drives and RAMdisk(s) is still an excellent computer, 40 column display and limited keyboard notwithstanding. If Funnelweb is used as the operating system shell then it loses very little in comparison with other machines over ease of use either. You should also ask your friends with other machines what happens to the contents of their RAMdisks when they have program crashes and lockups. Whatever happens your TI-99, with programs like FWB and Clint Pulley's c99 you can get in practice for moving on to newer machines. There is a subtle trap with the TI-99 in that once you get involved with TMS-9900 Assembler, you find that it is the most elegant for any micro that I know of and a real pleasure to write as such things go. The trap is that the TI-99 is a small enough machine that you are not forced into higher level languages by the tyranny of program size, so I never seem to get around to p-code or c99.

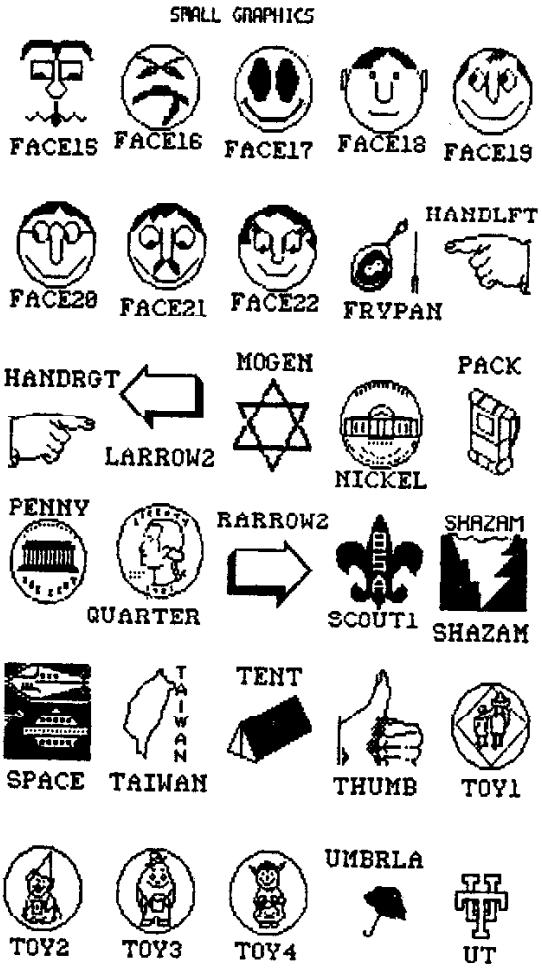
That brings up a recent advert seen here recently for the MG/Triton IBM cloner/adapter. As far as I can see you might as well just get a genuine straight-forward PClone. After all one of the prime

reasons for leaving TI land is to have a full-size keyboard. The real need here is not a half-assed PClone but merely a program which will transfer data and text files to MS-DOS disks, that one could then take to work and use with PClones there, or IBMs if you work in a high budget operation. The problem we have found here is not doing the Myarc direct access routines for writing to MS-DOS disks, but actually finding out in detail how MS-DOS lays out its directories and files on the disk. This information is supposed to be public domain but has proved extremely difficult to track down in usable form. I'm not sure that this level of understanding of detail isn't far more widespread in the TI-99 community than amongst IBM users. Maybe it's because of TI's attempts at secrecy made it more necessary for TI owners to dig into the details once they were on their own.

I think the stream of consciousness has run dry so it's time to close off this epistle to the Texans. --- Tony McG.



CSGD III



CREATED WITH GRAFHX.JOYPAINT
.JOYPAINT-PAL AND TI-ARTIST.

Reviews

FROM UP THE VALLEY.

Review of club disk for month of March.

These reviews were conducted by our members further up the Valley. Thanks to Peter, Ron and Noel for their contribution, and to Smithy for the fancy title page appearing opposite.

The first reviews are by Peter Smith.

Jolly Roger.

I found this "catlogue" program most useful - it allows easy personalisation of a disk catalogue.

My review, unfortunately, is not as complete as it could be as my printer, which has served faithfully for many years, has just been sent for repair. This makes full evaluation difficult, however I can comment on what aspects of the program I did use.

The print-out can be personalised by your name, other identifiers, if desired, and a (horror of all horrors) picture of a naughty pirate being printed at the top.(I hope the author of the program is not insinuating that the user of the program might do something un-nice, like copying a program subject to some restriction!).

Users can then make use of the most important function of this program..The ability to easily add comments about particular files listed.

This is a particularly useful feature as often file-names bear little resemblance to program names listed on menues etc.

Other comments regarding usefulness, loader types or the source of a file (eg PRB MEMBERS) might make use of the programs listed, easier to understand.

Although very easy to use and undoubtedly useful, I look forward to the next logical step which would see the commented catalogue saved to disk and updated by the program itself, when files are added and deleted on the disk.

This would save having to re-comment each time an updated catalogue was made.

Overall a very useful program especially for "full" disks.

AIR-DROP.

Being spoilt by the fast-action m/c games available, it was hard to maintain a state of wakefulness while the boring prelude and instructions(and complicated ones they are too) were displayed.

The game is not what one would call mind-bending, but could be reasonably amusing and challenging to people liking this type of game. Basically guide a plane over a forest and drop a parachute in a landing-zone (LZ).

To add demerits to this game ---it is obviously American and does not use our enlightened metric system and so, for this reason alone I would not recommend this program for children, unless amended to metric form.

SIAM

What can one say? This is the legendary Towers of Hanoi and involves moving disks from post 'a' to post 'c' in as few moves as possible. The only rule is that no large disk can be placed on a smaller one. (By the way there is a post 'b').

A timeless strategy game which will frustrate and amuse players of all ages.

STRIPPER.

Wow... No comment.

VIC-SEWERS

Collapsing sewers interrupt traffic flow through London's roads. Your job is to maintain a traffic flow and repair them with work gangs.

Steven Shaw has written a fine set of code which gets very involved and can provide an interesting night's entertainment.

6 levels to choose from and lots of planning to do.(To me about as interesting as watching the mushrooms grow on the walls of a sewer but then .. each to his own.)

WORD POWER 4

Based on 'Enrich Your Word Power ' from the **READER'S DIGEST.**

A good idea but possibly cheaper to buy R.D. than a computer.For those of you who like the real thing I found it challenging and of possible use with children.

It has the facility to use speech.

XOX

Another TIC-TAC-TOE program(We enlightened people of Australia call it naughts and crosses)

Good for kids on a wet day and with no paper.

The following reviews were carried out by Ron Pratt.

Hi there! This is the first time I've reviewed anything , and I felt a bit excited when asked to do so, but when I ran the three programmes I was given, there seems to be only one which might evoke any feelings of excitement in the user, and that's "Wordcount.".

It's not as handy as some I've seen, in fact all it does is count the number of words in a given file: but it does that very well.

After loading **WORDCOUNT** and inputting the file name you wish to count, the lines on your file zap past a window on the screen while a count is made of its length. Excellent for those wishing to check the length of essays etc., but I'm not sure what else you'd do with it.

After the joy of the foregoing, I'm afraid the rest is all downhill. **WLD/WORDS** is a game based on the various names of world currencies. On running this game, currency names, chosen at random, zig zag down the screen very quickly. In doing so, they sometimes overlap, making it difficult to read them. If that wasn't enough to tax your knowledge of "Crowns,Kroners and Kopeks", random letters are replaced with asterixes on each run down the screen.

Sounds good, but there are a few bugs. After failing to get the word right, you are asked if you would like to know the answer - fair enough, but it is flashed by so fast even speed readers miss it. Even if you are clever enough to get it right, the programme says you're wrong! Probably only minor bugs, and easily fixed if you know how.

CYBER/DICE seems quite a clever programme, with good graphics and plenty of combinations of dice games, but I don't know anything about dice games, so I can't give it a fair review.

The following 2 programs were reviewed by Noel Cavanagh.

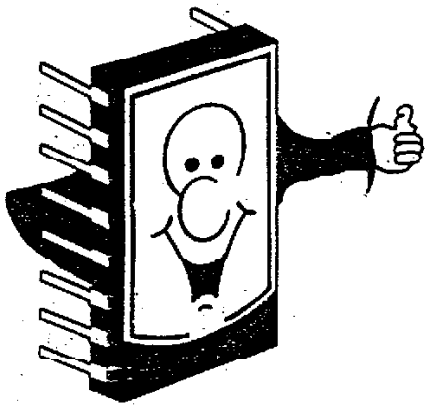
PROPERTY CALCULATOR

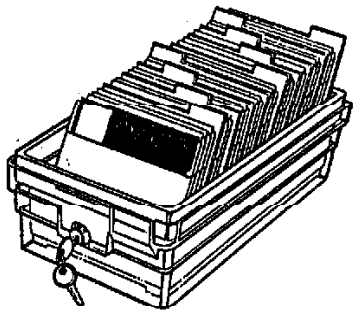
A useful program for house loan payments. In this age when interest rates are moving almost yearly, it is handy to be able to calculate what the latest changes mean to your personal situation.

I can see many uses for this program, especially with our budding house owners and Real Estate sales people. It would be of use for those at the cross roads - buy or rent?

WILD KING

Just another munch em, amazing, hunt em down, gobble em up type program - one played, easily forgotten.





SOFTWARE LIBRARIANS NEWS

BY AL LAWRENCE

HI 99'ers,

****

RE-INKING PRINTER RIBBONS

--------- plus ----------

LEAVING CLEAN FINGERS

****

As the demand for RIBBONS by my printer is insatiable and reading lots of articles on re-inking over the years that all had one thing in common - you always left a trail of FINGERPRINTS even Long John Silver could track down.

When out of the Black Hole a voice cried out "Hear me!!" ZZZZ "Hear me!" Have you ever heard of XXXX? Well, if you lubricate my tonsils with it I will tell you. Get PLUS from the Newsagent. PLUS is an ink used for Numbering Machines, quick drying and lubricating. If your ribbon cartridge has a felt roller or pad, put a few drops of PLUS on and wind forward.

If, like mine, yours has neither but has a place you can insert a section of felt pad or foam into somewhere, do this and then soak some PLUS onto it and wind forward. After the pad is soaked and ribbon covered in ink, a few drops at intervals as required will keep the print black and fingers white.

Thanks Ron if you're still down thar shovelling,put some PLUS into it.

** New Fairware Disks **

SIDEWAYS
A fairware disk that prints out MULTIPLAN and TI-WRITER files sideways. I have not had time to review this But our Treasurer is trying it out.

SARGON
A Public Domain game of Chess by Mike Swiridenko of Canada. Loaded via Funnelweb or E/A 3.

This version is a TI-9900 Assembly translation of original Z-80 SARGON by Dan and Kathe Spracklen.

There is no way I was going to win, it would not accept any moves I made! Reading the instructions did not improve it any, except to inform me that not much testing had been done and a few BUGS may be around!! but inputs had been idiot proofed?? The graphics are impressive and it is noted that the author does not want financial gain, but help to improve it. Good possibilities when you can move (algebraic chess notation). Possibly I have a corrupted copy, as you can set up to analyze games and SARGON flashes then moves.

BUZZARD
Another P.D game. Loadable as above and fast moving maze/Pacman, eat the dots type game by Randy Jones. I have put them on the same SS/SD disk for no reason other than to give you value for your money - it getting near the first of April.

TI-DIAGNOSTICS
At last released into the Public Domain by TI as a good will gesture to all the Faithful. 2 disks - one for use with Mini Mem & one for XB includes some paper work to fill your Library with. (N.B. this makes worthwhile reading). This software maybe too late as we now have Hardware to do the same job. Interesting never the less and it is

available for media cost only, plus postage where required. It will test the following:

1 P-CODE 2 P.E.B. 3 Impact(PIO/Serial)Printers. 4 RS232 5 MODEM 6 SPEECH 7 Disk Exerciser ?? etc. etc.

SILVERWOLF
A disk full of XB UTILITIES that is complete with Documentation on the disk. Gives ASCII control values, true lower case char sets, ability to save and recall text into the High Memory, Transliteration codes.

LIMA User Group
A disk full of top class programs and a good one if you have need to print and record cheques, as it does contain such a program with full documentation on it. You can if you like borrow the LIMA area Mag. "BITS,BYTES & PIXELS"to read and type it yourself if that is to your liking.

MASS TRANSFER 4.1.
Updated copies of this disk are now available.

All the Club Software is available on Demand. $4.00 P/P each Disk.

Keep on Computing.

Al Lawrence.

IN YOUR MAGAZINE THIS MONTH...

Random Bytes - 1st article from Bob Carmany, our US correspondent.

TI-Writer Editor - an inside view by Tony McGovern

TI-Writer Tip from Ottawa U.G.

Humour from Ron K.

FORTH with Richard Terry:-
- a Window routine
- Vectored Execution
- Oops

Geoff Shipton writes on his recent trip plus more!!!

Adventurers Map of ZORK II

The new IBM compatible TI

Assembly Language with Allen Wright
- Hints for beginners

Entomology Corner No. 11

XB Games reviews

plus much more!!!!