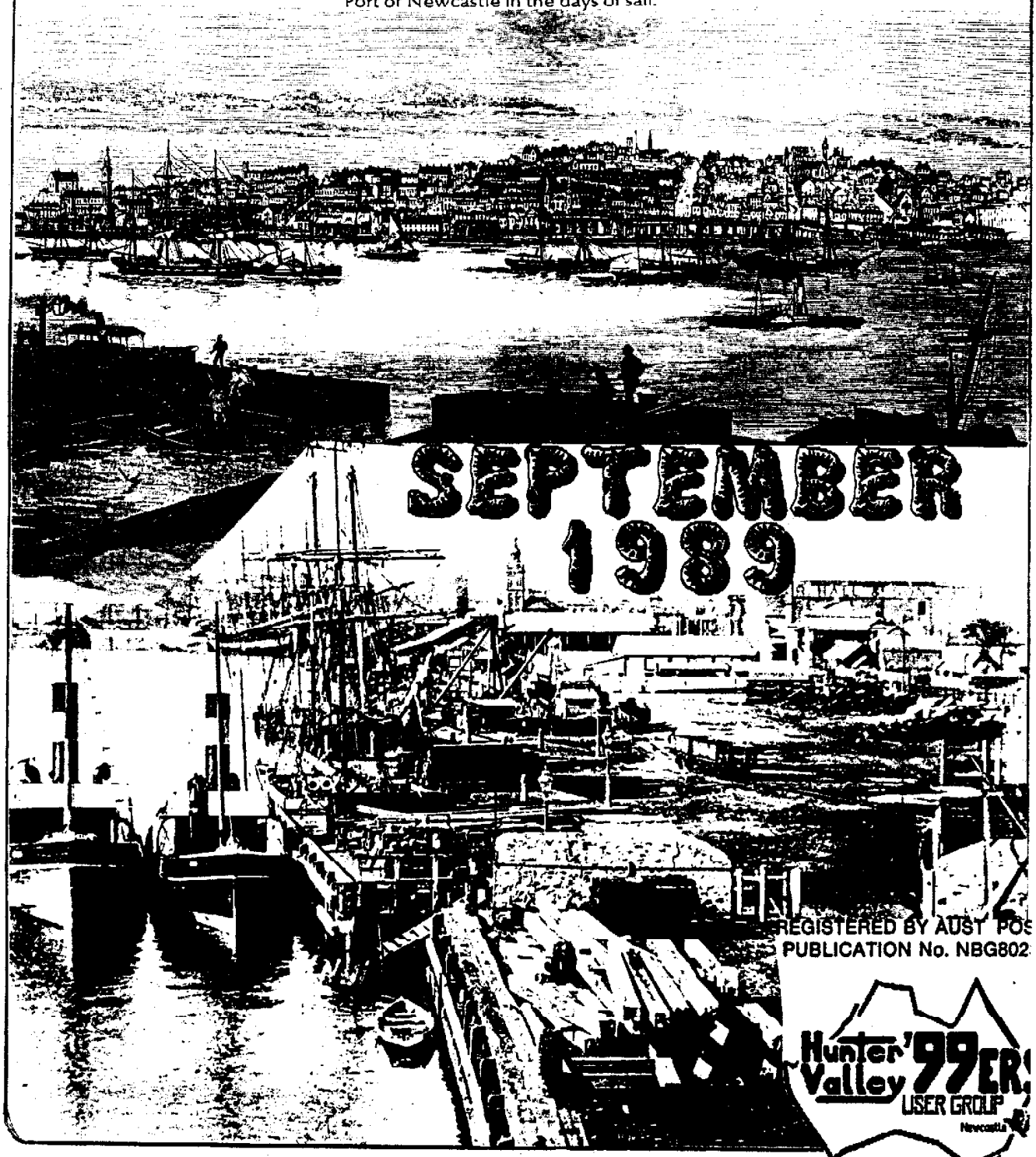
HUNTER VALLEY 99ERS USERS GROUP

HOME COMPUTER NEWSLETTER

Port of Newcastle in the days of sail.



SEPTEMBER 1989

REGISTERED BY AUST POS
PUBLICATION No. NBG802

Hunter Valley 99ER USER GROUP Newcastle

YOUR COMMITTEE

PRESIDENT

Peter Smith
8 Glebe St.,
EAST MAITLAND 2322
Phone 336164 V/tl 493361640

VICE PRESIDENT

John Paton
1 Parien Close,
RUTHERFORD 2320
Phone 326014 V/tl 493260140

SECRETARY

Brian Woods
9 Thirlmere Pde.,
TARRO 2322
Phone 662307 V/tl 496623070

TREASURER

Noel Cavanagh
378 Morpeth Rd.,
MORPETH 2321
Phone 333764

Software Librarian

Stewart Bradley
14 Hughes St.,
BIRMINGHAM GARDENS 2287
Phone 513246

EDITOR

Allen (Joe) Wright
77 Andrew Rd.,
VALENTINE 2280
Phone 468120

PURCHASING/HARDWARE CO-ORDINATOR

Alan Franks
822 Pacific Highway,
MARKS POINT 2280
Phone 459170

SOCIAL SECRETARY

Robert (Bob) MacClure
75 Deborah St.,
KOTARA SOUTH
Phone 437431

PUBLICATIONS LIBRARIAN

Ken Lynch
9 Hall St.,
EDGEWORTH 2285
Phone 585983

COMMITTEE MEMBERS

Don Dorrington
36 NELSON St.,
BARNSLEY 2301
Phone 531228

Tim Watkins
36 The Ridgeway,
BOLTON POINT 2283
Phone 592836

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.

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, ALL other club related correspondence should be addressed to The Secretary.

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.

The views expressed in the articles in this publication are the views of the author/s and are not necessarily the views of the Committee, Editor or members.

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.

From President PETE'S Quill

How long is it since we saw some new members join our club?

Ok! I know what you thought but really there are **lots of reasons** why a TI owner **SHOULD** join our club.

The fact that our machine has been orphanned certainly is the greatest incentive. With no official support available, the services, which the club provides, certainly makes the machine, which would otherwise be sitting in the cupboard, capable of being kept alive and productive.

To get one's "money's worth" from their investment and learn about computers, the support and wealth of knowledge available in the club, should certainly be tapped.

The reasons are clear... but **HOW** do we go about locating and enthusing these people.

Well there obviously are people interested in our machine.. Take for instance the example of Al Laurence. Every time he tries to borrow a book on the TI from his local library he finds at least one of the titles out. Someone out there is trying and with the number of the machines sold in this area it should not be surprising.

Members of the committee are trying to attract some attention to the club with notices in shopping centres and magazines in libraries.

Can you help? We certainly hope so.

"How?" is up to you, but certainly encourage anyone you know who has a TI to join or rejoin.

Meetings, hopefully, will become a little more interesting in the future. (At least they will be a bit different). We intend to have a 'demo' of some useful nature, then a demonstration by members of one of our classes followed by a practical activity, hopefully related to the demo, or a trouble/shooting, help/providing activity.

The software library will be active during this later period and with the updated list of software available, should do a roaring trade.

Talking about the software list, those of you who still have disks to be catalogued at home, please return them **NOW** (even if you haven't completed your job.).

On another front its great to see our newsletter getting better all the time. Congratulations Joe.

A couple of fellows have shown some interest in supporting the Melbourne group and attending their FAIR, so ask around if you are interested , you may be able to get a lift.

Till next month. Peter.

EDITORS LAMENT.

****

What a lovely position for a newsletter editor to find himself in!!. This month's newsletter is a couple of pages over the target size. I have held over the 'C' section of the Glossary until next month, plus several other articles have also been held over. **BUT** don't stop sending them in. If need be the target size of the newsletter can be taken out a few extra pages

We have a saying here in this country that everybody is entitled to his two bobs worth. That is everybody can have their say. That is especially true this month for Bob Carmany. His article managed to get mangled during printing last month. So this month I have included last month's and this month's. Two Bob's worth!!
This is not a plot to spread out the newsletter.!!

Christmas is fast approaching, I hope that you have started work on your contibution to the Christmas Bumper Newsletter.

I have also received a lot of letters this month, thank you to those people. I will reply to each and every one as soon as I have finished this month's magazine.

Joe Wright

Brian Woods REPORTS From the Secretary's ** Desk **

MAILOUTS

Our next great mailout should be underway in the near future. It has proven to be a huge success with our out of town members in the past, and soon some of the older newsletters in our publications library will go out to those members who, I am sure, are eagerly awaiting more news from around the TI world.

MONTHLY MEETINGS

October's meeting will feature a demonstration of Joe Wright's newly updated Genealogy program. Joe has received feedback from all points of the globe regarding this program, and many of the most asked for features have been included in this release. Joe has decided to release the program into the public domain, rather than a fairware release, thereby foregoing heaps of $$$'s in fairware payments!!!

If you are interested in tracing your family history this is the program for you! Make sure you get along to the October meeting for a demo by the author.

SOCIAL ACTIVITIES

The Annual Childrens Christmas Party will be held at the Boolaroo Ambulance Station on Sunday 3rd December, commencing at 2.00pm. The afternoon will feature computer games, party games, bar-b-que, table tennis & a look over an ambulance! Make sure you keep this date in free & bring along the whole family. Further details will be announced as the time comes closer.

Don't forget the Adults Annual Christmas Dinner that was announced in last month's newsletter - it should be a great night out, so if you haven't already, get in touch with Bob MacClure NOW!!!

SOFTWARE LIBRARY

The recording of our software library to PR-BASE is almost complete, and we expect to be able to release full details of the library very soon. To all those members that have written to the Group asking for library details, please bear with us a little longer, it should be worth the wait.

WE NEED YOUR HELP

In order to provide the members with what they want, we need feedback. If the committee does not get any response regarding the likes and dislikes, the preferences, the needs and the comments of members it makes it extremely difficult (almost impossible, actually) to organize activities or articles to maintain your interest in computing in general & computing the TI way in particular. Please give us some feedback - either positive or negative, remember in the long run it is to your benefit that the Group is catering to YOUR needs.

NEW TI CHIP ANNOUNCED

It was reported in the Sun-Herald (Sydney) newspaper on August 27 that TI was still active in the microchip business. They reported...

"Texas Instruments has developed an integrated chip set that will allow manufacturers to build a complete 33-MHz 80386 based system with a total of only eight chips (in addition to the 386 itself)."

"Currently, most high speed 386 machines need 30 to 60 chips on their system board."

"The chip set will make it possible to build a complete 33-MHz 386 system on a 10cm by 13cm board, at about half the normal price, according to the TI spokesmen."

NEW BOOK

J Peter Hoddie reported in the August newsletter of the Boston Computer Society that "At the Washington DC show, Tony Lewis hopes to be able to premier his new technical manual for the TI. His plan is to create a single reference manual that TI developers can turn

to for hardware & software information pertaining to peripheral design for the TI... When the official announcement arrives, please consider supporting this venture."

ADVENTURERS PLEASE NOTE!

Gary Cox, in his column in the Mid South User Group newsletter reports that a new adventure is available.

"Asgard Software has released a new adventure for the Adventure Module called Zoom Floom. It is described as being set in a water-park. The object of the game is to have as much fun as you can within your budget in this 'amusing new type of adventure game'. The game is available on either disk or cassette for $7.95 plus postage and requires the Adventure Module."

Contact Asgard Software for further info or to order.

PAGE PRO 99

From a review of this program in the August issue of the Mid South User Group Newsletter, written by Michael Dorman.

"Page Pro 99 is an assembly program from Asgard Software that makes pages. You can intermingle text with up to 28 different pictures on a page of 66 lines and 60 columns. With Page Pro 99 you can use 2 fonts at one time - a large font (16 x 24 pixels) and a small font (8 x 12 pixels)."

"You can also draw lines easily. Change all three fonts by loading new ones from disk - 1. small font, 2. large font, 3. line drawing font."

"Page Pro 99 is NOT a desktop publisher. Page Pro 99 (nor any other 4A program) does not have the ability to use limitless font combinations. It is also NOT a drawing program. It uses fonts and pictures to create a good looking graphics page but it does not create the fonts and pictures."

The program is available from Asgard Software at the price of $24.95 plus postage.

SOFTWARE LIBRARY ADDITIONS

We recently received from the Brisbane Users Group a disk of utilities written by Col & Gary Christensen. There are Hard Disk Utility programs, a Lotto/Pools selector and of special interest, a program that allows you to catalogue a disk, mark them, then copy them to tape, one after another, without further interference from you. A great idea that would have helped me no end when I upgraded to disk drives a few years ago.

Great work chaps, keep up the good work. It goes to show that not everyone has moved to clones, and that there are still some TI programmers out there.

TIPS & TRICKS

To conclude my column this month, I reprint some shortcuts printed in the September issue of the Spirit of 99ers newsletter & compiled by Charles Dament.

1) Want a free ride to level 3 of Parsec? Try this:
a) Crash 1 ship before firing.
b) Work up to Bynites.
c) Crash after each Bynite.
d) Press Redo before the game ends.
e) Crash 1 ship before firing.
f) After Swoopers come Killer Satellites.
g) Now you are at level 3.

2) FCTN 4 too far to stretch your fingers? Try this - hold down Fctn J & press the spacebar.

3) Bell Program

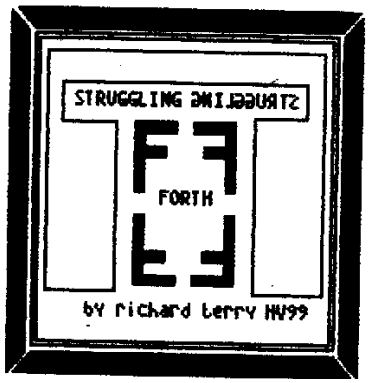
```
100 FOR X=1 TO 4
110 FOR C=0 TO 7
120 CALL SOUND(-500,6000,C,4000,C,
    2000,C)
130 NEXT C :: NEXT X :: END
```

Well folks, that's it for this month. Don't forget to offer your help, suggestions or articles so that we can help everyone get more from their TI.

All men need something to poetize and idealize their life a little-something which they value for more than its use, and which is a symbol of their emancipation from the mere materialism and drudgery of daily life.

Theodore Parker.

```
SCR #130
 0  ----------------------------------------------------------------
 1  -                                                               -
 2  -                                                               -
 3  -                                                               -
 4  -   STRUGGLING FORTH                                           -
 5  -                                                               -
 6  -   HV99ERS  SEPTEMBER/89 ARTICLE                              -
 7  -                                                               -
 8  -   FURTHER DETAILED PLANNING                                  -
 9  -   OF YOUR FORTH PROGRAM                                       -
10  -                                                               -
11  -                                                               -
12  -                                                               -
13  -                                                               -
14  -                                                               -
15  ----------------------------------------------------------------
```



As I mentioned last month, I find it extremely useful to sit down and sketch freehand with a pencil my anticipated screen designs, as a way of coming to terms with the concept of the program, and of getting a rough idea of the flow of the program. One can then construct a macro flow diagram after we have made a deeper analysis of the sort of routines you will need in your program.

There are situations where it is much easier to start your program writing with the screen design, and others where it is much easier to write your program specific code, even before general routines, and others where it is best to write the relatively program specific code prior to the most specific code. After that mouthful I feel I should audition for "Yes Prime Minister", as it sounds like something Sir Humphry would have said if he had brains enough to own a Ti994a! Anyway, you will get the feel of what is the best as you start to write several programs.

Before designing your screen and window designs, we have to decide in more detail what we want the program to do - an extension of what we started to talk about in the last article. As I promised to write a program as we go along, we might as well continue with our projected screen editor design.

Lets first look in more detail at what capabilities we would like in an ideal design editor. We can design our program from the top down either mentally or on paper as described above, visualising the steps that will occur when it is being used.

THE MAIN MENU

a)Start of program to consist of a main titlescreen, with bar-menus allowing the user to run up and down and select each option by hitting enter.

b)Main menu options will include Editor, Documentation to guide users, System configuration, Rebooting of any forth disk and Quit.

THE EDITOR

Options before entering editor to Edit, Load, Save, Erase, View, and copy screens.

a)Edit option to deposit user in edit mode - within which there will be identical function key usage and treatment of screens as does the forth editor with which we write out source code. However in addition to allowing text input it must allow (using special function or control keys) the user to draw graphic displays. Additional features within edit mode to allow user to do

a screen dump of what they are drawing and perhaps localise the editor to part of screen to operate on smaller areas. Include "help" windows.

b)The load, save, view options to use the inbuilt forth screen handling mechanisms. Ensure that on exiting the editor everything is "flushed", however allow an "oops" key to empty-buffers to prevent overwriting something if you've made a mistake.

DOCUMENTATION.

The extent of this section will depend on who is using the program, yourself alone, or others if you intend to unleash it on the public at large. It will consist of a number of displays which will give the un-initiated user an overview of the program, its capabilities and its key-usage, and include a comment re fairware and give authors name and address etc. Include capabilites of either sending to printer or viewing alone +/- with individual VDU screen dump. ?should I make this as DV80 file or as forth screens? Use either F4/6 or E.X to allow paging up/down, and allow escape at any time with F9/F5.

CONFIGURE.

Here I would jot down a few ideas about this. Do I want the program to be for my use alone, or do I intend it definitely to be released as fairware or to the public domain. If the former then I can be more rigid on the codings ie I can "hardcode" my system config because even if I change my defaults I'll have the codings to recompile at a later date. If the latter I will have to include the ability to allow the user to update the defaults. Note these will include things like no.drive/screens, printer names; also the ascii values assigned to graphics characters sometimes vary from printer to printer, as sometimes to control codes. I may want to save this configuration back to disk. Do I want to cater for hard-disk users and ram disk users?

At this point in your planning you are usually asking yourself "do I really want to go to all this trouble? I really should just keep these great programs to myself". However your dedication to your club and your altruistic nature wins out!!

RE-BOOT

I like to include this in most of my programs to allow the COLDing of any version compatible forth disk.

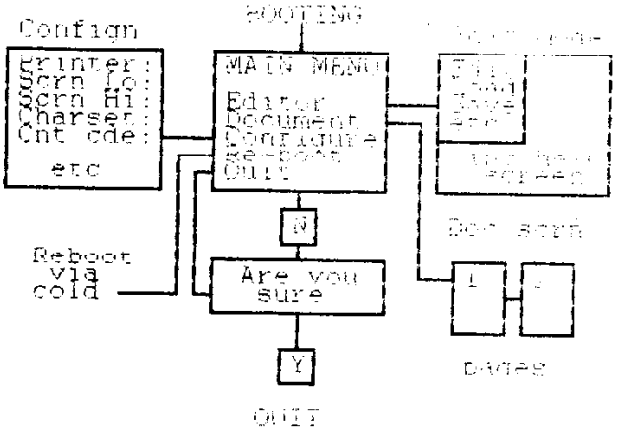
QUIT

This interposes between the user a "Are you sure Y" message and wiping out the program and having to reboot.

OTHER

Included in here will be thoughts about all the general routines my program might need.

At this point its worth taking a breather and sitting down with your pencil and paper and writing a series of "MACRO" flow diagrams visualising your potential program flow. Afterwards we may have:



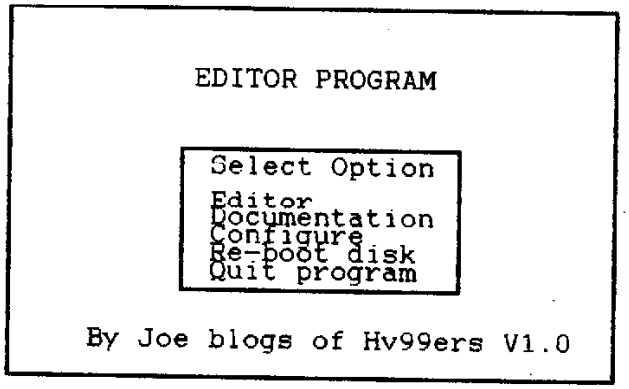
We now have a fairly good overview of not only what we want the program to do, but general impression of what it is going to look like in operation.

Using our objectives above in conjunction with our macro-flow diagram we can now further break down our routines to see what code we will need to write. At this point we will tentatively name a few of these routines according to a rough description of what they will do, without filling in any of the code.

Starting from the top down:

THOSE PERTAINING TO MAIN-MENU.

We will need a routine to display a whole vdu worth of data to flash up in front of us - ie a 40 column by 24 line screen. It has to get its 960 bytes from somewhere, initially we will pull them in from a forth screen. and they will consist of the visual displays we will later create with our editor program. Lets just call the routine ->VDU (put onto VDU display screen). As the program starts our intended user will now be sitting looking at something like this:

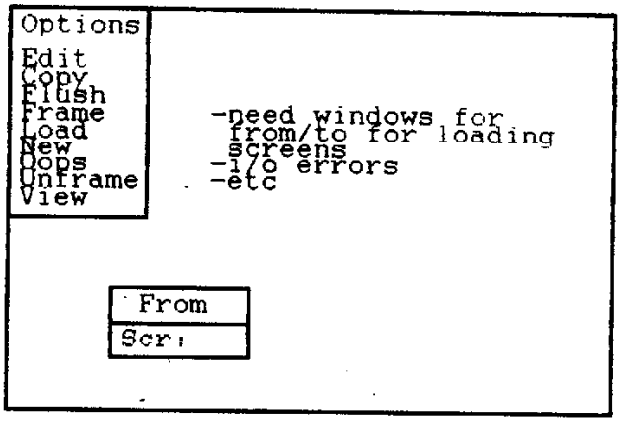


The ist option of our bar menu will be highlighted by being inverted. We can deduce our next routines will have to be to do with achieving this, and moving the bar up and down the menu until the user presses enter to run the option. Thinking ahead we realise we will probably have more than one bar-type menu as there will at least be another in the actual Edit option itself. Lets call this general routine BAR-MENU. It will have to do alot: start the first line inverted, sense users key input, allow the bar to move up and down, be able to be used with different menu's with varying numbers and width of lines, at differing spots on the monitor display according to which menu is being used, and finally allow some way of running the option the user chooses corresponding to the menu line it is on.

Sounds tough!!!!!!!!!!!!

Don't panic, because its actually very easy and logical if you think it through. Next month, I'll use it as an example in the development of general code.

Now lets assume we have chosen the Editor option and are confronted with the next display:



We already have our BAR-MENU to run this so what routines will be needed here?. As we project we are going to flash up a couple of windows overwriting the editor contents which we don't want to destroy. we will need to develop some window saving and writing code. Next, we will be in the editor. Something tells me that, being the guts of our program, this part isn't going to be as easy. Well, firstly we need the code of the editor itself. We can either be totally original or use the obvious - the codings for your existing forth editor with modifications. In fact I've already published my "anysize editor" in a previous HV99er magazine so we're half way home and hosed. Thinking about Loading and Saving etc. we'll need to develop some routines to accept the users input Eg GETNUMBER. along with associated routines to validate we are in fact only getting numbers and not letters etc: routines to do our screendump printout eg SCRN-DUMP and so on.

Similarly we can look through the Reboot and Quit and Configure routines and make annotations of the sort of things needed.

WRITING FROM THE TOP DOWN.

At this point we can make a few general notes re the code, starting from the top. As we write we

describe. in English. exactly what we want the program to do. not worrying at this point about how the routine will do it.

```
\ Theoretical source code

: DESIGN-EDITOR
     BEGIN
      TITLESCREEN
      X Y Z
      BAR-MENU
      MAIN-CHOICE
     UNTIL ;
```

Or alternatively use MYSELF

```
: DESIGN-EDITOR
     TITLESCREEN
     X Y Z BAR-MENU
     MAIN-CHOICE
     MYSELF ;
```

We may later choose to change either the actual routine name, or the internal details, but the above tells the gist of what the whole program will do before we even have the code to do it. There are many methods to have program flow back and forth between options. Here I have used the examples with BEGIN...UNTIL and the MYSELF routine. One could also use BEGIN...AGAIN or a method of re-entry using absolute addresses saved during compilation which we will look at later. Note at this stage we do not as yet know how BAR-MENU etc will work or what stack orders it will need, so I have just annotated some dummy values, X,Y,Z to remind us the routine will need some parameters to tell it where to start at.

Lets now break down the above definition and start to write its components. We need a word to flash up the first screen display in front of the user - which say, will include our central menu. At this point we will assume it will be loaded from a disk screen. but later we will short circuit that when we learn to keep all our visual graphics in the VDP chip for instant recall. Lets say we keep the display we want on scrn# 40. The routine that will pull that into memory if you check your book is BLOCK. (As an aside the way I learnt the meanings of alot of forth words was that on the occasions I was faced with knowing what I wanted to do, but not what the forth word was to do it. I'd literally flip through the glossary of terms and find one which from its stated description, should be expected to do the job - and then try it out). Once in memory we want to write it to the screen with vmbw:

```
: TITLESCREEN 40 BLOCK
              0 960 VMBW ;
```

This will work fine. Thinking ahead now, we anticipate that in many programs we will be flashing up more than one screen display, so lets factor this one out to give a more general definition:

```
: ->VDU         \ expt scrn #
       BLOCK    \ leaves addr
       0        \ start vdp of vdu
       960      \ text mode=960
       VMBW     \ write them
       ;        \ leaves nothing
```

Easy isn't it!

Moving on, we will forget BAR-MENU as I'm going to use it as an example next month on developing a complex piece of code using flow charting. Now we come to MAIN-CHOICE. This word will control the flow of the whole program. Again, there are many ways to do this. but a simple beginners way is to use the CASE-ENDCASE statement:

```
\ Non tested routine to run prog

: MAIN-CHOICE  \ exps n=menu line
    CASE 1 OF MY-EDIT   0 ENDOF
         2 OF DOCUMENT  0 ENDOF
         3 OF CONFIGURE 0 ENDOF
         4 OF COLD        ENDOF
         5 OF MY-QUIT   1 ENDOF
    ENDCASE :
```

How incredibly simple the whole thing is! The 0 or 1 left after each option will remain on the stack to be used by the UNTIL part of our definition DESIGN-EDITOR

TESTING OUR NEW WORDS.

We now have the up-front part of our program written. but with no "support code" to run it. However such is the beauty of forth we can write a few "dummy routines" to patch up the code that doesn't exist. The dummy word can be a definition in name only eg:

: CONFIG ;

which would do nothing but exist for a later word which needed it, or actually have an execution behaviour. This is a very useful concept, and allows you to debug your code as you go and to keep track of your stack

KEEP TRACK OF YOUR STACK

I've previously written a whole article on this in a back issue. Suffice to say that as forth words expect and leave parameters on the stack, its vitally important for you the programmer to know whats left on there after a routine. Sometimes as I write I inadvertently miscalculate (something I know you never will) and leave "rubbish" on the stack so that what I intend to pass onto my next word is underneath the "rubbish", so the "rubbish" gets used instead and the next routine doesn't work. Using dummy words you can easily test the routines and see if they work, and find your stack-order errors in small units. Its far easier to debug each routine as you go, because once you have a whole program running it can be devilishly hard to find out whats leaving that "rubbish" on the stack.

Why don't you type in and compile screens 136 to 142, and execute it by typing RUN, to verify that our up-top word DESIGN-EDITOR really does behave itself. Obviously our application is going to visually and behaviourally much more sophisticated than this. However this simple exercise gives you the idea of writing the code. Note that screens 137 to 139 contain the code to produce a visual display without needing a specific definition, ie. rather than waste hundreds of bytes writing a routine : TITLESCREEN CLS 10 10 AT ." EDITOR PROGRAM" ------etc ; and so forth, we can write some code to construct the graphics, save them back to disk and load them later as described above. Once we complete our editor program we will entirely circumvent even this intermediate step. I can hear our assembler programmers groaning about the massive "waste of space" in keeping 4 sectors for just one screen display - disks are cheap and we have what they lack - total flexability and the ability to write programs quickly! Refer to screen 143 to see what your titlescreen looks like on a forth screen after it has been saved.
Next month we will move on to examine the development of some general support routines you can use in any program.

ADDRESS FOR CORRESPONDENCE

R.TERRY
141 DUDLEY RD
WHITEBRIDGE
NSW 2290

(049) 436861

Program:Spt/89 article Date:19Spt89

Screen # 136

```
\ Struggling Forth Spt/89

\ display a forth screen of bytes to the monitor
: ->VDU                         \ expects Scn# on stack
            BLOCK               \ adr with scrn data
            0                   \ 1st byte of screen address
            960                 \ 960 bytes per monitor disply
            VMBW                \ write them up for user
            ;                   \ leaves nothing on stack
\ a routine to display the "up-front" screen with menu
: TITLESCREEN                   \ expects nothing
            143                 \ put your screen number here
            ->VDU               \ and display the screen
            ;                   \ leaves nothing on stack
;S NOTE: you must substitute here the screen number with your
       display data
     Eg : TITLESCREEN  50 ->VDU ;
```

Screen # 137

```
\ Struggling Forth Spt/89

\ Pause until the space bar is pressed
: WAIT      BEGIN ?KEY 32 = UNTIL ;  \ expects nil,leaves nil
\ temporary "dummy words" to test the other routines
: MY-EDIT   CLS 10 10 AT ." In editor now" ;
: CONFIGURE CLS 10 10 AT ." In config now" ;
: DOCUMENT  CLS 10 10 AT ." In Document now" ;
: MY-QUIT   CLS 10 10 AT ." are you sure ?  " WAIT PAGE QUIT ;
\ make a frame for our temporary front-end screen
: FRAME     CLS
            0  0 40 42 HCHAR
            0 23 40 42 HCHAR
            0  1 22 42 VCHAR
            39 1 22 42 VCHAR ;
\ heading for making our temporary titlscreen
: HEADING   9 4 AT ." HV99ERS DESIGN EDITOR " ;
```

Screen # 138

```
\ Struggling Forth Sat/89

\ menu for making a temporary titlescreen
: OUR-MENU  14  6 AT ." 1.Editor"
            14 10 AT ." 2.Configure"
            14 12 AT ." 3.Document"
            14 14 AT ." 4.Re-boot"
            14 16 AT ." 5.Quit"
            14 20 AT ." Choice:" ;
\ allow choice after the menu
: CHOICE    21 20 AT              \ expects nil
            KEY DUP  EMIT         \ show user his input
            48 - ;                \ keep a copy on stack
\ construct a titlescreen
: MAKE-TSCRN   FRAME HEADING OUR-MENU ;
```

Screen # 139

```
\ Struggling Forth Sat/89

\ saves back to disk what you have just created
: SAVE-TSCRN
            MAKE-TSCRN          \ flash it to the screen
            143                 \ or your screen to save to
            BLOCK UPDATE        \ load, and mark updated
            0                   \ start byte of vdp screen
            SWAP                \ 0,address
            960                 \ 960 bytes in a screen
            VMBR                \ read screen to disk-buf
            FLUSH               \ flush back to disk
            ;

;S NOTE ensure you substitute the screen number you want to save
   your copy of our main-screen back to and make sure it is
   empty or you will lose what you have on that screen!!!!!
```

Screen # 140

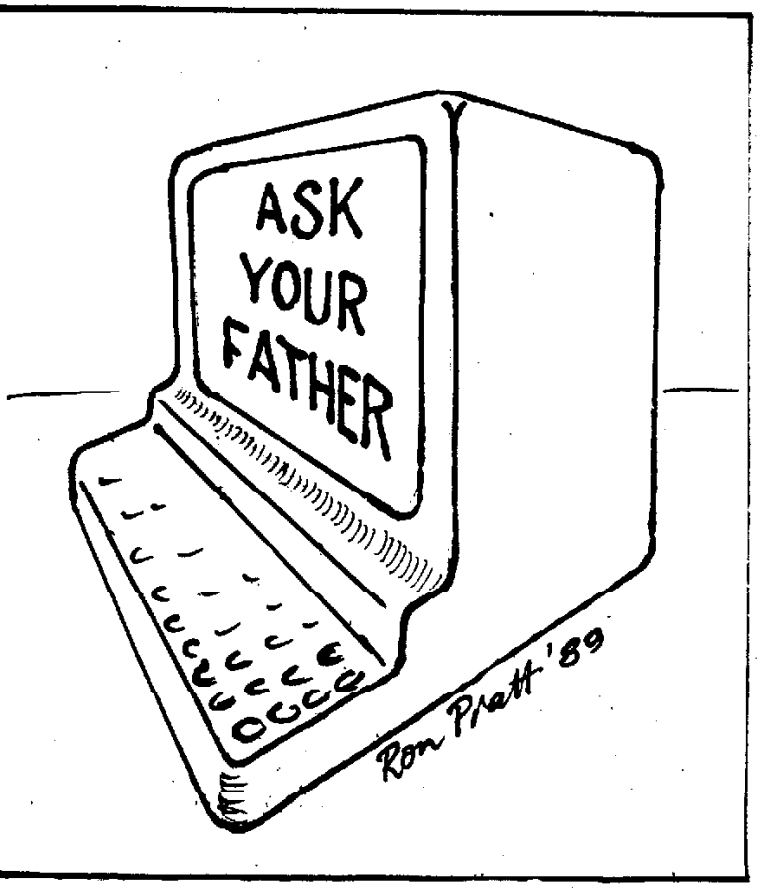
```
\ Struggling Forth Sat/89

\ a dummy cold!
: MY-COLD CLS 10 10 AT ." Re-booting!!! " ;
\ non tested routine to run the program
: MAIN-CHOICE                   \ expects n=line of menu on
        CHOICE                  \ user inputs number
     CASE 1 OF MY-EDIT   0 ENDOF \ go into editor mode
          2 OF CONFIGURE 0 ENDOF \ show user the documentation
          3 OF DOCUMENT  0 ENDOF \ configure the system
          4 OF MY-COLD   0 ENDOF \ do a "col" +/- message
          5 OF MY-QUIT   1 ENDOF \ "Are you sure Y"->quit
     ENDCASE
        ;                       \ leaves  nothing on stack
;S
  NOTE: The 0 or 1 after each option is left for the UNTIL
        of the routine  DESIGN-EDITOR.
```

Screen # 141

```
\ Struggling Forth Sat/89

\ Testing our new words
: DESIGN-EDITOR                 \ expects nil
     BEGIN
       TITLESCREEN              \ show main screen
       MAIN-CHOICE              \ choice for front end
       WAIT                     \ continue after space-bar
     UNTIL                      \ has been pressed
       ;                        \ leaves nil

: RUN DESIGN EDITOR ;

MAKE-TSCRN SAVE-TSCRN           \ construct and save tscreen
```


RON PRATT.

Every Group of people who get together to share a common interest has within it's ranks an amazing range of talents and abilities. The H.V.99'ers are no exception. Ron Pratt one of our well liked and dedicated members has a talent for drawing, (as well as wood turning and glass engraving). Many of our members have seen Ron's efforts at glass engraving when Tony McGovern was presented with his life membership. Ron lives in the small town of Dungog in the lower Hunter Valley. Dungog is one of the Towns established early in the history of European colonization of the Hunter Valley. He has sent me several cartoons, I have included one here, I will include the others over the following months.



Beating Around the BUSH

With Ron Klienschaffer

First and foremost let me pass on my personal thanks to the new committee. Why thanks?, well without the committee being there we wouldn't have a User Group would we, so I say again thanks.

El Presidente's report in the July issue about the "hot issue" has, for me anyway, bought a mixed reaction, one question that comes to mind is how would the group maintain the largest proportion of the groups activities related solely towards the TI?, if material for the newsletter, as in recent trends, become a little "slow" or scarce would we see an increasing number of articles being printed in the newsletter about activities from the special interest groups?, my being distanced from the main body of the group would only make such articles meaningless, because without any knowledge of the hardware or software being discussed I may as well go out and buy magazines from the newsagent and read those. If that is not the idea or intent behind the "special interest groups" ,which I am not in any way implying is, then I still feel that the groups constitution is still in force and that is roughly, "to further the knowledge and help to owners of the TI".

On the other hand there may be a real need for owners of some other machine to have some communication with others about those particular machines but still remain in close contact with TI users, as I can see it there are only three ways to do this and these are either, form a special interest group, join another machine oriented group, or both, and if it came to a crunch, for my money, I would rather have members that still have some interest in the TI than none at all!, what do YOU think??.

Here in Black hole county technology is catching up with the rest of the world, our local Social Club (don't laugh mate!), has recently purchased an IBM clone. This was necessary to handle the membership, and the bookwork on a soon to be expected licenced premises, real legal like!, this bloke couldn't help himself but to give the whole thing a work over, the machine in question is a Commodore IBM clone which performed better than expected, not too bad so far except for the part I want to get to and that is the multi package software supplied (sold) with the machine, the software package supports word processing, spread sheet, data base plus other operations such as communications etc, the software package named Able 1 has, without a doubt, as far as this reviewer is concerned, the worlds worst word processor package, if users of TI-Writer thought that it took some time to learn to drive it then just have a go at Able 1, just to delete a word of text requires a huge amount of keyboard work selecting the required function, outlining the required word from start to finish then more frantic keyboard work to delete the word, to top it off after doing this it required a great deal more work to get the rest of the text back into some order of sanity and mate that was an easy function. I don't know why software writers go out of their way to make a program as complicated as possible, maybe its a process to try to sell it against its other competitors by having a long list of "universal" operations to make for a convincing sales pitch!.

As a passing thought I personally just don't feel at home with DOS either, I don't like the way it takes a long time for it to tell me that there is no hard Disk (we didn't fit one) and that a printer is hooked to the system, something that I already know is there, I can SEE the B....Y thing!, (that should bring a reaction from the intending special interest people).

After my article in the July newsletter about the bug that existed in the Superspace II

cartridge, a couple of queries came in about similar problems with the QED 32K cartridge, the problem that existed with the Superspace II can NOT occur with the QED cartridge, good design took care of that, that is not to say that components in the QED Module could not fail, if anyone is having trouble with the battery going flat when the Module is left plugged into the console unused for a lengthy period then the most suspect component would be the diode that connects the +5 volt supply to the Module, this would allow the battery to drain down through the console system via the battery charging resistor, the next component to check would be the 1uF tantalum capacitor, it may be leaky, apart from that without having the Module to test diagnosis would be impossible. One thing to keep in mind is that the QED module uses a Nicad battery and these exhibit strange behaviour at times, they are well known to "go out to lunch" for seemingly no reason. As a matter of curiosity the Module draws so little current that the 1uF capacitor on the 5 volt rail to the 32K chip will keep most data "alive" for a least 60 seconds with the battery disconnected , the whole thing draws less than 1 micro/amp.

On the lighter side, some time back there appeared on a BBS an " advertisement ", after its appearance the " advertisement " made the rounds of various journals and you may or may not have read it but I thought that in the light of the search for a better understanding of what the future in computing peripherals may hold I reproduce it here, take it as you will.

"We at Bushlit Micro Supplies pride ourselves on doing things just that little bit bigger and better."
"Our Liquid Helium cooled laser printer can turn out ten A4 pages a minute at 10 meters. No need to fiddle around loading those cut sheets - simply line them up against the wall, push the button and stand back!"
"Each printer is supplied with a warning sticker for the door, a Klaxon which sounds when something goes wrong and an 'I was Tattooed by a runaway Laser Printer' badge to amaze your envious friends."
Ahh well!.

FORTH DISC HEADER from Brian Rutherford

For those of you going to Joe's and Richards FORTH classes, and have two double sided drives. Here are the changes needed to screen 40 to enable you to write a disk header for a double sided disk, to sector zero with the disk in drive two. DO NOT CHANGE SCREEN 40 ON YOUR ORIGONAL FORTH DISK, use one of your back up disks. Also I have included a word DISK-INIT to format a double sided disk, clear all the screens and put the disk header on it, with the disk in drive two. This is incase you have done the homework that Joe set for you, and find that you do not have a disk ready to FLUSH it to.

```
0 ( WRITE A BSSB ON DRV#2 HEAD COMPATABLE WITH THE DISK MANAGER )
1 BASE->R HEX
2 : DISK-HEAD B4 CLEAR  B4 BLOCK ( START SECTOR 0)
3   DUP !" FORTH     " DUP A + 200 SWAP !
4   DUP C + 944 SWAP ! DUP E + 5348 SWAP !
5   DUP 10 + 2020 SWAP ! DUP 12 + 201 SWAP ! DUP 14 + 24 0 FILL
6   DUP 38 + C8 FF FILL 100 + ( START SECTOR 1)
7   DUP 2 SWAP ! DUP 2+ FE 00 FILL
8   100 + ( START SECTOR 2)
9   DUP !" SCREENS  " DUP A + 5 SWAP !
10  DUP C + 2 SWAP ! DUP E + 2CB SWAP !
11  DUP 10 + 80 SWAP ! DUP 12 + 9A85 SWAP !
12  DUP 14 + 8 0 FILL DUP 1C + 03C0 SWAP !
13  DUP 1E + 2C00 SWAP ! 20 +  E0 0 FILL
14  FLUSH
15 ; R->BASE
```

```
0 ( A WORD TO FORMAT A DOUBLE SIDED DISK IN DRIVE 2 )
1
2 : DISK-INIT                          ( expect nothing on stack )
3      CLS 3 19 GOTOXY ." Put disk in drive 2 " ( prompts       )
4      3 21 GOTOXY ." Press space bar "         ( prompts       )
5      BEGIN ?KEY 32 = UNTIL                ( wait for key press )
6      2 3361a !                       ( tell forth double sided )
7      1 FORMAT-DISK                   ( format disk in drv# 2   )
8      3 23 GOTOXY ." clearing screen "
9      360 181 DO I DUP 19 23 GOTOXY .  ( set up loop           )
10     CLEAR FLUSH LOOP                ( to clear and flush scr )
11     4 184 2 SMOVE                   ( copy error msg to disk )
12     DISK-HEAD                       ( put the disk header on )
13     CLS 3 23 GOTOXY ." DONE " ;
14
15
```

BITS and Pieces
with Joe Wright

Wasn't going to write anything this month, the Newsletter is well and truly full. Left with a blank page; so a quick rummage through some back newsletters and the following programmes are the result. None the less I can not miss an opportunity to pass on a quote.

Nature is the most thrifty thing in the world; she never wastes anything; she undergoes change but there's no annihilation, the essence remains-matter is eternal.

BINNEY.

Makes me wonder why we humans insist on intervening, cut down forests, kill off whole species, and we call it progress???

MYSTERY PROGRAMME.
****
This first programme comes from the L.A. Topics newsletter. Chick De Martin found it in the OMAHA User Group Newsletter.

```
10 ! MYSTERY PROG.  #2 BY C.  Schram
20 ! Requires XB and 32K
30 CALL CLEAR :: CALL SCREEN(1)
40 CALL INIT
50 FOR X=1 TO 28
60 RANDOMIZE
70 CALL PEEK(-31808,A,B)
80                                  CALL
SPRITE(#X,46,16,A+1,B+1,A-128)    ::
CALL  PEEK(-31877,C)  :: IF C AND 32
THEN   CALL   SCREEN(10)   ::   CALL
SCREEN(1)
90  CALL  LOAD(-31744,A,"",-31744,B)
:: NEXT X
100 GOTO 50
110 END
```

Chick mentions that the programme is fascinating to watch but you have to FCTN 4 to stop. She made the following changes.

```
35 FOR TIME=1 TO 5
```

and changed

```
100 NEXT TIME
```

USING SEG$
****
This is from Jim Swedlow.

A number of the XB columns discussed alternatives to IF/THEN. Here is another. Suppose that A$ depends on the value of I. You might use;

```
IF I=1 THEN A$="FRED" ELSE A$="PAUL"
```

A simpler way is to use the SEG$ function.

```
A$=SEG$("PAULFRED",1-4*(I=1),4)
```

Will this work if the two variables have different lengths?? Yes! Remember that SEG$ does not produce and error if the length of the string (the last number) is longer than the source string. Try "PAUL" and "SAM".

```
A$=SEG$("PAULSAM",1-4*(I=1),4)
```

IF THEN ELSE AGAIN
Another thing I found in the "Teach yourself XB tutorial is how XB matches ELSEs with IFS.
Each ELSE is paired with the last unmatched IF. For example;

```
IF A THEN B :: IF C THEN  D  ELSE  E
ELSE F
```

In words:
If A is true, do B and then test C. If C is true do D. If C is false do E. If A is false do F.

FIN
***
See ya next month friends.

Joe

PRETENSION

Those who quit their proper character to assume what does not belong to them are, for the greater part, ignorant of both the character they leave and of the character they asume.

BURKE

random bytes

with

BOB CARMANY

I've been corresponding with Tony McGovern for a couple of years now --ever since I got my first copy of FUNLWRITER Vn 3.1. Throughout that time span, I've had to put up with the judiciously spaced "barbs" about by antiquated system. I guess in the long run Tony was justified in his remarks about my SSSD standalone system. Now, however, I have the last laugh! I finally upgraded --- a full PE-Box and a pair of Panasonic DSSD drives with an outlay of $150 after adding in what I got for the surplus standalone units that I sold. Ah, the luxury of it all! I suppose that Tony will start agitating for me to get an 80-column card and Quest 200. Not yet, Tony!

First of all a bit of a correction to the first installment of the PROGRAMS series. Of course, a quoted string should be enclosed in QUOTATION MARKS rather than parentheses. A ridiculous error but it crept into the text in spite of my myopic proofreading. I hope it didn't cause too much confusion.

Everyone seems to be concerned with what's new for the TI. There is some quality new software available but there is some equally old quality software around as well. Most of it is stuff that was marketed commercially that everyone has forgotten about in the mad rush to add new programs to their library. With this in mind, let's take a look at some "oldies but goodies".

One of the most interesting packages is the AMERISOFT extension of XB. Like most the the XB extensions, it uses a series of A/L routines that can be LINKed from the XB environment. Unlike most of them, it includes a large number of graphics commands. In fact, almost the entire lot of the European APESOFT graphics are present. You can construct windows, draw circles or ellipses, and other shapes from XB just by entering the necessary parameters. There are screen dumps and other utilities to go along with the package as well. It is well worth a second look!

Another software package worth a second look is the COMPILER from Ryte Data. It has some limitations but it can be of significant value for some of your XB programs. It will increase execution speed and it is very easy to use.

Before I end this column for this month, let's take a look at some values that can be used in CALL LOADS and A/L programs.

-31970	>831E	Step in NUM mode
-31888	>8370	Pointer to the end of VDP RAM --start of disk buffers
-31878	>837A	# of sprites in motion.
-31884	>8374	Keyboard # to scan (0 - 5)
-31882	>8376	Joystick Y from scan
-31881	>8377	Joystick X from scan
-31880	>8378	Random number generator.
-31808	>83C0	Random number seed

You can POKE and PEEK with these values and see what you can come up with.

One more thing worth mentioning before I get a "buffer full" message. Remember that all of those disk drive cleaning disks are abrasive. As such, they will wear down your disk heads. With this in mind, only use them as a LAST RESORT. In one of his newsletters Craig Miller said that he used his TI 7 hours a day 6 days a week and never found the need to use a disk driver cleaner more than once a year. Unless you use your drives more than that, you may never have to use one.

Once again it is getting close to the end of a column. There always seem to be "honey-do's" waiting --- you know, "honey do this"and "honey do that". Those of you who are married can relate to it well. Any suggestions for future topics to be covered in this column would be greatly appreciated. Send them on to Brian and he will forward them to me. 'Til next month . . .

RANDOM BYTES

I thought that I would include some programs in this column that you had probably overlooked. The first was inspired by Tony's column in the May HV99'er newsletter. He was wondering if <FCTN-V> returned the same value. The following program will return the keycode for any key or group of keys when pressed simultaneously.

```
100 CALL CLEAR :: CALL SCREEN(13)::
CALL INIT :: CALL LOAD(-31806,16)::
ON BREAK NEXT :: FOR D=0 TO 12 ::
CALL COLOR(D,16,13):: NEXT D

110 CALL HCHAR(24,1,126,64)::
DISPLAY AT(3,2):""""KEY-CODES"" BY
RAY KAZMER";;:" SAN FERNANDO VALLEY
99'ERS" :: T$="?" :: GOTO 170

120 ACCEPT AT(11,16)SIZE(-1)VALIDAT
E ("012345") :T$ :: IF T$="?" THEN
CALL SOUND(175,220,0):: GOTO 120
ELSE L=VAL(T$)

130 DISPLAY AT(9,1):"" :: DISPLAY
AT(13,1)BEEP:"PRESS ANY KEY OR
COMBINATION" :: FOR D=1 TO 100 ::
NEXT D

140 CALL KEY(L,K,S):: IF S=0 THEN
140 ELSE DISPLAY AT(13,1)BEEP:""
;;;;:TAB(12);" K = "&STR$(K)

150 FOR D=1 TO 400 :: NEXT D ::
DISPLAY AT(22,1)BEEP:"PRESS ANY KEY
TO REPEAT TEST"

160 CALL KEY(0,K,S):: IF S=0 THEN
160 ELSE DISPLAY AT(16,1):"" ::
DISPLAY AT(22, 1):""

170 DISPLAY AT(9,1)BEEP:"SELECT
CALL KEY TYPE # (0-5)":;:" CALL
KEY("&T$&", K,S)":;:" AND PRESS
ENTER" :: GOTO 120
```

This next program came from the 9T9 newsletter in Canada. It creates a series of sounds based on truncating the word "instructions". Just follow the instructions for a completely new series of sounds. You will need a Speech Synthesizer to run the program.

```
100 ! *** WEIRD SOUNDS ***

110 ! BY DAVID HUGGETT

120 !

130 ! 9T9ER USERS GROUP, TORONTO

140 ! The word INSTRUCTIONS in
line 160 may be changed to any word
in the resident vocabulary for
different effects

150 FOR Y=1 TO 88 :: IF Y=4 THEN
Y=7

160 CALL SPGET("INSTRUCTIONS",D$)

170 D=LEN(D$):: PRINT Y

180 D$=SEG$(D$,1,2)&CHR$(D)&SEG$
(D$,Y,D)

190 FOR X=1 TO 6 :: CALL SAY(,D$)::
NEXT X :: NEXT Y
```

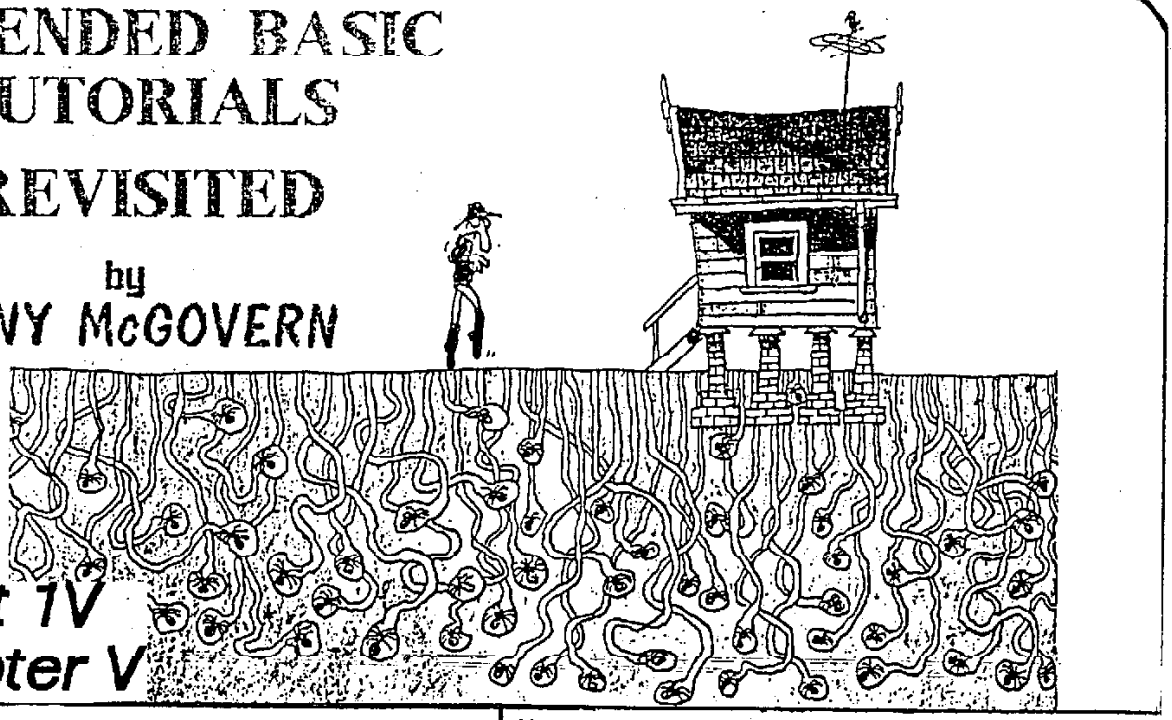
Oh yes, before you question whether line 190 is correct, let me assure you that it is. The CALL SAY(,D$) statement is the correct syntax in this case. It looks wierd but that is the title of the program, after all.

After a year of using the QED 32K cartridge, I have come to some definite conclusions. One thing that it has done is to postpone my need for a RAMdisk. Right now, the only real use that I can see for a RAMdisk is to load part of my F'WEB system into it and leave 360 sectors empty for workspace --- downloading programs from a BBS, etc. I have never been that keen on sheer, raw speed. I suppose that it would be nice to have instantaneous loading of some programs but I have never felt a loss at waiting a few extra seconds for a program to load from a physical disk drive. I do find that the convenience of having the QED to be almost indispensable, though. It makes switching between the various programs in my customized F'WEB easy and much faster than changing disks all of the time. Don't get me wrong, I would certainly latch on to a Horizon at the right price but I can live without one thanks to the QED. Thanks mates for sending it along!

I thought that I would end this column with a bit of controversy. Brian insists that he has discovered the best brew in the world --Toohey's Draught. I must take issue with that pronouncement (being a Foster's man myself). At any rate, don't forget the "inner man" --- drink a cold one for me, mate! 'Til next month . .

EXTENDED BASIC TUTORIALS REVISITED

by TONY McGOVERN



Part IV Chapter V

V. XB STYLE WITH SUBPROGRAMS

Let's now stand back a bit and look at the best way to construct XB edifices. Assume at this stage that we are in the process of developing a program, but not yet to the point where scrunching program length has become important. The first thing to note is that by giving the subprograms good descriptive names you have already gone a long way to making your program self-explanatory.

How big should individual subprograms be allowed to get ? After all, one of the reasons for using them is to break up big programs into manageable hunks. We will use the term 'line' to refer to a multi-statement XB line identified by a line number. My own prejudice is that, except in special circumstances, subprograms should be no more than about 10 lines long, and mostly rather less than that. What makes an exceptional circumstance? An obvious one is in title blocks, like that in SIMPLIST which was left as an almost bare stub. A full version would provide graphics and advice screens, which can be tediously long to write, but contain very little in the way of branching decisions or variable assignments. Another example is where a familiar routine, that already works, is used with little variation as in COLIST where the disk directory routine from the Disk Manual is incorporated as a subprogram with only minor changes. In any such situation where long subprograms are justified, the lists of parameters passed will be short or non-existent.

The other extreme is short one or two liners which are frequently CALLed for small special tasks, more or less your own customized extension of the built-in set of subprograms. In the middle there are middle-length subprograms with extensive parameter lists and the logic which carries the burden of program flow.

Some subprograms may be CALLed only once from within another subprogram but are of value in making your code easier to read and modify. These are associated with the branching of program flow by means of IF..THEN..ELSE statements. In either TI BASIC or XB, FOR-NEXT loops may extend indefinitely with NEXT acting as delimiter. Unfortunately in extending BASIC to XB, TI did not provide an "ENDIF" statement as in TI-FORTH, but only the 'endif' implied by the end of a XB line. This means that any alternative actions determined by the IF.. condition have to fit within that XB line or involve a GOTO somewhere else unless the usual simple drop-through to the next line is enough. The XB manual already explicitly forbids inclusion of FOR..NEXT loops within

IF..THEN..ELSE statements. No doubt you are already used to getting around this little deficiency by placing the looping code in a subroutine and using a GOSUB. Subprograms can be used instead, following THEN and ELSE to give more complex alternative possibilities, but still staying within the confines of a single line with a minimum of leaping about with GOTOs.

This brings us to the subject of the 'dreaded GOTO'. A great deal of heat, and not necessarily much light, has been expended on this subject. It is after all just another statement available in many languages, and has perfectly predictable immediate consequences. The real objection is that it leaves no trace as to where the program "came from". At the machine code level, jumps enable the computer to do more than just chomp along a single track of instructions. The question is whether it is help or hindrance in high level languages, and whether other ways of controlling program flow can replace its explicit use to advantage. TI-FORTH does without it, but that most procedural of languages, TI-LOGO, still finds it useful. Pascal tries to do without it. What we do have is XB, and XB can't do without GOTOs. If anything should be considered as reprehensible in a high-level language, it is any need to provide PEEK and POKE.

The great weakness of GOTO as a language element is that it is so readily abused, because undisciplined use makes the program code inefficient and hard for people to follow. The genuine message from 'structured programming' ideas is not that BASIC is bad for having GOTOs, but that most BASICs (TI console Basic is typical) make it necessary for the programmer to exercise real restraint if terrible tangles of GOTOs are to be avoided.

Once you use XB subprograms to chop up a program into small hunks, then you have automatically eliminated great leaps around with GOTOs. All you need then is to remember the comments on using subprogram CALLs as statements in IF..THEN..ELSE and take a little care in laying out the logic flow, you will find it very much easier to debug or develop programs. Backwards GOTOs over more than one or two lines of code, or any forward GOTOs at all, should only occur under the most regular of logical layouts, as in SUB BASICLINE in the SIMPLIST example. Single recursive lines such as in line 620 of SIMPLIST are very effective. It's a pity that the designers of XB didn't add the "MYSELF" function as in TI-FORTH to enhance such constructions.

One last little matter before we go on to other topics. Many languages with local procedures also allow specification of global variables, accessible from any part of the program. XB does not allow for separate global variables, and it can be quite tiresome when a parameter defined at the end of one subprogram chain is only needed at the end of another chain, and has to be passed all the way up and down in parameter lists. A way around this is to use the static value feature of XB subprograms.

```
3000 SUB PAGELENGTH(A,B):: I
F A THEN C=B ELSE B=C
3010 SUBEND
```

If the write flag is set as CALL PAGELENGTH(1,66) the value 66 is stored in the subprogram local variable C, while CALL PAGELENGTH(0,PL) will retrieve that value into PL. This is clumsier than having global variables, but is also more protected from unwanted interference. XB does not enforce any hierarchy of subprogram levels, so PAGELENGTH can be written to, or read from, at any level in the program. The example is for one parameter only, but is easily extended.

Using subprograms does carry some overhead expenses, both in the size of the program and in the time taken for XB to do a CALL as distinct from a simpler GOSUB. Unless you are absolutely desperate for bytes, the benefits of subprograms outweigh that price and they should always be used liberally in the early stages of program development. The speed argument is a mixed one, as in a long program the extra time cost of a subprogram CALL can be more than outweighed by the savings in time for the interpreter because it has only a short local list of variable names to search instead of a much larger global list.

VI. PRE-SCAN SWITCH COMMANDS

The little supplementary booklet that comes with the current Version 110 of Extended Basic introduces a new pair of reserved words, !P+ and !P-. These have the form of a tail remark (XB manual p38) and so are ignored entirely by the earlier V.100 of XB. If the XB interpreter finds an exclamation mark ! outside any DATA string or string enclosed by quotes, it treats the rest of that line as though it were a REM statement. The V.110 interpreter has the added ability to recognize this pair of words beginning with ! as being distinct from normal tail remarks when used as a single word statement. Their use is allowed only at the end of a line so that V.100 just ignores them, not creating any incompatibility problems between versions, something that TI was always conscientious about. TI then couldn't let these commands actually do anything! So why are they there ?

The XB manual addendum, p7, tells the story. These switch commands allow you to control the operation of the pre-scan through the program by the interpreter -- that agonizing time interval after RUN is entered before the program starts executing. The interpreter is grinding its way through your program, byte by byte, ignoring only the messages in DATA, REMs and tail remarks. Other than these there is nothing that it can afford to ignore until it has actually looked at it. The pre-scan sets up the storage areas and lookup procedures for variables, arrays, data, sub-programs and DEFs used by the interpreter as the program runs. Of course once it has set aside space for a variable and its lookup linkages, then it doesn't need to do it again or even to have to decide it has already fixed it up earlier. The pre-scan switch commands allow the programmer, from a superior vantage point, to turn the pre-scan off and on throughout the program so that it only looks at what it really needs to look at to do its job.

What does the programmer gain by going to all this extra trouble? The most obvious result is a reduction of pre-scan time. This can be significant in long programs. The 6 to 7 seconds for TXB, a 12K program, may still seem long but beats 4 times that. In a later chapter we will see how it can be used to fine tune run time behaviour as well. What price does the programmer pay for these benefits? The necessary penalty is the memory space taken by the extra statements. The hidden penalties, incurred while writing programs, are the inscrutable bugs that may be introduced into the code and the loss of some program checking during pre-scan such as FOR-NEXT nesting.

Let's work our way through the XB manual's prescriptions. Some of these help give insight into the way XB conducts its affairs. My experience is that some of the restrictions need not be followed strictly as laid down, as long as the essential spirit is observed, while some are absolute, and others are in between. These last are the ones where it is possible to imagine another version of XB doing things differently while still being according to the book. This is always the danger in using unspecified properties or "undocumented features". It is not such a problem with XB since TI pulled the plug on the 99/4a and made XB a language as dead as Latin. In retrospect this last statement is no longer quite true, and though people continue to use the original XB there have been enhanced alternatives, still GPL based using various GROM simulation schemes. There have also been utility programs to do automatically some of tasks we are discussing here and more. What has been sorely lacking has been any published exegesis of just how XB goes about its business internally. I have never seen any source code for XB, either original TI or reconstructed. I don't know whether TI source has leaked out, or even if it still exists, but if it has it is being closely held, and the writers of XB processing utilities do not seem to have felt any urge to share around the details of what they found.

(1) DATA statements :-

The pre-scan locates the first DATA statement and sets XB's data pointer for the first READ operation to use. If the first DATA is skipped in the pre-scan, then RESTORE must be invoked before the

first READ to set the data pointer correctly. If this is done, the XB manual's advice can be ignored.

(2) Variables :-

Each variable must be scanned once, otherwise XB won't have it in its linked list of pointers to names and storage locations. This can be the source of some truly evil program bugs, where a syntax error message results from a line of code which looks perfectly correct. The reason can be that injudicious positioning of pre-scan switch commands has left the interpreter with something that should be a variable, but can't be located as such. Being a non-variable is a much worse fate than merely being set to zero.

OPTION BASE 1 affects how storage is allocated and normally precedes any array references. If hidden from the pre-scan by!@P-then the default Ø will apply.

The manual says that the first occurrence of any variable or array must be included in the pre-scan. This would seem to be necessary for arrays, in the DIM statement, unless you are using the default (no DIM) dimensioning. Simple variables can be pre-scanned anywhere as long as it's at least once. Try the little sample program

```
100 CALL CLEAR :: !@P-
200 I=1 :: PRINT I
300  !@P+
400 I=2
```

Run this program and there will be no problems. Delete line 400 and see what happens. Now you will have a syntax error in a line that by itself is perfectly correct.

(3) Sub-programs :-

The XB manual recommends that the first CALL to any sub-program be included in the pre-scan. It would appear that if the first CALL to a user defined sub-program occurs after its own SUB (from within a later sub-program) then the necessary inclusion of the SUB and SUBEND markers suffices.

Built-in sub-programs of course do not have associated SUB statements, so a CALL must be included in the pre-scan if the program is to run normally. Try this example.

```
100 FOR I=1 TO 1000 ::  !@P-
200 CALL SCREEN(12)
300  !@P+
400 NEXT I
500 SUB ANYTHING :: CALL SC
REEN(3):: SUBEND
```

This will run even though SCREEN is pre-scanned only in a subprogram. Delete line #500 and it will crash if you are running XB with the 32K memory expansion. In VDP RAM (console only) it still executes but only at about 1/3 the speed.

What happens if an array is referenced in the parameter list of a sub-program, but not dimensioned until a later sub-program? If you recall the discussion on passing arrays by reference, you won't be surprised to find that XB is smart enough to hold over assigning space for the array until it comes across a genuine program reference. Try this little example

```
100 CALL SECOND
200 SUB FIRST(A()):: PRINT
A(20):: SUBEND
300 SUB SECOND ::  !@P-
400 DIM A(20):: CALL FIRST(
A())
500  !@P+
600 SUBEND
```

This program crashes with a syntax error in 400 in SECOND. Now delete the pre-scan commands and the program will run. If you further delete DIM A(20):: in line 400 the program will crash in 200 with a subscript error.

(4) DEF,SUB and SUBEND :-

Do as the book says. XB needs these in the pre-scan to set things up correctly.

The pre-scan switch doesn't have much effect unless the program is of substantial size, so it isn't worth worrying about too much in the early stages of a program's development beyond being prepared for the possibility. The XB manual supplement (p10) shows how all variable and sub-program declarations may be gathered together to minimize the range of the pre-scan, by using a GOTO to

jump over the list to the first executable statement. This can be gotten away with since XB does not do a complete check for correct syntax until it comes to execute the line. This is the only virtue one can ascribe to XB's failure to reject all invalid lines at entry time. The same technique can be used within a sub-program, and I have found it very convenient for this same GOTO to reserve a hiding place in which to tuck away the subroutines accessed by GOSUBs within the sub-program.

POWER CABLE WARNING

The SEC (Victoria) has warned computer users against unapproved imported power leads in their equipment. Ian Coleman, Victoria's chief electrical inspector, said the use of unapproved power leads could create a very dangerous situation when computers were switched on.

"We are aware of two types of power leads that have been imported and distributed in Australia in which the earth conductor was connected to the live or active pin of the three pin plug," he said. "This creates a very dangerous situation"

"Anyone who has either of the two types of leads should stop using it immediately and return it to the store from which it was purchased.
Coleman said the two types of power leads could be identified by the following markings:
1.The 3 pin plug marked "Stabile", SP-2
Flexible cord marked-"Ta Hsing, 344002
and
Appliance coupler marked-"Stabile", SC-1

2.The 3 pin plug black and unmarked and
Flexible cord marked-"Yeh Yang", 0444016
and
Appliance coupler black and unmarked.

If anyone should be in doubt about the safety of their power leads they should call (03) 6914470.

AL'S MARKET PLACE UPDATE

WITH *Alan Franks*

Well! I wonder how many were incapable of passing up the Mitsubishi drives "at the price they were". Not many from what I can gather. They are still available but the price has increased to ninety dollars a drive or one hundred and thirty dollars a pair; which is still a good buy.The number to ring is (049)540317 and ask to speak to Col.

I am afraid no similar bargains have come to light this month. There are still plenty of second hand specials available. Due to work and other comitments I was unable to check what was actually sold out of last months list. So if you are after anything, its only a phone call to find out if its still for sale.

I received a letter through the week from Jerome Munchenberg who asked me to advertise the following equipment for him.

1983 black and silver console, expansion box with 32k card,RS232 card,Disk Controller card,two Double Sided Teak slimline drives, ex basic,editor/assem,parsec, alpiner,tunnels of doom,video chess,disk manager ii plus fifty assorted disks of programs,disk box,manuals and newsletters and a customised table designed for the TI. The asking price for the lot is $990 ono and you can contact Jerome at 39 Morphett Road Camden Park, 5038. South Australia. or Phone (08)294-1791.

LOGIC

Men are apt to mistake the strength of their feeling for the strength of their argument. The heated mind resents the chill touch and relentless scrutiny of logic.

GLADSTONE

FINDING OUT ABOUT THE FOUNDING FOUR A 2

by a Fond Foster Parent

The story so far: If you were lucky enough to miss episode 1 - it began a review (with the benefit of hindsight) of a novice 4A owner's first encounter with his computer, aided? by Beginners Basic (BB) and the Users Reference Guide (URG). Page 54 of BB was as far as stage 1 had reached uneventfully. The saga continues ---

On page 55 of BB we come across the fallacy about 4 screen lines being the maximum per program line in TI Basic. I'm sure that all 4A foster parents are aware that a program line in TI Basic can be extended to more or less 6 screen lines (let's call them rows to prevent confusion).

More or less? - I'll explain. For those who have not tried the exercise before, this is the general procedure. Rather than making up some meaningful message to use as text, for this trial let's just use a string of letters by holding down a key and letting 'er rip. Start off the program line

10PRINT"AAAA--- etc until the cursor beeps (didn't you turn the sound down?) and blinks at the end of the 4th line. Type in the closing " and ENTER.

Go to Edit mode (10 and FCTN E). You will notice that the 4A has inserted spaces before and after PRINT and, as a result, the last A" has already wrapped around to the 5th row. Cursor (FCTN D) down to the closing " and kick off with As again to the end of the 5th row. Type in the closing " again and ENTER. Once again edit and cursor to the " at the end of the 5th row and kick off with As until there are 25 on the 6th row. Type a final " and that's it.

That's the LESS version, ie with two unoccupied spaces at the end of the 6th row. For the MORE version - try it with line number 10000.

And for a couple more on the 7th row try DISPLAY (who uses DISPLAY?) in lieu of PRINT. Whichever - the program line will 'hold' no more than 155 As (or other text) plus the opening and closing quotes.

When I first encountered this anomaly, I fiddled (fumbled more likely) around to see if I could extend this maximum still further. I tried

```
10G$="AAAAA--- etc
20PRINT G$
```

only to find that, with the string approach, the maximum was only 153 As. I guess the two " are part of the full string of 155 characters.

Undeterred, I decided that I may manage an extra A or two by eliminating the ". so next it was

```
10DATA AAAA--- etc
20READ G$
30PRINT G$
```

So much for that 'bright' idea. Once again 155 was the maximum. And this brings us to the comment on page 55 of BB in reference to the 4 screen lines limit. BB states that the exception to this is the DATA statement, and you are advised to see the Basic Reference section of URG for an explanation. I have not found this 'explanation' - does anyone know? I have since noted in an article somewhere that there is a limit of 30 commas in a DATA statement. This does not seem to be mentioned in URG.

Unwilling to accept defeat (aka pig-headed) I did a bit more fumbling and found that I could get 11½ screen rows per program line. Showing outstanding investigative technique I decided that, instead of "AAAA I could just hold the " key down and let 'er rip - after all, one character is as good as another.

Not so! With quotation marks

10PRINT"""""""" etc will enable 312 " in the one program line. The line can be entered and will LIST back to the screen intact. Then comes the let down - when you RUN this line you end up with just 155 " on the screen. This seems to indicate that the first quote and the last quote enclose 310 or 155 pairs of quotation marks. I don't know which member of each pair is PRINTed. 308 is the limit with a string and 312 using a DATA statement.

All in all, apart from some novelty value (one line number taking up half the screen), it's a pointless exercise. If anyone feels inclined to check it out (what! - nothing better to do!) here's some cautionary advice.

Quotation marks apparently mate for life, so always ensure that you have an even number of them. If you start off with 10PRINT"""" etc, when you go to Edit mode you'll get 10 PRINT """" etc (19) on the first row and will thus be limited to 27 on the bottom row, and so you'll never reach the wrap around stage. So line number 100 is the best choice for this exercise.

Extending a TI Basic program line to 6 (approx.) screen rows is no more than a somewhat tedious exercise without any particular relevance (that this little black duck knows of). The same line extending method can be used with XB to overcome the 5 row 'limit', in which case it does have relevance due to the multi-statement facility of XB. But that's another story.

Time (not before type) to move along to page 57 and Plain and Fancy PRINTing. PRINT is worthy of a lenghty article in its own right. No doubt it's been done before, but there's always room for another one (hint! hint!). Many of the Basic programs I've seen do not make effective or efficient use of PRINT, print separators and TAB. How many blank lines have been PRINTed?

The inexperienced novice is introduced to a typical example (of how not to) on page 64 of BB in the GO TEAM GO! example where line 30 (described as a 'hard-working' line) does well with TABs and print separators to set out the message. And then, instead of working the line just a little bit harder - to scroll the message into its final position on the screen - a loop of PRINT statements is used.

I can recall (way back then) that I was quite impressed with this 'clever' method of using a loop instead of ten separate (blank) PRINT statements. I now realise that 10 colons at the end of (that hard-working) line 30 will achieve the same result. It might be better to use the 10 colons directly after PRINT so that the scrolling is done first and the 'message' is then printed in the middle of the screen as required.

Will both (or either) of these methods work successfully? Perhaps someone will contribute an article on PRINT to explain what can and can't be done with print separators and TAB.

Moving right along to page 73 and RND, I can recall how pleased I was when I first typed (as instructed) PRINT RND and the screen showed .5291877823 and then .3913360723 as per the book.

This time around I wondered - do all 4As produce the same sequence? And (I surmise that they do) what unknown internal seed is involved? Although it's not mentioned in BB, RANDOMIZE also works in immediate mode. It may seem of little consequence but I have used it in immediate mode.

Detouring to URG for a moment - we are informed that, for RANDOMIZE with a seed, the seed is the first two bytes of the internal representation of the number which is in 7 digit Radix-100 mantissa form. Having no idea what that meant (I'm not RC there either I bet!), I did some experimenting with seeds. I decided to find out which seeds would produce the numbers 1 to 100 - first up in a random series - using INT(100*RND)+1 of course.

One rainy day (two are better still) if you have nothing - and I mean NOTHING - better to do you may care to try the exercise. As a bonus this will give you plenty of time to write an article for the newsletter while you wait for results. I used a FOR-TO-NEXT loop and soon discovered why the STEP facility was provided.

Because URG had informed me that, if I didn't use an integer, the 4A would INT the seed anyway, I started the loop at 1 and increasing. I <u>eventually</u> got my 1 to 100 'random' numbers.

If anyone bothers to check it out, here is a suggestion - after you've looped through integers (some BIG numbers eh?) try some <1ers, even though INTing any decimal less than 1 should give only Ø as the seed. And URG wouldn't make false or misleading statements surely!

You can make a couple of quick checks if you like using RANDOMIZE(a decimal seed) and PRINT RND - in immediate mode of course. Or, if you'd like to check it out more thoroughly, here's a small sample program.

```
10CALL CLEAR
20FOR L=1 TO 100
30FOR K=.0001 TO .01 STEP .0005
40RANDOMIZE(K)
50E=INT(RND*100)+1
60IF E<>L THEN 80
70PRINT E;K;INT(K)
80NEXT K
90NEXT L
```

What does it all matter anyhow? I did such experimentation when I was trying to learn about my 4A as well as trying to learn how to compute. I have since found use for the knowledge of what seeds produce what random numbers. No - it's not in a program to run a draw for a raffle!

Well - that was a lengthy detour (and typical rough going) but before leaving RND and RANDOMIZE, a couple of comments and a query. Does anyone know what seed system XB uses? It's quite different to the Basic one.

I noticed an article somewhere a while back about generating random numbers in the -10 to +10 range - I can't recall the source as I just jot down such brief how-to/try-this articles in an indexed notebook. Being reminded about it just now I decided to check it out. ' The formula given

```
(-1)^(INT(11*RND))*(INT(11*RND))
```

seems to work OK. So I checked TI's standard formula for - to + ranges. It's OK too. Maybe I missed some comment in the special formula article.

Mention of the random number formula brings us back to BB where the formula itself doesn't rate a mention although the procedure is explained in detail on page 82. After which, when this procedure is used on page 88, there is a follow-up comment that it may require a little explanation??

Apart from some minor mishaps, there's not much of critical interest in the remainder of BB. I do recall my first encounter with the flow chart on page 102 and thinking 'this must be how REAL programmers program'. As a result I subsequently chased about for further information about flow charts and I started to use them in my initial programming efforts. Disaster! I was continually in trouble trying to remember what shape 'box' to use. To overcome this problem I ended up using roughly square boxes for everything.

In a very short time I reached the why-bother-with-boxes stage. Now I just write down the various steps required in a kind of rough computerese. Then it's just a matter of fleshing them out - if and when I can!

Except for very small programs this is not a structured sequencing process - more of a first-the-main-bits then the bits-needed-to-make-the-main-bits-work followed by bits-to-tidy-up-and-tie-it-all-together process.

As I reached the final pages of BB, I was almost tempted to get Mr Bojangles to dance again. Somehow I doubted that the effort would generate the almost overwhelming excitement of the first experience.

Then it's page 123 - the last page of narrative - where readers are advised 'You are now well launched into your programming career'. I wonder how many have stayed afloat!

So it's time to close the covers of BB - for the last time I guess. I had the idea of trying a similar URG revisited exercise but decided to defer it until I could manage the additional loin-girding required. And I don't wish to overload the editor - even though he comes with a 12 month and unlimited number of articles warranty. And I have been neglecting several program projects I have in various stages of completion.

So, whilst I am in print, I will take the opportunity for a brief soapbox diatribe. Whilst I still use colour except when I CALL it, I do not write programmes - I write programs. It seems that, around the turn of the century, the word programme came into use. It was a variation of the original English word program. Most likely it was a time when such phoney French forms were 'fashionable'. As a result we are stuck with the introduced affectation as both words are currently acceptable. However I prefer the unadulterated version. Somehow I feel that there may be some antipodean Anglophiles who persist with programme because they consider program to be an aberrant Americanism. Or perhaps they just enjoy typing. Just thought I'd throw that spot of edification into the ring.

Happy computering!

MYARC DISK CONTROLLER

(This is a summary of a text written for the user group in Sweden PROGRAMBITEN 88-4 and 89-1. Received by Tony McGovern from Jan and only very lightly edited.)

by Jan Alexandersson, Springarvagen 5, S-142 61 TRANGSUND, Sweden

A new card from Myarc can control both hard disks and ordinary floppy drives. Up to 3 hard disks, each with up to 134 Mbytes, and 4 floppy drives with up to 720 kbytes each can be used.

The card is sealed but the connectors are brought out on the back of the card. You get two connecting cables for the hard disk and a 70 page manual and 3 disks: Disk Manager V, Geneve MDOS and Geneve GPL+Myword. You can use the first disk manager only with a TI-99/4A.

You can choose any CRU-address between >1000 and >1F00 (16 different).

FLOPPY DISK CONTROL

The disk drives are connected to the same connector as used in the TI-controller for internal drives. Notice that the extra contacts for external drives on the TI are missing from the Myarc. All other connectors are used for hard disks so don't hook floppies up to them.

You can use up to four drives which will be DSK1-4 on CRU >1100 and DSK5-8 on all other CRU-addresses. You can use two disk controller cards at the same time. I have used Myarc on CRU >1000 and TI on CRU >1100 which works well. TI will control DSK1-3 and Myarc DSK5-8. The latter are to be strapped as if they were DSK1-4 but will be addressed DSK5-8 by all programs.

There are DIP-switches for selecting the following items for all four drives individually:

- 40 tracks, 16 ms step time, max 360 kbytes
- 40 tracks, 8 ms step time, max 360 kbytes
- 80 tracks, 2 ms step time, max 720 kbytes

It is possible to use all disk formats between 90 kb and 720 kb:

- SS/SD 90 kb 9 sectors/track TI-original drive
- DS/SD 180 kb 9 sectors/track Max with TI-card
- SS/DD 180 kb 18 sectors/track TI-original drive **
- DS/DD 320 kb 16 sectors/track For old Myarc cards **
- DS/DD 360 kb 18 sectors/track Corcomp and Myarc
- DS/QD 640 kb 16 sectors/track **
- DS/QD 720 kb 18 sectors/track

Formats marked with ** should be avoided for maximum interchangeability between different TI-users. You can use up to 180 kb with the original TI-drive but it is better to change to double sided drives with 360 kb which also have 180 kb DS/SD, the most common type among TI-99/4A users. You can get a new half height drive for USD 90.

5.25 inch QD (quad density = 80 tracks/side) can read 180 kb SD and 360 kb DD but can only write QD. An 80-track drive has a read/write head of very small width compared to a 40-track drives. You can have the following cases:

1.tracks from wide head on a fresh disk
2.tracks from small head on a fresh disk
3.tracks from small head written over wide head track

The first two cases can always be read by both types of drives. In the third case you get read errors with a wide head drive and a rather uncertain operation with a small head drive if the head is somewhat off center of the track. You can also use 3.5 inch 720 kb drives.

Sector zero for a 720 kb QD drive will use blocks of 512 bytes which is the smallest unit you can use. This means that the file header takes 2 sectors and the data sectors are even in number so the file will be 1-2 sectors longer than on a SD or DD disk.

MYARC DISK MANAGER V 1.29

The disk manager is on a disk and works fine and will completely replace TI DM 2 and DM 1000. It can initialize, copy and test disks. Files can be copied, erased and protected. Catalog will also show the number of sides, density and sectors/track and the length of a PROGRAM-file in bytes. A comparison with DM 2 and DM 1000 shows the following:

	TI DM 2	OTTAWA DM1000	MYARC DM V
Number of DSK	1-3	1-8	1-9
FILE copy file	YES	YES	YES
copy with rename	YES	-	-
copy overwrite test	-	-	YES
move file	-	YES	YES
delete file	YES	YES	YES
recover file	-	YES	-
list D/V80 + D/F80	-	YES	YES
list all D/V + D/F	-	-	YES
protect file	YES	YES	YES
remove XB-protection	-	YES	-
rename file	YES	YES	YES
DISK multifile copy	YES	YES	YES
files per pass	one	one	several
copy used sectors	-	YES	-
copy all sectors	-	YES	YES
init with test	YES	YES	YES
init without test	-	YES	-
multidisk init	-	YES	-
sweep disk	-	YES	-
catalog	YES	YES	YES
printer control code	-	YES	-
set disk protection	YES	YES	-
remove disk protect	-	YES	-
rename disk	YES	YES	YES
TEST read test	YES	-	YES
write test	YES	-	YES

Multifile copy has its own command which is missing in DM 1000 but exists in DM 1000 modified for Funnelweb 4.13. DM V also warns you if you try to copy a new file over an old file by the question: File name already exists, Do you want to overwrite (Y/N/All). You can choose Y or N for every old

file. If you choose All then all will be copied without any more questions. This is similar to Replace String in TI-Writer. Several small files are copied in the same pass which not done by DM 2 and DM 1000. You can also use DM V with Horizon RAM-disks as DSK1-9. This is the first disk manager that can handle DSK9 which is missing in DM 1000.

SUBDIRECTORIES ON FLOPPY DISK

A disk can only use 127 files regardless of how it is formatted. It is possible, apart from the ordinary catalog (root), to create three subdirectories (DIR) which can hold 127 files each. The disk can then in total contain 508 files because the root can also have 127 files. Each subdirectory has a head which takes one sector. All unused sectors can arbitrarily be used by the three DIR and the root. It is thus much more flexible in use than Horizon MENU.

A file in a DIR can be called by DSK1.SUBDIR.FILENAME or DSK.DISKNAME.SUBDIR.FILENAME and a file in the root by DSK1.FILENAME or DSK.DISKNAME.FILENAME. The name of the DIR can be 10 characters long but you should use as short name as possible so it can be within the INPUT-length.

If you have four 90 kb disks then you can easily transfer these to the root and 3 DIR. All four disks have individual catalogs so the same file name can exist four times on the disk. Myarc DM V has a special FIND command that will list all places where a particular file name is used.

Another use can be when you write assembler programs and source and object code are saved to different DIRs and the PROGRAM-file is saved to the root. A C99-programmer can create DIRs for C-code, AL-source code and AL-object code and use the root for PROGRAM-files.

NEW BASIC COMMANDS

You can use OLD, SAVE, OPEN, DELETE and LIST in the usual way but also with DIR like SAVE DSK1.XB.GAME.

There are seven new CALLs which can be used in BASIC command mode and RUN of a program. For Extended Basic you can use them only in command mode.

CALL FILES decides how many open files there can be on the disk drives. The maximum number is 9.

CALL ILR is similar to CALL INIT.

CALL LR("DSK1.OBJECTFILE") is similar to number 3 LOAD and RUN in the Editor/Assembler-module. The command loads an object file in DIS/FIX 80-format .

CALL LLR("START") is similar to CALL LINK and starts a program as number 4 RUN in the EA-module.

CALL MDM will load DM V from DSK1.

CALL DT will set the clock. You can also do this with OPEN #1:"TIME" and PRINT #1:SEC$,MIN$,HR$,DAY$,MON$,YR$. You can read the clock with INPUT. This is a hardware clock so it will operate correctly also with 50 Hz and during access to external devices. It has no batteries so it must be set every time you start the computer.

CALL DIR(1) or CALL DIR("DSK1") shows the catalog. CALL DIR("DSK1.SUBDIR") shows the catalog for the subdirectory. A good thing is that the length of the PROGRAM-files is shown in bytes both for Basic and Assembler.

In the same way as for the TI-card you can read the catalog by opening a file with
OPEN #1:"DSK1.",INPUT,RELATIVE,INTERNAL
where "DSK1." can be replaced by "DSK1.SUBDIR." to get the catalog for a DIR. Notice that the dot must be used here but not in CALL DIR. When you read this file you get:
INPUT #1:"DISK$,ZERO,TOTAL,USED
INPUT #1:"FILE$,FILETYPE,SECTORS,LENGTH
There is some differences between TI and Myarc. TOTAL=360 and 720 for Myarc when TI has 358 and 718. Myarc shows the length also for PROGRAM-files when TI shows 0 for these. Myarc has a new FILETYPE=6 for subdirectory and FILETYPE=7 for emulate.

A thing that was unknown to me is that files opened with OPEN and DISPLAY have maximal length of 150 according to the manual, but can handle 156 when I test it. The same is true for the TI-card. You can also open a file with RELATIVE 400 and space for 400 records is reserved from the start (see XB manual p140 - Ed). This will store all records close to each other on the disk which speeds up search of a record. The same thing works both with the Myarc-card and the TI-card. If you open the file a second time with RELATIVE 1000 then Myarc will reserve more space on the disk which the TI-card will not do.

HARD DISK

Up to three hard disks can be used. Each hard disk is connected with an individual address cable and a common data cable. You can use up to 134 Mbytes per hard disk. The best size is 20 Mbytes which is the usual size for most personal computers. The cost in Sweden is about SEK 2000 + VAT. The most common type is from Seagate:

- ST-225 20Mb 65ms 5.25inch
- ST-125 20Mb 35ms 3.5 inch

The hard disk must have a ST506/412 interface but RLL cannot be used. Many 32 Mb drives have RLL so avoid these. If you want a bigger drive then you must choose 40 Mb. You can get a removable 5.25 inch frame for a 3.5 inch drive without any extra cost.

The PE-box cannot power the hard disk so you must get another power supply. The critical voltage is 12 V from which a 5.25 inch needs about 2.5 A and a 3.5 inch needs 2.0 A.

Format of the hard disk takes 4 minutes including test for a 20 Mbytes ST-125 drive. After formatting I got Used 66 sectors Free 78654. With my earlier EPROM H6 I had 2 bad sectors (used 70 free 78650) and with reformat after a week I had no bad sectors (used 68 free 78652). I hope these 2 extra sectors (more than the manual) are no bug. New with EPROM H11 is that it reserves a number of sectors (default 2048) for file headers and directories which will speed up the search and loading in the same way as Mike Dodd's MCOPY for floppy disks. The noise from the hard disk is much less with this new EPROM. The very fast format is done with test at the same time. I am somewhat suspicious about this but I

have no bad sectors today so I cannot try it. If you have a hard drive with bad sectors then you can compare the number of bad sectors after format and after a special test program that takes 20 minutes. Myarc says that they use multiple sector I/O in the first case but not with the test routine. It is possible that there is some kind of track-reading in the first case which is not suitable for TEST. The test of the hard drive does a read and write test without destroying data so when you start it you may not break. A small bug in DM V format of hard drives forces you to have it loaded from DSK1 even when you have changed reload to DSK5.

Each DIR takes 4 sectors and files are typically one sector longer than on a floppy disk.

The hard disk has a root directory and subdirectories to any number of levels. The root and each subdirectory can have 127 files + 114 DIR. These new DIR can also have 127 files + 114 DIR and so on. This means that you can call files by WDS1.GAME.XB.PB.LANDER or WDS.NAME.GAME.XB.PB.LANDER.

The Myarc card also has 32 kbytes RAM (bankswitched). There is always space for 11 open files so CALL FILES decides only the number of files on the floppy disks. The remaining files i.e. usually 11-3=8 can be opened at the same time on the hard disk.

In Micropendium Mar/89 there is a program in assembler for parking of a hard disk. Parking means that you position the read/write heads in a place where you have no data.

EMULATION OF DSK1, DSK AND DSK1-FILE

Some programs need the files to be on DSK1. You can create a subdirectory DSK1 so that the file can be loaded from hard disk even when it is called by DSK1.CHARA1. All other use of DSK1 is directed to the floppy disk. You can also use subdirectories below DSK1 without WDS1. I have used DSK1.FW.CFG which is good when the input only allow a small number of characters. If You use this DSK1-emulation then you should use the physical drive DSK1 as little as possible. All access to DSK1 goes first to the hard disk and then to the floppy drive. This can slow down some programs considerably which can be seen with Spellcheck or sorting with TI-BASE. Use your old 90 kb as DSK1 and use your bigger drives for DSK2-8. Usually DSK1 is only needed for copy protected disks like MG Explorer, MG Diskassembler, MG Diagnostics and Turbo-Pasc'99 (only from TI or Corcomp). All other programs can be placed in the DSK1-emulation (max 127 files, size not important) or if possible with another path to the hard disk. All file names in the DSK1-emulation cannot be reached on the floppy disk in drive 1.

Programs like Multiplan call files as DSK.TIMP.MPBASE. In this case you create a subdirectory DSK which directs the call to the hard disk.

There is a third emulation called File Emulation for DSK1 (CRU >1100 only) which is an exact sector for sector copy of a disk stored under one single file name. This works well with FORTH which accesses sectors directly without files. Several such emulations can be stored simultaneously on the hard disk but only one can be active at the same time. When it is active all calls will go to the hard disk including from a sector editor. Also ordinary disks can be stored in

this emulation. The search goes first to the DSK1-emulation and if the file is not there then the search goes to the file emulation. There will be no access to the physical drive 1. When this file emulation is active then DSK1 becomes DSK2 and DSK2 becomes DSK3.

PROBLEMS

The Myarc HFDC was delivered to me with EPROM H6 and later I sent it back to USA for repair and change to EPROM H11 with DM V 1.29. Myarc replaced a socket for one chip that prevented the clock from operating. I sent the card to USA as "SMALL PACKET" insured air mail. You should always investigate the terms for small packet because this is the cheapest way to send cards (Texaments and DIJIT use it but not Myarc).

If you don't use a hard disk then the first access to DSK1 will take 45 seconds before anything happens. With a connected hard disk this problem disappears but you should not buy the HFDC card without a hard disk.

In a letter to me from Myarc they say that some Fujitsu drives are marginal drives and may not be compatible with the Myarc HFDC especially those with stamped steel frames (zinc diecast is OK). Back-up of hard disk only works to DSK1-3 and not to DSK5-8. CALL MDM only works on CRU >1100 or if you have the MDM5-file in the DSK1-emulation on the hard disk. You must always load MDM5 from DSK1 the first time and then change in SETUP of RELOAD of DM V to your disk number like DSK5 or a path on the hard disk like WDS1.MYARC. It is a good idea always to have several disks ready with different paths so you can get the disk manager if something goes wrong with the DSK1-emulation. MDM5 is sensitive to the load path and I have experienced two problems but the usual load with Editor/Assembler or XB DSK1.LOAD is OK:
- Directory-Utility-Complete-Catalog can crash the screen because it never waits for input prompt (XB-FW-MDM5 or GK-FW-MDM5).
- When DM V is completely loaded then the screen is locked and no key can be used (TW-UTILITY-MDM5).

You cannot set the clock from DM V start menu but if you set it from Basic then it can be used to mark files and directories.

Myarc writes SD-sectors to the disk with deleted data marks F8 instead of FB for a TI-controller. ID/DATA SEPARATORS will be 00.

Another problem comes when you copy disks. When the Myarc writes a sector to disk it does not do a read of the same sector for checking. The problem is the same with both DM 1000 and DM V when the Myarc HFDC is used. It is thus very important to verify the disk when you format it. DM V always verifies but DM 1000 has the option not to do it. A TI-controller always does a READ after each WRITE of a sector. I find this much more secure. Myarc confirms this in a letter to me and says that MS-DOS works in the same way without a READ after WRITE. Can someone who knows IBM PC confirm this.

After long tests with DM V then I have decided not use Myarc HFDC for DS/SD 180 kb. I have tested it with four different

drives TEAC, Fujitsu (2) and Mitsubishi with the same bad result. The comprehensive test takes 10 minutes per loop and I run up to 10 loops which takes more than two hours. With single density (SD) the computer will always (several on each drive) lock-up after 1-5 loops. With DS/DD 360 kb I have run 10 loops (twice) on all drives without any problems (total 80 loops). DS/QD 720 kb also works perfect. I am not sure if the fault is in Myarc or in the disk drives. Is Myarc too fast or the drives too slow or will 50 Hz mains slow down the TI-99/4A (CALL SOUND and interrupt clock is 20 % too slow).

DS/QD 720 kb floppy drives show different numbers of sectors for a file with CALL DIR and DM V but I don't think it is any problem. If you save a very short text from TI-Writer to a QD-disk it will have an extra sector (apart from the problem above) compared to a Save File to a SD- or DD-disk. When this bigger file is copied to SD or DD then the file is still too big but I don't think it is a real problem. I will investigate this more in the future.

I have seen reports about heat problems with other Myarc cards. Myarc has a sealed card with no cooling of the voltage regulators. One Geneve user in Sweden had a real problem so he has decided to move the card far away from the warm AC/DC-converter (don't use slot 1), remove the card sealing and change the AC-setting to 240 V instead of 220 V. I have also seen concern from Australia about the Myarc 512 kb RAM disk and a possible need for an extra heatsink.

FUTURE EXPANSION

Myarc has prepared the card for tape streamer and 1.44 Mbyte floppy drives. Software for these is not delivered with the card.

A tape streamer is an easy way to back-up a hard disk. The price is about SEK 3200 + VAT in Sweden so you may instead want to buy a second hard disk. DM V has routines for back-up of floppy disks but you need a lot of them for a full hard disk. Each file has a back-up flag so you need only to save changed files to the floppies.

A DS/HD 3.5 inch drive can store 1.44 Mbytes and has 80 tracks/side and 36 sectors/track. HD stands for High Density. These drives cost 20 % more than 720 kb 3.5 inch and the floppy disks cost more than double. Today there is no economical reason to buy 1.44 Mb. The disks will store max 508 files if you use the root and three subdirectories.

REFERENCES

Micropendium:
Mar 88: Myarc ships HFDC
Mar 88: Power supply for hard drive
Apr 88: Organizing your hard disk
Sep 88: Myarc HFDC review
Sep 88: Tips for new hard disk users
Oct 88: Hard disk backups and downloads
Dec 88: Using HFDC emulate files
Mar 89: Parking your hard drive

Mr R Carmany
1504 Larson Street
GRENSBORO
NC 27407
USA

'ION PAGE

FORTH
Forth SIG will be held at Richard's Surgery again. First tuesday (3rd) in October. The last SIG set a programming task for those attending. Nice to see one of out of Towners, Geof Gray attending. This month we will look at the programming effort and advance further.

COMMITTEE MEETING.
Second tuesday, 10th October, once again at Boolaroo Ambulance Station. Last meeting was well attended. Main topics discussed were, new drives for the club, Great mail out, King's Theatre night and the Christmas do.

EXTENDED BASIC.
If you can make it to Bob McClure's house on the third tuesday in October (17th), Gary Jones will treat you to an evening on Extended basic. These Sig get-to-gethers are very informal and very informative.

GENERAL MEETING.
Jesmond Community centre at 7:00 pm sharp on 24/10/89. Joe Wright will be showing the test version of his new Genealogy Record Keeper.

MEMBERSHIP.

Membership fees for the Hunter Valley 99'ers.
AUSTRALIAN MEMBERSHIP.....$25:00
OVERSEAS MEMBERSHIP.....$40:00 Australian.

Address membership requests or renewals to Brian Woods, his address is inside the front cover.

CALENDAR 1989

Social secretary Bob McClure now requires deposits for the Kings Theatre night no latter than the next General meeting.

Our Christmas party will be held at the Boolaroo Ambulance hall in December, all members and their guests are welcome. We will be having plenty organised for the kids (both big and small). Any ideas of things to do on the day please see Bob McClure with your ideas.

GREAT MAIL OUT.

The great mail out will be starting with in a few weeks. Once Brian has an updated mail list for me the Mail out will commence. We will be sending out all the extra newsletters we had accumulated over the last couple of years. Each out of town member will receive a handfull of newsletters to read and keep or pass them on to a member who happens to live near you.

SOFTWARE LIBRARY.

Stewart Bradley our young librarian tells me that things have been a bit quite lately. The cataloging of the complete library is continuing, slowly, it must be admitted but continuing it is. We really do want to send out those listings to all our members ASAP.