HUNTER VALLEY 99ERS USERS GROUP

HOME COMPUTER NEWSLETTER

The Newcastle Star
&
The University of Newcastle
7th Newcastle Microcomputing Exhibition

Exhibition Hours

Thursday, 20th September - Midday to 9pm
Friday, 21st September - 9am to 5pm
Saturday, 22nd September - 9am to 5pm

REGISTERED BY AUST POST
PUBLICATION No. NBG8023

Hunter Valley '99ERS USER GROUP

AUGUST - SEPTEMBER 1990

YOUR COMMITTEE

President

Peter Smith
8 Glebe St.
East Maitland 2322
PHONE 336164 V/tl 493261640

Secretary

Brian Woods
9 Thirlmere Pde.
Tarro 2322
PHONE 662307

Treasurer

Noel Cavanagh
378 Morpeth Rd.
Morpeth 2321
PHONE 333764

Software Librarian

John Paton
1 Parlen Close
Rutherford 2320
PHONE 326014

Editor

Allen (Joe) Wright
77 Andrew Rd.
Valentine 2280
PHONE 468120

Purchasing/Hardware

Alan Franks
822 Pacific Highway
Marks Point 2280
PHONE 459120

CONTRIBUTIONS

Members and non members are invited to contribute articles for publication in the Hunter Valley 99'ers Newsletter.

Any copy intended for publication maybe typed, hand written, or submitted on disc media as files suitable for use with TI WRITER. 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. filename etc. The preferred format is 75 columns and page length 66 lines, right justified.

All articles printed in HV99 News (unless notified otherwise) are considered public domain. Other user groups wishing to reproduce material from HV99 News may feel free to do so so 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.

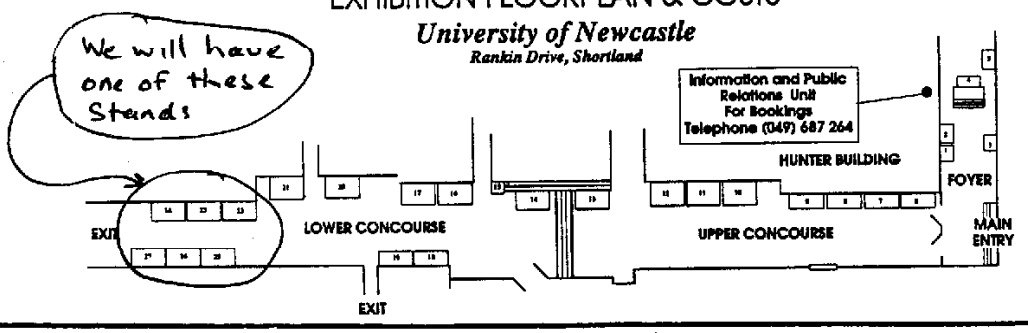
Whilst every effort is made to ensure the the correctness and accuracy of the information contained therein, be it of general, technical. or programming nature no responsibility can be acepted by HV99 news as a result of applying such information.

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.

7TH NEWCASTLE MICROCOMPUTING EXHIBITION.

EXHIBITION FLOORPLAN & COSTS

University of Newcastle
Rankin Drive, Shortland



PRESIDENT'S REPORT.

Well we've had our election and you can all see that you had a lot more sense than I thought. You realized that we need a few more than three people at the helm if the club is to continue and I'm glad you voted that way .

We look forward to a hard year with a couple of projects in the wind and more controvosy about special interest groups around different makes of computers.

At the moment the main topic for discussion revolves around the construction of the famous RD200 Quigg ram disk.

We are at present trying to assertain just how many of these wonderful devices we should prepare, as we have received a number of orders for same from heare and overseas.

We are simply selling the bare boards with the necessary PAL chips socketed on board so that the purchaser can build up the board to his/her specification.

If you would like one get your deposit of $50.00 (AUS) to our secretary BEFORE 30/9/90 cause that is cut-off day for orders. We cannot wait anylonger than that. Orders will be filled in the order they are received. If you are late, we can not promise you that we will supply your order. Its as simple as that. We have to order boards and pals, so if we haven't got enough, it would not pay to order only one or two as quantity decides price.

A number of the guys are enthused about the COMPUTER SHOW at the UNI (same old place, same old location) and a big thanks in anticipation to ROSS MUDIE who is coming from Sydney to demonstrate his train controller on the day.

We need signs and guys to man the display from 9.00 a.m. till 5.p.m. on Saturday, 21st September, 1990. If you have something you would like to demo or something you would like to see or have explaine, come along, its a good excuse to waste a day doing things like that.

Membership is surprising and renewals look good. Keep rejoining.. even if you are like most of us and have another computer. So what..

THE TI LIVES.
LONG LIVE THE KING.

Whoops getting a bit carried away there.

Tony and Ron are at it again with updates and modification. Great to see.. What about the rest of us.. are we just going to sit back and watch the death-throes or are we going to still do a bit?

The Sydney club is having a swap and sell day at their next meeting.. 1st Sat in September... Take down some money and some bits and pieces... It should be a good day.

Joe has been working hard on the s/w list again and it's nearly finished. Boy have we got some gear?

Don't throw the old girl yet There is a whealth of untried programs waiting for you.

Our meetings have been rather spontaneous lately and have been very good. What things would you like to see? do? watch? findout about??? Please let us know.

How about guest-speakers, visits to industry, visits to other clubs, exhibitions with other computers???...

Its up to you . Let the exec know and we'll try just about anything.

REGARDS
PETER.

GAMES LOADER.

Some of you may have in your possesion a set of disks which contain a whole host of games loaded down from module. Games include caverns,Attacker, Diablo, Froggid, Handydandy, etc.. too numerous to mention. I suppose like most of us you have some favourites amongst them and, also like most of us you have your favourites on different disks. Well if you follow the simple instructions below, you will be able to crash a whole disk-full of favourite programs at a time rather than just one favourite and a few "also-rans".

Firstly select the games you want to place on the disk, initialise a blank disk using your favourite formating program and then copy the files for your favourites from their disk, to your nice fresh one.(sometimes you may need to copy a couple of files for just the one program. eg "FOOTBALL1", "FOOTBALL2" for FOOTBALL or "NIGHTMASON", "NIGHTMASOO" for NIGHTMASON.)

Somewhere list the files you copied to your new disk and underline the FIRST of each set.(eg in the above example you would underline "FOOTBALL1" and "NIGHTMASON".)

On each one of those old disks you will find a program called "load" which enables the programs to be run from EXTENDED BASIC.

To complicate matters there happens to be a number of "load" programs on the disks, but one of them is easy to modify so that your assortment of favourites can have their names presented easily to you in a menu when your disk is booted.

To assertain on which disk the "good" load program is, you should try the following...

Place a games disk in your drive and allow it to boot to the menu.

Hold down <FCTN> and tap <4>.

This should halt the "load" program and allow you to...

List (simply type <list>) the program.

It should look something like this....

```
100 DATA 4,CHIS/TRAIL,0, HOPPER,0, JUMPY,0,KONG,0
105 DATA ROTOR/RAID,0,TREAS/ISLE ,0,WINGWARS,0
110 REM place #of programs, progname, 0,progname,0 etc on above line
120 REM this program has a hidden loader do not merge it
125 CALL SCREEN(4)
130 CALL INIT :: CALL CHAR(128,"F0F0F00F0F0F"):: CALL
COLOR(13,16,4)::                    DISPLAY AT(3,3)ERASE ALL :"ASSEMBLY
GAMES DISK 3"
135 DISPLAY AT(4,1):"Program files collated and/":" or converted by Col
C.":" THIS DISK NOT FOR SALE"
140 DIM A$(20),J(20):: READ M :: GOTO 150 :: CALL LOAD :: CALL LINK ::
CALL HCHA R :: CALL KEY :: X,K,S,T :: !P-
150 CALL HCHAR(1,1,128,32):: CALL HCHAR(8,1,128,32):: FOR X=1 TO M ::
READ A$(X) ,J(X):: DISPLAY AT(X+9,5):CHR$(64+X);" - ";A$(X):: NEXT X ::
CALL HCHAR(1,1,128, 32)
170 DISPLAY AT(23,3):"ENTER YOUR CHOICE." :: CALL HCHAR(24,1,128,32)
175 CALL KEY(3,K,S):: IF S=0 THEN 175 ELSE X=K-64
180 IF X>M OR X<1 THEN 170
190 DISPLAY AT(12,8)ERASE ALL:"NOW LOADING:": :TAB(8):A$(X) 200 REM .T = 0
LOAD VDP WITH E/A VALUES <>0 DONOT LOAD THEM
210 CALL LOAD(-30,J(X)):: FOR I=1 TO LEN(A$(X)):: CALL
LOAD(-300+I,ASC(SEG$(A$(X ),I,1))):: NEXT I
220 CALL LOAD(-305,5+LEN(A$(X))):: CALL LOAD(8196,254,160):: CALL
LINK("XFER")
230 DISPLAY AT(3,7)ERASE ALL:" ALPHA LOCK UP !!!" :: RESTORE :: READ M ::
GOTO 50
```

Of course the names in the data statement in line 100 will most probable be different, but don't worry , we intend to change them anyhow.
If the program looks comletely different, try another disk until....
When you have located the correct "load" program, copy it to YOUR disk which contains your favourites AND USE DM100 or similar, to UNPROTECT it (Change the p to a u in the last column beside the name "LOAD")
Insert your new disk in drive 1 and allow the menu to boot-up.
Once again hold down <FCTN> and <4> to stop the program.
Type 100 and then <FCTN> and an arrow key.
On the screen will appear line 100 of the "load" program.
Change the number after the word "data" to the number of programs you have on your disk, then change the names following to the file names you underlined a fair while ago. If necessary, delete any unwanted names but remember to leave ",0," between the names, and end the data statement with ",0" . When you have finished with that line, press enter.
Type "SAVE DSK1.LOAD" when you have finished this mumbo-jumbo and press return. If all is well you can now type run and your new menu should appear and hopefully run.
Thanks go to the menu deviser "Col.C" whoever he may be .. .

From The Secretary's Desk - AUGUST 1990

OUR NEWSLETTER

Due to a falloff in the number of articles that are being submitted for publication, The Hunter Valley 99ers Newsletter will in future be published every second month. As more & more members of our Group either move on to other computers or just lose interest in computing it has become more difficult to publish original material - a fact that is also evident in many of the newsletters we receive from around the world. If YOU want the club to continue publishing newsletters on a regular basis it is up to YOU to provide some input.

RD200

After a very good review of the Quest RD200 RAMdisk appeared in a recent issue of MICROpendium (written by our own Bob Carmany), my postman has been very busy delivering letters from the US, Canada & the UK, not to mention a couple of local users. If any of our members or readers from other Groups would like to add a RAMdisk (or 2) to their system, are advised to get in touch with me NOW. If the demand warrants it an order for more boards will be placed at the end of September - after that it will be too late.

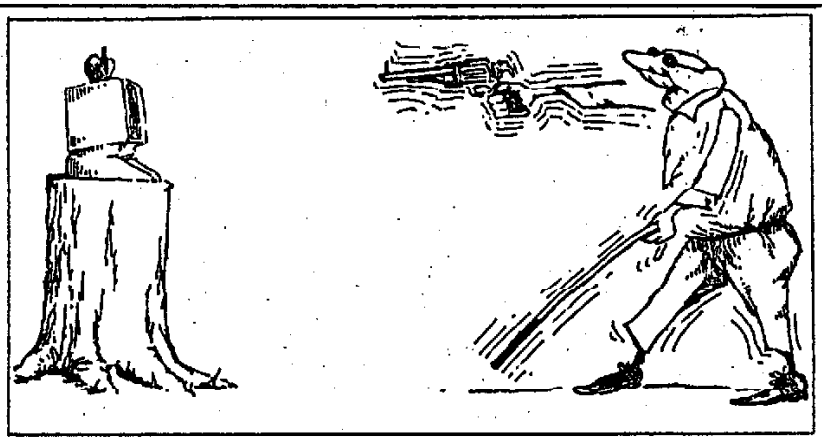
IN THE NEWS

Apart from MICROpendium, one of the best sources of TI-related information appears in Gary W. Cox's "in The News" column in TIdbits, the newsletter of the Mid South Users Group from Memphis, Tennessee. From various issues of this newsletter comes the following snippets...

Texaments has released Artoons, a three disk collection of cartoon artwork designed to be used with TI Artist PLUS! Artoons! is a collection of more than 60 famous cartoon character renderings stored in TI Artist 'Instance' format, which allows for modification and use to create personalized pictures and scenes. Artoons! is available for $US12.95 plus shipping & handling and requires TI-Artist PLUS!

Chris Pratt of ESD (Electronic Systems Development Corp) announced at the recent Lima Faire plans to release a hard and floppy disk controller. According to ESD, the card will have a "new revolutionary design using surface mounted parts". Reportedly, its "unique design eliminates heat problems on the card". Moreover, the card "features an EEPROM which allows future DSR upgrades to be loaded by disk," and "set up information stored in the EEPROM eliminates dip switches on the board". Modestly, ESD claims that the card is "designed by an electronics and computer corporation which supports its users." We'll see...

Asgard Software has released two new adventures for the Adventure module - Castle Darkhole & Rattlesnake Bend. Both adventures are available on cassette or disk for a suggested retail price of $US8.95 each.



TI-BUG

WHY SHOULD YOU LEARN TO PROGRAM?

This article was written by the "Tigercub" himself, Jim Peterson, and appeared in the April 90 issue of the newsletter of the Hamilton, Ontario User Group.

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

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

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

Why should you learn to program? For one reason, because I know that you would like to make some changes in the program that you use frequently. I know that because the only feedback I ever get is from people who wish that I would change this or that! You really wouldn't have to learn very much to change colours, add or silence a beep or burp, output to disk instead of printer, etc, etc. Beyond that, unravelling someone else's code can be tricky and frustrating (and I pity anyone who tries to unravel my code!) Often I find it easier to just rewrite the basic idea in my own way.

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

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

There are those that prefer to exercise their creativity with the soldering iron, those who can plug in chips and soup up a Model T computer to run like a Ferrari. I regard them with awe and wonderment, and I'm glad they are around. Without them I wouldn't have a RAMdisk, and my equipment wouldn't get repaired. Personally, I am the ultimate klutz. If I approach my car with a screwdriver, all four tyres go flat. My one feeble attempt to repair my P-Box resulted in failure, expense and embarrassment. But, without having more than a faint idea what goes on beneath that keyboard, I have learned to punch the keys (two right fingers and a left thumb) and create hundreds of programs and routines which have given me a great deal of satisfaction.

Its been fun! You should try it sometime.

Assembler Executing . . .

By Bob Carmany

(This is going to be a series of narrations about my "adventures" through the wonderous world of Assembly Language. I have Ron K and Tony McG to thank for prodding me into this. As these articles start, I have yet to write my first A/L program. If I can try it, anyone can!)

Richard Terry has long since quit "struggling" with Forth and Tony McGovern doesn't "live with spiders" anymore -- he bloody well has the lot of them trained! With that in mind, I have decided to try yet another adventure proposed by several of our UG members.

Some of the best "con artists" come from Australia. Not the least of which are our very own Ron Kleinschafer and Tony McGovern. "Learn Assembly Language" they said, "it's easy!" OK, I'll give it a try --- I may regret it later but . . .

The first thing to do is to find a book for the beginner dealing with the basic ideas of Assembly Language (hereafter called "A/L"). I discovered much to my dismay that the rather extensive manual that comes with the E/A cartridge assumes a prior knowledge of A/L. Anyway, I finally found a rather elementary text on the subject and decided to spend some time learning to program in A/L -- after all, it was supposed to be easy!

I quickly discovered that books aren't written in logical order. This one was making comparisons between XB and A/L coding and I decided that wasn't the best way to start. You have to understand some basics before you can get tha far. For example, there was a good deal of discussion of converting numbers from one base to another --- a good place to start!

There are three number bases that we have to deal with in A/L programming. I could see that this was going to be fun! Binary (zero's and one's) is the only language that the computer understands. Fortunately, we no longer have to program in binary -- an interpreter does that for us. The other two number bases are hexidecimal (base 16) and decimal (what we all learned in school). I could see that this was getting easier all the time. One of the number bases had already been eliminated. All I had to do was learn how to convert a number from decimal to hexidecimal and vice versa.

OK, let's see what the book has to say! You take the decimal number and divide it by radix 16. I didn't know there was gardening involved in this! I have a whole row of radixes planted out back --- my errór, that's radishes! Sorry, back to the task at hand. This is awful! You have to keep track of these "F's" and "A's" when you divide the numbers for the conversions. Anyway, I managed to get through the exercises in the book but it must have had the wrong answers in a couple of places because they didn't agree with my results at all! I could see right off that I had been sold a "blind horse" by Ron and Tony! You know what? I got through the whole chapter and you know what the book said? "The easiest way is to use a decimal to hexidecimal calculator or a computer program to do the calculations for you". Hmmm! I think I just happen to have a program that does that. In fact, it gives you the equivalent in all three bases! So much for that chapter and here is the conversion program.

```
100 ON WARNING NEXT :: CALL CLEAR :: H$="0123456789ABCDEF" :: PRINT
"DEPRESS YOU R ALPHA LOCK KEY": :"PRESS LETTER FOR INPUT BASE": :

110 PRINT : :"D=DEC # H=HEX # B=BIN #": : :: CALL SOUND(80,660,6)

120 CALL KEY(0,K,S):: IF S<1 THEN 120 ELSE ON POS("DHB",CHR$(K),1)+1
GOTO 110,13 0,140,150
```

```
130  INPUT  "DEC  #=":DEC  ::  IF DEC<-32768 OR DEC>65535 THEN 130 ELSE
A,DEC=INT(DEC -65536*(DEC<0)):: GOSUB 200 :: GOSUB 220 :: GOTO 160

140 PRINT "HEX #=" :: ACCEPT AT(23,7)BEEP SIZE(4)VALIDATE(H$):HEX$  ::
GOSUB 180 :: GOSUB 200 :: GOTO 160

150  PRINT  "BIN #=" :: ACCEPT AT(23,7)BEEP SIZE(16)VALIDATE("10"):BIN$
:: GOSUB 1 90 :: GOSUB 220 :: GOSUB 210

160 A=INT(DEC/256):: PRINT :"D=";DEC;TAB(12);A;DEC-A6 ::  IF  DEC>32767
THEN PR INT " ";DEC-65536

170   PRINT   "H=   ";HEX$:"B=   ";SEG$(BIN$,1,8)&"  "&SEG$(BIN$,9,8)::
HEX$,BIN$="" :: A ,DEC=0 :: GOTO 110

180  HEX$=SEG$("0000",1,4-LEN(HEX$))&HEX$  ::   FOR   I=1   TO   4   ::
A,DEC=DEC+(POS(H$,SEG$(HEX$,I,1),1)-1)  (4-I):: NEXT I :: RETURN

190              FOR              I=1              TO              LEN(BIN$)::
DEC=DEC-2 (I-1)*(SEG$(BIN$,(LEN(BIN$)+1-I),1)="1"):: NEXT I :: RETURN

200 A=A/2 :: BIN$=STR$(-(A-INT(A)<>0))&BIN$ :: A=INT(A):: IF A THEN 200

210 BIN$=SEG$(RPT$("00",8),1,16-LEN(BIN$))&BIN$ :: RETURN

220 A=DEC+65536*(DEC>32767)

230    HEX$=SEG$(H$,(INT(A/4096)AND     15)+1,1)&SEG$(H$,(INT(A/256)AND
15)+1,1)&SEG$(H   $,(INT(A/16)AND  15)+1,1)&SEG$(H$,(A  AND  15)+1,1)::
RETURN
```

Maybe this isn't going to be so bad after all. I managed to finesse having to calculate all of those conversions by hand with a short program that I found in my library. Let's see, there is something about registers in the next chapter.

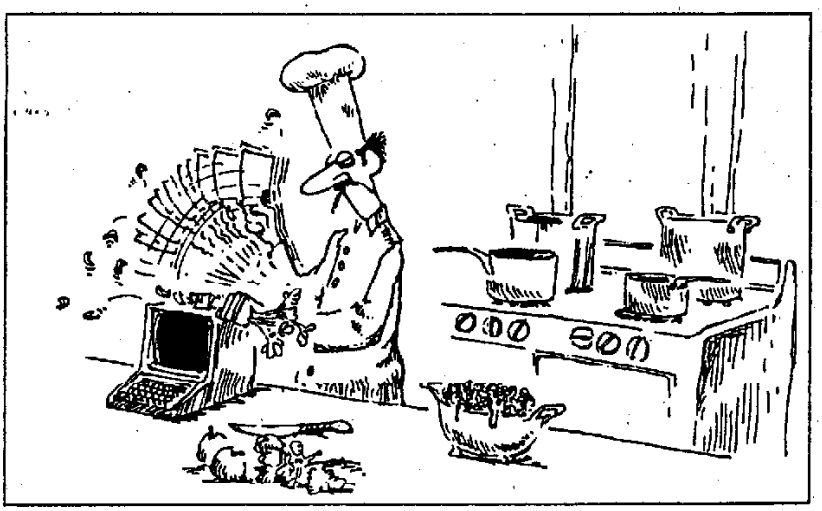
This one starts of with a rather innocent statement. It says that there are three internal registers in the TI CPU --- the Program Counter, the Workspace Pointer, and the Status Register. No worries, mate! This doesn't look to be too difficult! The Program Counter (PC) is a special register that keeps track of the address of the instruction to be performed. After the instruction is performed, the CPU adjusts the address to the next instruction. Right --- the same thing as a line number in XB! Geez, I might have to apologize to Ron and Tony for what I wrote earlier. This isn't too bad so far!

Now for the Workspace Pointer (WP) is another register that contains the address of the program's workspace. Whew! A sentence that says absolutely nothing! OK, a workspace is a memory area of 16 words of memory that are accessed faster than the rest of the computer's memory. Each of these words is referred to as a working register. Aha! I bet that is what they are talking about with those R0 to R15 things in the A/L source code. That means that the Workspace Pointer must point to the first of the working registers -- R0 --- and the rest must follow immediately in memory. This stuff is getting a little more complicated but I think I can grasp the concept.

Now for the last of these registers --- the Status Register (SR). The book says that it holds the individual status bits that are affected by the instructions are executed. Now that makes no sense at all to me. It seems that each of the status bits is affected differently depending on the instruction executed and they can be read by the conditional jump instructions to make the program branch to another routine. That sound like the "IF --THEN" statement in XB. I guess I'll have to wait until I work with the individual instructions to see which of the 16 status bits they affect. Hey! They even provided a chart:

Name	Abbreviation	Bit Position
Logical Greater Than	L>	0
Arithmetic Greater Than	A>	1
Equal	EQ	2
Carry	CY	3
Overflow	OV	4
Odd Parity	OP	5
Extended Operation	X	6
Not Used	--	7-11
Interrupt Mask	I0-I3	12-15

I've heard of some of these at one time or another but I guess I'll just have to wait and see how they can be tested and used by the various A/L instructions.
All of this reading and writing has made my mouth dry! It's time for a cold Foster's (no Toohey's to be had here) and a bit of a rest before I start the second article in this series. I think I'll look at the structure of a bit of source code and maybe see if I can translate some familiar XB statements into corresponding A/L source code. That should be interesting!



TI-BUG

W-AGE/99 * NEW-AGE/
99 * NEW-AGE/99 * N
EW-AGE/99 * NEW-AGE
/99 * NEW-AGE/99 *

* by JACK SUGHRUE, Box 459, East Douglas, MA 01516 *

#4

Many of my computer correspondents have a basic 4A system upgraded to include a tape recorder and that's where they want to (or have to) stay. Though I could hardly imagine life without multi-drives, RAMdisk, ugraded controllers, and all the rest, computer life in the slower lanes is not all that bad. After all, Harry Wilhelms (E-Z KEYS) and Eric LaFortune (ROCK RUNNER) produced two of the most powerful items in TI software using just the tape recorder. In the process they both discovered unknown (and thus untapped) potentials of our great machine. Most tape sources have dried up: IUG, Amnion Helpline, Tigercub. User groups, Triton, Asgard, Texcomp, and Kidware are about the only regular tape sources left. Some user groups (like Lima and MUNCH) still have extensive tape libraries for members. TI fairs everywhere still have piles of tapes available. At last year's New England Fayuh, for example, I purchased a dozen new (still in packages) tapes:

BEGINNER'S BASIC TUTOR (from TI), far better to use with a novice or kids than TI's TEACH YOURSELF BASIC (which is too mathematical for most casual users).

BEST COMPUTER COACH: TEXAS INSTRUMENTS (from Boston Electronic Systems Training) extremely clever. It comes with two cassettes - one with programs and data and the other an audio tape to listen to and easily follow along while computing.

LEMONADE (from Kidware), though less graphic than Apple's version, is many times better. I use both in my classroom. Kids prefer Kidware with more options and more intelligent control. All Kidware tapes have Side Two. LEMONADE contains a super code-breaker game. . Kidware stuff is always good TI stuff.

THE WIZARD'S DOMINION (from American Software Design and Distribution Co.) fantasy adventure with a superb manual (unusual for adventures) making it a joy to play.

COSMIC CAVER (from CompuTech Distributing Inc.) timed space arcade game with twists, including a possibly-bottomless pit.

COSMOPOLY (from Not-Polyoptics) has got to be the most bizarre form of Monopoly ever devised. The setting is the Universe of the future and the options in this fast-paced, ingenious game are wonderful.

HANG-GLIDER PILOT (from Maple Leaf Micro Ware) up to four players test "gliding/landing" skills.

STARSHIP CONCORD (from Futura) another spaceship game with a good manual and so-so graphics.

MISSILE WARS (from Asgard) by John Behnke is one of the best of this genre on tape.

AZTEC CHALLENGE (from Cosmi) well-done, multi-level ancient obstacle course game tha's fun and quick.

CAVERN QUEST (from Moonbeam) about as "acadey" as you'll get on tape and one of the best multi-level graphic obstacle games.

My final tape purchase that day, ROMEO (from Extended), was lost or stolen after I gave a demo of it a few years ago. I'm not very good about making backups of my originals, unfortunately. By the time ROMEO disappeared, it couldn't be purchased anymore. So my joy at seeing one

for sale at the fair was great. Cute Romeo has to get past a series of sand dunes via balloons, is unceremoniously dropped into a shark-ridden sea, swims into a dangerous cave, and so on in his quest for the fair Juliet. It's one of those delightfully addictive, nonviolent games. Now a new generation (my 5th-graders) are discovering the joys of noble quests.

These twelve tapes are things I didn't own but now use and enjoy. Original prices on these items were from $49.95 to $9.95. I picked up most for under $2 (not counting the ones from Kidware and Asgard still being distributed today).

When I came across these tapes in class the other day, I realized how often the kids continue to use most of them, along with some other tapes that I have in large bookcase-style tape racks. Tapes get used a great deal: Jim Peterson's always exceptional educational tapes; Intellestar's (CELLS), early TI's (HAMMURABI, WORD SAFARI), and many others. I teach ASL (American Sign Language) in class, and the kids use the PD FINGERSPELL program to learn, review, write, and decode through the manual alphabet. This is in EVERY user-group library.

Last week we were studying the skeletal system. I put on Regena's "Name That Bone." I often use the tape recorder on the disk-system TI I have at school, also. Once a program is loaded into memory, I take the little tape recorder to the next machine and repeat the process. Sometimes I bring a third computer in from home, but I still just go from one to the other with the same tape recorder.

But that day I loaded up "Name That Bone" by tape into the two TIs, and all the kids during the day had a chance to successfully complete this great program.

There's no problem using tape. I load them into the computers before school, while I'm getting my other stuff ready for the day. I keep the volume on the TVs high so I can hear when one computer had loaded; then I repeat the process for the others. By the time the kids arrive, I've had my coffee, put up the computer schedule, and we're all rarin' to go. I still think the 4A is the best educational computer tool in existence.

I often think about users with the basic diskless systems. There are still tapes readily available for the Adventure, Tunnels of Doom, and LOGO modules (though the last requires 32K). Triton still has cassettes of all kinds for as low as $1.99. I just bought a SAMS book for $2.49 (TI-99/4A GAMES) that included a cassette of all the games. I usually pay more for blank cassettes alone.

Peruse the mail order palaces to see the number of extremely low-priced MODULES still available. Triton's start at $2.49 and go up to $29.95 (for Extended BASIC). There are recreation (MOONSWEEPER, FATHOM, MUNCHMAN), productivity (PERSONAL REAL ESTATE, HOME FINANCIAL DECISIONS); education (READING FLIGHT, NUMERATION I); and other cartridges. TEXCOMP's module prices start at $4.95 and have many more cartridges not listed by Triton, including the last of the Atarisoft ones like Donkey Kong.

So a person with a very basic 4A system (console, TV, Extended BASIC cartridge, and tape recorder) still has an extremely powerful tool at his or her command with options for many other diskless peripherals. But most early owners have closeted or tossed their TIs. Recently, I went to a flea market in a nearby town and picked up a used (but very new looking) silver and black console with cables for $3! <u>That's</u> what I'm writing this article on right now. So DON'T QUIT! Your 4A is alive & well & kicking up its heels all over the world.

(If you use NEW-AGE/99 please put me on your exchange list.)

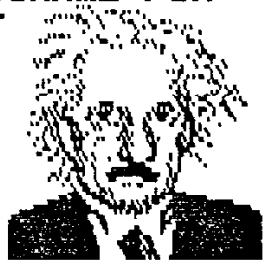
NEW-AGE/99

BY JACK SUGHRUE
BOX 459
E DOUGLAS MA 01516

#6 PAGE PRO - PART 1

DESKTOP PUBLISHING FINALLY ARRIVED FOR OUR 99. THERE ARE MATURE COMMERCIAL AND FAIRWARE AND PUBLIC DOMAIN GOODIES ALL OVER THE PLACE. THERE ARE SO MANY, IN FACT, THAT IT IS QUITE IMPOSSIBLE TO KEEP UP WITH THEM.

BESIDES TI-ARTIST-PLUS, WHICH WILL BE REVIEWED SOME OTHER TIME, THERE ARE PILES OF INNOVATIVE PRODUCTS FROM COMPRODINE WHICH I'VE READ AND HEARD ABOUT BUT HAVE NOT YET EXPERIENCED. THERE ARE CSGD'S GREAT PROGRAMS AND NUMEROUS PUBLIC DOMAIN AND FAIRWARE PROGRAMS FOR BANNERS, LETTER-HEADS, LABELS, AND SO ON.



YOU HAVE TO BE

TO FIGURE OUT SOME OF THESE PROGRAMS.

MANY OF THEM ARE SEVERELY LIMITED (THOUGH SOME LIKE GRAPHIC LABELER DO EXACTLY WHAT THEY'RE SUPPOSED TO IN A FANTASTIC WAY).

NOW THERE ARE A PAIR OF WONERFUL TREASURES FOR ALL 99ERS WITH DISKDRIVES AND PRINTERS:

TIPS

(TI PRINT SHOP - PUBLIC DOMAIN PROGRAMS WITH A HUGE COLLECTION OF PIX)

& PAGEPRO

(FROM ASGARD, THE BEST PAGE MAKER AVAILABLE FOR THE 99 AND GENEVE)

RON WOLCOTT, WITH ABLE HELP FROM BARRY TRAVER AND OTHERS, HAS GIVEN TIPS TO THE TI WORLD FREE OF CHARGE (SEE YOUR USER GROUP OR CONTACT JIM PETERSON FOR THIS AND OTHER PD AND FAIRWARE GRAPHICS PROGRAMS). TIPS LETS YOU MAKE BANNERS, LETTERHEADS, LABELS, GREETING CARDS, AND MORE. WRITTEN IN XB, TIPS TENDS TO BE SLOW AND A BIT CUMBERSOME. YOU MUST PRINT OUT THE DOCS AND FOLLOW THEM CLOSELY TO USE THE PROGRAMS SUCCESSFULLY!

BUT IT TAKES A LITTLE TIME TO MASTER AND TO PRINT.

ONE OF THE MOST IMPORTANT ASPECTS OF TIPS IS ITS COLLECTION OF GRAPHIC ART FROM THE BIG BLUE WORLD. THERE ARE ABOUT 5,000 PIX NOW A PART OF THIS PACKAGE, WHICH INCLUDES PROGRAMS TO CONVERT TO TI-ARTIST AND PAGEPRO, AMONG OTHER ITEMS.

THIS REVIEW USES ALL TIPS PICTURES WITHIN THE SUPERB FRAMEWORK OF PAGEPRO.

WHICH, OF COURSE, BRINGS ME BACK TO THE TOPIC OF THIS REVIEW. THE DAY I GOT THE ORIGINAL PAGEPRO I WAS UP TO THE VERY WEE HOURS PLAYING WITH IT IN ALL KINDS OF INGENIOUS WAYS. WITH PP YOU CAN PUT ANY GRAPHICS (OVER 5,000 IF YOU HAVE TIPS) ANYPLACE ON YOUR PAGE. YOU CAN TYPE OVER OR INTO THESE PIX (AS IN THE TOP AND MOON PIX). THE ORIGINAL PERMITTED 28 PIX PER PAGE. VERSION 1.5 ALLOWS UNLIMITED GRAPHICS BY SAVING PAGES FULL AT A TIME, ALONG WITH ALL TEXT.

Because of this saving method, PAGEPRO also permits an unlimited number of fonts per page:

Gothic 2 Sample, Script 1 Sample, **TULO SAMPLE,** etc.

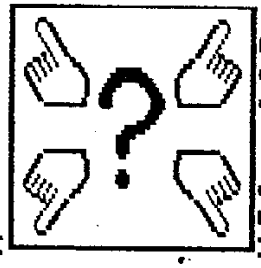
If you've written your text on FUNNELWEB or any DV/80 textfile maker, you can easily import that text into your PAGEPRO pages, though personally, I find it just as easy to type right onto my pages and place the graphics (TIPS) and borders and fonts (PAGEPRO) just where I want as I'm going along.

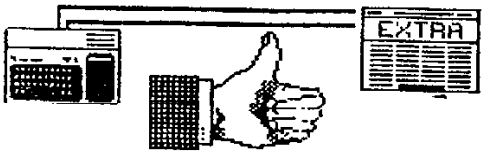
PAGEPRO us extremely user friendly. Of all the various pagemakers and semi-pagemakers I've used for the TI (and for some other computers), I've found PAGEPRO by far the easiest to use. The commands are mostly single keypress and FAST! If you have any kind of RAM capacity, the whole activity is almost instantaneous.

It's not one of those programs where you'll stop often for tea breaks while the program churns and mopes along.

The latest PAGEPRO also has some great improvements, such as cataloging from any cursor. For me, this was a... LARGE improvement.

The columnizer that is part of PAGEPRO has been improved to auto indent and auto page number; two nice features. d a f c
Y s i i t o o
o t n r r n
u i e a v
l a c n r e
m l n t y e n
a y i a i
y t o t l e
y n i n
m c
e e e

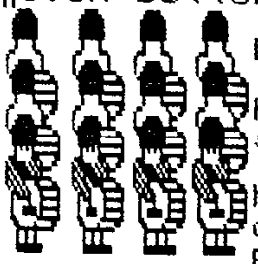




RATING

Typing in any direction let's you make quick borders, boxes, whatever, as well as puzzles and text patterns.

There are other neato changes in 1.5 regarding importing/exporting text, printing a page, and so on. But, basically, a great program was made even better.



So while the band's playing let me add that PAGEPRO does not stand alone.

Asgard also has PAGEPRO PICS of all kinds of pictures done in neat, thematic packages.

There is also PIXPRO which converts GRAPHX, TI ARTIST PIX & INSTANCES, RLE, PICASSO, MACPAINT, & PAGEPRO into most of the above formats. This is great even if you don't own PAGEPRO.

Also available are PP FONTS (a few of the 50 are shown on these two pages); PP TITLES (which are works of art unto themselves); PP UTILITIES (which allow some extraordinary manipulations of the graphics for all kinds of on-screen pasteups & designs.

PAGEPRO is a WYSIWYG ("wizzywig": What You See Is What You Get) program that more than lives up to its promise. It's the standard for TI as TI-ARTIST is for drawing.

(to be continued)

JUST ABOUT EVERYWHERE

Tony McGovern

This is the second instalment of the assembly tutorial on writing code to execute at addresses other than those where it was originally loaded and/or linked. We now look at how to write code that inherently does not care where it is placed. This is known as position independent code, or sometimes as PC (program counter) relative code in processors that have this as an explicit addressing mode.

The essential requirement for such code is that it cannot contain any references to absolute addresses within the code block itself, because by the very requirements it is not to be reliably found at any given address block. The only absolute address references allowable are to fixed system addresses or addresses elsewhere in the program that are established at load time. As we saw last time pure register operations or immediate addressing are fine because they are workspace pointer relative. So here is a simple piece of code that will execute snywhere on the left, and on the right a version that will not.

```
*  Position independent
START   EQU   $
        LI    R0,>100
START1  MOV   *R1+,*R2+
        DECT  R0
        JGT   START1
        RT
```

```
*  Not position independent

START   EQU   $
        MOV   @LEN,R0
START1  MOV   *R1+,*R2+
        DECT  R0
        JGT   START1
        RT
LEN     DATA  >100
```

The code on the left contains no explicit addresses in the block of code itself. It presumes, as does the other side, that the contents of R1 and R2 have already been set up. The LI instruction reads a data word immediately following. Here it is obviously intended as count data. In other circumstances (say if R1 were loaded with a pointer to data in this relocatable code block) such data words cannot be a pointer in the position independent block which is fixed by either the assembler or the loader. On the right hand side however the code is NOT position independent because the label LEN written in to the first instruction by the assembler and/or loader does not shift with the code. Data location LEN would have to live at fixed address somewhere and not part of the block for the code to be position independent.

Full word addresses local to the position independent block have to be adjusted on the fly. The essential requirement is that the code block must be able to determine its current address to have any chance of adjusting full word address references. Here we come to one of the most amazing omissions from the 9900 instruction set. It already has a STWP instruction so that a program can save the current Workspace Pointer and so find where its register set is located, and a STST for preserving the current processor Status Register contents of status bits and interrupt mask. That makes 2 of 3 of the registers that define the processor state. Then why on earth did TI not go the whole hog and have a "STPC" instruction to give the programmer complete control? It is in fact possible to work around this difficulty by a programming trick. Remember that BL, Branch and Link, loads R11 with the address of the

following instruction when it takes the branch. So do a BL to a dummy subroutine when the position independent block is entered and preserve the contents of R11. Remember that this BL itself must be a position independent call.

```
*  Method 1 to fake up "STPC"
PIENTR EQU  $
       BL   @RTADR
PCBASE EQU  $
       MOV  R11,R9
       ..

*  RTADR is address of an RT
*  R9 now contains PCBASE
```

```
*  Method 2 for "STPC" (better)
PIENTR EQU  $
       LI   R9,>045B
       BL   R9
PCBASE EQU  $
       MOV  R11,R9
       ..

*  Also here in R9
```

Either of these does the job. This first presumes a reliable fixed or relocated address is known for an RT instruction. This may not always be available, or the position independent routine itself is all the code there is. In this, the usual case, the second method must be used. Here we construct an RT instruction in a register and do a BL to the register. Writers of books on structured coding would shake their heads in horror at the thought of dynamically creating a code sequence in the registers and executing it there. Then you bring out the "X" instruction !! In fact it is a perfectly valid technique and a nifty solution to problems in bank-switching of memory. Actually the machine code produced by high level language compilers such as C or Pascal does far more incomprehensible things than this. Use of a high level compiled language is well recognised as a good way to make disassembly of binary files very difficult to follow. Assemblers for most microprocessors would balk at an instruction like this, but remember that in the 9900 family register addressing is just a short form of memory addressing relative to the workspace pointer. You could do a version with BLWP too, but it is not worth the extra trouble.

So how to use this ? Indexed addressing mode combined with careful setting up of all addresses as offsets from the PCBASE pointer is the solution.

```
*  Demo example for Position Independent code

START  EQU  $
       MOV  R11,R10          Save return
       LI   R9,>045B         Load R9 with RT instruction
       BL   R9               Branch and Link to it
PCBASE MOV  R11,R9           Use R9 to hold offset base
       BL   @SUBR1X(R9)      Call with index from R9
       ..
       ;;
       B    *R10             Return

SUBR1  EQU  $
       MOV  @FLAG1X(R9),@FLAG2X(R9) Indexed to make P.I.
       ..
       ..
       RT

FLAG1  DATA >0               Data/flag item
FLAG2  DATA >0               Data/flag item

*  Data, subroutine pointers as offsets from PCBASE
*  Must follow code to avoid Illegal Forward References
```

```
SUBR1X EQU  SUBR1-PCBASE
FLAG1X EQU  FLAG1-PCBASE
FLAG2X EQU  FLAG2-PCBASE
```

Use of the index register R9 containing PCBASE now converts the offsets with respect to PCBASE within the position independent code block to actual memory addresses at the time of execution of the code, and makes these references independent of the actual position of the code block, as this base reference is obtained on the fly each time the code is executed. The price paid is that one register is tied up as the relocation register, and that indexed addressing is no longer available on the dynamically relocatable addresses.

Very few complete programs on the TI-99/4a are written to be totally position independent. One such example is the UL User List utility program in the Funnelweb system. For reasons of flexibility within the system, this in later issues was rewritten in this fashion so that the same program could be used as a utility with no conflict with the central menu UL entry. If you SAVE a strictly position independent program as an E/A program file then all it takes to have it run at some other address is to change the load address in the file header. Do that on a normal E/A program file and it will usually crash.

If you have the TI technical manual look up ther DSR specifications. There it specifies a partial version of this sort of code. Obviously TI did not intend the PAD always to be at >8300, as for instance in the 9995 processor the fast memory block is on chip at a fixed address of >F000. Whatever the PAD address it was intended that the GPL workspace br at PAD+>E0 so offsets into the PAD could be derived bt a STWP when in the GPL workspace, followed by subtraction of >E0 to get the dynamic relocation index register. This restriction is responsible for some of the contortionist code found in TI DSRs, but is no longer necessary for new DSRs as in Horizon style RAMdisks as all TI-99/4a's out there use >8300.

A lesser form of position independence peculiar to the 9900 family is that necessary to write subroutines which can be executed from any workspace. We might call this "Workspace Independent Code". The usual problem which prevents this is the use of absolute addresses for the least significant bytes of workspace registers. The E/A utilities contain some well known offenders. Usually a judicious use of SWPB or STWP instructions will resolve even the worst problems at the cost of only a word of code or so.

Code fragment from the E/A module utilities

```
**  Write VDP address
*
R2LB    EQU  UTILWS+5
*
WVDPAD MOV   *R13,R2           Get VDP address
       MOVB  @R2LB,@VDPWA      Write the low byte of address
       SOC   R1,R2             Properly adjust VDP write bit
       MOVB  R2,@VDPWA         Write high byte of address
       MOV   @2(R13),R1        Get CPU RAM address
       MOV   @4(R13),R2        Get byte count
       RT                      Return to calling routine
```

R1 is either cleared or loaded with >4000 before entry to set write or read. The problem here is that the LSB of the VDP address as specified in R2 is accessed at the absolute address

of this byte. This makes the routine usable only from the UTILWS workspace. A better form of code that takes no more space is to use

```
MOVB @1(R13),@VDPWA
```

This is entirely in the spirit of the rest of the routine and in line with TI's recommendations in all their manuals for accessing register data from BLWP routines. I just don't know how such an ugly line of code got past any internal reviews of code at TI, unless the intention was to prevent outside use of this routine as its address is not made externally available by the standard mechanisms. While they were at it TI also broke another one of their system rules that you see in DSR or console ROM code - finding the VDP addresses from R13 of the GPL workspace. Again one of these rules that the orphan status of the 99/4a has rendered of no concern any more (except to the designers of the Geneve).

That about wraps up this brief introduction to some of the more advanced subtleties of 9900 programming. I hope it will prove useful as guidance for aspiring programmers out there.

Tony McGovern
Funnelweb Farm
Jul 24th /1990

SOME CURIOSITIES

Tony McGovern

Several items of curiosity value have appeared recently in the technical press. The first is of a new satellite to be launched during 1990, the Gamma Ray Observatory. This at 13 or so metric tons will be the heaviest satellite ever launched, and is intended to study short bursts of energetic radiation (gamma rays are many times more energetic than UV or soft X-rays in the usual classification of electromagnetic radiation). The reason it is of interest here is that its extensive computer systems are based on TI SPB-9989 microprocessors, a software compatible member of the 9900 family that powers the TI-99/4a. It is a bit of a cross between a 9995 and a 99000 and if I have the right one in mind is built in bipolar I-squared-L technology. TI-99 assembly programmers would easily recognize what lay behind the general descriptions given of how the satellite systems operate.

A few weeks ago an obscure IEEE journal arrived. I never did order it and I think it just comes as a consequence of some others I do subscribe to. There on the front page contents was PARSEC - Process Analysis etc etc to complete the acronym. I haven't seen that up and running for several years now, but the moduke was a major point in the history of the TI-99/4a. Now a name like that leaps to the eye so I had a look inside. Sure enough the article was written by a bunch of engineers at the TI wafer fab facility in Lubbock. Must be some tribal memories there though none of the biographies admitted to have ever worked on the Home Computer.

Not so directly TI related but still one for the curious, was on the Inmos Transputer. You may have heard of this device as a very fast micro especially well adapted to use in multiple sets for parallel processing. with communication between processors over a very fast single bit bus. This device has not really made anywhere near the impact it should have because of high prices and the idiosyncratic policies of its British makers on software. As I understood it they wouldn't even let anyone know what its assembly language was like, and insisted on the use of the special Occam language. This has its virtues, but was incompatible with anything else that programmers ever used, so generally they passed it by, and now mainstream RISC developments may well be doing so to the chips too. Anyhow the device was recently taken over by SGS-Thompson who are much more into mass marketing, prices are coming down to more reasonable levels, and what is more articles have started to appear on its internal arcnitecture and assembly language. The designers explicitly said that the two initial inspirations were the HP calculator stack and the TMS-9900 workspace pointer concept. So there is a 32-bit screamer out there that develops a idea familiar to TI-99 assembly programmers. It really is a pity that TI got stuck in the 99000 dead-end that never seemed to go anywhere.

Yet another item spotted recently concerned the spiritual ancestor of the TI-99/4a, namely the 990 series minicomputers with their DX10/DNOS operating system. When you reflect on it, the minicomputer provenance is what has given the TI-99 expanded system its remarkable staying power. The TMS-9900 processor was the 990 series architecture reduced to a single chip, and used in the low end of the 990 series. Other micros were based on grown-up calculator chips, and the influence is still felt in all those PCs based on Intel's flawed 8086-80286 architecture. Anyhow the sad news was that TI has just (Jun 1990) discontinued production of the 990 series, though support will continue to 1995. Sound familiar ? In this case the life cycle has pretty well been run through. All told 120,000 of various 990 minis were produced starting in 1971 and it is estimated that 5000 are still in active service. Can't say I have ever seen one, though I have seen plenty of PDP-11s and used DG Novas, which is the league the 990 was playing in. I suspect some of the earliest assembly programmers for the 99/4 learned their trade and even did some of their work on 990 minis. If you look at the opcode table in the 99/4 assembler, say by using <V>iew in DiskReview, you will see all sorts of things that do not exist on the 9900 including memory mapper instructions for the larger minis. This news finally does mark the passing of an era.

The Making Of a Quest

As Joe Wright says "You caught me with my tweeds down" when Ron Kleinschafer presented me with the Quest RD200 board at the meeting. I really had expected to purchase a couple of cards for the ol' TI while I was in Oz but never expected a present of that magnitude and unlimited utility. It's strange how the mind works, though. The meeting was less than half over and my mind was busy scheming how to come up with the requisite 17 chips to get it up and running. I had allowed a little extra in the budget and a couple of good deals on souvenirs in Queensland made it possible for me to have just enough left to send off an order when I got home.
Thursday and a parcel arrived with my 18 chips (1 spare) for the Quest. Now, it was time to start!!! I carefully laid out a work area, got my chip inserter/extractor set and started out. The first thing that I found out was that the pins on the chips were just a wee bit wider than the sockets. Jiggling and prodding the chips did little to solve the

problem -- they still wouldn't fit in the sockets. As much as I hated to, I decided that I was going to have to GENTLY bend the pins on one side of each chip to get them into the sockets. The smooth wooden desk top did an admirable job. This was going to be easy!! They popped in one after the other with the chip inserter --what a snap!!! All 17 were in place and the card was carefully inserted into the PE-Box. So far, everything was going well!

The DSR file ROK was loaded into the Quest and it was time to start testing in preparation to loading all of the "goodies". MURPHY'S LAW STRUCK!! The number of available sectors didn't come out right. Next step (said the docs) run the memory test. What a bloody disaster!! Error messages all over the place --Error in RAM 1, Error in RAM 3, Error in RAM 17, etc. Amazing this was, indeed! A lot of bad chips? Not that many, surely!

Back to the drawing board. I read the docs while the electricity in the PE-Box dissipated. Aha! There the answer was -- check the SOCKETS AND THE CHIPS. On close inspection, some of them did look a bit funny. I bet they weren't seated in the sockets properly. I put the "ultimate chip inserter" into action -- my fingers. After making sure that there was no static electricity about, I started seating all of the chips by hand. Indeed, some of them were a bit loose. They popped farther into the sockets when some pressure was applied. I even found one with a "gimpy" pin. That chip was extracted and the pin straightened with a pair of pliers and then it was re-inserted.

Ok, time to put the card in the PE-Box and give it another try. Once again, the DSR was loaded and this time the full 2048 sectors were found. Ready to start with the "goodies" at last!! I decided to partition the Quest into two drives numbered 4 and 5. Drive #4 was to be my working drive and #5 was going to be for BBS downloads, file storage, etc. Drive 4 was configured to the maximum 1600 sectors and drive 5 got what was left.

F'WEB was completely re-configured to load and run off drive #4 and I took a sector editor to SPELLCHECK and made the necessary alterations to it as well. Some of the D/F 80 files were converted with the RAG LINKER to program image and everything was loaded on three back-up disks. The whole lot was summarily dumped into drive #4. All of the load paths were checked and AUTO was enabled.

Time to try this sucker out!! TELCO was first up. TELCO is constructed of overlays and moving the appropriate code in an out from a physical disk drive can be a bit tedious at best. Up on the BBS for some hacking around and a couple of downloads. The usual efficiency on a download is between 75% and 85% of the maximum 1200 baud rate. The main reason is that TELCO saves the downloaded files to disk in 8K chunks and it takes time to dump them to a physical disk drive. Let's see what it does when it is downloaded to DSK5 (QUEST/B). After a couple of downloads of varying length, I found that the efficiency rating never went below 90% -- and that was with a TIMEOUT error and the re-transmission of a 1K chunk of data. Usually, I was pushing 95% !!!

Next on the agenda was the altered version of SPELLCHECK. The first file loaded so fast that I actually loaded the second file and was waiting for the file input before I knew what happened. A heavy finger on the <ENTER> key had taken me that far. You should have seen it check a document!! It went so fast that I thought the system was locked up the first time I ran it! Amazing!!!

This was going to present a problem! No more quick trips to the fridge while a long text file was being checked. I would have to get my "munchies" before I started on the computer!

April 3rd and time for our monthly meeting. I carried my PE-Box (and Quest) to the meeting for a surprise demo. You should have seen the expressions on the UG members faces when Quest started "strutting its stuff". One of the members even claimed that Quest appeared to be much faster than his Horizon!!

The only drawback that I have been able to find with Quest is this: I keep counting the empty slots in the PE-Box and figuring how many hours of extra work it would take for . . . What is the price of another one of those cards, anyway?

RANDOM BYTES

By Bob Carmany

This is the second in the series of Quest-related articles that will appear here. This one deals with the installation of Forth as a AUTO menu option. I started with my preferred Forth 'dialect' ---Wycove Forth.

Wycove Forth is probably one of the easiest additions to make to the AUTO menu. First of all, since the Wycove kernel is written as straight A/L program image files, it loads quite easily and quickly from AUTO. There was never the slightest problem getting it to load. I used version 3.0 but any of the other previous versions will do quite nicely.

Shifting the screen access to a quest is another matter entirely! Rather than hack about in the kernel trying to find the delimiter that limits access to DSK2, I decided to take the easy way out. Wycove has the facility to create a 'turnkey' system. That is, to create a system that is complete at START with all of the options that you want or need. That is just what I have done!

I added a good bit of the utility options to the basic Wycove system including speech and sound access, printer control, and a bunch of others. The end result is that I have 3 A/L program image files named FORTH, FORTI, and FORTJ instead of the usual two files. With Wycove structured the way it is with all but 1K of the kernel in low memory, it leaves more than ample programming space for applications.

I have started a re-write of the Wycove kernel to overcome the lack of access to DSK4 and DSK5 when loading application screens but it looks to be a long and drawn out effort without the source code. The 'turnkey' system seems to be the best solution as long as space is not a consideration.

The fact is, that even with all of the options loaded that I need, I still have almost 16K of memory left for programming space. I can't conceive of an application that would take up that much memory once the definitions are compiled into the dictionary.

TI-FORTH is another "beast" entirely! After experimenting with various and devious means to load it from the Quest menu, I can to the conclusion that the easiest way was to use the XB loader that I got some years ago from MICROpendium. A wee bit of changes as far as load drive number and TI-FORTH loaded very nicely indeed. Even without the extra LOAD program, it is still noticeably slower than Wycove when it boots. I haven't gotten so far as to alter the screen access code but I intend to do so before the summer is out.

What it boils down to is that both dialects of FORTH can be loaded as menu options with Quest. If the screen access is speeded up proportionately as much as the initial load, both should be astonishing when they run out of a RAMdisk. The only obstacle at this point is writing some code to access whichever drive you choose to store your FORTH screens. I like the idea of a short series of articles about Quest so much that I am going to continue it in the column for next month. At that time, we will explore the possibility of a pseudo HARD drive. We will look at creating a tree directory and structuring with the idea of both spped and ease of use. The column next month will be the last dedicated column with Quest as the only topic. You can rest assured that there will be mention of it from time to time, though. Since I now have a Quest, I'm sure that the mention will be fairly regular.

Well, the page is getting shorter and shorter and a 'Buffer Full' is once again on the immediate horizon. Any suggestions for future topics for 'Random Bytes' will be gratefully accepted. . .

The next couple of months are going to be a not-so-random byte column. Ever since I got the memory chips for the Quest that you so graciously gave to me I have been altering various programs to install into it. While I was visiting at Al Lawrence's, he mentioned a version of SPELLCHECK that he had that had been altered for Quest. Ever absent-minded, I neglected to get a copy before I left. That meant that one of the first tasks that I would have would be to alter my copy to work on the Quest.

Actually, the process wasn't as difficult as I had imagined. I had already modified it to run as a TI-Writer side option from F'WEB. One of the first alterations that I made was the one suggested in the F'WEB docs. Using DISK UTILITIES, a string search was initiated for the words 0460 0070. As per the docs, it was changed to 0420 0000. That allowed for an exit to the TI title screen instead of a crash when the program ended. With AUTO on, an easy re/entry to Quest was now possible.

I had previously installed SPELLCHECK in F'WEB simply by renaming its UTIL1 file to the two letter file designation in the slot in which I wanted it to reside. In my case, I renamed it DU. No problems so far! Now for the more difficult (?) part. I have my Quest divided into two drives ---DSK4 and DSK5. DSK4 (1536 sectors) contains all of my F'WEB options that I use frequently and is almost full. In fact, it is so full that the SPELLCHECK dictionaries will not fit. DSK5 has about 512 sectors free and it became the logical choice for the dictionaries. That is the background. Now for the particulars.

Luckily, the SPELLCHECK program is an easy one to alter. The second file load (ie. UTIL2) is hard-coded as is the dictionary load. Using DISK UTILITIES, do an ASCII string search in the first file of the series DU (formerly UTIL1) for the characters DSK. You will see "DSK1.UTIL2". Simply change it to the appropriate drive number ---in my case it became DSK4.UTIL2. Now, both of the files will load from the first half of my partitioned Quest. The second alteration (for the dictionaries) is just as easy. Just do an ASCII string search in the second file ---UTIL2--- again for the string DSK. You will see "DSK1.DICT1". All you have to do is change it to the appropriate drive number. In my case, it became DSK5.DICT1. The program automatically loads the second portion of the dictionary from the same disk drive so your task is completed once you have saved your altered files back to disk.

The biggest advantage in having SPELLCHECK on RAMdisk is the speed of the program. It now loads much faster and the execution and checking of a text file is also much faster than the original. I use the second part of my Quest as a cache for the temporary storage of text files, RBS downloads, etc. The result is that I copy the dictionary files into it at the start of a session (like this column) and then they are available for use whenever I need them. I might keep them there for a week or so until some other overriding need forces me to delete them to make room for something else. I have found it to be an excellent alternative to waiting for the physical drives to churn away while SPELLCHECK goes through another document.

Well, its getting toward the end of a page and I try to keep things as uncomplicated as possible for Joe. I'll have to think a bit about what to write for next month's column --- undoubtedly something to do with Quest or maybe some A/L since Ron K was nice enough to persuade me to try my hand at programming in it. At least my efforts will give he and Tony a few laughs but they had to start somewhere once as well.

Suggestions for this column are still (and forever) being accepted.

Buffer full . . . 'Til next month.

DATA COMMUNICATION

by Geoff Phillips

This article provides an update on data communications over the telephone network between digital computers, in particular TI-99/4A systems. While some readers may be fully familiar with the use of modems with TI computers, many club members may not be aware of the special knowledge picked up in the process of setting up communications for the first time. Some of my experiences are described here.

The requirements to start are a TI computer with RS232 interface and disk drive, software and a modem. Various varieties of modem can be acquired. The first one I used was a MicroBee manual modem which could operate at either 300 baud, or at 1200/75 baud (baud meaning bits/second - basically a speed rating). This modem was borrowed from a neighbour who had better toys to play with. The software package I used was TELCO available through TI clubs. TELCO runs from an extended basic loader, and runs most conveniently from double sided disks, although a cutdown version can be configured for a single sided single density disk.

The modem needs to be connected by an appropriate cable to the TI RS232 port, which consists of port 1:

PIN	MNEMONIC	I/O	FUNCTION
1		Ground	Protective ground
2	RD	Input	Serial data in
3	TX	Output	Serial data out
5	CTS	Output	Clear to send
6	DSR	Output	Data set ready
7		Ground	Logic or signal ground
8	DCD	Output	Data carrier detect
20	DTR	Input	Data terminal ready

and the less commonly used port 2:

PIN	MNEMONIC	I/O	FUNCTION
1		Ground	Protective ground
14	RD	Input	Serial data in
16	TX	Output	Serial data out
13	CTS	Output	Clear to send
6	DSR	Output	Data set ready
7		Ground	Logic or signal ground
12	DCD	Output	Data carrier detect
19	DTR	Input	Data terminal ready

The cable to connect a modem to port 1 would typically be a direct connection on pins 1,2,3,5,6,7,8 and 20. This contrasts with the pin wiring for connecting two TI computers by direct cable (for example using Terminal Emulator II on both machines) which is:
1-1, 2-3, 3-2, 6-20 and 20-6.

In practice, a common problem is that 2 and 3 are straight through when they should be crossed over, or they are crossed over when they should be straight through!! Further information on the pin connections is provided by the TI RS232 Interface Card Manual.

Having assembled the computer, cable and modem, the novice communicator needs to test his system. The easiest way to do this is to call a bulletin board. The Bulletin Boards in the Newcastle (049) area are:

BULLETIN BOARD	PHONE	BAUD RATES
Newcastle Microcomputer Club	685289	300-2400
Bill's Bulletin Board	602121	300-2400
Inquestor	683100	1200-2400
Communication 2000	592667	300-2400
Cybertron	602383	300-2400
Hunter Schools	692851	300-2400
Lake Macquarie	754120	300-2400
Local	621768	300-2400
Mega Technology	587099	300-2400
" "	587350	300-2400
" "	616803	300-2400
MIDI	563100	300-2400
Small Business	502411	300-2400

But before you call a bulletin board, start the communication program up on your TI computer. Some initial settings are required, and TELCO helps you organise these. The following should be set for TI communications with local bulletin boards:

PARITY	: NONE
DATA BITS	: 8
STOP BITS	: 1
DUPLEX	: FULL
TERMINAL TYPE	: ANSI (ADM3A also works on NMC and Bills)
BAUD RATE	: as required, for example 300

An interesting point here is that Terminal Emulator II will not work with the bulletin boards because of the fixed setting of 7 data bits in the terminal mode. An option of 8 data bits is available for file transfers only.

The baud rate of 1200/75 (1200 in one direction and 75 in the other) is not supported by the TI RS232 card, and so is not useful for bulletin boards. However, there is more to say about 1200/75, but that will have to wait for another time. To return to the main topic, calling a bulletin board from a manual modem is done as follows:

Set the Talk/Data switch to Talk and dial the number. Is it engaged? Bulletin boards are popular at the most convenient times of the day, but most of them run twenty-four hours a day. Try another number until the number rings, now you can switch back to "Data" and hang up (on the "Data" option the phone will not be disconnected when you put down the receiver.

The Bulletin Board now takes the initiative and you can simply respond to the prompts and menus presented. Your communications experiment is a success!! One final point is that when you exit from the bulletin board, remember to switch the phone back to "Talk". Otherwise you will remain connected to the bulletin board by phone, and you may be listed as a "Zapped user" on the board, after you are finally disconnected by the Bulletin Board program.

There are some points of interest about the bulletin boards in the Newcastle area. The Inquestor bulletin board has a TI 99/4A "file area" and a TI 99/4A "message area". Bill's bulletin board accepts games written by users and is recommended for interest. The Newcastle Microcomputer Club bulletin board lists Meetings on subjects of computing interest, these are generally held on alternate Monday evenings at Newcastle University.

There is a lot more to talk about - a future article will deal with the use of the more modern "smart" and "automatic" modems, point-to-point communication and other items of interest.

FILLER #2

WHAT THE IWP NOTED **

If you're interested in meaningful and worthwhile subject matter, skip this lot. However, if you are disposed toward fruitless pasttimes such as viewing TV quiz shows and the like, you may find something of interest in this lot.

The scene. A restaurant notorious for its inquisitive waitpersons (IWPs).

The protagonists. A taxidermist, a taxi driver, a sales person, a plumber, a doctor, a nurse, an accountant and a manager.

As it is an informal gathering, let's introduce Amanda. Bill. Bruce. Charlene. Harry, Joyce, Libby and Norm.

After they're seated, they order a round of drinky-poos, viz. beer, gin, scotch, lite beer, lemon squash, moselle, rum with orange, and vodka.

The IWP who attended their table during the evening noted that:

The moselle drinker was seated between Charlene and the taxi driver.

The nurse was opposite Norm.

Bill was opposite the gin drinker.

The lite beer drinker was opposite the accountant, who was seated next to the manager.

Libby was opposite the plumber.

The taxidermist was seated next to the nurse.

Joyce was opposite Harry.

The doctor was opposite the moselle drinker.

The plumber was seated next to the vodka drinker.

The rum and orange drinker and the taxidermist were opposite each other.

Norm sat next to Charlene.

Harry sat next to the doctor.

The nurse was next to the scotch drinker who was on the left of the manager.

The beer drinker was opposite the nurse.

The gin drinker was opposite the manager.

Charlene was seated between the beer drinker and the manager.

** By noted I mean that he jotted it all down on paper and passed it along to me (how else would I have acquired such intriguing information?) knowing my predilection for passing on such paltry pap for publication.

Next evening was 'off' for the IWP. So the IWP and the IWP's nearest and dearest endured a nightly brain damage session of TV, after which the IWP produced the jottings of the previous evening and posed the query - who's the teetotaller?

After 4 hours of evening TV this was too much for the N&D. But there may be perspicacious readers (hardly likely to have bothered reading this - but then - one never knows) who may care to hazard a guess.

In case any PRs lose interest/can't be bothered with such trivial trash (ie too hard), there'll be an answer next month. I won't have to work this one out as I have the GG from the IWP.

See ya later still.

SOFTWARE LIBRARY.

The next two artilces are documentation taken from software recently arrived in the library, "Newsletter Editor", a two column printing programme used to print the docs below. Space an arcade like game for the kids. The third article is the documentation from Ron Klienschafer's RD200 utility.

NEWSLETTER EDITOR AND FORMATTER

OVERVIEW:

The Newsletter Editor and Formatter is, as one might suspect, designed to help create newsletters. Unlike most other column format programs, NEF (my name for the program), will allow you to produce your entire newsletter in one program.

While the program is rather generic in that it is not feature-packed and it does not provide fancy fonts or the like, it does produce a nice two column page complete with a double wide title for the top of the page and boldface print for paragraph headings. For example, when you get around to printing the documentation you will discover that the OVERVIEW heading on the first paragraph is in boldfaced print. Similarly, other paragraph headings are in boldfaced print too.

NEF allows up to 114 lines of text to be entered into a single file. That is the number of lines that will produce a single page of text that's printed in two-column format. Each line may contain 36 characters maximum length.

To produce a newsletter, you may type in the text for each page free-hand, or it may be imported from a TI-Writer file. NEF saves all documents in DV/80 format, just as TI-Writer does.

When all text for each page has been entered into a file it may be printed with ragged right margins, or you may "run it through" the justification process prior to printing to enhance the appearance of the printed page.

Justification is selective in that it allows you to select only those lines that you want to justify. For example, one would not normally justify the last line in a paragraph. Lines that have only one word in them, or blank lines are ignored by the right-justification routine.

Pages may be numbered at the bottom of the page during the printing process. You will be prompted for the number to print and it will appear in the middle of the page.

PROGRAM OPERATION:

NEF uses a command mode and a text mode to provide all text and file processing features. The text mode is active when the cursor is flashing. The command mode is active when the cursor is not visible on the screen. Fctn X is used to toggle between the two modes.

Text mode is used to enter text that is to be saved or printed. The command mode is used to access any of the commands listed in the menu at the base of the screen. Commands are accessed by pressing the first letter of the command. For example, one would press H to read the Help screen.

NEF provides six text screens that are capable of holding the 114 text lines available in one file. Paging from one screen to another is done by the program, based upon cursor position, and is thus automatic.

Cursor movement is accomplished with the arrow keys and the ENTER key. Fctn E takes the cursor back one line at a time, while ENTER advances the cursor one line at a time. Fctn S and Fctn D move the cursor horizontally within a text line.

Command menu options include;

-Clear the current screen, -Delete a line of text, -End of file access with one key, -File name and path display, -Help screen access, -Justification of text, -Insert a blank line, -Load a file, -Output or input path change, -Print a file, -Quit the program, -Save a file and, -Top of file access with one key.

SECTOR PATROL must be unpacked by an Archiver program before it can be used. The program when uncompressed consists of the following program files:
END
GAME
LOAD
MAIN
READ_ME (this abstract)
RESTART
SP
You will need the following to run the program:
EXTENDED BASIC, SPEECH SYNTHESIZER, 32K, JOYSTICKS and at least 1 DISK DRIVE. You will also need at least 16 free sectors on your disk for the files the program will create.
This is a fairware program - payment information is in the program.

The object of the game is to rid your assigned sections of the galaxy of the dreaded robot ships. Planning is the key, not speed. Full on screen instructions, high resolution color graphics, speech, and sound are just some of the features.
THIS IS PROGRAMMED IN EXTENDED BASIC - if you are expecting arcade speed, DO NOT download this!
Unpacked, this package will need 200 sectors on a disk.

Enter EXTENDED BASIC and type RUN "DSK1.GAME". The game disk must be in drive number 1 and the disk must not contain write protection tabs as access is need for the files that are created.

QUEST RD200 RAMDISK UTILITY.

May 1990.

This document and the files on this Disk are an update from the versions dated 1st July 1989.

This update is fully compatable with existing DSR versions AA07 and does not require any reformating or reconfiguring of the DSR. The changes made are additions from suggestions of users of the Quest, along with some bugs removed and some internal and display cosmetic changes made. Read filename UPDATE.

The Program QUEST along with the DSR (filename RQK) on this Disk is to enable the Quest to be used to its maximum capacity, until now the DSR used was a temporary affair until work could proceed to get the Disk working as intended by its designer Neil Quigg, these files do just that.

QUEST is the utility for configuring the Quest as the user wishes, the Quest can be operated with from one to sixteen 32K chips (128 sectors to 2048 sectors), the latter the maximum capacity of 500K, no Ramdisk memory is used by the Quest for the DSR due to it having its own 8K memory chip for that purpose.

Some of the routines used in QUEST are from the source files for Q/O by Tony McGovern and have been modified for use in this Program, the excellent word inverting memory test routine has been retained in its original form, if the user gets ONE error report about a memory fault take it from personal experience that a fault DOES exist even if subsequent tests show OK, the offending IC should be replaced, or at least check the IC's Socket.

The DSR has been written to conform with the requirements of the Quest, all normal Disk Drive functions are available, except that the DSR does NOT support formatting so the only way to format the Quest is to use

the utility QUEST, the DSR RQK is NOT compatable with any other Quest DSR loader utility, the filename for the DSR has been altered from RQS to reflect the changes.

After loading the DSR the Quest can be formatted to either a single drive or can be used as a split drive (two drives). The size of the first Drive is limited to 1600 sectors maximum, If a Disk drive size is extended beyond 1600 sectors (12 32K memory IC's) problems would arise with other software. The user can divide the Quest into any number of sectors in each Drive as required within the limits allowed.

Basic CALLs are allowed. The Quest can have a POWER UP menu Program loaded at switch on, bypassing the Title Screen, the Basic CALLs for this are, CALL AON to turn Auto power up on, and CALL AOF to turn Auto power up off. The Program AUTO and AUTP, on this Disk, MUST be present on EITHER section of the FIRST Quest in the CRU address chain for Quests, IE: if there are multiple Quests and the first Quest is at CRU address >1000 then AUTO and AUTP should be on this Quest.

NINE Basic CALLs are valid to load Assembly Language Programs, the Programs can be CALLed from Basic (or Extended Basic) or with any Module installed, the Programs can be CALLed even without a Module but the action of the loaded Program will depend on its requirements, some look at which Module is present and acts accordingly, if the Quest is formatted into 2 Drives the CALL will search both drives and load and execute the CALLed Program from the Drive it is located on. EIGHT of the program names can be edited to suit the user, the other option of AUTO is fixed and cannot be edited.

The other CALLs are to write protect or unprotect each of the two drives in each Quest (if it is formatted into two drives), these calls are from basic and are CALL WOx (x = the number given the drive) to turn on write protection for that drive, and CALL WFx (x = the drive number given the drive) to turn off write protection, the format for the CALLs are a three byte call as above, this should prevent conflict with an Horizon Ramdisk if the two are in the same PE box, EG: if there is a Quest with the drive number 4 then to turn on protection simply enter basic and type CALL WO4, to turn off protection type CALL WF4, these calls are valid for ANY Quest either single or dual, which means that with a single Quest in the PE box that has been split into two drives then EITHER or BOTH sections can be protected as required, the call names are the same as used by the Horizon to make it easier to remember except for the three byte format for the calls.

When QUEST is loaded it first checks the system and finds any or all Quests that may be present, ignoring any other Ramdisk, even if the Quests have not been initialised, when found it then checks to see if the Quest has the DSR RQK installed, if multiple Quests are available the user will be prompted for the Quest to be accessed, after this the Program locates and stores the DSR information (if any), if the Ramdisk does not have this DSR (RQK) loaded then flags are set to enable the user to install the DSR and then configure the Quest as they wish, the Program checks the Quest for the TOTAL number of sectors available and displays that information along with a menu screen, it has the following options.

1. Load DSR
2. Save DSR
3. Reconfigure DSR
4. Reformat Drive/s
5. Test Ramdisk Memory
6. Next Quest
7. Exit

1. Load DSR.
Unless the DSR RQK has been previously loaded the options 2 to 4 will

not be available untill the DSR is installed with LOAD DSR, just follow the screen prompts (an example is below).

2. Save DSR.
The save DSR Option is as the option implies, to save the configured DSR back to Disk to enable quick loading if the Quest DSR becomes corrupt, if multiple Quests are being used give the SAVED files, names to suit, example RQ1, RQ2, RQ3, at any time if changes are made to the DSR then the DSR should be reSAVED back to Disk to make subsequent reloading valid.

3. Reconfigure DSR.
This option allows the user to reconfigure the Drive number/s, the users own prefered (configured) program colors and to rename the seven available Basic CALLed program names, the name lengths are limited to 5 characters in length.

4. Reformat Drive/s.
Once the DSR has been loaded, configured and saved then the Program allows the user to Format either one of the drives without affecting the other drive in the Quest, this option will appear automatically (if the Quest has been split) also the Disknames will be displayed to avoid confusion as to which Drive is to be reformatted, or if the user wishes the whole Quest can be reformatted. This formatting will be same as the Quest had been previously configured to but the user can change the sizes of the drives as required.

5. Test RamDisk Memory.
This test is usefull for checking ALL of the Quests memory, it reports any faults and where, it is NOT destructive and can be used at any time.

6. Next Quest.
If multiple Quests are in the PE box then the user can select the Next Quest to be reconfigured, tested, reloaded etc.

7. Exit.
This exit first checks if Auto Power up is on, if so then the return is to the AUTO screen, if Auto is not on, the Program checks if a QED Module is present and loaded with either the PAGE1x or PAGE2x files, if so then the return is to the Grom Menu header otherwise the standard GFL return is invoked.

EXAMPLE ONLY:
An example of a * FIRST * time setup for a Quest, with say 16 32K chips installed could go something like this.(the Program handles Quests with ANY number of 32K chips).

Load the Program QUEST with Option 5 E/A or Option 3 F'WEB, once loaded the Program checks and displays the number of Quest Ramdisks available, if only one Quest is in the PE box it then automatically checks the size available and will display :- TOTAL SECTORS AVAILABLE - 2048, if the total sectors shown are below that expected do a MEMORY TEST, there may be a faulty RAM or socket.

NOTE: The slow count of available sectors is done only on the FIRST time load of the Program and DSR, after configuring the Quest the total available sectors is displayed immediatly. After the initial loading of the DSR, Configuring and Saving of the DSR, IF AT ANY TIME SUBSEQUENT loads of QUEST displays the incorrect number of sectors available the Saved DSR should be reloaded from Disk. After the first load, QUEST gets the number of sectors available from the DSR to speed up the Program and if that section of the DSR has been corrupted the data will be incorrect, after reloading the DSR just do a 'Next Quest' and the data will be OK, also there may be an error of NO MEMORY FOUND, if so just press REDO and the program will do a recount of the available sectors.

Press 1. LOAD DSR, at the screen prompt enter the Disk Drive number containing the file RQK and follow the rest of the prompts, they are self explanatory.
When the DSR is loaded the Program then jumps to the format screen and prompts for either a [S]ingle or [D]ual drives, the default is Dual so just press enter, then you are asked for the number of sectors in drive A, although the Quest can be divided as the user wishes for this example we will divide the Quest into two equal sections so enter 1024, there will be a warning message displayed then formatting takes place.

If a Dual Drive is not required then at the screen prompt for [S]ingle or [D]ual just press "S", the one constraint on this is that if the Quest has more than 1600 sectors available in Ram it will not be formatted larger than 1600 sectors. If "D" is selected then the user is prompted for the number of sectors required in the first section of the Quest, an entry of more than 1600 sectors will return an error, if an entry is so low so as to make the second section larger than 1600 sectors another error will be returned, with a full card of 16 32K IC's the minimum for the first section would be 448 sectors and the maximum would be 1600 sectors, after entry of the required data for the first section the second section is automatically calculated and the Quest is formatted at that.

One requirement during entry by the user for the number of sectors required for the first section of the Quest is that if the number of sectors is below 1000 then the entry must a 4 digit figure, EG: if a first section of say 800 sectors is desired then enter 0800.

When this has been finished the Program then jumps to the configure screen and prompts for the Drive/s numbers, for the drive number for drive A, enter 4 (or what the user wishes), then for drive B enter 5 (or whatever, between 1 to 9), naturally if you have mechanical drives then you cannot use those numbers.

The next edit required is the users prefered colors for the Programs that are to be loaded, this edit does NOT change the colors already in the Programs, but merely serves to automatically change the colors of QUEST and AUTO, the colors selected MUST be already configured in the Program to be loaded, IE: if you have configured Funnelweb to White screen with Blue text then QUEST and AUTO will follow suit.

The third edit required is the names of the seven available Program names that can be CALLed from Basic or loaded with the third screen of AUTO (read AUTO/DOC), the names displayed will be PROG1 thru to PROG7, these can now be edited as required, one limit here is the name lengths can only be to a maximum of 5 characters, when satisfied press Proceed (FCTN 6) and they will be entered into the DSR, if no changes are required at each filename, press Enter.

Select option 2. SAVE DSR, follow the screen prompts but give the saved DSR a filename to suit, say RQ1. The saved DSR has all the information about the Quest and can be loaded at any time as required.

Press 6. EXIT.

A catalogue of the first Drive will show QUEST/A - USED 2 FREE 1022, similarly a catalogoue of the second Drive will show QUEST/B - USED 2 FREE 1022, the Disknames can be renamed to suit and future formatting will retain the original diskname.

After this intial setup QUEST can be loaded and ANY part of the Quest can be edited, altered, formatted as required, but ANY alterations (apart from formatting) requires the DSR to be reSAVED if the alterations are to be permanent for valid reloading of the DSR.

Ron Kleinschafer.
HV99ers.

THE INFORMATION PAGE

GENERAL MEETINGS
****

The Hunter Valley 99ers hold their monthly general meeting on the fourth Tuesday each month. The meetings begin at 7:00 pm. The venue is Jesmond Community Centre. Visitors are always welcome to attend these meetings. Each month we demonstrate new software and/or new hardware items which have become available. Discussions and presentations are also made by club members on their use of their computers. Meeting dates for the remainder of this year are;

SEPTEMBER	25 th
OCTOBER	23 rd
NOVEMBER	27 th

COMMITTEE MEETINGS
****

Committee meetings are held on the second Tuesday each month. All members are welcome to attend these meetings. The venue is changed each month because we hold the meetings at different members houses each month. If you want to attend a meeting contact Pete Smith the Monday night before the meeting regarding the location.

COMPUTER EXHIBITION
****

The Annual Newcastle Computer exhibition will be held on the 20 th -22 nd September at Newcastle University. The exhibition is in the Hunter Building concourse. The HUnter Valley 99'ers will have a display again this year. Because we all have to work the group will only have the display on Saturday the 22 nd. We won't be trying to sell computers etc. Our main purpose is to wave the TI flag for anybody who might like to join us. If you can get there, come along to the exhibition either to assist us and/or to have a look around the exhibition.

SOCIAL GET TOGETHER
****

We are trying to put together some sort of social outing for the near future. If you have any ideas please contact Pete Smith. Many suggestions are already in and include vineyard trips, a night out on the town, see a show in Sydney etc. We would like your suggestions and your participation.