march 86

HUNTER VALLEY 99'ERS NEWS

Hunter Region

TI 99/4A

HOME COMPUTER NEWSLETTER

NEWSLETTER No. 8

TEXAS INSTRUMENTS

Newcastle

& The Hunter Region

TI-99/4A

Home Computer

USERS' GROUP

DISCLAIMER

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

CONTRIBUTIONS

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

THE EDITOR
HV99 NEWS
15 GAYTON CLOSE
WARNERS BAY 2282
NEWCASTLE

General address for ALL other club related correspondence.

THE SECRETARY
HV99 USER GROUP
25 RESERVE RD.
WANGI 2267
NEWCASTLE

YOUR COMMITTEE 1986

A.WRIGHT	PRES.	PH.468120
P.COXON	SECT.	PH.751930
B.RUTHERFORD	TRE[illegible]	[illegible]H.498184
A.LAWRENCE	[illegible]	[illegible]H.486509
S.TAY[illegible]	[illegible]IT.	PH.487076
A.[illegible]	TECH.	PH.498520
T.McGOVERN	TECH.	PH.523162
B.MAC.CLURE		PH.437431
B.WOOD		PH.662307
D.WINTON		PH.591882

Positions Vacant

WHO THE HELL IS RON KLEINSCHAFER

I have often heard it said that the only advantage in being the Editor of a newsletter is that you get to be the first person to read it!!.
I am constantly amazed at the debth of talent we have in our group, none more so than last week when I received through the mail a package from Ron Kleinschafer. The contents were so exciting I immediatly sat down and formatted the article so to be sure that it would be in this issue. I know when you read it you will be just as enthusiastic as I am. A PE Box with Disk Controller and dual Disk Drives for $370.00 ......AMAZING!!!!.
Without wishing to go overboard I would have to say this is one of the more exciting projects to come from our group and is sure to have as much impact on the TI99/4A community as the 32K Matchbox System produced by the Perth Group.
Ron Kleinschafer is based in Grawin, a stones throw from Walgett, which is a couple of stones throws from Lightning Ridge which is, as we all know, isn't a stones throw from anywhere!!!!. Althoug4 isolated and working under less than ideal conditions Ron must be categorised as a true user group member, regularly in touch with our group via letters to the Editor, through which he shares freely with all members of our group his achievements in hardware and software design.
Well done Ron.

EL-PRESIDENTE"S FIRESIDE CHAT

DRIVING FORWARD.

In April 1985 T.I.99/4A Users in the Newcastle/Hunter Region of New South Wales made a painful but inevitable decision to burn some bridges and formed their own User Group. This has proven to have been significant for those Users. The flood gates to information have been opened and the flow of information into the Group has touched each and every Member.

I sincerely thank all of the T.I. Users in other areas who have been generous enough to join us in the TRUE spirit of User Group exchange. As for Groups and individuals who cannot see the necessity for this exchange, WELL! contact us when you do see the light. You can be as sure as night follows day that we WILL respond in kind.

We have genuinely tried to make the information flowing out of our Group as original and useful as our limited knowledge will allow and we make no apologies for trying.

The Executive, Committee and Members had tried to create an environment within our Group where all Members are encouraged to use their Computers to the full expanses of their individual abilities. The amount of original material which is published in our Newsletter is testimony to that policy.

The Newsletter is our primary outlet for information. Our public Domain software library is the secondary outlet for this information. The third side of the information triangle is the individual contacts made between our Members and Members of other User Groups.

For the T.I. Community to flourish all User Groups must have Committees whose foremost concerns are it's Members and their Machines. It must be understood that User Groups are about, people, machines, information and SHARING. Glossy magazines, pretty pictures and empty promises will not stand the test that the T.I. Community has faced and will continue to face in the future. If your Group is one which the above hat will fit then remember Mrs. Aquino.

With our Annual General Meeting now here I would like to take the oportunity to thank all the Members of the Group, the Committee, the Editor and fellow Executives for the support which they have given and made my period as President so easy and rewarding. I have the distinct feeling that 1986 is going to be an even better year than 1985 for the H.V.99'ers.

A WORTHY GESTURE.
At the Monthly meeting of the H.V.99'ers in February the Members of the H.V.99'ers bestowed on Tony McGovern life Membership of the Group.. This has been done in recognition of Tony's contribution to the H.V.99'ers and to the T.I.99/4A User Community in general. It had been intended to feature Tony, his Software and his Tutorials in this volume of the Newsletter. Due unforseen circumstance this could not be done and has been held over to the following Newsletter.

GOOD BYE STEVE. This Volume of our magazine will be the last produced by the current Editor Steve Taylor.

Steve,
The magazines have been a credit to you. The Group is indebted to you and your family for the time and effort you have expended. Without you the magazine would not have ever existed and had that been the case our Group would have been much the poorer. We will miss you and your informed opinions. For me personally I am going to miss a trusted and true friend. Good luck where ever the future may lead you.
Joe.

And to quote Hood!

Well for the drones of the
social hive that there are
bees of an industrious turn
willing, for an infinitesimal
share of the honey, to
undertake the labour of it's
fabrication.

Allen (Joe) Wright
President H.V.99'ers.

TEXAS INSTRUMENTS
CHIP-CHAT
©1981 TEXAS INSTRUMENTS

****
* TI BOOKS AT SALE PRICES *
****

Wilboprint and Computer supplies have just released their 1986 catalog which contains the following TI994/A books at sale prices.

Art and Graphics with your TI99/4A.	$5.75
Get Personal with your TI99/4A.	$8.25
Basic Tricks for TI99/4A.	$5.75
Get More from the TI99/4A	$7.00
Entertainment Games in TI Basic.	$5.75
Dynamic Games for your TI99/4A.	$5.75
Computes First Book of TI Games.	$10.75
TI Playground.	$5.75
Stimulating Simulations for the TI99/4A.	$5.75
I Speak Basic to my TI99/4A.	$8.25
TI99/4A Basic Programmes	$5.75
Fun and Games with your TI99/4A.	$5.75
32 Basic Programmes for TI99/4A.	$10.75
Getting Started with the Texas TI99/4A.	$7.00
Your First TI99/4A Programme.	$8.25

All the above prices include postage and handling (while stocks last) and can be obtained from.

WILBOPRINT and COMPUTER SUPPLIES
134 Abercrombie Street
Chippendale
N.S.W. 2008
PH. (02)699 9933

****
* FLIGHT SIMULATOR SOON ????? *
****

The FLIGHT SIMULATOR programme demonstrated at the November Chicago TI Faire was written by John Dow of DOW 4 Gazelle fame. The programme is very similar to the Microsoft version having a full instrument panel and the ability to look out the windows.

It appears that the only thing holding up the release of the programme is a disagreement between the author and distributor over reasonable price.

****
* MINI MEMORY BATTERIES *
****

In issue No.5 of HV99 NEWS was a description on how to modify a lithium calculator battery to replace your flat Mini Memory battery. Along with the description was a warning of the possible dangers of such a modification.

Now for the faint-hearted (including me) who were put of by the warning it appears there is another source for the batteries. If you would like a replacement lithium cell with welded on leads for $US 5.00 plus postage write to.

Thomas F. Spillane
DIJIT Systems
4345 Hortensia St.
San Diego,
California 92103

****
* INFOCOM ADVENTURE BUGS *
****

Dedicated adventure fans all agree that Infocom adventures are the best available. They are still the same suceptible to bugs as is any computer software, so this month I will look at one of the adventures "ENCHANTER" and describe some of the bugs that have crept into the programme

ENCHANTER.

If you send the Turtle into the temple, the game acts as if you had gone instead, and you end up dying. The RSPCA must have slipped that one into the game to give some protection to turtles.

If you guncho the magic rope that protects the jewelled box, the rope disappears and the box opens. However the box is empty, and if you LOOK, you can still see the box coiled by the rope. How's that for an optical illusion?

If you move the lighted portrait in the gallery, then LOOK, the portrait has disappeared in the blink of an eye.

The last bug has to do with keeping your water jug filled. As long as you have some water in it, you can refill it to the brim by saying TAKE WATER, no matter where you are.

Coutresy NEW ZORK TIMES.

"SOFT-SELL" BY RON KLEINSCHAFER HV99.

We have all seen the electronic notice boards in shop windows, well you can now create the same effect on your TI99/4A. The possibilities are endless, such as "No more PARSEC until your homework is completed" or "Dear hubby, dont forget to vacuum the carpet after you have finished doing the washing up." !!!!. I'm sure you will think of a lot more applications for this great program. What are you waiting for? Start keying it in.

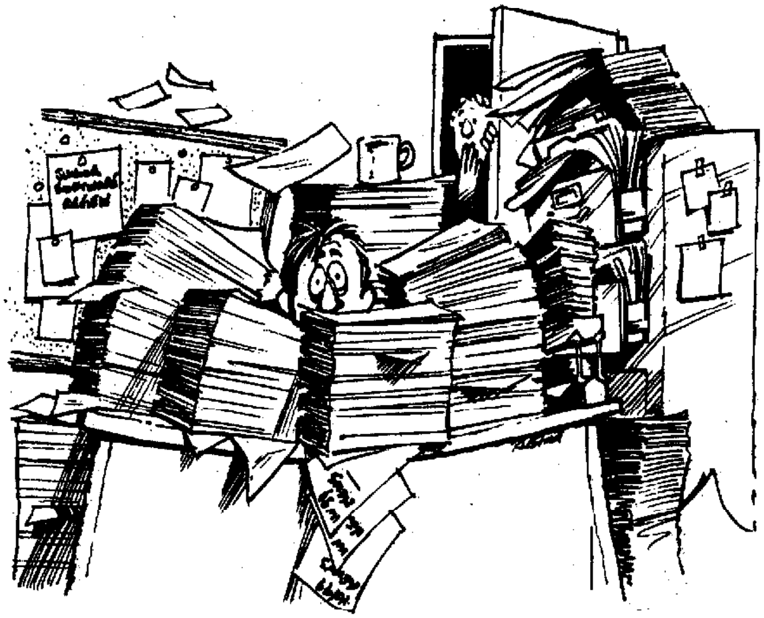
```
100 REM **SOFT  SELL**
110 REM *BY R.KLEINSCHAFER*
120 REM ***HV 99ers***
130 REM * EXT  BASIC *
140 REM *TYPE IN YOUR OWN
LOGO IN LINE 5040
150 REM *TYPE IN YOUR OWN
MESSAGES*
160 REM *IN LINES 600  TO
1200
170 REM *COMMAS SPLIT THE ME
SSAGES BY 28 SPACES*
180 REM *IF NO MORE MESSAGES
 ARE REQUIRED
190 REM * TYPE IN TWO "@@"
SIGNS
200 REM * AT THE START OF
THE NEXT LINE AFTER "DATA #"
210 REM * OR RUN THIS DEMO.*
220 CALL CLEAR :: CALL SCREE
N(9)
230 FOR I=1 TO 8 :: CALL COL
OR(I,2,1):: NEXT I
240 CALL CHAR(136,"7EC1COFE7
F03837E")
250 CALL COLOR(14,16,1)
260 CALL SCREENS
```

```
270 CALL SCREEN(2):: FOR I=1
 TO 8 :: CALL COLOR(I,6,1)::
 NEXT I
280 CALL LOGO
290 FLAG=0 :: ROW=6
300 RESTORE 600
310 READ M$
320 IF M$="@@" THEN 220
330 IF M$="#" THEN 410
340 CALL HCHAR(ROW-2,1,136,3
2)
350 CALL HCHAR(ROW+2,1,136,3
2)
360 M$=RPT$(" ",28)&M$
370 FOR SHOW=1 TO LEN(M$)
380 DISPLAY AT(ROW,1):SEG$(M
$,SHOW,28):: FOR DEL=1 TO 10
 :: NEXT DEL
390 NEXT SHOW
400 GOTO 310
410 CALL CLEAR
420 CALL LOGO
430 FLAG=FLAG+1 :: ROW=ROW+2
440 IF FLAG=1 THEN RESTORE 7
00 :: GOTO 310
450 IF FLAG=2 THEN RESTORE 8
00 :: GOTO 310
```

```
460 IF FLAG=3 THEN RESTORE 9
00 :: GOTO 310
470 IF FLAG=4 THEN RESTORE 1
000 :: GOTO 310
480 IF FLAG=5 THEN RESTORE 1
100 :: GOTO 310
490 IF FLAG=6 THEN RESTORE 1
200 :: GOTO 310
500 FOR C=1 TO 8 :: CALL COL
OR(C,2,1):: NEXT C
510 CALL SCREENS
520 GOTO 270
600 DATA THIS PROGRAM IS TO
SHOW HOW THE TI 99/4A COMPUT
ER CAN BE USED TO DISPLAY AN
Y MESSAGES YOU MAY WISH PEOP
LE TO SEE OR KNOW ABOUT
610 DATA WITH THE REM STATEM
ENTS REMOVED YOU CAN DISPLAY
 APPROXIMATLY 12000 BYTES OF
 INFORMATION WITH JUST THE C
ONSOLE***
699 DATA #,REM<--DO NOT REMO
VE THESE DATA LINES!.
700 DATA IT COULD BE USED TO
 ADVERTISE SOME SERVICE YOU
CARRY OUT OR TO DISPLAY SOME
THING YOU WISH TO SELL ! !
799 DATA #
800 DATA OR YOU MAY WISH TO
ADVERTISE SOME WARNING THAT
WILL REPEAT ITSELF,***SUCH A
S*** ------------->
899 DATA #
900 DATA WARNING! WARNING! *
** KEEP YOUR MEMBERSHIP SUBS
CRIPTION UP TO DATE TO RECIE
VE THE BENEFITS OF BEING A M
EMBER OF HV 99ERS.
999 DATA #
1000 DATA OR YOU MAY WISH TO
 JUST SHOW OFF THE CAPABILIT
IES OF YOUR TI 99/4A
1099 DATA #
1100 DATA ANYWAY IT COULD BE
 THE BASIS OF AN EYE CATCHIN
G DISPLAY TO USE IN MANY SIT
UATIONS.
1110 DATA THIS PROGRAM IS BO
UGHT TO YOU WITH THE COURTES
Y OF THE HV 99ERS TEXAS INST
RUMENTS COMPUTER USER GROUP.
1120 DATA SEE REM STATEMENTS
 FOR MORE DETAILS.
1199 DATA #
1200 DATA @@,REM<--**NO MORE
 MESSAGES REQUIRED HERE**.
1299 DATA #
```

```
5000 SUB LOGO
5010 CALL HCHAR(1,1,136,32)
5020 CALL HCHAR(5,1,136,32)
5030 FOR F=1 TO 5
5040 NM$="***LOGO***"
5050 X=LEN(NM$):: P=14-X/2
5060 DISPLAY AT(3,P)BEEP:NM$
5070 FOR D=1 TO 80 :: NEXT D
5080 DISPLAY AT(3,P):""
5090 NEXT F
5100 CALL CLEAR
5110 SUBEND

5120 SUB SCREENS
5130 FOR I=3 TO 29 :: CALL H
CHAR(1,I,136):: NEXT I
5140 FOR I=1 TO 24 :: CALL V
CHAR(I,29,136):: NEXT I
5150 FOR I=29 TO 3 STEP -1 :
: CALL HCHAR(24,I,136):: NEX
T I
5160 FOR I=24 TO 1 STEP -1 :
: CALL VCHAR(I,3,136):: NEXT
 I
5170 FOR I=9 TO 16 :: CALL S
CREEN(I):: FOR D=1 TO 300 ::
 NEXT D
5180 DISPLAY AT(12,4)SIZE(23
)BEEP:""
5190 ON I-8 GOTO 5210,5220,5
230,5240,5250,5260,5270
5200 NEXT I
5210 DISPLAY AT(12,10)SIZE(1
0):"W A T C H" :: GOTO 5200
5220 DISPLAY AT(12,11)SIZE(7
):"T H I S" :: GOTO 5200
5230 DISPLAY AT(12,10)SIZE(9
):"S P A C E" :: GOTO 5200
5240 DISPLAY AT(12,12)SIZE(5
):"F O R" :: GOTO 5200
5250 DISPLAY AT(12,13)SIZE(4
):"A N" :: GOTO 5200
5260 DISPLAY AT(12,6)SIZE(17
):"I M P O R T A N T" :: GOT
O 5200
5270 CALL SCREEN(6)
5280 DISPLAY AT(12,3)SIZE(23
):"A N N O U N C E M E N T"
5290 FOR D=1 TO 800 :: NEXT
D :: CALL CLEAR
5300 SUBEND
```



"FAREWELL"

As the saying goes "all good things must come to an end". In this case not only is this my last magazine as Editor of Hunter Valley Ninety Niner News but very shortly I will be leaving Lake Macquarie permanently and shifting to Tasmania.

The past 10 months as Editor has been one of the most enjoyable periods of my life for not only have I had the opportunity to assist in the formation of Australia's youngest TI99/4A User Group but I have met some people who will be true friends for life.

Anyone who has been at my house around deadline time will recognise the drawing above, perhaps now I will be able to finally get my study tidied up!!!. I'm sure a lot of people wonder how the magazine ever got to press when they have seen how disorganised the desk is. Perhaps if Joe and a couple of the other of the committee had seen it a year ago I feel sure I would have never been entrusted with the job of Editor!!.

From a very shaky first issue about ten months ago the magazine has slowly evolved and matured, learning by our mistakes as we went along, there were a couple of KNOCKERS earlier nitpicking about spelling mistakes and content but this attitude is soon remedied by saying "why dont you write the magazine well"... instant silence.

I would like to thank everyone who has contributed in any way to the magazine, for you are the people who did all the hard work, I merely stuck the bits together (hopefully in the right position). I hope that you continue your work and give our new Editor the same support you have shown me.

I intend remaining a member of the HV99 for a long time to come and hope to keep in close contact. If you ever intend to visit Tasmania feel free to come and visit in Hobart. Give us a couple of weeks notice so that Denise can get the coffee on !!!!!.

STEVE TAYLOR HV99

HOMEMADE PERIPHERAL EXPANSION UNIT OR HARDWARE THE HARD WAY

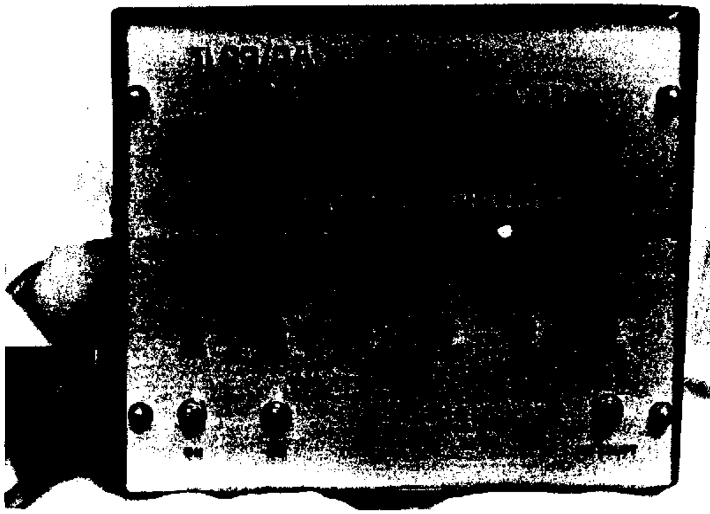
THIS PROJECT IS GIVEN FREELY TO THE TI99/4A COMMUNITY AND IS NOT INTENDED TO BE A COMMERCIAL ENTERPRISE. IT IS THE INTENTION OF HV99 TO PRODUCE PRINTED CIRCUIT BOARDS AND PERHAPS SUPPLY COMPONENTS IN A KIT FORM. IF YOU ARE INTERESTED IN THIS PROJECT PLEASE CONTACT A COMMITTEE MEMBER BY PHONE OR ATTENDING THE MEETINGS ON TUESDAY NIGHTS AT 7.00 PM.

By Ron Kleinschafer HV 99/ers.
Grawin via walgett.
NSW 2832.
AUSTRALIA.

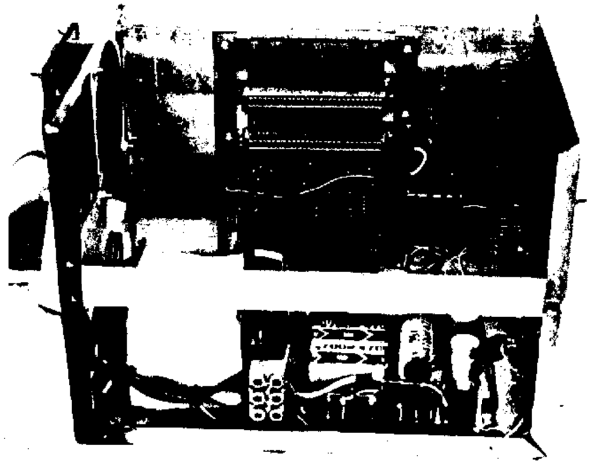
Belonging to the fold of those perculiar beings so aptly described by BERNIE ELSNER and PHIL WEST of WESTRALIAN INSTRUMENTS ?????, as a TISADUDD I have discovered from many discussions with other members of that genre one salient feature, and that is they do not give up very easily. After attending several meetings with that happy group of souls who shall hereafter be refered to as "THE CLUB", I would stand and gaze longingly at the equipement used during demonstrations and lectures at those meetings and listen to many recourses on the advantages of using the many and varied programs available to the "CLUB", but one major requirement that stood out was the nesessity for disc drives, RS232 (for modems and printers etc.); Now being blessed with just console, monitor and cassette I found that I was severely limited in choice and usage of programs for my personal orphan, so after months of searching for the seemingly now extinct P.E. Box and having a bank balance that would not withstand the shockwaves of purchasing a CORCOMP unit, I suddenly realised that I was after all a TISADUDD and with the

NOT
TO
KIT
TEE
PM.

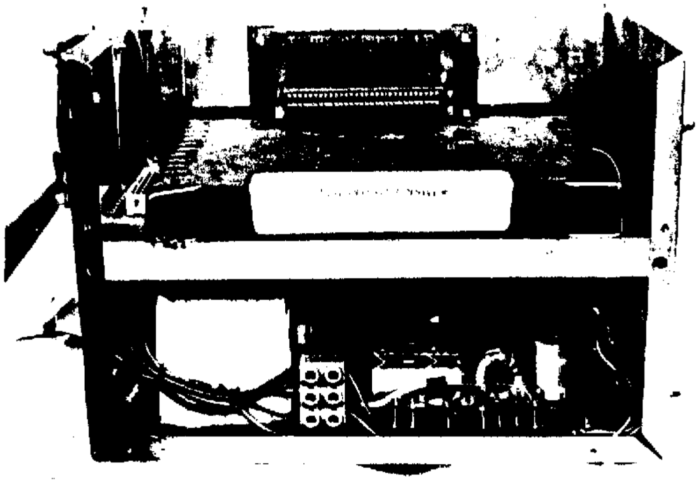
and
and
the
and
the
ent
for
and
sed
and
ely
of
so
the
and
not
of
I
ter
the



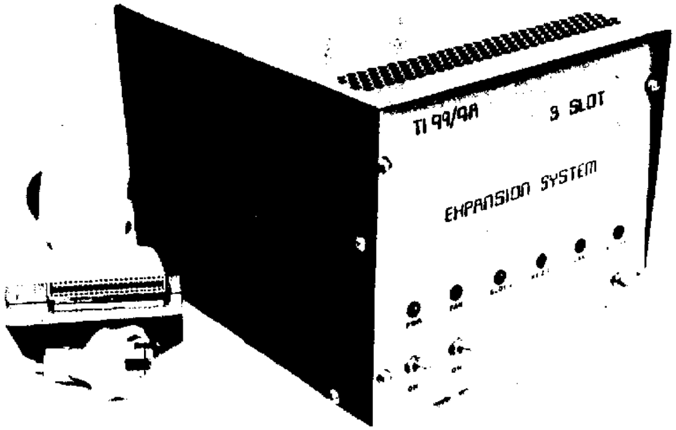
1



2



3



4

availability of hardware such as disc controller cards, RS232 cards, and disc drives (albeit SS/SD ,the drive you have when you dont have a drive) all A/LA T.I., still available at resonable cost (if you shop around) and better still as required to suit the wallet I decided to build one, the "ONE" being a P.E.U.

The first design consideration was to list what was needed, at this stage the 32K memory expansion was installed in the console, with thanks to WESTRALIAN INSTRUMENTS ?????. (scratch one), with an AXIOM printer interface attached my first choice was disc controller then RS232 (for modem) and perhaps a spare slot for experimentation or P-Code or something who knows ?), so I opted for a three slot P.E.U.

I must stress here that after building a one off project such as this that it is then that you realise vast improvements could be made in PCB design, component layout, and construction, but it works and that is what matters.

As a general description the unit measures 200mm wide 170mm high and 240mm deep, the transformer and power supply are mounted in the bottom of the case with the main PCB mounted vertically on the R/H side, the cards slot into the main PCB horizontally in the top half of the case and are supported on aluminium angle rails, the front panel layout is as per photo No.1, on the lower R/H side is the interupt button as described by BILL KNECT of the LA/99ers above that is the system active LED, then the card LEDS, DSK, RS232, SLOT 3, FAN !! I can hear the screams now "ARRGH!!! A FAN" perhaps I should digress a little and describe the conditions under which this unit operates, my power comes from a diesel generator of vintage construction and the supply voltage under certain conditions can reach levels of some 280 volts, then on top of that the daytime ambient temperatures run on average through summer from a cool day of 35 C to getting fairly warm at 46 to 47 C, now the cards generally employ on average three voltage regulators each and these devious little devices have thermal shutdown protection and after putting them into a closed box on odd occasions this did occur, hence the fan !! AM I FORGIVEN ???, anyway its switched and only used when needed, the console has never given any trouble under such conditions, although you could cook eggs in front of the module port, hows that for user friendly, (lets see a vomitore or crapple do that), in most areas of temperate conditions the fan could be safely left out, its up to you.

I decided to follow TI'S circuits and protocol as per the workshop manual and after studying these diagrams I soon realised that the so called flex cable connector card and the peripheral back plane could be etched onto one PCB. The flex cable connector card contains four lousy chips, a few resistors, one voltage regulator and a few small capacitors yet measures nearly 150mm BY 200mm, wasted space !!. Two other PCB's are required, one for the power supply and one for the I/O port connector, these I will attempt to describe in detail and then my own construction of the whole unit, but many variations could be made to suit the individual constructor.

** DISCLAIMER **

IF YOU UNDERTAKE TO CONSTRUCT THIS PEU OR ANY SIMILARITY TO IT USING THE IMFORMATION AS PUBLISHED HERE THE USUAL DISCLAIMER APPLIES, IE. BECAUSE EVERYTHING IS BEYOND THE CONTROL OF THE AUTHOR.
** IF YOU COOK SOMETHING !! ROUGH **.

** MAIN PCB **

Figures 1A and 1B.
This board is double sided and is a little unusual insomuch that the components are mounted on the main track side, this was done to simplify design, reduce the number of lead through holes, and make the registration of the two sides less critical. some soldering expertise is required because the chip leads must be soldered on both sides of the board and on the component side this is very close work, a suggestion is to use molex pins for the chips as they would be easy to solder in then just plug in the chips, also some resistor leads act as lead throughs, where there are no etch pads on the top side just cut the leads off close to the board after soldering on the main side,

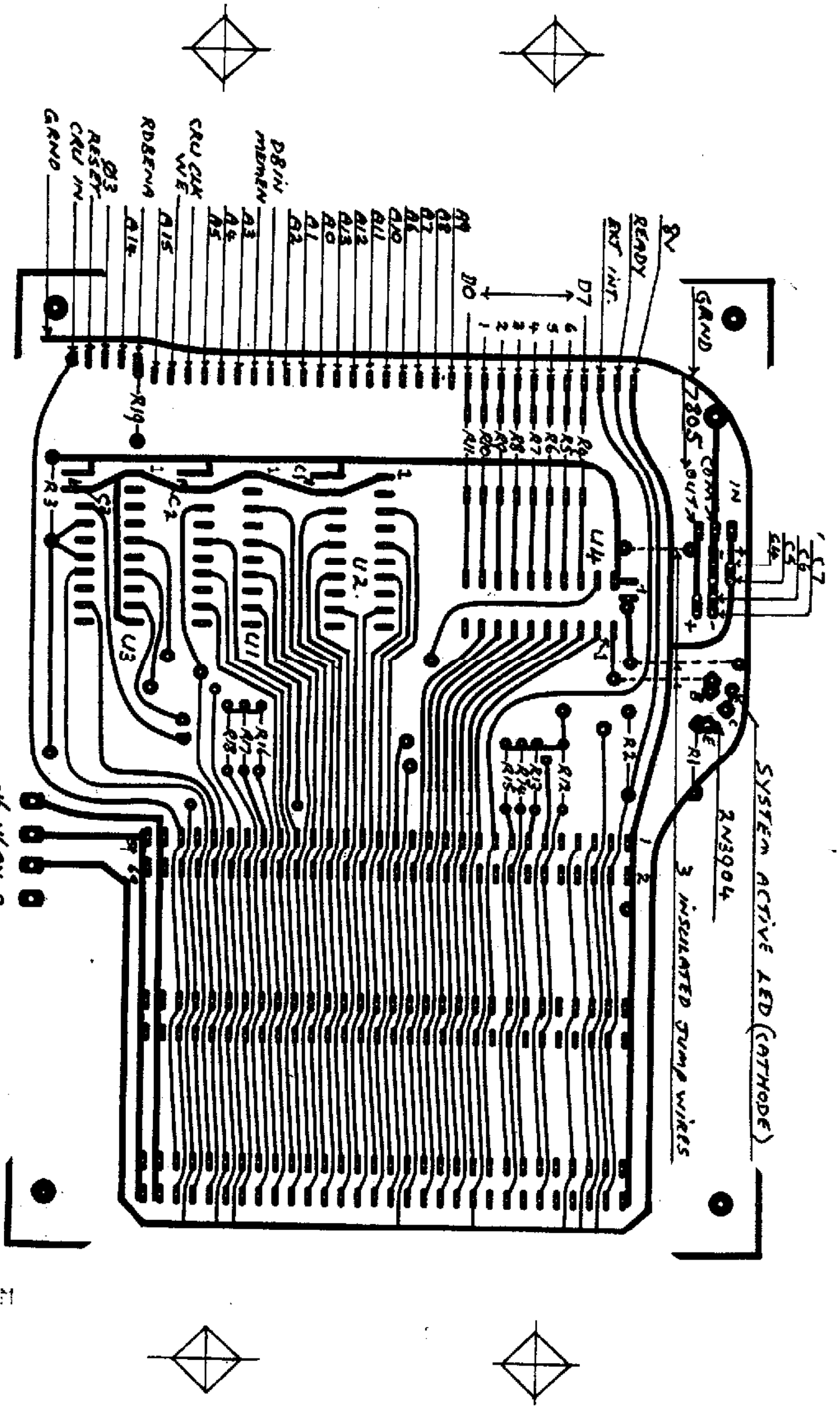
! AM
tched
the
ouble
you
f the
user
re or
s of
could
ou.

cuits
kshop
these
he so
d and
ld be
cable
lousy
ltage
itors
00mm,
PCB's
power
port
t to
y own
but
de to
.

THIS
USING
HERE
, IE.
THE

ROUGH

s a
the
main
to
mber
the
less
tise
eads
s of
side
a
for
y to
the
act
re no
t cut
board
side,



1-C3	.1uf MONOLITH
4,C7	22uf TANTALIUM
5,C6	470pf CERAMIC
1	100 Ohm
2	10K
3	1K
4/R17	All 47 Ohm
1/U3	74LS244
4	74LS245

FIG. 1A
MAIN P.C.B. COMPONENT SIDE
WARNING. MIRROR IMAGE. SEE TEXT

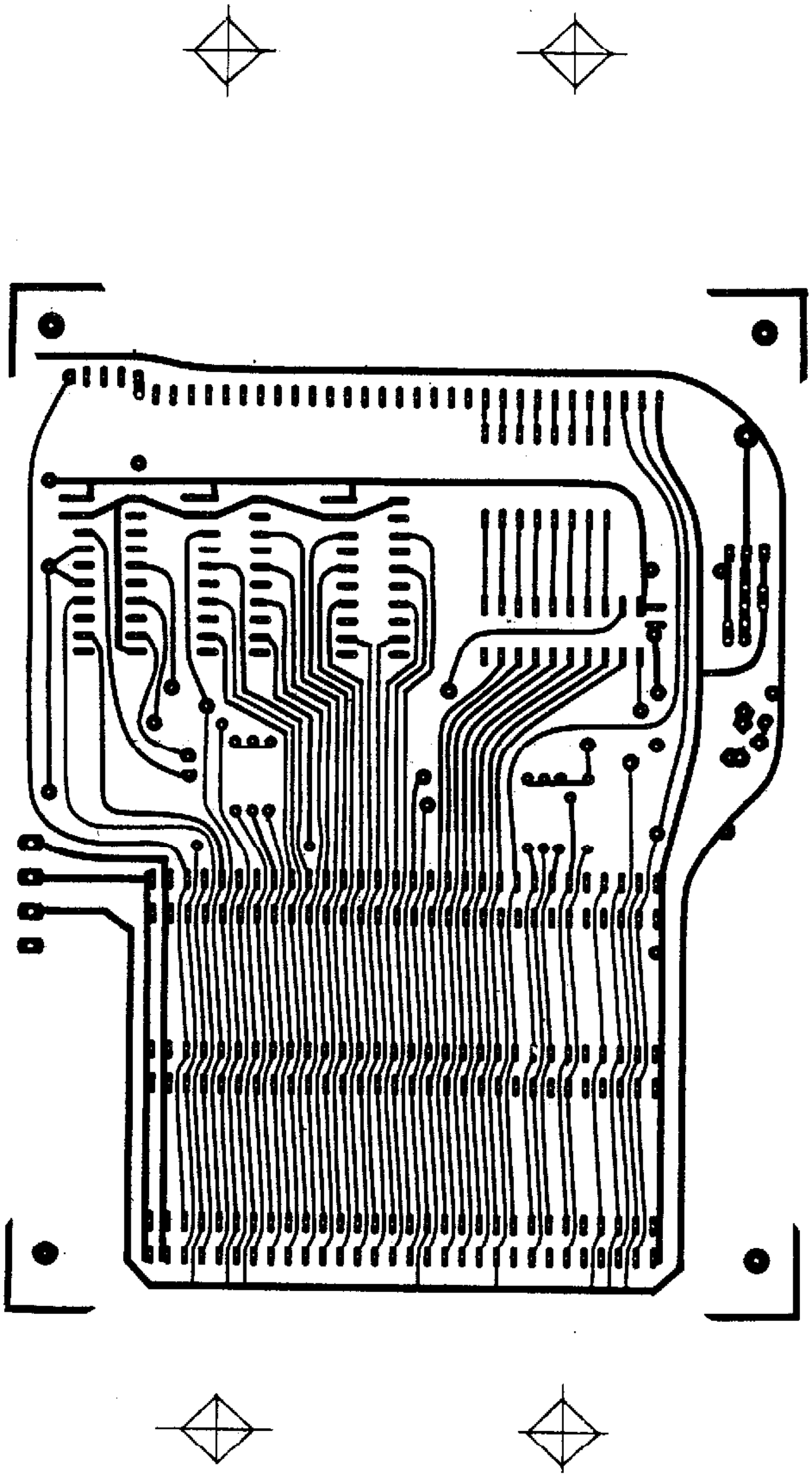


FIG. 1B

MAIN P.C.B. COMPONENT SIDE

WARNING. MIRROR IMAGE. SEE TEXT

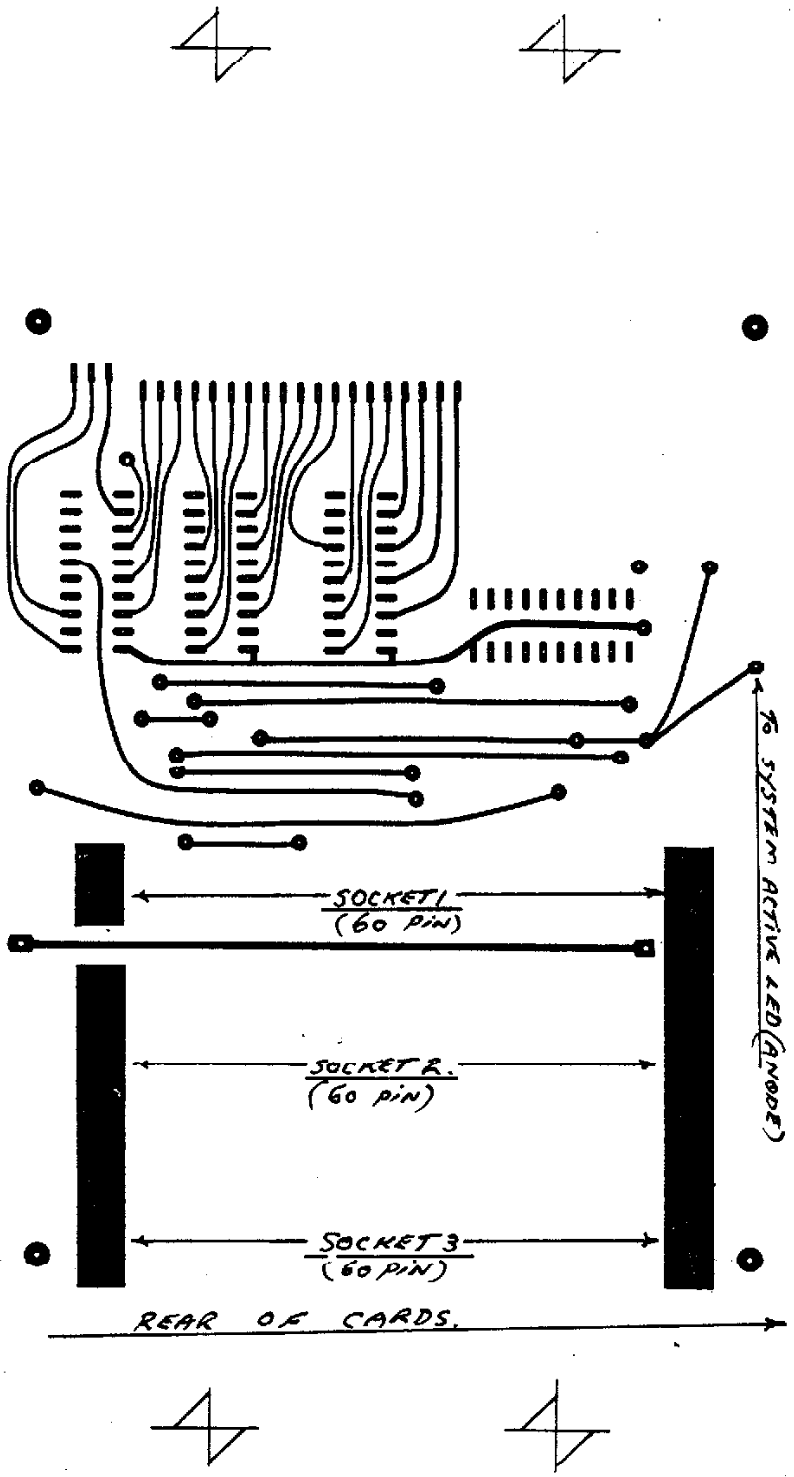


FIG. 2A
MAIN P.C.B. CARD SIDE

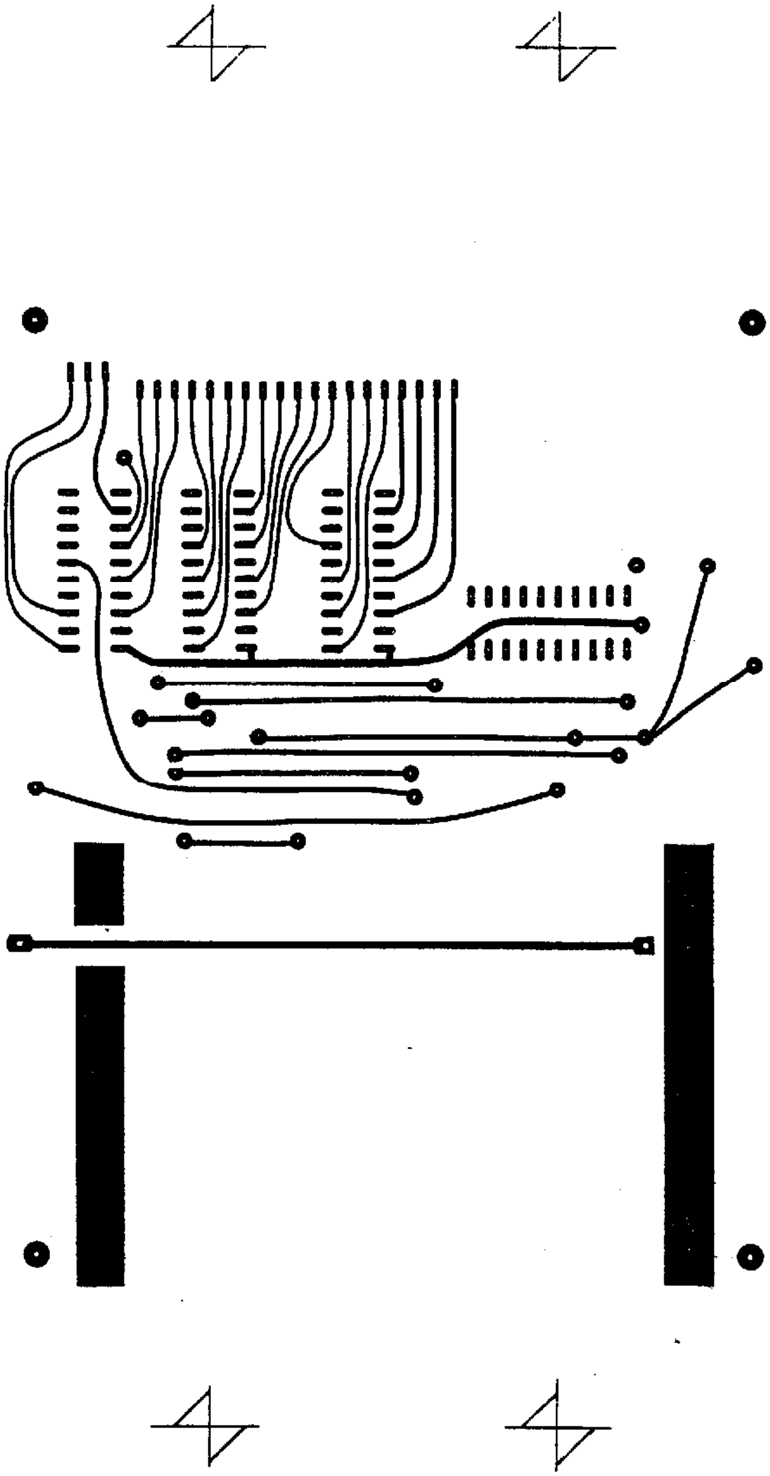


FIG. 2B
MAIN P.C.B. CARD SIDE

two
one
cabl
full
canu

***

Due
boar
phot
over
lami
two
ther
if
enth
proj
probl
could
with
being
full
have
etch
possi
fitte
are
the
socke
gold
these
her
pair
them
shown
some
all
fitte
port
the
plate
decou
ask
hole
or
olde
every
uch
eade
esig
rojec

he
ade
ut
s i
han
rtwor
s ne

* TH

igure
his
easu

two photostat copies are presented, one with component placement and cable connections tha other plain at full size so that photo mask copies can be made.

*** WARNING ***

Due to a photocopy error the main board side is a mirror image so if a photo resist mask is made turn it over before exposing to sensitised laminate, also registration of the two sides will have to be resolved there are many ways to do this but if you feel competent or enthusiastic enough to tackle the project then it should present no problems, alternativly new artwork could be laid up on 1/10" grid film with some suggested improvements being, replace the chip pads with full size instead of narrow cut as I have used and fill in the ground etch around the board wherever possible, after all components are fitted and any obvious lead throughs are soldered in there only remains the fitting of the three card sockets, I used 170 pin AMTRON TYREE gold plated .1" card edge connectors these were cut down and fitted as per figure 5(A), leave one extra pair of pins on either end, push them out and fit the shim brass as shown in figure 5(A), if you can get some 60 pin sockets all the better, all other sockets were modified and fitted the same way I.E. the I/O port connector and replacement of the module port socket with the gold plated series. There is a decoupling the supply, the final task is to check and recheck the whole board with a multimeter, check for continuity, check for correctly soldered connections, check everything, I cannot stress this too much, the prototype caused some headaches because of a faulty resistor so throughout the whole project check and check again.

The original as per photostat was made as a test, breadboard fashion, but worked so well that it was used as is, naturally if you want more than three cards merely extend the artwork to accomodate as many cards as needed.

** THE I/O PORT CONNECTOR **

Figures 3(A),3(B),4(A),4(B)
This board is also double sided and measures approx 80mm by 45mm the

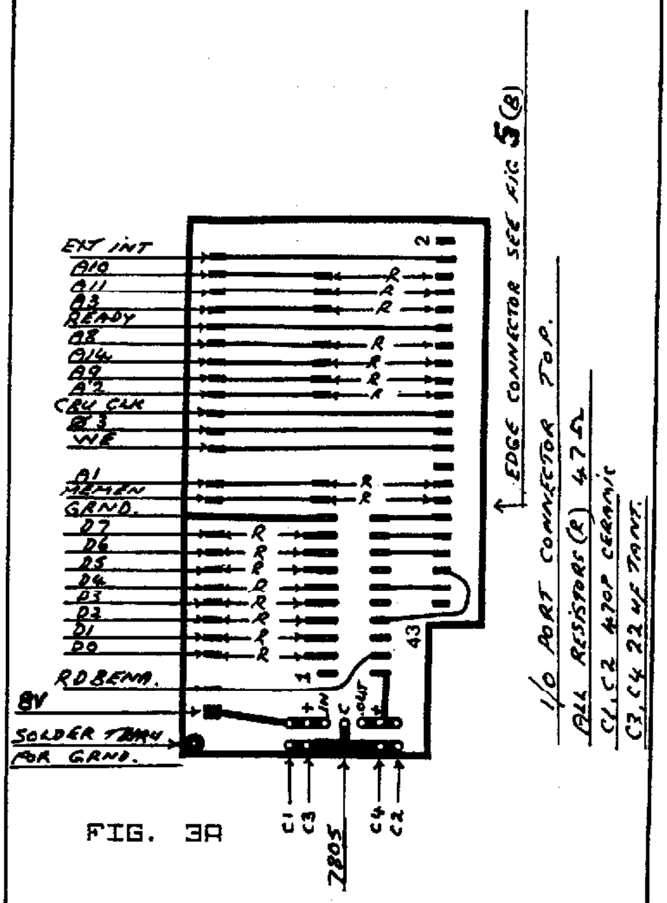


FIG. 3A

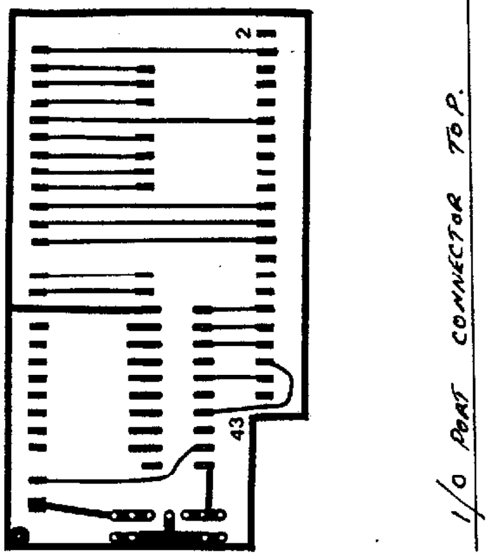


FIG. 3B

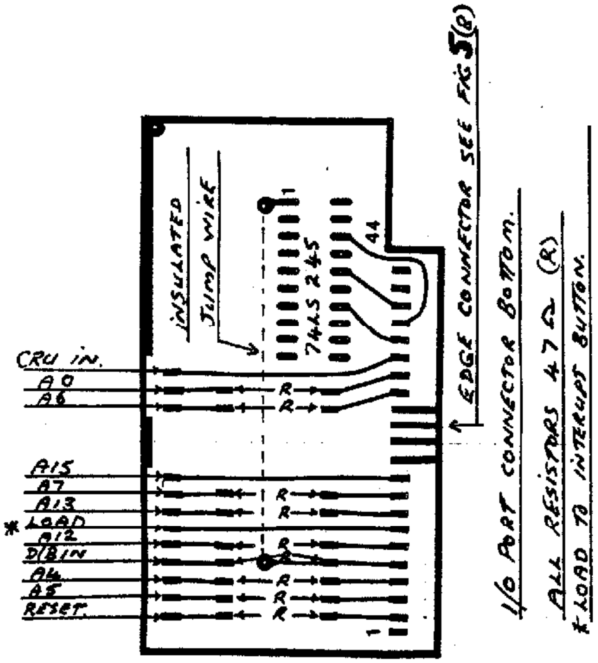


FIG.4A

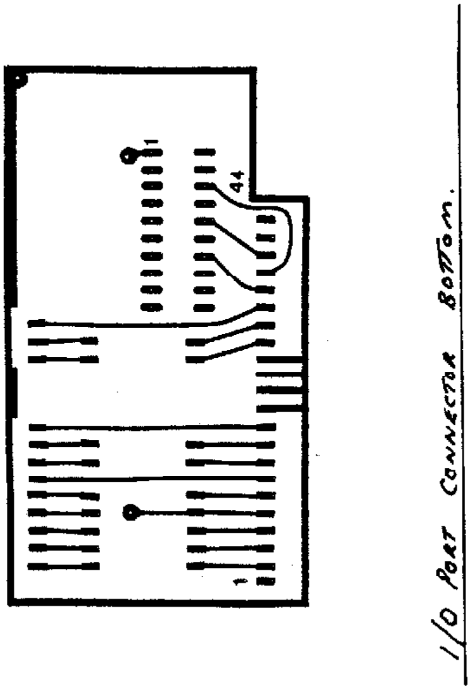


FIG. 4B

resistors are soldered onto pads on both sides of the board as per figures 3(A) and 4(A) again improvements could be replacement of the chip pads with full size ones and fill in the ground etch around the board wherever possable. The voltage regulator is soldered in and bent over and stuck to the top of the chip with some glue, it gives off no heat as the chip draws very little current. The 44 way edge connctor is fitted as per figure 5(B).

This board is very easy to make as the only holes are for the chip, voltage regulator and capacitors. The prototype was fitted into a black plastic jewellery case (see photo 4) and looks quite ok.

** POWER SUPPLY **

The power supply is straight forward as per the circuit diagram, this board was made as a one off with direct etch transfers, (figure 6), it is suggested that axial lead capacitors be used to reduce height, so get your tech team to belt up some artwork for a PCB.

The three chokes L1/L3 were made by filling 15mm OD. toriodal cores with 22 SWG enameled wire with as many turns as could be put on (single layer) and they seem to prove adequate against glitches, even SCR's on the power lines switching automatic water pumps and thats a real test !!. The bleedown resistors are to disharge the supply and the usual TI warning is in force IE. wait approx two minutes after switching off before removing any cards to prevent the possability of damage.

The transformer may be a little difficult to find, I had considered using two console transformers together but couldnt get my hands on them, and the prices I was quoted to wind one caused further shock waves if you can get something good but if you can't then >>ROLL YOUR OWN<<.

Delving into the dark recesses of an old B/W TV a "C" core transformer was extracted, these are fairly easy to dismantle, the secondary voltages were measured then it was dismantled and the windings removed counting the turns, with this known it is a simple matter to work out the

on
er
in
of
es
nd
he
nd
of
es
ry
ge
re

as
ip,
rs.
a
see

ard
his
ith
6),
ead
ht,
up

by
res
as
on
to
hes,
line
and
down
pply
orce
fter
any
y of

ttle
ered
mers
ds on
ed to
aves,
ut if
K.

of an
ormer
easy
tages
ntled
nting
is a
the

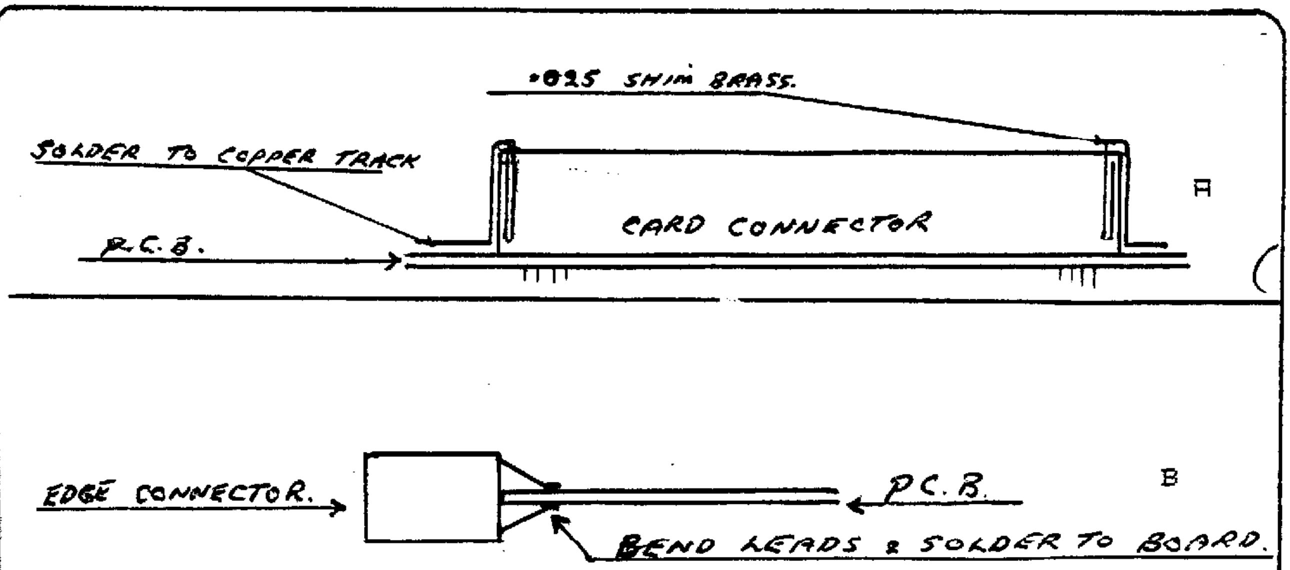
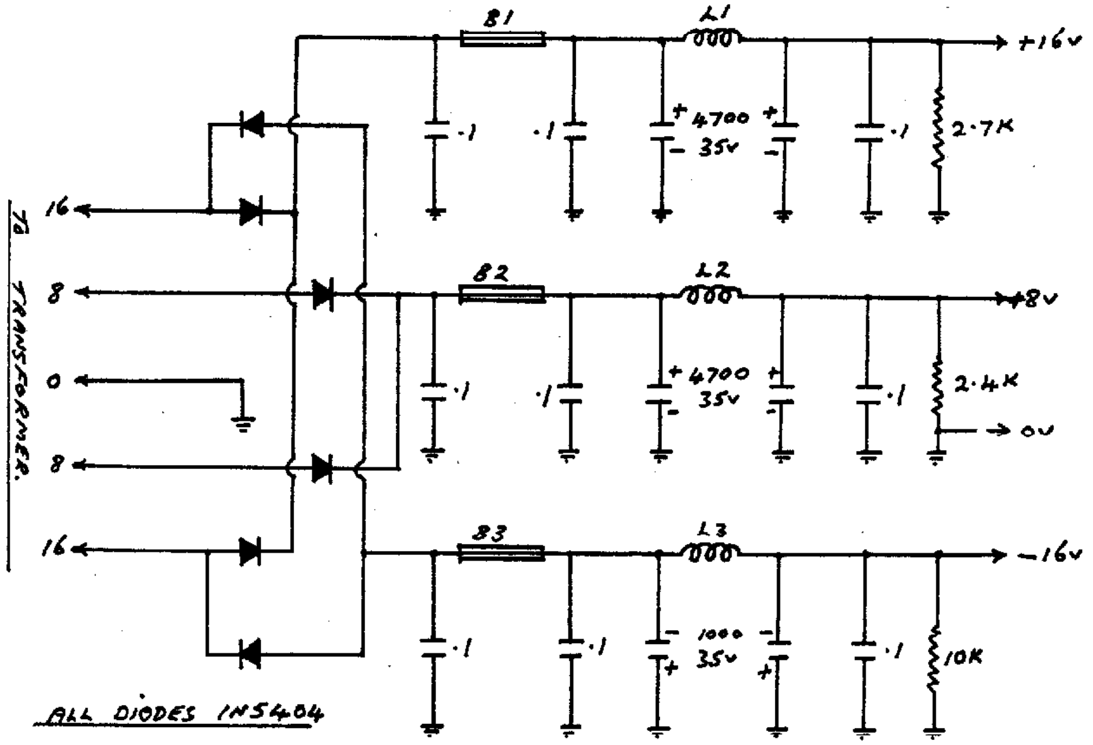


FIG. 5



ALL DIODES 1N5404

B1/B3 SINGLE HOLE FERRITE BEAD (12mm)

L1/L3 SEE TEXT.

FIG. 6

voltage turns ratio then rewind the secondary to provide the required output, the whole exercise took about two hours and the cost was ZILCH !!, although I suspect it is heavy enough to power half a dozen expansion systems, discs and all ??.

**GENERAL ASSEMBLY **

The main board and I/O port connector was in the original (as can be seen in photo 4) connected by IDC ribbon cable and soldered direct, considerable R.F.I was experienced with bare IDC cable and if you choose to use this some form of shielding will be required, through some horse trading I was lucky enough to obtain a length of round 36 way shielded cable and this has been since fitted with great results, connect the two boards by soldering one wire at a time, some crossovers will be nessesary at each end, if you use IDC cable it is a good idea to use a ground wire either side of the PHASE 3, CRU CLK and CRU IN, signal lines, dont forget to use one extra wire to go direct to the interrupt button.

The other side of the button connects to ground. I removed the cards active LEDS and fitted short lengths of hook up wire and fitted miniature polarised plugs and sockets to the wires and to the front panel LEDS so that it is easy to remove and replace any card, similar treatment was given to the main PCB active LED.

I also fitted a four pin polarised plug and socket between the power supply and main PCB. The case was folded up out of sheet aluminium, take care that the main PCB is positioned to enable the card connectors to protrude from the rear of the case, The L/H side of the cards are supported by aluminium angle (see photo 3) and a suitable clamp is fitted to make it secure, as can be seen in photo 3, I used a small clamp onto one of the card pull out wires and it is remarkably stable.

The all up cost by using many components that I had in the "junk" box came to about 80 dollars this together with the controller card (90 dollars) and two, TI SS/SD external disc drives (200 DOLLARS) total 370 dollars, CHEAP !!.

Incidentally the TI SS SD 110 volt drives that are avaiable only require replacement of the power transfomer with a 9/0/9 volt 1 amp transformer (I ratted two out of a couple of old large cassette players) and they work just fine.

Photo 1 is a front view of the unit showing the position of the LEDS, SWITCHES, and INTERRUPT BUTTON.

Photo 2 is a side view of the unit showing the position of the power transformer, power supply and main PCB mounted on a panel bolted to the R/H side of the box,(the third card socket had not been fitted as yet(I didnt have one !),and the aluminium angle rail to support the top? of the cards.

Photo 3 shows the system with one card fitted, note the small clamp fitted to the L/H pull up wire, it is very stable.

Photo 4 is the completed unit showing the I/O port connector and still fitted with the IDC cable but this has since been replaced with round 36 way shielded cable, the fan vent holes are on the front top, the front panel is finished in brushed aluminium and the rest was sprayed with black enamel.

It is a pleasure to have the expansion facilities and use programs such as FUNNELWRITER with which this was written. If anyone wishes further information all enquiries are welcome and if anyone builds the unit any feedback would be appreciated. just write to the above address.

To finish off I think that all dedicated TI users probably have had some remarks made by their wives or sweathearts about their labours on their machines and my wife is no exception, when I was showing her the new setup which now includes CONSOLE, CASSETTE, AXIOM PRINTER INTERFACE, PRINTER, HOMADE EXPANSION SYSTEM, TWO EXTERNAL DISC DRIVES AND TERRY ROSS'S 6 MODULE PORT EXPANDER, when she quipped "Whatever happened to our beautiful little computer that we used to play PARSEC on?, now it looks like a KEYBOARD WITH WARTS !!, Ahh well !!!!, Hmm a modem, now where is that "junk" box?.
R.K.

volt
only
ower
amp
f a
ette
.
unit
LEDS,

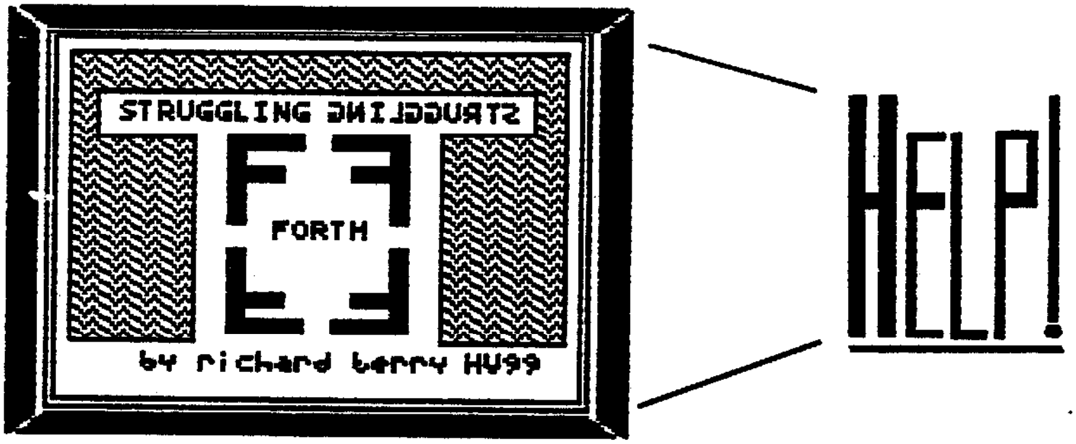
unit
power
main
o the
card
yet(I
inium
p? of

one
clamp
, it

unit
r and
le but
with
he fan
p, the
rushed
prayed

e the
use
ER with
anyone
all
anyone
k would
to the

at all
ave had
ives or
urs on
e is no
ng her
ncludes
PRINTER
PANSION
VES AND
PANDER,
happened
computer
on?, now
TH WARTS
em, now



Well, where to start. Sitting down to the computer after 3 months of computing inactivity is not easy. Steve rang me up 2 days ago and gave me a 2 day article deadline!

You may have noticed the big words to the right of our column logo. HELP HELP HELP etc. Todays column will be more of a request for the same rather than tuition. As you all know we at HV99 are just on a teach ourselves Forth kick. There must be many more experienced programmers out there to help us. I'll detail some problems as I progress. Whilst working in the last two days I've had horrific problems with the computer locking up and doing all sorts of erratic things. Today I've swapped my spare console and touch wood its running ok.

MASTER MENU'S.

To me one of the beauties of Forth should be the ability to run multiple programs quickly using booting of binary screens. Sure one can do that in extended basic but with even minimal required error checking its slower and the programs run abyssmally slowly. Now my first request for help. If what I am about to theorise is impossible can someone write and let me know.

As far as I can see there should be two methods:

METHOD 1

REFER TO SCREEN(7)

```
SCR #7
 0
 1 : MMENU PAGE ." MASTER MENU"       CR CR
 2              ." Choose option:"    CR CR
 3              ." 1.PROGRAM ONE"     CR
 4              ." 2.PROGRAM TWO"     CR
 5              ." 3.STRING EXAMPLE"  CR
 6              ." 4.FORTH"           CR ;
 7
 8 : CCHOICE 14 2-AT KEY DUP EMIT 48 -
 9           DUP 1 = IF 48 LOAD    ELSE
10           DUP 2 = IF 60 LOAD    ELSE
11           DUP 3 = IF 57 LOAD    ELSE
12           DUP 4 = IF QUIT PAGE  ELSE
13           THEN THEN THEN THEN DROP ;
14 : MASTERGO MMENU CCHOICE MYSELF ;
15
```

Here we have a simple method of constucting a master menu/choice giving us the options of loading various programs which we have saved on nominated screens in binary code.

When we compile screen #7 the key words to run the intended program ie your final definition, is not yet in the dictionary so it cannot be included (?correct-or is there some way around this ?)

ie one cannot say:
DUP 1= IF GO THEN etc

(GO starting our string program) so I chose the option of just referring to an alternate screen to load and then autostart the programs.

REFER TO SCR# 57.

At the end of each program in the exit option I have simply put COLD to reboot the non-core words and the master menu. Eg see screen 55 line 14 from the String programme.

```
SCR #57
  0
  1 58 BLOAD
  2
  3 60 ( AUTOSTART TO STRING EXAMPLE)
  4
  5    ( 58 & 59 HAVE BINARY CODE OF STRING EXAMPLE)
  6
  7
  8
  9
 10
 11
 12
 13
 14
 15
```

Problems:

Firstly though quicker and more versitile than E/Basic its lag time is too slow. Secondly I've encountered problems trying to boot programs in binary this way which have components accessing a Basic File, such as the basic disk catalogger, or file example I gave you. Its cbvious to me I don't understand somthing basic, but cant discover what! HELP HELP HELP!

In case you missed the first few examples, to enable you to reconstruct this type of menu we have included the listings of the string example at the end of this article.

METHOD 2

This one I can get to only partially work. In theory its much more elegant and quicker. Now you will have to use your imagination. I hope my concepts are roughly accurate.

Say we have our basic core FORTH WORDS as contained on SCREEN 8-19. In addition we have saved whatever options we so desire in binary to SCREENS 20 onwards till finished. For example I save some of the basic options modified to delete many graphics and sprite words (I just deleted them from a copy of the source code as I never use them) and added my own words such as a few stack operators and our string words.

```
SCR #30
  0 ( STRING:Basic Words 18Jly85)                         (  STACK EFFECTS)
  1 : GET$  TIB @ SWAP EXPECT 0 IN ! 13 WORD            ( Adr to put new )
  2         HERE OVER OVER C@ DUP ROT C! 1+ 1           ( string,count   )
  3         DO 1+ OVER OVER C@ SWAP I + C! LOOP         (       ----     
  4         DROP DROP ;
  5 : MOVE$ SWAP COUNT 1+ SWAP 1- ROT ROT CMOVE ; ( Adr1 Adr2--- )
  6 : ADD$  DUP >R SWAP COUNT ROT COUNT DUP ROT
  7         + SWAP ROT DUP ROT + R>  C! CMOVE ;  ( Adr1 Adr2--  )
  8 : SAME$ COUNT ROT COUNT ROT MAX 1 SWAP 0      ( Adr1 Adr2--flag
  9         DO >R DUP C@ ROT DUP C@ ROT =          ( Where these adr
 10            R> MIN SWAP 1+ ROT 1+ ROT           ( contain strings
 11         LOOP >R DROP DROP R> ;                 ( with counts
 12
 13 : SEG$  ROT DUP >R OVER SWAP C! SWAP ROT + R> ROT 0
 14         DO OVER OVER SWAP C@ SWAP 1+ C! 1+ SWAP 1+ SWAP
 15         LOOP DROP DROP ; ( From adr, adr to put, start,num--)
```

```
SCR #31
  0 ( Additional Stack manipulators)
  1
  2
  3
  4 : FREE      SP@ HERE - . ;
  5 : 2DUP      OVER OVER ;
  6 : 3DUP      OVER OVER >R >R >R >R DUP  R> R> ROT R> R>  ;
  7 : 2DROP     DROP DROP ;
  8 : 3DROP     2DROP DROP ;
  9 : AT        GOTOXY ;
 10 : CONTINUE  BEGIN ?KEY 32 = UNTIL ;
 11 : EB        EMPTY-BUFFERS ;
 12 : PAGE      CLS 0 0 AT ;
 13
 14
 15
```

Hence my core words saved in binar stretch from # 8-29. The modifie boot screen does a binary load on 2 (SEE SCREEN #3) and then leaves yo back in Forth.

```
SCR #3
  0 ( WELCOME SCREEN ALTERED SOURCE CODE)
  1 BASE->R HEX 10 SYSTEM  ( Clears screen)
  2  0 0 GOTOXY ." Booting... FORTH NON STANDARD WORD SET" CR
  3  B 2 GOTOXY ." -Saved Mar 5th  1985"  CR 10 83C2 C!
  4 ( QUIT off) B 3 GOTOXY ." -EDITOR" B 4 GOTOXY ." -PRINT"
  5  B 5 GOTOXY ." -GRAPH" B 6 GOTOXY ." -DUMP" B 7 GOTOXY ." -VD
  6 DES" B 8 GOTOXY ." -COPY"  B 9 GOTOXY ." -FLOAT"
  7      B A GOTOXY ." -STRING WORDS"
  8 0 C GOTOXY ." WARNING:This disk does not contain all  the sta
  9 rd definitions contained on   the TI-FORTH master disk. Many 
 10 ds    have been deleted including most sprite words,and others
 11 or example new string words and stack manipulators added."
 12 CR CR    DECIMAL 20 BLOAD
 13 1 VDPMDE ! 0 DISK_LO ! 180 DISK_HI
 14 ." MEMORY REMAINING:" FREE ." bytes." CR
 15 R->BASE
```

Now to my way of thinking one shoul be able to compile say the Strin example program on the top of th dictonary then save the binary imag of this to say screen 70 by typing

' MSG1 70 BSAVE

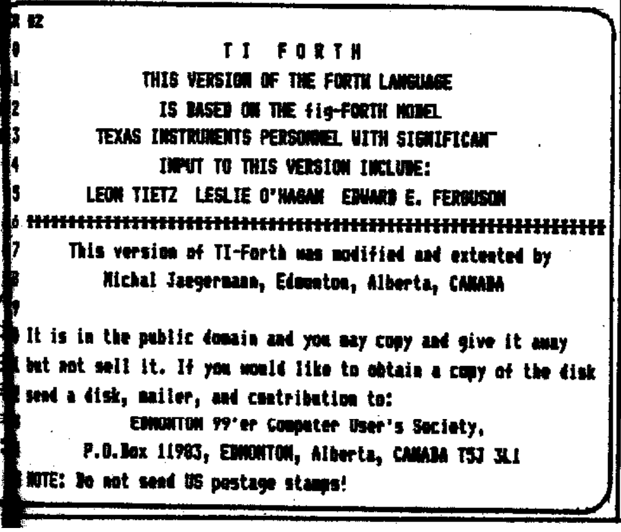
Since MSG1 is the first word of the application. Then if we type FORGET MSG1 this and all succeeding dictionary entries pertaining to this program will be deleted. Now we can load our second application and repeat the process storing it on other Screens in binary form, and so on. Since the core dictionary remains the same all the words in the different applications will contain correct address pointers when they need later to access a word in our expanded core.

low if we again construct a master enu we should be able to load ptions by doing a binary load on hatever screen we saved the riginal application to, as once hese words are loaded into the ictionary on top of what is already here they will be in the same ontinuity they were in when we aved them. When we want to return o the master menu we should in heory be able to FORGET back to the eginning of each application eaving the core words intact ITHOUT having to COLD the system.

reat in theory, but I must be doing omthing wrong . I can get from the aster menu into the application, un it OK, but crash horrifically hen I try and FORGET the pplication and return to the title creen. HELP HELP!

SIDE
----

o you get sick of that sluggish ursor needing repeated tappings of he keys to move along/up/down? Well on't despair. There are several odified versions of TI Forth with n autorepeat editor in the Public omain eg one may send to:



```
R #2
                         T I   F O R T H
              THIS VERSION OF THE FORTH LANGUAGE
                 IS BASED ON THE fig-FORTH MODEL
          TEXAS INSTRUMENTS PERSONNEL WITH SIGNIFICANT
                INPUT TO THIS VERSION INCLUDE:
         LEON TIETZ  LESLIE O'HAGAN  EDWARD E. FERGUSON
################################################################
      This version of TI-Forth was modified and extented by
           Michal Jaegermann, Edmonton, Alberta, CANADA

It is in the public domain and you may copy and give it away
but not sell it. If you would like to obtain a copy of the disk
send a disk, mailer, and contribution to:
            EDMONTON 99'er Computer User's Society,
      P.O.Box 11983, EDMONTON, Alberta, CANADA T5J 3L1
NOTE: Do not send US postage stamps!
```

And receive a disk. This particular version has slightly different and perhaps more awkward control keys to our original one but I suppose one gets used to these ok. Alternately one can snaffle such an editor from one of the many existing Forth programs floating around in the public domain by VLISTING the terminal and noting the last word above the editor (or you may keep the lot above it if you want), FORGETting back to this word and then overlaying your binary code on top of this. I snaffled one out of an excellent CALENDER program available through the club by author unkown. I was initially delighted but in the last couple of days it seems to be locking up all the time. I'm hoping that now I've changed the console it will be ok.

If anyone in Australia who reads this article has the source code to one of these could you post it to us. I KNOW SOME OF OUR ARTICLES HAVE BEEN REPRINTED IN OTHER MAGAZINES OUT OF STATE. THANKS EVER SO MUCH FOR THE ACKNOWLEDGMENT! WE'D JUST LOVE TO GAIN SOME RECIPROCAL ARTICLES TO RE-PRINT HERE.

Another mention of VLIST. This is an incredibly useful word to see where you are in applications with problems. Additionally I find it useful to keep a hard copy of all my words in the disk sleeve as I am increasingly finding so many versions of non standard Forth core words around one can forget what is in ones core.

Type SWCH 27 EMIT 15 EMIT VLIST CR UNWCH to miniaturise your printout into condensed print.

BASIC DISK CATALOGGER.

Well, I did open my big mouth and promise you a listing, so I'll have to cough up. Since this months column is a problems unsolved column I'll add I suspect this program is not without its bugs. This is V 1.0, I have a more sophisticated one around on the boil. Also it is set up for 2 drives so you will have to put a CONTINUE in to allow a pause before it trys to access your Forth disk if you have one drive and change the file descriptor on scr#60 line 13 to 1 for disk 1.

Once again, apart from what one can

learn about accessing disks from FORTH this is a useless excercise to fine tune into a whole program. It is pointless writing yet another disk manager when so many excellent ones abound. It does have its uses however as when developing a FORTH program which uses a BASIC file if we load the CATALOG program into the dictionary as we program above it we can check very easily what our development is doing to our basic disk without having to change to the Extended/Basic environment and then re-boot to FORTH all the time.

When developing this program I did so by trial and error by opening the disk index using OPN REC-NO RD etc, listing the bytes and seeing what came out and trying to make sense of the code. I believe though I am using one record at a time, the machine dumps a whole sector. Too sophisticated for me. Ask Tony McGovern the details.

Basically here I set up the appropriate space for dumping up to 127 records (its capacity), open the file, dump them to an array and then do a sort of VDP write routine to quickly dump them to screen.

REFER TO SCREEN 60/61

```
SCR #60
  0 ( BASIC CATALOG EXAMPLE 08Mar86)
  1
  2  0 VARIABLE DRV
  3  0 VARIABLE BUFR       48 ALLOT
  4  0 VARIABLE STORE    2413 ALLOT
  5  0 VARIABLE FILES-NO    2 ALLOT
  6
  7
  8  : MSG1 10 18 AT ." BUILDING FILE LIST" ;
  9
 10  HEX    PABS @ A + BUFR 1700
 11         FILE CATFIL
 12  : CAT# CATFIL SET-PAB INPT
 13         RLTV INTRNL F-D" DSK2." ;
 14 DECIMAL
 15
```

The buffers are set up to contain the drive number, BUFR to intially take the record from disk (size here is inaccurate-I just made it big enough)

STORE to take 127 record lenghs-if necessary one could shorten this manyfold if none of your disks contain more than a say a dozen files. Lines 11 and 12 set up the

```
SCR #61
  0 ( BASIC CATALOG EXAMPLE 08Mar86)
  1 : HUNDREDS 1+ DUP C@ 100 * SWAP 1+ C@ + ;
  2 : ST  13 + DUP C@ 65 = IF HUNDREDS ELSE 1+ C@ THEN ;
  3 : AV  16 + DUP C@ 65 = IF HUNDREDS ELSE 1+ C@ THE" ;
  4 : HEADING  10 12 AT ." Catalog Disk"
  5            10 14 AT ." Drive No.:2"
  6            10 15 AT ." Disk Name:" ;
  7 : GETDRV  20 14 AT KEY DUP EMIT 48
  8           - DUP 1 < IF MYSELF ELSE
  9           DUP 2 > IF MYSELF ELSE  THEN THEN DRV ! ;
 10 : SETUP PAGE ."      BASIC DISK CATALOG"     CR CR
 11              ." DSK" DRV @ . ." Diskname:" STORE COUNT TYPE CR
 12              ." Available:" STORE AV .
 13              ." Used:" STORE ST STORE AV - . CR CR
 14              ." Filename   Size   Type      P" CR
 15              ." --------   ---- ----------- -" ;
```

guts of the PAB as discussed in a previous article.

CAT#:this data is contained in a definition and included before accessing any basic file. One could leave it outside a definition, but if you use say the printer in between accessing a disk the PAB no longer points to your disk and all is lost in a rather heavy system crash!

REFER TO SCR# 75

Load the catalog program and run it to make sure its working. Then load screen 75. This opens the disk index and inputs the first 35 bytes of whichever record you request. Insert say FUNLWRITER (not your original) and type:

O BYTECHECK:

```
SCR #75
  0 ( BASIC CATALOG EXAMPLE 08Mar86)
  1
  2 : BYTECHECK CAT#                 ( refer to catalog file
  3           OPN REC-NO RD CLSE ( opn read record and close
  4           CR BUFR            ( carr return/bufr adr to stack
  5           35 0               ( loop indices
  6               DO             ( start of do loop
  7                  ." BYTE " I . ." ="
  8                  DUP         ( duplicate start adress bufr
  9                  I +         ( add loop indice ie offset
 10                  C@          ( fetch byte from this adr
 11                  .           ( print byte to screen
 12                  CR          ( another carriage return
 13                  CONTINUE    ( wait for bar press
 14               LOOP           ( end of loop
 15                      ;       ( end of definition
```

Lo and behold:

```
BYTE 0 =10 character count
BYTE 1 =70 F
BYTE 2 =85 U
BYTE 3 =78 N
BYTE 4 =76 L
BYTE 5 =87 W
BYTE 6 =82 R
BYTE 7 =73 I
BYTE 8 =84 T
BYTE 9 =69 E
BYTE 10 =82 R
BYTE 11 =8
BYTE 12 =0
through
BYTE 19 =0
BYTE 20 =8
BYTE 21 =65 next digit is 100's
BYTE 22 =7 ie 7 * 100
BYTE 23 =18 plus 18
```

BYTES 22/23 contain the disk sector size in my case here double sided single density of 718 sectors.

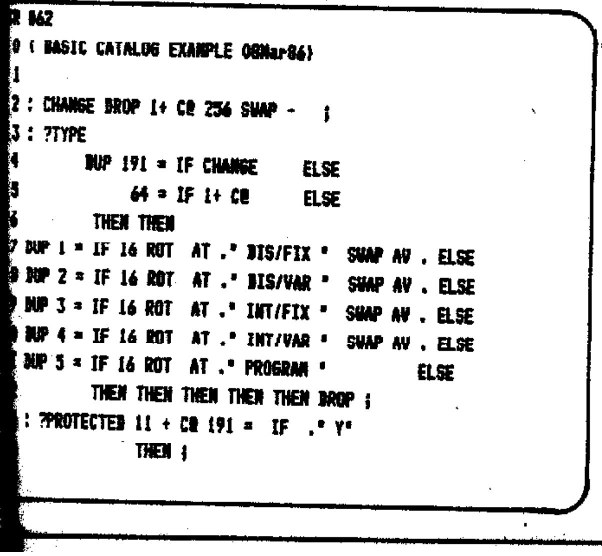
BYTES 24 through 28 =0

BYTE 29= 8
BYTE 30= 65
BYTE 31= 3 ie 323 sectors used BYTE 32= 23
his may be more than yours because y disk has other things on it.

ow if you repeat the process yping:

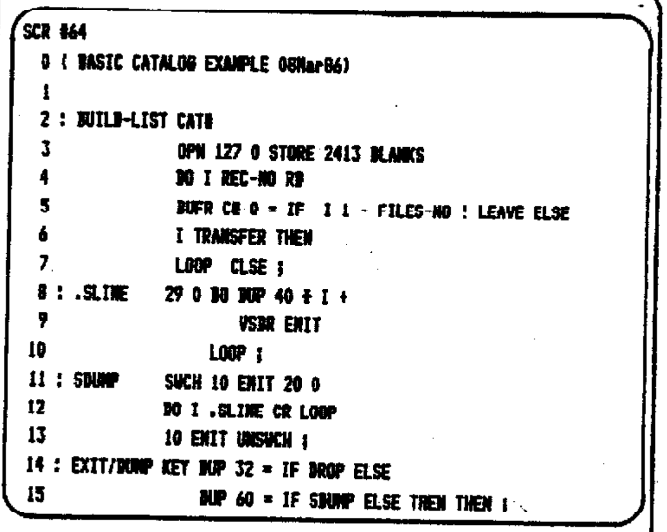
BYTECHECK:

t will be similar with the first yte containing the file name count, he ensuing bytes with the file ame. At the end of the filename he next 3 bytes have the code for he program type.



```
R #62
( BASIC CATALOG EXAMPLE 08Mar86)

: CHANGE DROP 1+ C@ 256 SWAP -   ;
: ?TYPE
       DUP 191 = IF CHANGE     ELSE
            64 = IF 1+ C@      ELSE
        THEN THEN
DUP 1 = IF 16 ROT  AT ." DIS/FIX "  SWAP AV . ELSE
DUP 2 = IF 16 ROT  AT ." DIS/VAR "  SWAP AV . ELSE
DUP 3 = IF 16 ROT  AT ." INT/FIX "  SWAP AV . ELSE
DUP 4 = IF 16 ROT  AT ." INT/VAR "  SWAP AV . ELSE
DUP 5 = IF 16 ROT  AT ." PROGRAM "          ELSE
        THEN THEN THEN THEN THEN DROP ;
: ?PROTECTED 11 + C@ 191 =  IF  ." Y"
            THEN ;
```



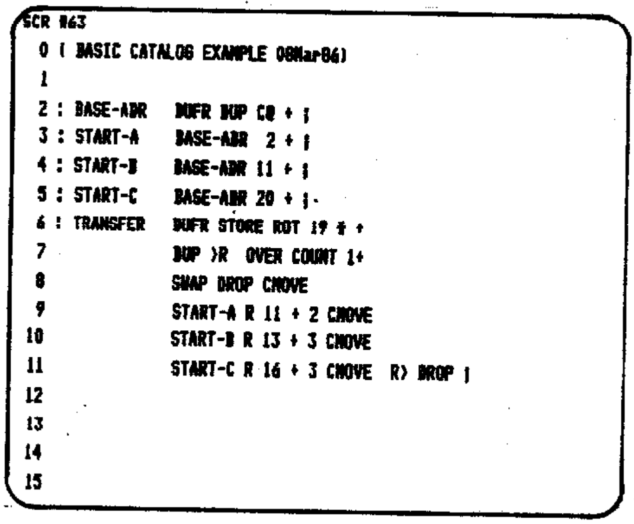
```
SCR #64
 ( BASIC CATALOG EXAMPLE 08Mar86)

 : BUILD-LIST CAT#
             OPN 127 0 STORE 2413 BLANKS
             DO I REC-NO R@
             BUFR C@ 0 = IF  I 1 - FILES-NO ! LEAVE ELSE
             I TRANSFER THEN
             LOOP  CLSE ;
 : .SLINE    29 0 DO DUP 40 * I +
                   VSBR EMIT
                 LOOP ;
 : SDUMP     SWCH 10 EMIT 20 0
            DO I .SLINE CR LOOP
            10 EMIT UNSWCH ;
 : EXIT/DUMP KEY DUP 32 = IF DROP ELSE
                DUP 60 = IF SDUMP ELSE THEN THEN ;
```

REFER TO SCR# 64

The word BUIL-LIST does most of the work. It uses CAT# to ensure the correct PAB is reference and OPN(s) the file and BLANKS the entire array (STORE) where we will put our information on each record. using the DO LOOP and the loop index I it sequentially reads each record into BUFR our transit house, checking if the first bite is 0 (ie no more records). If it is it leaves the loop. If it is not it uses TRANSFER to move the data from the temporary BUFR to our larger STOREhouse and deposits the record count in the variable containing the number of files on disk (FILES-NO) for later using in reprinting the correct number. The file is then CLSed.

REFER TO SCR# 63.



```
SCR #63
 ( BASIC CATALOG EXAMPLE 08Mar86)

 : BASE-ADR   BUFR DUP C@ + ;
 : START-A    BASE-ADR  2 + ;
 : START-B    BASE-ADR 11 + ;
 : START-C    BASE-ADR 20 + ;
 : TRANSFER   BUFR STORE ROT 19 * +
              DUP >R  OVER COUNT 1+
              SWAP DROP CMOVE
              START-A R 11 + 2 CMOVE
              START-B R 13 + 3 CMOVE
              START-C R 16 + 3 CMOVE  R> DROP ;
```

This screen does alot of work. In concept it is very useful in all sort of programs because it takes data in a constant format from a small buffer and adds it to a larger buffer giving an array. Within the array all elements bear a constant relationship making it easy to extract information in whole record/part of record form.

If you examine the byte structure as above using BYTELIST you will notice that as disknames/filenames vary in lengh there is no constant position of the various data eg program size etc from the START of each record.

Once past the name all the elements are at a constant distance apart -9 bytes. BASE-ADR calculates the end of the name bytes then START-A will be the start of information about program type, START B the program size and START C for lengh of DIS-FIX records etc.

To keep the elements in my array in STORE constantly related I have left 11 bytes (count + max 10 for a name) for each record name even when it may only be a few long. TRANSFER then moves only the essential couple of bytes into the array with the informaton on size/type etc, again in a constant relationship.

Due to pressures of space and time, and because this is a very important topic which bears detailed examination I will leave it till next month to examine in detail.

A SIMPLE SCREEN DUMP

The remaining words on SCR# 64 pertain to a simple dump routine to the printer.

SLINE prints a single line from the screen 29 characters wide(all that is needed for the catalog) by using VSBR (VDP single byte read) command and EMITing the result to the printer

SDUMP : SWCH's to the printer, 10 EMIT puts out an initial single line feed for the simple reason that my bloody printer always refuses to do an intitial carriage return hence overwrites lines one and two! then reads the 20 necessary lines(an arbitrary figure) using DO LOOP and UNSWCH's the printer and returns us to the terminal.

```
SCR #65
( BASIC CATALOG EXAMPLE  08Mar86)

: .SIZE  DUP ST . ;
: WLOOP FILES-NO @ 1 + 1
        DO I 19 * STORE + DUP
           1+ I 6 + 40 * 10 VMBW
           12 I 6 + AT .SIZE
           DUP DUP 11 + DUP C@
           I 6 + ROT ROT  ?TYPE
           28 I 6 + AT ?PROTECTED
        LOOP ;

: .NAME 20 15 AT CAT# OPN 0 REC-NO RD  CLSE BUFR COUNT TYPE ;
: CATALOG HEADING GETDRV .NAME MSG1
          BUILD-LIST SETUP WLOOP
          EXIT/DUMP ;
```

```
SCR #66
( BASIC CATALOG EXAMPLE 08Mar86)

: DISKSCREEN CLS 3  0 AT ." BASIC DISK MANAGER - DISK UTILITIES"
                 3  3 AT ." Select Option:1"
                10  5 AT ." 1.Catalog    3.Initialise"
                10  7 AT ." 2.Copy Disk  4.Master menu" ;

: DCHOICE 17 3 AT KEY DUP EMIT 48 -
          DUP 1 = IF   CATALOG         ELSE
          DUP 2 = IF   NOP             ELSE
          DUP 3 = IF   NOP             ELSE
          DUP 4 = IF   QUIT PAGE       ELSE
          THEN THEN THEN THEN  DROP ;

: RUN DISKSCREEN DCHOICE MYSELF ;

```

```
SCR #67
 ( BASIC CATALOG EXAMPLE 08Mar86)



 60 LOAD 61 LOAD 62 LOAD 63 LOAD
 64 LOAD 65 LOAD 66 LOAD










```

Note the program is not elegantly set up to take into account disks containing more entries than wil fit on a single screen. You ca easily modify if so desired.

EXIT/DUMP waits for a space bar t continue or a f'n 0 to dump th screen to your printer.You coul better incorperate this into DO...UNTIL loop.

REFER TO SCREEN # 65 66 67

Due to pressures of time and space I will leave elaboration to another day. WLOOP uses the FILES-NO variable to leave the upper loop indice on the stack, loops through the array and reprints the data to the screen impressively quickly.

```
SCR #51
 0 ( STRING EXAMPLE -Messages 18Jly85)
 1
 2
 3 : MSG1 5 12 GOTOXY ." Moving Strings" 5 14 GOTOXY
 4        ." From Bufl to Buf2" ;
 5 : MSG2 5 12 GOTOXY ." Adding Strings" 5 14 GOTOXY
 6        ." From Bufl to Buf2" ;
 7 : MSG3 8 23 GOTOXY ." Press any key" CONTINUE ;
 8 : MSG4 5 12 GOTOXY ." Comparing Strings" 5 14 GOTOXY
 9        ." In Bufl to Buf2" ;
10
11
12
13
14
15
```

```
SCR #52
 0 ( STRING EXAMPLE -Accepting   18Jly85)
 1 0 VARIABLE BUF1 20 ALLOT   0 VARIABLE BUF2 20 ALLOT
 2 : BLOT  0 12 160 32 HCHAR  0 23 40 32 HCHAR  5 20 15 32 HCHAR
 3        5 21  15 32 HCHAR 25 20 15 32 HCHAR 25 21 15 32 HCHAR ;
 4 : PROMPT 5 12  GOTOXY ." Enter Strings"
 5         5 14  GOTOXY ." First  string:"
 6         5 15  GOTOXY ." Second String:" ;
 7 : BLANKBUFS BUF1 20 BLANKS BUF2 20 BLANKS ;
 8 : GETBOTH BLANKBUFS PROMPT  19 14 20 32 HCHAR 19 15 20 32 HCHAR
 9                             19 14 GOTOXY BUF10 32 HCHAR 19 15 20
10  32 HCHAR                   19 15 GOTOXY BUF2 10  GET$ ;
11 : BUFAFTER 25 20 GOTOXY BUF1 COUNT TYPE
12           25 21 GOTOXY BUF2 COUNT TYPE ;
13 : ENTER$ PROMPT GETBOTH BUFAFTER MSG3 BLOT  ;
14
15
```

```
SCR #53
 0 ( STRINGS EXAMPLE -Move & Add 18Jly85)
 1 : BUFBEFORE 5 20 GOTOXY BUF1 COUNT TYPE
 2           5 21 GOTOXY BUF2 COUNT TYPE ;
 3 : DELAY 10000 0 DO NOP LOOP ;
 4 : ?SAME$  MSG4 BUFBEFORE MSG3 CONTINUE BUF1 BUF2 SAME$ BUFAFTER
 5          5 15 GOTOXY 1 = IF ." STRINGS IDENTICAL" ELSE
 6          ." STRINGS DIFFERENT" THEN MSG3 DELAY CONTINUE BLOT ;
 7 : DOMOVE MSG1 BUFBEFORE MSG3 BUF1 BUF2 MOVE$ BUFAFTER
 8         DELAY   MSG3 BLOT ;
 9 : ADDTHEN MSG2 BUFBEFORE MSG3 BUF1 BUF2 ADD$ BUFAFTER
10          DELAY MSG3 BLOT ;
11 : BUFSCREEN  0 18 GOTOXY ." Contents Before"
12           20 18 GOTOXY ." Contents After"
13            0 20 GOTOXY ." Buf1:" 20 20 GOTOXY ." Buf1:'
14            0 21 GOTOXY ." Buf2:" 20 21 GOTOXY ." Buf2:" ;
15
```

```
SCR #54
 0 ( STRING EXAMPLE -Subsegments 18Jly85)
 1   VARIABLE FROM 2 ALLOT 0 VARIABLE TO 2 ALLOT
 2
 3 : SEGHEADING 5 12 GOTOXY ." String Segments"
 4              5 13 GOTOXY ." Enter String:"  BUF1 15 GET$
 5            5 14 GOTOXY ." Segement from:  For:   Characters " ;
 6 : ?FROM 19 14 GOTOXY KEY DUP EMIT 48 - FROM ! ;
 7 : ?TO   25 14 GOTOXY KEY DUP EMIT 48 - TO   ! ;
 8
 9
10
11
12 : SEGMENT SEGHEADING ?FROM ?TO  BUF1 BUF2 FROM @ TO @ SEG$
13          5 15 GOTOXY ." Segment is:" BUF2 COUNT TYPE CONTINUE
14          BLOT ;
15
```

```
SCR #55
 0 ( STRING EXAMPLE -Main menu 18Jly85)
 1 : OPTIONS  5 0  GOTOXY ." STRING HANDLING WORDS"
 2            5 2  GOTOXY ." Select Option:"
 3            5 4  GOTOXY ." 1.Enter Strings"
 4            5 5  GOTOXY ." 2.Move from bufl to buf2"
 5            5 6  GOTOXY ." 3.Compare Strings"
 6            5 7  GOTOXY ." 4.Concatenate join Strings"
 7         5 8  GOTOXY ." 5.String Segment" 5 9 GOTOXY ." 6.End" ;
 8 : CHOICE  19 2 GOTOXY KEY DUP EMIT 48 -
 9          DUP 1 = IF ENTER$    ELSE
10          DUP 2 = IF DOMOVE    ELSE
11          DUP 3 = IF ?SAME$    ELSE
12          DUP 4 = IF ADDTHEN   ELSE
13          DUP 5 = IF SEGMENT   ELSE
14          DUP 6 = IF COLD      ELSE
15          THEN THEN THEN THEN THEN THEN DROP MYSELF ;
```

```
SCR #56
 0 ( STRING EXAMPLE -Load screen)
 1
 2
 3 PAGE ." LOADING STRING EXAMPLE-PLEASE WAIT"
 4
 5 51 LOAD 52 LOAD 53 LOAD 54 LOAD
 6 55 LOAD
 7
 8 : GO CLS OPTIONS BUFSCREEN CHOICE ;
 9
10
11
12
13
14
15
```

Well thats it for another month. Hope I havn't left you too confused. Next time we may expand on arrays and accessing them together with a foray into the realms of VDP reads and writes and some application of them.

ADDRESS FOR CORRESPONDENCE:
RICHARD TERRY
141 DUDLEY RD
WHITEBRIDGE 2290
NSW 049 436861/22450.

LIBRARY NEWS WITH AL LAWRENCE

HI 99'ers,

Disk news

FUNNELWRITER V 3.0 has been improved yet again!!. Function 9 now brings the Utility Menu back for Selection from anywhere EXCEPT after a QUIT action. Option 5 Switches the Option 2-Choice from FORMATTER to ASSEMBLER and depending which is selected when choosing Option 1 it tailors the EDITOR to suit.
This is V3.1 and is available NOW.

COLLIST has now been made available to the public domain.
If you do not know what COLLIST does it will list your Programs into 2,3 or 4 columns when you want a Hard Copy for De-bugging. Thus saveing paper and doing away with the need for arms 2 Metre's long. It also is used in producing this Newsletter, printing programs that work without the need to Re-type in a suitable format. It also prints out the last page columns into equal sizes. This is the greatest walking, Talking, Usefull Utility I have in the Club Library with a sense of Humour to BOOT. Again it is brought to you out of FUNELLWEB FARM and if you buy the Disk, at no extra charge is a game that received 5 star raves in Computor Magazines and sold for heaps of the folding stuff

Library Access

Any clubs or indivuals interested in obtaining any PUBLIC DOMAIN software in volume disks have 2 choices

(a) Send blank initialised disks to us with return postage or send us disks with programs on it and we will send at our cost an equal number of disks filled with programs requested or volume disks.

(b) We can supply programs, or volume disks on our disks for the cost of disk and PE(\$4.00)

PUBLIC DOMAIN DIS

#	TITLE	SOURCE	COMME
001	FLWTR V3.1	HV99'ers	McGove
002	COLLIST	HV99'ers	McGove
003	XB.Tutorls	HV99'ers	McGove
004	JW-DISKCAT	HV99'ers	J.Wri
005	HV99-PDL01	HV99'ers	Vari
006	TIWR+MPLAN	TEXAS-INST.	UGPD
007	SSPD1	S.Shaw(U.K)	UGPD
008	DM-1000	OTTAWA 99.UG	B.Ca
009	NO-FRILLS	PD85(TIshug)	B.Ca
010	NEATLISTER	FREEWARE	Mich
011	FREEDUMP	FREEWARE	Mich
012	ALROUTINE1	NEW-HORIZONS	Clul
013	ALROUTINE2	NEW-HORIZONS	Clul
014	PROGRAMS	CHANNEL 99'	Vari
015	4th.		XB-1
016	4th.		MM-1
017	4th.Scode	TEXAS-INST	2 Dis
018	4th.	TEXAS-INST	UGPD
019	4th.Calndr	San Diego	Winkl
020	4th.Acount	San Diego	Winkl
021	4th.Demo	AMNION Free Access	
022	A*38	AMNION "	Game
023	A*39	AMNION "	Game
024	E* 8	AMNION "	Bus/
025	F* 7	AMNION "	TecM
026	G* 6	AMHION "	PerH
027	UK-PD3/85	TI*MES (U.K)	Vari
028	UK-PD4/85	TI*MES (U.K)	Vari
029	XBDataBase	SLAVE(UTAH)	V.Par
030			

D-I-Y Pitfall

For all those who after explor the innards of the console or a cleaning the Module port to Lock Up. Or maybe after instal the 32K expansion in the console may find some of the Keys on Keyboard may not work. This is to the short inflexable connec from the Keyboard to the Motherb open circuiting (breaking with bending).
The cure is simple, unsolder all connections to the Keyboard either replace the orignal c after preparing the 15 ends removing the old solder from Keyboard, or renew with a flexiable type.
There is another problem with Keyboard, that of 2 or characters printing out while ty normaly (not holding the key d this is due to dry joints or cr in the solder on the board at connections. This flexes typing causing the extra charac to be printed out. Again reso the joint and the problem sh dissapear.

Happy Computi
Al Lawrence.

ENTOMOLOGY CORNER FROM FUNNELWEB FARM

BY
TONY MC. GOVERN HV99

Well, here we are back for the New Year. I know there has been an issue of the Newsletter already this year, but any Editor who wants copy for a bi-monthly production at the end of January in Australia will get just about the response deserved (sorry about that Steve). Since coming back from holidays in Melbourne and assorted National Parks in between, we have finished the new issue of FUNLWRITER and it is now available from the club librarian for for the usual very nominal fee.

It turned out that the funnelweb sized bug we were fighting at last report was really two bugs. One was a tricky bug in my own code which was readily understandable when finally isolated. The other bug, whose effects were mixed up with the first seems to be plumbing the undocumented depths of the machine in its CRU keyboard setup, and the only cure was to avoid stirring it up. It was associated with sensing of single function key entries by use of TB instructions. You have already seen this in the SD routine that FUNLWRITER provides for the Editor where it causes no ill effects. Use elsewhere in the program was, after reloading of the Editor or Formatter, disabling cursor flash and autorepeat, while everything else remained normal: Very strange indeed ! I did find a fix - I won't dignify it with a better name - but in the absence of deeper understanding of either the bug or the fix, I decided just to walk right around that particular funnelweb hole. Efforts to isolate it even included translating the 9901/CRU initialisation routine in the monitor power up code from GPL into an assembler routine, without making any apparent difference I might add.

The current issue is Vn 3.2, an update of the Vn 3.0 announced by Al in the last issue. Actually the program is being updated continually with minor bug fixes, or little improvements as often as not resulting from suggestions from people using the program. I will usually grumble that it can't or won't be done, but if it is a reasonable and sensible suggestion it will be thought about. Sometimes what look like minor changes on the surface require major changes underneath. However it's not the size of the changes that determines the issue of a new version number. Vn 3.1 has been distinguished from 3.0, not because of any accumulation of externally apparent changes, but because an assembly language interface has been fixed for the future that will allow programs written to run with FUNLWRITER to call on internal routines such as the file name mini-editor, so long as it is to be run with 3.1 or later version. The details will be given to assembly programmers wishing to do this. Vn 3.2 adds some new features One of these, instant re-entry to the Editor(s) on reselection after exit required substantial modification to the Editor files which are now incompatible with Vn 3.1 and have been renamed. The utility load options 1-3 which handle program files will now load these from cassette if filename CS1. is given. The process is very similar to that described in the E/A manual - you ignore the instruction to rewind the tape in between loads of multiple files. It is now even possible to load the TI-Writer Editor completely from cassette on a machine with 32K but no disks and print out from it directly, but not to store or retrieve files from cassette. This last could very well be added as well to a cassette version, but is something I feel no motivation to do. It's rather like Dr Johnson's talking dog.

Unlike V2.1 which was listed as Public Domain, V3.0 and later remain in the copyright of the authors, with no rights relinquished. They are however given out on condition that they be passed on freely in the original form, and users of the program are invited to make their appreciation known in tangible form at a level they think appropriate. This approach was known as "freeware", but like much California psycho-babble it has its nastier aspects. Not the least of these is that a California publisher has registered the name and it cannot be used freely. User group newsletters in the US are canvassing alternative terms, "fairware" being one frequent suggestion. We won't look too askance at HV99 members who don't cough up, as it's one of the minor benefits of being a member of the most active TI user group in Australia (after that comment I am just waiting for the Trans Continental Missile from the West). Just take it as a hint that there can never be too many people in a group doing their share or more.

If you pass on the program please do so exactly in the form in which you obtained the distribution disk from the library, which you should keep as your personal backup copy anyhow. If you have your own working version with other files on it, please keep that as a separate item, since the HV99 group name is on the program. I am aware from feedback from the USA that this request has not always been observed. Also I have deep concern and annoyance that a test version complete with serious bugs, released locally for private test and report only, has leaked out. It is of course now obsolete, but the damage has been done.

I even have reason to believe that the ante-diluvian original version without a number is floating around on USA bulletin boards. This is the one with 4 choices on the title screen, the last being for an XB disk directory routine. This was eliminated from the official V2.1 release to speed the initial load of the program, as the SD function in the Editor did the job anyway. Nevertheless the lack of a really quick SD function has been felt ever since, and the solution is now to hard in the form of the SD routine in Clint Pulley's small-c compiler, an enormously interesting "freeware" offering from Canada. The files on this disk will not run from FUNLWRITER because they assume the presence of the E/A GROM, but can be modified to do so. This is particularly easy for SD as its source code is supplied and I hav done so for my own working disk. I will not be put on the distributio disk for lack of room and mor importantly because we have not ye been in direct two-way contact wit Clint Pulley. In the meanwhile can tell anyone who wants to ru small-c under FUNLWRITER th necessary code patch to th compiler, and give them the modifie SD for their FUNLWRITER workin disk.

I think Canada is a bit lik Australia in that programs an articles produced range enormousl in quality. Unfortunately one o the less satisfactory items fro Clubline 99 was reprinted in las month's issue. This colum "Entomology Corner" started out a an occasional bug report befor metamorphosing into a more genera format, so we will now revert to th original form as an altogether to large fraction of that article need bug squashing. Collections o "tips" or "hints" are a form that consider an abomination, mos especially when some are wrong o conditions for relevance not given.

Followers of the XB Tutorials in th past will of course pick up some o the silly season aspects of th article immediately, while so other things have not yet be covered. There are also so perfectly good parts in th curate's egg of an article. Anyw let's work our way through it detail. Of course it may have be a subtle plot on the Editor's pa to stir me into writing on XB aga but I'm sure he won't fess up.

The first recommendation is to u an array to hold screen values that an array reference can be us rather than GCHAR. The gains fr this are overstated since the arr must also be written to as well read from. It still may worthwhile provided the program is to be run with 32K memory expansion. It is on balance the wrong way to go in programs to run in unexpanded consoles. The first reason is one of program size. The array suggested takes over 6K bytes, almost half the available memory which is already too small for serious programming. One of the major design decisions in writing game programs for console only is whether to use the screen image table area of memory as the implicit storage for what is going on on-screen using GCHAR to retrieve items. A game of any complexity will usually force the decision, and it is best taken early in the piece. It is not necessarily such a bad way to go for speed either, for though GCHAR is no ball of speed, the overhead of array references is worse when XB runs in console only, and in my experience can even end up slower than sensible use of GCHAR. Remember that if the 32k is not available for XB, then all numeric references have to be fetched from VDP RAM also.

Item #2 is sound advice. Always remove unnecessary calculations from inside loops, and even if a slow calculation is necessary, look at the possibility of pre-calculating a look-up table.
Item #3 was discussed in the last Tutorial. The statement on speed is far too weak. Now for the "little hints". In a well written program use of multiple statement lines makes only a marginal contribution to speed. The real reason is to save bytes as discussed in a recent Tutorial. The next item is a little curate's egg all on its own. As suggested avoid the use of DEF, not because it is inherently a bad way to go, just the opposite in fact, but because the TI implementation is so hopelessly slow. On the other hand the advice given on user SUBprograms is just wrong headed as any follower of the Tutorials would recognize. SUBprogram overhead is a worthwhile price to for the benefits in program organisation and may even improve speed by breaking a large monolithic program into smaller pieces. SUBprograms were TI programmers' finest contribution to XB unlike DEF which was left by the wayside.

The third little hint may be useful depending on the situation, though it would seem to be something XB programmers would normally do anyway. It can even be counter productive in subtle enough circumstances. The fourth is mostly codswallop. The prime reason for using GOSUBs is to save bytes, and in a language that supports proper procedures as well as XB does there is no reason to use GOSUBs at all except to save bytes. If a GOSUB doesn't do that there is no reason to incur its execution overhead or code confusion.

Now that those details are disposed of, we should have a look at what the future holds. One thing that I think is certain is that for all its problems and limitations, we are going to remain with the present console until the computer as a whole just fades away, finally obsolete rather than worn out. Reports of a new design based on the TMS9995 will never get much beyond vague reports. That approach died when TI canned the 99/8 and it wasn't taken up immediately elsewhere. As it is we are only just starting to see home computers that offer any real improvement on the 99/4a system (and by that I mean the full system - the console by itself is little more than a toy). The Commodore 64 came along later and is now fading from the scene. It was a better games machine but an inferior as a computer system. The C128 has only a little more to say for itself. Apple II's have always been ludicrously overpriced and the Macintosh is only just coming out in versions that do its processor justice, though still grossly overpriced. The original Macintosh had the smell that the TI-99/4 had several years earlier - made easy to use with many new and good ideas incorporated, but which crippled the powerful micro inside with a closed architecture and secretive attitudes contemptuous of the user. Still, besides its being too expensive for what it offers, that tiny display screen is no good for old fogies like me with fading eyesight. I like the old second hand 22 inch console TV up on the desk sitting over the PE box.

What else is there ? The Japanese MSX machines, excepting perhaps the specialist offerings, say Yamaha for music, don't seem to do much that the TI-99/4a couldn't have done if TI had got it right in the first place years ago. Any new TMS9995 based machine would have to compete with the next generation MSX II, but now without major industrial and marketing muscle to back it up. The Amstrad disk machines now look good value for someone starting a home computer system, but not worth dropping a TI system for. These too are oddball, even though they run CP/M their disk drives isolate you from the CP/M software world. At least they do have disk drives.

Then there is the whole world of IBM PCs and clones. Like the Macintosh this puts you into another price league, where software is priced to match the machine rather than on its intrinsic value. The cheaper clones, who knows of what quality, are now getting in reach on price and occasional pieces of software are to be found at reasonable prices. Such machines are not necessarily an improvement on what is already available. I gather that Dick Smith's current IBM clone from Taiwan runs Wordstar much slower than the locally produced Microbee Z80 machine. The IBM JX series falls in between, no doubt destined for the wealthier private school market, and no doubt bought with subsidies from my tax dollars that don't seem to find their way towards Lambton High's computer room. Now the TI versions look to be the best engineered of the lot - would we expect any less from the company which brought out the PE Box as part of a home computer - but are priced beyond the market being discussed here. If TI were to give me one of their machines or if one were to fall off the back of a truck passing Funnelweb Farm, I would start to write programs for it, but there is no way I could justify buying one out of my own pocket for home use. This leaves the Atari 520ST and the Commodore Amiga as the future prospects that look good enough to relegate the old 99/4a to the back room before it wears out.

Now let's take an assembly language programmer's view of the situation. One of the reasons I bought the TI system was to come to grips with assembly language in the comfort of home, after having avoided it ever since micros came on the scene (FUNLWRITER Vn 3.2 is the evidence that something has been learned). Over the years I have been a very occasional low level user of high level languages. My professional training was in EE and in retrospect I think that electronics engineers were the last group to make serious use of computers. Having learned on a 16 bit machine there is no way I could go back to a 8 bitter. The Z80 looks like a rococo nightmare to this 9900 programmer. Will, who has never known anything else came back from a session doing machine code on a friend's C64 with the comment in the delicate language of high school - "8 bit sucks !".

So what makes a particular micro-processor more satisfying to deal with than others ? They all do the job more or less well. There is a certain elegant symmetry with a 16 bit machine addressing 64K bytes - any word in the machine, in registers or memory can also be a valid memory address anywhere in memory and the converse is also true. Now what about 8 bit machines ? Well, 8 bits really only address 256 bits of memory, so you have to do special doubling up tricks of one sort or another to get to 64K. I think the reason that the 6502 processor and minor variants has been such a success (Apple, BBC, C64 etc) is that it allows fast operation with a limited operation code set in a 256 byte page of memory addressed by the 8 bits.

So where to go beyond 64K. Any 16 bit machine will have to resort to special tricks. The 9900 series has to do it with external hardware and has the CRU bus for control, while the Intel 8088 etc have a segment register that controls paging. In either case an address space of say a megabyte is not really a clear wide open megabyte, but a collection of 64K byte pages which programs can only access one at a time. Programs to use more than 64K start to get complicated and to have funny restrictions, or you find Basic on a machine with 256K of memory that can only use 64K. This is why I'm not really interested in the TI Professional or other IBM PC type machines -- if I am going to pay to break out of the 64K prison cell, I want to do it to the wide open spaces, not just into another bunch of cells on the same corridor. If the 99/8 had come out when its time was ripe, then we would have lived with it happily. Now however the horizons are wider with Motorola 68000 based machines with honest linear address spaces, from Commodore and Atari out there in the price range below IBM and Apple.

So is the future rosy ? Apart from wondering where the money will come from to move on and up, I do have reservations about the manufacturing quality to be expected of these. Commodore has always been infamous for poor quality control, and who knows about Atari now ? But this is a programmer's view here, so what of the Motorola 68000 ? The trouble is the complexity of its instruction set. If the Z80 is a rococo bad dream then the 68000 is positively Wagnerian - and I can't abide 19th century music. I think the 9900 instruction set is a very nice compromize between confusing complexity and brain damaged simplicity. That's the trouble with 32 bits - the designers have to resist the temptation to use the 32 all at once for increasing the sheer variety of instructions. The message of the 6502 is strong here - use short instructions running fast - and some newer designs have done just this. Kind of like a TMS9995 running from workspaces in its on-board memory when you come to think of it. Maybe by the time we can afford a replacement computer some of these new reduced instruction set processors may be available in a low cost machine.

I was reading a Hewlett Packard blurb article that arrived today (Mar/86) on their new computer architecture. Some curious comparisons there ! If you have been worried that the TMS9900 doesn't have an automatic stacking instruction for subroutine calls, then you will find H-P's latest effort interesting as it doesn't either, It does use R1 instead of R11. Stacks and register saving are handled by short routines. If you have ever disassembled the disk DSR ROM that won't seem an to be an unfamiliar method either.

That should be enough of this stream of TI-99/4a computer consciousness. Time to wrap it up and start pulling the machine apart to install the new DS disk drives which arrived a couple of days ago.

```
************************************
*      JUNIOR SOFTWARE REVIEW      *
************************************
```

TI-WORDS

TI-Words is based on the familiar board game, Scrabble and is played in the same way.

1. The board is drawn on the screen.
2. Players enter names
3. Computer initialises random letters for all players.
4. Game begins.

After play commences and you choose to enter a word it asks you a host of questions like what the word is and where it is to go. The computer then gives you a score which is totalled during play and at the end tells you which person wins.

The only disadvantage is that you can only build one word per turn unlike the board game where you can make as many words as possible from your letters. For a person who likes Scrabble why not give TI-Words a go.

By Simon Treloar HV99

FROM TOES TO THE TI

BY PAUL MULVANEY. HV99

Man first realised the need for a computing machine when he ran out of fingers and toes while working out mathematical problems. To help with counting, objects such as stones and seeds were used. As time progressed the seeds were placed on sticks supported by a surrounding frame, so the ABACUS was born, some time around 1000 BC. The Abacus is still a very useful portable type of computer because you can physically see the sum on the wires, - the position of the beads forms a 'memory' of the sum. The major drawback of the Abacus is that it is not automatic, requiring a fair degree of expertise to be proficient in its use.

Blaise Pascal, a Frenchman, invented the world's first mechanical calculator in 1642. It worked perfectly, carrying numbers from the units column to the tens column by a trip device, in the same way a cars odometer registers the kilometers travelled. The PASCALINE as it was called did not sell well but it did spark off great scientific interest. No significant advances were made until 1822 when Charles Babbage showed his DIFFERENCE ENGINE to the Royal Astronomical Society. Babbage, a mathematical genius, had grown frustrated at amending the many mistakes he found in logarithm tables and had built the machine to take the drudgery out of the calculations. The name derives from an abstract mathematical process known as the method of differences.

Together with Ada Lovelace, the daughter of Lord Byron, he set out on a more ambitious project to build an 'Analytical Engine'. This machine was designed to calculate values of mathematical functions that were far more complicated than the logarithmic functions. This machine was fraught with problems from the start. The drawings that have survived show that the construction was huge, filling a large workshop. The hundreds of cogs, rods and wheels had to be specially turned on lathes and current metal technology simply wasn't good enough. When Babbage had built his little model the minor inaccuracies it produced could be shrugged off, but once he tried to get the full-sized machine going, the minor inaccuracies became greatly magnified.

Babbage was on the right track, and if he had been able to get the parts machined sufficiently well, it is probable that his analytical machine would have worked. Much of the logical architecture and design structure of todays computers can be traced back to Charles Babbage and he is remembered as one of the founding fathers of modern computing. One important notion that occured to Babbage was that his engine could be 'programmed' or 'taught' to do any mathematical task. It wasn't until 1936 that proof was provided for Babbage's notion. It appeared in an obscure paper called 'On Computable Numbers', published by a young Cambridge mathematician, Alan Turing. Turing's contribution is fundamental to the development of the ideas that had to be generated before the computer could become a reality. Scientists had for a long time reasoned that mathematics was not a mysterious art but a science totally controlled by logical rules and that if you gave a machine these rules and a problem, it should be able to solve it. However, all the efforts of the most able mathematicians had failed to develop such a machine. Turing decided to approach the problem in a different way. He looked at the type of problem that a machine following logical rules could solve and tried to list them all. If they comprised the whole of mathematics then the conjecture would be solved.

Turing led a research team in Buckinghamshire and developed the most secret invention of the Second World War, Colossus, the world's first electro-mechanical computer. It was this machine that cracked the German 'Enigma' message codes throughout the war. One of the reasons why Turings name is virtually unknown is because he was working for MI6 and enveloped himself and his work in secrecy. The British government did not release details of Turing's pioneering work until 1975.

With electricity becoming more commonly available the development of electro-mechanical machines rather than purely mechanical devices was accelerated. The relay and the solenoid were the electro-mechanical devices used in the Mark 1, developed at Harvard University in 1944. It was a large machine (15.5 metres long and 2.4 metres high) which was fed instructions on punched tape and data was input on punched cards. The output results were recorded on cards by an electric typewriter. The Mark I could multiply two numbers in about three seconds. In 1947, the Mark II could perform the same multiplication in about 0.25 of a second. This was 12 times faster and at the time a great step forward, but compare it with the present day computers which can perform thousands of mathematical calculations in one second.

Something obviously happened to bring about this phenomenal increase. The invention of the

va
th
el
tu
mu
mi
19
sh
In
wa
ar
pr
pl
sw
re

The
fir
com
the
The
gen
of
much
cond
cool
requ
The
was
exce
majo
inte

The
was
from
aroun
less
requi
heat
the
been
were
circu
repla
downt
these
genera
more
used
same
genera

Third
appear
circuit
circuit
integra
Jack K
1958.
is a
interco
compone
compute
more r
used le

vacuum tube heralded the start of the electronic age. The first electronic computer used vacuum tubes and could perform multiplication in about 2.8 milliseconds. It was developed in 1946 and called ENIAC which was short for Electronic Numerical Integrator and Computer. The ENIAC was used by the Army to calculate artillery firing tables and was programmed by manually changing plug-in connections and setting switches which, as you can image, required a lot of time.

The UNIVACI was built in 1951, the first mass-produced electronic computer and signalled the start of the computer industry.
The vacuum tubes used in these first generation machines required a lot of electrical power and generated so much heat that many tons of air conditioning were required for cooling. This air conditioning also required a lot of electrical power. The reliability of these machines was so poor that down time often exceeded operating time. Another major drawback was the lack of internal memory.

The second generation of computers was ushered in by the changeover from vacuum tubes to transistors around 1959-60. The transistor was less expensive, more reliable, required less power, produced less heat and was much smaller than even the smallest vacuum tube that had been developed. The transistors were mounted on plug-in printed circuit cards which could be easily replaced to reduce maintenance downtime. The net result of all these factors was that the second generation computer was smaller, more reliable, less expensive and used less electrical power for the same computational power as a first generation machine.

Third generation computers began appearing around 1966 when miniature circuit modules and integrated circuits made their appearance. The integrated circuit was invented by Jack Kilby at Texas Instruments in 1958. Inside the integrated circuit is a small silicon chip with many interconnected transistors and other components. The third generation computers using I.C.'s were smaller, more reliable, less expensive and used less electrical power for the same computational power as a second generation machine.

The fourth generation began with the advent of the microprocessor. The microprocessor is a complete Central Processing Unit (C.P.U.) on one integrated circuit chip. More recent technological advances have permitted memory and input/output functions to be included on the same chip so that a total microcomputer is on one chip.

The TI-99/4A is a fourth generation computer, and it is interesting to compare its capabilities with that of the ENIAC. The ENIAC weighed 30 ton, took up 1,400 square metres of space, required a small power station to run it and cost almost a million dollars. The TI-99/4A can be carried by a small child, fits on a small coffee table, can be run all day for only a few cents, is several times more powerful, tremendously faster and costs less than $200.00.

The fifth generation computer - will it be self programming and here by the 1990's?

MAKING USE OF 32K

BY EVERT SMIES

EXTENDED BASIC ASSEMBLER PROGRAMS ON CASSETTE

*This article originally appeared in TI*MES, the magazine of the U.K. TI Users Group.*

Some time ago I was asked if it was possible to record assembler programs on cassette for use with the Extended Basic module. The problem arose because many people now have a D.I.Y. 32K RAM expansion without having a Disk Memory System.

At first sight the answer to the question seemed negative, but I have found a sort of half-hearted solution to the problem.

Unfortunately it is not (yet?) possible to record programs on cassette without disk drive, but once recorded the assembler programs can be loaded at will. So you need a friend with a disk drive for the following procedure.

The method is based on the possibility to MERGE Extended Basic programs. SAVE DSKx.*filename*, MERGE records an Extended Basic as a DIS/VAR 163 file. Each Basic instruction is assigned a "token", the ASCII value of which is written on file. Of course it is possible to make one's own file of ASCII values, which may be merged with another program.

From Peter Brooks' table you will remember that DATA has ASCII value 147, an unquoted string has ASCII value 200 and that the length of the string gets the corresponding ASCII value. A program line is ended by ASCII value zero and a program is terminated by two ASCII values 255.

Knowing this the rest is simple.

The following program loads the assembler program and then converts the assemble program into DATA lines in MERGE format by means of CALL PEEK instructions.

```
100 CALL CLEAR :: PRINT "FILE NAME
OF ASSEMBLER":"PROGRAM " ;:: INPUT
A$ :: PRINT : "OUTPUT FILE ";::
INPUT O$
110 OPEN #1:O$,DISPLAY,VARIABLE
163,OUTPUT :: CALL INIT :: CALL
LOAD(A$)
120 CALL PEEK(8194,A,B,C,D) :: LN=90
::      X$=CHR$(A)&CHR$(B)&CHR$(C)
&CHR$(D) :: GOSUB 200
140 AY=0 :: FOR AD=9460 TO A*256+B-1
:: GOSUB 180 :: NEXT AD ::GOSUB 200
160 AY=0 :: FOR AD=C*256+D TO 16383
:: GOSUB 180 :: NEXT AD :: GOSUB 200
170 PRINT #1:CHR$(255)&CHR$(255) ::
CLOSE #1 :: STOP
180 CALL PEEK(AD,X) :: AY=AY+1 ::
X$=X$&CHR$(X)   ::   IF  AY/150=
INT(AY/150) THEN GOSUB 200
190 RETURN
200 LN=LN+10 :: L1=INT(LN/256) ::
L2=LN-L1*256 :: PRINT #1:CHR$(L1)
&CHR$(L2)&CHR$(147)&CHR$(200)
&CHR$(LEN(X$))&X$&CHR$(0) ::X$="" ::
RETURN
```

Address 8194 (>2002) contains FSTLOW and LSTLOW, which indicate the length of the assembler program and the DEF table. The assembler program, provided is was relocatable, was loaded from address 9460 (>24F0) through FSTLOW-1. The DEF table runs from LSTLOW through 16383 (>3FFF). FSTLOW and LSTLOW are peeked first and put into the first DATA line with line number 100. Then the assembler program follows and the file is completed by the DEF table. Assembler program and DEF table are written in 150 byte DATA lines.

If the created DIS/VAR 163 file is merged with the following Extended Basic program, a new program is created which loads the assembler program from its own DATA lines. This new program may be saved both on diskette and on cassette. The latter is what we meant to do.

```
1 DIM A(3) :: CALL INIT :: AD=9460
:: READ X$ :: FOR I=0 TO 3 :: A(I) =
ASC(SEG$(X$, I+1, 1)) :: NEXT I ::
EA=A(0)*256 +A(1)-1 :: TA=A(2)*256+
A(3)
2 FOR I=1 TO INT((EA-9460)/150)+1 ::
READ X$ :: FOR J=1 TO LEN(X$) ::
CALL LOAD(AD, ASC(SEG$(X$,J,1))) ::
AD=AD+1 :: NEXT J ::NEXT I
3  AD=TA  ::  FOR  I  =1  TO
INT((16383-TA)/150)+1 :: READ X$ ::
FOR J=1 TO LEN(X$) :: CALL
LOAD(AD,ASC(SEG$(X$,   J,1)))  ::
AD=AD+1 ::NEXT J ::NEXT I
4 CALL LOAD(8194,A(0),A(1), A(2),
A(3)) :: STOP
```

Loading an assembler program in this manner takes much more time than loading a DIS/FIX 80 object file by CALL LOAD. An advantage, however, is a considerable reduction of required disk space (in one example 15 sectors instead of 31). Further to my knowledge it is the only way of saving an assembler program for Extended Basic on cassette.

The same trick may of course be applied to save assembler program for TI Basic and Editor/Assembler or Mini Memory on cassette. Owners of an Editor/Assembler module will however have a Disk Memory System and the Mini Memory itself allows saving assembler programs on cassette without the fuss of Basic programs.

Evert J. Smies
Meiendel 13
2036 HN HAARLEM
NETHERLANDS.

TI DISK OPERATING SYSTEM

BY: T.L. ATKINSON

TINNS U.S.A.

TI DISK OPERATING SYSTEM

by: T.L. Atkinson

Some new and old terminology:
TI DOS-denotes the standard TI Disk Operating System.
Nybble or nibble-4 bits.
Byte or bite-8 bits.
Word-16 bits.
RPM/S-revolutions per min/sec.
FDC-floppy disk controller.
TRN-Terminating Resistor Network.
SI-Sector Index (hole).
RW-Read Write.
WPN-Write Protect Notch.
OC-Optical Coupler.
1S (or SS)-Single Sided.
1D (or SD)-Single Density.
2S (or DS)-Double Sided.
2D (or DD)-Double Density.

Note.
The terms 1S,1D,2S,2D were coined by BASF and are fast becoming the more popular terminology.

GENERAL

A disk drive is a complex and precise piece of hardware, both mechanically and electronically. The disk drive spins a diskette at 300RPM, positions a RW head, and communicates with the FDC. Disk accesses are in a semi-random fashion; although tracks are accessed at random, the Sectors are accessed sequentially, as the disk spins.

GENERAL-TI

TI-DOS supports both 1S1D or 2S1D formatting. The latter, only if you have other than TI drives; (generally Shugart). You may have noticed, that, on the DMII, the options are available to format up to 2S2D. This is because TI was in the process of producing a 2S2D controller prior to it's demise. However, CORCOMP has come through, and produced an excellent FDC. Using this, along with a couple of half-height TEACS, gives the capability of 2S2D formatting. A total of (approx) 360K per disk, of online storage is possible. How did I arrive at this figure? Well, let's break it down. Normal TI drives (1S1D): 40 tracks 9 sectors per track and 256 BYTES per sector=90K bytes (approx). 2S1D would double that figure; 180K. 2S2D would double again, 180K*2=360K per disk. Of course, not all of this disk space is available for programs and data, but neither is the 1S1D TI format. This will become clear in later tutorials.

TRN SHUNT PAKS

The purpose of the TRN is to "pull up" the inactive FDC lines to +5v at the drive inputs. For optimum performance, these TRNs should be removed from all but the last drive. Some drive cables have gaps designed into their connectors to "open" unused drive select lines. The "shunts", however MUST NOT be removed, as these select, among other things, the drive #. One hint here. For convenience sake the shunt paks may be replaced by 8-position MINI-DIP switches. This will make re-configuring the system easy. TEAC drives, however, have a relocatable jumper plug which makes things even easier.

FLOPPY DISKS-GENERAL

A Mylar disk is contained within a protective sleeve. It is coated with mixtures of ferrite oxide (magnetic), lubricants, and binders. This disk spins freely inside the envelope.The envelope itself is multi-purpose. It: a) protects; b) supports (some degree of rigidity); and c) cleans the disk (as it spins, dust particles are forced into the lining). The envelope has 5 cut-outs. These are:

1) Alignment notches. Located on the bottom of the diskette The disk is put into the drive notches-first. This seats the disk in the drive such that it is properly aligned with the internal components of the drive.

PLATO package in this manner. So, it can't be all that bad!! I do use some common sense, though. For example, I do not flip the disks that contain essential data, or disks that I use often. for trouble.

Here's how to flip disks, should you decide to go ahea: with it. You will need a good hand-punch (available at stationery outlets). You also need a template. (SEE ISSUE NO.4 HV99 NEWS.ED.) I use a disk sleeve from a ruined disk. Just remove the disk from the sleeve, and you have a template which will last forever. The holes that you must punch out, are the WPN notch, and the SI holes (on both sides).

The WPN notch is no problem. Merely lay the disk down, place the template on the top, align them, and mark the WPN hole of the disk to be flipped. Use a soft-leaded pencil with light pressure. Then line the hand-punch up with the mark and punch-out the WPN.

SI holes are tricky. First, prepare a piece of paper, such that is will slide between the disk and the sleeve through the hub hole. This will protect the disk from possible harm. Slide the paper in through the hub hole and aligned opposite to the SI hole which is already present. Now lay the template over the disk, and gently trace out the SI hole onto the disk to be flipped. Next, slide the hand-punch between the paper and the sleeve such that, when you punch the hole, the residue falls outside the disk. This also allows you to precisely align the marking with the punch's hole. When you do the punching, you should make it in one, swift movement. Once you start pressure, do not stop, otherwise, you may make a sloppy job of it. Ok, so that's one side, now repeat the steps above and do the other side in the same manner. Once completed, initialize the disk, and you are in business. If ANY residue has fallen inside the envelope, it must be retrieved, otherwise, damage to the disk or the RW head could result.

That's about it for this small article. To fully explain everything would take a book, but this should get you started.

"DON'T BENCH ME IN"

It is an unfortunate fact of life that we as human beings begin to slow down as we become older. Fortunately this isn't true in the case of computers whose speed is controlled by it's internal clock.

Why then are we TI99/4A owners continually being told that our machine isn't any good because it is old fashioned and way too slow. Slow in relation to what I say. I dont mind sitting around a few extra seconds waiting for my machine to execute a program, besides who cares if the Macintosh completes a benchmark "X" seconds quicker than the TI, I usually spend the time reflecting on the number of dollars saved by purchasing a TI in preference to a Mac or Amiga or whatever.

I'm sure speed is of little concern to the dedicated TI owner for we all know we have it over the rest when it comes to accuracy and user friendliness right. Anyway accuracy is a subject for another article, for now lets try to figure out why it is so slooow.

It is a well known fact that the TI is a snack to program in Basic and Extended Basic and this is the very reason why the machine is slow, because the computer doesn't understand basic it only understands machine language, so an interpreter is needed. We all know how difficult it is to converse with someone who doesn't understand English, well the computer has the same problem.

However this problem is not unique to the TI99/4A so one way of comparing how fast various machines are, is by having them each perform several "benchmark" programs and timing the results

It should be noted that running benchmark programs is only a measure of how fast the machine is it doesn't measure how accurate, reliable, or easy to use the particular model can be.

PROGRAMS

```
100 REM BENCHMARK 1
110 PRINT "START"
120 FOR K=1 TO 1000
130 NEXT K
140 PRINT "END"
150 END
```

```
100 REM BENCHMARK 2
110 PRINT "START"
120 K=0
130 K=K+1
140 IF K<1000 THEN 130
150 PRINT "END"
160 END
170 RUN
```

```
100 REM BENCHMARK 3
110 PRINT "START"
120 K=0
130 K=K+1
140 A=K/K*K+K-K
150 IF K<1000 THEN 130
160 PRINT "END"
170 END
```

```
100 REM BENCHMARK 4
110 PRINT "START"
120 K=0
130 K=K+1
140 A=K/2*3+4-5
150 IF K<1000 THEN 130
160 PRINT "END"
170 END
```

```
100 REM BENCHMARK 5
110 PRINT "START"
120 K=0
130 K=K+1
140 A=K/2*3+4-5
150 GOSUB 190
160 IF K<1000 THEN 130
170 PRINT "END"
180 END
190 RETURN
```

```
100 REM BENCHMARK 6
110 PRINT "START"
120 K=0
130 DIM M(5)
140 K=K+1
150 A=K/2*3+4-5
160 GOSUB 220
170 FOR L=1 TO 5
180 NEXT L
190 IF K<1000 THEN 140
200 PRINT "END"
210 END
220 RETURN
```

```
100 REM BENCHMARK 7
110 PRINT "START"
120 K=0
130 DIM M(5)
140 K=K+1
150 A=K/2*3+4-5
160 GOSUB 230
170 FOR L=1 TO 5
180 M(L)=A
190 NEXT L
200 IF K<1000 THEN 140
210 PRINT "END"
220 END
230 RETURN
```

```
100 REM BENCHMARK 8
110 PRINT "START"
120 K=0
130 K=K+1
140 A=K^2
150 B=LOG(K)
160 C=SIN(K)
170 IF K<1000 THEN 130
180 PRINT "END"
190 END
```

MACHINE COMPARISONS

MACHINE	BM1	BM2	BM3	BM4	BM5	BM6	BM7	BM8	AV
Commodore Amiga	0.75	2.07	4.53	4.87	5.60	10.39	7.84	11.33	5.92
Apple Macintosh	.24	.65	6.00	6.40	7.10	8.60	15.90	52.30	12.15
TI Professional	1.00	4.20	9.30	9.70	10.50	19.00	29.50	31.00	14.28
Amstrad CPC-464	1.09	3.28	9.16	9.61	10.20	19.03	30.18	34.20	14.59
IBM PC	1.50	5.20	12.10	12.60	13.60	23.50	37.40	35.00	17.61
Sinclair QL	2.10	6.40	10.70	10.30	13.20	26.10	61.80	25.80	19.55
Apple II	1.30	8.50	16.00	17.80	19.10	28.60	44.80	107.00	30.39
MicroBee	2.70	10.00	18.10	17.90	20.90	39.40	67.30	95(est)	33.90
Tandy Color Comp	2.00	11.30	22.20	23.90	27.00	41.50	61.10	130.00	39.90
Commodore 128	2.00	11.80	22.00	23.30	26.50	42.40	67.30	126.00	40.10
ZX Spectrum	4.80	8.70	21.10	20.40	24.00	55.30	80.70	253.00	58.50
TI99/4A (XB)	4.50	11.50	29.50	28.50	31.50	59.00	90.00	210.00	58.60
Tandy Model 100	3.50	9.50	26.50	29.50	31.50	43.00	64.00	321.00	66.06
Atari 400/800	2.30	7.40	19.90	23.20	26.80	40.70	61.50	431.00	76.60

The TI99/4a results recorded in the above table were taken on my fully expanded rig using Extended Basic. The stopwatch was a very old analog model and times recorded were on the conservative side. No doubt if the times were taken using a digital stopwatch I'm sure we would have beaten the ZX Spectrum. Now there's something to brag about!!!.

The Benchmark programs and results for the other machines are courtesy of AUSTRALIAN PERSONAL COMPUTER MAGAZINE.

FUNNELWRITER DEMONSTRATION.

The first 'special interest group' demonstration on FunnelWriter was an outstanding success thanks to the efforts of Tim Watkins, the coordinator, and Brian Woods, the demonstrator. Brian's formatter demo was spot on, explaining the main points on a neatly formatted front sheet, with how it was actually done printed out in editor format on the next sheet. The demo was very timely for me as I had spent till midnight the night before trying to get the formatter to work, all was revealed in ten easy steps. (The worth of the demo will be proved by the faultless formatting of this article.) After a short break for coffee the enthralled masses regathered and Brian showed how to set up a standard letter file with defined prompts for the date and the persons name and how to bring it all together with the formatter.

Thanks must also go to Tony and Will McGovern for their work in producing FunnelWriter. The latest update allowing you to go from editor to formatter without having to go back to the title screen is also greatly appreciated.

The next get together will be on the Command Mode of the program

BEGINNERS BASIC.

The Beginners Basic Course has started again for the year 1986. The first class was used to discuss the direction, speed and subject matter of the course. The group now has a set syllabus to follow. The topics for the next three classes are summarised below.

1. Entering and controlling programmes, RUN, LIST, NEW, RES, NUM and VARIABLE NAMES.
2. Mathematical operation, parentheses, E notation, storing and retrieving programmes.
3. Entering numbers into a programme, printing out to screen and screen controls.