INTERNATIONAL TI-LINES

THE   WORKSHOP!

Formerly OXON TI USERS

A PUBLICATION
OF THE
INTERNATIONAL
TI USER GROUP

PETER G. Q. BROOKS
96 BANBURY ROAD
OXFORD OX2 6JT

OXFORD 510822

WORKSHOP AT BLOXWICH: SATURDAY, MARCH 28TH. PUT IT IN YOUR DIARY!

The photo on the title page is of the main entrance to the Community
School where the TI Users Workshop is to be held. If you have any query
which has not been answered by editorial in this issue, please contact
GORDON PITT on 0922 476373.

A number of articles have been held over until next issue, including the
Multiplan tutorial from BRENDA NOELL and the ADVENTURE HELP column from
JO ANN and SCOTT COPELAND, who are eager to receive your Adventure
questions. They can only guess at the difficulties which Adventuruous
ITUGers may experience, so please give them some guidelines for future
articles.

See you at Bloxwich on the 28th.!

```
-------------------------------------------------------------------------
       B A L D I E ' S       B U R B L E
-------------------------------------------------------------------------
```

Ouch! The blank pages on the last issue of IT are almost capable of
causing snow-blindness, aren't they ? Unfortunately, the photocopier
had been throwing a wobbly prior to my attempt to use it, and the
engineer wasn't due to sort things out until two days after my final
deadline for production, so I took a gamble - and lost. Despite being
put on the lightest possible setting, the blasted machine insisted on
chucking out huge gobs of toner, causing all my plans to crash to the
ground. All the photographs (which were not quite within the required
tonal range, as they say) had to be dropped, and I had not carried any
artwork fillers with me (the photocopier is 18 miles away!), so the
result was the large expanses of white which you have seen.

It's not even a case of better luck next time, as I found later when I
tried reproducing the photos using another system; in order to get a
decent reproduction of photos which haven't been specially developed
and printed, I need to have a "screen" made up, which can work out at
a fiver a picture. I cannot justify that cost at present, so until the
next Bloxwich Workshop (at the end of this month, don't forget) gentle
readers who have not managed to attend will have to wait to see what
sort of things we get up to.

That's always assuming that I manage to tear myself away for long enough
to take the pictures in the first place...

If anyone has any queries about the Multiplan tutorials which have been
appearing in IT courtesy of BRENDA NOELL, please send them to her care
of me, and I will forward them.

Any former TIHOMErs who remember MIKE O'REGAN might like to know that
although he's no longer an active 4A buff, he's apparently got the
User Group bug. He was recently mentioned in the ACC NEWS page of
February's PERSONAL COMPUTER WORLD, as he has started a Psion Organiser
User Group. PRACTICAL COMPUTING also featured a letter from him.

So you see, old TI Users never die, they simply pop up somewhere else...

STEPHEN SHAW, writing in the recent Winter issue of TI*MES (published
by TI-EXCHANGE), comments on an undocumented error message which he
obtained from Extended BASIC. It appears if you make a User-defined
DEF statement refer to itself, and then attempt to use it in a program.
Thus: 100 DEF T(N)=T(N-1) followed by 110 PRINT T(N) produces a
UDF REFS ITSELF error. Sounds positively painful...

Over the page you'll find details of the next BLOXWICH WORKSHOP.

))MORE)

```
*********************************************************************
*                                                                   *
*                                                                   *
*  T H E    N A T I O N A L    T  I   U S E R S    W O R K S H O P   I I  *
*                                                                   *
*                                                                   *
*          ORGANISED BY THE WEST MIDLANDS TI USER GROUP             *
*                                                                   *
*                  AND SUPPORTED BY ITUG                            *
*                                                                   *
*                                                                   *
*********************************************************************
```

The Workshop will take place on SATURDAY, MARCH 28th. at the SNEYD
COMMUNITY SCHOOL, VERNON WAY, (off Sneyd Lane), BLOXWICH, WALSALL.

It will start at 10.00 am, and finish at 6 pm. Exhibitors/Demonstrators
may enter and set up at 9.00 am by prior arrangement with Gordon Pitt.

Catering will only be available if Gordon receives sufficient demand IN
ADVANCE. Otherwise, only beverages will be available.

The following events are planned:

1. Presentation on Funnelweb/TI-Writer by JO ANN and SCOTT COPELAND
2. Presentation on TI Multiplan by PHIL HARTLAND
3. Demonstration by MECHATRONICS of an 80 column card, mouse (which can
   run with TI Artist 2), Super Extended BASIC, and THREE NEW MICROS
4. Hopefully a demonstration of the new MYARC 9640 PERSONAL COMPUTER,
   by RICHARD SIERAKOWSKI
5. Demonstration of the ITUG TIBBS No. 1 by GORDON PITT
6. Demonstration of the MYARC 128 Operating System by TREVOR DAVIES
7. Demonstration of the NEW HORIZONS RAMDISK by GORDON PITT
8. UCSD PASCAL Interest Group chaired by STAN DIXON
9. TI BASIC Programming Clinic with yours truly

Details on the new IBM PC clone from MG (Miller's Graphics) should be
available, and we hope that there will also be a hardware clinic. I am
told that Gordon's engineer will be on hand with his toolkit should
anything play up and need a bit of quick soldering!

Members of the CORTEX USER GROUP will also be attending (the Cortex is
a 9900-based micro, a project from Electronics Today International, and
its Power BASIC is very similar to TI's), and we hope that contingents
from the Bournemouth and the Bedford TI Users groups will be able to
make it.

Remember: the entry charge PER PERSON is £1.50 IF BOOKED BEFOREHAND, or
£3.00 at the door on the day. Cheques payable to GORDON PITT please.

As before, Gordon has tried to cater for the whole family should you
wish to cart them along, and Swimming, Squash, Badminton, Weights, and
the Saturday Club (6-16) are available IF ARRANGED IN ADVANCE WITH
GORDON.

Elsewhere in this issue there should be a simple sketchmap of the route
from the motorway. Further details concerning bus routes, etc., can be
found over the page.
>>>MORE>

The nearest station is WOLVERHAMPTON, and there is a bus service between
a road near the Community School and the bus station, which is near the
Rail Station. The bus number's changed since last year but is still the
"Bloxwich Bus". You ask the driver to tell you when the CRAB LANE stop
is reached (or the stop AFTER the one at "the square at New Invention").
Crab Lane is right at the bottom of the map, and it is a few minutes
walk from the Sneyd Community School (which is signposted).

Unfortunately, although the School is within spitting distance of the M6
and there is a junction nearby (10a), there is no connection (the link
is to the M54!) and junction 11 is the nearest. Take the WILLENHALL
(WALSALL) turning off the roundabout, and follow the road (WARSTONES
ROAD) looking for the fourth exit on the left which is SNEYD LANE. This
is a small crossroads, and don't be confused by the fact that Warstones
Road becomes BURSNIPS ROAD on the way — it'll all come out in the wash.

About 800 yards along Sneyd Lane is a signpost for the School on the
left, pointing to VERNON WAY.

Find a spot to park the car (there should be plenty of space), and walk
round to RECEPTION, where (depending on your arrival time) you will be
greeted by GORDON PITT (or someone), OR... you may have to ask at the
office, when you will hopefully be guided by a native runner...

If you come by bus from the station (should you be so intrepid), you
should head for RECEPTION as above. I am advised that the bus driver
may smile happily upon you if you tender the EXACT fare of 32p so he
doesn't have to search around for change!

I look forward to seeing as many of you as I can. Please do support
the Workshop — it's a chance for you to get some serious learning
(and maybe even teaching!), and you can only benefit from contact with
fellow Users.

Some recent feedback from eminent personages within ITUG has yielded an
opinion concerning the newsletter, its quality, and its content over the
last eight months. I have found it to be in agreement with my own,
which is, basically, that the newsletter has risen in quality but
dropped in content. This is largely due to the space constraint placed
upon its size by the finance available, but the odd thing is that the
content does reflect the wishes of ITUGers as recorded in the
questionnaire last year.

It is not a criticism of the material which has featured in IT over the
last eight months, but refers to the omission of types of articles which
IT used to carry (referring to Volume 2).

However, almost nine months have passed, and no doubt a number of people
have moved on from their position last year (greater experience, better
knowledge, more equipment, changed interests, etc.), so I will be
putting out another questionnaire in due course for ITUGers to once
again let me know exactly what they think. Last year brought a very
surprising response of over 80% — better than for national elections,
even. I'm hoping for a similar turn-out (or send-in ?) this year, and
if even those who are considering NOT renewing their subscriptions could
also send in their questionnaires (they should go out with V3.12) then

>>>MORE>

they could let me know why they decided not to renew and maybe help me get a better perspective on the group's performance as a whole.

The minor change in content will involve the inclusion of a number of specialised hardware articles which have been hanging around since last June (!) waiting for me to find a way of publishing them. Some authors, understandably, would not want their work relegated to a specialist supplement within a specialist magazine, but equally ITUGers had said that the upper hand should be given to descriptions of the application of software (for which I am grateful to JO ANN and SCOTT COPELAND, to BRENDA NOELL, and to COLIN HINSON for providing regular coverage of applications).

This issue therefore sees the publication of a project for owners of the TI STANDALONE DISK DRIVE CONTROLLER. A joint effort by COLIN HINSON and MARTIN ROSS, this valuable piece of work describes how to modify the controller so that it can handle DOUBLE-sided drives. There were quite a few standalones sold, I understand, so there should be a number of people able to benefit from this.

TWO EPROMS are necessary to complete the project, and they are available from me, price £15 excluding post, packing, and insurance. A set of full-size (A4 format) originals of the instructions are available also, although everything necessary has been included in this issue of IT - even two separate A4 sheets with diagrams. If you want the full-size set of instructions it will cost you an additional £1.

Another comment which has recently been made is that there is a large list of unfinished articles/projects which have been started or proposed over the last few years, but which have either never seen the light of day, or appeared for a short while and then petered out.

This is a fair comment, and has two answers. One, the most obvious, is that a negative response (or a total lack of response) from ITUGers is the main reason for projects not appearing. There is no point spending large amounts of time and money preparing articles/equipment if no-one is going to derive any benefit, or if no-one has any interest. The proposed project to produce video tutorials is a classic example: only two people evinced any interest, and the questionnaire last year was returned with a massive NOT INTERESTED response. The cost, in terms of time as well as money, does not justify preparing masters for only two interested parties!

I was surprised at the response, especially as the video tutorial is an ideal way of learning about something at your own pace, and being able to see exactly what to do would, I thought, be a massive advantage. I also thought that a large proportion of ITUGers had video recorders; maybe I was wrong. Either way, other things have come along to take up time, so unless we receive a sudden overwhelming postbag filled with entreaties to resurrect the video tutorials, they're dead.

The second answer, related to the first, is that many articles require such a large amount of preparation, that in the case of those which I penned myself, I found that I was unable to complete some of them while I still have the enormous task of producing the newsletter. I can only do one thing at a time, and IT usually comes top of the list. I can only promise to do better this year, and my success will depend on how much ITUGers themselves participate.
>>>END

WORD PROCESSING

By JO ANN COPELAND

The Grey-Haired Toothless Wonder

Recovered from our last article? I know that you believe you
understood what you think I wrote, but I am not sure you realize that
what you read is not what I meant. (It's gonna' be a good one folks!)

How many of us have had enough of Special Character Mode, Control U,
Commands? STAND UP and CHEER because we're taking a break from it!
Hurray, applause, cheers! (Although I'm bound to pop up with something
on it every now and again!)

However, we're going to back-track and get back into TI-Writer a/k/a
Funlwriter fun stuff. (Boo hiss!) Yes folks, we're going to discuss
the dreaded "Include File" command. Oh no.....

Include File (.IF) can be helpful in many cases. Ever typed out a
document that took up several files? In my case, I have (what do you
mean "who cares?") and have found this command useful more than once.

Let's say you typed out a manual - for whatever reason - and had
several files of it, perhaps named DSK1.MANUAL1, DSK1.MANUAL2,
DSK1.MANUAL3, and last but not least DSK1.MANUAL4. Well, you can go to
the Formatter to type these out and individually call up each of these
by name, or make up a very small file named, let's say DSK1.MY/MANUAL.
In this MY/MANUAL file, you could put the following:


```
0001     .FI
0002     .AD
0003     .LM 10
0004     .RM 65
0005     .PL 55
0006     .HE Manual: History of TI-Writer
0007     .FO Page %
0008     .IF DSK1.MANUAL1
0009     .IF DSK1.MANUAL2
0010     .IF DSK1.MANUAL3
0011     .IF DSK1.MANUAL4
```

        (Save this under DSK1.MY/MANUAL)

Okay, when you go to your Formatter to type out your documents you no
longer need to individually call up DSK1.MANUAL1, DSK1.MANUAL2,
DSK1.MANUAL3, etc. Wonder of wonders - now you can call up only one
file, DSK1.MY/MANUAL, and this takes care of the rest of the files.

)))MORE)

7

What happens in the above case is: DSK1.MY/MANUAL is instructing the
Formatter to type the files Filled In and Adjusted (remember them?),
Left Margin of 10 and Right Margin of 65, Page Length will be 55 lines,
a Header starts pages with Manual: History of TI-Writer, and a Footer
prints out the page numbers. The pages will follow through from 1 to 2
to 3 and on and on, following suit with each new file. So if
DSK1.MANUAL1 has pages 1 - 20, DSK1.MANUAL2 starts off on page 21 and
up. So, the Formatter reads DSK1.MY/MANUAL and sees that it needs to
set up margins, headers, footers, etc., and then prints each individual
document file (MANUAL1, MANUAL2, MANUAL3, MANUAL4) in consecutive order
as listed. Each file has consecutive page numbers, with equal margins,
etc. Now you can call up the one file (DSK1.MY/MANUAL) (a Mass Print
File) and go about your business until you hear your paper jam in your
printer!!

Just a couple of things to remember on this. If you have DSK1.MY/MANUAL
in drive one, but the other files in drive two - the Formatter can't
call them up because it is looking for - DSK1 - so you have to watch
what drive number you are listing. Remember in your Mass Print File
what disk drive your files are in and list them appropriately. For
example:

        Mass Print File (DSK1.MY/MANUAL)

        0001 . IF DSK2.MANUAL1
        0002 . IF DSK2.MANUAL2
        0003 . IF DSK1.MANUAL3
        0004 . IF DSK1.MANUAL4

In this case, you go to the Formatter and call up DSK1.MY/MANUAL - the
Formatter will search, in order, disk drive two (2) for MANUAL1, then
MANUAL2, then disk drive one (1), for MANUAL3 and 4.   Okay?

Also in this case, you have shown no left or right margins, etc., and
the documents will be printed out individually as shown in each actual
file.

One more thing - if you have a particular diskette name - say TEXT -
the Formatter can look for the diskette by name, and not location. If
you have your Mass Print File set up like this:

        0001 . IF DSK.TEXT.MANUAL1
        0002 . IF DSK.TEXT.MANUAL2
        0003 . IF DSK.TEXT.MANUAL3
        0004 . IF DSK.TEXT.MANUAL4

Well, I hope that's okay - now on to more fun stuff! Ugh - Using the
Form Letter Option and/or/with Mail Merge Option.   Don't you just hate
it? (It's not as difficult as the manual makes it sound!)

We're going to cover just a little bit of this at first, and then go
into more detail later. Right now, have your TI-Writer or Funlwriter
loaded and put a blank, initialised disk in after you've selected the
Editor. We're going to make up a sample letter. Select (T) Tabs and

>>MORE)

8

have left margin of zero and right margin at 60 and press Enter.
Following the example listed go to the beginning of the line (left
margin) and type the following on each line number:

```
0001 Your first name and last name ENTER
0002 Your street address  ENTER
0003 Your city, state, post code ENTER
0004 (enter for blank line)
0005 Type in current date ENTER
0006 (enter for blank line)
0007 (enter for blank line)
0008 Mr. Peter Brooks ENTER
0009 96 Banbury Road  ENTER
0010 Oxford, England OX2 6JT ENTER
0011 (enter for blank line)
0012 (enter for blank line)
0013 Dear Peter, ENTER
0014 (enter for blank line)
0015 (Now, type a short letter to Peter about how much you like
.... receiving ITUG every month - or thereabouts - and how much
.... you enjoy reading the TI-Writer/Funlwriter reviews!)
....
0030 Sincerely yours, ENTER
0031 .SP 4           ENTER (will go down 4 lines)
0032 Type your name   ENTER
```

Save this to your disk under the filename of LETTER (DSK1.LETTER or
DSK2.LETTER depending on your number of drives). Enter Command Mode
and print this out with line numbers - example: FCTN 9 for command
mode, PF for print file mode, enter, type L - one space - and the
printer devicename. ENTER. This prints out your letter along with the
line numbers. You should have a listing of each line number and what
is on that line!

Now, we're going to use this again so keep it in your appropriate disk
to create "variables" with form letters - huh?

Ever wonder how companies send out thousands of computerized letters,
but your name and address are typed in separately distinguishing you
from those other thousands? You can do this too with TI-Writer a/k/a
Funlwriter by setting up variables. You can change certain things
within one standard letter (names, addresses, salutations).

These items are "variables" - simply stated, things which vary. So, we
are going to use the DSK?.LETTER we just made up to put in "Alternate
Input" commands. Confused yet? Okay, let's go step-by-step.

Pull up your file DSK?.LETTER and get it on screen. Pretty huh? Locate
line 8 and place your cursor at the beginning of the left margin.

Press FCTN 1 (delete) and take out all BUT the carriage return symbol.
Type in the following:

```
    0008 *1* *2* *3* ENTER  (one space between *1*, *2* and *3*)
```

>>>MORE)

Go to line 9 and delete all BUT the carriage return symbol. Type in:

    0009 *4* ENTER

Go to line 10 and delete all BUT the carriage return symbol. Type in:

    0010 *5* ENTER

Go to line 13 and place your cursor over the "P" in Peter and delete
all BUT the carriage return symbol. Type in:

    0013 *6* .*2* and type a comma ENTER

What we are doing (besides going crazy) is assigning values. Where the
1 *2* *3* are it means we are placing a variable for Title (Mr.,
Mrs., Miss, or Ms.), *2* locates a variable for the recipient's first
name, and the *3* for the recipient's last name. *4* equals the
variable for the recipient's street address. *5* the city, state, post
code. *6* in line 13 locates the variable for any title (Aunt, Uncle,
Sister, Brother, Cousin) and *2* in line 13 relocates the recipient's
first name in its second use. Got a headache yet?

Well, if we survived, all of our variables should be located and
assigned. Go to Command Mode (FCTN 9) and save this file (SF) under
the filename of FORMLETTER (DSK1.FORMLETTER or DSK2.FORMLETTER
depending). If you'd like to print this, go to command mode, PF,
enter, and type in an L, one space, and your printer devicename, enter.
This printed copy should help you follow the next exercise. (You mean
there's another one!!??!!)

Now we will discuss the method of construction of a "value file". The
Formatter can read appropriate information two ways. (1) Typing in
your values as you print each letter, and (2) Creating a separate value
file - thus the value file method. Let's look at it.

A value file will give, in list form, a value for every variable - one
list per letter. These lists are going to be called Data Sets, which
are created for each letter you want to print out from your FORMLETTER.
A value file can be used with a maximum length for a value of 77
columns. Definitely got a headache now, huh?

The form letter we made up had 6 variables (*1*, *2*, *3*, *4*, *5* and
6). We are going to construct a value file with separate Data Sets
so the Formatter can use these to print our letter to separate people.
Oh Really? I can't wait...

Let's start (what do you mean "let's just pretend we did?"). Go to your
editor and set up a new file. Set your tabs - no real tab sets
required. Locate line 1 at the left margin at the beginning of the
line. Type the number 1, one space, and then type Mr. and press ENTER.
Now, this means we have assigned the value "Mr." to variable "1" in our
first letter. On line 2, type 2 and one space. Put in "Peter" and

>>MORE)

press ENTER. Now, "Peter" is the value for variable "2" in the first
letter. Follow on like this:


```
     0001 1 Mr.                        ENTER
     0002 2 Peter                      ENTER
     0003 3 Brooks                     ENTER
     0004 4 96 Banbury Road            ENTER
     0005 5 Oxford England OX2 6JT     ENTER
     0006 6 Babbling Brooks            ENTER
     0007 *
```

Okay, this is one Data Set and we are done, so on line 7 type in an
asterisk and press ENTER. The asterisk denotes "end of data set". To
set up a second Data Set go to line 8 and assign values for your second
letter.


```
     0008 1 Mrs.                       ENTER
     0009 2 JoAnn                      ENTER
     0010 3 Copeland                   ENTER
     0011 4 P. O. Box 5927             ENTER
     0012 5 Lakenheath IP27 9PN        ENTER
     0013 6 Gray-haired          Wonder ENTER
     0014 *
```

Line 14 shows by the asterisk (*) that Data Set number 2 is done and,
if we wanted to, we could assign a third Data Set by the following:


```
     0015 1 (assign no value but press enter after one space)
     0016 2 Ulysses and Margaret
     0017 3 Smith
     0018 4 123 TI-Road
     0019 5 TI Texas 01010
     0020 6 (assign no value but press enter after one space)
     0021 *
```

Third Data Set shows finished by the asterisk (*). One more and let's
finish:


```
     0022 1 Mr.
     0023 2 James
     0024 3 McArthur
     0025 4 Great TI Tabernacle Church
     0026 5 South America on the Bend
     0027 6 Reverend
     0028 *
```

Let's see what we can do with all this. Save this file under the name
of VALUEFILE (DSK1. or DSK2.). Enter Command Mode and select Quit and
Exit - (more applause?).

)))MORE)

Let's print with the Mail Merge Option of the Text Formatter. Usually,
you go to your title screen and press (2) for Formatter.

In this case, we will use the Mailing List. So, select Text Formatter
and type in DSK?.FORMLETTER. (Enter).   (? denotes your drive number.)

Check your printer devicename and, if correct, enter. When you see the
prompt "USE MAILING LIST?" type Y (yes).   (Enter). Press enter for
WHICH LETTER(S) and get your paper ready. Press Enter to bypass
pausing at end of page.

When you see "MAILING LIST NAME:" type in DSK?.VALUEFILE and press
enter. Your printing should begin. Let's see what happens...

You should have printed four letters. Each should have kept your
name, address, city/state/post code and current date as we originally
typed it in our letter. Each should have the same text, with the same
closing. However, the names and addresses and salutations should have
changed to those we entered in our VALUEFILE. So we typed this letter
once, but can send it to four people. Neat huh? (What do you mean it
didn't work? It did on my computer!)

Those letters should be addressed as follows (and show the following
salutations):

Letter 1:   Mr. Peter Brooks
            96 Banbury Road
            Oxford England OX2 6JT

            Dear Babbling Brooks Peter,

Letter 2:   Mrs. JoAnn Copeland
            P. O. Box 5927
            Lakenheath IP27 9PN

            Dear Grey-Haired   Wonder JoAnn,

Letter 3:   Ulysses and Margaret Smith
            123 TI-Road
            TI Texas 01010

            Dear Ulysses and Margaret,

Letter 4:   Mr. James McArthur
            Great TI Tabernacle Church
            South America on the Bend

            Dear Reverend James,

If perhaps you only wanted certain letters in your VALUEFILE to print
out, you could go to the Formatter, and when it prompts you to answer
"WHICH LETTERS?" you could type in 1,4 instead of ENTERing for ALL.
This time you would print out letters 1 and 4.

>>>MORE)

12

This may sound confusing, but what it comes down to is only making 2 files: first, DSK?.FORMLETTER and second, DSK?.VALUEFILE.

DSK?.FORMLETTER is your letter (any letter) that you wish to type only once to send to several people. You would put your name, address, city, post code and date in. Instead of actually addressing your letter, put in your variables where you would normally type the recipient's name, address, etc. Type your text, your ending, and save your file.

Then set up a new file for DSK?.VALUEFILE. DSK?.VALUEFILE is where you place your data sets, separating each with an asterisk. Type in as many data sets as you need letters and save this file. Go to your Formatter and print them out!

Easier said than done? Now, go ahead and take some aspirin for that headache!

PHEW! Had enough for a while? Have a workout with this and experiment with what we've covered in this column. We'll be covering a little more next article, including "Define Prompt Command" (a different way of doing what we did today, without typing in names and addresses, etc.), and "Address Lists" (for printing mailing labels).

Until then, have fun and I hope your headache goes away!

'Bye for now.....

>>END

```
---------------------------------
    PROGRAMS THAT WRITE PROGRAMS
---------------------------------
```

By JIM PETERSON

Part 1

   Way back in 1982, in the old
99'er Magazine, Vol. 1 Nos. 3
and 4, John Clulow wrote two
articles entitled "How To Write a
Basic Program That Writes Basic
Programs". At that time I
thought I would never understand
what he was writing about!
   But really, it's simple. Have
you ever LISTed a program to the
disk, and noticed that the
resulting D/V80 file took up many
more sectors than the program
itself? That is because the TI
saves programs in a compacted
form, with each statement
represented by a single token
ASCII.
   When a program is saved in
MERGE format, by SAVE DSK (file-
name),MERGE it is saved in this
same compacted form, but in a D/V
163 file. And of course a D/V
file can be created by a program
- so you can write a program
which will create a D/V 163 file
in the form of a program, and
then MERGE that file into memory
and RUN it as a program, and SAVE
it as a program.
   You ask, why use this
roundabout way of writing a
program? Why not just key it in
directly? Well, for one thing you
can write program lines that
could not possibly be keyed in
directly. As for instance, the
famous "line zero". Key this in,
run it with a disk in drive 1,
then enter MERGE DSK1.ZERO and
LIST the result.

```
100 M$="BETCHA CAN'T DELETE
THIS!"
110 OPEN #1:"DSK1.ZERO",VARI
ABLE 163,OUTPUT :: PRINT #1:
CHR$(O)&CHR$(O)&CHR$(131)&CH
R$(200)&CHR$(LEN(M$))&M$&CHR
$(O)
```

```
120 PRINT #1:CHR$(255)&CHR$(
255):: CLOSE #1 :: END
```

   Actually, there is an easy way
to delete that line - but no way
to key it in directly.
   Here's another one - the full
ASCII string.

```
100 OPEN #1:"DSK1.FULLSTRING
",VARIABLE 163,OUTPUT
110 LN=100 :: GOSUB 190 :: A
$=L$&"M$"&CHR$(190)
120 FOR J=1 TO 127 :: C$=C$&
CHR$(J):: NEXT J :: A$=A$&CH
R$(199)&CHR$(127)&C$&CHR$(0)
130 PRINT #1:A$
140 GOSUB 190 :: B$=L$&"M2$"
&CHR$(190)
150 FOR J=128 TO 255 :: D$=D
$&CHR$(J):: NEXT J :: B$=B$&
CHR$(199)&CHR$(128)&D$&CHR$(
0)
160 PRINT #1:B$
170 GOSUB 190 :: F$=L$&"M$"&
CHR$(190)&"M$"&CHR$(184)&"M2
$"&CHR$(0)
180 PRINT #1:F$ :: PRINT #1:
CHR$(255)&CHR$(255):: CLOSE
#1 :: END
190 L$=CHR$(INT(LN/256))&CHR
$(LN-256*INT(LN/256)):: LN=L
N+10 :: RETURN
```

   Run that, then enter NEW, then
MERGE DSK1.FULLSTRING. The
string contains every ASCII from
0 to 255 in sequence, and there
is no way to enter many of the
unprintable ASCII codes from the
keyboard. You can of course
create that string in a program -
FOR J=0 TO 255 :: M$=M$&CHR$
(J):: NEXT J
but it saves a few seconds to
have it predefined.
   The full ASCII string is very
useful for a quick shuffle
without duplication. For

`>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>MORE>`

instance, to scramble the numbers
200-250, -

```
100 M$="
              !""#$%&'()*+,-./
0123456789:;<=>?@ABCDEFGHIJK
LMNOPQRSTUVWXYZ[\]^_'abcdefg
hijklmnopqrstuvwxyz{|}~ "
110 M2$="
                         "
120 M$=M$&M2$
130 M$=SEG$(M$,200,50)
140 L=LEN(M$):: RANDOMIZE ::
 X=INT(L*RND+1):: N=ASC(SEG$
(M$,X,1)):: M$=SEG$(M$,1,X-1
)&SEG$(M$,X+1,255)
150 PRINT N;:: IF LEN(M$)=0
THEN STOP ELSE 140
```

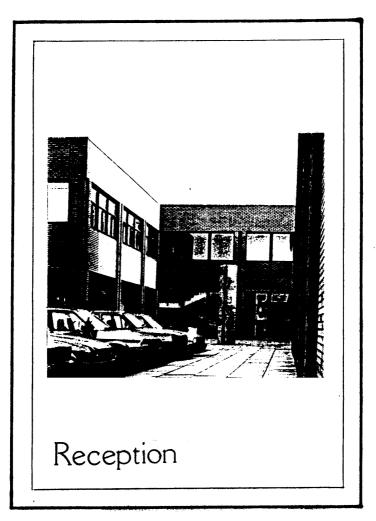   One more example - can you run
this program and get these
results? You won't even be able
to key in that last line!

```
)LIST
100 FOR J=1 TO 7 :: READ M$
:: PRINT M$ :: NEXT J
30000 DATA AAAAAAAAAAAAAAAAAA
AAAAAAAAAAA, BBBBBBBBBBBBB, BB
BBBBBBBBBBBB, CCCCCCCCCCCCCC,
                DDDDDDDDDDDDDDDD
30010 DATA "TESTING",,,,,,,,,
,,,,,,,,,,,,,,,,,,,,,""TEST
ING""
) RUN
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
BBBBBBBBBBBBB, BBBBBBBBBBBBBBB
CCCCCCCCCCCCCC
                DDDDDDDDDDDDDDDD
"TESTING"
,,,,,,,,,,,,,,,,,,,,,,,,,,
""TESTING""

*READY*
```

   Next month - the answer to that
puzzle, and a more useful program
that writes a program, and then
we will start learning how you
too can write programs that write
programs!

{Note that the space following the tilde (~) above is in reality the
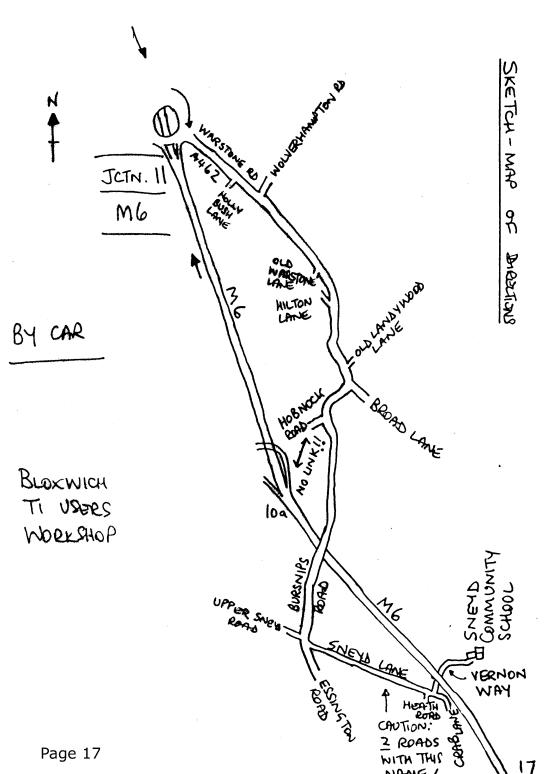character DEL - ASCII code 127, and obtained by FCTN V. PB}

>>>>>>>)>>END

Reception

THE
BLOXWICH
WORKSHOP

SATURDAY, MARCH 28TH., 1987

10.00 AM TO 6.00 PM

ENTRANCE FEE PER PERSON : £1.50 IF BOOKED AND PAID

£3.00 IF PAID AT THE DOOR

SKETCH - MAP OF DIRECTIONS

N

JCTN. 11
M6

BY CAR

BLOXWICH
TI USERS
WORKSHOP

WARSTONE RD
A462
WOLVERHAMPTON RD
HOLLY BUSH LANE
OLD WARSTONE LANE
HILTON LANE
M6
OLD LANDYWOOD LANE
HOBNOCK ROAD
BROAD LANE
NO LINK!!
10a
BURSNIPS ROAD
UPPER SNEYD ROAD
M6
SNEYD COMMUNITY SCHOOL
SNEYD LANE
VERNON WAY
ESSINGTON ROAD
HEATH ROAD
CAUTION: 2 ROADS WITH THIS
CRAB LANE

DOUBLE SIDED STANDALONE DISK CONTROLLER CONVERSION

OBJECTIVE

The aim of this project is to convert the PHP 1800 standalone disk
controller which is currently dedicated to single sided drives in 40
track single density format to work with either single or DOUBLE SIDED
disk drives, again in 40 track single density format.

ORIGINAL RESEARCH AND INSTRUCTIONS

Colin Hinson

DIAGRAMS, LAYOUT AND AMENDMENT TO INSTRUCTIONS

Martin H.R.A. Ross

PROJECT PARTS LIST

1 off 74LS30 Integrated Circuit
1 off 7438   Integrated Circuit
2 off 24 Pin Integrated Circuit sockets
Assorted lengths of coloured wire
Solder
2 off TMS 2532 Eproms containing the required software. (These
are available from the International TI User Group. Contact
Peter Brooks at 96 Banbury Road, OXFORD, England, OX2 6JT)

PROJECT TOOLS LIST

Fine tipped soldering iron (see NOTE)
Solder sucker
Wire stripper
Fine-nosed electrical pliers
Cross-headed screwdriver
Small plain-headed screwdriver
A sharp implement capable of making a cut in a circuit board track

NOTE: A fine tipped bit of approximately 1/8th inch is ideal for
the general work of heating socket pins for solder removal etc.
It is also perfectly suitable for experienced hobbyists to
undertake the soldering in this project, indeed, Colin used an
1/8th inch bit to do all his soldering.

I must confess that I used an interchangeable smaller bit
of approximately 1mm for all the fine detail work, only using
the larger bit for desoldering.

Martin H.R.A. Ross

>>MORE)

CONVERSION OF STAND-ALONE DISK CONTROLLER TO USE DOUBLE SIDED DRIVES

Original research and instructions by Colin Hinson

Diagrams and corrections to instructions by Martin Ross
    (having found the errors the hard way)

TECHNICAL INTRODUCTION

    There are three reasons why the Stand-alone disk controller will
not control Double sided drives as it stands;

    1/.   The 'side select' signal to the drive is connected to +5v
           through a resistor, and so stays at '1' all the time

    2/.   The software in the DSR (Device Service Routine) ROMS do not
           contain the software to deal correctly with double sided disks

    3/.   The FD1771 floppy disk controller chip is not fully decoded to
           allow sufficient space for the additional software required

    To overcome these problems it is necessary therefore to add the
logic required to drive the side select line and also the logic to
fully decode the FD1771, allowing sufficient space to add the double
sided software. In the following instructions, the 74LS30 is the
decoder and the 7438 is the side select line driver - the EPROMS
(TMS2532s) contain the required software.

INSTRUCTIONS

    Remove the top of the unit (6 cross-headed screws at the bottom)
and take out the loose piece of plastic at the rear which surrounds the
disk drive ribbon cable edge connector. If the sliding cover on the
right hand I/O edge connector falls out, locate it in the top of the
unit and push it right up. (It should stay there.)

    Remove the top of the metal box which covers the disk controller
electronic boards (3 screws), - one tab may be soldered.

    NOTE: The dismantling instructions assume that the unit is sitting
with the mains lead connector and disk drive edge connector towards you
with the transformer and on/off switch furthest away from you.

    Unclip the top board (the retaining tabs push into the pillars) and
carefully fold the top board over to the right, taking care not to
damage the 'ribbon' connector. Unplug the power supply leads from both
the centre and transformer boards then remove the four screws retaining
the centre and bottom boards. Clip the top board back onto the centre
board.

    Unfold the centre AND top boards up and away from you, remove the
four round polythene stand offs from the posts which go through the
bottom boards and allow the top and centre boards to return to their
normal position.

ITUG 01/02

>>MORE>

Lift the whole assembly up including the bottom metal case so as
to clear the posts which go through the bottom board and remove from
the black baseplate. You now need to remove the electronics from the
bottom metal case — this is easy to say but may not be so easy to do
depending upon your unit. Basically it is a case of gripping the black
right hand edge connector (at the sides — not the top and bottom as the
metal case covers the bottom), and sliding the assembly to the right so
as to slide the left hand edge connector out of the hole in the metal
case. On some units there appears to be a problem in that the small
screws in the metal case at the left hand side foul on the centre
board. When you have accomplished this feat, the electronic boards
should be free from the unit. Place the bottom metal case back in
position on the black baseplate and reconnect the connector on to the
transformer board.

OPERATIONS ON BOTTOM BOARD

    Place the boards on a good flat surface with what was the top
board face down (the top board has the four largest holes in it) with
the plastic edge connector socket facing you. The ribbon cables
connecting the boards should now be on your left hand side.

    In order to get the extra decoding required, it is necessary to
'piggy back' a 74LS30 on top of the 74LS244 labelled U2. Not all the
pins need to be soldered to the 74LS244 which will be made clear in the
following text.

    Firstly, all the pins of the 74LS30 should be tinned unless they
have been previously solder dipped. Pins 3,5,7,9,10,17 and 20 of the
74LS244 should also be tinned. Care should be taken not to short the
pins to the tracks which run between them or to adjacent pins. If you
are unfortunate enough to short them, ask someone to hold the unit
upside down. Then, after applying a little fluxed solder, run the
shorting solder onto the soldering iron bit.

    The next stage is to deal with the pins as detailed below.
Reference should also be made to the Bottom Board detail diagrams which
should prove helpful as a guide.

    NOTE: IT IS ASSUMED IN ALL THE FOLLOWING INSTRUCTIONS THAT THE
NOTCH IS AT THE TOP OF ANY INTEGRATED CIRCUIT, THEREFORE PIN 1 IS
ALWAYS THE PIN AT TOP LEFT

   Pin 1   -   Bend out at right angles

   Pin 3   -   Bend towards Pin 4 to overlap

   Pin 5   -   Bend out at right angles

   Pin 7   -   Bend out at right angles

   Pin 8   -   Bend out at right angles and shorten the leg as shown in
                 the diagram

   Pin 11   -   Bend towards Pin 12 to overlap

   Pin 14   -   Bend out at right angles and shorten the leg as shown in
                 the diagram

                                                          ITUG 01/03

>>MORE>

After checking with the illustration, the 74LS30 should be placed
on top of the 74LS244 (U2) so that pin 2 of the 74LS30 is directly
above Pin 3 of the 74LS244 (U2). The notches on both I.C.s should be at
the top. Now solder the following pins together:

    Pin 2   (74LS30) to Pin 3 U2 (74LS244)

    Pin 3 + 4   (74LS30) to Pin 5 U2 (74LS244)

    Pin 6   (74LS30) to Pin 7 U2 (74LS244)

    Pin 11 + 12 (74LS30) to Pin 17 U2 (74LS244)

    The following pins of the 74LS30 which were bent at right angles
should be connected by soldered wire as follows

    Pin 1   (74LS30) to Pin 5 U7 (74LS04): IMPORTANT: See NOTE

    Pin 5   (74LS30) to Pin 9 U2 (74LS244)

    Pin 7   (74LS30) to Pin 10 U2 (74LS244)

    Pin 8   (74LS30) to Pin 4 U9 (74LS138): IMPORTANT: See NOTE

    Pin 14   (74LS30) to Pin 20 U2 (74LS244)

    NOTE: If you are NOT confident that you can make the connections
to both U7(74LS04) and U9(74LS138) without melting the ribbon cable,
the wires can be taken over the edge of the board and connected
underneath. If you HAVE damaged the ribbon cable, use soldered wire
to restore the connection or connections. Please note that any pins on
the 74LS30 NOT detailed above need not be soldered.
    The legs on Pin 8 and 14 have both been shortened to avoid the
possibility of any shorting to the metal case when reassembled.

    Now fold the board over to the left taking care not to damage the
ribbon cable and put some form of support under it. Carefully cut the
printed circuit track between Pin 4 of U9(74LS138) and Pin 6 of U7
(74LS04) as shown in the Bottom Board diagram. Take care not to damage
an adjacent track as it takes a bit of effort to cut the track due to
the high quality of the board.

    This completes the operations on the bottom board.

OPERATIONS ON MIDDLE BOARD

    The DSR ROMs U16 and U17 which are both 24 pin devices need to be
removed and replaced by 24 pin sockets. These will then be used to
mount the EPROMs which are in our shopping list.

    The following guidance notes should prove useful for any occasion
where a device needs to be removed from a circuit board without damage
to either the device or the circuit board.
                                                            ITUG 01/04

))MORE)

REMOVAL OF SOLDERED-IN DEVICES

The main requirement to undertake this task is a soldering iron
with a bit large enough to get the heat in fast and a hand held solder
sucker. (A soldering bit 1/8th inch wide should be adequate).

The aim is that each pin/hole should have all the solder removed
and also that the pin should be central in the hole or at least not
touching the sides of the hole.
Ensure that the board is firmly supported and then deal with one
pin at a time. Determine which way the pin will have to be moved to put
it in the centre of the hole and then apply the iron to the side which
will need pushing to centralise the pin.

Wait for the solder to melt then allow a further two seconds for
the solder to melt right through to the other side at the same time
centralising the pin. Remove the iron and IMMEDIATELY apply the nozzle
of the solder sucker vertically over the hole and press the button.
(You DID remember to 'cock' the mechanism first, didn't you?)
If the solder shows some reluctance to melt, then apply some MORE
fluxed solder as this will help the melting process. Repeat the
operation for all the pins then turn the board over to examine the pins
from the top. Ensure that you can see through all the holes from which
you think you have removed all of the solder. If you can't see through
all of them, then apply MORE fluxed solder and RE-SOLDER the offending
pin and try again. (It is virtually impossible to melt the solder at
one side of a hole when there is insufficient solder on the other side
to carry the heat through. Re-soldering solves this problem.)

When you can see through ALL the holes, take a pair of fine nosed
pliers and grip each pin from the bottom of the board and move it from
side to side (towards and away from the centre line of the I.C.). If it
moves freely, it is O.K.
If it moves, but stiffly, wobble it a few times and it will
probably go 'click' as the solder holding it to the side of the hole
breaks.
If it will NOT free up, then examine the hole/pin to determine why
it will not move. (You will probably find that all the solder has not
been removed, so, following the instructions above, resolder and try
again.)

Theoretically, when all the pins are 'free', the device should
literally fall out when the board is inverted and this happens about
75% of the time. If it doesn't fall out, then put a screwdriver blade
under one end of the device and twist the screwdriver VERY gently. It
is likely that there will be a 'click' and the device will fall out.

If you have to apply undue pressure, there is probably one pin
with some solder retaining it. If necessary, resolder as advised above.

WHEN THE DEVICE COMES OUT ENSURE THAT NO TRACKS ARE LIFTING AND
DETACHING FROM THE BOARD.

The two EPROM sockets should now be soldered into the holes and
the EPROMS inserted into the appropriate sockets with Pin 1 (the
'notch' end) facing the disk drive cable edge connector.

ITUG 01/05

>>>MORE)

OPERATIONS ON MIDDLE BOARD (CONTINUED)

    With the populated side of the middle board uppermost and the disk
drive cable edge connector at the bottom, check if there is a capacitor
'bodged' onto track 32 of the disk drive cable edge connector. (This is
the second track in from the right hand side. It is easily identifiable
as it is adjacent to track 34 which is marked on the circuit board.)

    Remove the capacitor by unsoldering and discard it as it is not
required again for this project. Ensure that the hole in the circuit
board at the top of track 32 is clear.

    It is now necessary to 'piggyback' the 7438 on top of the 7438
which is identified as U20 on the middle board. Although U20 is not
actually marked on the circuit board it is easily identifiable as it is
immediately above C28 which, in turn, is immediately above the left
hand edge of the disk drive cable edge connector.

    Firstly all the pins of the new 7438 should be tinned unless they
have been previously solder dipped. Pins 4,5,6,7,8,9,10,11,12,13 and 14
of the 7438 on the circuit board should also be tinned.
    Care should be taken not to short the pins to the tracks which run
between them or to adjacent pins. As advised earlier, if you are
unfortunate enough to short them, ask someone to hold the unit upside
down, then, after applying a little fluxed solder, run the shorting
solder onto the soldering iron bit.

    The next stage is to deal with the pins as detailed below.
Reference should also be made to the Middle Board detail diagrams which
should prove helpful as a guide.

REMINDER: It is assumed that the notch is at the top of any Integrated
Circuit, therefore Pin 1 is always the top left pin.

    Pin 1   -   Bend out at right angles

    Pin 2   -   Bend out at right angles

    Pin 3   -   Bend out at right angles

    After checking with the diagram, the 7438 should be placed on top
of the 7438 (U20) so that each pin connects to its exact counterpart.
e.g. Pin 4 to Pin 4, Pin 5 to Pin 5 etc.

    NOTE: In this particular case, it is easier to work on the middle
board with the disk drive cable edge connector at the bottom of the
board. The notch on the onboard 7438 will therefore be at the bottom.
    Ensure that the notch on the 'piggyback' 7438 is at the same end
as on the onboard 7438 and solder the pins together.

    The following pins of the 7438 which were bent at right angles
should be connected by soldered wire as follows, noting that pins 1 and
2 are joined together.

                                                        ITUG 01/06

>>MORE>

Pin 1 + 2   (7438) to Pin 12 U15 (74LS259)

Pin 3   (7438) to the disk drive cable edge connector track 32
             through the hole from which the discarded capacitor was
             removed

    That's it folks! - If you got here successfully, it should work
when you put it back together. Before you reassemble it all, check and
recheck the connections you have made. Check for shorts between pins
and tracks where you have been soldering including the two sockets.
    Look under the Eproms, between them and their sockets and make
sure that no pins have been bent under the Eproms instead of going into
the socket. Check that the 'notches' on both the '74' I.C.s and Eproms
are the correct way round by referring to the relevant portion of the
instructions and diagrams.
    Make sure that the red, black and green leads which are connected
to both power plugs have not become detached on either side of the
lead-throughs in the lower metal casing. Check that the insulating clip
is correctly positioned over the ends of the lead-throughs inside the
lower metal casing.
    Hold the black baseplate assembly (with the power supply on it)
upside down and tap gently to ensure that there is nothing in there
that shouldn't be!
    Also, make sure that there are no bits of loose wire or solder on
the electronic boards.

    NOTE: If any of the power connection leads have become detached,
the wire colours are the same at both ends of the stand-offs. The white
wires are soldered to the metal case.

RE-ASSEMBLY

    Lay the black baseplate in front of you with the mains lead
connector towards you and the transformer and on/off switch furthest
away from you.

    Place the lower metal casing in position over the pillars with the
single tab with through hole at the bottom and the copper coloured clip
on the left hand side of the casing. Reconnect the power plug from the
lower metal casing to the transformer board.
    Lift the left hand side of the casing up so that the base of the
metal casing is level with the top of the two left plastic pillars. At
the same time, take the three part circuit board assembly with the
black connector to the right hand side. Ease the left hand side of the
board through the hole with the copper coloured clip, at the same time
locating the four plastic pillars through the four holes in the lower
circuit board. Check that the lip on the right hand edge of the metal
casing is located over the front of the black plastic edge connector.

    Lift the top two circuit boards up at the front edge and place the
four round polythene stand offs over the top of the four black plastic
posts which are projecting through the bottom board. Fold the centre
board back down. Unclip the top board and carefully fold it over to the
right. Take the four screws and tighten the centre board down which
will also retain the bottom board.

>>MORE)

24

Plug the loose power connector into its socket on the centre board,
placing the excess wiring on top of the centre board. Clip the top board
back into place.
    Remove the ITUG customer service information details from detail
diagram 2 by cutting along the dotted lines and adhere to the outside
surface of the top metal casing. Do NOT obstruct the ventilation holes.
After allowing it to dry, locate the top metal casing over the whole
assembly.

    Before tightening the three screws, check that no wiring has been
trapped under the casing and that the lip on the right hand side is
located over the front of the edge connector.

    Check that both the mains connector socket and on/off switch are
correctly seated on the black baseplate i.e. as far down as they will
go. Replace the loose piece of plastic which surrounds the disk drive
ribbon edge connector with the wide side at the bottom and the ridges
towards the inside. Place the top cover in position over the black
baseplate and screw it back together with the six cross headed screws.

TESTING

    Reconnect the controller to your system and test it, assuming that
you already have a double sided disk drive.

    Initialise a disk indicating that you want double sided formatting
and you should be rewarded with 718 sectors free. - See NOTES below.

Welcome to the club! You can now get double the storage on every disk..
(360 x 2 = 720 - 2 directory = 718 free sectors.)

    If it doesn't work,then you have done something wrong or have yet
to switch the unit on or reconnect the power lead...

    Either way, mine works so don't ring me, I'll ring you!!

DISK MANAGER NOTES

'DISK MANAGER' Module

    The original 'DISK MANAGER' module is NOT capable of formatting a
double sided disk although all the other functions of the module are
compatible with double sided disk drives.
    It is recommended that users of this module obtain a disk utility
such as DM1000 which is compatible with double sided drives as well as
providing extra functions which are not available on the module.

'DISK MANAGER 2' Module

    The 'DISK MANAGER 2' module IS capable of formatting a double sided
disk as it comes, so no problem will be encountered. Users may still
wish to obtain a disk utility such as DM1000 to take advantage of the
extra utilities available.

                                                        ITUG 01/08

>>END

                                                            .25

```
       ---------------------------------
       B E G I N N I N G   B A S I C
       ---------------------------------
```

By PETER BROOKS

LESSON 2 (continued)

If humans were to behave exactly like computers (especially the 4A) then
anyone who faithfully read the last article in V3.7 would probably have
starved - and died puzzled. This terminal turn of events would have
occurred (a) because you would still be staring at a TI BASIC screen
and (b) I didn't include the instruction <ENTER> after the last
example, and you (being a computer) would still be waiting for a
command.

You might have some idea therefore, as to why computers are not about
to take over the world...

Back to more serious (but light-hearted) things. This series is
intended to educate without reproducing the nastier aspects of school
life (like school dinners, detentions, boring History lessons, and
teachers with odd habits), so if at times it seems to talk down to the
reader, well, get up off your knees...

To continue. The last example was intended to show that HELLO,MOTHER
might not be what it seems, and that the punctuation marks mean more
than you might think. You may have noticed that without those
quotation marks (or quotes for short), the phrase HELLO,MOTHER becomes
0   0. Those two characters are not letters O but zeros, and
their distance apart is caused by the comma. The explanation behind
this bizarre event is both simple and complex: once you are in
possession of a few facts, it's simple, but providing you, the reader,
with those few facts is a complex task.

We'll start by looking at the word PRINT which was introduced in
episode 2 but not explained. It is one of the instructions which the
4A can "understand" and is considerably more complex than at first it
appears. I won't delve too deeply into its innards here - we'll
examine it properly later on. It is a useful "tool", since it can be
used without learning to write an entire program, and it will enable us
to examine other facilities.

The example from episode 2 was acted upon by the computer as soon as
you pressed <ENTER>. This "immediate" response lends its name to the
IMMEDIATE MODE, which is the mode that the machine is in when you first
select TI BASIC.

The term "mode" can be confusing - why modes exist, and why you should
want them in the first place, are TI's little secret, but they have
seen fit to describe various behaviours of the machine as "modes", and
you're bound to see instructions like "do this in the Immediate mode"
or "enter RUN mode" in other publications, so I suppose it does pay to
at least have a basic understanding of what modes are.

For example, if you tell me to go and jump in a lake - but I don't do
it (!) - you could say that I had listened to you in "joke" mode or in
"take whatever you say with a pinch of salt" mode. If I was in an
"Obey" mode or an "Anything you say O Master" mode, then I would
probably now be quite wet.

All this garbage is intended to try and convey the idea that the
machine will respond differently according to the mode it is in. In
one mode, pressing a key might have one effect, but a totally different
effect in another mode. When the computer is not running one of your
programs, and not having one written into it, but responds immediately
when you press <ENTER>, it is said to be in IMMEDIATE MODE. Initially,
all our examples will be given in Immediate mode, but later we'll
examine other modes and their uses.

And so to the word PRINT. Well, almost. To make sure that there are
no misunderstandings, check the keyboard for a key labelled ALPHA LOCK.
It is located at the bottom left of the unit, and behaves a little like
SHIFT LOCK on typewriters. It has two positions - Pressed (down) and
Depressed (up - confusing, isn't it ?).

Press it several times and you'll soon recognise the two positions. For
our purposes, we want it in the PRESSED position. To check, type the
letter A and then the digit 1. If the A is much shorter than the 1,
the alphalock key is in the wrong position. Press it again and type A1
again. If all is not well, you may have a faulty keyboard! (Contact me
for repairs).

If all is well at this point, you can press <ENTER> if you wish - the
computer will treat you to an error message, but ignore it - or press
and hold down the key labelled FCTN (bottom right) and press the 4.
This is a simple way to bypass the computer's error message, and only
works like this in the Immediate mode. It's called either BREAK or
CLEAR, (later you'll find out why) and you might like to remember it
for future use if you make any mistakes. Later we'll examine more of
the keyboard functions, but for now let's concentrate on PRINT.

The computer will respond to this word, if you type it in and press
<ENTER>, by scrolling the contents of the screen up one line. Because
we are using the Immediate mode, and in this mode the computer likes to
start a fresh line after each successful ENTER, the screen will scroll
up two lines just to cause confusion.

PRINT can be used in programs, sometimes to scroll the screen up a line
or so, but it can also be used in conjunction with other things. Those
"other things" are the real subjects of this and subsequent articles,
and we will use PRINT to explore them.

One of the many things you can tell PRINT to do is to print a number,
thus:

    PRINT 24

As you saw from V3.7, PRINT can also be used with things in quotes,
like this:

    PRINT "HELLO FOLKS AT HOME!"

>>MORE>

These two examples show the two basic types of "information" that this computer can handle — NUMBERS and STRINGS. Anything in quotes is a STRING - even PRINT "24", which as I hope you will come to see, is different from PRINT 24.

Some people find it hard to understand (a) why anyone should need to distinguish between numbers and strings, and (b) how a number can also be a string!

The answer to (a) is that the computer can treat numbers and strings differently. Numbers can be used in calculations, but strings can't, while strings can easily be chopped about, but numbers can't. We'll look at the processes involved in "calculating" and "chopping about" later.

The answer to (b) is that a number can also be a string of digits: "99/4A" contains two 9s and a 4, with the letter A and an oblique, but it can't for example be used in a calculation. Under certain circumstances the computer can change strings into numbers and vice versa, and the reasons why anyone would want to do this will become apparent later.

Now try this:

        PRINT "2+2"

The computer simply does as it is told. Do the same again, but this time omit the quotes:

        PRINT 2+2

The computer prints 4 instead. In the first example, the quotes tell the computer that it is dealing with a string, but once they're removed the machine regards 2+2 differently. It "sees" the numbers but also recognises the "+", and without being asked to, calculates the result of 2+2 and prints that instead. It does so, not because it's a smartass, but because it has been designed to. Any valid mathematical equation or expression, when used with PRINT, will cause the computer to calculate the result, but ONLY when there are no quotes.

Now, there are exceptions to this, but to keep life simple we'll not cover them here. For the time being, as far as you are concerned, items in quotes are TEXT, items not in quotes are equations for the computer to evaluate.

I've only used "+" here, but there are, naturally, others. However, in order to avoid confusion, (and becase there are no alternatives) the computer uses the hyphen (SHIFT OBLIQUE or -) for the subtraction sign, the ASTERISK (SHIFT 8 or *) for multiplication, and the OBLIQUE (/) for division. Thus 4 divided by 2 is 4/2, 8 multiplied by 7 is 8*7, etc.

        PRINT 4/2

Try a few simple (or complex) equations to become familiar with them (and possibly confuse yourself with the results!).

>>END

28

```
---------------------------
C O N C A T E N A T I O N
---------------------------
```

What it is, and how it's done...

Roughly speaking, the process of CONCATENATION is one of tacking things
together. In its application in BASIC on the 4A, it applies to
strings, although of course you could do it to numbers by turning them
into strings, performing concatenation, and turning them back into
numbers again.

The symbol for concatenation is the ampersand (&), and it is an
extremely useful operator. It can only be used with strings and
string variables, and its effect is to join two strings together:

```
     PRINT "JOE" & "BLOGGS"
```

It doesn't automatically insert spaces, so unless you left a space
within the quotes (either after JOE or before BLOGGS) you will have
printed JOEBLOGGS.

It can be used in PRINT statements, as above, in LET assignments:

```
     A$="JOE " & "BLOGGS"
```

in IF-THEN tests:

```
     IF A$ & B$ = C$ & D$ THEN 1000
```

and so on.

Probably its most useful application is when SEG$() is being used to
segment or slice a string or the contents of a string variable. For
example:

```
     FOR I=0 TO 150
     PRINT SEG$(STR$(I/1E3)&"000",2,3) & " ";
     NEXT I
```

This rather awkward example shows a simple routine to print out the
numbers 0 to 150 in such a way that all the numbers have leading and
trailing zeros where necessary. Thus 0 becomes 000, 10 is 010, and 100
is 100. (I leave you to find a reason for wanting to do this...)

The routine works like this:

I is divided by 1000 (1E3 is a sort of scientific shorthand for 1000).

>>>MORE)

29

This produces a range of values from 0 to .15 inclusive, ensuring leading zeros.

The STR$() function then turns these numbers into strings, and the first & tacks "000" onto the end of each.

This produces a range of strings from "0000" to ".15000", ensuring the existence of trailing zeros.

SEG$() then extracts the "substring" or segment, which starts at position 2 (2nd character in the string) and is 3 characters in length, removing the first character (which is the period/decimal point for all cases except "0000" when it is 0).

This produces "000" to "150", with numbers like 1, 10, etc., being converted to "001", "010", etc.

Finally, another & is used to tack a trailing space onto each substring of three characters (to space the groups out), and a "pending PRINT" is set up through the use of a "trailing" PRINT separator – the semi-colon (;). This ensures that the groups are printed the maximum number of times per line (i.e., seven).

Alternatively, you may want to manipulate a "number" like 123456789, to allow for selection/deletion. One way to do this involves a bit of maths and the use of "data packing techniques" (the subject of a future article). To delete the 3 for example, you could divide by 1E7 (or 10,000,000) giving 12.3456789. Copy the INT() of this value into a variable and subtract it from the value:

$$12.3456789 - 12 = .3456789$$

Now multiply by 10 to get 3.456789, subtract the INT() again to give .456789, and divide by 10 to give .0456789. Now add in the variable (12):

$$12 + .0456789 = 12.0456789$$

and finally multiply by 1E7 to get 120456789.

Phew! The 1E7 is related to the 3 mathematically – 1E(10-N), where N is the digit concerned. If 4 was to be replaced by 0, then 1E(10-4) or 1E6 would be used.

There are easier methods, and one of them involves &. Turn the number into a string:

N$ = STR$(123456789)

and the 3 sits in the 3rd position in the string. Slice it up into two substrings:

)))MORE)

30.

SEG$(N$,1,2) and SEG$(N$,4,6)

and simply stick a "0" in between the two:

        N$=SEG$(N$,1,2) & "0" & SEG$(N$,4,6)

This results in N$ containing the string "120456789" which could be
turned back into a number using VAL(N$).

Note that the "general" form for the split into two substrings is:

        SEG$(N$,1,N-1) and SEG$(N$,N+1,255)

where N stands for the position of the digit to be removed/altered, and
also stands for the digit itself. The 255 value is a facility of
SEG$(), where its use indicates that the length should be equal to the
remainder of the string.

If you have followed these explanations, and understood the examples,
you have probably seen all sorts of pitfalls (like what happens in the
last example if you delete the "1" and then try and turn the string
back into a number!), but bear in mind the fact that these are simply
short-term examples. Catering for all the pitfalls (or holes) would
have complicated the examples no end, and you wouldn't have seen the
wood for the trees!

Finally, another use of &, which came out of a question posed at the
first BLOXWICH WORKSHOP in 1986. A 4A programmer wanted to take a
string and reverse its contents, so that for example "FRED" would
become "DERF". He had devised a simple BASIC routine using a loop and
SEG$(), but found that it didn't work the way he expected. The
resulting reversed string always contained only one letter - the first
in the original string - and it had him puzzled. This isn't his
routine, but it is similar:

        100 CALL CLEAR
        110 INPUT N$
        120 FOR I=LEN(N$) TO 1 STEP -1
        130 A$=SEG$(N$,I,1)
        140 NEXT I
        150 PRINT A$
        160 GOTO 110

If you enter and RUN this, and type in, say, "FRED", you'll see the
effect.

The solution involves &. What's happening on each cycle of the loop is
that one letter is being assigned to A$ from N$, but that letter is not
being tacked onto the end of A$ contents. Instead, it is REPLACING
whatever was previously in A$, with the result that, at the end of the

>>MORE>

loop, A$ contains only the last single character extracted from N$.

Line 130 should read:

```
130 A$=A$&SEG$(N$,I,1)
```

so that it adds the letter selected from N$ onto the end of the existing contents of A$ — compiling the string, if you like.

In addition, the loop in 120 could have been written as:

```
120 FOR I=1 TO LEN(N$)
```

which would mean that 130 might change to:

```
130 A$=SEG$(N$,I,1)&A$
```

Two other alternatives are A$=SEG$(N$,LEN(N$+I-1,1)&A$, or A$=A$&SEG$(N$,LEN(N$)+I-1,1). Can you see what form the loop would take with each alternative ?

And can you see one glaring fault with my example routine ? The GOTO makes the routine run again, and on the second and subsequent passes, the new entries in N$ will be reversed and added to a growing A$. Eventually A$ will overflow (greater than 255 characters) and you'll get a warning message (of the STRING TRUNCATED variety).

The solution is to "clear" A$, either before the loop or after it:

```
155 A$=""
```

The two quotes are placed together with no intervening space — it's called a NULL STRING. Don't do it before line 150 or you'll never see what the reverse of N$ is!

If you have any problem with getting a routine to produce a specific effect, or if you have managed to overcome a problem in BASIC which had seemed insurmountable, why not send in your information/question to me and I will try to answer the question and/or publish the information.

>>>END