Tidings V1 N4 July 1981

The name is not shown in this issue but this IS Tidings No 4.

This may be the first issue not printed by TI???

The first of the very long articles by Peter Brooks. Peter had an NTSC TI99/4, and went on to write a TI Book (Mastering the TI99/4a) and produce a number of his own magazines (various names but often referred to as TI Lines or later International TI Lines). Peter then moved to the USA and took no TI material with him.

This issue has a membership list with a total of 19 names.
If we take off the overseas addresses, Paul Dicks, and commercial addresses, we have a UK membership of 11 members. The PAL 99/4 had not yet appeared.

In this issue is news dated 26.6.81 about the upgrade to 4A.
The next day TI in the UK invoiced me for a yet-to-be-available PAL 99/4.
So we know where the old 99/4's ended up-
fitted with a PAL video chip and connector and shipped out to Europe.
I had to pay GBP 300 for what was really a discontinued line.

blackbox

TIHOME

THE TI 99/4 USERS GROUP

VOLUME 1 , NUMBER 4 MORDEN SURREY 1st JULY 1981

In this issue:

- - - - - - - - -

EDITORIAL

It strikes me that the time has arrived when as many of us as possible should get together at a convenient place and time, and perhaps over a few drinks, to discuss what WE are going to do to set the group on a more formal path. Please get in touch with me and let me know where and when would be convenient to you, and I will try to arrange a convenient date and place, and will arrive with a sample Constitution and Rules that we can discuss.

The other thing I would like you to do for me is to check your name and address and telephone number in the membership listing and let me know any changes. At the same time perhaps you could let me know precisely what TI equipment you have. I think it will be useful if we keep a register of the hardware available among members.

I do not, obviously, expect people to fly in from America, Holland or Australia but I will report the results of our meeting either in the next newsletter or in a personal letter to all members.

Here's hoping we can arrange a date and time.

I look forward to meeting you all.

FOR PETE'S SAKE!

Over the next few issues of TI HOME I hope to present a number of useful programming hints for the 99/4; however, because I'm writing 'blind' - I don't know what programming experience you all have - I will need some form of feed-back if my suggestions are going to be of any use to you.

So if you think the explanations are toosimple, not simple enough, or that it's all a load of old wombat droppings, please write in and say so.

Right, enough waffling, down to some brass tacks.

If you are an avid reader of computing magazines, you may have noticed that other machines can implement the Boolean operators 'AND' and 'OR' . The 99/4 is also capable of implementing them; their counterparts are '*' and '+' , respectively, when used in conjunction with relational expressions within parentheses, and they are most often used in IF...THEN statements, especially when performing input validation on values obtained using the CALL KEY command.

For example, you may (for whatever reason) want a program to branch thus:-

```
2000  IF  A = 1  OR  B = 10  THEN  4000
```

which will result in the program passing control to line 4000 if either or both of the expressions is TRUE.

On the 99/4 that becomes:-

```
2000  IF (A = 1) + (B = 10)  THEN  4000
```

What is implied in this statement is:

```
2000  IF (A = 1) + (B = 10) <>  0  THEN  4000
```

(Remember Paul Dicks' reminder in issue 3 ?)

If you check pages 42 (Relational Expressions) and 79/80 (IF..THEN..ELSE) of the handbook, you will see that the 99/4 will evaluate a relational expression and return a value of -1 if the expression is TRUE, or 0 if FALSE. Placing the expressions within parentheses ensures that the correct expression is evaluated;

leaving out those parentheses in the example would permit the 99/4 to evaluate ' (1 + B) = 10', and to compare the result (-1 or 0) with the contents of A, which is not what was intended.

This is how it works.

If neither expression is TRUE (i.e., when evaluated, the value returned in both cases is 0) then the sum of the two values will also be 0, and the program will not branch - control will pass to the next line (or to an ELSE statement, if one is present).

If, however, one or both of the expressions is TRUE, the sum of the two values will be either -1 (if only one is TRUE) or -2 (if both are TRUE) - at any rate, the result is non-zero, so the program will branch to line 4000.

This type of statement can also contain string, or mixed string and numeric, expressions; for example:-

```
2000  IF (A>1) + (A$<>"UP") + (B<=C*D) + (B$=C$(7,2))  THEN  4000
```

Those, then, are examples of the implementation of 'OR' on the 99/4.

The implementation of 'AND' is similar; for example:-

```
2000  IF  A = 4  AND  B = 17  THEN  4000
```

for the 99/4 becomes:

```
2000  IF  (A = 4) * ( B = 17)  THEN  4000
```

Again, the parentheses are necessary to avoid the comparison of the contents of A with the evaluation of '(4 * B) = 17'.

In this case, if neither, or only one, of the two expressions is TRUE, the product of the two resulting evaluation values (i.e., -1 or 0) will be 0; that is, 0 * 0 , or 0 * -1, is 0, and the program will not branch. Only when BOTH expressions are TRUE (i.e., -1 * -1) will the product be non-zero, and the program will branch.

Again, this type of statement can also contain string, or mixed string and numeric, expressions; for example:-

```
2000  IF  (A$ = "UP") * (A = 4) * (B$(2,1) > SEG$(C1$,1,6))  THEN  4000
```

Don't forget that the branch will only occur when ALL the expressions are TRUE.

A mix of 'AND' and 'OR' -type expressions can also be used; for example:-

```
2000  IF  (A$ = "YES") * (B = 6) + (B$ = "ON") * (C = 10)  THEN  4000
```

This statement is equivalent to:

```
2000  IF  A$ = "YES"  AND  B = 6  OR  B$ = "ON"  AND C = 10  THEN  4000
```

In other words, a branch will occur if either one, or both, of the ANDed pairs of expressions is TRUE.

Having waded through all that (I hope you're still with me !), how would we use such relational expressions for input validation, with, for example, the CALL KEY command ?

Let us suppose that, as part of a game, we wish to restrict the valid input to the keys 1 - 4 inclusive, using CALL KEY, and without disturbing the screen with PRINTed error messages. The CALL KEY command, when used with a key-unit of 0 (the entire keyboard - see the relevant pages in the handbook), returns values for 'key' (the ASCII code for the key if one is pressed, or -1 if no key is pressed at the time the command is used) and 'status' (either whether a key is being pressed, or whether the same key is being pressed as when the last CALL KEY command was used). Doing it the long way round:-

```
500  CALL KEY (0,K,S)
510  IF S = 0 THEN 500
520  IF K = 49 THEN 560
530  IF K = 50 THEN 560
540  IF K = 51 THEN 560
550  IF K = 52 THEN 560 ELSE 500
560  continuation of program
```

No-one, I hope, would seriously try to program it that way.(Hands up all those who have !)

You could conceivably try the shorter:

```
500  CALL KEY (0,K,S)
510  IF S = 0 THEN 500
520  IF (K = 49) + (K = 50) + (K = 51) + (K = 52)  THEN 530  ELSE 500
```

However, if I've led you gently enough through the reasoning, you should be able to see that, as the keys 1 - 4 constitute a nice, simple range, we could test for keys that lie OUTSIDE that range rather than within it, and save a bit of space and time thus:-

```
500  CALL KEY (0,K,S)
510  IF  (K<49) + (K>52)  THEN   500
```

Just in case you haven't followed me, I'll explain how it works. In this instance, we don't HAVE to test the status, as the value returned to the 'key-variable' (K, in case you hadn't guessed) lies well outside our valid range of ASCII codes of 49 to 52. If we press a key which is outside our valid range and which has an ASCII code which is lower than 49, then one of the two expressions will be TRUE (i.e., K<49) and the sum of the two values will be non-zero, so the program will jump back to line 500. The same applies if the key we press is outside our valid range and above ASCII code 52; the program will jump back. If however the key we press is either 1,2,3, or 4, then NEITHER of the two expressions will be TRUE, so the values returned will both be zero, the sum will therefore also be zero, and the program will continue from the next line number (or from an ELSE statement, if present).

You should now be able to see that it is possible to validate split ranges of keys; for example, what about the keys which represent the Hexadecimal numbering system ? They fall neatly into two ranges: 0 - 9, and A - F. To accept only these keys requires simply:

```
510  IF (K<48) + (K>57) + (K<65) + (K>70) = -2  THEN   500
```

Now those of you who got their Observation badges in the Cubs will have spotted a teensy little inconsistency in the above program statement. Hands up all those who noticed that '-2' which has crept in. (Who's that snoring ?) The best way of explaining why the -2 is necessary is by detailing the TRUTH table for a given input, and this is given on the next page. The -2 could be rendered unnecessary by performing the ABS function on the second and third relational expressions, but this would lengthen the statement line, and I would have had to explain away even more !

Key pressed; code	(K<48)	(K>57)	(K<65)	(K>70)	SUM
ENTER ; 13	-1	0	-1	0	-2
3 ; 51	0	0	-1	0	-1
B ; 66	0	-1	0	0	-1
G ; 71	0	-1	0	-1	-2

TRUTH TABLE FOR INPUT VALIDATION

I hope that you can see that valid inputs produce a value sum of -1, while invalid inputs produce a value of -2; so simply using the usual test for the sum being non-zero will not differentiate between valid and invalid inputs. (You could also try including '1 + ...' at the beginning of the list of relational expressions, so that only invalid keys produced a non-zero sum: the list of possibilities is virtually never-ending !). By now, if you are still awake, you may be becoming aware that there is no 'right' way to program: there are so many ways of implementing the same function that it eventually boils down to deciding whether to consume space at the expense of speed, or vice versa.

For a moderately large number of valid keys, which perhaps do not fit into neat ranges, one possibility is to store them in an array, and loop through each time there is an input, jumping out of the loop if a match is found, and returning to the input line when no match is found. A faster method is to use the POS function, taking the actual valid characters rather than their ASCII codes. This form of input validation is also suitable for use with the INPUT command, provided that the valid keys or key groups (i.e, words) are separated by some character not used - e,g., the asterisc '*', - to prevent someone from causing errors by typing in only part of the valid keyword, which will be accepted by the POS method of validation. It's simpler when using CALL KEY; for example, if we wanted to restrict the valid keys to '1,2,3,C,D,L,R,U,Z', a routine might read:-

```
750  CALL KEY(0,K,S)
760  IF  S = 0 THEN   750
770  IF  POS("123CDLRUZ",CHR$(K),1) = 0  THEN   750
```

This time we really ought to have a status test included, as if no key is

pressed, K is assigned -1, and an error will result from the attempted evaluation of CHR$(-1). You could, of course, use ABS on K within the POS statement, and therefore dispense with the status test, as CHR$(1) is a valid operation. This might cause problems if one of your valid keys was Shift A, however; see Paul Dicks' remarks in issue 3's Tips For Programmers. Strictly speaking, line 760 ought to contain S<= 0, as just S = 0 permits the program to 'fall through' if more than one CALL KEY command is used and the key is held down too long: this could cause the user to overshoot or select incorrect options in, for example, a routine offering a menu of options.

You don't have to place your valid key characters within quotation marks in the POS statement, of course. You can store them in string variables, arrays, concatenate them while performing the POS command, etc., etc. You could even restrict the valid keys themselves by altering the position given in the POS statement for the start of the search for the presence of string 2 in string 1: the handbook gives further details.

The use of POS and relational expressions is not restricted to IF...THEN statements; you can use POS with for example ON...GOTO or GOSUB thus:

```
720  CALL KEY (0,K,S)
730  ON 1 + POS("12ABUDLRC", CHR$(K), 1) GOTO 720,1000,2000,3000,4000,
     5000,6000,7000,8000,9000
```

In the above example, 1 is added to the value returned by the POS command, (if not found, 0; if found, then a value between 1 and 9 depending upon position), so that if the input was invalid, POS returns 0, add 1 and the program will pass control to line 720 (the first in the line list), and if the input was valid, control will pass to the relevant section of program beginning at the line number specified in the line list. Neat, eh ?

You could also try:

```
720  CALL KEY (0,K,S)
730  ON ABS((K = 71) + (K = 78) * 2 + (K = 89) * 3) + 1 GOTO 720,100,
     2000,5000
```

In this example, if none of the expressions is TRUE, the sum of all the values within the ABS() brackets will be zero, add 1, and the program will jump back

to line 720. If the first relational expression is TRUE, the sum of all the values will be 1, add 1, and the program continues from the second line number in the line list; if the second expression is TRUE, then the sum of all the values will be 2, add 1, and the program continues from the third line number, and if the third expression is TRUE, the sum of all the values will be 3, add 1, and the program will continue from line 5000 (the fourth line number). Note that this is NOT a mixture of AND and OR -type statements, as the relational expressions are not multiplied by each other. The ABS function is necessary as the results of the evaluation of the relational expressions and their multipliers will be negative if TRUE, and, as the handbook will tell you, if the values obtained are not positive, non-zero numbers, an error will occur, and the program will crash.

If you have been bewildered by the explanation of the operation of the last example, try working it out with your 99/4, or on paper, bearing in mind that there is a 'hierarchy' to the mathematical operations; i.e., multiplication will be performed before addition (see page 40/41 of the handbook on Numeric Expressions), and expressions within parentheses will be performed first.

Well, that should do for this month. I had originally intended to write only a couple of sides, but as Paul Dicks will readily attest, my attempts to be brief usually fail dismally.

If there are any points in this article which you feel have not been explained to your satisfaction, please put pen to paper and let someone know, and if there are particular functions which you would like clarified, feel free to draw on the cumulative knowledge and experience of the whole membership. Someone is bound to have the answers to your questions.

Next time I intend to cover the insertion/deletion of string contents, outputting text to the screen using H or VCHAR, and some manipulations of graphics definition hexstrings, if anyone can stand the strain.

Good programming,

Pete Brooks

The good news is that TI Bedford have now returned my 99/4 to MicroInstruments, and I have only to transport it back home and start inundating you with programmes (I hope). The bad newsis that they sent no details of any fault found, so presumably they didn't find anything - but it took them seven weeks not to find it. I wonder if they'll extend the warranty to cover the time it was away.

You and I know what a flag is, how to produce one, and how to implement it, but how many of the other members do? And an explanation of Recursion, and why professionals frown on it's use (at least, if you believe the popular computing press, they do)? In short, why not publish as a separate sheet at some future date, a detailed list of buzzwords and their explanation, together with their relevance to the 99/4?

One other point about any programme for the 99/4: the shorter you can make the labels of the variables, arrays, etc. the better. Although the 99/4 has a fantastic potential for creating meaningful (whatever that overworked word means) labels like RUNNINGAVERAGE or SPACEINVADER146. they take up a lot of room both on screen and in the variable list.

Pete Brooks
A.E.R.E. Harwell
Didcot

QUOTES FROM U.S. USERS-GROUP NEWSLETTER:

Texas Instruments will release for the more serious programmer UCSD Pascal. We have seen a demonstration of this system and find that a more powerful language is handled quite well by the 99/4. This will pave the way for many new programs to be developed for the 99/4 which should run faster and more efficiently.

Extended Basic greatly expands the flexibility, programming efficiency and general capability of the 99/4. The added ability to merge, auto boot from disc, find program size and protect programs all pleases us. Some programs will have to undergo some slight alterations due to changes in color and Char definitions. The speed of executing a program in Extended Basic is no faster than in regular T.I. Basic and in some cases it is slower. We also noticed a loss of memory core with the Extended Basic Command Module plugged in, 13,928K without a disk controller and 11,840 with. The ability to move up to 28 different moving objects (sprites) and many other new features of Extended Basic overcome any negative reaction we may have had about it.

New Expansion RAM is a 32K version that does in fact plug into the right side 1/O port and is the same size as a disk controller and the RS232. In order to access the expansion RAM you must use the Extended Basic Command Module.

QUOTES FROM U.S. USERS-GROUP NEWSLETTERS (CONTINUED):

With the recent announcements of new peripheral devices and higher program language capabilities the 99/4 certainly possesses the potential of being one of the most powerful personal computers.

The long awaited release of both Extended Basic and Expansion Ram has been delayed by Texas Instruments. T.I. at the last minute found some slight bugs in Extended Basic and rather than ship product that may not meet the consumers expectations a decision was made to do a total rework.

There is an effort under way by Mr. Paul Dicks of England to start a European Users Group; however, Mr. Dicks said he was experiencing some of the same problems and difficulties that we faced when we first started our Users Group, this being the lack of product availability and software.

The 99/4 is not a 16K machine. Basic language resides in 1,208K so we start with 14,792K and if we add a disk system we loose an additional 2,088K so we are now down to 12,704.

For those of you who use cassette tape only, there may be several reasons why a given program will not load properly. Not all cassette recorders are compatible with the 99/4 computer. If programs that you receive from us will not load, 9 chances out of 10 your cassette equipment is not compatible with ours.

TI NEWS

Last issue I gave you some details of the new PAL version of the 99/4. I suggested that its introduction would mark the beginning of a new stage in its life. TI have now released the one vital piece of information that we have all been waiting for. I understand that the price of the PAL 99/4 will be £299.95, including the modulator. This price brings the unit into the reach of a great many people. Less than the cost of a video recorder. Perhaps we are seeing the beginning of the genuine home computer taking its place in the household of the future. I expect that TI will shortly be receiving supplies of hardware in this country and the units will be promoted and sold shortly afterwards. I look forward to seeing our membership rise at a realistic rate.

Unfortunately, my visit to TI to look at and try some of the new goodies has had to be postponed but I hope to get up there early in July and will report back to you as soon as possible.

Users' club

A USERS' club for the Texas Instrument T199/4 home computer has been formed in Surrey in association with the American 99/4 Users' Associaton. Called TIHOME, the club is open to all users, and members receive a monthly newsletter and access to a software library. For further information contact Paul Dicks, 157 Bishopsford Road, Morden, Surrey. Tel: 01-640 7503.

Computer Weekly

19.6.81

SOME NOTES ON THE TI THERMAL PRINTER

I ordered my TP 'on spec' soon after buying the 99/4 and, by and large, I am now quite satisfied with its performance so far.

As a onetime letterpress printer, I was at first disappointed at the quality of the print. After some experience (and not a little reading up about computer printers in general) I found it best to concede that print quality on even the most sophisticated daisy-wheel job would never quite equal the best letterpress printing.

Inevitably, the first use I put the TP to was program listing. I soon found that leasurely 'debugging' a longish program in the form of hard-copy is much less arduous than trying to work with the VDU. This facility alone makes the TP worth while. However, it was soon on to bigger and better things. I already had on tape-file (via the PERSONAL RECORD KEEPING module) detailed records from my office. Sorting these into various listings, I ventured a printout (in tabular form). The daunting six-foot-plus strip which emerged was soon chopped into A4-length pieces and pasted up, three-abreast, on to A4 sheets, which photocopy beautifully (You can't even see the join if you are careful). I now do an update every month in the same way and find it more than adequate for the job.

Spurred on by this minor success, I looked for other ways of 'stretching' my TP. Whilst producing one or two NO-SMOKING type signs, using 'banner' characters along the length of the paper, I discovered one of the snags of the TP, namely the gap separating each matrix from its neighbour. Why couldn't the 'U.' mode have an 8x8 matrix, like the screen, and the 'S.' mode really close the gap between lines, thus enabling 'solid' blocks of graphics to be printed. This seems to be a common fault with thermal printers.

The next job was to re-define the 'number' characters (0-9) as random graphics and use these as the basis for a simple but amusing logic game (see ARITHMAGRAPHS in the TIHOME library)

My latest effort is to try to adapt the WORD PROCESSING program from the Library, for the TP. This was mainly to get some idea of how 'wo-pro' programs work and I have submitted a reasonably workable version to the Library.

Mike O'Regan

POSTSCRIPT

Once upon a time, I went out to a business lunch and enjoyed the wine, amongst other things. Back in my department, I overheard a conversation between confirmed PET User Group members, and thought to myself, 'why haven't I got a user group?' It was probably the wine speaking. So, using the service of my secretary, I dictated a short letter which was sent to all computer newspapers and magazines. That, Heaven help us, was in September, 1980. I hung on and waited to see what sort of response I would get. When I had sufficient members to warrant production of a newsletter I issued the first one, that was the 1st April, 1981. This has been number 4, four issues in six months - not bad considering the static state of the market. Where it will go from here depends a great deal on you, the members. For information I refer you to the postscript of the issues dated the 1st July, 1982, and in subsequent years.

PMD

STOP PRESS Computer Weekly 26.6.81

Texas fights 'customer resistance' to 99/4 with an upgraded model

TEXAS INSTRUMENTS in the US has introduced an upgraded version of its 99/4 personal computer, called the 99/4A. The 99/4, with all its selling problems, will probably be phased out of production.

Since its launch over a year ago, the talking home computer with colour capabilities has not attracted the expected level of sales.

TI refers to the lagging sales as "customer resistance" to the keyboard, software and cost.

Priced originally in the US at $950, the 99/4 was unattractive compared with its competitors in the colour computer field. Tandy's unit with 16K of RAM, which was recently launched in the UK, costs $399 in the US, and Commodore launched its 5K Vic colour computer for just $299.

Since then, TI has dropped its price to $650, after a rebate programme in the US failed to improve sales. The programme was introduced last summer and involved $200 worth of coupons given to each customer for $100 worth of TI software and $100 in cash.

Atari reduced the price of its 16K 400 colour computer when it realised that its competitors' units were cheaper. The price was knocked down from $630 to $399, when Commodore launched the Vic colour computer.

The 99/4A is now being offered in the US for $525, a drop of $125 on the 99/4 price. It also has an enhanced keyboard with upper and lower case lettering, in a standard typewriter style rather than the previous flat style. Users blamed the old keyboard style for the 99/4's slow movement.

Its new features include an automatic repeat function and keys designed to access specific computer functions. Like the 99/4 it contains 16K of RAM, expandable to 48K and 72K of ROM. The price cut was eased by designing out the 99/4's built-in equation calculator program.

The 99/4A will be sold in the US through TI's existing 99/4 distribution network, including the company's retail stores. However, delivery will not begin until the unit gets approval in terms of radio and television wave interference from the US Federal Communications Commission.

In the UK, the 99/4 has made quite a number of sales, according to a spokesman for the company. It is currently being refitted to take the British standard colour system, PAL, so that the unit can connect up to a television in the home. Up to now, users have had to buy a special colour monitor using the US colour standards.

No date for release of the 99/4A in the UK has been set as yet.

David Abbott
15A Langton Road
Tunbridge Wells
Kent 08.922-9641

Shane Anderson
P.O. Box 101
Kings Cross
Sydney
New South Wales
Australia

Ian Beattie-Edwards
20 Normanhurst Close
Three Bridges
Crawley
Sussex
RH10 1YL 02.932-0565

Peter Brooks
19 Hillside
Aldfield Estate
A.E.R.E. Harwell
Didcot
Oxford OX10

Steve Cooper
Scan Computers
Chanctonbury House
Church Road
Storrington
Sussex 09.066-5432

Datron Micro Centre
2 Abbeyfield Road
Sheffield
S7 1FD 07.425-85490

Paul Dicks
157 Bishopsford Road
Morden
Surrey 01.640-7503

W Gilchrist
13 Chorley Avenue
Sheffield
ST10 3RP 07.423-04351

Dave Hamilton
85 Swanspool
Ravensthorpe
Peterborough 07.332-65575

Clive Johnson
6 Windy Ridge
Gossops Green
Crawley
Sussex 02.932-9273

Paul W Karis
Blauwgras 2
3902 AA Veenendaal
Holland 010.31.8385-16000

Charles LaFara
P O Box 95148
Oklahoma City
Oklahoma
U S A Oklahoma City 73143

John Ashley
Texas Instruments Ltd
Manton Lane
Bedford
Beds 02.346-7466

Mike O'Regan
130 Stapleford Lane
Toton
Beeston
Notts NG9 6GB 06.076-5482

Brian Pain
40A High Street
Stoney Stratford
Beds 09.085-66660

Koenraad Rutgers
22 Marriotts Close
Felmersham
Beds
MK43 7HD 02.347-81730

Stephen Shaw
10 Alstone Road
Stockport
Cheshire
SK4 5AH

Martin Simpson
11 Main Street
Belton
Nr Uppingham
Leics 05.728-6625

Stewart Vane-Tempest
20 The Broadway
Woking
Surrey
GU21 5AP 04.862-71991