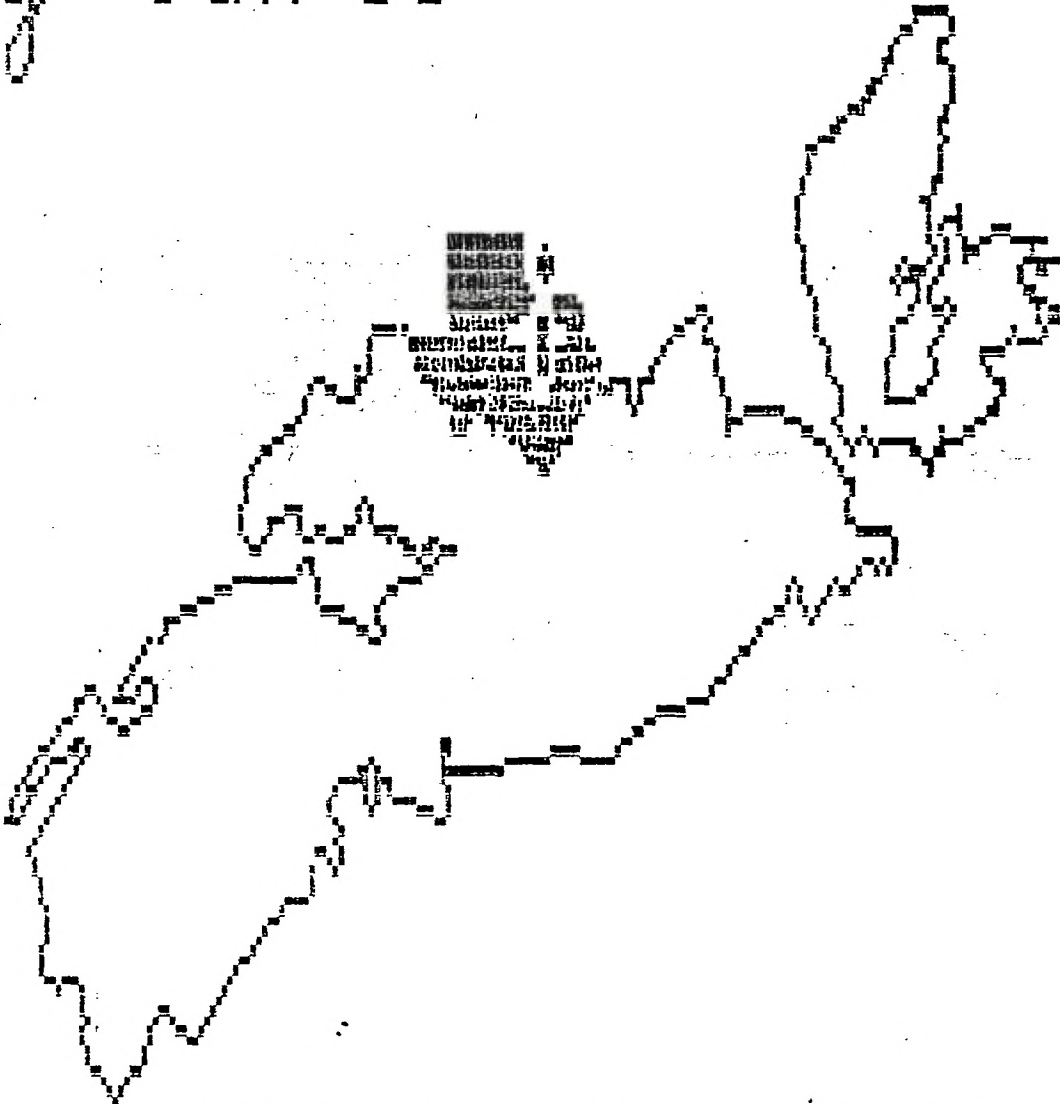


Sackville Nova Scotia

June 86

May - Jun 86



NEELER

TINS Newsletter is published on a monthly (or bi-monthly basis if there isn't enough to print), as the means of communicating ideas obtained from solicited sources to the general membership. Material appearing in this newsletter is copyrighted by compliance with federal regulations for basic copyright protection. The effective date of copyright is the 1st day of the month appearing on the edition.

Articles and programmes appearing in this newsletter are, to the best of our knowledge, original works except where indicated by the inclusion of the source. Arrangements have been made with other newsletter staffs and magazines for permission to reprint certain material. These articles are accepted on trust and the newsletter accepts no responsibility for searching the right-to-print of the originating periodical. Errors or omissions should be communicated to the editor as soon as possible.

Groups wishing to include material from these Newsletters in their club newsletters may do so providing the author and this source are mentioned. Other use of the material is subject to release by the editorial staff.

Space for advertising is available to merchants wishing to display their wares. Full page \$25, lesser sections at appropriately reduced rates [payable to TINS]. All commercial ads must reach the editor in pre-prepared, ready to print form, not later than the 1st of the month in which the ad is to appear. The Newsletter is on sale to members for \$1.00 per copy, non-members \$1.50. The price of each issue is solely to defray publication costs.

Back issues of the Newsletter are available on written request from the editor (Jan-Dec 86, 13 issues=\$15) (individual issues at \$1.50 each). All queries and newsletters should be forwarded to the address below, other correspondence should be directed to the Club at PO Box 3391, Dartmouth East, N.S. B2W 5G3.

Editor TINS Newsletter
321 Iony Hill
Lr. Sackville, NS
B4E 1M6

I just had the opportunity of getting my hands on a letter quality printer that didn't cost an arm and a leg. In fact it was most reasonable at about \$200 Cdn (PIO cable extra).

Since I have been concerned for some time about the condensed print being a bother to those with limited vision, I have decided that this will be an excellent opportunity to switch back to Elite 12 style.

The only difference will be the amount of information contained in the newsletter each month. This should not present too much of a problem, since very few people have been making contributions to the newsletter anyway.

More about the printer. This low cost letter quality printer is a Riteman LQ. It uses an impact printing system with a fully formed font style that is operated from multiple printing elements. Unlike the 'daisy-wheel' printers this unit has a drum of five wheels that spin back and forth to present the correct character to the printing surface. No ribbons are used as the print drums are brought into contact with an ink-pad in the printing process.

96 characters are available in Elite 12 only. Letter size is 2mm by 2.5mm. Common letter size paper 215.9mm in width (8 1/2 inch), single sheet friction feed. Print speed is only 12 chars/sec and the unit is bi-directional. Overall dimensions are W=298mm, H=63mm, L=198mm (roughly 12 inches across the front, 8 inches front to back, and rises almost 2 inches off the desk top)

If you have a use for a printer such as this, drop me a line. I can arrange to get you one quite easily. Please send comments on newsletter print style to the address in previous column.

Editor!



Computer shown at Ottawa Faire:

Yes, it was there, and it was operational, although it was still the wire-wrapped version. Steve Lamberti of Texaments gave the briefing on it's capabilities and limitations. I am not going to go into all the details, many of which have been published in MicroPendium and several other rags. The demo given was very short and actually did not prove too much. It showed GREAT speed to those who understood what the demo was accomplishing, but to others, it was not a demo at all. I was certainly disappointed at this aspect, as I had hoped by now that a more comprehensive demo could have been produced.

Regarding speed, it IS phenominal. Using a TMS9995 chip, running flat out at 12mhz (16mhz capability), it is faster than the 520ST. Steve said that a target date of June is set for production, but don't hold your breath. I will be surprised to see it by Xmas. Whatever....when it does come out, it will be shipped with an IBM style keyboard at a cost of about 600 bucks (US). For those die-hard TI'ers, I'm sure the cost will not be prohibitive. The unfortunate thing about it is, because it is a CARD computer, one would need a PEB to be able to use it. So, if you don't have a PEB, better start looking for one soon. I doubt Myarc will have a similar box ready in the near future. Another big drawback to this system is that to utilize the 80 column mode, you will need a hi-rez 80-column monitor. To purchase a fully-blown sytem, then, would cost more than a 520ST. So, for new users, I doubt if the system would sell very well. I guess time will tell. Meanwhile, work continues on the non-card type of computer. I guess they are running into all kinds of problems with it, and is going to take some time to sort out.

More impressive (to me) was the ongoing demo of Myarcs Xbasic. The speed of this alone has convinced me to consider purchasing it. The hi-rez graphics demo showed unbelievable

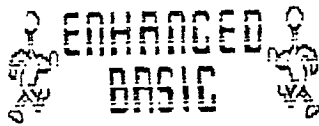
detail. In comparison with the demo on a 520ST, the ST had far better color resolution, but was equal in other respects. Now, when the new computer gets on-line, the above will undoubtedly change because it will be capable of one of 256 colors in any of the 524x424 pixels. Barry Comer, eat your heart out!

Well, that is the extend of what I want to say about the Ottawa show. I know that the Ottawa group will be writing a detailed article for their newsletter, and I certainly don't want to take the steam away from them. Maybe I will reproduce their article for TIBBS at a later date.

I will mention, though, that the trip was well worth it. The show was very exciting, the dinner afterwards was excellent. Jane LaFlamme is to be commended for a very fine first-time effort, and others who worked their butts off to make this a success are given my heartfelt thanks. I'm certainly looking forward to next years show, and hope to see many of the friends (old and new) at the Chicago show this fall.

I think Paul and I can answer any questions you may have. Keep them in mind





Instalment Two
Peter Brooks (TI-LINES Nov 84)

An enhancement of TI BASIC available through STATISTICS and PRK Modules.

Ref: TI Document
ARCHIVE.PRK.DIC.SUBRLS1 courtesy of TI
Articles by, and personal
communications with, Paul W. Karis

CALL P() - The PREP Subprogramme

CALLS A and D form the easy part of Enhanced BASIC. We move into murky wates with CALL P(), and I have to warn you that they will become murkier still over the next few instalments. You will also come across some new error messages - ILLEGAL CALL for one - upon which to exercise your debugging skills (an article on Debugging should appear after these articles).

In this episode we will examine the function and use of CALL P(), the PREP subprogramme. PREP is presumably short for PREPARATION, and performs something called PARTITIONING. The process essentially involves persuading the computer that it doesn't have quite as much memory as it thought it had.

On other computers you have tighter control over this aspect. On those you can store data, character definitions, machine code routines, or even another BASIC programme in the 'hidden' area. You can then use the 10K which the machine DOES know about, to write your BASIC programme which will make use of the data stored in the 'cordoned-off' 6K.

On our 99s, CALL P() tells the computer how much memory is available to TI BASIC. The remaining memory is then available for one use, and one use only, and that is as a DATA FILE (i.e., what PRK and STATISTICS are intended to create and manipulate).

CALL P() can only be used when there is NO BASIC programme in memory. Presumably this ensures that any existing BASIC programme will not be interfered with by its execution. In

turn this means that CALL P() cannot be used within a programme.

CALL P() is therefore entered in the IMMEDIATE MODE.

It must then be followed by NEW, in order to make the computer put into effect the changes in the allocation of memory. Although information is hard to come by, certain events can be assumed to occur. It seems that when NE is used, the computer checks one or more memory locations, and the contents of those locations dictate how much memory is available to TI BASIC. Normally the values at these locations are set by a built-in programme when the computer enters TI BASIC from the TITLE PAGE. However, CALL P() alters those contents, so that NEW will then alter the limits of TI BASIC's memory.

Once the partitioned area of memory has been created, NEW will have no further effect on it, nor will TI BASIC programmes write over it. Also, errors in any programme which cause the programme to stop, will not affect the partitioned area either. This means that a set of data put into this partitioned area can be used by one or more BASIC programmes, which can be OLDed into memory, run, and replaced by further programmes, all without disrupting the stored data. This is in direct contrast to the effect of editing programmes, or OLDing programmes, on the contents of any variables, with which you will probably by now be familiar.

OK, you say, I think I follow all that, but what use is this cordoned-off memory? TI BASIC doesn't have PEEK or POKE, so what now?

The answer is:

CALLs G, H, L and S

These are used to create, access, and transfer data to and from peripherals and the partitioned memory.

How can we see CALL P() at work, then? With CALLs A and D it was easy to give examples of them at work, but how do we tell if CALL p() has actually set

aside an area of memory?

The answer is fairly simple. There is a method whereby you can measure approximate memory use in TI BASIC

Come from the title page into TI BASIC. Enter this two-line programme and run it:

```
1 A=A+B
2 GOSUB 1
```

When the computer stops with a MEMORY FULL IN error, type PRINT A and press ENTER. The number printed on the screen is a good approximation to the amount of free memory.

Insert either a PRK or STATS module into the machine, and go through the title page and into TI BASIC again.

This time type CALL P(10000) and press ENTER, followed by NEW and ENTER. Then type in the two-liner again and run it. PRINT the contents of A when the programme stops, and notice the difference: a sizeable chunk of memory has 'gone missing' as described above. Experiment and see what is the largest value accepted by CALL P() - your module manuals should give you some hints.

General Description of PREP

The PREP subprogramme is a subroutine resident in the PERSONAL RECORD KEEPING and STATISTICS command modules. When the command module is plugged in, this routine can be called from a BASIC programme. It is used to define a fixed length data area in the VDP RAM. This area can then be used by the other subprograms to manipulate data files. The CALL statement is used to execute the routine and takes the following form:

CALL P(V)

where V = number of bytes to reserve
(numeric expression)

V is a numeric constant, variable, or expression which specifies the number of bytes to be reserved for the data area. The maximum size for PRK

file is 10048 bytes. However, the disk controller reserves some VDP RAM for disk I/O. When a disk controller is attached to the system, the maximum size for a PRK file drops to 10000 bytes. For a STATS file the maximums are 7440 bytes without the disk controller and 5392 with the disk controller. When using the PREP subprogramme to allocate a data area, the maximum values are as follows:

With no disk controller or
after CALL FILES(0)
(Technically not feasible) 13820

After CALL FILES(1) 12768

After CALL FILES(2) 12250

For each additional file subtract
518
bytes.

If V evaluates to a value greater than the appropriate maximum, a "NOT ENOUGH MEMORY" error will occur.

In order to avoid problems with variables that may have already been allocated, a call to PREP should always be done imperatively, and should be followed by a NEW command to avoid unpredictable results. If PREP is called with a BASIC programme residing in VDP RAM, an "ILLEGAL CALL" error will result.

Once a data area has been allocated with the PREP subprogramme, that area will remain allocated until BASIC is exited through a BYE command or by pressing FCTN = (Shift Q).



M.E.S.

PAUL A. MEADOWS
321 TOMY HILL
LR. SACKVILLE, MS
394E 176
902 665 7671

M.E.S.

TEXAS INSTRUMENTS EQUIPMENT AND PARTS



Instalment Three

CALL H() - The Header Subprogramme

We will begin this discussion of CALL H(), the HEADER subprogramme, with a brief examination in general terms of the function and use of all of the remaining subprogrammes:

- CALL G() - GETPUT
- CALL H() - HEADER
- CALL L() - LOAD
- CALL S() - SAVE

Having created a 'reserved' area in memory with CALL P() last instalment, we can now emulate some of the processes of PRK and STATS. When you create a file using either of the moduls, you first have to define what the layout of each record will be. This layout is then superimposed, if you like, upon the reserved memory (or the reserved memory is re-arranged to conform to this layout) through the use of CALL H(), the HEADER subprogramme.

Headers, in computer terms, tend to contain information about information - and CALL H() here stores the characteristics or specifications of the records. The use of CALL H() is equivalent to designing a form on which data is to be entered, but where each entry is contained within little boxes, one letter or punctuation mark per box:

```

+++++-----+++++
|F|r|e|d| |B|l|o|g|g|i|n|s| | | |
+++++-----+++++

```

You may have noticed that on some commercial circulars addressed to you, your details have been abbreviated - to make them fit into the boxes.

Note that CALL H() isn't used to fill in the form, only to create the layout. Filling it in is the function performed by CALL G(), the GETPUT subprogramme. This subprogramme can also be used to read entries in the file, hence GET (read) PUT (write).

The transfer of the entire file (i.e. bunch of filled-in forms, where

each group of boxes is a FIELD, and each group of fields is a RECORD, and each group of records is a FILE) either from an external storage device into the reserved area, or vice versa, is performed by CALLs L and S. These two subprogrammes transfer data files using the same format as that used for storing BASIC programmes (called PROGRAM FORMAT), so that they are both much faster and more powerful than INPUT and PRINT.

These programme format files can appear as programmes to BASIC's OLD, so it is possible to successfully OLD a PRK or STATS data file, or even to use CALL L() to load a BASIC programme as if it were a PRK or STATS data file. Having OLDed a programme format data file into BASIC you cannot then LIST or RUN it, because the necessary locations in memory, which BASIC refers to when looking for the extent of a programme, have not been updated, and anyway, the structure of the data file would be totally different to that of a programme.

Rarely you may be able to CALL L() a BASIC programme into memory and 'examine' it in detail, but what you will see is largely grabage, and you would need to understand the use of TOKENS in BASIC anyway (which has been covered in earlier issues of TINS NL).

Now to examine CALL H() in more detail.

The best way of describing what CALL H() does is to follow through the opening sequence which either PRK or STATS requires before you can enter data when using those modules. I will use PRK for this example, but STATS differs only in some minor respects so the general principle still applies.

The first thing that happens when you select PRK from the menu is that the PRK title appears. You can either wait or press ENTER and get into the opening sequence. After waiting for the module to set itself up, you are then asked for the date. Contrary to what you will read in the rewrite of TI's own document, you can enter a

number between 1 and 31 for both month AND day, so that either American or European date formats can be used (i.e. month-day-year, or day-month-year). The system won't let you get by without entering a date.

You must also enter a year, and the PRK has been set up to force an entry between 79 and 99, so don't try using the module to keep track of your earnings before this date!

After that, you are asked whether a printer is attached and for its name in one is, and then it asks you if you want to create a new file or load an existing one.

Assuming that you want to create a fresh file, you make your selection, whereupon the module prompts you for the name of the file.

An interesting thing here is that the '[' and ']' keys will return up and down arrows (bear in mind my PRK was written in the days of the 99/4 when alphalock, FCTN, and CTRL were but a twinkle on somebody's drawing board), both of which form valid file name characters. (And yet the FCTN C open single quote (') is not regarded as existing at all. Odd, innit? Not only that, but the underline (_) is redefined as well).

So, we have entered a date and a file name, items which will be used when the file header is written using CALL H().

From this point on, you will be defining the names and characteristics of the fields in the records in the file (puff, puff). Each field is called an ITEM, and there can be up to 15 of them in a PRK record.

For example, one field might be labelled FORENAME, while another might be SURNAME, or AGE, or whatever. A set of such fields - full name and age - would then constitute one record. Entering data for one set of fields would be the same as filling in one form. Put the forms together (or records) and you have a file.

In the General Description of the Header Subprogramme you can see that there are two forms of CALL H() - one for numbers and one for strings:

```
CALL H(R/W,INFO,FLD,V)
CALL H(R/W,INFO,FLD,V$)
```

where:

R stands for Read,
W for Write,

INFO for the Header Item Number (an integer between 1 and 14 - details in the General Description),

FLD for the Field Number - a number between 1 and 15 on the PRK and 1 and 99 on the STATS module-,

and V or V\$ for the variable which is used to receive/provide data from/for the storage area in memory.

The General Description also provides the information on what each of the codes and numbers stands for - for example, in the R/W entry, zero means Write and one means Read - and for the most part there is little point in duplicating that information here.

Note that in some cases the subprogramme automatically updates some of the header items without requiring your intervention, and by and large the only INFO items which would concern you are 1 to 4, 9 to 12.

These cover the file name and the date, and the field names and their attributes, and would be defined at the start of the production of your database.

I must confess that I have had little use for anything other than CALLs A and D, and therefore this section of the Enhanced BASIC discussion is sketchy to say the least. Future articles by Paul Karis, who has made good use of Enhanced BASIC and who has kindly provided the applications articles which he has written, will be appearing along with these installments.

ENHANCED
BASIC

```

100 ! DIRECT SOUND CONTROL
110 ! DEMO PROGRAM
120 ! BY Tim MacEachern
130 !   PO Box 1135
140 !   Dartmouth, NS
150 !   Canada  B2Y 4B8
160 !
170 S=31744 !           ADDRESS OF SOUND CHIP >8400
180 V1=0 !           VOICE 1 FLAG
190 V2=32 !          VOICE 2 FLAG
200 V3=64 !          VOICE 3 FLAG
210 N=96 !           NOISE FLAG
220 C=128 !          COMMAND FLAG
230 F=0 !           FREQUENCY FLAG
240 A=16 !           ATTENUATION FLAG
250 WHITE=0 !        WHITE NOISE FLAG
260 PERIODIC=4 !     PERIODIC NOISE FLAG
270 CALL INIT :: CALL CLEAR
280 ! DEMO—START VOICE ONE
290 PRINT "SET VOICE 1 IN THREE LOADS"
300 CALL LOAD(S,C+V1+A+0)! SET ATTENUATION TO 0
310 CALL LOAD(S,C+V1+F+0)! SET BOTTOM FOUR BITS OF COUNTDOWN RATE TO 0
320 CALL LOAD(S,33)!   SET TOP 6 BITS OF COUNTDOWN RATE
330 GOSUB 820
340 PRINT : "SET VOICE 1 IN A SINGLE LOAD"
350 CALL LOAD(S,C+V1+A+0,0,C+V1+F+0,0,22)
360 GOSUB 820
370 PRINT : "ATTENUATION DEMO"
380 CALL LOAD(S,C+V1+A+6,0,C+V1+F+0,0,56)! START VOICE ONE AS A
    REFERENCE (VERY QUIET)
390 CALL LOAD(S,C+V2+A+15,0,C+V2+F+0,0,48)! TURN V2 OFF BUT PRESET T
    ONE
400 FOR I=1 TO 5
410 FOR ATTEN=15 TO 0 STEP -1
420 CALL LOAD(S,C+V2+A+ATTEN)
430 FOR DELAY=1 TO 40 :: NEXT DELAY
440 NEXT ATTEN
450 NEXT I
460 GOSUB 820
470 PRINT : "COUNTDOWN RATE DEMO"
480 CALL LOAD(S,C+V1+A+1)! SET V1 ATTENUATION
490 FOR RATE=0 TO 1023 STEP 16
500 FOR BOTTOM4BITS=0 TO 15
510 CALL LOAD(S,C+V1+F+BOTTOM4BITS,0,RATE/16)
520 NEXT BOTTOM4BITS
530 NEXT RATE
540 GOSUB 820
550 PRINT : "CALCULATION OF RATE FOR MIDDLE C (FREQUENCY 261.63)"
560 FREQ=261.63
570 RATE=111860.8/FREQ
580 CALL LOAD(S,C+V1+A+0,0,C+V1+F+(RATE AND 15),0,RATE/16)
590 GOSUB 820
600 PRINT : "NOISE CONTROL OPTIONS"
610 PRINT : : "WHITE NOISE TYPE 0"
620 CALL LOAD(S,C+N+A+0,0,C+N+F+WHITE+0)
630 GOSUB 820
640 PRINT : "WHITE NOISE TYPE 1"
650 CALL LOAD(S,C+N+A+0,0,C+N+F+WHITE+1)

```


T I M E L I N E

+++++

Canada's User - Oriented Computer Service

WE cordially invite you to sign up with Canada's Fastest-Growing Computer Service: TimeLine

For Less than Nine cents a Minute (300 bps) you can:

- 1) Play a simulated space war against several other users in a real time environment.
- 2) Computer Ease Shop from you own home for a wide selection of computer peripherals and accessories at incredibly discounted prices. Delivered from our Door to yours!
- 3) Communicate with any other user in North-America for less than a FRACTION of the cost of a long distance call.
- 4) Upload/Download SIG (special interest groups) for FREE software
- 5) Read and Post bulletins/messages on our 36 SIG User Network. Large Commodore, Amiga, Apple, IBM, TRS-80, Ti-994/a, Ectera sections for all users.
- 6) Send and Receive Electronic Mail. You may also store your letters online for future reference.
- 7) Keep up to date with your user's group news! (provided your group has requested a private section)
- 8) Participate in a on-line user's conference in either open or private modes, all in Real time.
- 9) Play a large selection of exciting single-user games... Try your hand at our complex maze solving Adventure, or play a REAL-TIME StarTrek...and more.

TimeLine can be reached by a local call from almost any Canadian City through the DATAPAC Network. All you need is any personal computer/terminal and modem.

Membership is only \$25.00 (one time fee) for your lifetime membership users's manual, and passwords.

-----cut along this line-----

T.U.G.: TimeLine User's Group
680 - 100th Ave, Laval, Quebec, Canada , H7W 3Z6
or
Phone for more info: (514) 681-2280

Name _____	Address _____
City _____	Province _____
Postal code _____	Phone _____
Type of Computer _____	Baud Rate _____
Screen Width _____	Date _____
Would you like to billed by mail _____	MasterCard _____ Expiry Date _____
Credit Card NO# if applicable _____	
Signature _____	
Where did you hear about TUG: _____	



Enclosed please find the \$25.00 membership fee (cheque or M.O.) payable to T.U.G.

Note: 1200 baud usage is available at 10 cents per minute.
DATAPAC is a registered trademark of BELL CANADA ENT.

```

810 GOSUB 810
820 PRINT : "WHITE NOISE TYPE 2"
830 CALL LOAD(S,C+N+A+0,C,C+N+F+WHITE+0)
840 GOSUB 820
850 PRINT : "WHITE NOISE TYPE 3"
860 CALL LOAD(S,C+N+A+0,C,C+N+F+WHITE+0)
870 FOR DELAY=1 TO 500 :: NEXT DELAY
880 PRINT : : "CONTROL NOISE TYPE 3 THROUGH FREQUENCY OF VOICE 3"
890 FOR I=1 TO 10
900 RATE=RND28
910 CALL LOAD(S,C+V3+A+15,O,C+V3+F+(RATE AND 15),O,RATE/16,O,C+N+F+WHITE+3)
920 FOR DELAY=1 TO 300 :: NEXT DELAY
930 NEXT I
940 GOSUB 820
950 STOP
960 ! TURN OFF ALL VOICES
970 FOR I=1 TO 500 :: NEXT I
980 PRINT : : "TURN OFF ALL VOICES"
990 CALL LOAD(S,C+V1+A+15)
1000 CALL LOAD(S,C+V2+A+15)
1010 CALL LOAD(S,C+V3+A+15)
1020 CALL LOAD(S,C+N+A+15)
1030 RETURN

```

CENTIPEDE  MICROSURGEON
 DEFENDER  POLE POSITION
 SHANUS STAR TREK PROTECTOR2 MS-PACMAN
 SPIDERMAN DUCKAROO BANZI HULK ADVENTURE

ADVENTURE





TI99-4A ATARI

WORLD



TRS80 COLOUR

SOFTWARE

7,2 MELROSE AVE, HALIFAX, NS
 B3N 2E2 902 443 1791


SECTOR
SOFTWARE


MORE FINANCIAL DECISION/MANAGEMENT PKG
 EDITOR ASSEMBLER SPEECH SYNTHESIZER


**DOWN
 DOWN
 WALK**


MORE FINANCIAL DECISION/MANAGEMENT PKG
 EDITOR ASSEMBLER SPEECH SYNTHESIZER



"PRINT USING" and your printer

Author unknown

D'loaded from CIS by Terry Atkinson

One of the more obscure statements available with TI Extended BASIC is one called PRINT USING. Even more obscure is the fact that this statement can be used to format variables and constants that will be dumped to your printer. The Extended BASIC manual, on page 150, shows several examples of how PRINT USING can be used to format data for screen display, but nary a word of how to do the same with open files. It can be done, and is much more powerful than you may realize.

Any discussion of PRINT USING will require an understanding of the IMAGE statement, so if you are not familiar with it, you better brush up on it first. The PRINT USING statement uses IMAGE in one of two ways, either with a string expression, or a line number reference. I prefer the latter, as it allows for more flexibility, but since these different methods are explained in the manual, I will limit this to a few simple examples that are not shown in the manual.

```
100 TCOST=19.55
110 IMAGE ##.##
120 OPEN #1:"PIO"
130 PRINT #1,USING 110:TCOST
```

Running this sample program will effectively show how the PRINT USING statement will work with an open file. Of course, there are many other variations of IMAGE that can be used, so experiment with them and watch how it performs when line 130 dumps it to the printer. Shown below are a few more examples for use with an open file.

```
110 IMAGE "##.## ##.##"
130 PRINT #1,USING 110:COST1,COST2
```

This IMAGE statement will allow you to print two (or more) variables at a pre-determined spot on the same line. The length of the string expression in the IMAGE statement can be as long as you wish, up to the limit of an

Extended BASIC line.

```
110 IMAGE "##### ##.##"
130 PRINT #1,USING 110:"TOTAL
COST",TCOST
```

This version shows how you can format the printed line for string data as well as numerical data. A string variable could be used in place of the string constant, as below.

```
105 A$="TOTAL COST"
110 IMAGE "##### ##.##"
130 PRINT #1,USING 110:A$,TCOST
```

It is also possible to place the IMAGE statement inside the PRINT USING statement, as shown below. First, delete line 110.

```
130 PRINT #1,USING "##.##":TCOST
or
130 PRINT #1,USING "#####
##.##":A$,TCOST
```

A few other points to remember include the fact that IMAGE and PRINT USING can be used to round off calculated variables. A single string expression such as "#####.##" will round off and decimal align numbers as small as .01 up to 999999.99, and print the number at any designated location. This function could save many hours of algorithm development for accomplishing the same thing. So, in the long run, the PRINT USING statement is one that any programmer should be very familiar with, and use as much as possible.

Horizontal Lines In BASIC or XBASIC

Author unknown

D'loaded from CIS by Terry Atkinson

If you've ever tried to use a solid line in a program, but the best you could get was a broken dotted line, try this:

By typing the command, CALL CHAR (95,"OOF"), at the beginning of a program you will find that every time you type the underline character, FUNCTION-U, it will appear as a solid line while the program is running.

The number of zeros that you use controls the position of the line. You

may lower the line by adding more zeros (14 zeros are the maximum number that you may add).

The thickness of the solid line can be changed by the number of Fs that you use.

Branching On Two Variables Author unknown

Downloaded from Compuserve by Terry Atkinson.

Many problems have more than one correct solution. For instance, how to branch to one of six locations depending upon a particular combination of TWO variables. For reasons of memory speed efficiency, we needed the absolute minimum number of variables lines of code. The two variables involved were X Y. X could be equal to either 1, 2, or 3, Y could equal either 2 or 17.

The problem: How to combine X Y in such a way as to have the total equal 1, 2, 3, 4, 5, or 6.

One solution to this problem is to temporarily change Y to either 0 or 3, then Y can be added to X to achieve the desired output. This can be done with a series of IF-THEN statements of a "dummy" variable for Y. However, the number of lines extra variables required in this solution proves to be excessive.

Upon re-reading the ON-GOTO information in the User's Reference Guide, one will find when the numeric expression is evaluated, the result is reduced to an integer. By re-reading information about the INTeger function, we will discover that the function rounds the fractional values down. This means that a positive fraction which is less than 1 will yield an integer result of 0 a decimal number of 3, plus a fraction will yield 3. We now have a possible solution.

Dividing Y by 5 will yield 0.4 when Y=2, 3.4 when Y=17. When those numbers are added to X, the result will be 1.4, 2.4, 3.4, 4.4, 5.4, or 6.4. When the computer reduces the result to an integer, the expression will evaluate to 1, 2, 3, 4, 5, or 6, respectively.

Shown here is the algorithm submitted. Three alternate ones are also shown to illustrate the space efficiency savings when using the ON-GOTO numeric expression. Submitted algorithm:

```
100 ON X+(Y/5)GOTO 200, 300, 400,
500, 600, 700
! 200 (Code for X=1, Y=2)
! 300 (Code for X=2 Y=2)
! 400 (Code for X=3 Y=2)
! 500 (Code for X=1 Y=17)
! 600 (Code for X=2 Y=17)
! 700 (Code for X=3 Y=17)
```

Alternate No. 1

```
100 IF Y=17 THEN 120
110 ON X GOTO 200,300,400
120 ON X GOTO 500,600,700
```

Alternate No. 2

```
100 D=0
110 IF Y=2 THEN 130
120 D=3
130 ON X+D GOTO 200,300,400,
500,600,700
```

Alternate No. 3

```
100 IF Y=17 THEN 130
110 IF X=1 THEN 200
120 IF X=2 THEN 300 ELSE 400
130 IF X=1 THEN 500
140 IF X=2 THEN 600 ELSE 700
```


Wico Trackball
A review
by Terry Atkinson

Having used many types of joysticks including Atari (good), Wico Ball (vg) Frostick II (vg), and even the old dual TI joystics (poor), I decided to try yet another product from Wico...the Wico Trackball.

Having used it for a few days, I am now wondering how on earth I have ever done without it! I would rate it at SUPERB, and this is probably an understatement.

The Wico trackball is very easy to use. Just plug the adapter (supplied) into a wall outlet, the other end into the jack, then plug the other line directly into the joystick port. No more fussing with a "converter" cable. The trackball is made for the TI!

For use in all tested programs, this device performs beyond my expectations. For example, with TI-ARTIST, extraordinary control is exhibited. Even for free-drawing circles, because it has a full 360 degree movement! It takes only a few minutes of practice to "get used to" this unique tool. For games, it is also excellent. For example, 4A-FLYER: with the joystick it is very hard to "steady-up" on a specific course/altitude. With the trackball, this is a piece of cake! Heck, I even made it to the second screen on Parsec for the first time with this device! My son did the majority of testing with games, as I am not much of a game player. He reports superb control in all games...in particular TENNIS. He played the computer an "expert" game, and consistently won the match. I think that speaks for itself!

The only drawback is that it cannot be used in "two-player" games. It also cannot be used as joyst2.

I strongly recommend the purchase of the Wico trackball as an additional joystick, and for those considering buying another joystick, I would wholeheartedly recommend the purchase

of the Wico Trackball. They are available from TENEX at a mere \$14.95 (U.S.) each, or just about any TI supplier at various prices.



MYARC Computer Specifications

Equipment

A plug in card to be placed in a TI Expansion box or a Myarc Expansion system. An IBM Keyboard.

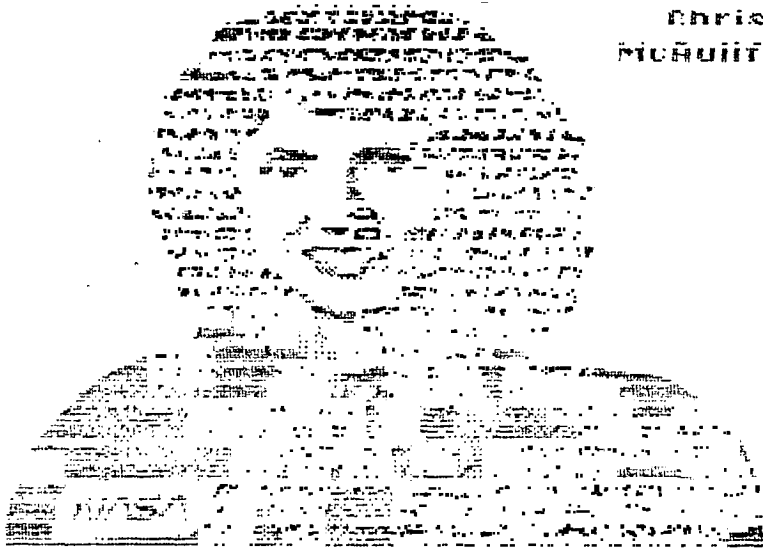
Specs

512K Ram for CPU.
128K Additional Ram for Video. Can add an additional 512K to reach 1 megabyte.
32K Rom.
6-8 times faster than the 99/4A.
Supports a Joystick with a mouse and light pen port.
Supports a video modulator.
Supports an RGB Monitor.
All modes compatible with 99/4A.
Will run 95% of all TI software.

Graphics

26 Lines down.
80 Column Text Mode.
Hi-Res 512x424 definable to 16 colors.
Res 256x424 definable to 256 colors.
Black White monitor mode. 32 shades of gray.
Line drawing commands.
Dot plotting.
Move video memory from one block to another.
Move blocks of colors.
Color Point command.

Christa
McQuilte





The following was taken from Timeline as uploaded by Terry Atkinson (TIBBS). It is written by Dae Wakely of the Chicago Users Group.

It is true! It is here! It works! I saw it! Just as he promised to do, Lou Phillips of Myarc showed up at the Chicago Consumer Electronics Show on Sunday with several working Peripheral Expansion Boxes running his "computer-on-a-card." These were fully functioning printed circuit boards running 80-column and hi-res graphics demos.

I spoke to Mr. Phillips for over an hour at the show, and yet I almost missed him entirely. His small booth was unmarked, and I only spotted it by recognizing him, having seen him at our Chicago TI Faire last November. He was even wearing a badge with someone else's name on it, but denied that he was here incognito. The booth was not shared, it was entirely Myarc's. With Phillips behind the table was John Keown, author of Module Emulator, who is now doing extensive work with Myarc.

The new computer is named GENEVE, but will also be known as the "Model 9640 Family Computer". Phillips stated that Texas Instruments asked him not to use "9900" in the name, but he retained the "9" and added the "640" because that is the amount of RAM which comes with the machine. The following is directly from the one-page information sheet which was handed out:

- o 99/4(A) compatible, runs over 100 existing TI cartridge programs
- o 99/4(A) compatible, runs over 95% of all Assembly language programs utilities
 - o BASIC 3.0
 - o TI-WRITER, now a full 80-columns.
 - o Multiplan, also 80 columns.
 - o FASTER At least 2-3 times
 - o LARGER Standard 640K RAM 2 MEGABYTES Addressable RAM MYARC Memory Card Compatible With Myarc 512K Card, Supplies 1.1 MEGABYTES RAM
 - o IBM TYPE KEYBOARD included
 - o PHONE TYPE CABLE Replaces Old Hex

Bus Cable

- o MOUSE SUPPORT
 - o On the back of the page was a list of more features, including:
 - o Composite Video Output
 - o RGB Output (Note: I was informed this is Analog RGB, with "thousands" of colors available)
 - o 40/80 column display
 - o Mouse Output Port
 - o Joystick port
 - o 128K VDP RAM memory

Phillips stated that you will no longer need the flex cable or even the TI console. The card plugs into the PE Box, and a cable goes from there to the IBM type keyboard. Your other cards will work as usual. When asked about using cartridge software with a machine which has no such port, Keown stated that a copy of his program, Module Emulator, will be included with each machine, and you will be expected to dump whichever cartridges you want. They will then run from disk on the new machine. The only cartridges they have been unable to use are those which call console BASIC routines, such as Personal Record Keeping, Statistics Tax/Investment Record Keeping, and a few others. They are not sure when they will attempt to correct this. Everything else, including the present Extended BASIC, will dump and run on the new machine.

You probably won't need XB anyway, since the machine will come with Myarc's BASIC 3.0, as well as 80 column versions of TI-Writer and Multiplan. Myarc plans at this time to include the keyboard, and the suggested retail for all of the above is \$495.00. When asked if they would sell the machine without a keyboard for less, Phillips quickly added that it was likely they would do so.

Availability of the machine is planned for "about mid-July". Phillips stated he has been told to expect completed components from his suppliers "around the end of June". (Sounds like too tight a schedule to me). The initial production run is for "about 1,000 units".

Phillips stated that he considers the hardware, the card itself, "done". Several beta testers already have the card. He is presently working on the native, or boot-up, DOS. Among other things, he is trying to decide whether to use a TI-like directory system, perhaps with a boot-up menu which can call TI emulaton, or presumably something similar to MS-DOS, where TI emulation must be called in by the user from disk. Either way, once in "TI mode" the machine will function as your TI does now, except with all the new "goodies" added.

At this point I asked Mr. Phillips directly if, in essence, there will ever be another mode to use, namely, an IBM compatible system. He very quickly stated that this is one of their goals and that he expects "in about six months to be at the same stage we are now with this card". It was pointed out to him that this will roughly coincide with the 4th Chicago TI Faire, to which he was immediately invited. He has tentatively accepted our offer to present at that show.

There are still plans to produce a stand-alone new computer, but this seemed, at least to me, somewhat vague. Phillips also hinted that Myarc will probably be producing the equivalent of a Peripheral Expansion box of their own, which would also seem to me to remove the need for a self-contained machine.

Back to IBM compatibility, Phillips did state that with a double-density disk controller (such as Myarc's) there is no reason why the new machine could not read IBM (MS-DOS) formatted disks. The problem is that the TI 9995 processor knows not what to do with 8088 instructions, so IBM programs are out until the compatibility card comes along. But I asked if, for example, word processing files and possibly even saved Multiplan files could be used by either system, and, after thinking about it for a second, Phillips stated he couldn't see too many problems with this. This should give us some immediate "productivity compatibility" at least.

In wrapping up, Phillips noted that the quad density chip upgrade which has been talked about at other shows is now ready, and to contact dealers for info. He also stated that the fine hi-res demos which were running at the CES were written by Chris Faherty of Inscebot, and that copies of the demos would be included with Myarc's release of BASIC 2.1 free of charge.

I would like to conclude by noting that there have been those who have publicly doubted the intentions of Myarc with regards to this machine. After waiting over 2.5 years for a replacement/upgrade machine, perhaps such skepticism is understandable among TI owners. I will say that Lou Phillips comes across as a sincere, straight forward guy. For those who, also quite understandably doubt appearances, I will tell you both that the machine DOES exist, and whatever else he is doing, Lou Phillips just spent one HECK of a lot of money for a table at this CES to show it.

I apologize for any editorializing I may have done in this article, and for anything I may have omitted. I believe I have outlined everything of importance I spoke about with Phillips, but if you have questions I haven't anticipated, by all means contact Myarc!

Incidentally, I inquired about the origin of the name. Keown jumped in and stated that it was his idea. It seems that a few days before the CES, while they were working together, Keown told Phillips that he felt there should be a name for the new machine instead of just a number, "the 9640". As they were heading down the staircase from Phillips' office there was a framed print on the wall. The name at the bottom was "Geneve", and when Keown suggested this, Phillips agreed to it.

Disk Summary
86/6 TINS/ARTS
Programme Listings

FILENAME	SIZE	TYPE	P	DISHNAME
*B>ART+	8	PROGRAM	Y	TINS-31A
*B>ART/DOC	17	DIS/VAR	80 Y	TINS-31A
*B>ARTCC+	8	PROGRAM	Y	TINS-31A
*XB>ART+	5	PROGRAM	Y	TINS-31A
*XB>ARTCC+	11	PROGRAM	Y	TINS-31A
*XB>ARTDOC	32	DIS/VAR	80 Y	TINS-31A
AMAZON_C	25	PROGRAM	Y	TINS-31A
AMAZON_P	25	PROGRAM	Y	TINS-31A
ATOMS_C	25	PROGRAM	Y	TINS-39/1A
ATOMS_P	25	PROGRAM	Y	TINS-39/1A
BARNEY_P	25	PROGRAM	Y	TINS-39/1A
BARS_P	25	PROGRAM	Y	TINS-39/1A
BEETHOVEN	21	PROGRAM		TINS-27/1A
BEETHOVEN9	13	PROGRAM		TINS-27/1A
BIO-RHYTHM	28	PROGRAM		TINS-27/1A
BLOOM_P	25	PROGRAM	Y	TINS-39/1A
BULL_P	25	PROGRAM	Y	TINS-39/1A
BUNNY_P	25	PROGRAM	Y	TINS-39/1A
CABLE_C	25	PROGRAM	Y	TINS-39/1A
CABLE_P	25	PROGRAM	Y	TINS-39/1A
CALL/KEY	8	PROGRAM		TINS-27/1A
CHRISTA_P	25	PROGRAM	Y	TINS-31A
COLOR	25	PROGRAM		TINS-27/1A
COLOR	25	PROGRAM		TINS-27/2A
COM-ART	4	PROGRAM		TINS-27/1A
DAY/WEEK	14	PROGRAM		TINS-27/1A
DAYS/BETWN	5	PROGRAM		TINS-27/1A
DENEUEVERLE	27	DIS/FIX	128 Y	TINS-31A
DENEUEVE_P	25	PROGRAM	Y	TINS-31A
DESIGN	7	PROGRAM		TINS-27/2A
DR-Z	9	PROGRAM		TINS-27/1A
DRAGON_C	25	PROGRAM	Y	TINS-39/1A
DRAGON_P	25	PROGRAM	Y	TINS-39/1A
EASLE_C	25	PROGRAM	Y	TINS-39/1A
EASLE_P	25	PROGRAM	Y	TINS-39/1A
EUROPE_P	25	PROGRAM	Y	TINS-39/1A
FLITEPLAN	13	PROGRAM		TINS-27/2A
HAUNTEDHSE	32	PROGRAM		TINS-27/1A
INTRFACE_C	25	PROGRAM	Y	TINS-31A
INTRFACE_P	25	PROGRAM	Y	TINS-31A
JINGLEBELL	27	PROGRAM		TINS-27/1A
KWIKDRAW	27	PROGRAM		TINS-27/1A
MAZE/MAKER	5	PROGRAM		TINS-27/1A
MSG/GRAPH	40	PROGRAM		TINS-27/1A
MUPPETMUSC	43	PROGRAM		TINS-27/1A
MUSIC	8	PROGRAM		TINS-27/1A
MUSICMAKER	19	PROGRAM		TINS-27/2A
OLDLANGSYN	30	PROGRAM		TINS-27/1A
PIANO	30	PROGRAM		TINS-27/2A
PLAY/ORGAN	38	PROGRAM		TINS-27/2A
PLOT3D	8	PROGRAM		TINS-27/1A
PUPPY-TOWN	34	PROGRAM		TINS-27/2A

RLE/DOC	10	DIS/VAR	80 Y	TINS-31A
RLE1	38	DIS/FIX	80 Y	TINS-31A
RLE2	38	DIS/FIX	80 Y	TINS-31A
R. BOOBOGY	17	PROGRAM		TINS-27/1A
SILENTNITE	14	PROGRAM		TINS-27/2A
SNOCFNWAS	28	PROGRAM		TINS-27/1A
SPRITES1	4	PROGRAM		TINS-27/1A
SPRITES2	8	PROGRAM		TINS-27/1A
STARTREK	30	PROGRAM		TINS-27/1A
TI*SOUND	6	PROGRAM		TINS-27/2A
TITLER	21	PROGRAM		TINS-27/2A
TRUE/FALSE	7	PROGRAM		TINS-27/2A
VENUS/NITE	26	PROGRAM		TINS-27/2A
W/BOOGIE	26	PROGRAM		TINS-27/2A
XMAS*CAROL	16	PROGRAM		TINS-27/2A

TINS Library Note

In an attempt to improve the use and understanding of the TINS LIBRARY I have reformatted the files and listings. The older section of the LIBRARY will still have the names TINS-21/1A for example, but I will catalogue the listings separately. That is the Arts section will be catalogued on its own and the Bussines, Education, Forth, Games, Home Computer Magazine, Logo, Miscelanious, and Utilities. I hope this will allow all to more easilly understand the listing they are looking for and intending to use.

To futher help each section will have the disk summary in it, the program listings, and a per disk catalogue in it. I hope that this will allow faster and easier access to the programs you want. For example you can read a file name you like in the program listing and see the disk it is located on, review that disk and get a copy of programs that you expected to see.

We are also fortunat to have a member who has been writting up reviews on each disk he gets which will be filled with the listings. These give a short description of the programs on that disk.

I hope this helps. I will print one section a month in the club paper. The new section of the Library will take a bit longer to layout but it will be handled in a simalar method.

Ron Weagle
TINS Library



LET ME
CALL YOU
GENEVE
MY LITTLE
MILKADEE
LOVE