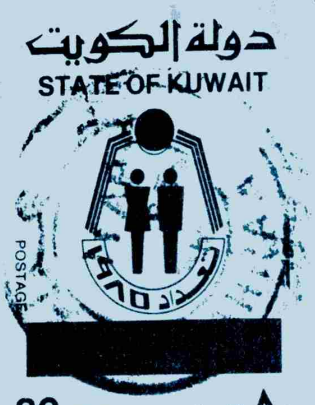
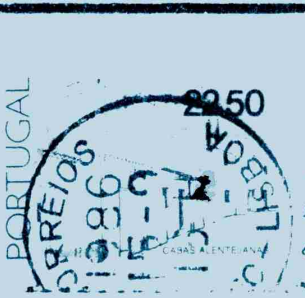
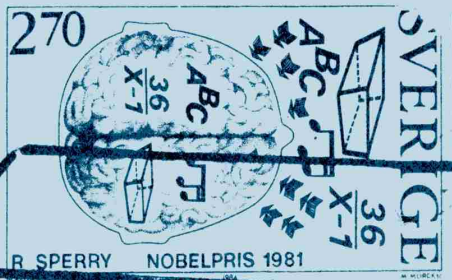
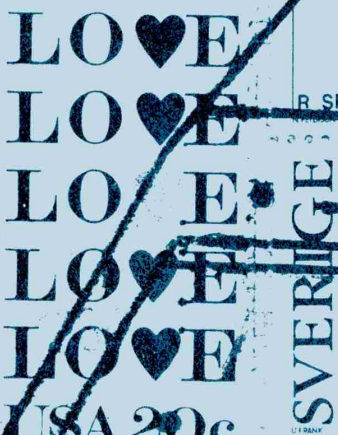
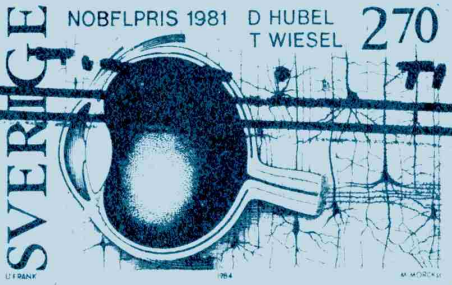
TI*MES

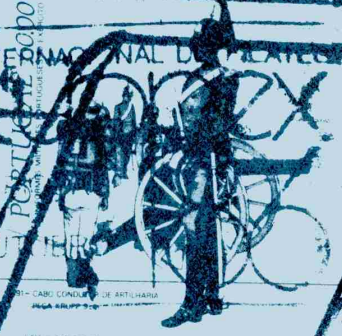


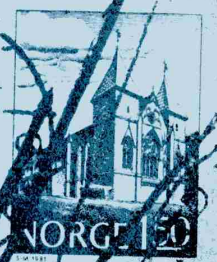


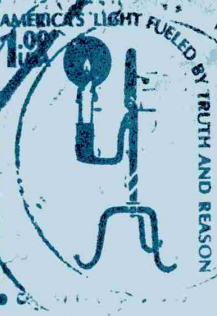












FRED McCUBBIN On the Wallaby Track 1896 Art Gallery NSW

This page is blank

TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES TI*MES
WINTER ISSUE NUMBER FIFTEEN

40,Barrhill,Patcham,BRIGHTON,East Sussex,BN18UF.Tel:0273 503968(evenings)

Another Great Year for TI99/4a

Already a number of you have asked about the recent changes in membership renewals. If you do not already know these start and end in May of each year. There is another reason and that is to enable a smooth handover to a brand new Group Committee which will be formed within the next few months.

This means that TI99/4a Exchange will cease to exist. However SUBJECT TO YOUR CONTINUOUS SUPPORT The UK TI99/4a Users Group will remain. It will continue to be a non profit making group interested only in the survival of the TI99/4a and its expansion. TI*MES magazine will stay in circulation but again subject to an editor being appointed. There is no shortage of material which is produced daily from all over the World. I have no doubts that our orphan TI99/4a Home computer remains with us for many more years. But as regards a BRITISH TI USER GROUP it is now down to you to ensure that we continue to fly the flag overseas.

Now after four years and fifteen issues of TI*MES I am not sad but very excited about the prospects of a NATIONAL UK TI USER GROUP. With a strong team on a committee there is so much more than I ever hoped to achieve for the British 99er.

You will find an application / comment form to complete. THIS IS MOST IMPORTANT THAT YOU READ CAREFULLY AND ACTION TODAY. Your attention will be appreciated as of course your offer of help. DO NOT BE SHY say what you want now.

The Computer Jamboree held in November at Mackie Hall has now passed. We were very proud to show off the TI99/4a against machines costing thousands of pounds. Only the TI99/4a demonstrated speech, colour, music and singing!!!. A number of loyal 99ers came from afar to visit. We were able to discuss the recent developments such as the latest Funnelweb. We also had comment from those who are without any expansion. How surprised and delighted we were to learn that those who joined us recently have computers which have changed hands a number of times. A warm welcome is extended to you. Yes I am aware that those new owners of TI99/4a want more BASIC content of this magazine. I can understand total frustration as TI*MES becomes more sophisicated in editorial. I have to admit that those folks are right to complain. However as this Group has been around along time we can only continue to move up and not stand still, but do write and tell us what you want from this Group. I also wish to make a repeated request on behalf of Stephen Shaw our greatest supporter of TI*MES, what do you want to see in RAMBLES ? Your reponse is required.

This leaves me to thank most sincerely Stephen, the many UK 99ers and contributors of TI*MES, including all WORLDWIDE TI99/4a Users Groups, who over the years continue to enrich the use of our machine. Audrey and myself wish all a very Enjoyable and Bright 1987.

Happy 99ing,

Clive Scally.

INPUT STATEMENT WITH FUNCTION BREAKOUT

Anyone who has used the Personal Record Keeping module or the Disk Manager or any number of TI programs will be familar with the use of the Function Keys such as BEGIN, BACK, AID etc as a means of routing to other parts of a program even when the program is waiting on an input statement. This can be particularly useful to:

-Restart the program (eg BEGIN)
-Correct earlier inputs (eg REDO)
-Return to previous routine (eg BACK)
-Forward to next routine (eg PROCEED)
-Get a help message (eg AID)

There are any number of uses for such function keys, but you will no doubt have been disappointed to find that in both Basic and Extended Basic, neither the INPUT or ACCEPT functions allow such function key breakout. I have often wondered whether TI intended the ExBas ACCEPT to have such a feature, since I have noticed that, if you use the VALIDATE(UALPHA) qualifier, the only non-upper case keys that don't elicit a "honk" are the function keys.

So are there any ways we can get such a function using Basic? The answer is yes, and there are several methods that can be used.

ENHANCED BASIC

Many of you will be familiar with the so-called Enhanced Basic, which the name given to the Basic resident in the PRK and Stats modules. This features a number of extra CALLs which enable many clever things to be done, including storing data in PROGRAM format. However the CALL A statement is not only similar to the ExBas ACCEPT statement, but it also allows function key breakout. The format is:

CALL A(R,C,L,F,A$) for string variable input

and

CALL A(R,C,L,F,A,MN,MX) for numeric variable input

where

R=ROW C=COLUMN L=FIELD LENGTH F=FUNCTION VARIABLE
A$=ACCEPTED STRING VARIABLE A=ACCEPTED NUMERIC VARIABLE
MN=MINIMUM VALUE MX=MAXIMUM VALUE

This is not the place for a full tutorial on Enhanced Basic. Suffice to spell out how the function key breakout works. If one of the function keys is pressed while the program is waiting for an input, control passes to the next statement with the variable F set as follows:

AID	3
REDO	4
PROC'D	5
BEGIN	6
BACK	7
CLEAR	2 (also ENTER when no data present)
ENTER	1 when data present

By using a ON F GOTO ... statement following the CALL A, you can reroute your program at will.

Those of you that have Oaktree's Display Enhancement Package will not only have the advantage of being able to use 40 columns and many types of input and output statement, but also the INPUT and ACCEPT statements support function key breakout once this is enabled with the CTLON statement. I won't go into this in detail as my main reason for writing this article is to consider how these features can be incorporated in normal Basic.

EXTENDED BASIC USING CALL KEY

I have developed an ExBas subprogram which incorporates function key breakout. To get round the lack of this function in INPUT and ACCEPT, I have written an input routine using the CALL KEY statement which thus allows all key presses to be captured. I apologise to those without ExBas, but I decided to use this version of Basic for several reasons:

1 Greater speed, using multiple statement lines, so that key presses are not missed

2 A sprite is used for the cursor, thus minimising problems as the cursor passes over existing text.

3 The MIN and MAX functions simplify the control of the input field.

Even ExBas cannot however give the speed that this sort of statement really requires. The response to Enter is very slow because of the repeated use of CALL GCHAR and CALL HCHAR. Similarly the Insert and Delete routines work at a snails pace and I have incorporated a CALL SOUND so that their completion is signalled. In the following demonstration program, you can see how the CALL IPUT function works. The subprogram itself is on lines 310-650, the rest demonstrates its use. Lines 100-130 are necessary for fully featured use.

NOTES ON PROGRAM

Lines 100/110 These lines can only be used if you have the 32k memory attached. They cause the QUIT key to be disabled and therefore capable of being detected along with the other FCTN keys.

Line 120 This disables the CLEAR key from halting the program and therefore allows it to be detected along with the other FCTN keys.

Line 130 Defines the sprite cursor character.

Line 150 The subprogram is called. The format is:

CALL IPUT(R,C,L,F,A$) where

R=ROW C=COLUMN (NB GRAPHICS COLUMNS 1-32)
L=FIELD LENGTH F=ASCII VALUE OF FCTN KEY PRESSED
A$=STRING ACCEPTED ON USE OF ENTER, DOWN ARROW OR UP ARROW KEYS.

Line 160 The use of the POS function allows the GOTO list to have outlets in the order:

(ERROR), CLEAR, BEGIN, PROC'D, AID, REDO, BACK, QUIT, ENTER DOWN ARROW, UP ARROW.

Line 170-240 Display the FCTN key used.

Line 260 This allows you to compare operation with a normal ACCEPT statement.

Line 280 Allows CLEAR key to either halt program or continue.

Line 290/300 Allows QUIT key to exit to Master Screen or continue. (32K only)

Line 310 Subprogram starts here.

Lines 380-410 This is the main loop in the routine as characters are input.

I hope the REMarks make the bulk of the routine understandable.

THE PROGRAM

```
100 CALL INIT
110 CALL LOAD(-31806,16)!DISABLES QUIT KEY
120 ON BREAK NEXT !DISABLE CLEAR KEY
130 CALL CHAR(143,"FFFF")!DEFINE CURSOR CHAR
140 CALL CLEAR
150 CALL IPUT(3,3,30,F,A$)
160 ON POS("BNLAFOEMJK",CHR$(F+64),1)+1 GOTO 150,280,170,180,190,200,210,290,240
,220,230
170 DISPLAY AT(24,12):"BEGIN" :: GOTO 260
180 DISPLAY AT(24,12):"PROCEED" :: GOTO 260
190 DISPLAY AT(24,12):"AID" :: GOTO 260
200 DISPLAY AT(24,12):"REDO" :: GOTO 260
210 DISPLAY AT(24,12):"BACK" :: GOTO 260
220 DISPLAY AT(24,12):"DOWN ARROW" :: GOTO 250
230 DISPLAY AT(24,12):"UP ARROW" :: GOTO 250
240 DISPLAY AT(24,12):"ENTER"
250 DISPLAY AT(12,1):A$
260 ACCEPT AT(14,1):B$
270 GOTO 150
280 DISPLAY AT(24,12):"CLEAR  STOP? N" :: ACCEPT AT(24,25)VALIDATE("YN")SIZE(-1)
:B$ :: IF B$="Y" THEN STOP ELSE 260
290 DISPLAY AT(24,12):"QUIT? N" :: ACCEPT AT(24,18)VALIDATE("YN")SIZE(-1):B$ ::
IF B$="Y" THEN 300 ELSE 260
300 CALL LOAD(-32730,32)!MASTER TITLE SCREEN
310 SUB IPUT(R,C,L,F,A$)
320 R=MIN(ABS(R),24):: C=MIN(ABS(C),32)!ENSURE R,C IN LIMITS
330 A$="" :: SC=C :: FC=MIN(32,C+ABS(L)-1)!SETUP START & FINISH COLUMNS
340 DR=R*8+1 !SET DOT ROW
350 IF L>0 THEN CALL HCHAR(R,SC,32,ABS(L))!ERASE FIELD IF REQ'D
360 CALL SPRITE(#1,143,13,1,256)!CREATE SPRITE CURSOR
370 CALL SOUND(100,400,5)!INPUT PROMPT
380 DC=C*8-7 !CALCULATE DOT COLUMN
390 CALL LOCATE(#1,DR,DC)!MOVE CURSOR
400 CALL KEY(5,F,V):: IF V=1 THEN 410 :: IF V=0 THEN P=0 :: GOTO 400 ELSE IF V A
ND P THEN 410 ELSE P=-1 :: GOTO 430
410 IF F>15 THEN CALL HCHAR(R,C,F):: C=MIN(FC,C+1):: GOTO 380 !DISPLAY CHAR
420 ON F GOTO 450,450,530,610,450,450,480,570,590,500,500,450,500,450,450
430 FOR Z=1 TO 100 :: NEXT Z :: GOTO 400 !DELAY FOR REPEAT KEY
440 REM EXIT ROUTINE
450 CALL DELSPRITE(#1)
460 SUBEXIT
470 REM ERASE ROUTINE
480 CALL HCHAR(R,SC,32,ABS(L)):: C=SC :: GOTO 380
490 REM ENTER ROUTINE
500 FOR CL=SC TO FC :: CALL GCHAR(R,CL,K):: A$=A$&CHR$(K):: NEXT CL
510 GOTO 450
520 REM DELETE ROUTINE
530 FOR CL=C+1 TO FC :: CALL GCHAR(R,CL,K):: CALL HCHAR(R,CL-1,K):: NEXT CL
540 CALL HCHAR(R,FC,32):: CALL SOUND(100,1000,5)
550 GOTO 400
560 REM LEFT ARROW ROUTINE
570 C=MAX(SC,C-1):: GOTO 380
```

```
580 REM RIGHT ARROW ROUTINE
590 C=MIN(FC,C+1):: GOTO 380
600 REM INSERT ROUTINE
610 CALL KEY(5,F,V):: IF V=1 THEN 610
620 IF F<16 THEN ON F GOTO 450,450,530,610,450,450,480,570,590,500,500,450,500,4
50,450
630 FOR CL=FC-1 TO C STEP -1 :: CALL GCHAR(R,CL,K):: CALL HCHAR(R,CL+1,K):: NEXT
 CL
640 CALL HCHAR(R,C,F):: C=MIN(FC,C+1):: DC=C*8-7 :: CALL LOCATE(#1,DR,DC):: CALL
 SOUND(100,1000,5):: GOTO 610
650 SUBEND
```

THE FINAL SOLUTION?

Given the lack of speed of the above routine, what is clearly needed is an Assembly Language subprogram that does the same function ie CALL LINK("INPUT"....). I have seen several packages that contain such input lines, but is there anyone out there willing to write the source code for a stand-alone utility, complete with flashing cursor?

In the meantime, I hope the above program is of use to those with ExBas.

Peter Walker

MAILBOX

JANUARY 1987 PCW

The long-awaited 'keyword search' facility on Prestel is due to have its public launch shortly. One of the biggest criticisms of Prestel is the difficulty of finding your way around the system; it can also be a problem to find your way back to a particular page or menu that you want to refer to. Keyword search and an associated facility called 'pagemarker' should make life a lot easier for Prestel users.

As most of you will know, Prestel information is contained in pages, each page having its own number. The quickest way to any page has been (and still is) to go directly to it by entering its number. This is fine for regular users of a particular service who get to know the page(s) they want to use. Even so, the numbers can be quite long and difficult to remember. For the casual and irregular user, finding the page containing the information you want can be a real headache. (If you have never used Prestel, I should tell you that the system is built around a series of menus, each with a number — approximately 10 — of choices. As a rule each choice leads to a submenu, and so on, until you come to the information you are looking for.)

Such a menu system is all very well until you realise that Prestel has a third of a million pages: trying to index these back to a single main index is no easy task. The problem is further compounded by the fact that there are often a number of organisations, called Information Providers (IPs), providing information on a particular topic. Each IP has its own block of pages and these blocks are not necessarily in the same part of the system, so that adjacent choices on a menu could lead to totally different page numbers. The final straw is that, for the most part, the IPs produce their own indexes, so it is very difficult to ensure consistency.

There is a subject index (starting on page 199), and a magazine containing a directory is published four times a year. In spite of all this, many people find it difficult to track down the things they are interested in. It can also be frustrating if, when looking for a particular item, you notice something you want to follow up later on an intermediate menu. Finding your way back can be very difficult. The hardened Prestel user has a pencil and paper at hand to jot down interesting page numbers so that he can refer to them in his own time.

Keywords

All this will now change. Keyword searching will allow you to type *subject# and you will find yourself at a menu page for the subject you have chosen — as long as it is one of the subjects contained in the subject index. I had a sneak preview of the system back in October and was very impressed.

The keyword search facility is a vast improvement, but it isn't perfect. You are limited to words in the subject index, so you can't find all the pages containing the word 'Amstrad', for example. Nor will it accept part of a word; you have to type it in full (and spell it correctly) for the procedure to work. Nevertheless, this facility is a great leap forward. It is much easier to remember *YORK# than *22063539#; or to find the British Rail pages by entering *TRAINS# or *BR# rather than having to search from the main menu.

Neither are you limited to one-word subjects. For example, if you were looking for somewhere to stay in France, you could specify:
*ACCOMMODATION FRANCE#
or
*SELF-CATERING ACCOMMODATION#.

You can also repeat the subject by entering */, which allows you to add words to your previous choice. In this way you could set the keyword to *SPAIN# and then search for *SPAIN AIR TRAVEL#, *SPAIN ACCOMMODATION# and *SPAIN CAR HIRE# without having to type Spain several times. There is a special keyword, *LAST#, which redisplays the last search string and allows you to edit it, and a *HELP# feature to explain how the system works.

Page marking

Pagemarker is another part of the new package on Prestel. This allows you to allocate a label to a particular page and then go back to that page by entering the label you used. For example, if you reach a menu page that you think you will want to return to, you type *SAVE FRED# (where 'FRED' can be any word of your choice). When you want to go back to that page, just enter *FETCH FRED# and you go straight there. You can mark up to five pages in this way, but the marks are lost when you log off the system. The command words can be abbreviated to their first letter.

BT hopes that these developments will make Prestel easier to use, leading to increased usage, which will in turn encourage more IPs to provide more and better information.

Wide choice

There is something to suit everyone It's all there: train times, weather forecasts, traffic problems, sports news, film reviews, Government information (want to know the name of your MP? It's there), teleshopping services, small ads, gardening and much more besides. In the small start-up directory sent to all new subscribers there are around 1500 entries covering subjects from Activity holidays to Zimbabwe.

If you want to have a look round the system and already have access to a Prestel terminal or a micro that can behave like one, you can use the demonstration account number of 44444444 and password 4444. If you don't know the number to dial, directory enquiries should be able to help. If you don't have Prestel graphics, but have a modem and ordinary terminal software, then you can call (01) 680 8245 (300-baud only). Alternatively, if you have an account, you can use PSS — A23411002002018 is the address for Prestel without graphics (A23421920102517 for normal Prestel).

Prestel statistics

Details	1986	1985	Change
Terminals attached to Prestel	67,000	43,000	+56%
Proportion in business	36,800 (55%)	27,000 (63%)	+36%
Proportion in homes	30,100 (45%)	24,000 (37%)	+40%
Frames available for access	291,600	321,000	-9%
Frame accesses per week	8.2 million	4.2 million	+95%
Messages sent per week	122,300	47,700	+156%

Thanks to VNU Publishing and Peter Tootill for this extract of the PRESTEL feature"Hey,presto!" The full text will be found in Vol10 No1 Personal Computer World.

Peter Tootill

PCW JANUARY 1987

KRACKERSNACKS

by Tom Freeman

1. GRAM PACKER

Let me state this right at the outset - this new program by J. Peter Hoddie is what I've been waiting for ever since I got my GRAM KRACKER one year ago. The fine programs that came with the Gram Kracker, and the utility disk that was released recently allowed us to move TI-Writer and/or Editor/Assembler to the Grams that we wished, or to combine them, but this still didn't take full advantage of the copious amounts of GRAM available. Now with Peter's help we can finally do it!

First a brief explanation of what GRAM PACKER can do. Once the main program "GP" is loaded (by Load Module of the GK, option #5 of E/A, or #3 of TIM) there are three main "branches" depending on what it is you want to do. First of all you can load a complete E/A #5 type program into the GRAM of your choice. Along with this clone "GP" also inserts a short GPL loader, and information for the main TI menu screen. If you then choose this item from the main menu, the loader transfers the program back out to CPU where it belongs, and then gives control to the transferred program. This takes up quite a bit of the GRAM (one full GRAM for each 33 sector segment, and there is a chance that this won't be quite enough) but boy does it start up fast! DM1000 takes up two GRAMS, FAST-TERM one, for instance.

Your second choice is to insert a GPL loader that merely takes the program off the device you have specified. What actually happens is that a loader equivalent to the E/A #5 loader, is transferred to CPU, and this then takes over the job of accessing the program on disk. There is no advantage of speed here, unless you have a RAMdisk, as I do, but you save the time of having to type in the program name, and you don't even need to have E/A in place. And it IS nice to see these on the menu.

For both of these 2 choices, you can also assign subprogram names, and DSR names so that from Basic or Extended Basic (assuming they don't interfere with your programs in terms of GRAM locations) you can CALL xxx or OLD xxx to access the programs. This is a nice feature.

The final "branch" is an equivalent loader for GRAMS themselves, essentially replacing that part of the Gram Kracker (so you don't have to go back to Loader On). As an example, when I use SBUG6, my XBasic is destroyed, so I have an item on the menu to reload the appropriate RAMbank for XBasic, and reconstitute it. There are several additional files on the disk that substitute for the E/A #5 loader, or the GK module loader, in a more general way, i.e. when the program runs you must type in the name of the file to be loaded, but I found these less useful, as my E/A is always in place, and of course the GK loader is there if I need it. There are also some files for use with Mini-Memory, and a sample source file for GPL utilities equivalent to the ASL utilities.

Now to be a little more specific as to how the program works. First you load the file GP as stated above. That part is easy! As soon as the program runs you are asked which GRAMS to begin and end with, and then you are off and running. Next you are asked to input the filename, e.g. DSK1.UTIL1, if that is the name of the program you want to load, and the device where it is PRESENTLY located. Then you are requested to assign a name to appear on the menu - this is mandatory. Next you are given the choices to choose subprogram or DSR names, if you wish (they are optional) and then finally to do the "pack". GP then goes to work and puts all of this into the assigned GRAM. It keeps track of the amount of GRAM used, so that you can do more if there is room, and in fact you are asked if you wish to do another. If you do, the whole process is repeated. Finally when you indicate you are finished, you may save the GRAM to disk (equivalent to Save Module on the GK, so you haven't lost anything if you don't do it now) and then the program quits.

If however, instead of inserting the whole program into GRAM, you only wish the loader that will bring it off your chosen device, the process is slightly different. The instruction manual was unfortunately rather murky here, and necessitated a "HAALP" call from me to Peter. There was rather a lot of explanation of how the appropriate files work, and little on how to use them. HERE is what you must do. The "filename" to load is DSK1.EA5;S;L or DSK1.EA5;S;H The first loads into low memory, the second into the top of high memory. You should choose the one that is NOT overwritten by the program file you wish loaded (see my article in the last issue of TopIcs to figure out where your program loads). Files that load at >A000 should work with either loader, although I discovered that, for reasons I still don't understand, PTERM which does load at >A000, would not work with the low memory loader). You then provide the same information for menu name, subprogram name, and DSR name, as before. However when the file is loaded there is a flag that then makes the program prompt you for "additional filename". This is where you insert the information for which program is to be loaded - you must make a permanent choice for device location, e.g. DSK1., or in my case RD., so that you might type in DSK1.UTIL1 or whatever you want. I suggest that you make a utility

disk with all the programs that you will have loaded from the menu, and keep it in the appropriate drive while you are working, except when you need another disk there. Of course, it does not have to be in drive #1. Be careful that your utility disk IS there when you choose that item from the menu - this version of GP has no error checking, and the computer will lock up if there is an error.

The method to load modules is the same except the loader program is called GK;S It will provide you with the same prompt for an "additional filename". These module and program loaders are very short, so that there is room for lots of them in one GRAM. You may therefore very well run out of room on your menu screen - see the next article for various alterations that may or must be made to your operating system to allow for more than nine items on the menu. I have one additional quibble with this FINE program. If you make a single mistake in setting up a series of loaders for the menu, the whole GP will crash, and you must start over. Similarly, if it turns out that one of your loaders doesn't work, as I found out with PTERM, you must also start over.

The GRAM PACKER is an incredible value at $10.00 (available from the Club). Final grade:

PERFORMANCE:	A
EASE OF USE:	A-
DOCUMENTATION:	B
VALUE:	A+
FINAL GRADE:	A

2. OPERATING SYSTEM MODS

Several modifications have to be made to your operating system in GRAM 0 in order to make full use of the GRAM PACKER. You will be using your Gram Kracker Editor to accomplish these (option #5 from the GK main menu). Rather than describe all the keystrokes each time, I will remind you of the general method here. First of all, when you get to the editor screen, press FCTN 1 once to get to GRAM memory. Now when you are instructred to search for a string, press FCTN 5 for search. The cursor will be on the "start" address. Accept the default of 0000 if it is there, or type it in, then press enter to get to "finish" and type 2000. Now press FCTN 9 and type in your search string, remembering FCTN = to get to hex if that is what you are searching for (in general it will be). Back up the cursor one space to get it over the last character in the search string then press enter. If the string is not found, the edit field will not change - if it is found, the address in the upper left hand corner will reflect the location of the first byte of the string found. Now press FCTN 5 again to get out of SEARCH, then FCTN 9 to edit, and type in the appropriate changes. You will need WP off in order to type in the changes - remember to turn it back on when you are finished typing.

The following set of changes need to be made only if you will wind up with more than 9 items on your main menu. There would be two problems if the changes were not made: 1) you wouldn't see any after 9 because of the double spacing! and 2) even if you could the key presses would be : ; < = > etc. some of which would actually be two keys (SHIFT and key). We will therefore enable single spacing on the main menu (thanks to Craig Miller in The Smart Programmer for this information) and change the sequence of key presses from numbers beginning with 1 to letters beginning with A.

First, to change to double spacing: Search for (hex) A3 52 00 3A. In many consoles this will be at 02E0. Change the 3A to 1A. Next comes a problem of another routine using temporary storage where we will need it (not actually involved with the double spacing, but needed if there ARE more than 9 items for the menu). Leaving the start and finish addresses the same, get back to SEARCH by pressing FCTN 5, FCTN 9 and type in 00 02 28 60 for the search string. You should find it at about 0380. FCTN 5 to get back to the memory window. The top line should read:

00 02 28 60 00 D6 28 AA 43 95 D2 29

change the 3rd, 7th, and 12th bytes:

00 02 40 60 00 D6 40 AA 43 95 D2 41

You should also insert the small capital character set into the TITLE SCREEN Characters using the NEWCHARS program on the original GK utility disk, otherwise the characters will touch each other top to bottom and be almost impossible to read. Note that you can only have 16 items on the menu if you are preserving TI Basic in GRAMS 1-2 because the start address is destroyed by the 17th item. I believe it is possible to use the 17th if you are using GRAMS 1-2 for other purposes, such as all of these programs!

Now to change the key presses to letters - this is simpler. First change your start address back to 0000, then search for BE 58 30. You should find it at about 0275. Change the 30 to 40. Next search for A6 75 31 (should be at 02FC) and change the 31 to 41. You will now see letters instead of numbers on the main menu.

I found another problem with many programs: they do no bother to change the keyboard unit to be scanned, assuming it to be 5, since that is where the E/A module is when option #5 is chosen. The problem is that the operating system is using keyboard unit 3 at the time the menu is set up (for this reason you can use lower case letters for the key press on the menu - they will be converted to upper case). Here is a simple fix: 12 bytes

past the 41 you just typed in you should see 06 03 Cx A4, where the x is probably a E. Change the first three to 05 18 00. Now FCTN 9 to get out of memory window, move the cursor to the address after the g in the upper left hand corner, and type in 1800. Now FCTN 9 again, press enter to "home" the cursor, and type in the following: 06 03 Cx BE 80 C6 02 05 03 0y where x is the same as you just found above, and y is 3 higher (in hex) than the address where you found the 06 03 (if that was 030A as it was in my console, then y would be D). This changes the keyboard unit to 5.

For those of you using SBUG6, as I do often, and who wish to use it from the main menu, you may have found that the small character set is not loaded, which is a PAIN! It's OK if you have loaded it from E/A #5. Here is a fix: it incorporates MG's GPLLNK inserted directly into memory and then a simple BLWP @GPLLNK DATA >004A and then return to the beginning of the actual program. You will need a sector editor for this. First find the FDR of the file (catalog sector). Change byte 16(>10) from 92 to EA. Second find the first actual data sector of the file. Change byte 3 from 92 to EA and bytes 24-25(>18-19) from 62 84 to 7D D2. Finally go to the LAST sector of the file (there are 30 data sectors) and starting at byte 146(>92) carefully type in the following over whatever is there:

```
7D 7E 7D 9E 7D C2 17 6C 00 50 00 00
00 00 00 00 00 00 C8 1B 83 EB C8 3E
83 EC C3 20 20 0E C8 09 20 0E 02 E0
83 E0 06 94 C9 20 7D 92 83 02 05 E0
83 73 04 60 00 60 C1 20 16 6C 06 94
02 E0 7D 7E C8 0C 20 0E 03 B0 02 00
0B 00 C8 00 83 4A 04 20 7D 8C 00 4A
04 60 62 84
```

3. XBASIC PROGRAMS DIRECT FROM THE MENU

I once asked Craig whether it was possible to run XBasic programs directly off the menu, as MSAVE does with Basic programs. The answer was no, and essentially that is true, at least as far as having them run directly from GRAM is concerned, since the XML instruction needed exists only in Basic. But I kept on thinking that if XBasic can load a program called LOAD automatically from drive #1, why can't it do others as well! What follows is my method for doing this - it is rather cumbersome since it involves typing code directly into GRAM, no program like GRAM PACKER to do it for you. At least it is rather short! The method involves the following concept: when XBasic starts up, it does a certain amount of housekeeping, and then inserts the string DSK1.LOAD into the crunch buffer in VDP ram, preceded by the length byte >0B and followed by byte >00, and then "pretends" that you typed it in with RUN, and runs it. It turns out that this area is never touched by the housekeeping chores, and hence can be done right at the start. Thus my method involves inserting the program name of your choice there instead, and setting up proper code to make an additional item on the menu. If the program isn't there, you get the same result as XBasic if LOAD isn't in drive 1 - just the "ready" prompt.

First we need to do a little patching of XBasic so that DSK1.LOAD isn't pushed in over what we will put into VDP. Go to your GK memory editor and press FCTN 1 to get to GRAM, then type in 6300 for the memory address. Now FCTN 9 to get to memory window. You should see 06 64 BE. Disable the W/P switch and type in 95 over the BE. This bypasses the move of DSK1.LOAD to VDP. [NOTE: I am using Version 110, I hope all the addresses are the same for you!] Next examine location 6006-6007. This should contain 633B which is a pointer to the first application program (and last in the case of the pure XB module). Now move to 633B, or whatever you found and look at the first 5 bytes. They will conform to TI's standard for application programs. The first two will point to the next application program (in this case 00 00 because there aren't any, but you will eventually replace them because you are creating one) the the next two refer to the start address for this program, and should be 6372 in this case. You will be changing this. the fifth is the length byte for the following text and the text is what will go on the main menu screen. We will use the same set-up for our own application program.

You will now have to decide WHERE you are going to place the code to be described below. If you are using a pure XB module, you can use 7800 since 7800-7FFF is free. I am using Danny Michael's combination XB/EA and so this space is used. There is free space in GRAM 7, starting at F5CE however. Replace the 6372 with the start address you are using. Now go to the address you have selected and type in the following:

```
31 00 0B A8 20 63 51 05 63 72
```

If you did nothing more you would still have the same functioning XB module, because the code you just typed in performs the move of the text DSK1.LOAD and then branches back to where XB normally starts. Now the fun begins. The next address, which is F5D8 will be the location of the next application program, so go back and type it in over the 00 00 at 633B, indicating that there WILL be more. Now decide on the menu name for your program and determine the length of the name (in hex of course). I believe the maximum allowed for the actual name is 18 (>12). Now go back to F5D8 and type 00 00 00 00 xx "text" where xx is the length byte you just determined, and "text" is the actual text for the menu.

Next determine the length of the devicename.filename you wish to have loaded, e.g. DSK4.MENULOAD has a length of >D. Directly after the text for the menu you have just typed in, type this new length byte then the devicename.filename. YOU MUST FOLLOW THE DEVICENAME WITH A 00!!! Note down the address where the length byte of the filename is located AND the address just following the 00 (I will call these ADD1 and ADD2). Now type in the following code:

31 00 yy A8 28 ADD1 05 63 72

where ADD1 is the TWO bytes address just determined, and yy is the length of the filename PLUS TWO. Finally go back to the application program header and type in the two bytes of ADD2 over the 3rd and 4th 00's.

As an example, if the program to be loaded was named MENULOADER and was on DSK4 and your title for the menu was MISC. PROGRAMS, then the code beginning at F5CE should look like this:

```
>F5CE 31 00 00 A8 28 63 51 05 63 72 00 00
>F5DA F5 FC 0E 4D 49 53 43 2E 20 50 52 4F
>F5E6 47 52 41 4D 53 0F 44 53 4B 34 2E 4D
>F5F2 45 4E 55 4C 4F 41 44 45 52 00 31 00
>F5FE 11 A8 28 F5 EB 05 63 72
```

You would then follow the same general rules if you wanted to add more programs to your menu. Now at one key press the program DSK4.MENULOADER would load and run.

By the way, here is a short program allowing you to set up all your favorite programs to run without typing in the names: you merely insert them in the DATA statement, and follow the last with a "" If you save this program on your utility disk under the name you used in the above auto load then you will quickly get a menu of these programs when you press the "MISC. PROGRAMS" key and be able to pick your program with one more key press. This way you can still have the auto load of DSK1.LOAD for use with programs that need it. For this program to run properly you MUST type in line 170 first, exactly as written!

```
100 DATA RD.PRO1,RD.PRO2,""!ADD MORE IF YOU WISH
110 CALL CLEAR
120 X=X+1 :: READ A$(X):: IF A$(X)<>"" THEN 120
130 DISPLAY AT(1,1)BEEP:"PRESS FOR" :: FOR Y=1 TO
X-1 :: DISPLAY AT(2*Y+1,2):Y;"  ";A$(Y):: NEXT Y
140 CALL KEY(0,K,S):: IF S=0 THEN 140 ELSE K=K-48
150 CALL INIT :: B$=A$(K):: L=LEN(B$):: CALL LOAD(
-45,L+4):: CALL LOAD(-42,L)
160 FOR X=1 TO L :: CALL LOAD(X-42,ASC(SEG$(B$,X,1
))):: NEXT X :: CALL LOAD(X-42,0)
170 RUN "0123456789ABCDEF"
```

THOMAS S. FREEMAN

HOME COMPUTER

```
PRESS FOR

A  FOR FUNNELWRITER
B  FOR MYARC DISK MANAGER
C  FOR LOAD TX6
D  FOR LOAD EA/XB
E  FOR DISK+AID
F  FOR PTERM
G  FOR MASS TRANSFER
H  FOR DISKASSEMBLER
I  FOR RAPIDCOPY
J  FOR SKIP & CTRL CODES
K  FOR MENU
L  FOR GK EXTENDED BASIC
M  FOR EDITOR/ASSEMBLER
```

```
1 !HERE IS A SPACE FILLER.THIS PROGRAM WILL PRINT ALL DV/80 FILES ON A DISK,CATALOG FIRST,NEW PAGE TO START EACH FILE
100 OPTION BASE 1
110 DIM A1$(127),DV80(127)
120 OPEN #2:"PIO" :: PRINT #2:CHR$(27)&"N"&CHR$(2);!SKIP OVER PERF
130 CALL CLEAR
140 FOR L=1 TO 5 :: READ T$(L):: NEXT L
150 DATA DIS/FIX,DIS/VAR,INT/FIX,INT/VAR,PROGRAM
160 DISPLAY AT(12,1):"DRIVE #1" :: ACCEPT AT(12,8)SIZE(-1)VALIDATE("1234")BEEP:D$
170 OPEN #1:"DSK"&D$&".",INPUT ,RELATIVE,INTERNAL :: INPUT #1:A$,J,J,K
180 PRINT #2:"DSK";D$;" - DISKNAME= ";A$:"AVAILABLE=";K;"USED=";J-K:" FILENAME  SIZE   TYPE   P"
190 PRINT #2:"---------- --- ---------- -"
200 FOR L=1 TO 127 :: INPUT #1:A1$(L),A1,J1,K1 :: IF ABS(A1)=2 AND K1=80 THEN DV80(L)=1
210 IF LEN(A1$(L))=0 THEN 270
220 PRINT #2:A1$(L);TAB(12);J1;TAB(17);T$(ABS(A1));:: IF ABS(A1)=5 THEN 240
230 B$=" "&STR$(K1):: PRINT #2:SEG$(B$,LEN(B$)-2,3);
240 IF A1>0 THEN PRINT #2 :: GOTO 260
250 PRINT #2:TAB(28);"Y"
260 NEXT L
270 CLOSE #1 :: PRINT #2:CHR$(12);
280 FOR L=1 TO 127 :: IF A1$(L)="" THEN CLOSE #2 :: STOP ELSE IF DV80(L)=0 THEN 310
290 OPEN #1:"DSK"&D$&"."&A1$(L),INPUT :: PRINT #2:CHR$(14);A$;".";A1$(L)
300 IF EOF(1)THEN PRINT #2:CHR$(12);:: CLOSE #1 :: GOTO 310 ELSE LINPUT #1:B$ :: PRINT #2:B$ :: GOTO 300
310 NEXT L
```

CASSETTE DATA FILE PROCESSING ON THE TI.99/4A

by JOHN ROE.

Cassette data file processing on the TI.99/4A is not one of its better features, so until I could afford a disk system I set about trying to make the best possible use of cassettes. How this can be done depends on your hardware configuration. It is easier with PRK or Minimem since both allow you to save data files in "program" format. Extended Basic for all its powerful features gives you no help in this respect.

Is there any difference between saving programs and saving data? and if so, why? There certainly is a difference but why it should be so is a mystery to me in my present state of knowledge of the technicalities. It appears that programs are saved in blocks of 256 bytes. Presumably there is some process to ensure that a program statement is not split between two blocks, so there may be a few blank spaces at the end of a block, but otherwise a block when passed to the tape is all solid information, and there is no wasted tape.

When saving data files however the system is different. It depends on the size of your data record in your OPEN statement. The User Reference Guide suggests you can make your records what size you like up to a maximum of 192. Well, yes you can, and presumably the computor in its own internal workings accepts whatever size you make it. It's a different matter however when it comes to transferring the records to tape. The recorder will only record in blocks of 64, 128 or 192 bytes choosing that one which is next above the record size in your OPEN statement. Any unused bytes on the tape block are filled with blanks. Thus if you make your size 30 than the recorder will record in blocks of 64 bytes, the first 30 being related to your data and the other 34 being filled with blanks. Half the tape is wasted and it takes twice as long to load or save as it should for the amount of data.

So what can we do about it? The short answer is . use Minimem or PRK. The are special and individual problems in using program format with these modules and I propose to deal with those later, and for now set what can be done if you have only the basic console or Extended Basic. First of all however I will get the question of data compression out of the way. This simply means so codifying your data that it takes up less bytes. There are various means of doing this but they all mean extra program statements in codifying the data to record and then decoding again to use or display the data. You may not save all that much memory taking these extra statements into account, and it will probably slow down the execution of your program a little. Data compression may not be relevant to the question of economy of tape usage anyway. There's not much point reducing your records size from 30 to 20 if the records are still going to take up 64 bytes each on your tape,.

My way of saving tape usage is to try and fill each block whatever size it is with useful data and not blanks. To do this I make use of the "pending print" and "pending input" facility (see the Users Reference Guide pages 127 and 133). First of all, let's look at the way a typical data file is recorded. You have an array containing your data which has been DIMensioned as A$(20,3). Your routine to save this array to cassette will include the following statement:-

```
PRINT #1: A$(I,1),A$(I,2),A$(I,3)
```

where I is the index of a FOR-NEXT loop. This statement PRINTs a record on to the tape, but this record consists of 3 separate items, with a comma as separator. Why do these three apparently separate items all go onto one block of the tape? The secret is the comma which appears to be quite a powerful operator.

For our (human) purposes we call the separate items "fields" and the three fields together we call a record. The computor doesn't give a damn what we call them, they are simply separate items of data. Any items of data which are included in one Print statement and separated by a comma constitutes one record for the computor and are recorded in one block of 64, 128 or 192 bytes on the tape. The question arises;

can we include several of our (human) records in one tape record? The answer is yes, we can, as long as they are included in one PRINT statement and separated by a comma. Thus, we could modify the print statement mentioned above to read:-

```
PRINT #1: A$(I,1), A$(I,2), A$(I,3), A$(I+1,1), A$(I+1,2), A$(I+1,3)
```

The FOR-NEXT loop would have to have STEP 2 added to accommodate this PRINT statement. We now have 6 data items recorded in one block on the tape. For the computor it is one record; to us it is two. It could equally have been six records of one field each, but it would still be one record on the tape.

We could reach a position where, in trying to cram as much data into one tape record particularly if it is a tape record 192 bytes long, the entries exceed the allowable length of a program line. So what do we do in this case. The short answer is use two program lines. But a problem arises here. The two lines will constitute two separate PRINT statements and so will be recorded on to two separate tape records. This is where the pending print facility comes in. Simply add a comma after the last entry of the first line as well as the usual commas between the items, and the two lines are treated as one PRINT statement and recorded in one tape record. I said the comma was a powerful operator, didn't I.

All the above applies when you're saving a file. It applies equally when loading a file. The necessary statements will be identical in form, but with INPUT substituted for PRINT.

And now for a practical example. I have a data file of some 120 records of 3 fields dealing with the players in a chess league. For this particular purpose I need to record each players club and his grade which is a two or three digit number calculated on his performance in competitive games over the previous season. This particular file does not process the data arithmetically so the number is recorded as a string. The name of the club is recorded as a two or three letter abbreviation. So first of all an array DIM F$(120,3) with Option Base 1. My first record, which will be record number 0 on the tape will be a Header Record (a useful device copied from the PRK module). So H1$="GRADING", H2$="NAME", H3$="CLUB", H4$="GRADE", Add two numerical fields to this Header Record, R the number of records in the file and PN the "page number" where a "page" consists of 10 records and is calculated from R by the formula PN=INT((R-1)/10).

Players' names are recorded surname first with initials after with no full stops, e.g. SMITH WT. Recorded like this the shortest name takes 5 bytes and the longest I have yet come across 16, the average being about 9. Recorded in INTERNAL format on the tape each field uses one more byte than the number of characters, so the name will take (on average) 10 bytes, the club (3 characters) 4 bytes and the grade (3 characters) also 4 bytes, a total of (on average) 18. Ten records will therefore use 180 bytes on the tape, and with a record length of 192 there are a few bytes spare to allow for the probability of a run of longer than average names in any particular group of ten records.

To PRINT 30 items in one PRINT list takes up two program lines, so the pending print facility described above is used to link the two program lines into the one tape record. An alternative, and quicker to program although possibly slower in execution, is to use a FOR-NEXT loop for each group of ten records, as follows:-

```
310 FOR I=1 TO 10 :: FOR J=1 TO 3
320 PRINT #1: A$(I,J),
330 NEXT J :: NEXT I
```

There is a snag here however. In line 320 the comma at the end of the line is there to link up the records in one group of ten into one tape record. Unfortunately it doesn't finish there. When the next group of ten starts the pending print is still in

operation and the and the data is included in the first group of 10 on the tape record. Of course, it soon goes over the 192 and the program crashes with File Error. To avoid this the FOR-NEXT loop should only go up to 9 and the 10th record is PRINTed following the loop and without the pending print comma.

My file program is as follows, with only the saving, loading and display routines given in detail.

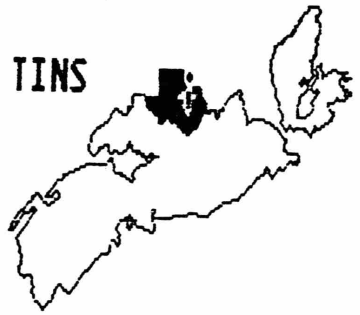
```
100-120 SCREEN & DIM INITIALISATION
130-170 MENU DISPLAY & CHOICE ROUTINE
180-260 ADD RECORDS ROUTINE (Last line is260 PN=INT((R-1)/10)   )
270 CALL CLEAR::    REM - SAVE FILE ROUTINE
280 OPEN #1:"CS1",INTERNAL,OUTPUT,FIXED 192
290 PRINT #1:H1$,H2$,H3$.H4$,R,,PN
300 FOR P=0 TO PN
310 FOR I=1 TO 9 :: FOR J=1 TO 3
320 PRINT #1:A$(P*10+I,J),
330 NEXT J :: NEXT I
340 PRINT #1: A$(P*10+10,1),A$(P*10+10,2),A$(P*10+10,3)
350 NEXT P
360 CLOSE #1
370 CALL CLEAR :: GOTO 140:: REM - RETURN TO MENU
380 CALL CLEAR ::  REM LOAD FILE ROUTINE
390 OPEN #1:"CS1",INTERNAL,INPUT,FIXED 192
400 INPUT #1:H1$,H2$,H3$, H4$,R,PN
410 FOR P=0 TO PN
420 FOR I=1 TO 9 :: FOR J=1 TO 3
430 INPUT #1: A$(P*10+I,J),
440 NEXT J :: NEXT I
450 INPUT #1: A$(P*10+10,1),A$(P*10+10,2),A$(P*10+10,3)
460 NEXT P
470 CLOSE #1
480 CALL CLEAR : : GOTO 140:: REM - RETURN TO MENU
490 CALL CLEAR ::   REM - DISPLAY ROUTINE
500 DISPLAY AT(1,7):"FILE STRUCTURE": : :"FILE NAME;TAB(16);H1$
510 DISPLAY AT(6,1):"No. OF RECORDS";" ";R
520 DISPLAY AT(8.1):"FIELD No. 1";TAB(13);H2$
530 DISPLAY AT (10,1):"FIELD No. 2"TAB(13);H3$
540 DISPLAY AT (12,1):"FIELD No. 3";TAB(13);H4$
550 DISPLAY AT(24,1):"PRESS ANY KEY TO CONTINUE"
560 CALL KEY(0,K,S) :: IF S=0 THEN 560
570 PRINT:"PRESS ANY KEY TO SCROLL RECORDS"
580 FOR I=1 TO 200 :: NEXT I :: REM  DELAY LOOP
590 FOR I=1 TO R
600 PRINT:A$(I,1);TAB(18);A$(I,2);TAB(24);A$(I,3)
610 CALL KEY(0,K,S) :: IF S=0 THEN 610
620 NEXT I
630 CALL CLEAR :: GOTO 140:: REM - RETURN TO MENU
```

There are other ways of doing this , it depends on your particular application which is best, but all would use the pending print facility in some way to pack the tape records with useful information.

One last word regarding the general use of file handling programs whether on tape or on disc. I have used a number of commercial file handling programs (on cassette) and also the PRK module. On the whole they are quite useful but have their limitations. They all tend to impose restrictions on the number of records, fields or chars in a field which may not be acceptable in your particular application. Furthermore, in the nature of things these programs have to be fairly general in character to provide for all foreseeable requirements of your application. They thus necessarily include more programming and more complicated programming than may be necessary for your purpose. This uses up valuable memory which could be better put to use for data storage.

As an example of the effect of restrictions I tried using the PRK module to create a file of all my classical music records. This has a limit of 15 chars for any item. This is OK for the usual musical forms as one can use abbreviations, Sym for symphony, con for concerto etc. But what do you do about those pieces which don't fall into these categories? How do you fit Mozart's Eine Kleine Nachtmusik into an item of only 15 chars? Of conductors Beecham or Klemperer go in all right, but what about Schmidt-Isserstedt? Also if you want to scroll through the whole file this module is painfully slow, and even worse if you want to sort the file.

I have come to the conclusion that it is best to write an individual program for each application using the minimum of simple programming that will serve your purpose so saving memory for the maximum amount of data. If you have a commercial program in TI Basic you could start with that, set up a pilot file to test it out, then amend it, simplify it and cut out any programming not required. The stripped down program could be tested again with your pilot file to debug it, and when debugged it could then be used for your full scale operation. If your program is in Extended Basic, hard luck, it will almost certainly be protected so you can't do anything to improve it.



The Faire!

BY TERRY.
DARTMOUTH, NOVA SCOTIA

01-Nov-86 07:30 PM

From the excited pen of Ron Albright comes the first report of the Faire, while it was still going on. ....(from CIS)....

Well, folks, I am here at the TI Fair and I can tell you that I feel this is gonna be the best Fair YET! There has got to be 1000+ cruisin' here and it is packed ABSOLUTELY! The vendors are doing a brisk business and the folks are buying like crazy! CAN YOU BELIEVE IT? This lil' orphan is, indeed, ALIVE AND WELL! I hope to be able to be up tonight and get something ONLINE but we will have to see how that goes... ANYWAY, I can say unequivocaly, that the TI FAir #4 is a BIG SUCCESS. Thanks to the Chicago UG for making this a nother beauty! Back with more later! LOU PHILLIPS JUST FINished his presentation and (from what I gaterhed) not much is new...they HOPE that they can begin shipopping prior to the end of the year, but the hang up is Mitsubishi...the "gate array" are wrong the first time and they are being re-done. Nice presentation by ASGARD software earlier and a TI User Group seminar on problems that face them all, and an EXCELLENT presentation by Clint Pulley on c99...a VERY bright guy for sure. The Fair is STILL smokin' and the crowds are big.(.I mean REALLY big. If I had to estimate, I would say well over 2000 total attendance throughout the day. Again, it has been another BIG success. I have seen Art Byers, Terrie Masters, Coe Case, Warren Agee, Gary Cos, Todd Kaplan and MANY more of the CIS friends. Its been a great day and it is still goin... Out here! JUST GOT THE WORD ON ATTENDANCE... Seems over 900 folks turned the turnstiles here in the Windy City...the fair organizers attributed the lower attendance (compared to last year) to the inclimate weather (a polite way to say the weather was miserable!!)...the sellers on the other hand are VERY pleased...each and everyone have virtually sold out. They have literally sold it all! This has been the most "spend-thrift" TI Fair of all...I mean that! It has become a seemingly CULT life for the TI...selling everything from adoption certificates to buttons and (one UG representative - Dick Vandenberg - of the Mid-South UG - estimated a $75/member expenditure for their 11 attendees)...spend, spend, spend! A Fifth Annual TI Fair for next year! They love it, the vendors love it, and anyway, "a great time was had by all!!"

Comments by Art Byers: The fourth annual TI Fair hosted by the Chicago TI UG was a fine success from all points of view: Around 900 99/4A owner/users from the middle of the nation (and as far away as Boston, Washington DC and New York) had a chance to see the newest and best in software and hardware and to spend some money on goodies. So- The Vendors obviously did a great deal of business and will be encouraged to attend other TI shows around the nation. The speakers and events were interesting and well run. Those attending learned much that was new. Best of all, however, was the chance to meet and mix with the 99'er community, to exchange Ideas and information, and to renew our enthusiam as we proved once again the motto originated by Henry Hein: "We may be an Orphan, but we have a great future!". For me, the occassion meant putting faces together with many of the famous names such as Chris Bobbit (Asgard), Jim Horn and Jeff Guide (Disk Only Software, Ron Albright (The Orphan Chronicles), Theresa Masters (president o the LA UG),- plus well known programmers such as Peter Hoddie, Coe Case, Todd Kaplan, and Paul Charlton, - oh yes! and the Tigercub Himself and the most well known of them all, Jim Peterson. Without Jim half of most UG newsletters would be blank space! It was nice to talk to Walt Howe and Bob Demeter in person, instead via Telecom. What was new? Oh Boy! Lots was new! J Peter Hoddie has a new Graphics program "Font Writer" being marketed by Asgard. ("What! another graphics program" you are about to say..- but you have to see it to belive it. That was only a start. There must have been 50 new software items offered by vendors covering everything from games and education to disk utilities and home finance. Continued next message. DataBiotics long awaited PILOT language with extensive documenation was finished and on sale Plus Super Forth. New Hardware?? Yes and plenty of it! Mini PE Boxes for 3,4 or 5 cards,

and prototypes of everything from a sort of super widget that enables you to use the "Review Module Library" hidden software built into the Console to a whole new Computer (MYarc 9604). One of the big hits of the show was the new IBM type keyboard for our 99/4A, made by RAVE Mfg. I have an interview on tape with one of the principals involved and will print it in detail in the CALL SOUNDS newsletter. During the lunch break, there was a fascinating rap session for the visiting UG representatives. Everything was covered from rumors of more software to be released to PD and UG's by Texas Instrument (a diagnostic disk) to fundraising and dues policies, methods of software library distribution, and newsletter exchange. One of the best talk/demonstrations of the day was the hour on Music and the 99/4A, put on by J Peter Hoddie. That included two linked computers playing music together, music that played while you are programming, and two pieces where Peter played the Cello accompanied by the 9974A!!! All in all the Chicago club is to be highly complimented. Everything ran smoothly. Those of you who could not make this show are advised to make the next one near you be it Seattle, Boston, Dallas or New Jersey. These fairs are FUN!! I think one of the most exciting announcements was the teaming of two REAL genuises together in future software/hardware projects.. Barry Traver and J. Peter Hoddie...that team is unbeatable for sure.

MYARC'S GENEVE

Who needs it? We do!
Scott Flinn Oct. 18, 1986

I have just finished reading The Orphan Chronicles by Ronald Albright. As advertised, it is a remarkable book which answers virtually all of the seemingly unfathomable questions left behind by Texas Instruments. Although I was never really bothered by being a computer orphan (I certainly don't remember where I was when I heard THE news about TI's departure from the market), I must admit to being extremely relieved upon finding the TI Nova Scotia (TINS) club. The mild but constant worries about blowing another power board (I had already blown one), or wearing out another cartridge port connector (I had already gone through three) were ended. This discovery coincided perfectly with my introduction to TMS 9900 assembly, and the answers, hints and advice I found at the club were invaluable (needless to say appreciated).

So what does this have to do with The Orphan Chronicles and Myarc's new computer? In his book, Mr. Albright makes it very clear that, although the TI community is currently thriving, we are all going to have to work very hard to ensure its continued existence. The two points he addresses most directly are, firstly, that we must actively pursue the production of hardware and software, either by producing it ourselves, or by buying it from others, thereby encouraging the talented programmers to continue; secondly, we must never become discouraged to the point where we may consider the possibility of getting a "better" machine. Toward the latter end, Mr. Albright strongly suggests that TI'ers forget about the prospects of a new compatible machine and concentrate on getting the most from their TI's. At this point I will be charitable, and will interpret Mr. Albright's remarks in a favorable way. I assume his reasoning is that if we allow ourselves to dream too much about a new and better computer, we may actually succeed in convincing ourselves that we actually need it. Once this happens, if the computer has not yet arrived (and Mr. Albright believes that one never will), then we may conclude that, since a new computer would be better, but a new TI compatible is not available, we must abandon TI and get a real computer like an Atari, or an IBM. This type of thinking could be very detrimental to the continued existence of the TI community, and Mr. Albright makes a noble and selfless attempt to warn people away from it. His most persuasive argument is the simple, almost rhetorical question "WHO NEEDS IT? WHAT CAN'T YOU DO ALREADY?".

This question has already been put to me several times, and I must honestly say that it stopped me dead in my tracks. My immediate response was "Well... um... ah... 80 columns. Yeah, that's the ticket, 80 columns! I need 80 columns." But try as I did, I was unable to think of a single example of an application which, though too demanding for the TI, could be handled by the new machine. Word processing, spreadsheet, data base, telecommunications,

computation, sound, even graphics; in spite of the 40 column limitation, all are handled extremely well by the existing machine, with the most exciting software and hardware only now reaching the market. A good friend of mine - an Apple \\c owner - just recently phoned me, more excited than I have ever seen him, to tell me about the latest technological miracle from Apple. Apparently, Apple is about to release a new computer which, though not unusually better than their existing machines, will have truly unbelievable sound capabilities; a fifteen voice, fully interfacable synthesizer built right in to the computer. I almost didn't have the heart to tell him that such a beast has existed for the TI for about three years now (namely the FORTI Music System, driven with FORTH based software). Once the inanity of GPL has been circumvented, it becomes all too clear that the TI-99/4A is actually a very fast, very powerful computer. My own favorite passtime is comparing the raw speed of the TI with other machines (ie. number of integer additions, multiplications, comparisons, branches, block moves, etc. per second). It compares VERY well with all but the newest machines, such as the Atari, the Amiga, the Panasonic Executive Partner, etc.

With a language like FORTH available, Myarc's XBasic II plus 512K card, etc., the question "What do you need a new computer for?" is a very good one. But it can be answered! In fact, there are three good reasons. The first one becomes obvious if, instead of asking what can't be done now, one asks oneself what can be done now. My own applications have included extensive use of TI-Writer, TE-II and Fast-Term, self-written programs for inventory management (23,000 item inventory), yacht club handicap calculations and records, one and two dimensional function graphing, games of all types, and many others. Each and every one of these applications would benefit from the capabilities of the new computer. The 80 column advantage really is quite large, making word processing, telecommunications, and spread sheet software far easier to use. Self written data base applications would certainly profit from the far greater memory. These applications are obviously disk based, and require hours rather than minutes to perform simple tasks. The mathematical and number crunching programs would certainly benefit from increased speed, and better graphics would be a non-essential but much welcomed plus in this area as well. The second reason can be summed up quite simply: imagine what its potential will be! Who would have guessed when they bought their TI five years ago that today their little machine would still be capable of doing all of the things the "newer, bigger and better machines" are only now achieving? If you had asked someone four years ago "What do you want a computer for? What could you possibly use it for?", you can be certain that the answer would not have been "Oh, I want to use my FORTI board to play 'Chariots of Fire' in twelve synthesized voices", or "Well for starters, I'm going to set up a BBS, and then I'll use it to write The Orphan Chronicles in my spare time". One thing is clear: a new machine will be a better machine. It can only be better, because, while keeping the TI spirit, it will be able to do everything the TI can do now... and much more. Thus, if the TI was capable of so much, and still compares favorably to other machines four or five years newer, it is only reasonable to expect that a new machine, when pushed to its limits, will do things that we can't even imagine now, because nothing else can do it yet. While on the surface, Myarc's claim of a machine "2-3 times faster" is not particularly stunning, once you realize just how fast the TI is now (when GPL is not used and the program is not graphics intensive), it becomes clear that the speed will only be equalled by the best of the modern machines. Its graphics, according to the description of the V9938 video processor given by Ryte Data and others, will be unquestionably better than any computer less than six times the price; and it has more internal memory than any small business is ever likely to need. Definite answers can not be given now as to what will be done with the machine that can't already be done. Only time will tell what will be produced when a machine with such capabilities is pushed to such limits as only TI'ers have had the need to achieve. Finally, we must all at some time face the hard fact that, although the TI-99/4A is a phenomenally durable creature, it cannot possibly live forever.

We are faced with two scenarios. Firstly, we can all be content in the knowledge that what we have now is a small piece of quality that will serve us unerringly for many fruitful years. With the products now available, and those that are undoubtedly waiting in the wings, we can build our systems to a point where the choice to

obtain an upgrade would be a matter of taste, not necessity. Further, there are those among us who will always be content with what they have, for whatever reason. If a person uses powerful computers at their place of employment, having something at home that plays reasonable games and calculates square roots accurately may well be all that is ever desired. However, no matter how powerful the peripheral system becomes, the fact remains that the computers themselves are not being manufactured. Eventually they will wear out, as everything must, and with the heart removed, the system will be useless. All but a determined and clever few will write off their investments as worthwhile, but expended, and will either get the newest computer currently offered, or will leave the computer world entirely. The second scenario includes the fact of a new computer. It would be nice if, when the heart of powerful system dies, it could be replaced with something equally good; a point which I believe is common to us all. But it makes even more sense to replace that heart now if the replacement brings with it all of the peripherals. It is certainly true that a TI with a 512K expansion card with RAM disk capability, the new eighty column card or the V9938 video processor installed on the mother board, cartridge software that permits direct use of the expansion RAM, and a number of compiler languages such as FORTH and 'Small C' as well as Myarc's "better" Extended Basic would probably be roughly as good as the new machine (note that not ALL of these features exist yet). But does it not make more sense to spend the same amount of money, perhaps even less, to get all of this rolled into one package, particularly when it replaces some of the most easily damaged and uneasily repaired parts of the machine such as the keyboard, cartridge port and power supply? And when we are guaranteed, at least for a little while, that such an upgrade will be supported by a manufacturer who has plans, however questionable, for IBM compatibility, a complete C language, and who knows what else? Perhaps the most reasonable thing of all, however, is that, while the TI community can not possibly be hurt by a better computer with a high degree of compatibility, it will be helped enormously, to the point of its very survival. The TI is now heading toward a dead end. Myarc is offering a chance, however small, for the TI legacy to continue in the face of a competition whose strength grows geometrically. When Mr. Albright made the remark that we should dismiss the idea of a new computer, he was working under the assumption that such a computer would never be produced, and that thinking about it was an unnecessary temptation. The chance of a new computer appearing is now quite high (reliable rumor has it that the machine is finished and will become available as soon as the documentation is complete), and we must begin thinking about what we will do should it arrive. Naturally not everybody is going to either want or need such an upgrade, and there should be no pressure to make the switch, particularly since the TI is, after all, quite adequate. However, the pioneers among us, new and old, must be willing to do our job as thoroughly and with as much commitment as we possibly can. It is up to us to see that the Myarc computer survives so that in the coming years, as TI systems begin to falter, it will in turn be able to ensure that the TI community survives. With so many disappointments in the past, it is easy to be bitter toward rumors of a promised land, but we must not deal ourselves the final blow by turning our backs on the only opportunity Fate is likely to offer.

TINS

THE FAIRE PART TWO

by TERRY.
DARTMOUTH, NOVA SCOTIA

And these comments from Warren Agee
Well I'm back from the Chicago faire, and I had a fairly nice time. Great to see familiar faces and a few new ones (PaulC, Ron, Clint, Todd K, Walt).
I also finally got to see a face that has eluded this individual from the start...the face of the Geneve.
I guess the best way I can describe how the faire affected my opinion of this machine is to say that going in to the faire I had a rather negative attitude. Coming out of the fair I have a **moderately** optimistic attitude.
As far as technical superiority, if you hear all the hype about the fact that the Geneve is in some cases graphically superior to the Amiga, BELIEVE IT! I was thoroughly impressed in this regard.

The info that encouraged me the most is the fact that Lou is talking to people about having an outside company package and produce and market the final version of Geneve, the standalone computer. I feel this is *vital* to the life of this machine beyond our tiny little forum of friends and hackers.
That is IF they ever get the gate arrays from Mistubishi. I'm running out of time here, so let me just clarify two things: it was originally announced that the Geneve would use a Microsoft mouse; I guess they finally decided on an Amiga mouse. Also, Clint Pulley just this last tuesday managed to, with just a little effort mind you, get c99 up and running on his prototype of the Geneve. Sounds pretty good.
At any rate, I will be content to sit back and wait and watch to see what kind of direction this plan of action will take...I bought one orphan, and I am not about to put out money for something that has NO chance for any kind of survival. I wish Myarc all the best.

[This was submitted for the Nov TINS newsletter. However, due to the importance of the material and the excellent presentation, I thought that we would be very negligent if we were to sit on it. Please feel free to run this article in any newsletters.]

BOX 3391 DARTMOUTH NS B2W 5G3 (CANADA)

The following handy TI-WRITER commands are reprinted from the June issue of the 99'er News published by the TI Users Group of Will County, Romeoville, Ill. This puts the most used commands on one page for handy access at your computer.

EDITOR COMMAND	FCTN	CTRL	EDITOR COMMAND	FCTN	CRTL	EDITOR COMMAND	FCTN	CTRL
Back tab		T	Ins. Blank line	8	O	Quit	=	
Beginning/line		V	Insert character	2	G	Reformat		2orR
Command/escape	9	C	Last paragrapph		6orH	Right arrow	D	D
Delete character	1	F	Left arrow	S	S	Roll down	4	A
Del. end of line		K	Left margin rel.		Y	Roll up	6	B
Delete line	3	N	New page		9orP	Screen color		3
Line #'s(on/off)	0		New paragraph		8orM	Tab	7	I
Down arrow	X	A	Next paragraph		40rJ	Up arrow	E	E
Duplicate line		S	Next window	5		Word tab		7orW
Home cursor		L	Oops!		1orZ	Word wrap/fixed		O

Load files: LF (enter) DSK1.FILENAME (load entire file)
LF (enter) 3 DSK1.FILENAME (merges filename with data in memory after line 3)
LF (enter) 3 1 10 DSK1.FILENAME (lines 1 thru 10 of filename are merged after line 3 in memory)
LF (enter) 1 10 DSK1.FILENAME (loads lines 1 thru 10 of filename)

Save files: SF (enter) DSK1.FILENAME (save entire file)
SF (enter) 1 10 DSK1.FILENAME (save lines 1 thru 10)

Print Files: PF (enter) PIO (prints control characters and line numbers)
PF (enter) C PIO (prints with no control characters)
PF (enter) L PIO (prints 74 characters with line numbers)
PF (enter) F PIO (prints fixed 80 format)
PF (enter) 1 10 PIO (prints lines 1 thru 10)
NOTE: The above assumes PIO. DSK1.FILENAME, and RS232 are also valid!
To cancel the print command press FCTN 4.

Delete file: DF (enter) DSK1.FILENAME

Setting Margins and Tabs: (16 tabs maximum)
L - Left margin R - Right margin I - Indent T - Tab
Use ENTER to execute or COMMAND/ESCAPE to terminate command.

Recover Edit: RE (enter) Y or N

Line moves: M (enter) 2 6 10 (moves lines 2 thru 6 after line 10)
M (enter) 2 2 10 (moves line 2 after line 10)

Copy: same as move except use C instead of M.

Find String: FS (enter) /string/ (will look for string in entire file)
FS (enter) 1 15 /string/ (will look for string in lines 2 thru 15)

Delete: D (enter) 10 15 (deletes lines 10 thru 15 in memory)

TIPS FROM THE TIGERCUB

TIPS FROM THE TIGERCUB

The READFILE subprogram on my Nuts & Bolts #2 disk has a backward parentheses in line 21161. This is the corrected line -

```
21161 DISPLAY AT(17,1):"OPEN
 PRINTER #":"NAME? " :: ACCE
PT AT(17,15)VALIDATE(DIGIT)S
IZE(-3):P :: ACCEPT AT(18,7)
:P$ :: OPEN #P:P$ :: GOTO 21
163
```

When Texas Instruments developed Extended Basic, they took away the ability of Basic to redefine or color the characters in sets 15 and 16, ASCII 144 to 159, in order to make room in memory for sprites (they did let us have color set 0 instead. That is why Basic programs which use sets 15 and 16 will crash if you try to run them in XBasic.

Finally, John Behnke published in the Chicago Times newsletter an amazing routine which gave us back those missing sets. His routine was 13 sectors long. Recently, Richard Heath published in the L.A. newsletter a shortened version. And, without having any idea how it works, I have managed to scrunch it down to only 4 sectors -

```
1 CALL BXB
29999 !BXB by Jim Peterson,
adapted from VDPUTIL2 by Joh
n Behnke/Richard Heath
30000 SUB BXB :: CALL INIT :
: CALL LOAD(8194,37,194,63,2
40)
30001 CALL LOAD(16368,80,79,
67,72,65,82,37,58,80,79,75,6
9,86,32,37,168)
30002 !
30003 FOR J=1 TO 136 :: CALL
 LOAD(9529+J,ASC(SEG$(]I[\[]$
,J,1))):: NEXT J :: SUBEND
30004 SUB CHAR(A,A$):: CALL
LOAD(9500,A):: CALL LINK("PO
CHAR",A$):: SUBEND
30005 SUB COLOR(A,B,C):: CAL
L LOAD(9492,8,15+A,(B-1)*16+
C-1)
30006 CALL LINK("POKEV"):: S
UBEND
```

Note tha line 30002 is missing. That's because there is no way to key it in. Once again we need a program that writes a program -

```
100 FOR J=1 TO 136 :: READ A
 :: M$=M$&CHR$(A):: NEXT J
110 OPEN #1:"DSK1.BXBDATA",V
ARIABLE 163,OUTPUT :: PRINT
#1:CHR$(125)&CHR$(0)&"]I[\[]$
"&CHR$(190)&CHR$(199)&CHR$(1
36)&M$&CHR$(0)
120 PRINT #1:CHR$(255)&CHR$(
255):: CLOSE #1
130 DATA 2,224,37,20,3,0,0,0
,2,5,48,48,2,6,37,2,205,133,
2,134,37,17
140 DATA 17,252,4,192,2,1,0,
1,2,2,37,1,2,3,18,0,212,131,
4,32,32,20
150 DATA 200,4,9,80,2,32,3,0
,2,1,37,2,2,2,0,8,2,7,11,0,2
,8,7,0,193
160 DATA 1,192,193,193,180,9
7,133,145,135,21,1,113,136,6
,198,145
170 DATA 135,21,1,113,136,21
0,70,10,198,177,137,220,198,
2,131,37,10
180 DATA 17,240,4,32,32,36,1
6,6,2,224,37,20,3,0,0,0,4,32
,32,32,4
190 DATA 192,216,0,131,124,2
,224,131,224,4,96,0,112
```

RUN that to create a file BXBDATA on the disk. Then load the BXB program, and enter MERGE DSK1.BXBDATA. The unprintable line will pop into place. SAVE this completed BXB routine in MERGE format, and merge it into any Basic-only program. If you want, the result can be run through a Compactor program and turned into multi-statement program lines for more speed.

Or, you can write an Extended Basic program using all 16 character sets for graphics and color - actually 17, because set 0 is also available. Even the characters 24 through 31 can be redefined! Craig Miller has warned against fooling around in that area of memory, but there seems to be no problem with redefining the cursor (30) or the edge character (31).

Sprites can only use characters between 32 and 143 and their color cannot be changed with CALL COLOR(#,_,_). I have not found any other bugs, but have not had time for much experimenting.

Here's an easy Tigercub challenge - run this one in Basic, not Extended Basic.

```
>LIST
100 DISPLAY AT(1,1):0
>RUN
 0
 0
```

Why did it print the zero twice?

I wrote this next one primarily for blind users. It converts each PRINT or DISPLAY directly to speech output and also provides a speech prompt for INPUTs.

```
100 !PRINT SPEAKER by Jim Peterson - to add OPEN #1:"SPEECH",OUTPUT and convert PRINT and DISPLAY statements to PRINT #1
110 !Also writes a PRINT #1 for INPUT prompts
120 !Program to be converted must first be SAVEd in MERGE format. Recommend it be RESequenced before SAVEing, to make room for INPUT lines
130 PS$=CHR$(156)&CHR$(253)&CHR$(200)&CHR$(1)&"1"&CHR$(181)
140 DISPLAY AT(3,1)ERASE ALL:"INPUT FILENAME?":"DSK" :: ACCEPT AT(4,4):IF$ :: OPEN #1:"DSK"&IF$,INPUT ,VARIABLE 163
150 DISPLAY AT(5,1):"OUTPUT FILENAME?":"DSK" :: ACCEPT AT(6,4):OF$ :: OPEN #2:"DSK"&OF$,OUTPUT,VARIABLE 163
160 PRINT #2:CHR$(0)&CHR$(1)&CHR$(159)&CHR$(253)&CHR$(200)&CHR$(1)&"1"&CHR$(181)&CHR$(199)&CHR$(6)&"SPEECH"&CHR$(179)&CHR$(247)&CHR$(0)
170 LINPUT #1:M$ :: P=POS(M$,CHR$(156),3):: A=POS(M$,CHR$(162),3):: Z=POS(M$,CHR$(181),3)
180 I=POS(M$,CHR$(146),1):: IF I=0 THEN 210 :: IF Z=0 OR Z<I THEN PRINT #2:M$ :: GOTO 240
190 M2$=SEG$(M$,1,1)&SEG$(M$,2,1)&PS$&SEG$(M$,I+1,Z-I-1)&CHR$(0):: PRINT #2:M2$
200 PRINT #2:SEG$(M$,1,1)&CHR$(ASC(SEG$(M$,2,1))+1)&SEG$(M$,3,255):: GOTO 240
210 IF P+A=0 THEN PRINT #2:M$ :: GOTO 240
220 M=MAX(P,A)
230 M$=SEG$(M$,1,2)&PS$&SEG$(M$,M+1,255):: PRINT #2:M$
240 IF EOF(1)<>1 THEN 170 ELSE CLOSE #1 :: CLOSE #2
250 DISPLAY AT(12,1)ERASE ALL:"Type NEW and Enter" :: DISPLAY AT(15,1):"Type MERGE DSK";OF$ :: END
```

```
*****************************
      MOLLY DARLING
*****************************
100 CALL CLEAR :: CALL SCREEN(5):: FOR SE=1 TO 12 :: CALL COLOR(SE,16,5):: NEXT SE
110 DISPLAY AT(3,8):"MOLLY DARLING": :" Written and performed by": :TAB(9);"Eddy Arnold" :: DISPLAY AT(24,1):"Programmed by Jim Peterson"
120 FOR D=1 TO 200 :: NEXT D :: DISPLAY AT(12,1):"Just a moment.....": :".....looking for my music..."
130 DIM N(100),N2(100),A(250),B(250),C(250):: F=110 :: FOR J=1 TO 90 :: N(J)=INT(F*1.059463094^(J-1)+.5):: NEXT J
140 DATA 16,11,8,16,8,11,16,4,11,18,11,8
150 DATA 20,16,11,23,11,16,25,21,16,20,16,21
160 DATA 23,20,16,23,16,20,23,11,16,23,16,11
170 DATA 20,11,16,20,16,11,20,8,11,20,11,8
180 DATA 20,11,16,25,16,11,23,11,16,20,8,4
190 DATA 18,16,10,18,16,16,18,16,10,18,11,16
200 DATA 18,15,11,18,9,15,18,11,9,18,9,3
210 DATA 20,8,1,20,13,8,20,8,13,20,13,4
220 DATA 27,20,18,27,18,20,20,18,12,20,12,18
230 DATA 25,21,16,25,16,21,25,13,16,25,16,13
240 DATA 27,23,21,27,21,23,27,23,18,27,18,21
250 DATA 20,23,20,20,20,23,20,20,16,27,16,20
260 DATA 30,21,13,20,13,21,27,21,13,25,13,21
270 DATA 23,20,16,23,16,20,20,11,16,20,16,11
280 DATA 30,23,13,20,13,23,23,20,13,20,13,16
290 DATA 25,21,16,25,16,21,25,21,16,27,16,21
300 DATA 20,23,20,20,16,11,18,15,11,20,11,15
310 DATA 16,11,8,16,8,11,16,9,1,16,1,9
320 DATA 16,11,8,16,8,11,16,1,8,16,13,1
330 DATA 25,21,16,25,16,13,25,13,9,25,9,4
340 DATA 23,20,16,23,16,11,23,11,8,23,8,4
350 DATA 21,18,11,21,11,9,21,9,6,20,6,3
360 DATA 21,16,11,20,16,11,20,11,8,20,8,4
370 DATA 18,13,10,18,10,6,18,6,1,20,13,10
380 DATA 22,18,13,20,22,18,27,18,22,25,22,18
390 DATA 23,18,15,23,15,11,23,11,6,23,6,3
400 DATA 23,21,15,23,15,11,23,11,9,23,9,6
410 DATA 16,13,8,16,8,13,16,13,8,18,13,9
420 DATA 20,11,8,21,8,11,20,11,8,18,11,6
430 RESTORE 140 :: T=16 :: GOSUB 480 :: RESTORE 140 :: T=4 :: GOSUB 480 :: RESTORE 180 :: T=12 :: GOSUB 480 :: RESTORE 140 :: T=16 :: GOSUB 480
440 RESTORE 210 :: T=28 :: GOSUB 480 :: RESTORE 170 :: T=4 :: GOSUB 480 :: RESTORE 250 :: T=4 :: GOSUB 480 :: RESTORE 280 :: T=4 :: GOSUB 480 :: RESTORE 190 :: T=8
450 GOSUB 480 :: RESTORE 140 :: T=16 :: GOSUB 480 :: RESTORE 290 :: T=48 :: GOSUB 480 :: RESTORE 140 :: T=16 :: GOSUB 480 :: RESTORE 410 :: T=8 :: GOSUB 480
460 RESTORE 310 :: T=8 :: GOSUB 480 :: GOTO 490
470 GOTO 490
480 FOR J=1 TO T :: X=X+1 :: READ A(X),B(X),C(X):: A(X)=A(X)+12 :: B(X)=B(X)+12 :: C(X)=C(X)+12 :: NEXT J :: RETURN
490 DISPLAY AT(10,1):"Control volume of 3 voices":"using 1, 2 and 3 keys for":"louder and Q, W and E for":"softer.":""
500 DISPLAY AT(15,1):"Control speed using 'F' for":"faster and 'S' for slower."
510 DISPLAY AT(18,1):"Change key using 'A' for":"higher and 'D' for lower."
520 DISPLAY AT(21,1):"Press 'Z' for minor key, 'X'":"for major key." :: V1,V2,V3=10 :: F,P,Y=0 :: X=200
530 FOR J=1 TO 192 :: CALL SOUND(-999,N(A(J)-Y),V1,N(B(J)-Y),V2,N(C(J)-Y),V3):: FOR T=1 TO X/50 :: P=1^X :: NEXT T
540 CALL KEY(0,K,S):: IF S<1 THEN 710 :: ON POS("123QWEFSADZX",CHR$(K),1)+1 GOTO 710,550,560,570,580,590,600,610,620,630,650,670,690
550 V1=V1-1-(V1=0):: GOTO 710
560 V2=V2-2-(V2=0)*2 :: GOTO 710
570 V3=V3-2-(V3=0)*2 :: GOTO 710
580 V1=V1+2+(V1=30)*2 :: GOTO 710
590 V2=V2+2+(V2=30)*2 :: GOTO 710
600 V3=V3+2+(V3=30)*2 :: GOTO 710
610 X=X-20-(X<2)*20 :: GOTO 710
620 X=X+20 :: GOTO 710
630 IF F=1 THEN GOSUB 700
640 Y=Y-1-(Y=-20):: GOTO 710
650 IF F=1 THEN GOSUB 700
660 Y=Y+1+(Y=6):: GOTO 710
670 IF F=1 THEN 710 :: GOSUB 680 :: GOTO 710
680 F=1 :: Y=0 :: FOR W=3 TO 27 STEP 12 :: N2(W)=N(W):: N(W)=N(W-1):: N2(W+5)=N(W+5) :: N(W+5)=N(W+4):: N2(W+10)=N(W+10):: N(W+10)=N(W+9):: NEXT W :: RETURN
690 IF F=0 THEN 710 :: GOSUB 700 :: GOTO 710
700 F=0 :: FOR W=3 TO 27 STEP 12 :: N(W)=N2(W):: N(W+5)=N2(W+5):: N(W+10)=N2(W+10):: NEXT W :: RETURN
710 NEXT J :: J=192 :: FOR V=10 TO 30 :: CALL SOUND(-999,N(A(J)-Y),V,N(B(J)-Y),V,N(C(J)-Y),V):: NEXT V :: FOR D=1 TO 500 :: NEXT D :: GOTO 530
```

This will be the last issue of the Tips from the Tigercub.

I started this newsletter over 3 years ago, as a means of promoting my software business. It has never been a success for that purpose, but I have kept it going because of the many interesting newsletters that I have received in exchange, and the many friends that I have made around the world.

I know, from the editors' comments in many of your newsletters, that many of you are finding it difficult to finance a newsletter for your shrinking membership, and even more difficult to find the time, and the material to print. For a one-man user's group pretending to be a business which is getting very little business, it has become impossible. User group members have never been good customers for anyone's software, for reasons which you all know, and those who are remaining active in the TI world are wanting more sophisticated software than I have to offer.

Some of you have offered to subscribe to my Tips, but I just don't have the time to get involved in anything like that. I have had some other projects on the back burner for too long, and it's time I got to work on them - they can hardly turn out to be less profitable than trying to sell software!

I am NOT going out of business, and I am NOT releasing my programs to the public domain. I will continue to sell them, and will continue some classified advertising.

My heartfelt thanks to the many user group editors and officers who have tried in many ways to encourage and help me. Many thanks to those who have purchased my programs.

I will greatly miss your newsletters. I do hope to keep in contact with some of you. Perhaps now I can find time to browse in the TI sections of CompuServe or GENIE, and perhaps I will meet you there.

The answer to the challenge in the last Tips? For a clue, try -
DISPLAY AT(24,1):0 in Basic.

Still don't get it? In Basic, DISPLAY is the same as PRINT, but AT is not recognized, so the computer thinks you are telling it to print the variable AT(1,1) - which, being undefined, is 0 - and advance to the next line (the :) and print 0.

I have always wanted a pocket calculator with several memories and a window to display the contents of each one. So, since there is plenty of room for windows on a TV screen, I wrote one.

It does not require any use of the Enter key, but each CALL KEY input must be validated and processed, so don't type too fast. It will accept such inputs as M1=7= or M1=7+1= or M2=1-M1= to put a value in a memory, or 6+7= or 6+M2= to calculate and display, or 6+7M1 or M1-.M2M3 to calculate and put into memory, and will even do multiple calculations such as 1+2-3/4*5%6, subtotaling after the first two.

```
100 CALL CLEAR :: CALL SCREEN(5):: DEF S$(X)=SEG$(A$,X,1)&" = " :: CALL PEEK(8198,A) :: IF A<>170 THEN CALL INIT
110 CALL LOAD(-31886,16):: ON WARNING NEXT :: GOTO 140
120 SET,M$(),K,S,A$,S$(),R,C,N,N1,N2,N1F,N2F,M1F,M,MF,DF,FF,VF,EF,FL,N$,F2,T,M2,MEM(),ST,NX,ZF
130 CALL COLOR :: CALL CHAR :: CALL KEY :: CALL SOUND !@P-
140 FOR SET=0 TO 4 :: CALL COLOR(SET,16,1):: NEXT SET :: FOR SET=5 TO 8 :: CALL COLOR(SET,5,16):: NEXT SET :: CALL CHAR(64,"0")
150 FOR SET=9 TO 12 :: CALL COLOR(SET,16,1):: NEXT SET
160 DISPLAY AT(1,10):"TIGERCUB":"  MULTIMEMORY@CALCULATOR": :"MEMORY #1": :"MEMORY #2": :"MEMORY #3": :"MEMORY #4": :"MEMORY #5"
170 M$(1)="0123456789.+-*/%=CXM" :: M$(2)="0123456789.ASMDPECXM" :: DISPLAY AT(20,1):"use ?":"(1) symbols":"(2) alpha characters"
180 CALL KEY(0,K,S):: IF S=0 OR K<49 OR K>50 THEN 180 :: A$=M$(K-48)
190 DISPLAY AT(20,1):S$(12);"add";TAB(16);S$(16);"percent" :: DISPLAY AT(21,1):S$(13);"subtract";TAB(16);S$(17);"equals"
200 DISPLAY AT(22,1):S$(14);"multiply";TAB(16);S$(18);"cancel" :: DISPLAY AT(23,1):S$(15);"divide by";TAB(16);S$(19);"clear all"
210 DISPLAY AT(24,1):"M1 to M5 = memories #1 to #5"
220 R=15 :: C=1 :: N,N1,N2,N1F,N2F,M1F,M,MF,DF,FF,VF,EF,FL,ZF=0 :: N$="" :: DISPLAY AT(18,1):""
230 CALL KEY(3,K,S):: IF S<1 THEN 230 :: CALL SOUND(50,500,5):: DISPLAY AT(R,C):CHR$(K):: C=C+1
240 ON POS(A$,CHR$(K),1)+1 GOTO 260,270,270,270,270,270,270,270,270,270,280,290,250,290,290,290,340,410,420,430
250 IF VF=1 OR MF=1 THEN 290 :: ZF=1 :: N$="-" :: GOTO 230
260 DISPLAY AT(R,C-1):"?" :: C=C-1 :: GOTO 230
270 IF MF=1 THEN 260 :: FL=0 :: VF=1 :: IF DF=0 AND ZF=0 THEN N=N*10+K-48 :: GOTO 230 ELSE N$=N$&CHR$(K):: GOTO 230
280 IF DF=1 THEN 260 :: DF=1 :: MF,FL=0 :: IF ZF=1 THEN N$=N$&"." :: GOTO 230 ELSE N$=STR$(N)&"." :: GOTO 230
290 IF C=2 OR FL=1 THEN 260 :: FL=1 :: IF FF=0 THEN 320 :: F2=POS(A$,CHR$(K),1)-11 :: IF VF=1 THEN GOSUB 480
310 GOSUB 520 :: N1=T :: DISPLAY AT(18,1):"SUBTOTAL";T :: N2F,N2=0 :: FF=F2 :: GOTO 230
320 IF VF=0 THEN 330 :: VF,MF=0 :: GOSUB 480
330 MF=0 :: FF=POS(A$,CHR$(K),1)-11 :: GOTO 230
340 IF C=2 OR(FF=0 AND M1F=0)OR(C=4 AND M1F=0)OR FL=1 THEN 260
350 IF C=4 THEN EF=1 :: M2=M :: N1F,MF=0 :: GOTO 230
360 IF VF=1 THEN GOSUB 480
370 IF EF=0 THEN 400
380 IF N2F=0 THEN MEM(M2)=N1 :: DISPLAY AT(M2*2+2,11):N1 :: GOTO 220
390 GOSUB 520 :: MEM(M2)=T :: DISPLAY AT(M2*2+2,11):T :: GOTO 220
400 GOSUB 520 :: DISPLAY AT(15,C):T :: GOTO 220
410 DISPLAY AT(R,1):"":"":"" :: GOTO 220
420 MEM(1),MEM(2),MEM(3),MEM(4),MEM(5)=0 :: FOR R=4 TO 12 STEP 2 :: DISPLAY AT(R,10):"" :: NEXT R :: GOTO 410
430 IF EF=1 AND MF=1 THEN 260
440 CALL KEY(3,K,ST):: IF ST<1 OR K<49 OR K>53 THEN 430 ELSE CALL SOUND(50,500,5):: M=K-48 :: DISPLAY AT(R,C):CHR$(K);:: C=C+1 :: MF=1 :: FL=0 :: IF VF=1 THEN GOSUB 480
450 IF N1F=0 THEN M1F,N1F=1 :: N1=MEM(M):: IF ZF=1 OR DF=1 THEN N1=VAL(N$&STR$(N1)):: DF,ZF=0 :: GOTO 230 ELSE 230
460 IF N2F=0 THEN N2F=1 :: N2=MEM(M):: IF ZF=1 OR DF=1 THEN N2=VAL(N$&STR$(N2)):: DF,ZF=0 :: GOTO 230 ELSE 230
470 GOSUB 520 :: MEM(M)=T :: DISPLAY AT(M*2+2,11):T :: GOTO 220
480 IF DF=0 AND ZF=0 THEN NX=N ELSE NX=VAL(N$):: DF,ZF=0
490 IF N1F=0 THEN N1=NX :: N1F=1 :: GOTO 510
500 N2=NX :: N2F=1
510 VF,N=0 :: N$="" :: RETURN
520 IF FF=1 THEN T=N1+N2 ELSE IF FF=2 THEN T=N1-N2 ELSE
```

```
IF FF=3 THEN T=N1*N2 ELSE IF
 FF=4 THEN T=N1/N2 ELSE T=N1
*N2/100
530 RETURN
```

I have always been annoyed by the difficulty of hyphenating with TI-Writer, when I want to avoid the gaping holes that wraparound and Fill and Adjust can cause. Manually filling and adjusting with carets is slow, and leaving a space after the hyphen is unreliable, so I wrote this program.

```
100 DISPLAY AT(2,10)ERASE AL
L:"TIGERCUB":" HYPHENATED F
ILL AND ADJUST"
110 DISPLAY AT(6,1):" Prepar
e text with TI-Writer":"Edit
or. Leave left TAB at 0,":"s
et right TAB at the actual"
:"value of the line length d
e-"
120 DISPLAY AT(10,1):"sired
(i.e., for a 28-char":"lin
e, set it at 28)."
130 DISPLAY AT(12,1):" Inden
t as desired. Center":"hea
dings as desired but be":"
sure to follow them with a
":"line feed (Enter). Hyphen
ate"
140 DISPLAY AT(16,1):"as de
sired and follow the":"hyp
hen immediately with a":"
line feed (Enter)."
150 ON ERROR 160 :: GOTO 170
160 ON ERROR 160 :: RETURN 1
70
170 DISPLAY AT(20,1):"INPUT
FILE? DSK" :: ACCEPT AT(20,1
6)BEEP:F$ :: OPEN #1:"DSK"&F
$,INPUT
180 DISPLAY AT(22,1):"OUTPUT
 FILE? DSK" :: ACCEPT AT(22,
17)BEEP:NF$ :: OPEN #2:"DSK"
&NF$,OUTPUT
190 DISPLAY AT(24,1):"LINE L
ENGTH?" :: ACCEPT AT(24,14)V
ALIDATE(DIGIT):L
200 LF$=CHR$(13):: H$="-"&CH
R$(13)
210 ON ERROR 210 :: GOTO 220
220 ON ERROR 210 :: RETURN 3
10
230 LINPUT #1:M$ :: IF M$="
" OR M$=LF$ OR M$="" OR ASC(
M$)>127 OR(LEN(M$)=L AND POS
(M$,LF$,1)=0)OR POS(M$," ",1
)=0 THEN 310
240 IF POS(M$,LF$,1)<>0 AND
POS(M$,H$,1)=0 THEN 310
250 IF POS(M$,H$,1)<>0 THEN
M$=SEG$(M$,1,LEN(M$)-1)
260 IF LEN(M$)=L THEN 310
270 P=1
280 X=POS(M$," ",P):: IF X=P
 THEN P=P+1 :: GOTO 280 ELSE
 Y,P=X :: IF POS(M$," ",P)=0
 OR P=L THEN 310
290 M$=SEG$(M$,1,X)&" "&SEG$
(M$,X+1,255):: IF LEN(M$)>=L
 THEN 310 ELSE P=X+2
300 X=POS(M$," ",P):: IF X=0
 THEN P=Y :: GOTO 300 ELSE G
OTO 290
310 PRINT #2:M$ :: IF EOF(1)
<>1 THEN 230 ELSE CLOSE #1 :
: CLOSE #2
```

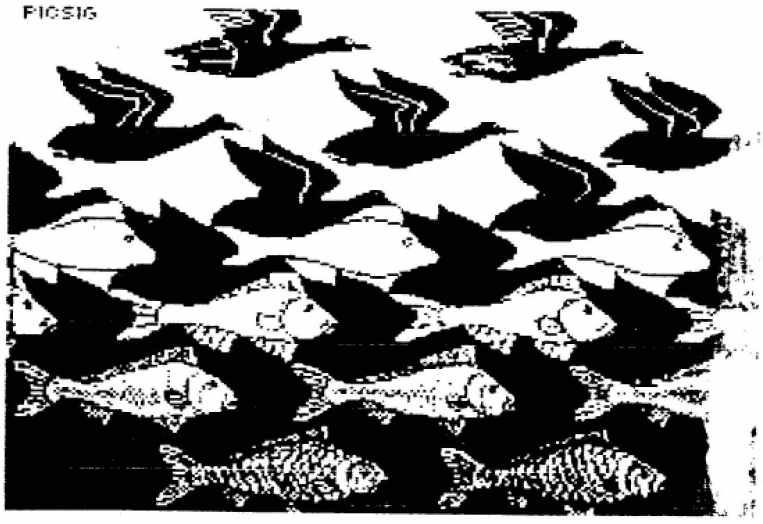
Here is one for the preschoolers -

```
100 CALL CLEAR :: CALL SCREE
N(14):: CALL COLOR(1,11,11,1
2,5,5):: DISPLAY AT(3,10):"S
EE-N-SAY": : :"PRESS ANY KEY
" !by Jim Peterson based on
a routine by Michael Lyons
110 DIM E$(16),PAT$(16):: CA
LL CHAR(123,RPT$("F",16))
120 DATA "    ","   {","  {
","  {{","  {  ","  { {","  {{
","  {{{","{   ","{  {","{ {
","{ {{","{{  ","{{ {","{{{
","{{{{"
130 FOR J=0 TO 15 :: READ PA
T$(J):: NEXT J
140 CALL KEY(0,K,S):: IF S=0
 THEN 140
150 CALL CHARPAT(K,CP$):: FO
R X=1 TO 16 :: Y=ASC(SEG$(CP
$,X,1)):: E$(X)=PAT$(Y+(Y>57
)*7-48):: NEXT X :: IF K>96
AND K<123 THEN K=K-32
160 CALL CLEAR :: CALL SAY(C
HR$(K)):: FOR X=2 TO 16 STEP
 2 :: DISPLAY AT(8+(X/2),12)
:E$(X-1);E$(X):: NEXT X
170 CALL SAY(CHR$(K)):: GOTO
 140
```


Jim Peterson

Copyright 1986

TIGERCUB SOFTWARE
156 Collingwood Ave.
Columbus, OH 43213
USA.



LOTTERY: by Rick Kellogg

HERE IS A LOTTERY PROGRAM THAT IS FULL OF 'BELLS AND WHISTLES'. THE PROGRAM RANDOMIZES AND SORTS ALL OF THE NUMBERS IT PICKS. THERE WILL BE NO DUPLICATION OF NUMBERS IN ANY GIVEN LINE; AND IT WILL GENERATE NUMBERS FOR EITHER THE 40 OR THE 44 NUMBER LOTTERIES. YOU HAVE A CHOICE OF PLAYING FROM 1 TO 10 GAMES AT A TIME. THERE IS A SPEECH OPTION THAT WILL READ OUT THE NUMBERS AS THEY ARE GENERATED, BUT USING THIS OPTION WILL SLOW THE PROGRAM DOWN SOME. THERE IS ALSO A PRINT OPTION THAT WILL PRINT OUT THE SORTED NUMBERS. THE PRINT OPTION IS SET UP FOR EPSON/GEMINI/STAR COMPATIBLE PRINTERS BUT CAN BE EASILY MODIFIED FOR OTHER PRINTERS. IF YOU CHOOSE THE PRINT OPTION, YOU'LL BE ASKED TO INPUT YOUR PRINTER DEVICE (DEFAULT IS 'PIO'). ALSO YOU HAVE THE OPTION OF DATING THE PRINTOUT. HAVE FUN, AND LET ME KNOW IF YOU WIN!

```
100 !********************
110 !*  L O T T E R Y  *
120 !********************
130 !by E.Duty/R.Kellogg
140 !  Extended  Basic
150 !  (speech option)
160 ! Randomize  & Sort
170 !
180 OPTION BASE 1 :: DIM A(6
):: CALL CHAR(48,"0038444C54
644438"):: A$="MM/DD/YY" ::
CALL KEY(3,K,S)
190 CALL CLEAR :: GOSUB 610
:: DISPLAY AT(7,1):"HIGHEST
POSSIBLE NUMBER? 40" :: ACCE
PT AT(7,27)BEEP SIZE(-1)VALI
DATE("04"):B :: IF B=0 THEN
B=40 ELSE IF B=4 THEN B=44
200 DISPLAY AT(10,1):"NUMBER
 OF GAMES TO PLAY? 10" :: AC
CEPT AT(10,26)BEEP SIZE(-2)V
ALIDATE(DIGIT):C :: IF C<1 O
R C>10 THEN 200
210 DISPLAY AT(13,1):"SPEAK
NUMBERS? (Y/N)      N" :: ACC
EPT AT(13,26)VALIDATE("YN")S
IZE(-1)BEEP:B$ :: IF B$="N"
THEN D=1 ELSE IF B$="Y" THEN
 D=0
220 DISPLAY AT(16,1):"DO YOU
 WANT A PRINTOUT?  Y" :: ACC
EPT AT(16,26)BEEP SIZE(-1)VA
LIDATE("YN"):C$ :: IF C$="Y"
 THEN DISPLAY AT(18,1):"DEVI
CE: PIO" :: ACCEPT AT(18,9)B
EEP SIZE(-19):D$
230 IF C$="N" THEN 250 :: ON
 ERROR 220 :: OPEN #1:D$
240 DISPLAY AT(21,1):"DATE O
N PRINTOUT?         Y" :: ACC
EPT AT(21,26)BEEP SIZE(-1)VA
LIDATE("YN"):E$ :: IF E$="Y"
 THEN DISPLAY AT(23,1):"(MM/
DD/YY)          ";A$ :: ACCEPT
 AT(23,19)BEEP SIZE(-8)VALID
ATE(DIGIT,"/"):A$
250 CALL CLEAR :: GOSUB 610
:: FOR E=3 TO 21 STEP 2 :: C
ALL HCHAR(E,3,F+1):: F=F+1 :
: NEXT E :: G=3 :: H=1
260 I=1
270 DISPLAY AT(24,6):"R A N
D O M I Z E" :: RANDOMIZE ::
 A(I)=(INT(B*RND)+1):: F$=ST
R$(A(I)):: ON I GOTO 330,280
,290,300,310,320
280 IF A(2)=A(1)THEN 270 ::
GOTO 330
290 IF A(3)=A(2)OR A(3)=A(1)
THEN 270 :: GOTO 330
300 IF A(4)=A(3)OR A(4)=A(2)
OR A(4)=A(1)THEN 270 :: GOTO
 330
310 IF A(5)=A(4)OR A(5)=A(3)
OR A(5)=A(2)OR A(5)=A(1)THEN
 270 :: GOTO 330
320 IF A(6)=A(5)OR A(6)=A(4)
OR A(6)=A(3)OR A(6)=A(2)OR A
(6)=A(1)THEN 270
330 IF D=1 THEN 470 :: IF A(
I)<10 THEN CALL SAY(F$):: GO
TO 470
340 IF A(I)=10 THEN CALL SAY
("TEN"):: GOTO 470
350 IF A(I)=11 THEN CALL SAY
("ELEVEN"):: GOTO 470
360 IF A(I)=12 THEN CALL SAY
("TWELVE"):: GOTO 470
370 IF A(I)=13 THEN CALL SAY
("THIRTEEN"):: GOTO 470
380 IF A(I)=14 THEN CALL SAY
("FOURTEEN"):: GOTO 470
390 IF A(I)=15 THEN CALL SAY
("FIFTEEN"):: GOTO 470
400 IF A(I)>15 AND A(I)<20 T
HEN F$=STR$((A(I)-10)):: CAL
L SAY(F$,,"TEEN"):: GOTO 470
410 IF A(I)=20 THEN CALL SAY
("TWENTY"):: GOTO 470
420 IF A(I)>20 AND A(I)<30 T
HEN F$=STR$((A(I)-20)):: CAL
L SAY("TWENTY",,F$):: GOTO 4
70
```

```
430 IF A(I)=30 THEN CALL SAY
("THIRTY"):: GOTO 470
440 IF A(I)>30 AND A(I)<40 T
HEN F$=STR$((A(I)-30)):: CAL
L SAY("THIRTY",,F$):: GOTO 4
70
450 IF A(I)=40 THEN CALL SAY
("FORTY"):: GOTO 470
460 IF A(I)>40 AND A(I)<45 T
HEN F$=STR$((A(I)-40)):: CAL
L SAY("FORTY",,F$)
470 DISPLAY AT(G,I*4):A(I):
 FOR J=1 TO 150 :: NEXT J ::
 I=I+1 :: IF I<7 THEN 270
480 I=6 :: CALL HCHAR(24,1,3
2,32):: DISPLAY AT(24,8)BEEP
:"S O R T I N G"
490 K=A(1):: L=1 :: FOR M=2
TO I :: IF A(M)<K THEN 510
500 K=A(M):: L=M
510 NEXT M :: N=A(I):: A(I)=
A(L):: A(L)=N :: I=I-1 :: IF
 I>1 THEN 490
520 M=1
530 DISPLAY AT(G,M*4):A(M):
 IF M=6 THEN 540 ELSE M=M+1
:: GOTO 530
540 FOR J=1 TO 100 :: NEXT J
 :: CALL HCHAR(24,1,32,32):
 IF D=0 THEN 560
550 FOR J=1 TO 200 :: NEXT J
560 G=G+2 :: IF H<C THEN H=H
+1 :: GOTO 260 ELSE 570
570 IF C$="Y" THEN GOSUB 620
580 DISPLAY AT(23,6)BEEP:"Pr
ess 'M' for More":"      Pres
s 'Q' to Quit"
590 CALL KEY(0,O,P):: IF P=0
 THEN 590
600 IF O=77 THEN 190 :: IF O
=81 THEN 660 ELSE 580
610 F=96 :: CALL CHAR(95,"00
FF"):: DISPLAY AT(1,3):" $ $
 WEEKLY LOTTERY $ $":"
_____" :: RETURN
620 DISPLAY AT(24,7):"P R I
N T I N G" :: PRINT #1:"
$ $ WEEKLY LOTTERY $ $" :: I
F A$="MM/DD/YY" OR E$="N" TH
EN 640
630 IF E$="Y" THEN PRINT #1:
TAB(12);A$:"  ------------
------------" :: GOTO 650
640 PRINT #1:"  ------------
--------------"
650 FOR Q=3 TO (C*2)+1 STEP
2 :: FOR R=1 TO 32 :: CALL G
CHAR(Q,R,S):: PRINT #1:CHR$(
S);:: NEXT R :: PRINT #1:""
:: PRINT #1:"" :: NEXT Q ::
PRINT #1:CHR$(27);"a";CHR$(3
5):: CLOSE #1 :: RETURN
660 CALL CLEAR :: END
```

```
$ $ WEEKLY LOTTERY $ $
        10/17/86
-------------------------
a   9   13  38  40  43  44
b   5   7   9   25  26  35
c   6   10  12  14  24  29
d   10  14  15  21  27  43
e   3   4   7   9   18  38
f   2   6   20  21  32  43
g   5   13  16  20  28  31
h   5   15  16  25  26  34
i   3   5   15  16  21  42
j   13  14  17  23  31  35
```

RICK'S COLLECTIONS: by Rick Kellogg

HERE ARE SOME ITEMS OF INTER EST THAT I HAVE COLLECTED OVER THE PAST SEVERAL YEARS. SOME I HAVE CREATED, SOME ARE BORROWED FROM SEVERAL SOURCES.

SPECIAL SCREEN CHARACTER CODES:

Slashed Zero	CALL CHAR(48,"0038444C54644438")	Ø
Right Arrow	CALL CHAR(? ,"000804027F020408")	→
Left Arrow	CALL CHAR(? ,"00102040FE402010")	←
Up Arrow	CALL CHAR(? ,"081C2A4908080800")	↑
Down Arrow	CALL CHAR(? ,"000808084 92A1C08")	↓
Solid Line	CALL CHAR(95,"00FF")	—
Copyright Symbol	CALL CHAR(? ,"003E415D515D413E")	©
PI Symbol	CALL CHAR(? ,"0000FE2828282828")	π
Cent Sign	CALL CHAR(? ,"00083C4848483C08")	¢
Check Mark	CALL CHAR(? ,"0002020404482810")	✓

PROMPT 'BEEP' CALL SOUND(150,1390,2)
PROMPT 'HONK' CALL SOUND(70,218,1)

CONVERSIONS DEF LEFT$(X$,Y)=SEG$(X$,1,Y)
Microsoft Basic DEF MID$(X$,Y,Z)=SEG$(X$,Y,Z) - Conversion not needed.
to TI Basic DEF RIGHT$(X$,Y)=SEG$(X$,LEN(X$)-Y+1,Y)
(From the Chicago TI User Group newsletter, 'The Chicago TImes')

TO PAUSE DURING "PRINT" or "DISPLAY"

```
6000 CALL KEY(3,K,S)
6010 IF S=0 THEN 6050
6020 CALL KEY(3,K,S)
6030 IF S=0 THEN 6020
6040 IF S=-1 THEN 6020
6050 RETURN
```

Use a GOSUB 6000 after every PRINT or DISPLAY statement that you want to be able to Pause in.
Any key pressed will cause a Pause.
Press again to resume the PRINT statement.

DISK CAUTION LABELS: by Rick Kellogg

HERE IS A SHORT ROUTINE THAT CAN BE USED TO CREATE A CAUTION LABEL FOR ANY DISKETTES THAT YOU PLAN TO SEND THROUGH THE MAIL. THIS WILL PRINT OUT ON A STANDARD 15/16" x 3 1/2" PRESSURE SENSITIVE LABEL THAT YOU CAN EASILY APPLY TO THE DISKETTE MAILER. THIS PROGRAM IS DESIGNED TO RUN OUT OF BASIC OR EXTENDED BASIC. IT IS ALSO SET UP FOR EPSON/GEMINI/STAR COMPATABLE PRINTERS, BUT COULD BE EASILY USED ON OTHER PRINTERS BY INSERTING THE PROPER PRINTER CONTROL CODES WHERE NEEDED.

```
100 REM ******************
110 REM *  DISK CAUTION  *
120 REM *     LABELS     *
130 REM *       by       *
140 REM *  Rick Kellogg  *
150 REM *  Epson/Gemini  *
160 REM *   compatable   *
170 REM * Basic & XBasic *
180 REM ******************
190 REM
200 CALL CLEAR
210 INPUT "How many labels?
":A
220 OPEN #1:"PIO"
230 PRINT #1:CHR$(27);"G";
240 FOR X=1 TO A
250 PRINT #1:CHR$(27);"-1";C
HR$(14);"DO NOT BEND!"
260 PRINT #1:""
270 PRINT #1:CHR$(27);"-0";"
MAGNETIC MEDIA ENCLOSED"
280 PRINT #1:"_____
_____"
290 PRINT #1:CHR$(27);CHR$(1
4);"DO NOT XRAY!"
300 PRINT #1:""
310 NEXT X
320 CLOSE #1
330 END
```

DO NOT BEND!

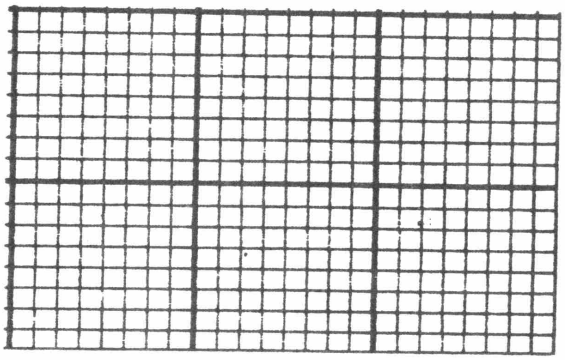
MAGNETIC MEDIA ENCLOSED

DO NOT XRAY!

GRAPHSHEET PROGRAM: by Rick Kellogg

HERE IS A WONDERFUL PROGRAM THAT I FOUND IN THE CHICAGO TI USER GROUP'S NEWSLETTER, 'The Chicago TImes'. IT WAS WRITTEN BY JOHN BEHNKE AND RUNS IN EITHER BASIC OR EXTENDED BASIC. IT IS DESIGNED FOR EPSON/GEMINI/STAR COMPATABLE PRINTERS. WHAT IT DOES IS PRINT OUT A FULL 8 1/2" x 11" SHEET WITH A 8 x 8 GRID THAT YOU CAN USE FOR CHARACTER AND/OR SCREEN DESIGNS. NOW CREATING YOUR GRAPHICS AND CONVERTING THEM TO HEX WILL BE MUCH EASIER. THE FULL SIZE SHEET WILL GIVE YOU PLENTY OF ROOM TO CREATE SEVERAL DESIGNS.

```
100 REM ********************
110 REM * GRAPHSHEET MAKER *
120 REM *  by John Behnke  *
130 REM * Chicago TI U.G.  *
140 REM * Epson or Gemini  *
150 REM * Printer required *
160 REM * Basic or X-Basic *
170 REM ********************
180 CALL CLEAR
190 INPUT "Number of sheets?
 ":A
200 CALL SCREEN(2)
210 @$=CHR$(27)
220 FOR I=1 TO 228
230 A$=A$&CHR$(128)
240 NEXT I
250 B$=SEG$(A$,1,7)
260 C$=CHR$(255)&SEG$(A$,1,6
)
270 FOR I=1 TO 4
280 FOR J=1 TO 8
290 E$=E$&C$
300 NEXT J
310 E$=E$&CHR$(255)
320 NEXT I
330 F$=@$&"K"&CHR$(484)&CHR$
(0)&E$
340 G$=@$&"K"&CHR$(228)&CHR$
(0)&A$
350 OPEN #1:"PIO.CR"
360 FOR B=1 TO A
370 FOR C=1 TO 11
380 PRINT #1:@$&CHR$(64)&@$&
"3"&CHR$(16)
390 FOR D=1 TO 8
400 PRINT #1:F$;F$;CHR$(10)
410 NEXT D
420 PRINT #1:G$;G$;@$&"3"&CH
R$(2)
430 NEXT C
440 PRINT #1:@$&"3"&CHR$(17)
450 FOR I=1 TO 9
460 PRINT #1:CHR$(13)&CHR$(1
0)
470 NEXT I
480 NEXT B
490 CLOSE #1
500 END
```



THE "CIN-DAY NEWS", IS PUBLISHED THROUGH THE EFFORTS OF THE:

CIN-DAY USER GROUP

PIQUA, OHIO

Howard Greenberg .....yawn!

Well, he's back. I hear all sorts of rumours about me, from a nervous breakdown to bankruptcy and even of my death. Well, I'm alive and well and still in business.

I'd missed the last couple of issues due to an assortment of reasons, but now have enough to say to fill the page.

First of all, what of the Myarc computer so heavily featured in earlier issues this year? Well, surprise it still isn't available. Now Myarc have made some excellent products to date, but it strikes me that their ambitions have outstriped their ability. But what sticks in my craw is never being given a straight story by them. The range of reasons and excuses for non-delivery of product varies from the credible to the incredulous and I have to say that until they get their act together regarding reliable information and delivery dates, then my love affair with them is at an end. This does not mean though that I won't be able to obtain Myarc products. I currently have all of their cards in hand and can obtain the rest through a wholesaler. This may cut my margins, but will be more reliable.

The idea and concept of the new computer has changed. From the original idea of a whole new console, it has changed to what it should have been in the first place. A card. I saw one of these a few weeks ago in the U.S.A. alhtough it was in prototype form and didn't work. Whether this ever makes it to production is something we'll have to wait and see.

There is though more news on the idea of a new computer. First of all though, I know that the cost of expansion at all is so horrific that many owners are now looking to other brands. Now if you're determined to follow this route, then please talk to Harry Pridmore or myself if you'd like to stick with retailers you know and trust. I know we've both built up good relationships with the people who keep us solvent, so it would be a shame to lose that just because no-one knows we also handle Mr. Sugars products as well. But back to the T.I. For those who have a fully expanded system, the idea of changing is appealing, but when weighed against the idea of acquiring not only equivalent hardware, but the software base as well for an alternative machine, then that idea too palls. All too often, the choice of alternative computer is either an IBM or an IBMulator.

Whilst in the States (during November) I put it to one company that it might be an idea to use that very tough peripheral box as the starting point for a conversion, with the slots being used to add various IBM style cards, and an IBM keyboard connector at the back. It seems the idea wasn't original, but that the problem was the bus, which wasn't compatible. It seems someone has thought up all the ideas and come up with a solution. Now I haven't seen it yet so I can't say how well it works, but I can vouch for the companys integrity and quality of workmanship. So when this comes through, it's likely that we will finally have a realistic method of using IBM software and hardware, without the hideous cost of a total conversion. To quote the blurb I've received so far:

This unit could be thought of as an IBM expansion system as it will not only allow you to use IBM software but you can also add IBM cards to the system! Now you will be able to run things like Lotus 123, dBase, Microsoft Flight Simulator etc. In January 1987, this company will make their announcement as to price and availability.

The only catch is, although it's being developed by one company, it's for release and marketing by another. I've not been given any names, but if it's Myarc, then as I've said, they'll have to get their act together before I spend another cent directly with them.

What else have I seen or heard about? Well one trick is something I approved of in everything but price. The IBM keyboard is probably the best unit around with its' numerous functions and separate curser and other keys that we all find take a combination of presses for us to achieve. Someone has finally come up with a way of attaching the PC keyboard to the TI99/4A. (And I don't know who) The earlier version of this idea was originally conceived as a card interface to go in the box. The version I saw was easier to design, but harder to fit as this one just replaced the existing keyboard with a blank panel, with a connector for the PC keyboard. The catch as I said is cost. Although there are numerous cheap keyboards around, the minimum cost of one of these seems to be around the £50.00 mark. Then there's the cost of the interface to be added as well, and that's where the designer earns his money, so it's unlikely to come in at under £75.00. Shame really, 'cos it's a nice idea and one that this typist would choose for himeslf.

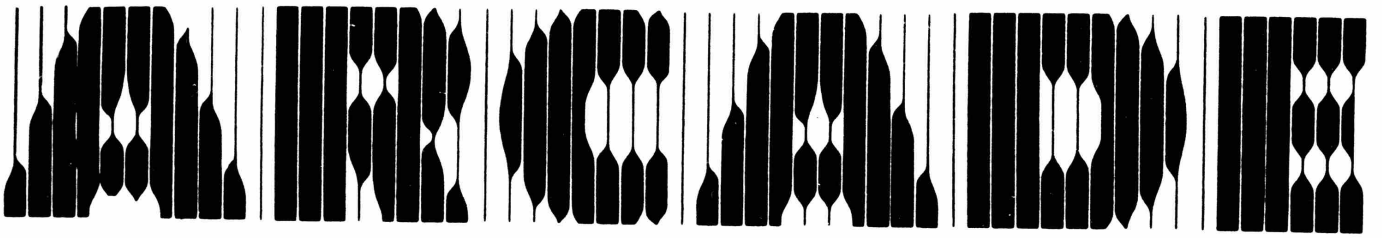
I didn't actually purchase much whilst in the U.S. of A, but did manage to pick up a few items that until recently had been on my out of stock list. Amongst them are; Navarone Database and Widgit, Centronics stand alone interface and Hitch Hikers guide to the Galaxy. This last item was the only Infocom that I not only originally sold most of, but was the first to run out of stock.

There seems to be a problem with Compute! publications. Because of the agreement with the British distributors, I can't get these books direct from the States, and because of the TI99s lack of popularity in the U.K. the British distributors won't stock the books any longer. Catch 22.

Time to close, so I'll wish everyone a happy new year and hope you've all recovered from the excesses of the festive season.

Howard

Howard.

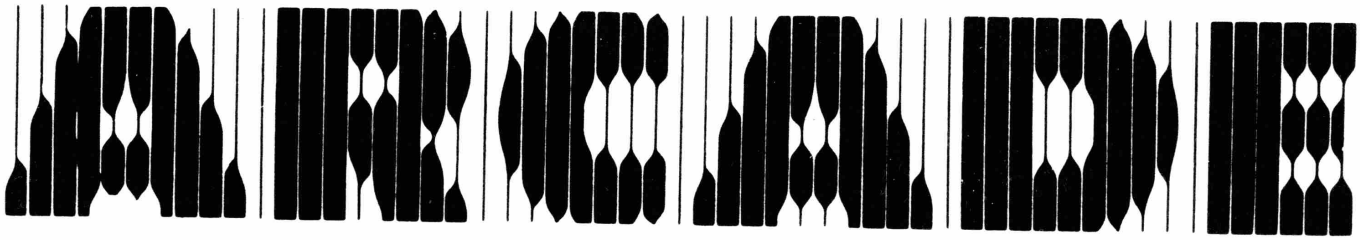


HARDWARE

214 Horton Road, Fallowfield, Manchester M14 7QE. Tel: 061-225 2248

UTILITY PROGRAMS (disc based)	
NAVARONE DATABASE module/disc	£65.00
EXPLORER	£24.95
ADVANCED DIAGNOSTICS	£19.95
DISKASSEMBLER	£19.95
MULTIPLAN	£24.95
HARDWARE	
MYARC RS232 CARD	£125.00
MYARC DISC CONTROL CARD	£185.00
MYARC 128k CARD	£230.00
CENTRONICS stand alone	£ 75.00
COMPETITION PRO JOYSTICK	£ 19.25
SINGLE SIDED INTERNAL DRIVE	£125.00
SINGLE SIDED EXTERNAL DRIVE	£145.00
DOUBLE SIDED INTERNAL DRIVE	£145.00
DOUBLE SIDED EXTERNAL DRIVE	£175.00
SHINWA CPA 80+ MATRIX PRINTER	£230.00
SPECIAL EX-DEMO JUKI 2200 DAISYWHEEL TYPWRITER/PRINTER	£199.00
NAVARONE WIDGIT	£ 35.00
NIGHTINGALE MODEM	£145.00
CASSETTE CABLE	£ 7.50
PARALLEL CABLE	£ 18.00
CASSETTE 'N' GAME FILE	£ 22.50

HAPPY NEW YEAR TO ALL OUR READERS!!!



HARDWARE

211 Horton Road, Fallowfield, Manchester M14 7QE. Tel: 061-225 2248

GAMES MODULES

Frogger (requires joysticks)	£10.00
T.I. Invaders	£5.00
Munchman	£5.00
Car Wars	£5.00
Tombstone City	£5.00
Chisholm Trail	£5.00
Anteater	£19.95
King of the Castle	£14.95

EDUCATIONAL MODULES

Frog Jump	£5.00
Star Maze	£5.00
Pyramid Puzzler	£5.00
Space Journey	£5.00
Number Readiness	£5.00
Dragon Mix	£5.00
Minus Mission	£5.00
Early Learning Fun	£5.00

ADVENTURE TAPES (For adventure module)

Mystery Fun House	£3.50
Golden Voyage	£3.50
Voodoo Castle	£3.50
Ghost Town	£3.50
Strange Odyssey	£3.50
Adventureland	£3.50

ADVENTURE DISCS (32k + Drive + Controller)

ZORK II	£25.00
ZORK III	£25.00
SORCERER	£25.00
DEADLINE	£25.00
WITNESS	£25.00
INFIDEL	£25.00
SUSPENDED	£25.00
STARCROSS	£25.00
HITCH HIKERS GUIDE TO THE GALAXY	£25.00
OLD DARK CAVES II (not Infocom)	£19.95

The Classified advertisements Page.

LOW PRICE MODULES
TELEPHONE FOR AVAILABILITY BEFORE ORDERING 0273 503968.....
MUSICMAKER £19.95 HOME BUDGET MANAGER £9.95 ALPINER £9.95 CHISHOLM TRAIL £4.99. BEGIN GRAMMER £2.95. MINIMEM £34.95 TI INVADERS £4.95. ADVENTURE/PIRATE £10.95. FROGGER £7.95. MUNCHMAN £2.95. EARLY LEARNING FUN £3.95. ADDITION/SUBTRACTION 1. £4.95. VIDEO GAMES 1 £4.95. >>>>>>>>>> TELEPHONE 0273 503968 to CHECK STOCK AVAILABLE, at these MAD prices stock will soon be exhausted.

TV AERIAL SPLITTER (SWITCHED) will prevent wear on TV socket and cable. Simply flick a switch to £2.25 +25 P+P.
Want a spare TI99/4a console? We have a number at £45.00 each
The best book for TI99/4a CONSOLE ONLY users. KIDS AND THE TI99/4a - Datamost...ARE YOU FED UP WITH THE LACK OF BASIC PROGRAMS IN TI*MES ?? We have the answer. Filled with EASY to follow TI BASIC programs and drawings to help you under- stand programming the material so you'll be writing your own programs by Lesson 5 and forever after. Learn to write computer music, create computer graphics and drawings make your computer ask questions create games and quizzes. Edit and debug programs fast organize your mind and think creatively and best of all it's fun. ITs NOT JUST FOR KIDS! 236 pages and spiral bound. £6.00 (only a few remaining)
SALE SALE SALE SALE.
GETTING STARTED WITH THE TI99/4A by Stephen Shaw.WAS£4.50 CLEAR OUT PRICE £2.50..
Send Order & payment to CHS, 40 BARRHILL PATCHAM BRIGHTON BN1 8UF.
PLEASE ALLOW 28 DAYS TO PROCESS
****
FOR SALE.
UCSD P-CODE SYSTEM & 5/4" 96 TPI DSDD disks in plastic Library case £12.00 for 10 +£1.50 PP Tel 0672 54975.
MYARC GENEVE for latest information tel 0672 54975.
****
TI*MES BACK ISSUES. The following issues ONLY are available *** 4,8,9,10, 11, & 12 cost £1 each to members. Write to TI99/4A EXCHANGE, BRIGHTON.

Wanted urgently TI RS232 and Printer for PE Box. Telephone Helensburg 71967

MINIMEM Conversion to Rechargeable Battery. Send Minimem and Crossed cheque £7.50, To- N.J.Petry, Tensal Technology, 15 PENRICE CLOSE, WORLE, W.S.M., Avon, BS22 9AH.

****
FOR SALE Full expansion system for TI99/4a Computer, speech sythesiser, joysticks, cassette lead, ext basic, parsec, video games, maths learning modules, programming books, £400.00 ono - telephone 0273 503968. after 8pm or weekends.
****
TI99/4a clearout..
Used modules ADVENTURE, PRK, SOCCER, TUNNELS OF DOOM, HOUSEHOLD BUDGET, ATTACK, NUMBER MAGIC, AMAZING, VIDEO GAMES. Make an offer 0273 503968.
****
WANTED Display enhancement cassette (Maple Leaf) for use with Minimem. Also blown orr damaged minimem module. Contact James at 36 Harrison Cres, Bedworth, Warks. CV128SL. Tel 0203 315947.
SWAP your 40 track Disk drive for my 80 track CUMANA Disk drive in first class condition. Tel PAGHAM 266188.
WANTED YOUR TI SYSTEMS. Best prices given. Contact Harry Pridmore for a great trade in deal. 0404 41856.
****
CLEAR OUT TIME *** used MODULES sale WITHOUT DOCUMENTATION... PAC MAN £3. BUCK ROGERS £3. PRK £4. PRG (for PRK) £3. TI Writer module without disk & manual £3. TI writer disk £4. Zero Zap £2. Navercne Disk fixer £4. STARTREK £3. SNEGGIT £3. MINER 2049ER £4. Autographed Getting started with the TI99/4a. £2. **TI99/4a CONSOLE £6!! Buyer collects. LIMITED SUPPLY PHONE between 9 & 10pm 061 432 6097.
****
PHOTOGRAPHIC BUSINESS CARDS.... just send a colour negative and your present printed card, we will produce a high quality Photocard overprinted with your company name and details. Cost only £60 for 500 cards or £95 for a 1000. We use only the best Kodak quality materials. ORDER ENQUIRIES 0273 503968 Access welcome.
****

ADVERTISE ON THIS PAGE FOR 5P A WORD. (MEMBERS ONLY) SWAP SHOP ADS FREE!Trade Advertisements Rates on application.

FURTHER DEVELOPMENTS TO THE FAST LOADER

By E.H.Shaw

Following on from the article in the last issue of TI*MES by Art Green I have spent some time developing the suggestions made for "hiding" an assembly routine inside an extended basic program running in expansion memory. By doing this the whole structure can be loaded much faster than by the conventional 'OLD DSK1.XXX' followed by 'CALL LOAD("DSK1.YYY")'.

The method in its current state requires MMM and Extended Basic and uses the assembly language relocating program in Fig 1.The exercise is carried out in three stages. In stage 1 using the MMM module the program is loaded into TI Basic, the relocating program is run and the program is resaved. A gap of 4k is created at the top end of VDP ram and it is here that the assembler routine can later be loaded. In stage 2 using extended basic the Xbas program is loaded first, then the assembler routine (which must be written in absolute code i.e. AORG >F000) and the program resaved. Finally stage three consists of peeking the Ref/Def table in low memory and adding this to the program in the form of CALL LOAD statements.

STAGE 1 (MMM)
CALL INIT
CALL LOAD("DSK1.RELOC") Source code in Fig 1
OLD DSK1.XXX The Ex Bas program
CALL LINK("RELOC")
SAVE DSK1.XXX Program is saved with 4K of free space above it.

STAGE 2 (XBAS)
CALL INIT
OLD DSK1.XXX
CALL LOAD("DSK1.YYY") Assem code starting from >F000.
SAVE DSK1.XXX

STAGE 3 (XBAS)
In immediate mode type
CALL LOAD(16376,A,B,C,D,E,F,G,H) and write down the values.
Then add these two statements to the Exbas program:
For one entry in ref/def table
10 CALL LOAD(16376,A,B,C,D,E,F,G,H) Where A-F are prog name G,H are the entry point.
20 CALL LOAD(8196,63,248) Last free address in low mem

Save the whole lot back to disc and there you are. This allows for a maximum Basic program size of 8K and assembly program size of 4K which is near the limit of VDP ram but leaves plenty of space in the 24K part of the expansion Ram. Now can anyone extend this even further so that the MMM module is not required?

FIG 1.

Source Code for relocating program.

```
        DEF RELOC
VMBW    EQU >6028
VMBR    EQU >6030
RELOC   LI R3,>8370     Highest VDP pointer
        LI R4,>8330     Line No
        LI R5,>8332     Table pointers.
        MOV *R3,R2
        MOV *R4,R0
        S R0,R2
        MOV *R4,R0
        LI R1,BUFFER
        BLWP @VMBR
        AI R0,-4096     4K )
        BLWP @VMBW          )This value can
        LI R0,4096      4K ) be changed
        S R0,*R4
        S R0,*R5
        RT
BUFFER  DATA 0          This buffer allows
        END             for the Xbas program
                        to be up to 8K.
```

RAMBLES

RAMBLES. by Stephen Shaw

Hello again. Huge mountain of material to cover... so much that I will offer the "surplus" as a Rambles Supplement Disk! Material suitable for owners without disk drives will take precedence in this issue... watch out for details of the disk later on!!!

Thanks to a little assistance from Clive, and not too much from British Rail, who contrived to take four hours for a three hour train journey, I paid a brief visit to the PCW show this year - how BORING! The only manufacturer in the home computer market seems to be Amstrad (if Commodore were there I missed them!) and there was huge interest in their stand, displaying the new PC512, while just across the aisle Acord(/BBC) were quite deserted....
I dont wish to criticise Amstrad, but their machines ARE "old tech", and the lack of anything really exciting left me feeling very happy with my present ancient set up - certainly I did not see any music or speech to compare to my present capabilities.
Did you go to PCW? I managed to find a stall with THREE TI99/4A modules on sale at low low prices - three different Atari modules.
Somehow I also managed to win a new watch for Cathy in one of the free draws - handy as she needed one!

The International 99/4 Users Group has now been finally wound up (no connection with Peter Brooks, whose choice of name for his group is a trifle unfortunate!). I speak of the original TI user group which was based in Bethany, Oklahoma, publishers of 99 ENTHUSIAST.
The IUG had net assets of US$17,865.92, and after preferential creditors of US$12,441.35, that left just US$5424.57 to meet the winding up costs and non-preferential creditors totalling US$79,390.55 The proprietor, Charles LaFara, has met with some lack of popularity recently, as he latterly sought to maintain the Groups income by claiming copyright on the public domain programs in the IUG library - and even took action against one unfortunate US User Group. A long way from the early days when the first UK User Group was established, and Charles sent over a few disks of programs to start the new Groups library off. I' remember the good times.

The Disk Library is still going (just!) and welcomes support! Over one hundred disks of assorted software. Some recent-ish additions include a new Assembler by Art Green (Rag-Mac), a Music Pre-Processor by Norm Sellers, and a Fractal Explorer by Steve Langouth. Why not send for a complete up to date listing? Just send a blank disk with return postage to me at:
10 Alstone Road, STOCKPORT, Cheshire, SK4 5AH.
Donations of programs (and graphics in any format!) are most welcome - I also have a number of RLE picture files, and TWO RLE file readers!

Received just as last Rambles was going to press was Myarc XB Version 2.1, which more than pleases me! A priority item in this issue is a long Benchmark article, so if you dont see a full review of Myarc XB anywhere, its available on the Supplement disk!(Vn 2.11 now in!)
Actually, receipt of the new version was a bit of a miracle, as Myarc decided to move Stockport to CANADA... and paid postage only to Canada, and did not make any Customs declaration. In Canada, the Canadian Postal Service thought the SK4 post code could refer to SASK.... and in Regina, Sask, some clever person traced Stockport and forwarded the package on. Fortunately Canada did not make any postal surcharge, and the UK customs failed to confiscate the non-declared package. Close call eh! But well worth having...

Also too late for last Rambles was the second issue of 4FRONT from New Dav (you dont subscribe yet??? Aaaagh...). The disk version contained the continuation of Graham Marshalls superb machine code tutorial/bit map graphics utility. Very very good, and many thanks to Graham. The extended basic game in Vol 2 was pretty snazzy too...
Now it doesn't cause problems, but for reference, you may wish to note that the use of ON ERROR in that game is wrong.
This is wrong:

```
100 ON ERROR 200
110 CALL SCREEN(-1000)
120 END
200 ON ERROR 200
210 PRINT "ERROR"
220 IF A=5 THEN 100
230 IF A=8 THEN 110
240 GOTO 120
250 END
```

This is how it SHOULD be done:

```
100 ON ERROR 200
110 CALL SCREEN(-1000)
120 END
200 ON ERROR 200
210 PRINT "ERROR"
220 IF A=5 THEN RETURN 100
230 IF A=8 THEN RETURN 110
240 RETURN 120
250 END
```

It may not look right but that is what TI ExBas expects you to do!
Check the definition of RETURN in your ExBas Manual. The options available to you are: RETURN, RETURN NNN, and RETURN NEXT.

TRIVIA TIME:
MICROpendium claims copyright but fails to give the information required by US copyright law. -Can you spot a (c) symbol? The publisher is quoted as JOHN KOLOEN but cheques are issued by BURNS-KOLOEN COMMUNICATIONS INC.

++++++++++++++++++++++++++++++++++++

TI ARTIST- mentioned here several times! - has gone through a very slight revision and is now into VERSION 2.01 - the changes are: a. The disk is now easily copied and configured for Myarc Ram Card, Myarc Hard Disk and New Horizon Ram Card. Removal of the image and image colours is now available through the ENHANCEMENT environment- you dont have to go back to the main TI ARTIST menu.

NOW IN: a major machine code game in the FREEWARE category. MUNCH MAN was fast- PAC MAN was a doze... so we now have BUZZARD BAIT with a fast screen set up, many different mazes, and four buzzards after the animated running man, who has only the odd limited fuel flame thrower to protect himself from them! Not very difficult, very nice graphics, and not too slow. Just write and ask for BUZZARD BAIT - some other XB games will be sent along with it- dont know what at time of writing! Usual copying fee: Send a blank disk and the total cost is two pounds (1 for copying plus 1 for handling order) or just send four pounds and a disk and packaging are supplied by me!

::::::::::::::::::::::::::::::::

You are tuned to RAMBLES by Stephen Shaw of
10 Alstone Road, STOCKPORT, Cheshire
ENGLAND, SK4 5AH

ISSUE 14 ERRORS:
Bug time again....
PAGE 7: C SOURCE CODE: The dreaded formatter strikes again...
Text should read:

```
int row,col;
main()
{ while(1)
```

main() must be on a line to itself. Please correct your copy of issue 14 now!

PAGE 29: TI TIPS : No 1. CALL LOAD(-31878,N)
This tip is only valid if you are that very rare bird, an owner of TI Extended Basic version 100. This tip is THAT old! Now we have Version 110 (and later) which automatically update this address, the tip is still appearing! You can safely ignore it if your CALL VERSION gives a number higher than 100!!

Moral: Before spreading tips around, and around, and around... if you can't follow what they are doing, try them! The tip as described here is in any case false: the number N should not be the NUMBER OF SPRITES YOU ARE USING - rather, it should be THE HIGHEST NUMBERED SPRITE YOU ARE USING ... not always the same thing... but YOU can ignore the tip completely anyway...

4) CALL LOAD(-31962,255) MAY work on somebody's console, but on mine all it does is return to the title screen with the wrong character set loaded. Please DO NOT use this tip in your programs (IF IT WORKS ON YOUR SYSTEM!!) as it doesn't work on all of them. (Has anyone made this tip work as described?)

TI WRITER TIP. Maximum disk file length 92 sectors eh? I have two TI Writer files here longer than that - one of them is 107 disk sectors long! The tip is not totally wrong, but assumes a certain average line length- the more blanks you have at the end of your lines, on average, the more disk sectors you can fill before you get BUFFER FULL.

ERROR CHECK: Good idea hiding here but it wont work quite like this! What you do is, when you get the error message DATA ERROR IN LINE XXX, LIST that line to the screen. It says something like READ A$. STILL IN COMMAND MODE you now type in PRINT A$ and it will print the last data item read in. See the difference between this advise and the tip as printed! Try them both... here is a sample program to give you an error message:

```
100 FOR T=1 TO 7
110 READ A$
120 NEXT T
130 STOP
140 DATA ONE,TWO,THREE,FOUR
150 DATA FIVE,LAST!
```

(Mistake is DELIBERATE!!!).
Run this and try MY hint above, then try using the hint on page 29 of issue 14.

MULTIPLAN TIP:
This applies to ALL keyboard usage, including TI Writer. If you keep the main key held down, you can release SHIFT, FCTN or CTRL once the character starts repeating.

==========

Graham Marshall has reminded me of a bug in TI Writer files, as issued with FUNLWRITER - and I have seen a comment somewhere that it is a bug in Funlwriter - which is not true... the bug can be found if you try RE (RecoverEdit) just after entering the editor- eg with truly empty buffers. This bug was introduced in the #fix1 file released to public domain by TI.

//////////////////////////

Checking out >83C4 when using Myarc XB seems to indicate that an interrupt routine is in use - and with only one tag available, the use of that tag by the system rather limits the use of interrupts! The values placed there are (decimal) 117 and 116, and although Myarc XB lets you CALL LOAD nil values into the address, it immediately rewrites 117 and 116 back!

==========

Tony McGovern tells me that the TI and CorComp disk controller cards do not use the index hole in the disk jacket for READS but that the Myarc card does. Hows that for trivia...

==========

Will McGovern has written a couple of disk copiers - whole disk sector copy, omitting ALL checking: not seeing if disk is initiallised and not checking what is written- and can copy a single sided disk in 16 seconds and a double sided disk in 33 seconds. Anyone beat that?

\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

Just in- two disks of System Test programs, with source code for the machine code bits. Main programs REQUIRE mini memory module but also some XB+32k programs. Even tests the P-Code card!
Latest revisions to C99 (to Vn 2.1+) mean the package is now spread over FOUR disks please- extras include much needed things like SCANF and EXTERN/ENTRY. Lots of C source code here for you to look at, compile and assemble. Mountain of material - on FOUR disks!

TI ARTIST : Help!

In MICROpendium of September 1986, there was a comparison between GRAPHX and TI ARTIST, which in the end came to the same conclusion I came to: "All in all, there is a good case for the purchase of both".

However, in the text there were some errors concerning TI ARTIST which may be common to many TI ARTIST users, so let's correct those errors now!

Incidentally, in preparing this article I came across an oddity: Chris Faherty is writing a special version of TI Artist for the new Geneve computer, so you might expect any revision of TI Artist to move in the right direction- unfortunately, although Version 2.0 of TI Artist works very happily with Myarc Extended Basic, the newer Version 2.1 fails to work at all!

ERROR ONE- quite a common one as I have seen it in another user groups mag as well!: "THE TI ARTIST OWNER HAS TO CONTENT HIMSELF WITH ONE SPEED". WRONG.
The TI ARTIST owner who looks at his manual (page headed KEY LAYOUT) will see reference to a SPEED key - you need to use the FCTN key as well as the indicated key!

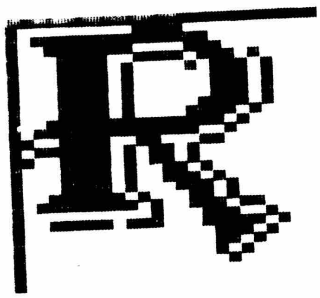
TI ARTIST gives you two speeds of cursor movement. The fast speed is pretty fast but the slow speed is just right for sensitive work!

ERROR TWO:"Fonts cannot be drawn letter by letter and stored... [you must use] TI Writer or Editor Assembler editor. ... Creating a new font with TI Artist can be a nightmare unless the user utilizes a program called CSGD."

I have CSGD and havent mentioned it because I cannot use it. The documentation is poor and the program is very unfriendly (or made to appear so). I will give below what I consider to be a very simple way to create a font for TI ARTIST and for our example we shall take the letter R from Rambles!

STEP ONE: Create the letter on the screen! Unfortunately the RLE files do not include fonts for exchange between computers, so the creative step is yours: draw the letter!
Now save it as an INSTANCE called - for the letter R -"DSK1.R" and it will be saved as a file called "DSK1.R_I"
When saving the instance, align the box so that the left most on pixel in the letter just touches the left hand line of the box. ALL your letters in one font should be aligned vertically, to prevent a wriggly appearance. You may have some leeway to adjust definitions later but make life easy and get them all lined up to start with!
The topmost on pixel of all the letters should just touch the top line of the INSTANCE box.

This is hard to describe- lets take a look at what it looks like on screen:--



The INSTANCE definition on the disk now needs a slight adjustment to make it into a FONT file:
This is how our R is defined:

```
3,4
0,0,63,32,15,15,15,15
0,60,255,195,61,66,64,64
0,0,0,128,208,232,232,232
15,15,15,31,63,79,63,79
64,65,67,255,254,49,90,90
232,232,208,160,64,128,0,0
15,15,15,31,63,0,31,0
77,78,71,163,211,17,113,0
0,128,64,64,176,200,244,232
0,0,0,0,0,0,0,0
0,0,0,0,0,0,0,0
208,160,64,0,0,0,0,0
```

The first two numbers indicate that the letter R is defined by a set of screen characters - three characters wide and four characters tall.
The 12 screen characters are then defined in the next 12 rows, in this order:

```
 1  2  3
 4  5  6
 7  8  9
10 11 12
```

Each character is defined by eight numbers which represent the eight pixel rows of the character. Instead of the usual character definitions that Basic uses:

```
8..4..2..1..!!...8...4...2...1   where
X--X--.--.-------.---X---.---X

defined as (8+4)=12=>C
     and     (4+2)=6
therefore= >C6

this pixel row is defined as:
128..64...32...16....8....4....2....1
 X    X    -    -    -    X    -    X
and the total is (128+64+4+1)=197
```

To shift a letter to the left, multiply every number by two, to shift to the right, divide every number by two. If there are any odd numbers then you cannot shift to the right!

The disk file we have is a DV80 file which can be amended with TI Writer to a FONT file, so lets do that for our R.

First load the TI Writer editor- if you have Funlwriter, load the ASSEMBLER Editor, otherwise turn word wrap off and save with PF instead of SF.

Now a FONT file requires just two extra parameters: the actual letter which we are defining! and the number of pixels to skip between the start of this letter and the start of the next letter. This last parameter defines the letter spacing, and allows proportional spacing, and varied density.

So, for the letter R, lets add to the above file so that it looks like this:

```
R
3,4,23
0,0,63,31,15,15,15,15
```

Why 23? Well, the letter is 3 screen characters wide and 3x8= 24 pixels. When we put the instance box around the letter we noted that the right hand edge of the box was well away from the right hand side of the letter so we probably want a width somewhat less than 3x8. We try 23 and see what it looks like- it is easy to change it later if we need to.

Now save the file as DSK1.R_F
and go on to the next letter.

When you are happy with all the letters, put them all together into one single file using TI WRITER editor and save them as sav DSK1.RAMBLES_F.

Remember to include a definition of space! or you will not be able to put a space in when typing your phrase. TI Artist only allows you to type in characters which have been defined in the font file.
Space could be defined as:

```
[space]
1,1,23
0,0,0,0,0,0,0,0
```

Here are some R's, with a width setting of 23, 24 and 25 in that order. I have placed a box around them for you to see how they move to the right!

Yes, creating fonts requires some work, but it is not insanely difficult. Perhaps now you will send me some fonts YOU have created!!!

Stephen

****

BENCHMARKS DIFFERENT LANGUAGES

In the October 1986 issue of Personal Computer World a new set of benchmarks was listed, suitable to test all languages - although some may not be applicable to some languages!

Looking over these benchmarks for the various languages available for the TI99/4A may help you to appreciate some of the good points of some of the languages - and some of the weak points too.

By giving listings in the various languages, you may also be helped to understand how to use them - and if you dont have them already, maybe thing about obtaining them (or not!!).

I shall look at each benchmark in turn, and start with the first, called INTMATH - here is the description:

```
DECLARE THREE INTEGER VARIABLES X,Y AND I
 ASSIGN X EQUAL TO 0 AND Y EQUAL TO 9
  WRITE "START" TO SCREEN
   REPEAT 1000 TIMES USING I AS THE LOOP VARIABLE
    ASSIGN X=X+Y-Y*Y/Y
   WRITE "FINISH" AND X TO SCREEN
      [ X SHOULD BE ZERO ]
```

In the various Basics available on the TI, Integer Math is available ONLY with version 2.1 of Myarc's Extended Basic:

```
100 DEFINT X,Y,I
110 X=0 :: Y=9
120 PRINT "START"
130 FOR I=1 TO 1000
140 X=X+Y-Y*Y/Y
150 NEXT I
160 PRINT "FINISH":X
```

NEW MEMBERS:
STAINLESS SOFTWARE CEASED TO TRADE SOME TIME AGO! PLEASE CONTACT NEW DAY OR THE USER GROUPS FOR YOUR SOFTWARE REQUIREMENTS!

This took 18 seconds.

In the following timings "MYARC EXBAS" refers to Version 2.0, using all real math, unless otherwise specified.

Integer math is also available in Machine Code, C-99, and Forth.
I will leave the machine code to someone else..

RAMBLES

This copy of Rambles has very little indeed for the CONSLE-ONLY owner. Sorry- but Rambles is based on letters received, and I have not received any console-only questions. If you have a console-only system and would like to see something for YOU, please write and tell me WHAT! in as much detail as possible.

```
TI FORTH:
 0 VARIABLE X   0 VARIABLE Y
 0 X !          9 Y !
 : TS  ." START"
   7 10001 1 DO DROP
    X @
    Y @ +
    Y @ -
    Y @ *
    Y @ /
     LOOP
 ."   FINISH"   .   ;
      TS
```

You may note that I have used a 10000 loop instead of just one thousand - mainly to give me a long enough period to time! This runs in 28.6 seconds, equivalent to 2.86 seconds for one thousand.

```
c-99
/* CONIO FIRST */ #define stdin    -1
#define stdout   -2
#define stderr   -3
#define EOF      -1
#define YES       1
#define NO        0
#define NULL      0
#define EOL      10
#define FF       12
#define BS        8
int x,y,i,loop;
main()
{ putchar(FF);
 x=0;
 y=9;
 while(++loop<11)
  { puts("start");
    i=0;
    while(++i<10001)
      { x=y+y-y*y/y;
      }
  }
puts("  finish");
}
```

NB: Machine code screen dump programs such as Danny Michaels "DUMP" and the Dataflex "DFX PRINT" do not function with Myarc XB. There is a vacancy here for a suitable program! (I suspect that different VDP addressing may be to blame?)

Notice that ten thousand was not enough for accurate timing! This ran at a rate equivalent to 0.4755 seconds for one thousand.

Before we pass to Benchmark two, a note on these timings- we know of at least 6 different console operating systems - and three different TI extended basics - so timings on other systems may differ somewhat! TI did not bother with version numbers, but the set up used was:
TI99/4A console, TI peripheral expansion box, TI disk controller, Myarc ram card, TI SSSD disk drive (made by MPI). The extended basic module used was the second version of what TI called Version 110.

So, onto BENCHMARK TWO, called REALMATH:

```
DECLARE TWO REAL VARIABLES X AND Y
 DECLARE AN INTEGER VARIABLE I
  ASSIGN X EQUAL TO 0 AND Y EQUAL TO 9.9
   WRITE "START" TO SCREEN
    REPEAT 1000 TIMES USING I AS THE LOOP VARIABLE
     ASSIGN X=X+Y-Y*Y/Y
      WRITE "FINISH" AND X TO SCREEN
       [ X SHOULD EQUAL 0 ]
```

Again, we have problems with using integers... and you do not need to declare variables (why when there is only one class!) in Basic and in some languages there is no loop counter to be called I... however, this is one benchmark we can test in MANY languages!!!

```
BASIC first:
100 X=0
110 Y=9.9
120 PRINT "START"
130 FOR I=1 TO 1000
140 X=X+Y-Y*Y/Y
150 NEXT I
160 PRINT "FINISH":X
```

Timings are:
TI BASIC: 17.7 seconds
TI EXTENDED BASIC: 22.0 seconds
MYARC EXTENDED BASIC: 31.8 seconds
(in these tests we have used Vn 2.0 of Myarcs Extended Basic - contrary to the ads, it does NOT support integer math)
[Using Myarc Vn 2.1 and defining I as an integer the timing is reduced to 23 seconds - a useful speed up]
Extended Basic allows multi-statement lines, and if the benchtest is compressed to a single program line in this manner, for both tested versions of Extended Basic, the running time is reduced by six per cent.

PILOT-99: In this language, use is made of 32 byte floating point math:

```
C: #X=0
C: #Y=-9.9
T: START
LP: 100
C: #X=-#X+#Y-#Y*#Y/#Y
EL:
T: FINISH #X
E:
```

This has been looped only 100 times to save insomnia, as the test runs equivalent to 1000 loops in 576 seconds!

We now turn to BENCHMARK 3: TRIGLOG

```
 DECLARE TWO REAL VARIABLES X AND Y
  DECLARE AN INTEGER VARIABLE I
   WRITE "START" TO SCREEN
    ASSIGN X EQUAL TO 0
    ASSIGN Y EQUAL TO 9.9
     REPEAT 1000 TIMES USING I AS THE LOOP VARIABLE
      ASSIGN X=COS(SIN(ATN(LOG(Y))))
   WRITE "FINISH" AND X TO SCREEN
    [ X SHOULD BE 1000 ]
```

Now as COS cannot produce a value of 1000. we shall read this as 1.000. which still leaves a problem: the benchmark does NOT specify that LOG must be to Base 10 nor that trig uses degrees - but if you want an answer of 1.000 that is what you must use. Our TI99/4A uses BASE e for logs and radians for trig - so for many of the tests below I have used these. There is a comparison test using the "assumed" bases. [PCW now clarified radians and e to be used]

Basic first:

```
100 PRINT "START"
110 X=0
120 Y=9.9
130 FOR I=1 TO 100
140 X=COS(SIN(ATN(LOG(Y))))
150 NEXT I
160 PRINT "FINISH":X
```

The remarkable accuracy of the TI99/4A. coupled with trig routines written in long GPL sequences means TI trig is slow. hence only looped 100 times. However the results below are for 1000 loops (100 x 10 !):

TI Basic: 624 secs
TI Extended Basic: 362 secs
Mvarc Extended Basic: 365 secs (Vn 2.0 AND Vn 2.1)

Using Log Base 10 and trig in degrees:

```
100 M=0.01745329251994
110 L=2.302585093
120 PRINT " START "
130 X=0
140 Y=9.9
150 FOR I=1 TO 100
160 X=COS(M*SIN(M*ATN(M*LOG(Y)/L)))
170 NEXT I
180 PRINT " FINISH ":X
```

The equivalent timings for one thousand loops are:

TI Basic: 640 secs
TI Extended Basic: 386 secs
Mvarc Extended Basic: 392 secs

I will leave others the pleasure of testing this benchmark in TI Forth, C-99 and machine code, but here is what PILOT-99 can do:

```
T: START
C: #X<-0
C: #Y<-9.9
LP: 100
C: #X<-COS(SIN(ATN(LOG(#Y))))
EL:
T: FINISH #X
E:
```

The equivalent time for 1000 loops is here 1710 secs!

BENCHMARK 4 is TEXTSCRN:

```
 DECLARE AN INTEGER VARIABLE I
  WRITE "START" TO SCREEN
   REPEAT 1000 TIMES USING I AS THE LOOP VARIABLE
    WRITE "1234567890QWERTYUIOP" AND I TO THE SCREEN
  WRITE "FINISH " TO SCREEN
```

Another problem looms... the benchmark makes no mention of scrolling, a time consuming process. [Now clarified: scrolling mandatory]. Not a lot of choice in TI Basic. but we do have the option in the other languages - so this benchmark has been run twice. with and without scroll. The results of this benchmark appear to be quite miraculous and will be commented on in the summary at the end:

```
100 PRINT "start"
110 FOR I=1 to 100
120 PRINT "1234567890QWERTYUIOP":I
130 NEXT I
140 PRINT "FINISH"
```

Times for 1000 loops:

TI BASIC: 260 secs
TI Extended Basic: 117 secs
Mvarc Extended Basic: 73 seconds

Now. without the scroll:

```
100 PRINT "START"
110 FOR I=1 TO 100
120 DISPLAY AT(3.1):"1234567890QWERTYUIOP":I
130 NEXT I
140 PRINT "FINISH"
```

Times for 1000 loops:

TI Extended Basic: 76 seconds
Mvarc Extended Basic: 34 seconds
(repeat: thirty four seconds)
but... incredible this - Version 2.1 of Mvarc ExBas still manages an improvement and take it down to just 30 seconds!

40

Now...TI FORTH...first.scrolling: : TEST
```
  ." START"
   101 1 DO
     ." 1234567890QWERTYUIOP" I .
         LOOP
  ." FINISH" ;
```

The equivalent time for 1000 loops is 68 secs.

Next. non-scrolling:
```
 : TEST
  ." START"
   1001 1 DO
    4 5 GOTOXY
    ." 1234567890QWERTYUIOP" I .
         LOOP
  ." FINISH" ;
```

This routine takes 33.6 secs for 1000 loops.

Before we move on to the fastest. let's try the slowest.
PILOT-99:
(non-scrolling):
```
C: #I(-0
T: START
LP: 100
C: #I(-#I+1
TC: 4.5
T: 1234567890QWERTYUIOP #I
EL:
T: FINISH
E:
```

For 1000 loops. this takes...980 secs

Moving quickly on to the fastest. first WITH scrolling. here is what C-99 can do:
```
/* console i/o first */
#define stdin   -1
#define stdout  -2
#define stderr  -3
#define EOF     -1
#define YES      1
#define NO       0
#define NULL     0
#define EOL     10
#define FF      12
#define BS       8
int I:
main()
{ putchar(FF);
  puts("start ");
   while(++i(5001)
   { puts("1234567890qwertvuiop");
   }
puts("finish");
}
```

HELP! The TI Extended Basic TEXT-TO-SPEECH disk fails to function with the MYARC RAM CARD due (I think!) to a CRU collision - the SPEAK section seems to be paging the RAM card! Is there anyone out there clever enough to change the CRU address used by the Text-to-Speech package?

Unfortunately my C skills have not yet become sufficiently refined to print the loop variable. but it should not make too much difference...
The time for 1000 loops here is 38.72 secs.

Now. take the scroll out - the upper part of the program up to main() is identical so I won't repeat it here:
```
main()
{ putchar(FF);
   puts(" start ");
    while(++i(5001)
    { locate(3.4);
      puts("1234567890qwertvuiop");
    }
  puts("finish");
}
```

and this version runs 1000 loops in 6.82 seconds.

Now on to Benchmark Five. GRAFSCRN:
```
 DECLARE INTEGER VARIABLES I AND J
  WRITE "START" TO SCREEN
   REPEAT 100 TIMES USING X AS THE LOOP VARIABLE
    REPEAT 100 TIMES USING Y AS THE LOOP VARIABLE
     PLOT PIXEL AT SCREEN LOCATION (X.Y)
  PRINT "FINISH" TO SCREEN.
```

OK- spot the deliberate error - the top line should ask you to declare variables called X and Y and I and J! Minor point... and speaking of points. all this benchmark is for is to find out how long it takes to plot 10.000 pixels on screen...

Not available in TI Basic or TI Extended Basic. but it can be done with Myarc Extended Basic:
```
100 CALL GRAPHICS(3)
110 CALL WRITE(1.22.2."START")
120 FOR X=1 TO 100
130 FOR Y=1 TO 100
140 CALL POINT(1.X.Y)
150 NEXT Y
160 NEXT X
170 CALL WRITE(1.23.2."FINISH")
180 GOTO 180
```

This took 268 seconds. Running the program in Version 2.1 and using integers. the time is reduced to 150 seconds.

FORTH also allows bit map graphics. so here is TI FORTH:
```
: TEST  PAGE  GRAPHICS2
    101 1 DO
     101 1 DO
       I J DOT
     LOOP
    LOOP
      TEXT ;
```

Notice that TI FORTH forces us to use those errant variables I and J!
This routine takes just 20.1 seconds.

Now. PILOT-99 also allows us to use bit map graphics. but those 32 byte floating point numbers seem to get in the way:

```
IG:
GC: 2.16
C: #X<-30
C: #Y<-0
TG: 12.9.START
LP: 50
LP: 25
PP: #Y.#X
C: #X<-#X+1
EL:
C: #X<-30
C: #Y<-#Y+1
EL:
TG: 13.9.FINISH
E:
```

Right- I have not used two loops of 100 each. instead. just 50 and 25. to give a total pixel count of one eighth of what it should be...and here is why- to plot 10.000 pixels would take one hour five minutes and 36 seconds.

Let's finish this benchmark off with something a might faster:
9900 Source Code.
VSBW. VSBR. and VWTR are externally referenced to routines in the Editor Assembler module. and also available in the disk file EAU with FUNLWRITER.
The routines SETUP. CLEAR and PLOT are not shown here - they were written by Graham Marshall and appeared in the diskazine 4FRONT published by New Day Computing:

```
****************
* bit map
* benchmark 5
****************
        DEF   BITMAP
        REF   VSBW.VSBW.VWTR
****************
ROW     DATA  >0000
COLM    DATA  >0000
BITMAP  BL    @SETUP
        LI    R2,>F100
        MOV   @ROW,R8
        MOV   @COLM,R9
POINT   MOV   R8,R0
        MOV   R9,R1
        BL    @PLOT
        INC   R8
        CI    R8,100
        JGT   ROWINC
        JMP   POINT
```

--------CONTINUED---->--->

```
ROWINC  MOV   @ROW,R8
        INC   R9
        CI    R9,101
        JLT   POINT
END     LI    R0,>F100
        BL    @CLEAR
        JMP   BITMAP
        * graham marshalls routines here
```

This routine took 8.66 seconds.

```
FINALLY:                                BENCHMARK 6: STORE:
 DECLARE AN INTEGER VARIABLE I
 WRITE "START" TO SCREEN        CREATE A DISK FILE "TEST"
   OPEN "TEST" [READY FOR] INPUT
    REPEAT 1000 TIMES USING I AS THE LOOP VARIABLE
     WRITE THE RECORD "1234567890QWERTYUIOP" TO "TEST"
   CLOSE "TEST"                                DELETE "TEST"
 WRITE "FINISH" TO SCREEN
```

This also gives us a problem or two. mainly because the TI Disk Operating system automatically does some of the work for us - we don't have to "create a disk file". the system does that for us when we open the file!
And if you are used to opening a file "OUTPUT" when you write to it. the reference to "open "test" for input" just might throw you there...
I've only run this benchmark in the Basics. for to TI Disk Drive and to Myarc Ram Card. I'll leave the other languages to better programmers...
In the listings below. the file is opened as the default of DISPLAY. UPDATE. VARIABLE 80. and a well used disk was used - the same one for every test. in exactly the same condition for each.
Now. DV80 may not be the fastest way of running this benchmark! but the conditions WERE identical for every run:

```
100 PRINT "START"
110 OPEN #1:"DSK1.TEST"
120 FOR I=1 TO 1000
130 PRINT #1:"1234567890QWERTYUIOP"
140 NEXT I
150 CLOSE #1
160 DELETE "DSK1.TEST"
170 PRINT "FINISH"
```

Using the MPI/TI SS.SD. disk drive:
TI BASIC: 166.5 secs
TI Extended Basic: 131.5 secs
Myarc Extended Basic: 82.8 secs

Using the Myarc Ram Disk:
TI Basic: 112.6 secs
TI Extended Basic: 68.6 secs
Myarc Extended Basic: 34.6 seconds.

++++++++++++++++++++++++++++++++

What can we deduce? Extended Basic tends to be faster than TI Basic. and while Myarc Extended Basic can be much faster than TI Extended Basic. it is NOT the rule for that to be so!
PILOT-99 is really held back by those 32 byte numbers and is not a language of choice for speed! TI FORTH can be fast. and C-99 is well worth looking at if you can't make it all the way to 9900 machine code.

Subsequent to the above. PCW has clarified certain obscure points on the new Benchmarks. and I have received Myarc XB Version 2.11

Clarification:
Textscreen: Benchmark to INCLUDE a linefeed after each print. Based on maximum size of screen available up to 80x25.
TRIGLOG: Based on use of radians and natural base logs. Accuracy- integers to 16bits signed (ok) and real math to six decimal places (not much real about that!- our machine goes to 13. with consequent slow down).

MYARC XB 2.11- compare with table...
INTMATH: 18 seconds- comparable with realmath in TI Basic!
REALMATH: 23.8 seconds - comparable to TI Extended Basic
TRIGLOG: 365 secs - comparable to TI Extended Basic
TEXTSCRN: Using the standard 32x24 screen. 73 sec. which is faster than any TI Basic. Using a 40x24 screen. slows down to 81 seconds.
GRAPHSCREEN: 151 seconds.
STORE: To TI 5 1/4" SSSD or DSSD drives: 80 sec
To Myarc Ramdisk: 28 seconds.

Note that time for STORE using the RAM CARD and compare it to using a disk with TI Basic...

Here are the programs used for these tests:

```
100 REM INTMATH
110 DEFINT ALL
120 Y=9
130 PRINT "START"
140 FOR I=1 TO 1000
150 X=X+(Y*Y-Y)/Y
160 NEXT I
170 PRINT "FINISH":X
```

```
100 REM BM2
110 DEFINT I
120 Y=9.9
130 FOR I=1 TO 1000
140 X=X+(Y*Y-Y)/Y
150 NEXT I
160 PRINT "FINISH":X
```

```
100 REM BM3
110 DEFINT I
120 PRINT "START"
130 Y=9.9
140 FOR I=1 TO 1000
150 X=X+SIN(ATN(COS(LOG(Y))))
160 NEXT I
170 PRINT "FINISH":X
```

```
100 REM BM4
110 REM 40X24
120 CALL GRAPHICS(2)
130 CALL MARGINS(1.40.1.24)
140 DEFINT I
150 PRINT "START"
160 FOR I=1 TO 1000
170 PRINT "1234567890qwertyuiop":I
180 NEXT I
190 PRINT "FINISH"
```

```
100 REM BM5
110 DEFINT X.Y
120 CALL GRAPHICS(3)
130 CALL WRITE(1.14.4."START")
140 FOR X=1 TO 100
150 FOR Y=1 TO 100
160 CALL POINT(1.X.Y)
170 NEXT Y
180 NEXT X
190 CALL WRITE(1.15.4."FINISH")
200 GOTO 200
```

```
90 REM STORE
100 DEFINT I
110 PRINT "START"
120 OPEN #1: "RD.TEST"
130 FOR I=1 TO 1000
140 PRINT #1: "1234567890qwertyuiop"
150 NEXT I
160 CLOSE #1
170 DELETE "RD.TEST"
180 PRINT "FINISH"
```

LIBRARY NEWS: BACK IN 1978 WHEN TI WERE PRODUCING TI CHESS. ANOTHER COMPUTER CHESS PROGRAM CALLED SARGON I WAS GAINING HEADLINES. WRITTEN FOR THE Z80 IT IS NOW TRANSLATED AND AVAILABLE! IT USES BRUTE FORCE WITH NO SHORT CUTS SO MOVES TAKE AN AWFUL LONG TIME! BUT IT PLAYS BETTER THAN THE TI MODULE! FROM STEPHEN. USUAL LIBRARY TERMS.

****

SPECIAL LIBRARY ADDITIONS:
Two disks of music from Ken Gilliland in sunny California-
1. STAR TREK ALBUM: Themes from the three films and the tv series. Single sided disk- no sectors spare!
2. WAGNER! Double sided ONLY-at same price as a SS disk. Requires a DSSD disk as DSK1. 609 sectors used for four pieces of music!
From Stephen. Send blank disk & return postage for up to date library list with prices!

A long time ago at the dawn of history. Paul Dicks asked for help in running a TI User Group and got none. More recently a bald gentleman in Oxford asked for help and got none.
Please read carefully this months TI*MES- you will find a gentle request. If you wish to see your valued computer supported think carefully what you can offer. DO NOT hesitate. but put pen to paper now. By spreading the load we can assure long and continued support.
WAKE UP AT THE BACK THAT BOY (and that girl - if we have any you aren't writing to me...).

STEPHEN SHAW

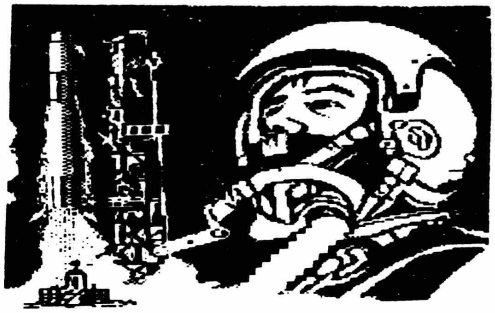
NOW... a brief summary of all those timings.

BENCHMARK:	INTMATH	REALMATH	TRIGLOG	TEXTSCRN	GRAFSCRN	STORE
---all times for 1000 loops as definition---						
LANGUAGE:						
TI BASIC	NP	17.7 SEC	TI Norm 624 SEC Base 10 640 SEC	SCROLL 260 SEC NOSCROLL NP	NP	DISK 166 SEC RAMDISK 113 SEC
TI EX BAS Vn 110	NP	22.0 SEC	362 SEC 386 SEC	117 SEC 76 SEC	NP	131 SEC 69 SEC
MYARC EXT BASIC Vn 2.0	NP	31.8 SEC	365 SEC 392 SEC	73 SEC 34 SEC	268 SEC	83 SEC 35 SEC
Vn 2.1	18 SEC	23 sec	as 2.0	untried 30 sec	150 sec	untried
PILOT-99	NP	576 SEC	1710 SEC untried	untried 980 SEC	65.5 MINUTES	untried untried
TI FORTH	2.86 SEC	untried	untried untried	68 SEC 34 SEC	20.1 SEC	untried untried
C-99	0.48 SEC	untried	untried untried	38.7 SEC 6.8 SEC	untried	untried untried
9900 MACHINE CODE	not tried	untried	untried untried	untried untried	8.66 SEC	untried untried

We long ago became accustomed to the TI99/4A being placed very very close to the bottom of the PCW benchmark listings.... but take another look at the times for TEXTSCRN above.

In the October 1986 issue, PCW gave the following timings, for these very new and very fast machines:

APRICOT Xi GW Basic.....291.2 seconds!
APRICOT PC GW Basic 3.1...243.5 seconds!
AMSTRAD CPC6128 Loco Bas..159.6 seconds!
COMMODORE AMIGA BASIC.....150.3 seconds!
IBM PC + BasicA.....100.0 seconds
SPECTRUM Basic.....84.0 seconds
IBM PC + Turbo Pascal.....76.4 seconds
ATARI 130 XE Basic.....66.9 seconds



(More screen columns explains some of these longer timings)
(- but only relevant if you use them all!)

Well well well....

THE SMART PROGRAMMER August 1986 issue had a useful and interesting machine code utility for Extended Basic. XB MIRROR. with CALL LINKS to save up to two screens of characters,and two pattern descriptor tables. to ram or disk. These would allow you to quickly change character fonts or change screen displays. And also FLIP and MIRROR routines - one acts on character definitions and the other the screen. Using both gives a mirror image! The source code occupies three pages and is fully commented.
Did you know Multiplan had an equivalent to ON ERROR to trap out divide by zero problems? This issue of the SP tells all.
FAST TERM NEWS: Sending a TI file using FastTerm. the file is sent as a straight DF128 file. If you want to send a file WITH the TI FILE HEADER. Barry Traver suggests you PROTECT the file first.(Not tested by me!).
++++++++++++++++++++++++++++++++++
The new HOME COMPUTER JOURNAL operated by the former owners of HOME COMPUTER MAGAZINE. sold at US$25 PER ISSUE have a cute way of handling complaints. They cancel your sub and refuse refunds... neat eh! I gather the local District Attorney is interested... The US TI community is splitting into two groups. one very very hostile to the other.
On one hand:
Texcomp- commercially selling freeware programs with names changed, pirating other material, shipping wrong or damaged goods, blackmailing anyone who says BOO to them
Ron Albright- as a result of such blackmail now not on speaking terms with MG.
The Forum(Compuserve)-making available pirated material and aids to piracy
and on the other hand...
MILLERS GRAPHICS...trying very hard to make a living with nothing but support for the TI, having new products ripped off and distributed by bulletin boards
BYTEMASTER...new publishers of Smart Programmer. for daring to cancel their Compuserve sub because of the pirate element
Without taking sides on what I am sure is incomplete information, I think you can spot a true TI99/4A supporter....
++++++++++++++++++++++++++
DISPLAY AT in EXTENDED BASIC. You wish to put the word DISPLAY on line 12, from column 4, what happens if you use: DISPLAY AT(12,4):"DISPLAY"
Right- line 12 is blanked from column 4 TO THE END!!!
Hum. We could use:
DISPLAY AT(12,4)SIZE(7):"DISPLAY" - that works ok.
However, using less memory with the same effect we could use instead:
DISPLAY AT(12,4):"DISPLAY";
- thats right, just a semi-colon after the string!
Thanks LEHIGH via Los Angeles Groups.
;;

MYARC EXTENDED BASIC Vn 2.11

This new version of Extended Basic is a far more than a TI-compatible : it is largely compatible with TI ExBas but has a number of enhancements which make it a worthy successor, a definate must for anyone who programs in Basic - and it could even win converts from other languages.
One major drawback: none of it is in GPL. In other words. it is not possible to fit even the TI-compatible bits into a module. With the extensions... therefore you MUST have at least a 128k ram card to run it. At present it will run with the Myarc and Foundation cards only. and a new EPROM for the cards is required - included in the price for a Myarc card and at very low cost for a Foundation card (Foundation have now ceased trading by the way).

So here is what you need in order to use Myarc ExBas: at least 128k ram. preferably 512k. and at least one disk drive, with disk operating system.

TI BASIC:
Myarc Extended Basic allows you to run 99% of programs written in TI BASIC. You may need to use CALL FILES(1) to load them from disk - but a clever bit of sidestepping allows the Myarc XB to load a longer TIB file than TI's XB can.

TI EXTENDED BASIC:
One major problem was found with running commercial programs: some commercial producers not only supply programs in PROTECTED format, but the program also checks the CPU RAM flag to see if protection has been removed, and if it has, it locks out the console (there are a range of CALL PEEKS that can do this).
Bad news: in an effort to increase the protection of PROTECTION, Myarc have moved the flag - so these programs take a look, fail to find the marker, and crash!

You will note the version number above! In the last issue I reported on Vn 2.0, and then received Vn 2.1. just as the last issue of Rambles went to press! I duly reported the problems with Vn 2.1 to MYARC, and in due course - very quickly actually - received Vn 2.11, in which most of the problem areas had been fixed. Note by the way that Myarc have included these two upgrades as part of the original price, something Texas Instruments NEVER did in the U.K.

The prescan check of for-next loops does not function correctly- and this can make debugging very interesting! If a NEXT is missing, the error message:
FOR-NEXT NESTING IN NNNN is given. Unfortunately the number NNNN has little resemblance to any line number in your program! Remember to check that every FOR has its NEXT yourself if you have problems! For existing TI XB programs this change to pre-scan makes no difference.

ACCEPT AT and DISPLAY AT also differ from TI XB, in an undocumented manner which could actually be helpful!
For instance:
ACCEPT AT(11,12)SIZE(32):A$
does what?
In TI XB it blanks from column 11 to the end of line 12, and accepts A$ from column 11 to column 28.
In Myarc XB however, it blanks from column 11, line 12, to column 11 of line 13, and the ACCEPT cursor appears on LINE 13!!!

This can cause problems! Fortunately, you are unlikely to come across this problem in many TI XB programs!

DISPLAY AT(11,12)SIZE(32):A$ makes the same deletion, spreading over two lines, but at least the display starts in the right place! Again you are unlikely to have problems with this.

Interesting... ACCEPT AT(24,12)SIZE(32):A$ will place the cursor on ROW 1!!

OK ... now to the useful bit. Lets be really odd and try:
ACCEPT AT(12,2)SIZE(255):A$
NOW... Beginning at row 12, column 2, count 255 screen characters... including the edge characters. THEN the cursor appears! BUT those 255 screen characters may not be blanks! They seem to be taken from the input buffer but I could be wrong. The result is not useful!

Now...ACCEPT AT(12,2)SIZE(-255):A$. Nothing is blanked, but the input field for A$ measures 255 chars from (12,2), including any positions outside current margins (eg edge characters). You can therefore type in quite a lot of text, and have it accepts with one key push into one string. This could be very useful. You can of course use HCHAR first to blank out that section of screen.
As this is undocumented it may not stay through to any subsequent versions, but I hope it does, as it answers a question asked often in Micropendium other TI-related mags and newsletters...
And indeed, in the 2.11 variation, Myarc HAVE kept in the above, at my request. The ability to load a string with 255 characters with one press of the ENTER key is not to be dropped lightly!

[Here is my comment on Vn 2.1:
BLACK MARK: I may be the only person to OPEN files with the APPEND declaration, but Myarc XB fails to support it. It will open the file without an error message, but then refuses to write to it! Bye Bye APPEND.]
Myarc claimed that Vn 2.11 had been fixed- unfortunately, only to the point of not issuing an error message when you try to write to the file- so you may think everything is going well - it isn't. Do not use APPEND!

The manual I have bears the following claim: "VASTLY IMPROVED ERROR HANDLING SUPPORT" - so of course I have had a close look at this area, and find the claim entirely false (sorry). In most cases, the same error messages are given as in TI XB, but in several instances, quite different messages appear (eg SYNTAX ERROR instead of BAD VALUE). In one case, an abbreviated error message is reported - eg only SYNTAX ERROR instead of SYNTAX ERROR : RECURSIVE SUBPROGRAM CALL.
In checking error messages out I came across an undocumented TI XB error message- Myarc have also not documented it but worse, have omitted the error trap entirely. Try this one:
100 DEF T(N)=T(N)-1
110 PRINT T(N)
Short program isn't it! In TI XB I received the message: UDF REFS ITSELF (almost an X-cert message eh!)
BUT in Myarc XB, these two lines crash the system!!!

And also an undocumented error message in Myarc XB, where a standard error message should be:
INVALID ERROR NUMBER IN NNNN
Thats a good one - this was produced by VERSION 2.1 with the following code: 100 GOTO 120
110 SUB TRAP
120 PRINT "ERROR MESSAGE="
130 SUBEND

However, version 2.11 produced:
WARNING
NUMERIC OVERFLOW IN 130
WARNING
NUMERIC OVERFLOW IN 130
WARNING
NUMERIC OVERFLOW IN 515
WARNING
NUMERIC OVERFLOW IN 515
Yes, FOUR warning messages - and yes, I know there is no line 515 in the code! After throwing these messages at you, your keyboard becomes unresponsive- only the QUIT key will function.
You CAN stop the WARNING messages with an ON WARNING NEXT, but the console will still become unresponsive.
The problem area seems to be the console being busy looking for somewhere to go to when it hits the SUBEND!

And in checking out this area of errors, I found something very interesting - not just a difference between the two XBs, but an unreported feature of TI XB. Read the manual on SUB PROGRAMS and it tells you that SUBEND can ONLY be followed by: Another subprogram, REM and END.
Try this:
100 CALL HEH
110 SUB HEH
120 SUBEND
130 PRINT "PROGRAM END"
Does it work?

No it doesn't, you get an error message in TI XB, whereas Myarc XB prints "READY" .
SO TRY THIS ONE in TI XB:

```
100 CALL HUM
110 SUB HUM
120 SUBEND
130 !@P-
140 PRINT "PROGRAM END"
```

and now TI XB works just like the Myarc XB. The word SUB appears to act in the same manner as STOP, and forces the READY message.
Now try this one - in Myarc XB the !@P- is not required:

```
100 PRINT "START"
110 CALL MIDDLE
120 PRINT "RETURNED FROM CALL"
130 GOTO 180
140 SUB MIDDLE
150 PRINT "IN CALL"
160 SUBEND
170 !@P-
180 PRINT "THIS IS AFTER SUBEND"
```

NOW... I am not advocating that you should place your SUB PROGRAMS in the MIDDLE of your programs - let's accept they are better at the end - BUT - they really do not HAVE to be at the end!!!

That covers normal TI XB, now onto the enhancements. The Myarc manual referred to is the one I have : it may be subject to revision!

The manual suggests that:
PROTECTED programs will auto-run and erase if CLEARed or at termination. WRONG. They act just like TI XB progrs!
DEF can be used with more than one parameter - wrong, it works exactly like TI XB.
RUN and OLD will load a program LISTed in DV80 format- wrong.[But Myarc say their new computer the GENEVE can do this]. DEFINT can be used in subprograms: wrong, it cannot- you CAN use the format suggested but the results are not those desired. Dont try it! [Myarc responded to this with the comment "?" - but it really does NOT work!]

Original review of Vn 2.1: DEFINT can be used for numeric arrays- wrong. Most unfortunate as this is the area that using INT can really save memory!
Myarc's reply on this one was to tell me how to do it! as the manual failed to do so and my guesses were not right! The way to DIM an array for INTEGER variables only is:

```
100 DEFINT DIM A(20)
```

The original manual suggests a command to force a garbage collection, which can be useful to prevent irritating problems with music and sprites when you least want them, but the command has not been implemented.- Myarc tell me they ran out memory space!

Now we move onto the GOOD bits of the ExBas side of things... and some of these are a lot better than some the ads mention...

First you have commands strangely similar to UNIX commands... PWD and CHDIR ... used in COMMAND mode. these:
CHDIR: sets the name of a default i/o device. If you do not use the command, the default device is DSK1. To use it you type in: CHDIR RD. or similar.
PWD: prints the current default device name.

What use are they? Suppose you are working on a program called PROGRAM, which is on a disk in drive 1 and the default device is DSK1.
To save PROGRAM to DSK1 you only type: SAVE PROGRAM
and to reload it you only type OLD PROGRAM

The default device name is inserted between OLD, RUN, and SAVE and the file name! Neat.

RUN when used in a PROGRAM uses a QUOTED string, and you can now have in your program:

```
100 INPUT "PROGRAM NAME?":A$
110 RUN "DSK1."&A$
```

or similar - something TI wouldn't let us do.

Also, in your program you can use a line like this one:
RUN "DSK2.PARTTWO",CONTINUE Know what this does? It retains all the variable values from program one for use in program two - including string variables!!! Think about it...

In COMMAND mode, RUN seems to use an unquoted string- quotation marks are not obligatory. They are also optional with OLD and SAVE.
thus RUN DSK1.LOAD and RUN "DSK1.LOAD" are equivalent.

PRINT and DELETE still require quotation marks.

You can also use RUN or OLD to load and run a machine code program which is on disk in memory image format. This may not always work though - see the section on Machine code below!

GRAPHICS are a real delight, and screen displays are set up SO QUICKLY! You can use the 32 column screen, the 40 column screen, and bit map mode, all very easy to use with simple EX BAS CALL commands.
The 32 and 40 column screens use identical commands, except for colour: in 40 column mode, text colour is set by using two parameters with CALL SCREEN!
In BIT MAP mode you have the extra CALLS:
CIRCLE, RECTANGLE, DRAW, DRAWTO, FILL, POINT, WRITE, DCOLOR
where WRITE is used to place text onto the bit map screen, using the 8x8 pixel character size. CALL HCHAR and CALL VCHAR are also available in bit map mode.

Another undocumented feature you will like- remember TI's canard that you could not have sprite automotion in bit map mode? This is repeated in the Myarc XB manual I have - but in truth, you CAN switch Myarc XB to bit map mode, and then call sprites with auto-motion. Neat.

In the 32 and 40 column text modes, you have the ability to "window" by setting margins for screen top, bottom, left and right. I recommend you do not LIST a program in a window two characters wide....! The margins can be changed several times, and HCHAR and VCHAR let you place characters outside the margins anyway..

In default, the margins are set for two unused columns on either side of the screen, allowing 28 and 36 printable columns respectively. Unlike standard TI XB, PRINT does NOT scroll characters in the margins (eg outside the window).

The reason you can run TI Basic programs without worrying about character sets 15 and 16 is that Myarc XB actually gives you 256 definable characters in 32 character sets - and 32 sprites for good measure. That is enough for most people! (In bit map mode Chars 216 to 255 are not available).

CALL CHARSET restores characters 32 to 95 only, maintaining compatability, but CALL GRAPHICS(N) restores all predefined characters, as well as restoring default colours and clearing the screen....

VARIABLES can now use lowercase, so that you can use A,a,b,B=2 and have FOUR variables set. CALL color is accepted and remains unchanged when you LIST but has exactly the same effect as CALL COLOR.

If you use Myarc XB to write a program and use lower case variables, the program WILL run in TI XB - but if you edit a line with a lower case variable, TI XB will re-crunch the line and change the lower case to upper case!
Here is one Myarc have kept quiet about.... MERGE format is a remarkably USEFUL thing that TI gave to us. BUT it was so slow... a long program could easily take an hour or so to load! MYARC have performed a miracle, and MERGE now saves and loads almost in a twinkling (especially if you use a RAM disk!). I was astonished at the speed up. Amazed even! Thanks Myarc!

INTEGERS... so often we hear that using integer math makes a CONSIDERABLE difference in speed. Maybe, but not what Myarc has done! You have the ability to define simple variables to be integers. If you do this, any decimal values are ROUNDED not decimal-stripped. The increase in speed? In a simple benchmark I clocked a 20% increase in speed. Handy but nothing to rave over in ads, when there are so many other improvements! The other change is that your integer variable eats up only two bytes instead of 8.

Nice new command: VALHEX. Ever wondered what >A000 was? Then use this command, as follows: PRINT VALHEX("A000") and there you are.
Unfortunately the reverse is not available!

You can now use a variety of keys to terminate a input, as well as ENTER and FCTN E and X. Whenever you do enter data, a predefined variable called TERMCHAR is set and can be referenced in your program at any time until the NEXT input! Then your program can branch depending on whether ENTER or AID or BEGIN!!!!..... catch the drift? Unfortunately REDO has not been implemented. Pity.
The various ram card commands can be used in your EXBas program now, such as CALL PART, EMDK, VOL, and RDDIR--- but for some reason these must each be on a program line of their own. Not serious, but undocumented.
Extra commands include PEEKV and POKEV.

TECHNICAL STUFF
Myarc ExBas comes in the form of a module, a disk , an eprom and a manual. Not bad for the price. Good value!

The MODULE contains ram (>6000 to >8000) into which the memory management routines go. These are needed as there is a lot of paging going on!
The 128k is divided into FOUR pages (of 32k each if your calculator is off...).
Looking at the 24k area >A000 to >FFE8 first...
PAGE 1= EX BAS PROGRAM
PAGE 2= VARIABLES
PAGE 3= STRINGS
PAGE 4= Basic Interpreter

So altogether you have 72k for program and variables.

In the 8k area >2000 to >4000, you have:
PAGE 1: For your machine code routines
PAGE 2: basic interpreter again, and a small area of what Myarc call RAM DISK.
PAGE 3: routines for VDP and Speech, and i/o buffers
PAGE 4: MORE basic interpreter and the value stack.

As you can appreciate, a lot of code has to be loaded from the disk, and life is great deal simpler if the first thing you do is copy the disk to ramdisk- to do this however you must have the 512k card!
Otherwise, everytime you QUIT you must reload the whole system from floppy.- now remedied in Vn 2.11 - Myarc are quick to react to minor comments like this! - in VN 2.11, the file 128KOS is modified and instead of slavishly loading everything everytime, checks memory to see what is there. In general, provided you have not switched the ram card off, the system will only need to reload two files from the system disk, a great saving in time!

If you have copied the disk to ramdisk, the module will load direct from the ramdisk, whatever it is called. The search appears to be: Is the file in RAMDISK? If yes then load, if no then what's on DISK 1.... and so on. Once the system is loaded from any drive, it takes a look at DSK1 for a file called LOAD. You can bypass this (undocumented) by holding down FCTN and 4 (CLEAR) until the READY message appears.

MACHINE CODE at last...

Firstly, just as with TI XB, your m/c routines have a little less than 8k to fit into. You can enlarge the area slightly by typing in:
CALL INIT :: CALL LOAD(8194,32,130)

This is about the only time you MUST use call init! If you use CALL LOAD to load a machine code program, it checks to see if CALL INIT has been used and if not, does one for you.

Naturally if you wish to move the m/c boundary back a little, you must first use call load, or else when you load your file, the system will see no call init has been done, do one, and reset the boundary!

Now then...

1. Programs in machine code written for TI XB loading and running MAY not function.
If you have any XB LOADing programs they will not operate correctly (eg the hidden code LOAD program with FUNLWRITER- and similar programs created by SYSTEX or ACE)

2. In general, machine code programs written for editor/assembler will work with the exceptions found below.
DF80 files may be too long to load in directly. Longer files can however be loaded using the UTILITY option of FUNLWRITER. (the error message MEMORY OVERFLOW tells you to try Funlwriter!).

MEMORY IMAGE files can usually be loaded using OLD or RUN.

Given that several machine code files in DF80 format are likely not to load into Myarc XB, for a whole variety of reasons, you can use them to create a memory image file which will run directly from Myarc XB.
By way of example, lets look at a game file which we shall call SNEGGIT, in DF80 format, which refused to load with Myarc XB. We then load Funlwriter and use the LOAD AND RUN loader option to load a little utility called SUPERSAVE (available from me) and use this to transfer SNEGGIT to memory image, choosing to carry over NO utilities. Yep, the new memory image file will now load directly with Myarc XB using a nice simple command like: RUN SNEGGIT.

You will find that some machine code programs - in either format - make assumptions regarding the operating environment which do not apply. Two areas of difficulty:
Comment on Version 2.1: -1. GRAPHICS: If the screen is blank or the graphics are not what they should be, try loading the program with the FUNLWRITER Utility option this restores the environment. I have only found CUBIT to fail, and even then the failure -2. SPRITES? The question mark indicates I have not traced the problem area, and have not cured it. Although many programs operate quite happily when loaded with Myarc ExBas (with or without Funlwriter) there are a few which have disconcerting side effects, apparently connected with sprite motion.
For Version 2.11, Myarc have provided an addition DF80 utility program called TIVDP which sets the VDP registers to correspond to TI XB, this apparently being the cause of the problem. It seems to cure almost all the faults I found, but I still found a black screen when loading MGR3 from the Funlwriter disk (cure: use CALL GRAPHICS(2) first- MGR3 file assumes a 40 column set up) , and FROG HAVEN by Fully Assembled Software had invisible logs for the frog to hop over - not easy!
The format is: CALL INIT :: CALL LOAD("DSK1.TIVDP") :: CALL INIT :: CALL LOAD("DSK1.PROGRAM")
and it is NOT a mistake to show CALL INIT twice above!!!
In the event of a LOAD error you will have a blank screen, and need to type NEW to recover the standard MYARC VDP register values.
If using TIVDP fails, then try switching to one of the other graphics modes (2 or 3) before loading the machine code program. The DM1000 file MGR3 on the Funlwriter disk is an example: try to load it in GRAPHICS MODE(1) and you meet a blank screen - although it will function correctly! To see what you are doing you need to type in: CALL GRAPHICS(2) :: RUN "DSK1.MGR3" and off you go.

3. FORTH: Do not use the original TI loader for Ed/AS, it will not function. Instead use the later modification for mini/memory (the Universal loader).

4. FUNLWRITER:
The LOAD program with Funlwriter will not work. UTIL1 seems to use an interrupt routine and crashes if you try to run it directly. The easiest way into FUNLWRITER is with:
CALL LOAD("DSK1.LDFW") - the file LDFW is part of the Funlwriter package.
Funlwriter remembers its source drive number, and resets various things to allow you access to otherwise difficult programs which may be anticipating an environment not matched by Myarc XB.

5. Any program with odd graphics or no graphics - try setting GRAPHICS mode to 2 (CALL GRAPHICS(2)) or loading TIVDP as above, or loading with FUNLWRITER loaders. Some programs make massive assumptions about the state of the console, when perhaps they shouldn't.

6. NEW HORIZON UTILITIES: If you have the source code for the several XB utilities which the New Horizon group have given us, you need to change any EQUates for:
PAD, GPLWS, SOUND, VDPRD, VDPSTA, VDPWD, VDPWA, SPCHRD, SPCHWT, GRMRD, GRMRA, GRMWA, SCAN, XMLLNK, KSCAN, VSBW, VSBR, VMBW, VMBR, VWTR, DSRLNK, LOADER, NUMASG, NUMREF, STRASG, and STRREF to external REFerences.
For example, change:
NUMASG EQU >2008
STRREF EQU >2014
to:
    REF NUMASG,STRREF
Then reassemble. ... in case the normal labels are not used, just watch out for any EQUates to >20NN. You will then need to look up the name!

7. INTERRUPT ROUTINES - as are used in graphics dumps such as DUMP for instance, or clock routines - do not seem to be available. There is only one vector available for interrupt routines and it looks as though Myarc XB is using it. It may be possible to adapt the environment so that interrupt routines can be used - my knowledge is insufficient to comment on this properly.

-Interrupt driven clocks: I have been able to load and run such things with Myarc XB, both from TI Forth and from XB itself, but in both cases with bad side effects. In Forth, using MON caused a lock up instead of a graceful return to the title screen! From XB, using GCHAR returned values 96 (>60) greater than the true value and NEW tried to load LOAD from DSK1... my observation therefore is that any program using the interrupt vector >83C4 is more than likely to misbehave or totally crash the system. A few programs use this address as an alternative to TI's standard auto-start.

8. CRU ADDRESS COLLISIONS: The Myarc ram card (needed for Myarc XB) has its own DSR in EPROM and some programs may disable it! - for instance, the disk copying program MASS COPY must be amended to run with the Myarc Ram Card with either version of Extended Basic!

9. The Myarc ExBas does a great deal of memory paging, and it is possible that a machine code program could interfere with this - may be the sprite problem listed above?
--- the Scratch Pad Ram (256 bytes) is used differently to TI XB but no data has been published as yet. Any machine code program using this area MAY have problems.

Cassette handling has not been included - possibly as a result of differing VDP mapping- or the thought that as few purchasers would want it, it was not worth the trouble of writing the code!

CONCLUSION:

Although there will be some programs that you cannot run with Myarc XB, this is inevitable with any upgrade - in some cases you can get round the problem with just a small bit of a rewrite. We are not looking at a Myarc failure here, but at programmers who have taken short cuts- in some cases taking advantages of bugs TI left in but Myarc have removed! In some cases a particuler upgrade demands that a degree of incompatability is introduced. Life is never easy! But by and large, I have had to use alternative modules to run very few programs.
(NB: A few programs written for Ed/As try to download code from the EdAs module- these are bound to crash using Myarc XB! Try to find the download code and patch over it, and save your curses for the program author!).

I will retain my TI ExBas module to run the odd bits of code that I do not have source code for but which require the TI XB environment. Otherwise, I can strongly recommend a Myarc ExBas purchase. Only snag is you do need the RAM card... but that in itself is WELL worth having!

Rediscover Extended Basic. Treat yourself!

(c)October 1986 Stephen Shaw

::::::::::::::::::::::::::::::::

TIPS FROM THE TIGERCUB, reprinted in THIS magazine from time to time, comes to an end with Issue 41, dated December 1986. If you found the tips reprinted here interesting, you can have a complete set for yourself, with the programs in ready to run program format!
Jim Peterson of Tigercub Software has released the Tips in Issues 1 to 41 on FOUR single sided disks, and they can be yours for just US$45 inclusive. Jim also has two disks of NUTS AND BOLTS, with over 100 ExBas subprograms on each, reviewed in past Rambles, and both disks for US$40.
Jim tells me he has had very very few orders from outside the USA, which is silly 'cos his disks are not overpriced, and represent good value. Why not drop him a line! Jim Peterson, Tigercub Software,
156 Collingwood Ave.,COLUMBUS,OH
U.S.A. 43213
(Remember: on personal imports you MAY have to pay the postman 22% of declared value as VAT and duty. Of course HM Customs are too busy to look at EVERY small packet and you MAY not be assessed...)
==========
Following on from Tom Freemans article in the last issue of TI*MES, he has now written a machine code program to print sideways at four times the speed. And in his latest TIPS, Jim Peterson has further amended the VDPUTIL program, version 3 of which appeared in our last issue.

THE DISK CONTROLS By Michael A.Ballmann

This series of articles will explain some of the secrets of the TI disk controller. To start off you need to know several memory locations and commands. At all of these address the data is inverted.

>5FF0 status read address
>5FF2 track address read
>5FF4 sector address read
>5FF6 data from disk
>5FF8 command write address
>5FFA track address write
>5FFC sector address write
>5FFE data to write

TYPE	COMMAND	BITS							
		7	6	5	4	3	2	1	0
I	Restore	0	0	0	0	h	v	R	r
I	Seek	0	0	0	1	h	v	R	r
I	Step	0	0	1	u	h	v	R	r
I	Step in	0	1	0	u	h	v	R	r
I	Step out	0	1	1	u	h	v	R	r
II	Read command	1	0	0	m	b	e	0	0
II	Write command	1	0	1	m	b	e	A	a
III	Read address	1	1	0	0	0	e	0	0
III	Read track	1	1	1	0	0	1	0	s
III	Write track	1	1	1	1	0	1	0	0
IV	Force interruot	1	1	0	1	j	k	l	n

h=1 load head at begining
v=1 varify track register
u=1 update track register
m=1 multiple records
b=1 IBM format (TI uses IBM format)
e=1 enable 10ms delay for head settling
Aa binary count data marks (FB.FA.F9.F8)
Rr binary count for step speed (6.6.10.20ms)
jkln various interupts (ues '0')

Don't be too worried if you do not understand how any of these instructions are used. All will be explained by the end of this series.

The CRU base addresss for the disk controller is >1100 and must be in register twelve. With R12 loaded these bits can be used for various control functions with the SBZ and SBO assembly language instructions.

SBO +0 turn on card
+1 motor on by toggling this bit
+2 activates the ready line
+3 sets head load line
+4 selects drive called DSK1
+5 selects drive called DSK2
+6 selects drive called DSK3
+7 selects side two of drive

Now for this month's program segment. This part will read a track on a disk in drive one. The track can be protected and does not have to be formatted for this program to read it. (No you can not read a track then write it to copy a disk some of the control information will change.)

```
*************************************************
*                                               *
* READ TRACK                                    *
*                                               *
*************************************************
       DEF  TREAD        DEFINE START
TREAD  LWPI MYREG        LOAD WORK REGISTERS
       LI   R12,>1100    SET CRU ADDRESS
       SBO  +0           TURN ON CARD
       SBZ  +5           NOT DRIVE TWO
       SBZ  +6           NOT DRIVE THREE
       SBO  +4           SELECT DRIVE ONE
       SBZ  +7           SELECT SIDE ONE
* zero track program goes here later
*AGAIN BLWP @GET#        for later program
*      BL   @SETTRK      for later program
       LI   R2,>1000     SET BYTE COUNT
       LI   R10,TBUFF    BUFFER POINTER
       BL   @SENDC       SEND COMMAND
       DATA >1B00        READ TRACK CMD INVERTED
       SBO  +2           ENABLE READY
TREAD1 MOVB @>5FF6,R0    READ BYTE
       INV  R0           MAKE NORMAL
       MOVB R0,*R10+     SAVE BYTE
       DEC  R2           ADJUST BYTE COUNT
       JNE  TREAD1       LAST BYTE? NO
       SBZ  +2           DISABLE READY LINE
       NOP               display routine
       NOP               goes here
*************************************************
*                                               *
* USE DEBUG TO VIEW DATA PUT BREAKPOINT         *
* WHERE THE TWO NOP'S ARE AND USE 'M'           *
* TO LOOK AT THE BUFFER 'TBUFF'                 *
*                                               *
*************************************************
STOP   JMP  STOP         WAIT HERE FOREVER
*      JMP  AGAIN        for later program
SENDC  MOV  *R11+,R0     GET COMMAND
       MOVB @>5FF0,R6    READ STATUS
       SLA  R6,1         GET READY BIT
       SBZ  +1           TOGGLE DRIVE ON
       SBO  +1           *
       JOC  WRTCD        READY SET? YES
       LI   R6,>7530     SET-UP DELAY
WAITL  SRC  R5,4         * SO MOTOR CAN
       SRC  R5,4         * GET UP TO
       DEC  R6           * SPEED
       JNE  WAITL        DELAY END? NO
WRTCD  MOVB R0,@>5FF8    SEND COMMAND
       SBO  +3           LOAD HEAD
       SRC  R5,8         KILL TIME
       SRC  R5,8         KILL SOME MORE
```

----> continued ----> ------>

```
       B    *R11          RETURN
MYREG  BSS  >20           WORK REGISTERS
       TEXT 'BSTART'      SO START CAN BE FOUND
TBUFF  BSS  >1000         BUFFER
       TEXT 'BEND'        SO END CAN BE FOUND
```

Next month's article will present the control information on the disk and a program to accept a track number from the keyboard. Now unless someone wants to write a program to display this information on the screen and share it with us it will be the last program presented. Save each of these program segments as a separate file.

THE DISK CONTROLS part 2
By Michael A. Ballmann

The FDC recognizes some data bytes as special when a write track command or a read command is given. These bytes and their meaning are:

F7-Write CRC characters used in error checking
F8-Data address mark (deleted data) Aa=11
F9-Data address mark (user defined) Aa=10
FA-Data address mark (user defined) Aa=01
FB-Data address mark (user defined) Aa=00
FC-Index address mark (hole in disk)
FD-Spare (no special meaning)
FE-ID address mark

When data is written it is intermixed with a clock bit between each data bit. In order to easly identify address marks they are written with some clock bits missing. The special coding is why data can not be written to the disk with the write track command.

x=data bit c=clock bit

```
1 1 0 0 0 1 1 1 =C7          1 1 0 1 0 1 1 1 =D7
cxcxcxcxcxcxcxcx             cxcxcxcxcxcxcxcx
 1 1 1 1 1 ? ? ?=F8.9.A.B.E  1 1 1 1 1 1 0 0=FC
```

When the FDC sees an F7 byte on write track. two bytes of data to be used for error checking are written. Remember TWO bytes.

```
  GAP  AM TRACK# SIDE SECTOR# SECTOR-LEN CRC
000000 FE  28     00    09       01       ??
```

The ID field tells the FDC where it is. This information indicates track 40 and sector 9 on a single sided disk with a sector size of 256 bytes. By changing this information or putting it someplace else on the disk. the data in the sector will not be found with the TI DOS (disk operating system). This a type protection used prevent backup of their software.

Under the 'IBM' disk format (TI's also 01) the sector length is used as a shift value of 0 to 3 giving a sector length of 128 to 1024 bytes. If the 'b' bit in the read and write commands is a zero. the length byte is used as a multiplyer. That's 16*(sector length)=bytes in sector. giving a sector length of 16 to 4096 bytes. (Don't use the reserved bytes F7 to FE.)

```
VREG1  BSS  >20           SET-UP REGISTERS
H30    DATA >0030         DATA FOR CONVERSION
H07    DATA >0007         DATA FOR CONVERSION
TRACK# DATA >0000         TRACK NO. TO READ
SCREN0 DATA PRESS,>0000   TEXT POINTERS
PRESS  DATA >02E2,22      ADDRESS AND LENGTH
       TEXT 'SELECT TRACK TO READ >'
GET#   DATA VREG1,BEGIN   BLWP POINTERS
BEGIN  LI   R0,>2000      BYTE TO WRITE
       LI   R1,>0300      BYTES TO CLEAR
       LI   R2,>0000      ADDRESS TO START
       BL   @CLEAR        CLEAR SCREEN
       LI   R10,SCREN0    GET POINTER FOR SCREEN
       BL   @TEXT0        WRITE SCREEN
************************************************
*  KEY SCAN                                    *
************************************************
       LI   R1,>02F8      LOAD WRITE LOCATION
KEY    LWPI >83E0         USE GPL REGISTERS
       BL   @>000E        SCAN KEYBOARD
       LWPI VREG1         GO TO MY REGISTERS
       CLR  R0            CLEAR R0
       MOVB @>8375,R0     GET KEY
       CI   R0,>FF00      CHECK FOR NO KEY
       JNE  KEY           YES. KEY PRESSED
KEY1   LWPI >83E0         USE GPL REGISTERS
       BL   @>000E        GO TO SCAN ROUTINE
       LWPI VREG1         USE MY REGISTERS
       MOVB @>8375,R0     STORE KEY
       CI   R0,>0D00      WAS KEY ENTER?
       JEQ  ENTER         YES.
       CI   R0,>3000      WAS KEY ZERO OR HIGHER?
       JLT  KEY1          YES.
       CI   R0,>3A00      IS KEY GREATER THAN 9?
       JLT  DECODE        YES. DECODE AND DISPLAY
       CI   R0,>4100      IS KEY LESS THAN 'A'?
       JLT  KEY1          YES.
       CI   R0,>4600      IS KEY GREATER THAN 'F'?
       JGT  KEY1          YES.
DECODE MOV  R0,R5         MOVE KEY TO R5
       MOV  R1,R2         GET WRITE LOCATION
       INC  R1            ADJUST WRITE LOCATION
       BL   @ADDST1       SET-UP WRITE ADDRESS
       MOVB R5,@>8C00     WRITE KEY TO SCREEN
       SWPB R5
       S    @H30,R5
       CI   R5,>000A      IS KEY A HEX DIGIT?
       JLT  SAVE          NO.
       S    @H07,R5       MAKE A HEX DIGIT
SAVE   SLA  R8,4
```

---> ---> continued ------->

```
       SOC  R5,R8        STORE NUMBER IN R8
       JMP KEY           GET NEXT KEY
ENTER  ANDI R8,>00FF     MASK OFF MISTAKES
       MOV  R8,@TRACK#   SAVE TRACK TO READ
       RTWP              RETURN TO READ PROGRAM
TEXT0  MOV  R11,R6       SAVE RETURN
TEXT1  MOV  *R10+,R0     GET POINTER TO TEXT
       MOV  R0,R0        WAS IT LAST TEXT?
       JEQ  RETN6        YES.
       MOV  *R0+,R2      GET WRITE ADDRESS
       MOV  *R0+,R1      GET LENGTH
       BL   @ADDST1      SET UP VDP ADDRESS
WTEXT1 MOVB *R0+,@>8C00  WRITE DATA
       DEC  R1           ADJUST COUNT
       JNE  WTEXT1       DONE? NO
       JMP  TEXT1        GO FOR MORE TEXT
************************************************
* CLEAR SCREEN                                 *
************************************************
CLEAR  MOV  R11,R6       SAVE RETURN
       BL   @ADDST1      SET-UP VDP WRITE ADDRESS
WLOOPA MOVB R0,@>8C00    WRITE BYTE
       DEC  R1           ADJUST COUNT
       JNE  WLOOPA       LAST BYTE? NO.
RETN6  B    *R6          RETURN
************************************************
* SET-UP VDP WRITE ADDRESS                     *
************************************************
ADDST1 AI   R2,>4000      MAKE A WRITE ADDRESS
       SWPB R2
       MOVB R2,@>8C02
       SWPB R2
       MOVB R2,@>8C02
       B    *R11         RETURN
```

Next month we put these two segments together.

THE DISK CONTROLS By Michael Ballmann

A while ago someone ask me the size the files in the disk directory. I have now come across the answer. The files are (decimal) 38 bytes long. Any other size will give an error.
I once asked what all the names and uses of subprograms in the disk service routine. I now have the answer. If you want to know just ask.

Now this program needs:
1) Error checking so you can't try to read past the last track.
2) Some way to quit the program.
3) A display routine.
4) A way to switch to the second side.

I feel there is some merit in typing in programs so the only way this program is available from me will be on paper. It's less than 200 lines long so it should be easy to input it in with one sitting.

I have not used some of the commands available on the FDC (WD-1771). Here's how they work:

SEEK: The track register at >5FF2 must have the current track number. load the data register at >5FFE with the desired track. then send the SEEK command. The FDC will then issue the necessary step signals to arrive at the requested track. If the 'v' bit is set in the command the FDC will then check of the track number on the disk.

STEP: This command just steps the head the same direction as the last head move.

READ: Position the head on the correct track. load the sector register at >5FF4 with the desired sector. then send the read command. The track register and the track data on the disk must agree! Data will be at >5FF6. If there is a CRC error in the ID field this command will abort.

WRITE: This command is like the READ command except data is out at >5FFE to be written to the disk.

READ ADDRESS: When sent this command the FDC will read the next ID data field. The data will be at >5FF6 and will be six bytes long. The bytes will be:
1) Track number as it appears on the disk
2) Side number. Not use by the FDC.
3) Sector number.
4) Sector length.
5) First byte of the CRC error checking data.
6) Second byte of the CRC data.
If there is a CRC error: this command. or any which check the CRC. sets the status bit number three. Remember these bits are not numbered backwards like the bits on the TI.

WRITE TRACK: Just like the READ TRACK except it writes data instead of reads it. Don't forget the different addresses for read and write.

FORCE INTERRUPT: When this command is received the FDC stops whatever it's doing and goes not busy.

THIS QUARTERS COPY OF RAMBLES has gone WAY over the top on length. and in consequence quite a lot is going to be cut out before you see it. The following articles. if they do not appear here. ARE available on disk. just send disk plus return post and a donation to cover copying time!
Machine Code: Array Search: On Pixel coincidence with sprites: sound to light program
Full review of Myarc XB Vn 2.11: New PCW Benchmarks
Details of TI disk usage with m/c source utility.

Now the status bits at >5FF0:

7) All commands- NOT READY
6) Write and type I commands- WRITE PROTECTED
   Read- 1st bit for data ID byte decode
5) Type I- Head engaged. Write- Write fault.
   Read- 2nd bit for data ID byte decode
4) Type I- Seek error
   Others- Requested data not found
3) All commands- CRC error
2) Type I- Track zero. Others- Lost data
1) Type I- Index (Beginning of the track)
   Others- Requesting service of data register
0) All commands- BUSY

```
*************************************************
* SET THE  DRIVE TO TRACK ZERO                  *
*************************************************
      LIMI >0000        DISABLE INTERUPTS
      BL   @SENDC       SEND INSTRUCTION
      DATA >F700        SEEK TRACK ZERO
      BL   @CBUSY       WAIT UNTIL DONE
      CLR  R7           POINTER TO CURRENT TRACK
*************************************************
* MOVE THE ABOVE SECTION OF CODE TO THE PLACE   *
* RESERVED FOR THE ZERO TRACK ROUTINE AND DELETE*
* THE '*' AT THE BEGINNING OF THE THREE LINES   *
* MARKED FOR FUTURE USE. DELETE THE STOP LINE   *
*************************************************
SETTRK MOV  R11,R13     SAVE RETURN
       MOV  @TRACK$,R1  GET DESIRED TRACK NUMB.
       C    R1,R7       COMPARE DESIRED / ACTUAL
       JLT  SOUT        IF LESS THEN STEP OUT
       JEQ  RET13       EQUAL THEN RETURN
SIN    BL   @CBUSY      CHECK BUSY BIT
       BL   @SENDC      SEND COMMAND
       DATA >BD00       SELECT STEP IN CMD
       INC  R7          ADJUST TRACK POINTER
       C    R1,R7       IS DRIVE AT WANTED TRACK
       JNE  SIN         NO.
       JMP  RET13       RETURN TO MAIN PROGRAM
SOUT   BL   @CBUSY      CHECK BUSY BIT
       BL   @SENDC      SEND COMMAND
       DATA >9D00       SELECT STEP OUT CMD
       DEC  R7          ADJUST TRACK POINTER
       C    R1,R7       IS DRIVE AT WANTED TRACK
       JNE  SOUT        NO.
RET13  B    *R13        RETURN
            * CHECK FOR DRIVES BUSY
CBUSY  MOVB @>5FF0,R0   GET STATUS
       SLA  R0,8        FIND BUSY FLAG
       JNC  CBUSY       IS IT SET? N
       B    *R11        RETURN
       END
```

Merge these three segments together in the order presented, then make the move and changes noted in this last segment, then assemble the result with the 'R' option. If when you assemble this code you want a printed copy; type in the correct print device. If you have PIO include a '.' after the name.

John Stocks has kindly sent me a source listing of his Mini-Memory graphics program mentioned in the last issue - it uses bit map graphics and QUICKLY produces a large number of odd designs! - due to the use of a look-up table to the trig. and the length of the source (5 pages without the tables!) I cant list it here unfortunately. but John is most happy to send a recording of the program to you - just send him a tape and return postage. The program plots 216 patterns. each with 1256 pixels. And the speed is splendid. For mini memory only. Send TAPES only.
John Stocks, 11 Stonehill Rd., Roxwell, Chelmsford, CM1 4PF

++++++++++++++++++++++++++++++++++++++

Gerry in Warley is just one reader with increasing problems with module and peripheral connection problems. I have already suggested the use of cotton buds to clean the contacts. but two products which give a more satisfactory solution have been brought to my attention, and are readily available by mail order from MAPLIN whose catalogue is sold at most branches of W H SMITH.
I am quoting from the 1987 catalogue and prices are good to February 14th. Check the TOOLS section (pages 449 and 450):

a. First cleanse the edge connectors using a special moist tissue described as "SAFEPADS" (ref FM81C, 28p for two). DO NOT use "Cleaning Strips". The moist tissue can be wrapped round an old lolly stick or even a cotton bud if you have problems getting at the edge connectors!

b. Now use CONTACT CLEANER LUBRICANT PEN ("SWITCH CLEANER PEN"), item FM77J, 98p. Soak a clean cotton bud with the lubricant and wipe it over both sides of the edge connector. Dont be shy!

This should solve most problems!
By EDGE CONNECTOR I am referring to the PCB in the modules, and in the peripheral sockets.
If you dont have cotton buds, Maplin can supply them too. ten safebuds (Item ref YK986- Data Buds Pk of 10) will cost you 18p.

PILOT-99: Problems? Pilot-99 is written in FORTH, and to load it with Funlwriter you must use the Forth set-up. Use K=60 in the User List selection.

FUNLWRITER./ti writer...

Q: I set left margin at 0 and right margin at 33, to avoid annoying windowing. However I want to use .HE and .FO with full width text... how can I get past the right margin to do it?

A: Although there is a left margin release (CTRL Y) there is no right margin release in TI WRITER. The problem is easily solved however, by changing the TAB setting just while you key in the .FO and .HE lines. TI Writer allows you to change the TAB setting as often as you wish - for every line if the fancy takes you - and although REFORMAT and REPLACE STRING (with word wrap on) will use the CURRENT tab setting to rework the CURRENT PARAGRAPH (in the case of RS, any paragraph changed), your work is otherwise NOT reformatted.

I keep my eyes open for short pieces of code with interesting things in them - it is easier to see what a small piece of code is doing. So here is a really odd bit of code which takes an input from the cassette port! Follow what it is doing, and perhaps try it!

```
********************
* SOUND ANALYZER 1 *
********************
* DISPLAY SOUND INPUT FROM CASSETTE
* PLAYER ON MONITOR SCREEN
      REF  VSBW    *REFERENCE VIDEO WRITE
      DEF  RUN     *DEFINE PROG START
RUN   CLR  0       * CLEAR REGISTER ZERO
B     LI   1,>1E00 * LOAD REGISTER ONE WITH >1E00
                   * WHERE >1E00 REPRESENTS CHR$(30)
                   * WHICH IS THE CURSOR
      TB   27      * TEST BYTE 27 or more simply:
                   * Read CRU bit 27 and set
                   * Equal Status bit to 0 or 1
                   * NB: See EdAs manual page 409
                   * CRU Bit 27 is Tape In line
      JNE  C       * If equal status bit is
                   * zero, go to label C below
      AI   1,>100  * Equal status bit is 1
                   * Add >100 to Register one
                   * >1E00 + >0100 = >1F00
                   * and >1F00 represents CHR$(31)
                   * which is the undefined edge
                   * character and appears on
                   * screen as blank.
C     BLWP @VSBW   * Write the character whose value
                   * is in register 1 to the screen
                   * position ( 0-767 decimal) to be
                   * found in Register zero.
      INC  0       * Add one to Register zero
                   * eg next screen position
      CI   0,>300  * Are we at the end of the screen?
                   * ( Register 0 = hex 300)
      JNE  B       * If not, then find out what value
                   * the next screen position is to
                   * have by jumping back to label B.
      JMP  RUN     * As the screen is full, lets
                   * start again! Go back to label RUN
      END          *
```

This is written to be assembled for Editor Assembler or Mini Memory. To run it from Extended Basic you need to change the reference to VSBW with the XB equate, which is in the Ed/As manual on page 416:
Do not enter the REF line.
After the DEF line add:

```
VSBW  EQU  >2020
```

With somewhat less comment here are two related source codes. Try to follow what they are doing yourself!

```
* SOUND ANALYZER 2 *
********************
* THIS PROG USES DIRECT VIDEO MEMORY
* WRITE TO IMPROVE DISPLAY SPEED

      DEF  RUN
* PREPARE VDP MEMORY FOR DIRECT WRITE
RUN   LI   2,>40   *SET REG 2 ADDRESS 0
A     LI   0,>300  *RESET SCREEN COUNT
      MOVB 2,@>8C02 *SET LOW BYTE
      SWPB 2
      MOVB 2,@>8C02 *SET HIGH BYTE
      SWPB 2
                   * READ AND DISPLAY SOUND
B     LI   1,>1E00 *>1E IS BAR CHARACTER
      TB   27      *TEST SOUND INPUT
      JNE  C
      AI   1,>100  *CHANGE TO BOX CHAR
C     MOVB 1,@>8C00 *DISPLAY CHAR
      DEC  0       *DECREMENT SCREEN COUNT
      JNE  B       *JUMP IF SCREEN ISN'T FILLED
      JMP  A       *RESET SCREEN
      END
```

```
********************
* SOUND ANALYZER 3 *
********************

* ALLOWS FREEZING OF SCREEN BY PRESSING
* FUNCTION KEY

      DEF   RUN
RUN   LI    2,>40
A     LI    0,>300
      MOVB  2,@>8C02
      SWPB  2
      MOVB  2,@>8C02
      SWPB  2
B     LI    1,>1E00
      TB    27
      JNE   C
      AI    1,>100
C     MOVB  1,@>8C00
      DEC   0
      JNE   B
D     TB    7     *TEST FUNCTION KEY
      JNE   D     *IF PRESSED GOTO D
      JMP   A
      END
```

VOTING RECORD: In response to the request in last Rambles, three votes were cast for articles to appear, suggesting that the items listed were not at all popular. The winner, the disk article, has been sent in with this text.

If you want your program to detect whether some other key is depressed, check CRU bit NN with TB NN:- where NN is 3 to test for =, 4 to test for SPACE, 5 to test for ENTER, 8 to test for SHIFT, and 9 to test for CTRL. And check whether using the JOYSTICK has any effect when this program is operating!

REFER TO The Smart Programmer, September 1986 for a useful CONSOLE CRU BIT MAP

++

ENTOMOLOGY CORNER

BY TONY MCGOVERN

The following text is by Tony McGovern and is taken from:
ENTOMOLOGY CORNER

The text has been edited to omit redundant items, and to cut down the size a touch!
Spring has sprung and the days are getting longer. Just as well as we have a bunch of rainforest tree seedlings to plant out. Have to keep the funnelwebs happy and feeling at home somehow. On the computing front from time to time minor changes and updates are made to Funlwriter. I'm going to have to think of a better name for that program now that it has outgrown its original bounds by so much. The latest version of DM1000 (Vn 3.3) was received from Bob Boone in Ottawa and promptly interfaced to F'Wr as part of the current issue.

The C99B adapter file has also been revised so that it works with both the existing Vn 2.0 of c99 and the upcoming Vn 2.1 as well. There is also a C99PF1 file modified by re-assembly to bypass unloading of the E/A utilities from GROM in all circumstances. This allows c99 program files to be run from F'Wr and E/A, which is how we usually use it. This may well be the last issue of the c99 compiler for the 99/4a as Clint Pulley is now working with a sample of the new Myarc Geneve computer. Clint sounds very happy with the new machine, with its faster operation of TI-Writer, and real single-key cursor operation on the IBM type keyboard.

Another no-no we were just painfully reminded of is turning off the power to a second or third external drive while it is still connected up to the computer. Attempts to access or even just catalog the first drive may then destroy disks. This happened to our first Newsletter Editor, Steve Taylor and I think it has just struck here too with disastrous results to the only copy of some Source code!

There does appear to be one residual bug in F'Wr that I am aware of but have not yet been able to fix. As reported by Woody Wilson from San Diego, the one that prompted several bug fixes and polishings. I never could reproduce the original, and assumed I had fixed it in the process. As you well know, program bugs just go into hiding unless explicitly squashed, and this one was no different.
Brian Rutherford unearthed it again the other day, when he Listed an assembly to his printer and found line feeds weren't being sent to the printer. So where was it hiding ? In the PIO routine. Why didn't we find it before ? Because we use a TI-99 printer always set to RS232.BA=4800, and the bug only shows up with PIO. This is very strange because the only obvious relevant difference between F'Wr and E/A is that E/A opens the LIST device from GPL while F'Wr uses its normal DSRLNK routine. The trouble with that as an explanation is that F'Wr's DSRLNK is identical to TI-Writer's and that works perfectly with PIO, otherwise we would have heard the screams long ago. I am going to have to dig deeper.

As assembly programmers already know from the E/A manual the RS232 card is an unruly beast which doesn't always follow the Tech Manual specs - and it runs deeper than just not preserving the GROM address. For the time being if you have a parallel printer, the work-around for Listing short to medium length assemblies is to list to disk and then use the Editor to print out the file. For long files it may be better to reset the printer to handle the absence of line feeds.
William has been busy too, mostly engaging in conversations with disk controller chips. His DISKHACKER program (part 1) was demonstrated at the last Club meeting and is now being sent out as a fairware release. [available from Stephen Shaw]. For those who weren't there it analyses sector patterns on a disk, track by track, and presents the analysed information for your inspection.
It starts off where MB's Advanced Diagnostics leaves off in a trail of deception, refusing to present perfectly good but non-TI-standard tracks, let alone present any of it in analysed form. The first release works with TI Controllers. It's not that he doesn't know how to make Corcomp and Myarc controllers jump through hoops already, but just that he can't face rewriting an existing working program when there are new worlds to conquer. It is an interesting exercise to inspect the details of various protection schemes used on commercial disks.
I have been embargoed from even hinting at the details of the protection methods used on more recent commercial disks until after later parts of DISKHACKER are released. Suffice to say that DH'r is a sector analyser.
This is all a lesson in the futility of disk protection as practised. All it does is make the programs so treated inconvenient and inflexible in use, and puts the serious user to the bother of removing the protection to make a backup.
As an engineer quoted in a fascinating recent article in the IEEE Spectrum on protection methods said -- "I regard disk protection as a bug, and when I find a bug I fix it". A disk is a fragile enough form of archival storage, let alone in regular use where it may encounter a malfunctioning computer, or even just one with external disk power not switched on. Our personal policy is to refuse to buy any program on a protected disk. In practice it doesn't cause any hardship as none of the programs we have seen in this format are such that they can't be lived without. Let's take as a recent example the Miller's Graphics DISkASSEMBLER.
From what we've seen it looks like a very good job has been done on it. Is the idea original ? Well, not really because it has been obvious for some while, and such programs are found on other systems. Will and I discussed the idea of writing a program to disassemble from disk files, recognising that it would have been convenient to have had earlier. With no pressing need apparent, and much work to be done on other things the idea was shelved. We weren't the only ones either, because there is a fairware program Universal Disassembler written in FORTH available from Stephen

Apart from initial inspection we haven't had occasion to use that one either, but it looks a good program too. DkA came our way as the protected disk with a request to make a backup for the owner. William can't resist a challenge like that, and it took him precisely one day to clone the disk using only programs he had written himself, apart from using the DkA on its own loader.
Since then we have had reports that a cracked version is circulating in the USA. This news seems quite believable because it took Will only one day more to reduce DkA to E/A program files. And if a 16 year old high school kid in Newcastle can do it that quickly, then how many more must there be across the whole US of A who can do the same ? Now comes the silly side of the protectionist's paranoia. I decided that DkA looked like a worthwhile program to have and use, and that if we used it, it would be from a genuine original.
So I wrote to Miller's requesting price quotation on an unprotected version of the program, and received a rather prissy reply to the effect that they didn't sell it unprotected. So there was one sale lost to protection, and I'm sure the refusal won't have helped curb piracy one little bit even if Miller's wouldn't trust us to respect their copyright.

I think it is also true now that many of the best and most distinctive programs for the TI are not protected, and come as fairware for that matter too. And outside the TI world you only have to look at the quality, value, and success of Turbo-Pascal along with Borland's later products.

All writers of disassemblers seem to have one thing in common, share holdings in suppliers of printer paper and ribbons.

While the thought of protected software brings the fragility of disks to mind I should remind you of TI's advice in their Software Development Handbook --- always maintain 3 copies of important program or source files that you are working on. Why not just two ? If there are three you have no excuse for ever having all your copies in the machine at once, even for updating purposes. Also never update both backups at the same time without checking the validity of the initial copy first.

Now for some impressions of the TI-99 scene. Micropendium (that doesn't sound like a Texas size appendage) continues it's timely appearances. It carries a range of purely TI-99 relevant material of interest to all levels of TI-99 user. Just don't take too seriously the first letter to the editor on any particular technical topic until the later mail has come in. All in all well recommended for a subscription, and far superior to anything that existed in the days when TI still produced the machine.

The Smart Programmer has now reappeared with similar content to that of old, and promise of timelier publication schedules. As it has higher technical pretenses than Micropendium I will judge it accordingly. The first two of the new series are a mixed bag. Previous issues made a big deal of publishing details of the TI ROM code. In fact very little detail appeared of relevance to the assembly programmer in the various maps, that couldn't be found out very quickly, and it never really was much more use than the E/A and Technical Manuals.
A handy reference on occasion but that's about all. If you want to see a real exegesis of the console internals see the TI-Intern book from Germany. Our copy came from Bernie Elsner in Perth. That will will show you how it should be done. I'm sure the Miller's stable have an equally thorough knowledge, but are not prepared to share it.
That's understandable for commercial reasons but I do object to all the hype generated to convince people that the SP is really telling them something. This gap between between hype and reality is characteristic of all the Miller's output, at least on a hard-nosed engineer's judgment of the real utility of the products. Which is not to say that some really high class work hasn't gone into MG's program and gadget output.

The new series carries on the tradition. The second issue contains a dissection of the MINIMEM module which says nothing more than is in the manual possessed by every owner of that module. Big deal ! The other articles are of more interest. The one on disk/cassette load and transfer utilities for machine code program files presents useful programs.
They are about on a par with Will's beginning efforts in this vein, and at least they inspired him to redo the job to Funnelweb Farm standards incorporating his own text-mode machine code file-name editor that he had written for DISKHACKER. It is now on issue as fairware. [CASSTRANS and CASSLOAD from Stephen]. Another article on the TMS-9995 processor was of interest in the "what might and should have been" category (and may yet be if Myarc's machine gets off the ground).

The high point of the issue was a DSRLNK/GPLLNK that used the console routines so that it was shorter even if slower than the usual assembly routine. A quick glance shows 4 bytes could be removed from the GPLLNK, and there must be something I haven't quite cottoned onto in the DSRLNK which allows multiple varied use under error conditions. Not surprisingly it uses indexed addressing to support GROM paging, a fairly reprehensible omission from Funlwriter perpetrated for reasons of code squeezing, but then I don't have any interest in flogging Gramkrackers and always advise people to spend their money on an honest straightforward RAMdisk instead.

The XML used (the eXecute Machine Language escape from the clutches of GPL) is one I have noticed earlier . a freak accidental in a data table near the end of GROM #0. I never would have used it. because I could have had no absolute confidence that this table was oreciselv the same in all consoles. let alone residino at the same absolute address. The method of searching GROM #0 for a regular XML is sounder. in the absence of complete knowledoe. Now what this article did do that very few individual owners are in a oosition to do is claim that the routines. which used absolute addresses in the console includino this XML. would work on all models of 99/4a oroduced.

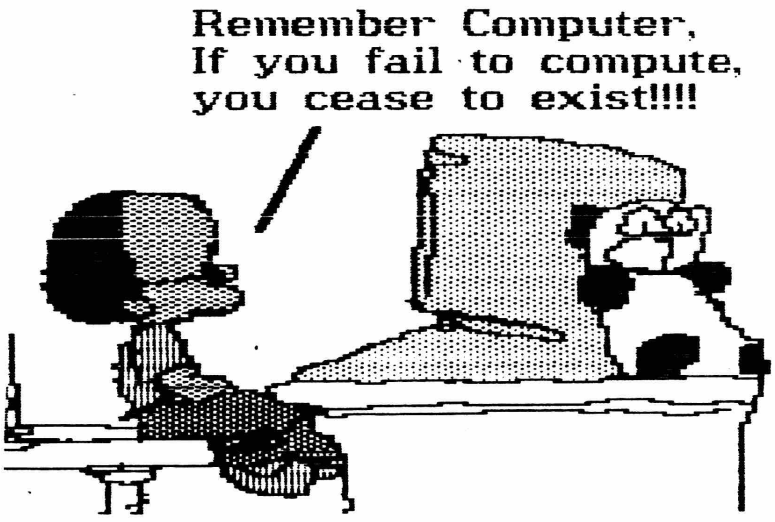
This little oem made the second issue verv much better than the first which was mostly a plug for MG's hardware offerino - the Gramkracker (a pun that doesn't translate from the American). I don't think I would lay out anv monev for this oadoet as it doesn't seem to do anvthino that we want to do that wouldn't be done better with Funlwriter and a RAMdisk. I have heard F'wr described as the "poor man's oramkracker".
Combine that with the availabilitv of oroorams to dump GROM onlv cartridoes and run them from memorv exoansion with their own GPL interoreter in RAM as well. and there isn't much call for the MG device unless you have monev to spare. Mv advice is to go for a Horizon or Myarc RAMdisk instead. The main item of interest in the first issue was a FORTH article. biased towards Wvcove Forth.
I have heard from several sources that the Wvcove version is better than TI's. beino smaller and faster. The Wvcove is a commercial oroduct and has the distinction of beino one of the verv few machine code oroorams available from a source other than TI before thev orphaned the 99/4a. and orobablv the best too. I can't comment on the relative merits from personal experience because after a verv short flirtation with TI-Forth I rapidlv decided that the elegant TMS9900 assembler code was easier to use. When we oet back to high level languages it will be with c99. and Pascal if I can ever oet the p-system workino on DSDD disks.

We have received from France a copv of the TI-Writer disk sold there bv TI. This is comes up as Version 2.0 on the Editor's end of file message and the Formatter is dated 1983. The disk fullv supports the foreign language capabilities. With the version sold here selectino one of the module's language entries would bring up the GROM resident oarts (selection screen and SD) in the language selected, but the Editor and Formatter internallv were still in English. or what oasses for it in word processor oromot lines. In the continental version there are three extra sets of files. a character file for each language with true lower case CHARA1-CHAR61. and a set for each of the Editor and Formatter which contain the commands and prompts in each language.

The bug in Recover Edit that was introduced with the *fix 1 update of Vn 1.0 issued to User Groups bv TI has been repaired. I'm not sure what the copyright status of Vn 2.0 is. I have that feelino that if TI had ever imagined that somethino like F'Wr was possible. which thev clearlv didn't. that thev might never have released *fix 1 of Vn 1.0 to the oublic domain. I suspect TI don't ever want to be reminded of the existence of the 99/4a but thev are a laroe corporation, from the land of litioation. that can afford lots of lawvers.

There is an incompatibilitv between the Editor versions also in that tab records written bv Vn 1.0 are not recognised as such bv Vn 2.0 but print as a line of special characters. The character files have an extra set of entries above #128 for extra screen characters eg for French small characters with accent marks.
I don't know whether thev cater for Canadian French. In the absence of a manual I'm not sure how these are entered from the kevboard. That all can be lived with but the reverse problem is more serious - the tab record written bv Vn 2.0 locks up Vn 1.0 on the way in. I temporarilv lost this file until I went back to TI-Writer and Vn 2.0 and used PF to rewrite the file again without a tab record. As a curiosity the English language version is British rather than American. Seeino as we use £ often and pounds sterlino but rarely, the American version is more appropriate here in Australia.

Other recent arrivals in the fairware line include PRBASE Vn 2.0 which looks like becomino the database orooram of choice, and the RA6 macro-assembler. [both available from Stephen] .Haven't had a chance to look at either of these in detail yet.



ENTOMOLOGY CORNER

TONY MCGOVERN

ENTOMOLOGY CORNER

Cont'd

While I was able to get near the machine, I got down to work on various ideas and lo and behold FUNNELWEB Vn 3.4 is now with us. Note the renaming. The previous title Funlwriter has for some time seemed just too restricted in its implications. All of the improvements that accreted to Vn 3.3 have been retained and changes made to the user interface. Now the central menu screen has two parts, toggled by pressing just about any key. This replaces the "switch" cycle which had outgrown its welcome. Functions of interest to programmers are concentrated on the second screen. There are various other detail improvements and a great deal of tidying up of internal code which had grown haphazardly as items were added to the switch list. It all adds up to an improvement so noticeable that we are reluctant to use Vn 3.3 any more.

Bob Boone told me from Ottawa that Vn 3.5 of DM-1000 was being released for the Chicago TI-Fair last weekend, and that they have incorporated the code for return to FWR as a permanent feature. We should have that soon to incorporate in the package. The good news is that the 1-sector file problem has been resolved. The 16/18 sector problem remains with Myarc controllers. We always use the Myarc DM to do 18 sector per track formatting in double density and this then leads to trouble with DM-1000 which detects the card and sets for 16 sector. A solution to this problem is at hand in that Will bashed up a routine that will format Myarc at 18 sectors as reliably as Myarc does it. When Vn 3.5 arrives I will look at incorporating this so that all DD disks are handled at 18 sectors.

The cooperation between Ontario and the Hunter Valley is helping to refine a very comprehensive and easy to use system. I certainly have no desire to get into the complexities of emulating the file system and the DM module and more that Bruce Caron took on when he created DM-1000. It works the other way around too. There is only so much that can be done in the time available by any one person. I had a letter from the USA from someone who used IBMs at work and the 99/4a at home, and because of FWR much preferred the TI for ease of use.

We have in the last few days taken delivery of a Myarc 512K RAMdisk card for the PE-box. At this stage I can give only initial impressions as we haven't had time to dig into it yet. The main one is that it works like a charm. It helps to have a Myarc disk controller as the DM for that is aware of the details of the ramdisk even when it isn't emulating a regular disk. Physical construction is on a par with the Myarc disk controller - not quite up to the overkill in mechanical standards of the TI cards but perfectly adequate. The heart of the card is 16 256K dynamic RAMs and their TMS4500A controller. The TI 32k expansion is replaced by 32k of the ramdisk and the remaining 480k is bank-switched in 32k at a time under control of the DSR ROM. Up to 400K of this is available as ramdisk, the remainder being available for a print spooler. Why 400k ? The reason would appear to be that with the standard TI disk format which the ramdisk follows, the bitmap on sector 0 allows for this much worth of 256 byte sectors and compatibility at the sector addressing level is retained for normal software such as DPATCH or various cataloging programs. The card resides at CRU address >1000 so that it sees standard DSRLNK searches before the normal disk DSR. This means that when set up by the CALLs from Basic/XB or the Myarc DM to emulate an existing drive number, the ramdisk is found first. The only unfortunate choice that Myarc made was to have it call itself RD. in its native state rather than DSKR. which makes it less compatible with existing software than it could have been. The print spooler appears to work by having an interrupt routine flagged in the DSR ROM header which would be found by the normal VDP interrupt routine. This then goes to code which addresses the RS232 card and presumably spits out some of what has been stored in the spooler memory. We have not fully explored how this system works with programs. I would expect that if a program suspended VDP interrupts by LIMI 0 that spooling would cease. The TI-Writer Editor and FUNNELWEB enable interrupts during their keyscan routines, that is to say while the computer is waiting for user input. The Editor turns off the system reset by the appropriate bit in the system flag in PAD, and so allows other VDP interrupt functions such as screen blanking after a period of no activity to continue normally, and now the print spooling as well. We shall have to find out from someone else how it is affected by a hung PIO. The Myarc documentation is less than informative about the details. In fact the booklet follows the disastrous TI tradition of providing essentially no technical information beyond Basic programming advice, and much of it is a reprint of the TI disk manual, there being only one short section of hints for the Assembly programmer. The card appears to have a connector for an external power connection to the card but this is not documented, even by a inserted leaflet. Maybe this is just that they haven't got around to it, but the possibility remains that it doesn't work properly or even is a danger to other cards in the box. I just wish they would tell the story and give the necessary specification for the external supply if it is in fact usable. At least one well known program, DSKU does not react happily to the presence of the Myarc ramdisk.

What I have done is set up a special disk of FUNNELWEB and utilities configured to run from DSK5. At the start of a session the Myarc DM is loaded, either from Basic using the disk card CALL ILR and LR (they really are a bit more useful than I gave them credit for in an earlier article, but from command mode only) or else from E/A which is our normal module. This is then used to set the ramdisk to DSK5 and then to do a file copy of the disk into the ramdisk. Then LDFW is used to load FWB from the Myarc DM and we go from there. FWB and the RAMdisk are an ideal combination. It really is a flying finger exercise to switch from Editor to Assembler and back again. Assembling a program is a disk intensive activity and gives a handy speed comparison. The main FWB program used to take about 8 minutes on the TI controller in uncompressed object format so that it could be loaded with XB's CALL LOAD. Using the Myarc disk controller card with DD disk format and compressed object code (loadable with CALL LR) reduced this to under 4 minutes, and using the ramdisk has now brought this under 2 minutes. Apart from reduction of the tedium of the edit and assemble cycle the best thing is that you don't have to listen to the disk drives grinding away.

Currently the competition for Myarc is from Horizon and Corcomp. The Horizon looks to be the least intrusive and most flexible, being reprogrammable and using only the DSR portion of the memory map. It also has battery backup built in for its low power static rams. The only disadvantage apparent at the moment is that it is limited to DSSD equivalent size until the price of higher capacity static RAMs comes down to more reasonable levels. Also David Romer writes letters back, in contrast to my experience with Myarc so far, and has a very good engineering sense of what and how things should be done with the TI 99/4a system as it was bequeathed to us here in the orphanage.

Early reviews of the Corcomp 512K were not very encouraging, as there were a lot of firmware bugs apparent, and incompatibility with standard directory routines. It does have its own built in directory routine as does Myarc (CALL RDDIR) also, but what is really needed is the disk manager built in to one or other of the ramdisk or disk controller. Corcomp did have the good sense to give it the default device name DSKR. which is much easier to live with than Myarc's choice. I have heard from a couple of Corcomp owners since then and they seem rather more pleased with the device. From Australia I would wait until they had got their act together a bit more. I suspect each device has its own strengths and weaknesses.

It was recently pointed out to me that a spider is not a bug, and that this column should be called Arachnid Corner or some such name. Well, I was conscious of the distinction from the start, but this series started out being about bugs, not written by bugs, so the name seemed entirely appropriate for the subject matter. Still does too. Long daylight savings evenings have been spent clearing out the jungle that had grown up during the preoccupation with programming Funnelweb. Lots of the little critters' holes out there where the Tradescantia grows -- over everything. No punishment is too cruel or unusual for the moron who introduced that noxious weed to the area. The same goes for privet too.

First I'll continue with the hardware review from last time. We have accumulated more experience with the Myarc 512K RAMdisk card and now are running a HORIZON RAMdisk as well, which arrived in from Bob Boone just after last month's article was written. That makes it a computer with over 700 Kbytes of RAM and 130 Kbytes of ROM that I'm using right now. That's right - my lil' old 99/4a. It could take 3 more Horizon cards for another 576 Kbytes without either physical or logical distress. I would happily have one more, but 3 more would be sheer excess.

Word is filtering in on the auxiliary power situation for the Myarc - the undocumented socket at the rear of the card. My summary of the San Diego experience is that unless you do it properly don't bother. Let's interpret that.

Initial inspection showed the the external line coupled to the on-card regulator through a diode for reverse isolation. The obvious source of power is a plug-pack in a mains socket which remains switched on. This has the problem that there is very little filtering in such gadgets of rubbish on the mains - just look at all the extra components TI put into the PE-box for this purpose. I think the real problem is that there is almost no protection against mains dropouts, as there are only small capacitors in the plug-pack and in the card. The card regulator would provide some protection against spikes, but is helpless if the supply drops out. So you either have to use a better supply or float rechargeable batteries across the supply as one San Diego user has done with success so far. This news is encouraging in that it means that it isn't affected by the PE-box bus. Myarc advise that the correct power up/down cycle should be observed. These aren't CMOS RAMs so a isolated battery would be good for only a short life without mains charging unless of substantial A/h capacity. Another solution to the short term mains drop-out problem would be a large capacitor, but I'm not sure what the rectifiers in a typical plugpack will handle and I wouldn't want to go squeezing more into the card itself.

Anyway we are not worrying too much about all that because we now have an Horizon card as well. How do they compare ? Installation is easy in either case if the card is the only one in the system. Only one Myarc card may be used, but several Horizons may be installed. If so, or if the Horizon is in addition to a Myarc card its address switches must be set. The circuit detail to note is that the Horizon uses CMOS memory. As news goes this is both good and bad. The good news is that the standby power requirements of these are so low that some on-card Ni-Cads are enough to retain memory without any worries about external supplies. The board can be removed, stored and replaced without losing its contents. (We know because our PE-Box self destructed (a shorted power rectifier diode) and the Horizon card contents were still there when replaced after repairs. Just don't remove it with PE-Box power on !!) The bad news is that they are considerably more expensive than dynamic RAMs of similar capacity. The Horizon card is limited to 180K (DSSD equivalent) by the physical size of the 8Kx8 RAMs. 32Kx8 are now available but are still expensive, and Dave Romer figures they will have to come down to $A12 or so to be viable in this application. They would certainly make the board less crowded, even if its present design limits it not much more than DSSD size.

Unlike the Myarc which bank switches the 32K memory expansion, the Horizon stays entirely within the confines of the assigned DSR space. It does this by bank switching the upper 2K of this space (>5800 - >6000) by CRU addressing. Since only 128 CRU bits are available for each DSR a little arithmetic gives the estimate above. On the Myarc scheme, 32K at a time, much larger sizes could in principle be handled, but it is much trickier to deal with from a programmer's point of view because it is switching the RAM in which the normal assembly programs reside. The Horizon in this sense remains truer to the original TI concept of peripheral devices sharing a predefined address block. The Horizon DSR is also in CMOS RAM, occupying the lower 6K and some of the 2K blocks. This means that the programmer can alter the DSR or install new versions from disk , and complete source code and auxiliary programs are provided. A welcome contrast to the traditional TI approach which Myarc also follow. This is why I said 2 Horizons would be quite a thinkable prospect. Once you have one there is the temptation to experiment, but neither do you want to surrender the existing RAMdisk function. The card has DIP switches which allow the CRU base address to be set. The normal disk controller resides at CRU base >1100. The first address >1000 is normally vacant (I think TI used it for production test equipment according to the Technical Manual) and is usually taken by RAMdisks because it gives the quickest response. The Myarc device hogs this address, while the Horizon can be switched to any vacant CRU base. So ours is set at >1200.

So how does it handle. Basically like a charm. It is a little slower than the Myarc because it uses VDP in the same way as the TI disk controller, but both are so much faster than physical disks that it hardly makes any difference. So far neither has lost any data because of a machine lock-up. I have identified only one bug so far and this I expect will soon be fixed. This stems from the TI DSR system design which never envisaged that there could be more than one disk card in a system. As you recall it works, the DSR LiNKing process that is, by searching through the DSRs until it matches the device name being requested. Now we accept the limitation that if two devices call themselves DSK1. that the one earliest in the search path is the one that will always end up doing the work. The problem comes with routines not normally visible to the Basic programmer which of necessity have the same name for all disk cards, such as sector read or write. The present problem with Horizon is that it doesn't scream loudly enough if asked to do one of these, but at a disk number other than the one it is currently emulating. This means that sector reading routines unless specifically designed to recognize the Horizon currently terminate at the first Horizon card in the system, whether or not they actually worked properly. The special routine used in the Horizon auxiliary programs works fully but is too lengthy to use in FUNNELWEB and assumes all sorts of internal details of the Horizon ROS. The current issue of FWB has routines that will find the first Horizon card in a system, and will find all of them when the Horizon ROS (RAMdisk Operating System) is updated. Edgar Dohmann of the JSC UG in Texas has run into the same problem in developing Super-Forth, as TI-Forth makes heavy use of sector accessing.

Now these comments could equally well apply to the Myarc, but the designers of that had thought about and circumvented it in an entirely transparent fashion. I'm somewhat chagrined that we had the Myarc operating for a couple of weeks without realizing that there might have been a such a problem. I guess that's some sort of tribute. What happened was that as soon as the Horizon was installed at CRU base >1200, sector accessing routines like QD, SD and so forth just ignored it. A little detective work revealed the sector routine common name problem, and also that even physical disk sector access was occurring at the CRU base >1000 for the Myarc card rather than at the disk controller CRU base of >1100. I haven't gone into it at all but I expect it is doing the same sort of thing as Will found last time for the print spooler function, namely going out into its own private 32K bank and

driving the regular disk controller from there, returning all error codes correctly. This method is all right as far as it goes, but there is no way that the Myarc knows about any other disk emulators at higher CRU bases. It could of course search for them in the same way that FWB now does, but its DSR is in ROM so there is no way of making changes when it is found to be inadequate for purposes beyond those it was designed for. The Myarc does appear so far to do correctly everything it was intended to do, but its designers never allowed for more RAMdisk cards in the system. A shade arrogant perhaps, but no program can ever take care of every eventuality.

The signals we get on the new Myarc computer continue to be mixed. News from Ottawa dated a couple of months ago was that costs had been cut by redesign and the operating system was bug free. Initial production run was to be for 2300 with 1000 sold already. The Nov 86 Ottawa Newsletter says one was received there, but without any operating system as yet. The most recent letter from the US said it was still not out, and that word was that the DOS was rather skimpy with some big drawbacks, details unspecified. Look how long it took for software to appear that used the 99/4a to anywhere near its potential. You might say that was because of TI's self defeating secretiveness, but if you find consolation in that thought remember that Myarc seem to be just as secretive about details as TI ever were. That doesn't augur well for a machine that by its nature will sell only in limited numbers to the existing 99/4a user base. I certainly don't think we are interested in the machine at all if its architecture is not completely open. And if only the same proportion of Myarc owners respond to "fairware" as ever did of 99/4a full system owners, then there will be very little incentive to add to the curiosity. A liking for TI 9900 family assembly code can stretch only so far when it comes to spending real money on a successor machine.

A simple hardware modification I have done on E/A and XB cartridges is to add a reset button. All that is involved is to install a pushbutton switch on the name-plate end of the cartridge. Essentially it does what the reset button on the Widget does, but then you don't have to have that thing in the road of your right hand. The act of inserting a cartridge into the console slot causes a computer reset. The 99/4a was always designed so that you could hot plug cartridges(unlike some other machines) and the purpose of the reset was to make sure that the GROM address in the cartridge was reset or the machine would lock up. We also all know now that the cartridge slot is the weak point of the machine and should be used as little as possible, so that removing and re-inserting a cartridge id not a good way to reset the machine. TI on the other hand did not supply any other form of reset other than switching off and back on (I believe IBM did the same with the PC at the behest of software manufacturers to make it difficult to find out what the software was doing). If you are developing assembly code where lockups are likely to happen a reset is much preferable to cycling the power because it gives you a chance to find out what happened.

The cartridge reset is by a resistor which discharges an RC network connected to the 9904 oscillator chip. This recharges when the cartridge and its resistor is removed (it takes a while which is why rapid re-insertion can fail to reset the machine). What you need is a momentary contact normally open spst switch. I used Tricky Dick's S-1102 plastic push button type which I had around. These are as physically large as can be tolerated, and I filed the ridge flat on one side so it would fit closer to the top of the cartridge case. Mount the switch in a hole made in the front of the cartridge, making sure it doesn't foul circuit board components. This is a lot trickier with XB. Solder two wires to it and wrap with insulating tape. On the circuit run from the end contact on the lower side of the circuit board (in E/A, a GROM only cartridge there are contacts only on the lower side) you will find a resistor, probably 100 ohms. Lift one end of this resistor and solder the wires from the switch to the free end and the hole it was lifted from. Use a little more insulating tape and carefully reassemble the whole thing. If you have mounted the switch properly there should be no unexpected mechanical interference. I first did it to a Carwars module which had had a brain transplant with a E/A GROM sent over by a friend in the US of A, and then I did it to the spare XB module. These are the two that we are likely to lock up the machine with.

We now also have a "Cache Card" made by Tom Spillane in San Diego. This has a E/A GROM and an 8K CMOS RAM with lithium cell battery backup. Very handy, either for FWB with CTRAM or for DEBUG or SBUG6. I gather that a model with CRU addressable 32K RAM is to be introduced by DataBiotics in California. This could make an interesting combination with the Myarc 512K card used as memory expansion. This same outfit has brought out Super-Forth by Edgar Dohmann and friends which loads in the 8K. It is basically TI-Forth revamped with all that extra space available for the user program.

A little bit of bug-squashing to fit the title of the column before we leave. There is a bug which can be fatal in the first issue of Vn 3.4 of FUNNELWEB. This can cause a crash because the system thinks GD is loaded when it may have been overwritten. Use Disk-Patch to look at the sector >10 after the start sector of LOAD (if LOAD starts at sector 1ED then it will be found on 1FD or very close by and comes about half a sector after the ASCII messages cease, the last being DSK0.UTIL). Look for the hex word FF20 (about a line or two after 9FFF which is easier to spot). The word preceding FF20 should be C810. In LOAD these may or may not be word aligned, depending on the length of the XB part of LOAD. Either make a similar correction to UTIL1 or more easily run UPATCH as prescribed in the documents. Before doing any of this check your FWDOC/REPT file to see whether it has already been attended to.

That should be enough for now. A merry Xmas 86 to everyone and may you get at least as much fun during the frustrations of our TI-99 computer hobby in 87 as you did in 86.

TONY MCGOVERN
FUNNELWEB FARM

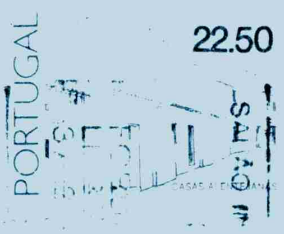
Funnelweb.V.3.4

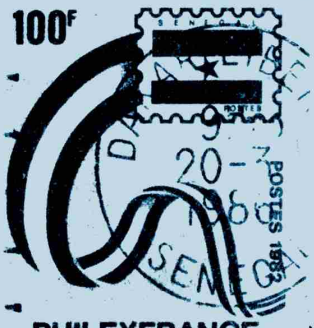
FUNNELWEB V.3.4 is now available in 10 living colours, no spiders, and, as an extra bonus, a modified DISK EDITOR with auto-repeat cursor, Byte Record counter to assist in easier hacking of your disks as well as DIRECTORY as Option #3.

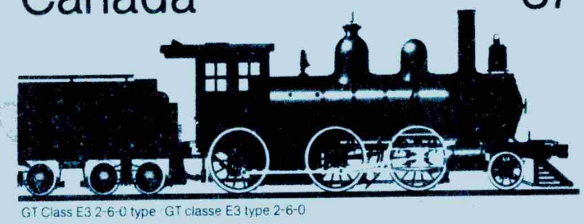
Lots of other refinements make this the Number One, most USEFUL UTILITY as far as the web has spread and it has far outgrown it's early XBasic TI-WRITER loader origin.

Just in - DM1000 V3.5. A stand alone version at the moment but including all the documentation to get the full use of this fantastic FAIRWARE disk from the Ottawa 99'U.G.and Bruce Caron.

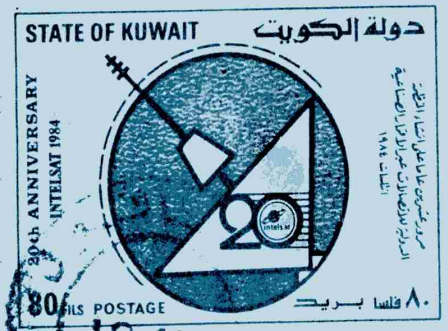
This page is blank



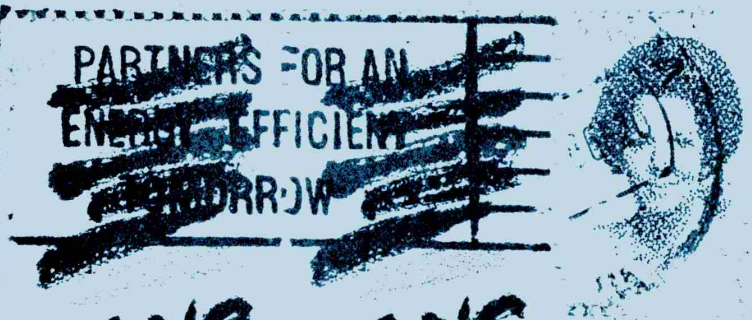


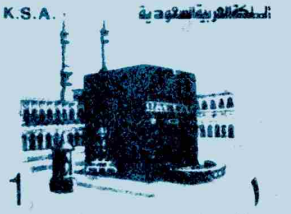


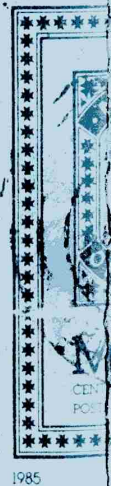
PHILEXFRANCE
PARIS-82
SENEGAL



TI 99/4a EXCHANGE,
UK TI User Group,
40 Barrhill,
Patcham, Brighton.
BN1 8UF
(Tel. 0273-503968)







LOVE
USA 22

LOVE
USA 2

TIA9-9A EXCHANGE UK

SINGAPORE 75c