CALL SOUNDS

The Newsletter of

The CENTRAL WESTCHESTER 99'ERS

JUNE 1986

ANNOUNCEMENTS:

Next meeting: Thursday June 19th at the American Legion Hall, 50 Broad St. Hawthorne, NY at 8pm sharp.

Business: Election of officers for 1986/87 season. Candidates are: President- Carney W. Mimms , Vice Pres/Program- Al Trudeau, Secretary- Charlie Willoughby, Treasurer- Art Byers.

Program for June 19th; Auction, buy, sell, swap night. Time permitting, Question Answer session and a short session on TI WRITER.

IMPORTANT: all members who have posession of any of the DBT beta test products **MUST** bring them to the June meeting. If you can't make the meeting, phone Carney or Art.

DUES: If there is a red circle on your mailing label, you owe dues for the 2nd quarter. They must be paid by the end of the June meeting or you will not receive the July Newsletter.

The **LENDING LIBRARY** cannot work well unless you return your takeout after one month. Some of you have kept modules or newsletter binders for several months. This is unfair.

MAY MEETING review: business accomplished at the May meeting included voting to buy disks to backup the library, the formation of at least two SIG's - Telecommunications and Xbasic. As only two people signed up for the Word Processing SIG, We will try to schedule some short wp sessions at regular meetings.

Also at the May meeting, some club tasks will have new chairmen: Robert Amenta will be the Lending Librarian and Ed Borneman will be in charge of the Hospitality at meetings.

SIG MEETING time and date: The **Telecom** SIG will meet at 8 pm, Thursday June 12th, at the home of Carney Mimms, 69 Tanglewylde Ave, Bronxvlle. 914-961-5993

The **XBasic** SIG will meet at 8pm, Tues June 12th, at the home of Al Trudeau, 7 Taylor Road, Elmsford 914 592-2080

DISK SOFTWARE for the June meeting will be FUNL WRITER. Please phone Bob Sweeney, 914 337-1660 (work) or 914 961-8024 (home) and reserve a copy. You must bring a blank disk to the meeting to exchange. Funl Writer is Fairware. If you like and use it, you are expected to pay the author.

DISK LIBRARY RECOPIED

We have replaced the main disk library that was lost due to the disappearance of Steve McCalla -which points out the adviseability of having backup copies.

Those desiring major portions of the library, please contact Bob Sweeney.

MYARC PES shown at May meeting

Hubert Deri showed us his new (as of the beginning of this year) MYARC Peripheral Expansion System. All in all it was pretty impressive.

First, it is about a half the size of the TI box and is QUIET because it has no noisey fan. But the really good news is the two half-height DSDD drives holding 1380 sectors each. Hubert fits on one disk what it takes most of of us four disks to store.

The system plugs into the side port of the console and contains 32k Memory (we understand in can be upgraded to 128K) plus I/O's for PIO and RS232/1 and 2 - and of course the disk drive controller.

It uses all standard TI modules (It comes with TI's Disk Manger II module). The disk OLD and SAVE times seem to be much less than those of the stock TI drive in the PEB. This is not only because double density is a faster read and write, but the system disk controller is faster.

This is definitely a much more "portable" set up than the big heavy PEB. If you combine this with one of the new smal color or B W tv's as a monitor it will have a very small footprint on your desk.

All TI Basic and XB programs run with no problems. Unfortunately, there seems to be some incompatability problems in loading assembly programs from XB. Huwever, we did load the TECHIE BBS on to it and did Access Compuserve using FAST TERM, so these problems look solveable.

Hubert!! **THANKYOU THANKYOU** for the demo.

IN THIS ISSUE

INSTRUCTIONS for June 19th Auction

TAGGING: Mark all items that you wish to have auctioned with your name and the **MINIMUM** price at which to start the bids. If there are no bids at that price, you will be given the option to lower your starting price. If you are offering items as a bundle, for example: a set of books or magazines, try to tie or bag all the things together, or mark the items as being part of a bundle.

PAYMENT: We suggest you bring cash or a check book. We frown on I.O.U.'s.

COMMERICAL SOFTWARE: **ORIGINAL** copies of cassettes or disks only. No backup copies of any commercial software will be accepted for auction. NO EXCEPTIONS!

PARTIAL LIST OF CLUB PROPERTY GOING ON THE AUCTION BLOCK:

DATABIOTICS NAP: DISKMASTER I Disk Based disk manager with ability to Read/Write to disk. MINIWRITER II and MINIWRITER III modules. TWO modules- SUPERSPACE -E/A with extra 8K of battery backed RAM. Each comes with a disk of demo programs and utilities. 4A/TALK Disk based telephone communications program.

CLUB OWNED ITEMS to be auctioned: Items from the club library that seem to have made all the rounds: HCM Magazine Vol 4#5 with cassette of programs to match. HCM Vol 5#1 with cassette of programs to match. HCM Vol5#2 with cassette to match. HCM Cassettes only WITHOUT magazines: V4#1, V4#2, v4#3, V4#4, V5#3, V5#4, V5#5.

CASSETTES - Each bound in a padded vinyl booklet: Teach Yourself Basic, Teach Yourself Extended Basic, Giant book of Basic Games, Giant Book of Extened Basic Games.

MODULES: MUNCHMAN, TERMINAL EMULATOR II

Current Officers:
PRESIDENT - Carney W. Mimms 914-961-5993
VP/Program - Art Byers 914-528-5402
Treasurer -Kathy O'Brien
Committees:
Equipment Manager -Pat Leigh
Hosptality -Ed Borneman
Lending Librarian -Robert Amenta
Disk Librarian - Bob Sweeney
Backup Librarian- Art Byers
Cassette Software -Hubert Deri
Newsletter -Art Byers

All the articles published here, except those reproduced by photocopying an article from another club newsletter, are avialable as DV/80 files, on disk. just provide a blank disk and postpaid return mailer with your request. Other 99'er clubs are welcome to reprint any material in these pages. Please give credit to the original source.

We currently exchange newsletters with some 44 clubs plus a few people on the courtesy list. The newsletters we receive are 3-ring bound each month and go into our lending library. WGe hope all you editors out there will leave sufficient left margin for hole punching.

TELECOM SIG

Choosing a terminal emulator program.

COMPARING 4A/TALK and FAST TERM and TE-II

We 99'ers are very fortunate in having two excellent terminal emulator programs available for our computer: The fairware program FAST TERM and the commercial 4A/TALK.

The prices are comparable. Paul Charlton, the author of Fast Term asks $15.00 but you supply the disk and must print out your own instructions. 4A/Talk comes with a very fine clear and professional printed booklet and, of course the disk is supplied. 4A/Talk has some rather sophisticated protection on the disk, but the publisher, DataBioTics, will replace the disk if anything goes wrong.

In all honesty, either will do the job for you and do it well. However, a little comparison cannot hurt.

What about TE-II?? Well if you have only cassette storage an a stand alone RS232, this module is the way you are forced to go. It will enable you to log on to BBS's and infomation services, exchange information with outher TI computers, but its capacities are limited. It will, however, access the speech synthesizer which no other emulator is able to do.(7)

FEATURE		TERM	TALK	TE-II
Capture buffer dump to disk		9.5k	8k	no
printer spooler, print while reading		8k	8k	no
Disk Directory		yes	yes	no
HELP screen		no	yes	no
Use without smart modem		yes	yes	yes
keyboard dial w/smart modem	(1)	yes	yes	no
Auto dial w/smart modem from list		no	yes	no
Send message prepared in advance	(2)	yes	yes	no
ASCII text capture	(3)	yes	yes	yes
TE II protocol		yes	yes	yes
XMODEM protocol		yes	yes	no
Configure default file	(4)	yes	yes	no
Change configure while running	(5)	yes	yes	no
On screen call timer	(6)	yes	no	no
Speak the screen content		no	no	yes

(1) FT would only dial out on Full Duplex, 4A/T on both Half and Full.

(2) My personal preference was for Fast Term's way of handling this feature.

(3) TE-II will only screen dump, 1 screen at a time.

(4) With Fast Term you can set up a separate configure file for each host or BBS simply by giving it a different file name. 4A/Talk could only use a file named CONFIGURE and you had to call up the configure screen and redo it if a BBS called for different defaults. FT is better here, by far.

(5) 4A/T recalls a whole screen of defaults, FT can change one default at a time with a combo of key presses.

(6) FT's on screen timer stops, for obvious reasons, while reading from or writing to disk. Timer is a guide more than an accurate clock.

(7) Bill Wright, Sysop of TURBO TECHIE BBS in New Jersey, tells us he has four or five "regulars" accessing his BBS with TE-II and Cassette.

FAIRWARE LIST 5/15/86

The following Fairware disks are available on request from BACKUP LIBRARIAN, Art Byers, 914 528 5402. Remember if you like and use the program, you must send the requested money to the author.

Where needed, all programs have documentation on the disk. If two diskettes are required, Fairware will be supplied as a "Flippy". If you require a particular brand of disk, you must supply it in advance.

2-D Graphics by Jean-Pierre Morin. Version 3.0. Written in FORTH. A color drawing program.

BACKUP - by M. Ballman. A true track copier. requires **two disk drives** and E/A module or BEAX.

BEST SONGS by Bill Knecht - a remarkable disk of 99/4A MUSIC.

BEST HYMNS by Bill Knecnt - Your all time favorites, beautifully played and displayed

C-99 Release #1. the "C" language for the 99/4A. We understand a much improved release #2 is available. $20.00.

CATLIB by Marty Kroll - a FINE quick disk library cataloger. Produces both an alpha list as well as list of disks.

COMPACTOR/UNCOMPACTOR - by Monty Schmidt. Takes an uncompressed Assembly file and will compress to about 2/3 disk space and yield faster load. UNCOMPACTOR - opposite of above.

DIRECTOR - by Ron Rutledge A database that allows cataloging disk-based library. This takes a different approach than CAT-LIB

DM-1000 v.3.1 - BY the Ottowa Canada UG A disk based Disk Manager which rivals CorComp's. Fast easier to use than DM2 module.

DISASSEMBLER - by Marty Kroll. A multifeature all Assembly program to read M/L code.

DISK ENVELOPE MAKER - by Trio Software. Allows you to enter description and comments about each program on the disk, and then prints out a disk jacket with catalog and comments.

FAST TERM - by Paul Charlton An Excellent Terminal Emulator for your modem. Dump to Disk or printer. Transfer files in ASCII, Xmodem, or TE II protocols and much more.

GREEN'S UTILITIES: 5 useful programs. 1) DISKCOPY (sector copier) 2) DISK INIT (Initialize a disk sssd or dssd 3)printer set up just what it says 4) TERMINAL makes your computer into a terminal for 300/1200 baud

HBM PRINT - by Bob Lawson Ed/Assem required. Written in FORTH. Auto loads from option #3 Load Run. Prints out hard copy from files of the Home Budget Manager Module.

HOME FINANCE by Fred Guyton - Three programs: Checkbook Manager, Home Budget, and Home Financial Decisions. all well done.

MAIL CALL By Gary D. Watts - A data base for Address and mailing lists. Sorts,prints, etc.

MAIL LIST by - Fred Guyton - A data base for mailing lists, sorts, prints etc.

MASSCOPY -BY Steve Lawless. A three pass sector copier for 1 to 3 drives. Will make 2 copies if you have three drives.

MASS TRANSFER by Stuart Olsen A terminal emulator program with the ability to transfer a whole disk via telephone 300/1200 baud.

MONTE CARLO - An authoritative, authentic reproduction of the game of Roulette as played in Monte Carlo.

NEATLIST - by Danny michaels, XB. Quick easy reference to variables and line #'s of your programs. If you are an XB programer, you **must** have this

PRBASE v 1.3 - by Mr Warren. Touted as perhaps the best data base to date fo the 99/4A. Version 2.0 is in work with additional features.

RAPIDSCROLL -by Jurgen Switalski. A fantastic program to read DV80 files. chioce of 40 col or 64 col. scrolls up, down, right left. Very fast, very good.

SCREENDUMP -by Danny Michaels 2 versions #1 Works with Epson compatable printer, #2 W/ Prowriter. Features double size or single size, vertical or horizontal page printout.

SIDEWAYS by Mario Tomietto - prints sideways so you can print out a spreadsheet w/o pasteup. Not a screen dump.

SUPERBUG II by Edgar L. Dohmann A debugged and improved rewritten version of TI's debugger/disassembler. Also has a version for Superspace or Supercart modules.

SUPER COPY - by Tom Knight. Disk duplicator by sector copy. Allows input of start and stop sector number. 1 or 2 drives.

TECHIE BBS - A Bulletin Board system for the 99/4A. Minimum: two sssd drives or equivilant. Has capability for XModem U/L and D/L

TRIVIA99er - by Robert Wessler; A TI version of the famous game. Very well done! America's second most popular recreation.

TURBOCOPY -by Barry Boland. Backs up a SSSD in less than 30 seconds. Requires **two drives** and E/A module or BEAX

UNIVERSAL DISASSEMBLER - by Rene'LeBlanc . A total of nine utilities including one that will dissassemble off of a disk!, search a disk for a string, read/write sectors, display directory,locate fractured files, VDP CPU memory dumps, DSR dump disassembler. Plus a help file.

WEATHER FORCASTER - by Gary Cox, Just what the title says, **Plus** a free arcade game and many other goodies on the disk, including a cassetted based data base.

X_DISASM - by Fred Hawkins. An XBasic disassembler. If you are willing to pay, excellent documentation is available.

WHERE TO GET HELP

Sometimes you need help outside of the meetings. Here are some of the club members who can aid you:

XBasic- Art Byers, Al Trudeau, Multiplan - Ted Mills PASCAL - Nils Solderman, Bob Sweeney, General DOS problems, software problems - Art Byers, Al Trudeau, RS232, Adding Disk Drives,- Carney Mimms, Art Byers, Al Trudeau, Telecommunications- Bob Cataldo, Carney, Art.

An updated club roster, names addresses and telephones will be given out at the JUNE meeting.

TMS 9900 ASSEMBLY LANGUAGE TUTORIAL
PART 5-DSR's and PAB's

by STEVE ROYCE WNY 99'ERS

File handling is not quite as easy in Assembly as it is in BASIC or Ex-BASIC, but it really isn't too complicated. The main issues involved are setting up a Peripheral Access Block (PAB) to establish the attributes of the file, writing the PAB to an address in VDP, manipulating the bytes in the PAB to accomplish specific functions, reading from and writing to a buffer in VDP and invoking the built-in Device Service Routine Link (DSRLNK) utility. These functions, in and of themselves, are a lot to cover; and, once we start trying to establish some order in the handling of files, it becomes even more lengthy. So, we are going to cover file handling in at least three articles, each one building on the previous articles until we finally arrive at a(in my opinion), an efficient set of our own utilities which can be used to make file-handling a lot easier.

This month and next, we will deal with data files to be stored on disk. Since I am a firm believer in the value of RELATIVE data files, even though they may take more space on a disk, I will only deal with RELATIVE disk files in these articles. Once you understand RELATIVE files, SEQUENTIAL files are even easier. It's a lot like learning to drive a car with a standard transmission, then the automatic transmission is easy.

In your BASIC and Ex-BASIC manuals, you will read that INTERNAL data format is more easily interpreted by your computer. My first attempt at creating a disk file in Assembly was therefore to use INTERNAL format. I tried for weeks to create a FIXED, RELATIVE, INTERNAL file, but, every time I attempted to simply open then close a file with these attributes, then switched to DISK MANAGER to see if it existed, it WASN'T THERE!!

I modified my PAB, changed buffer addresses, put in the Assembly equivilant of a "CALL KEY" to check my work at every step, used the Assembly Super-Debugger--you name it, I tried it. It took quite a while for me to finally realize that, in Assembly, if you want INTERNAL format, you must do the conversion from DISPLAY to INTERNAL yourself. Sorry, but I'm not about to try that yet, so we will stick to DISPLAY format.

PAB BYTES

Your E/A manual, pages 293 to 299, does a reasonable job in explaining the function of each byte in a PAB, so I'm not going to cover that ground in this article. Review those pages and note that, since our file will be a DISPLAY, FIXED, RELATIVE file that our choices for byte 1 of the FAB are >01, >03 or >05.

STRUCTURING

BASIC, Ex-BASIC and many other languages are designed around a well structured set of standard routines and addresses which are dedicated to specific purposes. You would never know it from your BASIC manual, but the internal addresses for every function are defined and, for the most part, inflexible. Assembly defines some specific addresses such as the screen image table, some CPU addresses, the memory mapped devices, but leaves you to your own creativity for the balance of the program organization. What I hope to end up with in my Assembly programming is a well organized yet flexible set of routines which I may incorporate into any program that needs them. In this way, once my subroutines and addresses are established, a lot of custom structuring for a specific program becomes unnecessary. I think it's a wonderful idea.

Let's structure an area of VDP RAM for PAB's and a read/write buffer area. Remember last month we started our sound table at >1000. I have reserved a block of >500 bytes for my sound table, based on the fact that the song and sounds in TOMBSTONE CITY take >03CC bytes, and that I may want a few more sounds in my programs. So, my VDP area for PAB's will start at >1500, and I have reserved >0100 bytes for PAB's. That should be enough to set up eight PAB's, which should be enough for any program I can imagine. The next block of VDP, starting at >1600, will be used as a read write buffer area for the PAB. Since the maximum record length in a file is >FF bytes, I had initially set aside a block of >100 bytes. However, I have expanded that to >200 bytes. I'm not sure why yet, but I'll make use of that extra >100 bytes somehow. So, our read/write buffer area is from >1600 to >17FF.

Remember, all these areas are still flexible in use. If I don't use all >500 bytes of my sound table, I can allocate them for other use. But, I at least have the ability to use a structured and organized set of pre-defined areas of VDP..

Our structured VDP table so far looks like this:

>0000 to >02FF	SCREEN IMAGE TABLE
>0300 to >037F	SPRITE ATTRIBUTE LIST
>0380 to >039F	COLOR TABLE
>03A0 to >03FF	UNUSED
>0400 to >07FF	SPRITE PATTERN TABLE DEFAULT
>0780 to >07FF	SPRITE MOTION TABLE
>0800 to >0F7F	PATTERN DESCRIPTOR TABLE
>0F80 to >0FFF	UNDEFINABLE CHARACTERS >F0 TO >FF
>1000 to >14FF	SOUND TABLE
>1500 to >15FF	PAB DEFINITION AREA
>1600 to >17FF	READ/WRITE BUFFER AREA FOR PAB'S

Before we get into the routine to open and close a file, let me offer the following suggestion. Edit and assemble the code, then make a copy of the object code using DISK MANAGER. Use that disk for the LOAD AND RUN. If anything should go wrong, you won't have destroyed the source code. After you LOAD AND RUN the object code, use DISK MANAGER again to see whether the file 'DSK1>FILE1' has been successfully created. Next month, we'll write a record to the file.

TURBO SPEECH

by Stephen Shaw

Excerpted from the TI99/4A Exchange TI*MES of Great Brittain. Issue #6 Autumn 1984.

Now onto something really juicy, SPEECH. Old hat Huh? Well, this information will give you speech in TI Basic with the Minimemory or if you have XB +32k will give you speech just a mite faster than CALL SAY which slows programs down no end. (Should also work with TI Basic and Ed Assem or Super Space or Super Cart.-ed). For this information I am indebted to Neil Lawson who has been delving.

Program framework for timing purposes:

```
20 CALL INIT
30 S=-27648
100 FOR I=1000 TO 100 STEP -100
102 PRINT I/10    (countdown mod by ed)
104 NEXT I
110 PRINT "START"
120 FOR X=1 TO 20
130 REM INSERT TEST ROUTINE HERE
140 FOR T=1 TO 30
150 PRINT ">";
160 NEXT T
170 NEXT X
180 PRINT "END......."
```

This standard routine sets up a framework to test our new routine and gives a basic time reference. Note: times quoted are for my system. Yours may be different, but the ratios should be similar.

Running the above program with the loop in line 140 running 30 times as shown, takes 10.7 seconds from "START" to "END". Change line 140 to loop just 20 times and the timing is 12.7 seconds.

Now we can insert two possibilities. The first is available only in XB:

```
130 CALL SAY("#THAT IS CORRECT#")
```

Run the program again. If line 140 is looped 20 times, the time is 44 seconds. If Line 140 is looped 30 times, time is 50 seconds, a 6 second difference.

The time for speech is constant, it adds about 21 seconds to the program. Now for something different, (Also works with Minimemory)!!!:

```
130 CALL LOAD(S,70,"",S,65,"",S,72,"", S,70, "",S,64,
"",S,80)
```

If you now run the program, it says the same thing as many times but look at the timing! If line 140 loops 20 times it takes 26.3 seconds, looped 30 times takes 26.5 seconds.

We know that looping 140 and extra 10 times adds 6 seconds. Where have those 6 seconds gone?

The CALL SAY routine holds everyting up until it has finished speaking, but using the CALL LOAD equivilant, while the computer is speaking, it gets on wit the next chore too. The "dead time" is used and soaks up those 6 seconds.

Thus using the CALL LOAD equivilant, the computer speaks faster and also permits your program to run more quickly if there is is work for it to do between speech outputs.

That's the clever demonstration! (Impressed?) Now for the theory now that you're interested!

References: Ed/Assem Manuel: pages 351,355,422,427. (Errata: the reference in Para. 1 note page 355 should be to section 22.1.4, not as printed in the maual)

Address -227648 is the SPEECH WRITE address. We keep on feeding it with bytes, and in due course the computer speaks!

The bytes to load to that address are found out as follows:

First, decide what you want to say from he standard vocabulary. Then look in the table (pp 422-427) for the address of that word or phrase. "THAT IS CORRECT" is given as 6816. That is Hexadecimal, not a decimal number. The four numbers are reversed and become 6186. Now we offset them by Hex 40 and feed them in. As we are dealing with wit decimals with our CALL LOAD, that means that we ADD decimal 64 to each digit in turn:

```
(6+64)  (1+64)  (8+64)  (6+64)
  70      65      72      70
```

If the numbers were Hex A-F these have a decimal value as follows: A=10 B=11 C=12 D=13 E=14 F=15.

Now we must indicate end of word by loading a zero, again offest: Thus 0+64=64. Finally, we must instruct the computer to speak by loading Hex 50 (decimal 80). Thus we have loaded, in order 70, 65, 72, 64 and 80.

Check back to the listing. NOTE the way CALL LOAD has been used: a single command to load the same address with several different values.

To assist your experimentation, here are some HEX addresses from the manual. Remember to reverse them, translate to decimal and offset!

TEXAS INSTRUMENTS	6696	THAT IS RIGHT.	68FE
WHAT WAS THAT	77E9	READY TO START	56B3
YOU WIN	7DDB	AGAIN	17A5
ANSWER	1913	CHECK	1DB2
CHOICE	1DA2	COMMAND	1F1A
ELSE	28B6	GOODBYE	3148
HELP	3571	HURRY	3757
INSTRUCTIONS	39BD	I	3793
I WIN	37CF	JOYSTICK	3AED
NAME	47C0	NICE TRY	49A5

This is not only a useful programming aid in its own right, but also by demonstrating a part of the Editor Assembler manual's sometimes complex instructions, it should assist you when you are ready to move on to Forth or Assembly language proper.

XB USER WRITTEN CALL ROUTINES Part 1

By **Art Byers**

One of the big complaints about the BASIC language, in general, is its lack of structure. GOSUBs and GOTO's jump every which way. A collary complaint is that it is very difficult to follow the flow of a BASIC program unless there is a remark with each GOSUB or GOTO.

To illustrate how much easier it is to understand and follow a structured XB program using your own CALL subprograms, let us assume we have written some routines to convert the three most common number bases used in programming, one to the other. (Binary,Decimal, Hexadecimal, and Octal)

Here is part of the same program using these routines as written two different ways.

```
100 GOSUB 1000
110 GOSUB 1300
120 ON C GOSUB 3000,3250,3500,3750,4000,4250,4500,4750,
5000,5250,140
130 GOTO 100
140 END
```

```
100 CALL MENU
110 CALL CHOICE(C)
120 IF C=1 THEN CALL DECTOBIN
130 IF C=2 THEN CALL DECTOHEX
140 IF C=3 THEN CALL DECTOOCT
150 IF C=4 THEN CALL HEXTOBIN
160 IF C=5 THEN CALL HEXTODEC
170 IF C=6 THEN CALL HEXTOOCT
180 IF C=8 THEN CALL OCTTOBIN
190 IF C=9 THEN CALL OCTTODEC
200 IF C=10 THEN CALL OCTTOHEX
200 IF C=11 THEN END
210 GOTO 100
```

Obviously, I've stacked the deck a bit to make my point, but you can see how much more the listing means when you have written your own CALL subroutines.

TI EXTENDED BASIC, as shown, can be structured very well and made easy to understand with the use of user written CALL routines. A Few simple comparison examples follow.

Before going on, please review pages 180 through 184 of the XB manual.

First, let's take a look at two simple common routines used in many different programs: #1 A short delay routine. #2 A program to hold the screen while we read or make a decision. We will name #1 SUB DELAY and #2 SUB ANYKEY

```
1500 SUB DELAY(D1)
1510 FOR DELAY=1 TO D1 :: NEXT DELAY
1520 SUBEND

2000 SUB ANYKEY
2010 DISPLAY AT(24,1)BEEP:"<<PRESS ANY KEY>>"
2020 CALL KEY(0,KEY,STATUS) :: DISPLAY AT(24,19):CHR$(30)
:: CALL DELAY(75):: DISPLAY AT(24,19):" "
2030 IF STATUS=0 THEN 2020
2040 SUBEND
```

Line 1500 tells the computer where it can find the DELAY when it is CALLed. The delay sub expects a parameter value to be passed by its call. This could be a concrete value as in line 2020, a count of 75, or it can be a variable. The delay sub can be used any place in the program where a delay is desired, and by changing the value passed, be long or short as needed. Line 1520 is the equivilant of RETURN in a GOSUB subprogram.

Similarly, line 2000 locates the SUB for its CALL. We could have used ACCEPT AT instead of the keyboard scan, but unless "ENTER" is the key chosen, that would mean two keys have to be pressed - first any key and then "ENTER". The display of CHR$(30) and " " simple simulate a flashing cursor to let you know the program is waiting for your action. The delay between them is to slow things down enough for your eye to be able to see the flashon flashoff.

In the 99/4A world, perhaps the ultimate master of user CALL SUBS is Jim Peterson of Tiger Cub Software. If you are programming in XB or want to learn how to program in XB, Jim's two "Nuts and Bolts" disks are chock full of useful routines that are an education in themselves.

BE A CAREFUL SHOPPER!

As with anything else you need or want to buy, often careful shopping can save you substantial amounts of money. Here are a few examples

The TI video Modulator: listed in the TENEX everything book, latest issue at $15.50 plus shipping cost. Also available from your local Radio Shack at $4.95 plus local sales tax, via mail order from Lolir electronics for $3.50 plus shipping.

RS232 "Y" cable: From Tenex $49.95; from Lolir (listed as a 'Y' game cable) $4.00 plus shipping.

Volksmodem 12 Triton $69.95 plus shipping but includes cable. Tenex $59.95 plus calbe and shiping. Tex-Comp $39.95 includes cable add shipping.

CorComp stand alone 32k memory: Triton $99.95; Tex-Comp $89.95

Stand alone extra disk drive, in box w/ power supply: Triton $149.95 pluss shipping, Tex-Comp $129.95 plus shipping. Both Carney and Walter bought the same at the flea market section of TIOOFF for $70.00

We could go on, but you should have the point by now. We recommend that, on occasion s, you buy the Computer Shopper. It is a good guide and has many sources of ites we

ERROR TRAPPING - Part 2
by **Art Byers**

This article is a more concrete follow up to Ted Mills's overview article of May '86. I suggest you reread it before proceeding with the specific examples that follow:

Here is an example of a routine I use for debugging my Extended Basic Programs. Once they are running properly, the lines are erased.

```
1 ON ERROR 5000
4999 REM routine to locate errors and print on screen
5000 CALL CLEAR :: CALL ERR(E1,E2,E3,E4) :: PRINT "ERROR
CODE";E1:"ERROR TYPE";E2:"ERROR SEVERITY";E3:"LINE
NUMBER";E4 :: PRINT
5010 STOP
```

Obviously, the above breaks the program, and hopefully, after making a written note of the information I will be able to find where I goofed -HOPEFULLY!

Next, is the general format Ted describes for using ON ERROR in a program. First, the general error message that returns you to the main driving menu.

```
90 ON ERROR 6000
99 REM Main Menu
100 DISPLAY AT(2,5)ERASE ALL:"MAIN MENU" :: Etc
-----
5999 REM General error handle routine
6000 DISPLAY AT(10,1)ERASE ALL BEEP:"ERROR ERROR"
6010 DISPLAY AT(12,1):"1 = SAVE ALL DATA":"    AND CLOSE
ALL FILES":"2 = RETURN TO MAIN MENU":"3 = ABORT PROGRAM"
6020 DISPLAY AT(24,1):"ENTER CHOICE 1" :: ACCEPT AT(24,14)
VALIDATE("123")SIZE(-1):CHOICE
6030 ON CHOICE GOTO 7000,90,10000
-----
6999 REM routines to save all data and close all files
7000 Put save and close routine here end with a GOTO 90
-----
10000 END
```

Here is how to use ON ERROR to catch simple things like trying to read a file that is not on the disk or forgetting to close the drive door. OUTFILE$ might be something like DSK1.ADDRESSES. FILETYPE could be DISPLAY VARIABLE 80. In the case of Cassette storage the OUTFILE$ accepted in a prior INPUT or ACCEPT would be CS1 and the type could be something like INPUT, DISPLAY, FIXED 64.

```
640 ON ERROR 9000 ! to catch I/O and/or hardware errors.
650 OPEN #1:OUTFILE$, FILETYPE
------
------
8999 REM Disk or file error
9000 CALL ERR(E1,E2,)
9010 CALL CLEAR :: PRINT "ERROR CODE";E1:"ERROR TYPE";E2
9020 PRINT :"Check data storage device, check file name":
:"Press enter when ready" :: INPUT E$
9030 CLOSE #1 :: GOTO 90 !close file and go back to main
menu
```

This same routine could pick up a misspelled printer name from an INPUT - oh say you made a typo and put in RS232 instead of RS232.

The preceeding are examples of ERROR TRAPS, -catching errors and preventing the program from crashing. However, there is an old motto about an ounce of prevention being worth a pound of cure. You can prevent many errors by controlling how the program user can input information.

For example if you want only a "Y" or "N" answer to a prompt your program might read like this:

```
TI BASIC
230 INPUT "IS THIS CORRECT Answer Y/N":YN$
250 IF (YN$<>"Y")*(YN$<>"N") THEN 230
```

TI EXTENDED BASIC would use the plain English OR instead of the "*" sign. XB could also use an ACCEPT(n,n)VALIDATE ("YyNn") etc as per the forgoing example.

Both TI BASIC and EXTENDED BASIC could use the keyboard scan to prevent a wrong input. For Example:

```
250 PRINT "Is this Correct, Answer Y/N"
260 CALL KEY(3,KEY,STATUS)
270 IF STATUS=0 THEN 260
280 IF (KEY<>78)*(KEY<>89) THEN 260
```

Line 260 scans the keyboard. Line 270 says if the STATUS is equal to zero it means no key has been pressed and the program returns to the keyboard scan. Line 230 uses the ASCII value of the key pressed. N is 78 and Y is 89. Therefore if the key pressed is not one of those two, the program returns to the keyboard scan.

An alternative to line 280 could read IF (CHR$(KEY)<>"N")*(CHR$(KEY)<>"Y") THEN 260.

WOULD YOU LIKE SOME XB HOMEWORK? O.K. Write a program that reads ten numbers from a data statement, but only put 6 numbers in the data statement. Start the program with an ON ERROR XXX and write a routine that will keep the program from crashing and prints out "You tried to read past the end of the file"

TIPS FROM THE TIGER CUB

Are available at meetings. Check the "pockets" on the **Club Cork Board.** Tips #34 was given out at the May meeting. It contained some remarkable programming by **Jim Peterson.**

RS232 FOLLIES

BY Carney Mimms

As TI's support for our machine dwindles, it becomes more and more important to understand the hardware and to be able to adapt it to non-TI peripherals. (Ain't that the TRUTH!-ed). Fortunately, this is one area in the anarchic world of microcomputers in which there is actually an industry standard, of a sort. That standard is designated RS232 and is used by TI in communications between the TI 994/A and such external devices as printers and modems.

For most of us, RS232 means the RS232 card in the PE Box, although there are standalone RS232 Boxes around. The RS232 Card contains both types of commonly used device interfaces, parallel and serial, only one of which, the serial interface, is an RS232 connection. Parallel interfaces, used almost exclusively to connect computers to printers, pass data from computer to printer 8 bits at a time on 8 separate lines and have their own standards (again of a sort).

Serial interfaces, with which this article is concerned, pass data one bit at a time on a single line. This is both the serial RS232 interface's strength, and its weakness. Because the single bits must be reassembled into bytes which the computer can send and receive, serial transfer is inherently slower than parallel (though stil faster than most computers and other devices can make use of). However, since only one line at a time is needed to transmit data, serial interfaces can operate with as little as three lines, making it possible to use the telephone lines and other media which will not accommodate parallel transmission.

The RS232 serial transmission standard, which usually seems confused and arbitrary, makes a great deal more sense when its origins are known and a few simple principles are understood. These are nicely set out by Ted Drude in the Commodore (UGH!) section of the April "Computer Shopper." First of all, the RS232 standard was originally designed for connections between a terminal (usually a teletype terminal) and a modem, back in the Dark Ages before the microcomputer (say, twenty years ago) and not to connect anything to everything, its current use. Because of this, RS232 transmission is based on two Commandments.

FIRST COMMANDMENT: THOU SHALT NOT CONNECT MORE THAN TWO DEVICES TO EACH OTHER AT ONCE.

SECOND COMMANDMENT: ALL DEVICES ARE EITHER TERMINALS OR MODEMS, OR MUST BE MADE TO ACT LIKE THEM.

The second Commandment explains the two most widely used and frequently obscured pieces of RS232 jargon, DTE and DCE. DTE, or Data Terminal Equipment, devices are configured to behave like terminals, whereas DCE, or data communications equipment,devices are configured to behave like modems. Aha!, you cry, computers(using terminal emulation software) are DTE devices, modems are DCE devices, and printers ...? You see the problem already. Whenever two devices are joined together, one of them has to look like a computer and the other a modem(FIRST COMMANDMENT). As a result, sometimes the computer has to be configured to look like a modem and the modem may have to be configured to act like a terminal. (Printers, by the way are usually configured as modems).

This is precisely what I discovered when I was given an excellent Rixon 300/1200 baud smart modem to use, but without a cable. Great, I thought, I just take an RS232 cable and plug it into the serial port on the RS232 card. When this, predictably, failed,I turned to the modem's manual and the TI RS232 card manual for guidance and was immediately plunged into the thick of DCE's, DTE's, pin configurations, connectors, voltage levels, and the like. Once I began to understand a little of what this meant, I was startled to find that the modem was configured as a DTE, that is ,as a terminal. Recovering from this, I began examining which line and which pin did what at the RS232 port and which did what at the modem port. In time, I realized that two of these lines were reversed and would have to be switched somehow before the modem would work.

What I had learned, haphazardly, is the conventional wisdom of the RS232 business: most of the time, there are only 8 lines that really matter. This is in spite of the fact that RS232 connectors contain 25 lines, most of them unused, and that the 8 lines may not be in the same location on any given device. On the TI RS232 port, the Big Eight are as follows(see p. 24 of the RS232 Card manual).

PIN #	MNEMONIC	FUNCTION
1		protective ground
2	RD	data into RS232
3	TX	data out of RS232
5	CTS	clear to send
6	DSR	data set ready
7		logic ground
8	DCD	data carrier detect
20	DTR	data terminl ready

Once you recognize the Big Eight, it is not difficult to figure out which lines need to be switched. From browsing on the TI Form on Compuserve, I have learned that most often it is lines (or pins) 2 3, which send and receive data which are misplaced. Fortunately, switching them turned out not to be difficult, even for a 'soldering iron klutz like me. I simply carved up a TI RS232 Y-cable given me by Art Byers (sorry,Art!) and wired a new 25- pin male plug from Radio Shack on one end. The modem then worked like a dream on the first try.

Some of you will say that there has to be more to it, and there is. If you are curious to find out more, as I am, I suggest a recent book "The RS232 Solution" by Joe Campbell (source of the Big Eight) which reviewers have found both exhaustive and clearly explained. Still, even without in depth knowledge, the RS232 interface can be made to work for you and is, in fact, one of the simplest and most satisfying ways of exploring the hardware- tinkering side of TI 994/A ownership

QUESTIONS? - Courtesy of Mel Gary of NEWJUG

A frequent use of home computers for educational purposes is to have the computer present questions and answers in a random order from a list of questions in an array. To avoid question repetitions, another array can be set up to indicate which questions have not been asked. This method makes question presentation quite slow when most of the questions have been asked. A faster method is presented below. Instead of a flag array, a question order array is set up and its contents randomly ordered. The questions are then read into the questions array in accordance with the order array. This routine assumes that the questions are arranged in groups of increasing difficulty. The groups are presented in order with the questions in random order within the group. I assume ten questions per group.

```
100 DIM QA(10,2), ORD(10)
110 REM SET UP ORDER ARRAY
120 FOR I=1 TO 10
130 ORD(I)=I
140 NEXT I
150 FOR GROUPS =1 TO N
160 REM RANDOMLY MIX THE ORDER ARRAY
170 RANDOMIZE
180 FOR I=10 TO 2 STEP -1
190 RAND=INT(RND*I)+1
200 TEMP=ORD(RAND)
210 ORD(I)=ORD(RAND)
220 ORD(RAND)=TEMP
230 NEXT I
240 REM LOAD QUESTIONS AND ANSWER ARRAY WITH FIRST
    GROUP PER ORDER ARRAY
250 FOR I=1 TO 10
260 READ QA(ORD(I),1),QA(ORD(I),2)
270 NEXT I
280 REM DISPLAY QUESTIONS AND ANSWERS
290 FOR I=1 TO 10
300 PRINT QA(I,1)
:::    GET ANSWER
:::    COMPARE WITH QA(I,2)
:::    OTHER APPROPRIATE CODE
390 NEXT I
400 NEXT GROUP
```

If your questions are stored on tape or disk, the number of questions is virtually unlimited. Add lines to OPEN the approptiate storage device and modify line 260 as necessary. If only one group is desired, change 10 to the number of questions desired in lines 100, 120, 180, 250, and 290 and delete 150 and 400. You've got the framework; Go to it!

_________________By Mark Hodges

Attention all Newsletter Editors
[illegible]

In January of 1986, the Central Westchester 99'ers put together a "flippy" disk of 718 sectors of the very best articles gleaned from 1985 99'ers newsletters from coast to coast. We send it out FREE if you provide the postpaid mailer and a new blank disk, or for $3.00 if you want us to provide the mailer and disk.

However, the purpose of this notice is to let you know that for 1986 we will put out about 1400, sectors available on either flippy or as DSDD.

We ask that you send us the BEST from your club, to be included.

[illegible] The article must be an original by a member of your club that appeared or will appear in 1986 in your club's newsletter. It can cover any computer related topic, such as copyrights, programming in any language useable on the 99/4A, product reviews, etc. and more etc. If a program is part of the article, a separate "runnable" copy must be submitted. All material should be on disk with articles in DIS/VAR 80 format. A return postpaid mailer must be included.

In the past, we made the selection ourselves. This year we intend to impanel a group of editors of 99'er newsletters as judges. If you are interested in acting as a judge, please let us know. Selection of Judges will be closed Sep 1st. Deadline for receipt of material on disk is December 10th 1986.

Address Mail to: The CW 99'ers c/o Art Byers, 1261 Williams Drive, Shrub Oak NY 10588. CIS 073547,2044

All clubs submitting material will receive the Best of the 99'er Newsletters '86, free. You can distribute it to your membership, publish them in your N/L, etc. as you wish. Let us hear from you!!

DATABIOTICS NAP Report #5

Work has begun on programs for the new big, 128k to 512k, memories available for the 99/4A. Bill Moseid of DataBioTics posted an announcement on Compuserve of the following currently under R & D especially for the new larger memories:

(1) A new wordprocessor (2) A new Bulletin Board System (3) Supertalk, a new terminal program (4) SuperForth (5) SpreadHog - Larger than a spread sheet (6) Noteworthy - a music generator.

As a member club of DataBiotics National Advisory Panel it is possible we may be able to beta test some of the above.

DataBioTics listens. When we gave a good review to Diskmaster I, the only beef we had was that single drive owners had to swap disks too many times to sector copy a full disk. DataBioTics read our report and programmer Todd Kaplan reports he has finished a new version that can use the extra memory of SuperSpace or Mini Memory to drastically reduce the number of swaps. [illegible] you can see our participation [illegible] NAP has had specific results.

NEW PRODUCT NEWS: The TMS-9999/BS Chip * * *

Soon to be available is a new microprocessor chip that can be incorporated into the TI-99/4A computer. Our research staff has been able to uncover a list of new opcodes that distinguish the 9999/BS as a major breakthrough in computer technology. The list is presented here for your information.

ABBA - PLAY SWEDISH ROCK
JTZ - JUMP TO ZAXXON PROGRAM
ADGB - ADD GARBAGE
KAL - FLY OVER RUSSIA
DAD - DARK AT DDB
MDB - MULTIPLY AND DROP BITS
BBL - BRANCH ON BURNED OUT LIGHT
MWK - MULTIPLY WORK
BAH - BRANCH AND HANG
NOPE - REFUSE TO DO ANYTHING
BFEI - BEG FOR EXPANSION INTERFACE
OCD - OPEN COMMODE DOOR
BLI - BRANCH AND LOOP INFINITE
PAS - PRINT AND SMEAR
BPB - BRANCH ON PROGRAM BUG
PIP - PULVERIZE INTERFACE PERIPHERAL
BPO - BRANCH IF POWER OFF
PSD - PERFORM SAFETY DANCE
BPM - BEGIN PIRATE MODE
RBT - READ AND BREAK TAPE
BRN - BURN UP HUG CHIP
REST - REST FOR 12 CYCLES
CFP - CALL FOR PROGRAMMER
RPM - READ PROGRAMMER'S MIND
CLD - TRY TO COOL DOWN HUG CHIP
RRT - RECORD AND RIP TAPE
CMS - CATCH MOUSE
RTR - REFUSE TO RUN

CRN - CONVERT TO ROMAN NUMERALS
RWD - REWIND DISK
CSD - CREATE STATIC DISCHARGE
SINK - SINK INTO I.C. SOCKET
DAO - DIVIDE AND OVERFLOW
SRZ - SUBTRACT AND RESET TO ZERO
DEVO - START NEW WAVE (SINE)
SSD - SEEK AND SCRATCH DISK
EIP - ERASE IF PIRATED
STI - SELL TI STOCK
ERS - ERASE READ-ONLY STORAGE
TLK - START SPEECH SYNTHESIS
ETOY - EMULATE COMMODORE-64
TPR - TEAR PAPER
HFA - HIRE FROM ATARI
TRS - TRASH PROGRAM
HCF - HALT AND CATCH FIRE
WED - WRITE AND ERASE DATA
HFC - HIDE FROM CHILDREN
WID - WRITE INVALID DATA
HFP - HIDE FROM PINTO
WOJ - WEAR OUT JOYSTICK
HIC - HELP INTEL CHIP
XBRA - GOTO ZOO
IAD - ILLOGICAL AND
XIO - EXECUTE INVALID OP CODE
IBM - INTERRUPT BAD MNEMONICS
XOR - EXECUTE OPERATOR

IOR - ILLOGICAL OR
XPR - EXECUTE PROGRAMMER
JOF - JUMP ON FLOOR
JOM - JUMP ON MOTOROLA
JOT - JUMP OFF TABLE
CNP - CALL NATIONAL PARTS
RTM - RETURN TO MOTOROLA
CPB - CREATE PROGRAM BUG
RTT - RETURN TO TI
CPM - CORRECT PROGRAM MANUAL
RSD - READ AND SCRAMBLE DATA

*** Above by Karl Schuneman, Port Huron, Michigan, with tongue planted firmly in cheek!

CW-99'ers Newsletter Exchange
c/o Arthur J. Byers
1261 Williams Drive
Shrub Oak, NY 10588

HESTER, NY
MAY
29
1986
PM
John
USA 22

DALLAS TI HOME COMPUTER GROUP
1221 MOSSWOOD
IRVING TX 75061