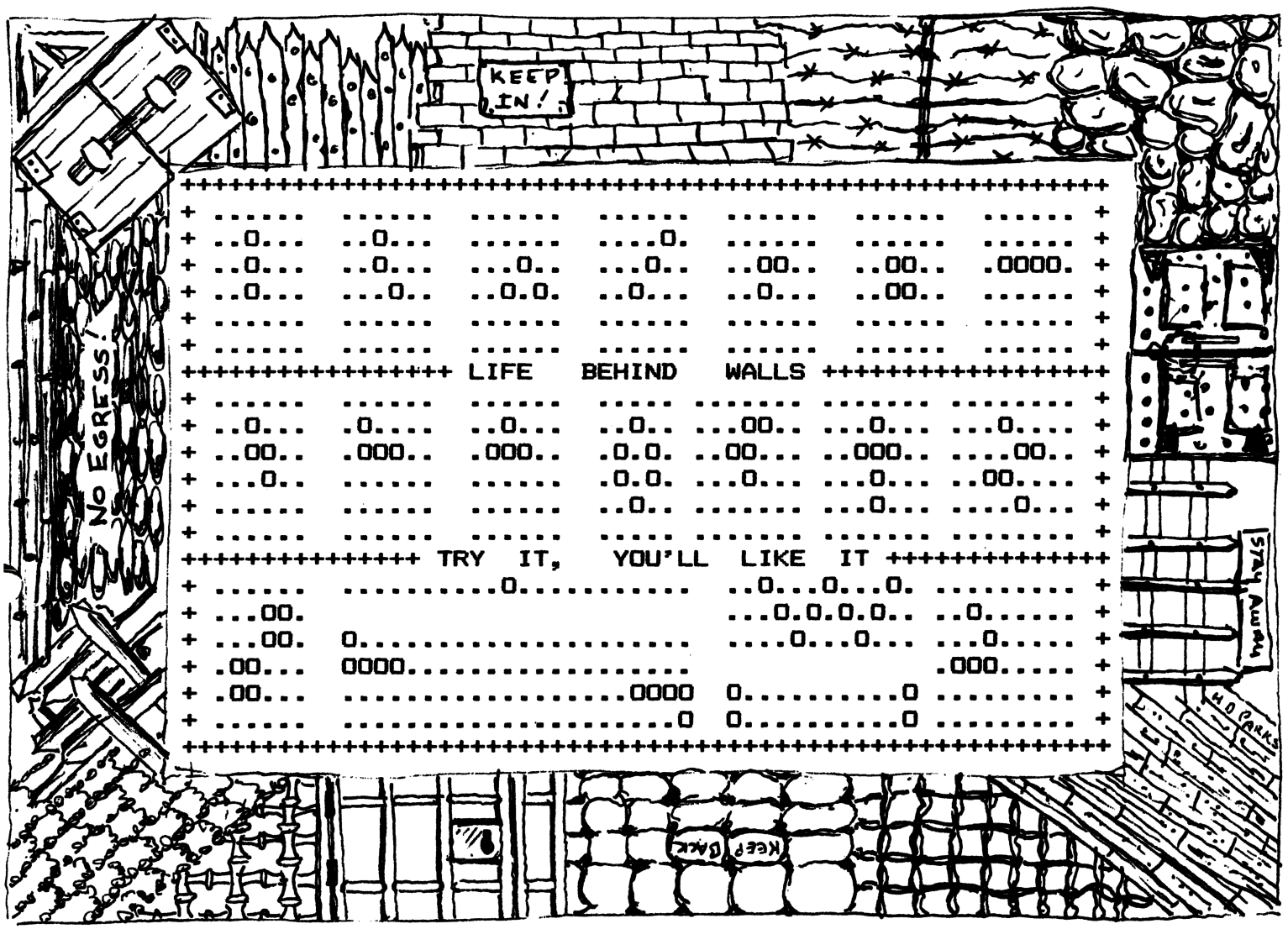
# Next Meeting February 5, 1985
# 7:30pm Details on page 20




See Life Behind Walls Article, pg. 14

# VOICE OF THE '80

FEBRUARY 1985
VOLUME 5 NUMBER 9

# ENTOMOLOGY

or

The Study of "Bugs"

by Alan Abrahamson

## Creepus Logicus

This BUG form is when the programmatical flow is not as devised.
**Suggestion: Use TRON (Trace) to check logic program flow.**

## Variableus Logicus

This BUG rises when the variable in question does not have the proper value.
**Suggestion: Use breakpoints (STOP) in program to check value of variables at crucial times.**

## Syntaxus Argumentus

This is often the most difficult BUG to hunt, since what appears to be the Syntax of the line, may not be.
**Suggestion: Use EDIT mode to step through line for hidden codes.**

## Complexus Statementus

This BUG is shrouded by the complexity of the code.
**Suggestion: Break the program line down to its elements of logic.**

## Fileus Corruptus

This BUG creeps into your files, even though "you" didn't put it there.
**Suggestion: "Look" at the file with a TEXT EDITOR (Sequential) or DOS LIST or ZAP program (Random) to see what is really on file.**

## Basicus Non-Interpretus

This BUG appears when Basic does not seem to be interpreting your instructions as intended.
**Suggestion: Try loading another copy of Basic and check results.**

## Systemus Errorus

This BUG is when your system has become corrupted.
**Suggestion: Clear memory or try a backup system.**

## Hardwareus Problemus

This BUG can be hard to isolate.
**Try to check the simple things first. Check all connections (cables, peripherals, power, etc.) first. Then try to substitute components, if possible. If all fails, call in a "*raven*".**

## Mysterious Messageus

This is NOT a real BUG, it only seems that way.
**Suggestion: Read and understand all the types of errors detailed on the next page.**

My appologies to all the Latin scholars out there.

## Level II Basic Error Codes & Messages

| | | | | | |
|---|---|---|---|---|---|
| 1 | NEXT without FOR | 2 | Syntax error | 3 | RETURN without GOSUB |
| 4 | Out of DATA | 5 | Illegal function call | 6 | Overflow |
| 7 | Out of memory | 8 | Undefined line | 9 | Subscript out of range |
| 10 | Redimensioned array | 11 | Division by zero | 12 | Illegal direct |
| 13 | Type mismatch | 14 | Out of string space | 15 | String too long |
| 16 | String formula too complex | 17 | Can't continue | 18 | No RESUME |
| 19 | RESUME without error | 20 | Undefined error | 21 | Missing operand |
| 22 | Bad file data | 23 | Disk BASIC feature | | |

## DISK BASIC Error Codes & Messages (NEWDOS-80 Ver. 2)

| | | | |
|---|---|---|---|
| 51 | Field Overflow | 52 | Internal Error |
| 53 | Bad File # | 54 | File Not Found |
| 55 | Bad File Mode | 56 | File Already Open |
| 58 | DOS Error | 59 | File Already Exists |
| 62 | Disk Full | 63 | Input Past End |
| 64 | Bad Record # | 65 | Bad File Name |
| 66 | Mode Mismatch | 67 | Direct Statement In File |
| 68 | Too Many Files | 69 | Disk Write Protected |
| 70 | File Access Denied | 71 | Sequential # Overflow |
| 72 | Record Overflow | 73 | Illegal To Extend File |
| 75 | Previously Displayed Error | 76 | Can't Process Line # |
| 77 | Bad File Type | 78 | IGEL Syntax Error |
| 79 | IGEL Item Syntax Error | 80 | Bad/Illegal/Missing IGEL Item Prefix |
| 82 | Bad Record Length | 83 | Statement Uses 2 File Names |
| 84 | Bad File Positioning Parameter | | |

## DOS Error Codes & Messages (NEWDOS-80 Version 2)

| | | | |
|---|---|---|---|
| 0 | No Error | 1 | Bad File Data |
| 2 | Seek Error During Read | 3 | Lost Data During Read |
| 4 | Parity Error During Read | 5 | Data Record Not Found During Read |
| 6 | Tried To Read Locked/Deleted Rec. | 7 | Tried To Read System Record |
| 8 | Device Not Available | 9 | Undefined Error Code |
| 10 | Seek Error During Write | 11 | Lost Data During Write |
| 12 | Parity Error During Write | 13 | Data Record Not Found During Write |
| 14 | Write Fault On Disk Drive | 15 | Write Protected Diskette |
| 16 | Device Not Available | 17 | Directory Read Error |
| 18 | Directory Write Error | 19 | Illegal File Name |
| 20 | Track # Too High | 21 | Illegal Function Under DOS-CALL |
| 22 | Undefined Error Code | 23 | Undefined Error Code |
| 24 | File Not In Directory | 25 | File Access Denied |
| 26 | Directory Space Full | 27 | Diskette Space Full |
| 28 | End Of File Encountered | 29 | Past End Of File |
| 30 | Directory Full Can't Extend File | 31 | Program Not Found |
| 32 | Illegal Or Missing Drive # | 33 | No Device Space Available |
| 34 | Load File Format Error | 35 | Memory Fault |
| 36 | Tried To Load Read Only Memory | 37 | Illegal Access Tried To Protected File |
| 38 | File Not Open | 39 | Illegal Initialization Data On System Diskette |
| 40 | Illegal Diskette Track Count | 41 | Illegal Logical File Number |
| 42 | Illegal Dos Function | 43 | Illegal Function Under Chaining |
| 44 | Bad Directory Data | 45 | Bad FCB Data |
| 46 | System Program Not Found | 47 | Bad Parameter(s) |
| 48 | Bad Filespec | 49 | Wrong Diskette Record Type |
| 50 | Boot Read Error | 51 | DOS Fatal Error |
| 52 | Illegal Keyword/Separator/Termin | 53 | File Already Exists |
| 54 | Command Too Long | 55 | Diskette Access Denied |
| 56 | Illegal MINI-DOS Function | 57 | Operator/Program/Parameter Require Function Termination |
| 58 | Data Compare Mismatch | 59 | Insufficient Memory |
| 60 | Incompatible Drives Or Diskettes | 61 | ASE=N Attribute. Can't Extend File |
| 62 | Can't Extend File Via Read | | |

# "DRIVING" YOURSELF CRAZY

By: Lucien R. Greif
Greif-Associates, Inc.

On Wall Street, there's a fool-proof principle known as "Greif's law." When Greif buys ... sell; when Greif sells ... buy. It never fails.

It's no surprise, then, that when I bring home a real bargain from a recent New Jersey computer show - and plug it in - I find it doesn't work! After all, when you pay $170 for two new disk drives, with power supply, in a very handsome cabinet, you have no right to expect trouble-free operation first time out. For anyone ever faced with adding drives 2/3 to a model III TRS-80 - and they're not the ready-made (i.e. expensive) versions offered up by Radio Shack - my trial by panic may prove helpful. So here goes my chronology:

The drives I bought are Shugart model L-400 which, I am assured by all who know, are truly 40-track drives. That's the first thing you have to watch out for. Make sure you don't pick up 35-trackers. Cheaper, yes; but you may have considerable aggravation if you try to mix-n-match them with 40-trackers. So far, so good. I had the right type.

Cable. Purchased ready-made, with three connectors (one for the CPU, one for each drive), you'll be out some $30+. Make it yourself, and you can do it for less then ten. But don't stint on the 34-conductor cable. There's no such things as a cable that's too LONG - only too short.

Cable made up, expansion disk drives connected, CPU turned on - and I'm all set for the thrill that comes once in a lifetime: my very first, my very own drives 2 and 3 in operation. Bah, humbug. "ILLEGAL OR MISSING DRIVE" is all my NEWDOS80v2 can think of saying. I knew it - the damn drives are no good. There goes $170. And the outfit is in Framingham, MA (Horizon Sales). Actually, a decent bunch (more on this later).

Remembering what a helpful computer-freak at the show had told me about installing auxiliary drives, I opened the case (with much trepidation, to be sure). Inside, there are two DIPs (14-legged "bugs") that you have to make friends with. One of them is the selector shunt, the other is the terminator. If each drive has a terminator, take one out. Only the last drive in the series (drive 3) needs to be terminated. (Drives 1/2 inside the model III are NOT terminated, so there is nothing you have to remove or change.) How can you tell which one of these little bugs is the terminator? Easy. It will be very close to the 16-conductor edge card connection on the drive, to which you attach your cable. (There's a slot in this connector, so theoretically you can't connect the cable wrong-way-to. The slot is between conductors 4 and 6.) There will be TWO removable DIPs in that area. Take them both out. One consists

of seven 150-ohm resistors strapped across corresponding pins (i.e. 1 to 14, 2 to 13, 3 to 12, etc). THAT's the terminator. Reinstall it only for drive 3.

The other removable DIP connects FOUR drives (all shorted together!); leads to some sort of multiplexer; and provides power to the drive motor. All connections but two - motor power and ONE drive - have to be cut or opened. Radio Shack's technical manual gives you this much of a hint, based (I believe) on their own drives, probably Tandon models:

For drives 0 and 2 ... remove pins 12, 14 and 32.

For drives 1 and 3 ... remove pins 10, 14 and 32.

(You may also have to open/remove the multiplex connection, similar to the work I did for the Shugart drives, as follows:)

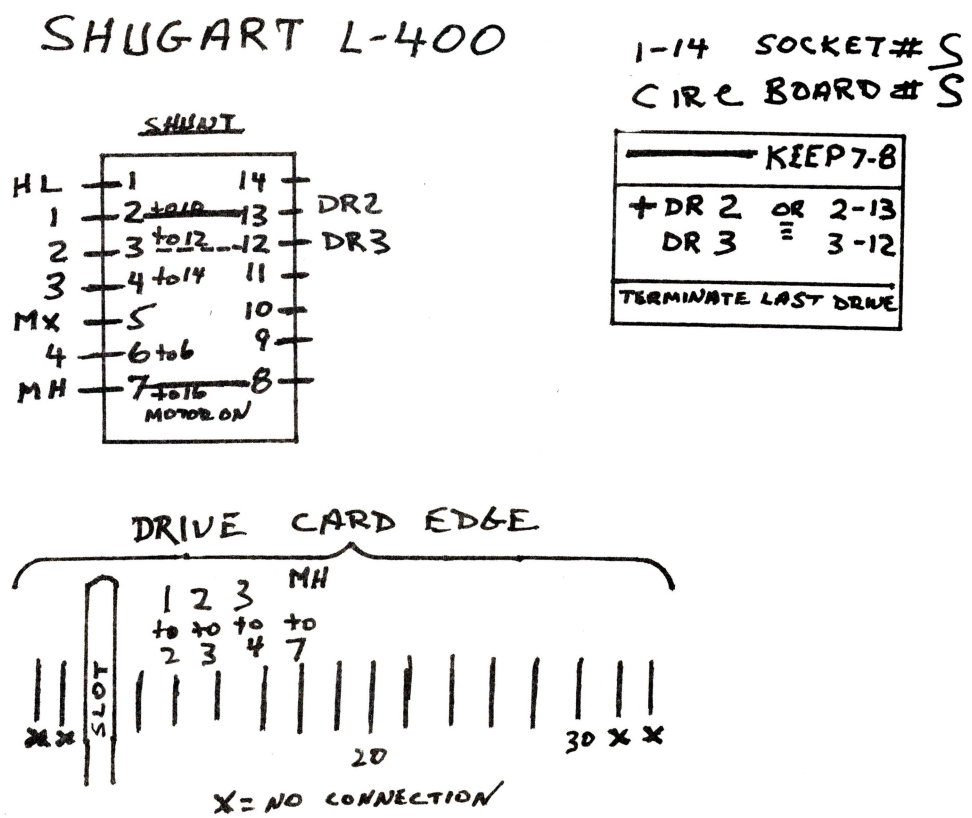
The L-400 SHugart drive has - right next to the terminator socket, another socket which holds the DIP consisting of seven shunts. Cut them (or bend the pins, in case you change your mind later), KEEPING ONLY two crossstraps:

For drive 2 ... KEEP 2-13, and 7-8. (DIP numbers)

For drive 3 ... KEEP 3-12, and 7-8. (DIP numbers)

Open the connections labeled (ON THE CIRCUIT BOARD) HL, 3, MX, and 4. See FIGURE 1.

Figure 1

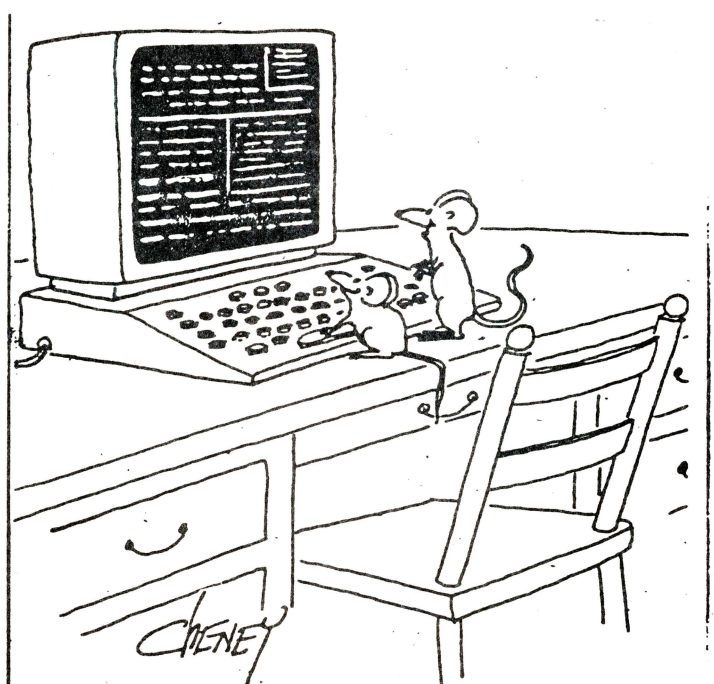


Special thanks must go to Alan Abrahamson who provided gracious encouragement and helpful hints on how to solve my perplexing problem. The biggest "scientific breakthrough" came from his suggestion that I must first reset my PDRIVE conditions if I'm to eliminate that nasty comment, "ILLEGAL DRIVE." In the NEWDOS READY mode, (1) PDRIVE,0 <ENTER>. Note how many of your drive specifications have an * next to them. (Mine had only two drives signed up.) To get the machine ready to work with four drives: (2) SYSTEM,0,AL=4 <ENTER>. (3) RESET. (4) PDRIVE,0 <ENTER> and you'll note you now have drives 0 to 3 asterisked - ready to go to work.

Success did not come easy - but it was, oh, so sweet. Several lessons learned: (1) I am grateful to Alan and my other friends at FCUG who are willing to share their hard-won knowledge with a total stranger, (2) Horizon Sales is a decent outfit (unasked, they refunded a $60 overpayment before I even noticed it); (3) TRS-80 manuals are helpful IF you're willing to improvise and experiment a little when you run into slight differences among various pieces of equipment; and (4) what a beautiful day it can be when the drives finally fire up. I thank you all.

PS: Don't count on Shugart to give you any help! I wrote to them weeks ago, when I first discovered the drives weren't functioning properly. Their assistance fits into this space( ).

# # # # #




"All right! We're into the shipping records of the Castington Cheese Company. How much and where do you want it delivered?"

## New Items from Tandy for 1985.
By George Saladino

Last month I mentioned some areas that you would see computers in 1985 more than in the past. Well here are some new items from Tandy for 1985. There is a new computer catalog from Tandy for 1985 that may have come out by the time you read this article. The number on the catalog is # RSC-14 (This is the second catalog that will come out with the 1985 date on it.) Here are some of the new items that are mentioned in it. Note: I mention some prices for the new items in this article. I just heard that Radio Shack has just told its stores that some of the prices in the new catalog are wrong! So the prices I quote you may not be right!

The Tandy 6000 (Note that the Tandy name is being used instead of the Radio Shack name now!) replaces the Model 16. It is a Multi-user computer (Used with Xenix) that comes with 512K memory and two 8 inch disk drives. It uses a Z80A and a MC68000 for the processors. This sells for $4499.00. Also, the 6000 comes with one disk drive and a 15 megabyte hard disk drive. You can add an external hard disk drive to a have a total of 50 megabytes of hard disk storage. The unit with the hard disk sells for $5499.00. The 6000 is expandable to one megabyte of memory.

The next new item of note is the Tandy 200. Yes the new 100! However, the 100 will also be sold. The 200 has a flip up 16 line by 40 Character liquid crystal display. I think that they should have made it 80 characters! It has the five built in programs of the 100 and also Multiplan as the 6th program. The memory of this new unit is 24K and is expandable to 72K. The unit looks much like the 100 but with a flip up screen. The keyboard is the same as the 100 except the arrow keys are placed in a group like the NEC unit. The 200 also has a 300 baud built in modem and can also use the Disk Video Interface that the Model 100 uses. I hope the cable connection is improved.

Specifications on the 200 list also that there is an improved version of the "TEXT" program that is in the 100. The "TELCOM" program with the modem produces a "TONE" dialing, so you can use it with long distance services. The price on the 200 is $999.00 for the 24K unit. Memory up grades of 24K at $249.95 each.

New Accessories for the 100 or the 200 include prop legs for the 100 that you can slip into the two rear holes of the 100 to tilt the 100 up for a better angle while using it on a desk. ($3.99) Another new item is soft carrying case for the 100 or 200 that sells for $39.95.

One other item is a monochrome monitor that you can use with the Disk Video Interface that sells for $159.95

I do not know when the Model 6000 is to come out, but the Model 200 is due out 2/28/85.

The Model 200 looks that it is a Model 100 with the expanded memory, Multiplan and also the flip up screen of 40 characters by 16 lines.

Well, I have a 100 with 64K. Remember I mentioned that I bought the PG 32K Ram and installed it in my 100. The 200 will be banked switched the same as my 100 with the PG Ram in it. I do not know if the 200 will have software that will let you interact between banks as the software that PG Design is bringing out for their memory addition.

I can get Multiplan or other plug in Roms such as Lucid (A spread sheet) to plug into my Model 100.

In the December 1984 issue of PCM, there is mention of a 80 Character by 25 line screen that will hook up to the 100!! It is one inch thick and is three pounds. It is available from Axonix Corporation of 417 Wakara Way, Salt Lake City, UT 84108. Phone is 800-821-7093. No price given.

Here is a short program I modified that came from the Model 100 board on CompuServe. I put in some additions. The program sets the status of the "TELCOM" program. See lines 50 & 60. Then turns on the sound so you can hear the status of your connection with the modem. The last line is a call that calls up Telcom. Line 10 pokes in a clear function when you push "F6" in term.

```
 5 CLS
10 POKE 64268,49 : POKE 64269,66
20 FOR A = 63067 TO 63071
30 READ B : POKE A,B
40 NEXT
50 DATA 77,55,73,49,69
60 REM   M  7  I  1  E
65 SOUND ON
70 CALL 20806
```

Other calls, peeks and pokes for the Model 100 that may be of interest are listed below:

```
If PEEK(65442)=32 the caps lock is on.
If PEEK(65442)=0 the caps lock is off.
Break key lockout POKE 63056,128
To unlock the break key POKE 63056,0

Entry calls are -
Basic  Call 27721
Text   Call 24046
Telcom Call 20806 (used in my program above)
Term   Call 21589 (goes into the terminal program directly.)
Addrss Call 23400
Schedl Call 23407
```

One other new item. For you computer programmers, the EC-4075 Programmer's calculator, is a very useful calculator. It converts Hex to Decimal (and the other way), picks out random numbers, has a clock to keep the time and date, a stop watch function (Counts up or down) with a lap function, and it even calculates. It is very handy when programming in machine language or riding on the subway and timing how long one is delayed due to some problem! The calculator is priced around $24.00. Cat #23-468. Just the calculator for the computer programmer.

A last note. Alan put in the newsletter last month an article on how to submit articles. It is really easy. I send or bring him a disk with my article on it every month. Just format a disk in any format other than two sided and put your article in ASCII. He will read the disk file. Let others share your knowledge about computers. There must be someone else other than me that can contribute articles to this fine newsletter.

# TRSDOS 6.2

by John Wasmuth

Reprinted from Bits and Pieces
December 1984, Volume 4, Number 12

I have been playing with my TRSDOS 6.2 for about a month now and I feel it is definitely worth the $19.95 price being offered to registered Model IV owners. Here are some of the features included in TRSDOS 6.2:

**Disk I/O** is much faster in this version than in 6.1.2, even though I was running my version of 6.1.2 with 6ms step rate.

There is an extensive **HELP** command available, but it requires over 20K of disk space for its help file.

**CTRL-R** can be used to repeat the last command entered. This can be handy when formatting several disks or when you need to change disks and reenter a command.

**DIR** now defaults to ALL, giving you a full description of each file entry.

**CAT** replaces the short listing of filenames provided by DIR in version 6.0.

**CLS** has been added allowing you to clear the screen in a JCL file.

**TOF** has been added to eject the printer to the top of the next page.

**LOG** has been added to allow switching to a different SYSTEM disk without resetting the system. This seems to primarily be intended for those who have a drive other than single sided, single density and want to change the type of disk being used as a system disk.

**DIR** now accepts a range of drives. For example, you can specify DIR 2:-3.

**FORMAT** now allows you to specify the track which the directory will reside on. Also, you may now recreate a fresh directory on a hard Disk using the FORMAT command.

**SETCOM** has some minor enhancements in how parity can be set.

**SYSTEM** has the following added capabilities:

You may disable the DATE prompt when booting.

You may specify that interrupts will be enabled or disabled when doing disk I/O. (This allows you to make a trade-off between faster disk I/O and having the type-ahead feature work correctly when a disk is being accessed.)

**CLICK/FLT** now produces a much more audible sound. Also, when setting up CLICK/FLT, you can specify any one character that will produce a click. For example, you could produce a click only when carriage return is pressed.

Also, included in the technical portion of the manual are instructions for converting a program to make it part of the system. The only advantages to doing this are that the program will load faster and you can access it by typing "*" and enter instead of having to type in the full program name.

Several new features have been added for the Assembly Language programmer including the ability to set up user-defined SVC's and the ability to code a replacement command interpreter for TRSDOS.

## Definition of the Month

Address - Where a bit has its mail sent.

# A s k A l a n

by Alan Abrahamson

**Q:** *How do you use a "DO" file and how can it benefit me in my programming?*

**A:** The concept of "DO" or "JCL" files is one of the most powerful areas available to all programmers. This facility allows you to in many ways automate your system or application. To better understand, let us take a look at some of the "JCL" (Job Control Log) constructs available on the more popular operating systems.

TRSDOS 1.3, MULTIDOS, and DOSPLUS 3.4 support the simple "DO" file concept. These files can either be built with the "BUILD" DOS command or in any word processor capable of ASCII output. There is NO logic involved. These files are merely a list of instructions for your computer to execute in sequence and should have the extension "/BLD" ("/IDO" for Multidos).

NEWDOS-80 Version 2 supports a bit more complex "JCL" file concept. Several logic controls are possible with this system. The file can also be segment controlled and keyboard commands can be sent to the file. This type of file is slightly harder to construct, due to the special control codes used. The file can be constructed from BASIC or with a "JCL" file building utility. These files should have the extension "/JCL".

LDOS, TRSDOS 6.x and DOSPLUS 3.5 have tremendous "JCL" capabilities. These files are easy to construct using the "BUILD" DOS command or a word processor, but the real power lies in the logic that this list file can contain. This is really like another computer language for your micro. Such sophisticated constructs as IF-THEN, JUMP, READ and LOAD allow processing of data in a completely automated sequence.

Why would I want to use this system? If you had your program running in an atmosphere where the operators of the applications were not expert in computers, you would want your procedures to work flawlessly even though they may be complex. "JCL" files to the rescue. For example, you could automate the system back-up procedure so that all that would be necessary to invoke a system back-up would be to type "DO BACKUP" or something similar. This would invoke the file "BACKUP/JCL" and all necessary operator procedures would be prompted and followed in exact sequence according to the contents of the file "BACKUP/JCL". That's the way to reduce system errors!

# HI RES

by Steve Cox
Reprinted from CINTUG
Vol 1, No 1, January 1985

In my quest for an article I came upon a program I wrote in BASIC which will take an ASCII Picture Text File and print it on an EPSON with Graphtrax-80 in a form of Hi-Res. I say a form because there is not enough detail in these TEXT files to produce a true Hi-Res picture. (Ed Note: See Page 13 for some samples of output from this program.)

Since each EPSON character is made 5 dots wide and 7 dots high and it is the shape of the characters which determine the picture shading this must be taken into account. If each character is converted into one dot and each space into a blank the resulting picture becomes out of perspective. That is it gets too short and fat. It also loses detail due to the lack of shading.

My approach was to convert each character into a similar dot pattern. I selected 3 dots wide by 5 dots high. This allowed keeping perspective and a certain amount of shading. The pattern for I, for example is the top and bottom dots in the first column, all 5 dots in the second and the top and bottom dots in the third. A comma is the bottom two dots in the middle column and a period is the bottom dot in the middle column ... etc. The hard letters are the slanted ones since we only have 3 rows in which to produce the slant. A "V" for example, is the top two dots in the first and third columns and the bottom three dots in the middle column.

My intention was to do the same for all keyboard characters however, since the picture I was working with used only a few characters, I only converted them and I therefore never finished a complete HI-RES character set.

Line 230 converts the length of the input string into two numbers which are then sent to the printer in line 240. These tell the EPSON that this is the bit graphics mode and how many rows of dots to expect. Line 190 sets the line feed to five dots high per line. Lines 270 to 350 do the character to dot conversion and in line 360 any character not recognized is converted to a black box (all five dots in all three rows).

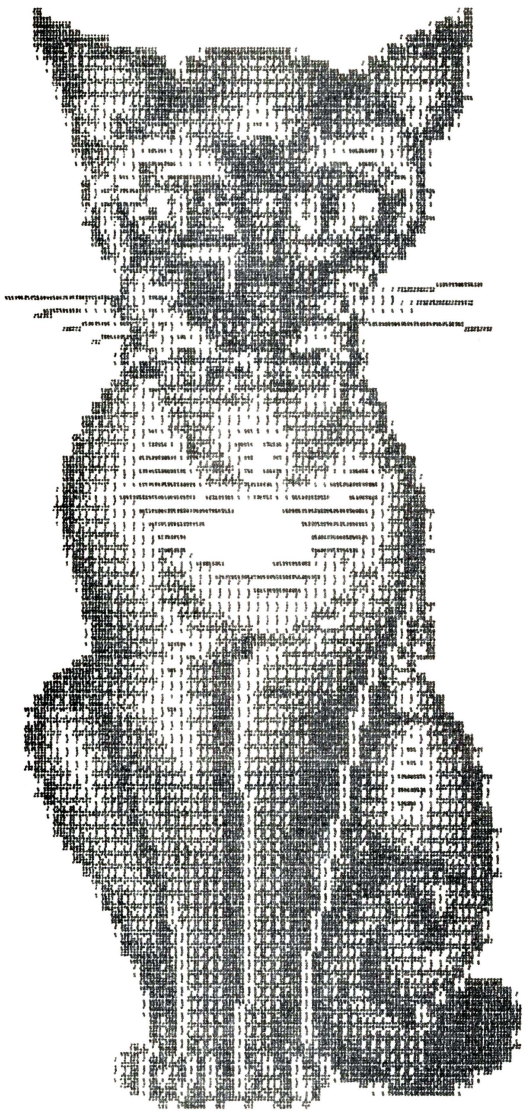
Line 270 is the space character. Notice that on a Model I a control 0 can not be sent to the printer. This is a fault of the Level 2 ROM's. The 0 is POKEd to the printer and the subroutine at 400 waits for the printer ready signal.

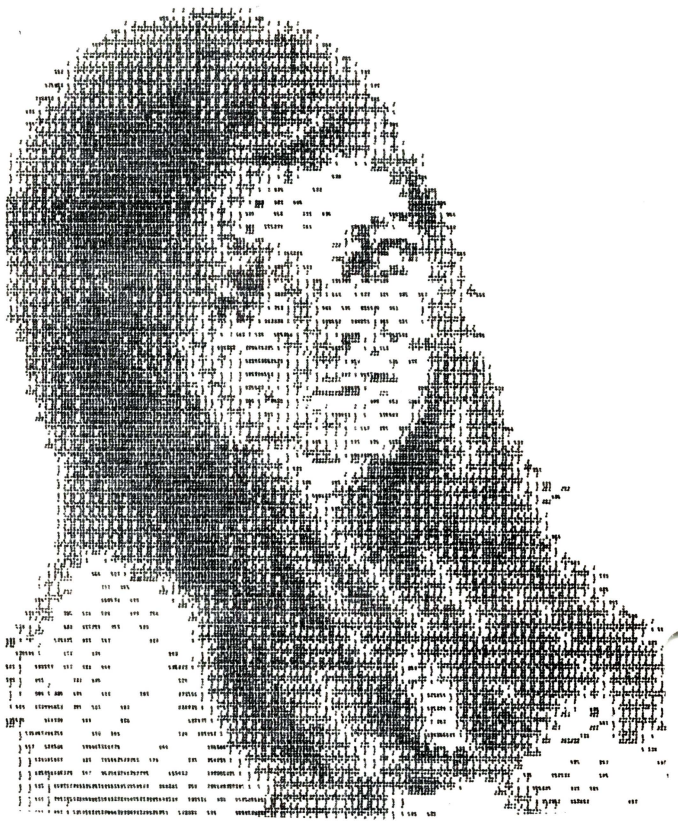
As you can see from Page 13, the results are not bad. I'm sure that the quality could be improved by fine tuning the dot character set. Also this file did contain some characters that i did not convert. The picture I was working on was not suitable for our front cover, if you know what I mean.

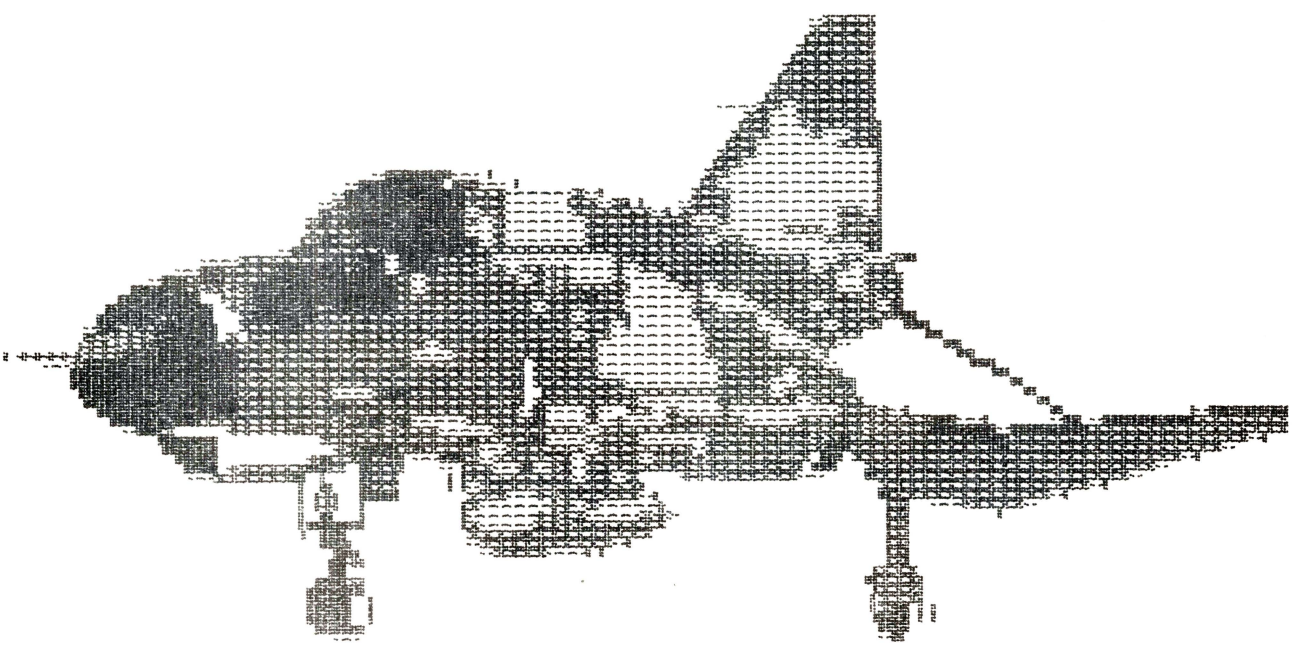
# HIRES/BAS

```
10 CLS : PRINT"********** ASCII Print Converter SIZE 2
**********
20 PRINT"This program will convert an ASCII text file"
30 PRINT"Picture, such as Sharron, to HI-RES graphis"
40 PRINT"on an EPSON printer able to print Hi-Res"
50 PRINT"Graphics.  It should work with Graphtrax-Plus"
60 PRINT"To use it the picture MUST be saved in ASCII on"
70 PRINT"disk.  This program assumes that the standard"
80 PRINT"conventions were used for shading in the ASCII"
90 PRINT"picture."
100 PRINT"********** By Steve Cox ***** 1984 **********"
110 FOR ZX = 1 TO 2000 : NEXT : OUT 254,12
130 CLS
140 CLEAR 10000 : DEFINT Z,X,D
150 LPRINT CHR$(27)"@"
160 ONERRORGOTO 390
170 INPUT"Enter filename";FI$ : IF FI$ = "" THEN 170
180 OPEN"I",1,FI$
190 LPRINT CHR$(27)CHR$(51)CHR$(15);
200 LINEINPUT#1,A$
205 IF LEN(A$) = 0 THEN 200
210 D = D+1 : PRINT@512,"on line #"D;
220 PRINT@640,A$;
230 AC = LEN(A$)*3 : IF AC>256 THEN AL = AC-256 : AK = 1 ELSE AK
= 255 : AL = AC
240 LPRINT CHR$(27)"K"CHR$(AL)CHR$(AK);
250 FOR X = 1 TO LEN(A$) : PRINT@960,X;
260 B$ = MID$(A$,X,1)
270 IF B$ = " " THEN FOR ZX = 1TO3 : GOSUB 400 : POKE14312,0 :
NEXT : NEXT : GOTO 380
280 IF B$ = ","ORB$ = "." THEN FOR ZX = 1TO3 : LPRINT CHR$(3); :
NEXT : NEXT : GOTO 380
290 IF B$ = "I" THEN LPRINT CHR$(6)CHR$(31)CHR$(6); : NEXT :
GOTO 380
300 IF B$ = "A" THEN LPRINT CHR$(7)CHR$(24)CHR$(7); : NEXT :
GOTO 380
310 IF B$ = " : "ORB$ = ";" THEN GOSUB 400 : POKE14312,0 :
LPRINT CHR$(27); : GOSUB 400 : POKE14312,0 : NEXT : GOTO 380
320 IF B$ = "'"ORB$ = CHR$(34) THEN FOR ZX = 1TO3 : LPRINT
CHR$(24); : NEXT : NEXT : GOTO 380
330 IF B$ = "-" THEN FOR ZX = 1TO3 : LPRINT CHR$(6); : NEXT :
NEXT : GOTO 380
340 IF B$ = "H"ORB$ = "8" THEN LPRINT CHR$(31)CHR$(6)CHR$(31); :
NEXT : GOTO 380
350 IF B$ = "V"ORB$ = "O" THEN LPRINT CHR$(24)CHR$(7)CHR$(24); :
NEXT : GOTO 380
360 FOR ZX = 1TO3 : LPRINT CHR$(31); : NEXT
370 NEXT
380 LPRINT " " : GOTO 200
390 CLOSE1 : LPRINT CHR$(27)"H" : CLS : CMD"E" : GOTO 140
400 IF (PEEK(14312) AND 240) <> 48 THEN 400 ELSE RETURN
```

# Sample outputs from HIRES/BAS







# LIFE BEHIND WALLS

By Herman D. Parks

LIFE! Some are born, some will die, and some hang on for a few more seconds.

Few mathematical concepts have intrigued computerists like the game concept called "Life". It was developed by mathematician John Conway in the late 60's and popularized by Martin Gardner in his Mathematical Games column in the Scientific American. It is a product of Conway's studies in group and number theory. Whatever its higher mathematical implications, it has a great appeal to the imagination and since Gardner's article appeared many later articles and computer programs have followed.

For those who have not been introduced to Life, here is a brief description. Imagine a checkerboard grid, usually unlimited in area. The spaces of the board are called cells. Some of the cells are occupied, are "alive". The remainder are empty, are "dead". An intriquing aspect for the player is the selection of an initial pattern of live cells.

Thereafter the game procedes in stages called "generations". In each generation every cell, whether alive or dead now, is subjected to a set of simple rules which determine whether it will be alive or dead in the next generation.

The rules are these: Think of each cell as being in the center of the other eight cells which are adjacent to it. The number of alive adjacent cells determine whether the center cell will be alive in the next generation.

If the center cell is presently alive then it will remain alive if it is adjacent to either 2 or 3 live cells. Otherwise it will die from either lonliness or suffocation.

If the center cell is dead, then it will be alive in the next generation, if it is now adjacent to exactly 3 live cells.

The status of a cell is not changed until all cells have been evaluated. Then all changes are made and a new generation is displayed. For example, consider Fig. 1:

```
.....   .....   .....   .....   .....   .....   .....   .....
.....   ..O..   ..O..   ..O..   .....   ..O..   .....   .....
.OOO.   .O.O.   .O.O.   ..O..   .....   ..O..   .....   .....
..OO.   .O.O.   .....   .....   .....   ..O..   .....   .....
.....   .....   .....   .....   .....   .....   .....   .....
 1a      1b      1c      1d      1e      1f      1g      1h
```

Fig. 1a is the original pattern, 1b etc.are the subsequent generations. By generation 1e alles kaput. The reader can check these out using the rules given above. But other patterns fare much better. See if you can figure out what happens in 1f.

So what does all this have to do with personal computing? Well, Gardner reports that when Conroy and buddies first worked on LIFE, they used a printed grid sheet with markers to show live cells. Each generation was computed by hand. Many people, myself included, upon reading the article, thought "what a great application for a computer".

I had to wait some 12 years before I got my chance. In '79 I wrote a Life program for my first PC, the OSI C1P (Tape, 8K). But, because of the limited display of 25 by 25 characters, I added a feature. I mathematically wrapped the grid space around horizontally and vertically so that each of the four edges was continued on the opposite side of the grid.

To this I added the ability to define grids of any length and width up to the limits of the display and called the resulting system "Life Behind Walls". Whether Conroy would approve or not, I don't know, but I found the resulting "mirror" effects using small or narrow grids very interesting.

Now I have written essentially the same program for the Model III, and again the language is BASIC. When you run LIFE1/BAS, or restart it, you are asked for the number of horizontal and vertical cells. The limits are 62 and 13. The computing of generations is relatively slow so, at least in the beginning, stick to moderate sizes like 6 by 6.

The requested grid is shown outlined by walls of plus signs, and a faint "X"-cursor will appear in the upper left corner of the grid. The cursor can be moved to any position in the grid using the four arrows keys. The effect of the cursor on each cell it occupies (seen when the cursor leaves the cell) is determined by the last pressing of one three keys: <D> for draw, <E> for erase, or <M> for move. If sucessive position of the cursor are to have the same effect, then only the arrows need be used after a single pressing of one of the three letter- keys. With the cursor in a given cell location the last letter-key pressed will determine the setting of that cell - showing an "O" for live cells and a blank for dead ones.

When you have settled on an initial design, press <ENTER> to start the next generation. As each cell is processed a dot will flash in it to show progress, and any change in the cell will show at that time. A counter in the lower right corner of the screen shows how many generations have passed.

To start the next generation, press <C> or hold it down for continuous processing. Press <R> to restart the program including the entry of new grid dimensions.

While the action is much faster (and more reliable) than by hand on a physical grid, it can be made even faster through the use of machine-language coding. To this end I have written an ML module intended to be integrated into the Basic program using DATA statements. I expect a speed increase in the order of 20 to 1 or better. The module is ready but the integration is not. If you would like a copy, send a SASE to the editor.

The cover illustration shows a number of patterns which you can explore. Don't miss the one in the lower right corner. Hint: it is called "the glider".

To those of you who get hooked on Life, I have only one thing to say. "L'Chaim!".

## LIFE/BAS

```
10 '*** LIFE1/BAS *** 1/12/85
20 'BY HERMAN D. PARKS
30 'FOR THE FAIRFIELD COUNTY COMPUTER USERS GROUP, INC.
50 CLS : CLEAR 1000 : DEFINT A-Z : DIMA1(65,17,2)
200 '===== MAINLINE =====
220 '@300 - GET USER PARAMETERS
230 '@400 - GENERATE BLANK DISPLAY FIELD
240 '@500 - GET USER INITIAL CELL PATTERN
250 '@700 - LOAD ARRAY FROM SCREEN
260 '@900 - COMPUTE NEXT PATTERN AND DISPLAY IT
270 '@1100 - QUERY USER TO CONTINUE OR RESTART OR EXIT
299 '
300 '=====  GET USER PARAMETERS =====
301 '
310 INPUT "HOW MANY HORIZONTAL CELLS "; CX
320 IF CX>62 THEN PRINT "MAX = 62, RE-ENTER" : GOTO 310
330 INPUT "HOW MANY VERTICAL CELLS "; CY
340 IF CY>13 THEN PRINT "MAX = 13, RE-ENTER" : GOTO 330
399 '
400 '=====  GENERATE BLANK DISPLAY FIELD =====
401 '
405 CLS
410 FOR A = 0 TO CX + 1 : PRINT@A," + "; : PRINT@(CY + 1)*64 +
A," + "; : NEXT A
420 FOR A = 1 TO CY : PRINT@(A*64)," + "; : PRINT@(A*64 + CX +
1)," + "; : NEXT A
430 CN = 1 : PRINT@1019,CN;
499 '
500 ' =====  GET USER INITIAL CELL PATTERN  =====
501 '
505 PRINT@15*64,"";TAB(55); : PRINT@15*64,"<M>OVE, <D>RAW,
<E>RASE WITH ARROWS - <ENTER> TO START"; : X = 1 : Y = 1 : CC$ =
""
510 PL = 15360 + X + 64*Y
520 SC = PEEK(PL) : 'SAVE THE CHARACTER AT PL
530 POKE PL,88 : POKE PL,SC : 'ALTERNATE SAVED CHARACTER AND "X"
```

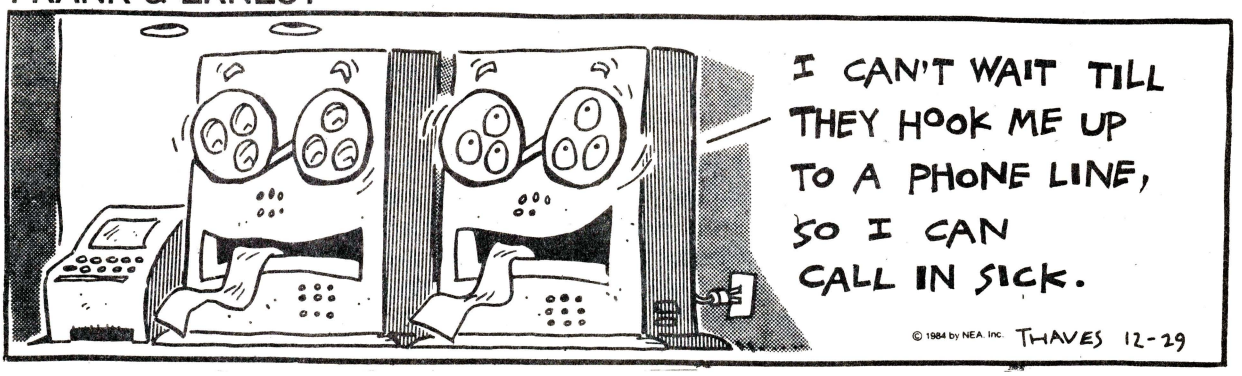
```
535 A$ = INKEY$ : IF A$ = "" THEN GOTO 510
540 IF A$ = "M" THEN CC$ = "" : GOTO 510
550 IF A$ = "D" THEN CC$ = "O" : POKE PL,ASC("O") : GOTO 510
560 IF A$ = "E" THEN CC$ = " " : POKE PL,ASC(" ") : GOTO 510
570 IF A$ = CHR$(8) THEN IF X>1 THEN IF CC$ = "" THEN X = X-1
ELSE X = X-1 : POKE PL,ASC(CC$) : GOTO 510
580 IF A$ = CHR$(9) THEN IF X<CX THEN IF CC$ = "" THEN X = X + 1
ELSE X = X + 1 : POKE PL,ASC(CC$) : GOTO 510
590 IF A$ = CHR$(10)THEN IFY<CY THEN IF CC$ = "" THEN Y = Y + 1
ELSE Y = Y + 1 : POKE PL,ASC(CC$) : GOTO 510
600 IF A$ = CHR$(91) THEN IFY>1 THEN IF CC$ = "" THEN Y = Y-1
ELSE Y = Y-1 : POKE PL,ASC(CC$) : GOTO 510
610 IF A$ = CHR$(13) THEN GOTO 700 ELSE GOTO 510
699 '
700 ' =====  LOAD ARRAY FROM SCREEN  =====
701 '
720 FOR X = 1 TO CX : FOR Y = 1 TO CY
730 IF PEEK (15360 + X + 64*Y) = ASC("O")  THEN A1(X,Y,0) = 1
ELSE A1(X,Y,0) = 0 : 'LOAD FRONT ARRAY FROM DISPLAY PATTERN
740 NEXT Y : NEXT X
745 'NOW COPY THE OUTSIDE ROWS AND COLUMNS TO THEIR OPPOSITE
EDGES
750 FOR Y = 0 TO CY + 1 : A1(0,Y,0) = A1(CX,Y,0) : NEXT Y
760 FOR Y = 0 TO CY + 1 : A1(CX + 1,Y,0) = A1(1,Y,0) : NEXT Y
770 FOR X = 0 TO CX + 1 : A1(X,CY + 1,0) = A1(X,1,0) : NEXT X
780 FOR X = 0 TO CX + 1 : A1(X,0,0) = A1(X,CY,0) : NEXT X
790 FOR Y = 0 TO CY + 1 : FOR X = 0 TO CX + 1 : A1(X,Y,1) =
A1(X,Y,0) : NEXT X : NEXT Y : 'COPY ZERO-PLANE TO ONE-PLANE
899 '
900 ' =====  COMPUTE NEXT PATTERN AND DISPLAY IT  =====
901 '
905 CN = CN + 1 : PRINT@1019,CN;
910 IF Z1 = 0 THEN Z1 = 1 ELSE IF Z1 = 1 THEN Z1 = 2 ELSE IF Z1
= 2 THEN Z1 = 1 : 'CHOOSE SOURCE PLANE OF ARRAY
920 IF Z1 = 1 THEN Z2 = 2 ELSE Z2 = 1 : 'CHOOSE DESTINATION
PLANE
930 FOR Y = 1 TO CY : FOR X = 1 TO CX : PRINT@X + 64*Y,".";
940 ACC = 0 : FOR T = -1 TO 1 : FOR U = -1 TO 1 : 'ACC =
ADJACENT CELL COUNT
950 IF A1(X + T,Y + U,Z1) = 1 THEN ACC = ACC + 1
955 NEXT U : NEXT T
960 IF A1(X,Y,Z1) = 1 THEN ACC = ACC-1 ELSE GOTO 990 : 'CENTER
CELL LIVE?
970 IF ACC<2 OR ACC>3 THEN A1(X,Y,Z2) = 0 : PRINT@X +
64*Y,CHR$(32); : GOTO 1000
980 A1(X,Y,Z2) = 1 : PRINT@X + 64*Y,"O"; : GOTO 1000
990 IF ACC = 3 THEN A1(X,Y,Z2) = 1 ELSE A1(X,Y,Z2) = 0 : 'CENTER
CELL NOT ALIVE SO IS IT BORN OR NOT?
995 IF A1(X,Y,Z2) = 1 THEN PRINT@X + 64*Y,"O"; ELSE PRINT@X +
64*Y," ";
1000 NEXT X : NEXT Y
1005 'NOW COPY THE OUTSIDE ROWS AND COLUMNS TO THEIR OPPOSITE
EDGES
1010 FOR Y = 0 TO CY + 1 : A1(0,Y,Z2) = A1(CX,Y,Z2) : NEXT Y
1020 FOR Y = 0 TO CY + 1 : A1(CX + 1,Y,Z2) = A1(1,Y,Z2) : NEXT Y
1030 FOR X = 0 TO CX + 1 : A1(X,CY + 1,Z2) = A1(X,1,Z2) : NEXT X
1040 FOR X = 0 TO CX + 1 : A1(X,0,Z2) = A1(X,CY,Z2) : NEXT X
```

```
1099 '
1100 ' ====  QUERY USER TO CONTINUE OR RESTART  =====
1101 '
1105 PRINT@15*64,TAB(55);
1110 PRINT@15*64,"HOLD DOWN <C> TO CONTINUE. PRESS <R> TO
RESTART";
1120 A$ = INKEY$ : IF A$ = "" THEN 1120
1140 IF A$ = "R" THEN Z1 = 0 : CLS : GOTO 300
1160 IF A$ = "C" THEN GOTO 900
1170 GOTO 1120
4999 '
5000 ' =====  PRINT OUT ARRAY A1 (A TEST PROCEDURE ONLY) =====
5001 '
5020 FOR Z = 0 TO 2 : FOR Y = 0 TO CY + 1 : FOR X = 0 TO CX + 1
5030 PRINT@ (CY + 3 + Y)*64 + 3*X + 20*Z,A1(X,Y,Z);
5040 NEXT X : NEXT Y :
5050 NEXT Z : RETURN
```

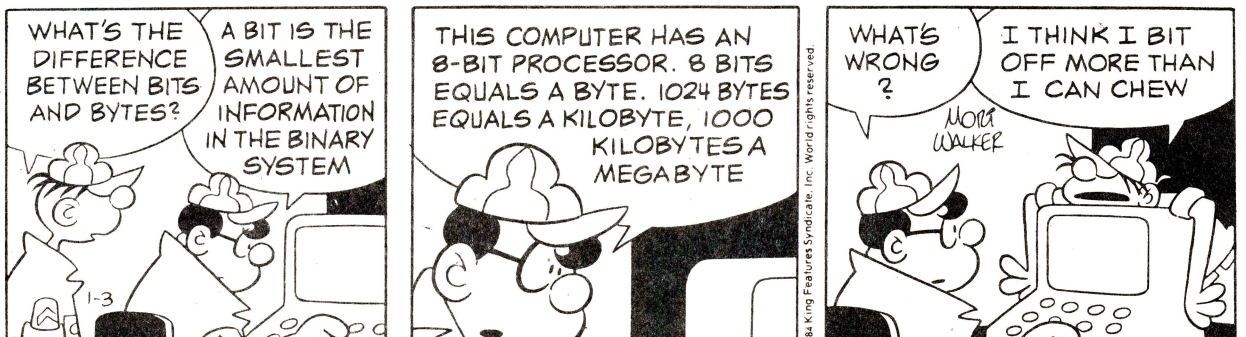
## SPECIAL ABACUS AWARD

Hear ye, Hear ye, all FCUG bulletin board users. Robert Fowkes has had the dubious honor of being caller number 1000 on the system. This places great responsibility on him. From this day forward and forever more, (and even longer if possible), Bob is responsible for keeping track of the FCUG bulletin board caller count, without computer assistance. As an aid in this important FCUG function, he will be allowed to use the official FCUG abacus, (purchased of course at his own expense), in tracking this data. Bob, we shall be anxiously awaiting your first report at the next meeting.

FRANK & ERNEST




BEETLE BAILEY



# RECOVERING TEXT

by Don Chapman
Reprinted from APCU Newsletter
December 1984, Vol. 6, No. 12

I've been using Apparat's FLEXTEXT modified version of Model I Scripsit so much I forgot that standard Scripsit does not allow recovery of the text in memory if you exit the program. I don't have a Model III/4 version of FLEXTEXT; the Apparat conversion doesn't "take" on Model 4 Scripsit. So the inevitable happened. Running the Model 4P, I exited to TRSDOS to check a directory. Back to Scripsit; no text! Arrgh! (I know; I should have been saving it to disk more often to minimize the loss, but this was a hasty session.)

DEBUG to the rescue! This technique works even if the system was reset, as long as power stayed on and you haven't loaded something that rewrites the text area of RAM.

From TRSDOS Ready, type DEBUG <ENTER>, then <BREAK>. Use the S command to get a full-sector display, and A to see the codes in ASCII characters (LDOS and TRSDOS 6 show ASCII at the right of display all the time). Now, step through RAM using the <;> and <-> keys. You'll see fragments of your text in some blocks, but ignore them. Look for your text in a recognizable continuous block. Note the addresses of the bytes at the beginning and end of the useable text. Don't complain if some at the beginning or end is gone; save what you can. Write these down, then exit to TRSDOS Ready by keying G402D (Model I) or <0> (Model 4), then DEBUG (OFF).

Now enter: DUMP filespec (START=addr1, END=addr2, ASCII). In which addr1 and addr2 are the addresses of the beginning and ending bytes noted from DEBUG, and filespec is a disk file to which you would like to save the rescued text. The ASCII parameter causes the RAM contents to be saved in standard text format rather than as a loadable machine code program.

Now, reload Scripsit and load "filespec" to it. You should have a good text display to edit and resave. If the text is severely truncated, you may have gotten a "zero" byte in the saved file. Repeat the process (most of the text will still be there in RAM) but use the M command (Model I) or A or H commands (Model 4) to change any "zero" or "255 (FFhex)" bytes you find in the text to valid ASCII codes before leaving DEBUG.

This illustrates one of the delights of TRSDOS and its relatives; most programs that access text files use the same file format or provide a parameter to do so, VisiCalc (both /VC and /DIF files), Profile, Scripsit, Basic programs, and Basic Sequential data files all can be saved in this form and swapped in various ways.

# The President's Page

```
*********************************************
** >>>>====>  NEXT FCUG MEETING  <====<<<< **
**                                         **
**   DATE :  February 5, 1985 ( Tuesday )  **
**                                         **
**   TIME :  7:30 pm                       **
**                                         **
**   PLACE:  New Canaan Country School     **
**                                         **
*********************************************
```

In the last meeting we had two great demos. Jeremy Gratt explained how to program in machine language from the view point of a new learner. A few people came up to him after the demo and told him that his talk clarified some of their misconceptions. Later in the evening Roger Giler tackled the difficult topic of file handling which led to a thirty minute question and answer period.

Today we are in the midst of the information revolution where more people are involved in the processing of data than in the manufacturing of products. There are new jobs, such as information brokeraging, to replace old jobs. Information can be bought and sold, or it can be shared, and it is this sharing that we are involved with in this club. Dollar for dollar FCUG, INC. offers the best value for your money.

This month we are going to have two powerful speakers (we should sell tickets to this one). SID GROSS will give us a talk on S.L.O.B. I thought he was going to talk about me, but it turns out this is a Simple Lesson On Boolean. And / Or but Not inclusive of programming Nor logic. Figure that one out.

And by popular demand, a speaker unparalleled on the computer circuit (pun), LARRY GILBERTI will bring us up to date on The Integrated Office System. Now here is a quiet spoken man who does not settle for the mediocre in software. Most of us sit back and lament what our software will not do, but Larry calls the authors and ask WHY NOT, and these hard working gentlemen drop everything and rewrite the software. This is going to be a meeting to remember. Be there.

Bob Jackson

# Voice of the '80
## Index for February


ENTOMOLOGY (Bugs) .... Page 1
Dick Basic Error Codes Page 2
DRIVING YOURSELF CRAZY Page 3
TRSDOS 6.2 ..... Page 8
HI-RES Epson Graphics Page 11
HI-RES Sample Pictures Page 13
LIFE/BAS ..... Page 16
RECOVERING TEXT ..... Page 19
Level II Error Codes . Page 2
NEWDOS-80 Error Codes Page 2
New Items From Tandy . Page 6
Ask Alan (JCL files) . Page 10
HIRES/BAS ..... Page 12
LIFE BEHIND WALLS .... Page 14
SPECIAL ABACUS AWARD . Page 18
The President's Page . Page 20



The Voice of the '80 is a publication of the Fairfield County Computer Users Group, Inc. Permission to reprint is granted for non profit and non-commercial uses. Credit is appreciated. Submissions should be sent to the Editor at the address below:



PRESIDENT: Robert Jackson
VICE PRESIDENT: Herman Parks
SECRETARY: Patricia Brinson
TREASURER : Roger Giler
EDITOR : Alan Abrahamson
BULLETIN BOARD: (203) 846-6728
Ask Alan HOTLINE: Evenings (6PM-10PM) (203) 866-7883



This Newsletter printed by Sir Speedy Printing Center, Wilton CT
{{{{{{{{ This months cover art by Herman Parks }}}}}}}}
Licking, sticking, folding & stapling by: Herman and Jeff Parks



FAIRFIELD COUNTY COMPUTER USERS GROUP
% Alan Abrahamson
10 Richlee Road
Norwalk, CT 06851


FCUG

FIRST CLASS

TO: