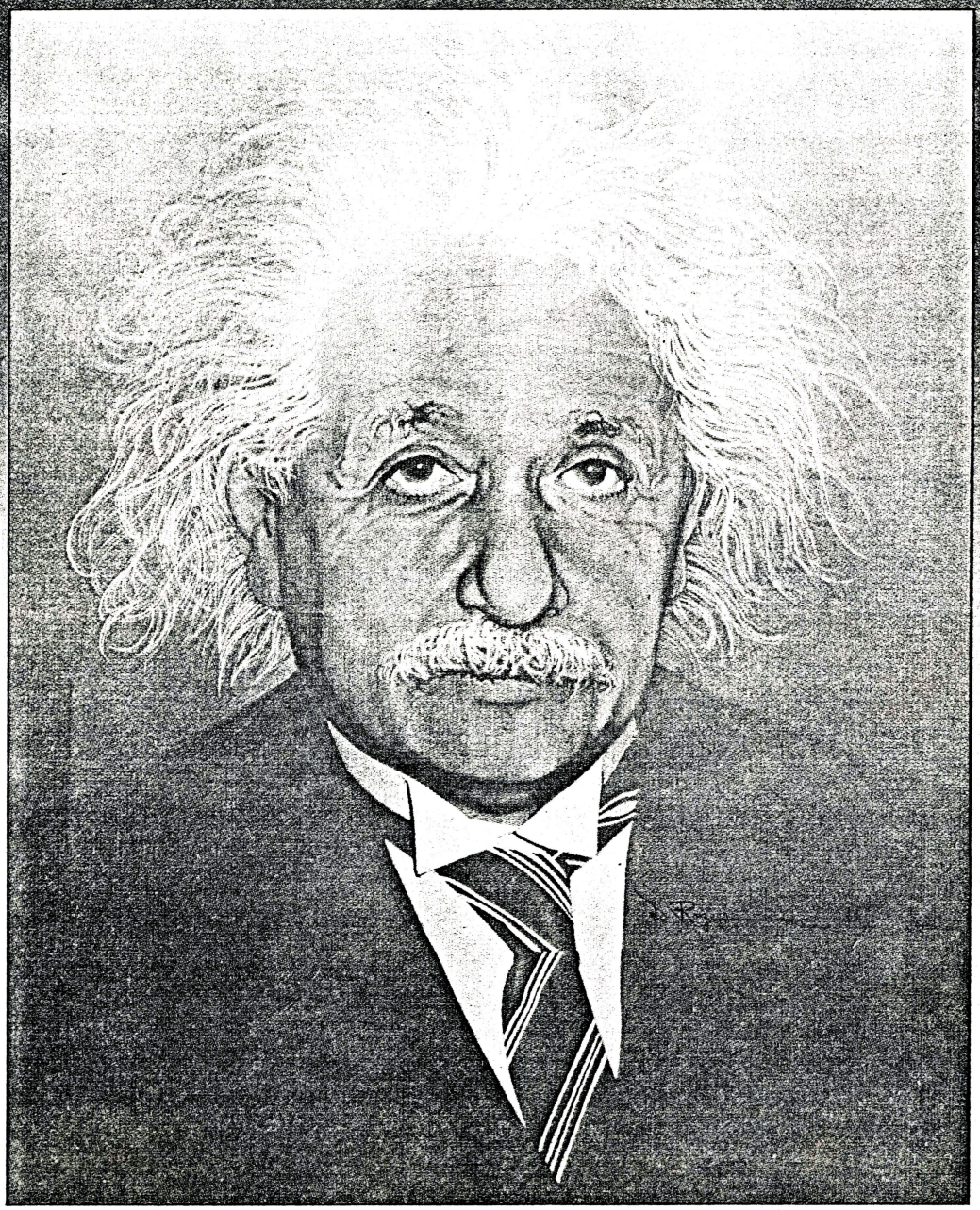
"Everything should be made as simple as possible, but not simpler."



Albert Einstein, 1879–1955

The Voice of the '80
Newsletter
Volume 3 Number 4

Using LMOffset

by John T. Blair & Peter B. Hall

Reprinted from DATA PORT, June 1982

This article will attempt to remove some of the mysteries surrounding LMOFFSET, and show how to use it to its fullest extent.

LMOFFSET is one of the utilities supplied by Apparat with their NEWDOS+ disk operating system and stands for Load Module OFFSET. In its simplest usage, it is used to move a machine language program from tape to disk. It can also be used to copy a program from disk to disk. But why, especially since there is a COPY function, you ask? We'll go into that in a few minutes.

For those readers who are just getting started with their TRS-80, machine language is the language that the computer really talks in, not BASIC. These are the programs that, when loaded from BASIC, had to be loaded by the SYSTEM command. When displayed with DIR command in DOS, they are the ones which have the extension "/CMD".

When LMOFFSET is first executed, it displays the sign-on message:

```
APPARAT LOAD MODULE OFFSET PROGRAM, VERSION 1.1
 SOURCE FROM DISK OR TAPE? REPLY "D" OR "T".
```

Assume that you have a machine language program on tape which you want to move over to your newly acquired disk. A good example of this would be all those SYSTEM games tapes or the tape version of ELECTRIC PENCIL. (Keep in mind that just because the program has been moved to disk, it will NOT allow any disk LOADs or SAVEs, unless the program originally had that option. In this example of ELECTRIC PENCIL, you will still have to save files out to tape. The only thing you have gained is the decrease in loading time of the program itself.)

Referring to figure 1, a "T" was entered. LMOFFSET then loads the program into a buffer, above LMOFFSET. Immediately after the T is entered a "*" appears in the upper right corner of the CRT and will not blink. After the A5 sync byte is found, two more *'s appear. These will blink unless an error is found. The error codes are:

C = Bad check sum,
P = Leading extraneous bytes,
I = Inbedded extraneous bytes.

When the tape has finished loading, program storage area is displayed in the third line. There are three possibilities as to where the program is loaded, and three messages:

1. Program loads from 7000H or higher.

```
MODULE LOADS TO XXXX-XXXX
ENTRY POINT = YYYY
NEW LOAD BASE ADDRESS (HEX)?
```

Since this program will not conflict with DOS all you have to do is press "ENTER" and it will prompt:

```
DESTINATION FILESPEC?
```

(NOTE: If you have more than one drive, be sure to assign the drive number.)

In the next two cases you will have to specify a "new load base address" in order to load the program (more on this later).

2. Program loads from 4000-51FFH.

```
MODULE LOADS TO XXXX-XXXX
MODULE LOAD OVERLAPS DOS RAM (4000-51FF)
MODULE LOAD WILL OVERLAP "CMD" PROGRAM AREA (5200-6FFF)
ENTRY POINT = YYYY
NEW LOAD BASE ADDRESS (HEX)?
```

In this case, if any or all of the new program loads in the DOS overlay area, the new program and DOS will collide and cause a REBOOT.

3. Program loads above 51FF but below 7000H.

```
MODULE LOADS TO XXXX-XXXX
MODULE LOAD WILL OVERLAP "CMD" PROGRAM AREA (5200-6FFF)
ENTRY POINT = YYYY
NEW LOAD BASE ADDRESS (HEX)?
```

If a program were to reside here, you will lose the ability to read and write to the disk.

NEW LOAD BASE ADDRESS has to be answered before you can continue. What this is asking for, is where in memory the user wants the program to be stored. If the program read from tape loads above 7000 (hex) then just press ENTER. If it did not enter 7000 (more on why this is done later).

The program will now ask "SHALL APPENDAGE BE SUPPRESSED (Y OR N)?". If you entered any number in the previous step, you SHOULD answer "N". This appendage is a new LOADER which will move the program into the correct area for execution. (More on this later.)

The program calculates the new storage area and entry point and displays this in lines 9 and 10 of the printout. Again a NEW LOAD BASE ADDRESS is asked for. If you made a mistake in entering it before, this is where you can correct it. If no changes are to be made, press the ENTER key.

Now the interrupts can be disabled. The answer to this question should be "Y".

Finally, the file specification is requested. If you plan to use this program and are moving the program from tape to disk, you must conform with the DOS requirements, the filename should have an extension "/CMD". However, if you plan to work on this copy, we suggest the extension "/MOV" to keep from confusing it with the original. If the extension "/CMD" is not used you will be able to execute the program from DOS, but you will not be able to LOAD it.

LOOKING DEEPER

What does LMOFFSET really do? To answer that let me just brush on machine language for a minute. Most machine language programs are NOT relocatable. This means that when they were written, several instructions made reference to a specific memory location. Using ELECTRIC PENCIL as our example, from the printout in figure 1, the program loads from 4000 - 51FF. Many memory cells in this program are used for data storage, such as: the number of lines to be printed, the end of text area, etc. To get the data from these areas E.P. will load the contents of the desired memory cell. Suppose that 4400 contains the end of text pointer. If this program were moved to 7000, the data stored in 4400 is now at, say 7400. When the program wants this information it addresses 4400, not 7400. Therefore, the data will be wrong. This means the program is not relocatable. There are other instructions such as LD's, CALL's, and JP's, and the some data tables are arrangements, that make a program not relocatable.

Who cares about all this mumbo jumbo? So why do we have to move a program anyway, you ask? The reason is simple. The memory map in the back of the Level II manual, shows that the memory from 43E8 and up is used for program storage. When the disk is added then memory from 43E8 to 5200 is used to store the Disk Operating System (DOS). If ELECTRIC PENCIL were not moved up in memory, as it loaded from disk, it would overwrite the DOS. EVentually, the DOS would crash. Therefore, you could not load PENCIL. This is why we moved it up to 7000. But didn't we just say something about relocatable, and that the program would not run in that memory area? Yes we did. So how can we use it? The answer is simple. You CAN'T! But that is the reason for the LOADER.

The LOADER is nothing more than a "block move" program. This appendage consists of 15 bytes of machine code which is tail ended to the original program. The first byte is an F3 (to disable interrupts) or a 00 (NOP) depending on the answer you give a later question. The next 3 bytes are 21 XX YY (LD HL,YYXX) which sets where the program is to be moved FROM. The next 3 bytes are 11 SS TT (LD DE,TTSS) which sets where the program is to be moved TO (the "New Base Address", so by Golly we'll put it back there). The next 3 bytes are 01 DD EE (LD BC,EEDDH) which sets the number of bytes of program to move (the difference between the original starting address and the original ending address without the appendage). The next 2

bytes are ED B0 (LDIR). This little instruction does a lot for its size. First it takes the bite pointed to by the HL register pair and moves it to the address pointed to by the DE register pair, and decrements (subtracts one) from the BC register pair, and checks to see if the number in the BC equals 00. If it does it stops the moving and lets the machine go to the next instruction. The final 3 bytes are C3 FF GG (JP GGFF) which tell the computer to go and execute the original program. That's it! All 15 bytes, and the new program is where it wanted to be. (We'll cover an exception later).

NEW LOAD BASE ADDRESS

The NEW BASE ADDRESS was entered above 6FFF, to keep it out of the DOS program area. This allows the system to load the program from disk to memory. The entry point was changed to the loader. After the program has been loaded, execution is transferred to the LOADER. Since the DOS is no longer required (for a program that does not allow disk I/O), the program is moved down to the area where it is required to run. Then the loader jumps to the original entry point, and you are off and running.

What do you choose for a "new load base address"? It depends on how much memory you have, and what you want to do with the new module. Assuming you want to disassemble a program to modify it, you have to decide what tools (utilities) you will use to do the work. Once you have decided that, you must know where it loads. Now based on where the utilities reside in memory, you can determine if the new module will load above 7000H and below your utility or if you must load the new module above the utility.

The start of memory, for the new module, is what is entered for the "NEW LOAD BASE ADDRESS".

WORKING ON A PROGRAM

Figure 1 is a printout of the new block move routine. After the question, "should the appendage be suppressed?", LMOFFSET will display the new storage area, and the new entry point. Make a note of this information.

To work on this program, you must now load it into memory using the LOAD (be sure to include the extension with the file name) command from the DOS, and execute the monitor or utility you wish to use.

There are two methods for working on a program once you have it in memory. The first is to work on it in the new area. This method is not recommended since many of the machine language instructions will be directly addressing a memory location, and they will all be referring to the cells where the program was designed to run. This will make decypherin the program difficult.

The second method is to load the moved version into memory, and execute the monitor. If you wish, you can "block move" the program down to where it is designed to run, or modify the loader. To modify the loader, disassemble it and change the address of the JP instruction to that of your monitor. Now execute the program with the new entry point. After the loader has moved the program to where it is to run, it will return control to your monitor. After you have modified the program, you must block move it back to where it was. (Remember, you do not have any disk functions if the program was to load below 5200H, and if it loads below 6FFFH many of the DOS utilities will write over it.)

To save your modified program out to disk, load any utility you have, and save the contents of memory back over the disk file.

EXCEPTIONS

1. Scott Adam's adventure program may or may not load and run. It will be evident when the screen goes "BONKERS". To correct this, Reboot while pressing the shift and up arrow. When the DOS READY prompt comes up let up on the keys. You have now disabled the key debounce. (See how sloppy your key board really is!) This routine conflicts with other programs as well. You can now run the Adventure program and enjoy.

2.If the program has a loader that loads the program (i.e. Sublogic's FS-1, Flight Simulator) you might be out of luck. The easy way to find out is compare the time it takes to load and run the program from tape to the time it takes to load it into LMOFFSET. If it only takes one minute to load a program into LMOFFSET and normally takes 5 or 6 minutes it has a loader. You will now have to disassemble the loader and try to figure out what is going on, or find a friend who has already broken the loader and gotten the program on disk.

ADDITIONAL TOOLS

We have found the following programs very useful. For an assembler, we prefer EDAS from MISOSYS. They also sell an excellent disassembler (that will build an equate table, labels all memory references, and can save to disk in a format that any of the Radio Shack compatible EDITOR ASSEMBLERS can read). RSM-2D is an excellent monitor, but lacks two very important features. It can not relocate a program, only block move it. Nor can it write memory to a disk file. MON4 will handle these two short-comings, but we have listed about ten items in it we do not like. Armed with these and the other utilities from NEWDOS+, you can really tear into the DOS's or any other program.

One example of what you can do if you set your mind to it, with the help of a friend we broke the loader for FS1 Flight

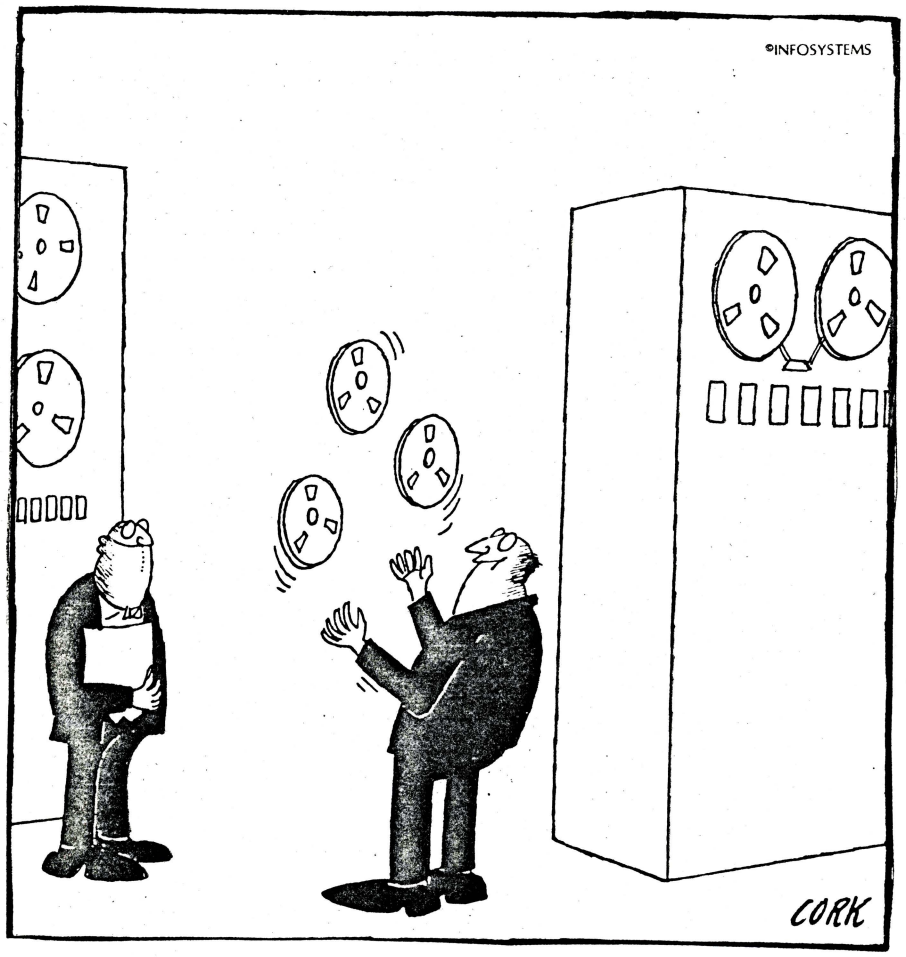
Simulator from Sub logic and moved it to disk in less than eight hours.

There is a good bit of trial and error when you first go into a program to modify it, or to break a funny loader.

---APPA

```
RAT LOAD MODULE OFFSET PROGRAM, VERSION 2.0
SOURCE FROM DISK OR TAPE? REPLY "D" or "T"? T
MODULE LOADS TO 4200-5500
MODULE LOAD OVERLAPS DOS RAM (4000-51FF)
MODULE LOAD WILL OVERLAP "CMD" PROGRAM AREA (5200-6FFF)
ENTRY POINT = 4350
NEW LOAD BASE ADDRESS (HEX)? 7000
SHALL APPENDAGE BE SUPPRESSED (Y OR N)? N
MODULE LOADS TO 7000-830F
ENTRY POINT = 8301
NEW LOAD BASE ADDRESS (HEX)?
DISABLE INTERRUPTS? (Y OR N)? Y
DESTINATION DISK OR TAPE? (D OR T) D
DESTINATION FILESPEC ? PENCILT/CMD
DONE
**************ANOTHER DESTINATION FILESPEC? (Y OR N) N
ANOTHER SOURCE FILE TO PROCESS? (Y OR N) N
```

Figure # 1 -- LMOFFSET Output for the ELECTRIC PENCIL



No Seasonal Respite for the Computer Widow

by Art Buchwald

For every home computer sold in America, there is a computer widow somewhere.

I dropped over to see the Bengals the other night. Mrs. Bengal offered me a drink.

"Where's Walter?" I asked Adele.

"Where he always is these nights. He's in the library talking to his home computer."

"He talks to a computer?"

"All the time. It's taken the place of television, conversation and foreplay," she said bitterly.

"I didn't know Walter was into computers."

"That's all he's into. As soon as he finishes dinner, he leaves the table and says, 'Well, I've got to go in and program a new household fiscal budget for 1983.'"

"At least he's working to save you money."

"He says he's working on a new budget, but I walked in last night and he was playing Star Wars. He told me he was just checking out the disk drive. I've never felt so alone in my life. At least when he watched football I could sit next to him. But now that he has a home computer he says he has to be alone with his software."

"You poor kid. Maybe he'll tire of it."

"No way. He reads computer magazines the way he used to read Playboy. I could compete with another woman, but I can't compete with a computer. I'm at my wits' end."

"Your not thinking of leaving him?"

"I threatened to last week and he said to hold off until he could program all the variables, and come up with a modified alternative."

"Maybe I should talk to him," I suggested.

"You can try, but I doubt if it will do any good."

I went into the library and found Walter hunched over his keyboard. "Hi, Walter. Am I disturbing you?"

"No," he said, squinting at me. "I was only justifying my margins."

"How's life?" I asked.

"Fine. I was having a problem with my cursor for a while, but I straightened it out by adding a protocol."

"You have to be careful of cursors," I said. "What news of Adele?"

"Wait a minute," he said, "I'll find out."

He put in a disk, pushed a code key, and typed on the screen ADELE. Then he hit his RETURN button.

"Here it is," he said. "She's either in the kitchen, the bath, her bedroom or went to a baseball game."

"A baseball game?"

Walter looked worried. "That doesn't sound right. But it's no problem. All I have to do is hit this DELETE button."

"Adele thinks she's losing you to a floppy disk retrieval system," I said.

"That's ridiculous," Walter said. "All I'm trying to do is store and index data that will be able to forecast how we can enjoy the September years of our life."

"I'm going to ask you a very personal question, Walter. How much do you love Adele?"

Without saying a word, Walter inserted a disk, and started hitting the keyboard.

"What are you doing?" I asked.

"I'm counting the ways. It's much faster to do it on a computer."

Special Condensed
Programs Section
by Voice of '80
STAFF

ADVENTURE

```
10 PRINT "YOU ARE AT THE BOTTOM OF A 20 FOOT PIT"
20 INPUT" NOW WHAT?";A$
30 X= RND (0)
40 IF X<.5 THEN PRINT"I DON'T KNOW HOW TO ";A$:GOTO10
50 IF X>.5 THEN PRINT"YOU WERE KILLED":END
60 IF X=.5 THEN PRINT"YOU WIN":END
```

ASTEROIDS

```
10 PRINT"HIT ANY KEY"
20 A$= INKEY$: IF A$ = "" THEN 20
30 PRINT"** ASTEROIDS ** YOU LOSE!!!":END
```

ELIZA

```
10 PRINT"HI. WHAT IS YOUR PROBLEM?"
20 INPUT A$
30 IF LEFT$(A$,1)="I" THEN PRINT "EGOTIST":GOTO10
40 IF LEFT$(A$,3)="YOU" THEN PRINT "YOU TOO": GOTO 10
50 PRINT" ARE YOU SURE ";A$;" ?": GOTO 10
```

STAR TREK

```
10 X = INT (RND(0) * 40): Y = INT (RND (0) * 40)
20 PRINT "POSITION",X,Y
30 PRINT"PRESS 1 TO FIRE, 2 TO MOVE"
40 A$= INKEY$:IF A$="" THEN 40
50 IF A$="1" THEN 10
60 IF A$ ="2" THEN 80
70 PRINT" YOU HIT THE DESTRUCT KEY * * * B O O M * * *": END
80 IF RND (0) = .1 THEN 100
90 IF RND (0) = .2 THEN 120
100 GOTO 90
110 REM
120 IF X = 14 ANDCLSINPUTREMREMTHEN22227789PRINT!!$$$)(COS*
RIGHT$NEXTGOTORETURNFOROFORFORCLSLEFT$STRING$USRDEFUSRINKEY$
RESTOREPUTGETGOTCHA!!!
```

MINUTES of AUGUST

by Alan Abrahamson

At approximately 8pm on August 3rd at the new canaan Country School the monthly meeting of FCUG was called to order by our President, Sid Gross. 50+ members and guests were in attendance. One special guest was Barry Kornfeld, President of the Metro TRS-80 Club and owner of Stoney Clove Computer Store. He presented a special deal on the Alternate Source magazine to the membership. It was not a "tough sell" since it is a top-notch magazine and no serious user should miss an issue. Thanks, Barry - please come again and bring some of the guys from Metro along.

After transacting some minor business, we proceeded onward to the "good stuff".

Our first presenter was Larry Gilberti, with his Integrated Insurance Adjusters Office System which includes, VISICALC, MAXICRAS, LAZY WRITER, PROFILE, TIME MANAGER and COMMUNICATIONS. Larry explained in great detail how he was able to get all these separate pieces of software to "TALK" to one another. It was quite a show. Larry's enthusiasm and working knowledge of all the software mentioned above was evident. With wife, Linda, prompting from the wings, he took all of us through this elaborate system. If you missed it, perhaps we can convince Larry to do an encore.

Next up was Dick Carricato with a super demo on USR calls from Basic. I can remember fighting all those POKEs several years ago in Level II. Dick's presentation was very well received as well as understood by most. He has the ability to take a highly complex subject and simplify it down to a level that can be understood. If I had not taken Dick's Assembly Language course I think that I would still be totally in the dark regarding machine language programming.

The concluding presentation was by Milt Thrasher with a new voice box from ECHO. It seemed to be a very nice unit and competitively priced with the VORTRAX. The beauty of this unit seemed to be the ease at which it could be programmed for speech using simple LPRINT statements. Milt combined the speech demo with his own version of a large letter graphic program that "wrote" on the monitor in 2 inch high letters, poems by Ogden Nash as the ECHO pronounced the words. Howls of laughter emitted from the group at the poetic quips. Dynamite Job, Milt.

The time to end another FCUG meeting came all too soon as we packed all the gear away for next month. See you then. Al...

A N N O U N C E M E N T S

Thanks to the generosity of FCUG member George Ahmuty, the FCUG now has a Bulletin Board. FCUG Central is available for your use 24 hours a day. Telephone data line (203) 227-4946 in Westport, Connecticut. FCUG Central can be used for public or private messages between members or notices to ALL members can be posted. The system in use is a Model I with 3 disk drives running Newdos/80 Version 2 and the Bullet/80 Bulletin Board system. UART settings are standard 8 bit, 1 Stop Bit and 300 or 1200 BAUD is available. If you would like to UPLOAD to the board - leave a message for George. This system is for your use and will be what you make it. I am considering making back issues of "The Voice of the '80" available for download. Let me know what you think of this idea.

Don't forget the Microcomputer Fair to be held on Sunday, September 26th at the Westport Center for Environmental Activities. This event will run from 11am to 5pm featuring:

SEMINARS - DEMONSTRATIONS - ROUND TABLES on the following:
BUSINESS - HOME - GAMES - INSTRUCTION - PROGRAMMING
HARDWARE - SOFTWARE - COMMUNICATIONS - etc. - etc.

BOOTHS FOR PRODUCTS, PUBLICATIONS, SERVICES, ORGANIZATIONS

FLEA MARKET - REFRESHMENTS - and LOTS of FUN !!!!!!!!!!!!!

If you have not as yet signed up to help or need more information, please contact FCUG Fair Chairman, Ira Friedman at 116 Mile Common, Easton, CT 06612. We need people to bring their equipment for demo's. PLEASE HELP OUT!!!!!!!!!!!!!!!!!

Al.....

From KINGS BYTE NEWSLETTER July 1982

PRINTER DRIVERS
by
Louis Shapiro

This article contains two printer driver programs, one each for the TRS-80 Model I & Model III. They are both written in source code. This source code takes a little explanation. Type it into an assembler, and then assemble it as PRT/CMD. Notice that there are different versions for Mod I and Mod III. If you don't have an assembler, I assume that the assembled versions will be available from the Kings Byte software library. (Note:- Also from FCUG library)

Once you call PRT from DOS, it will live in high memory until you reboot. From then on, anytime the symbol '#' appears in ANY text to be printed, the next character will be

interpreted as an Epson control character. The following is a list of possible commands:

- #0 resets Epson to normal type
- #B rings the printer bell
- #W goes to double-wide
- #w turns off double-wide
- #C goes to compressed
- #c turns off compressed
- #F issues a formfeed
- #E goes to emphasized
- #e turns off emphasized
- #S goes to double strike
- #s turns off double strike
- #G turns on graphics mode - adds 32 to graphics characters. (PRT begins in Graphics mode on.)
- #g Graphics mode off
- #I Italics on (requires Graphtrax)
- #i Italics off

The italics functions haven't been tested, since I don't have Graphtrax. Also remember that the graphics may conflict with DOS graphics options. That is if your DOS is adding 32 to each graphic block, you don't want PRT to do it too!

If you do the zaps to allow Scripsit to honor 4049H, and the ones to make it use the normal printer DCB, then PRT will let you place control characters within your Scripsit text (using '#x'). By the way, if the letter following the # is not a valid command, it will be printed normally. So, if you need to print '#', then you must type '##'.

The following zaps were part of a set distributed at an early Kings Byte meeting. Unfortunately I don't know where they originally came from. Thanks, Masked Man. These zaps are in file-relative sector, byte format. (Note:- Humfrey Udall has added the starting memory locations for the disk (and tape) versions of Scripsit. These locations start at the SECOND byte in the tables below. The first and last bytes are not changed and may be different in the tape version. Don't change them.)

To make Scripsit honor HI$ at 4049H:

```
00,63        CHANGE 7C 21 FF 00 25 7E 2F 77 AE 20 F9 22
525D (435D)     TO 7C 2A 49 40 00 00 00 00 00 00 00 22
For Mod III:           11 44                          (HI$=4411H)
```

To make Scripsit use the normal printer DCB:

```
00,05        CHANGE 3D 43 4F 50 59 52 49 47 48 54 20 31 39 37
5203 (4303)     TO 3D FD E5 D5 F5 CD 3B 00 F1 D1 FD E1 C9 37

00,47        CHANGE 0A 32 E8 37 AF
5244 (4344)     TO 0A 00 00 00 AF

13,9A        CHANGE 60 3A EB 37 CB
5F63 (4F18)     TO 60 3E 3F 00 CB

20,92        CHANGE C9 3A E8 37 E6
663F (5617)     TO C9 3E 3F 00 E6
```

```
20,A3        CHANGE 60 3A E8 37 E6
6650 (5628)      TO 60 3E 3F 00 E6

20,B1        CHANGE 0D 32 E8 37 32
665E             TO 0D CD 02 52 32
(5636)                 CD 02 43

21,79        CHANGE 0D 32 E8 37 D3
6722             TO 0D CD 02 52 D3
(56EC)                 CD 02 43

41,3A        CHANGE 20 32 E8 37 08
7A97             TO 20 CD 02 52 08
(69B8)                 CD 02 43

41,41        CHANGE 5F 32 E8 37 C3
7A9E             TO 5F CD 02 52 C3
(69BF)                 CD 20 43
```

I'll repeat, PRT will let you output Epson control codes ANYTIME you send text to the printer. You can LPRINT "#EHello", and the word "Hello" will be printed in emphasized mode. If the first line in a BASIC program is: 1 REM #C, it will LLIST in compressed mode.

Some combinations of commands are not permitted by the printer, such as emphasized with compressed. Also double-wide print gets turned off after each carriage return. Don't blame those on me. If you have any questions, drop me a note on a local bulletin-board, or at MicroNET 70210,306 or Source TCB792.

(Note:- The source code printed is for Model I. For Model III lines 100,110,160,170,290 and 1380 need to be changed. The changes are given in the comment space following a colon.)

```
;PRT1/SRC WRITTEN BY LOUIS SHAPIRO : III  PRT3/SRC
;PRINTER DRIVER FOR MOD I SCRIPSIT : III
;AND EPSON PRINTER
;FROM KINGS BYTE NEWSLETTER JULY 82
;BROOKLYN NY
DISPLA  EQU     4467H   ;DOS CRT CALL : III  021BH
        ORG     4049H   ;MOD I HI MEM : III  4411H
        DEFW    BUSY-2
        ORG     0FE00H
START   PUSH    HL
        LD      HL,MSG
        CALL    DISPLA
        LD      HL,(4026H)
        LD      (OLDDVR),HL
        LD      HL,BUSY
        LD      (4026H),HL
        POP     HL
        RET
MSG     DEFM    'MODEL I PRINTER DRIVER' : III
        DEFB    0DH
        DEFB    0
```

```
BUSY    LD      A,(CTLFLG)
        CP      0FFH
        JR      Z,CTLOUT
        LD      A,C
        CP      '#'
        JR      NZ,NORCHR
        LD      A,0FFH
        LD      (CTLFLG),A
        RET
NORCHR  LD      A,(GPHFLG)
        CP      0H
        JR      Z,CONTIN
        LD      A,C
        CP      127
        JR      C,CONTIN
        ADD     A,32
        LD      C,A
CONTIN  DEFB    0CDH
OLDDVR  DEFW    0000
        RET
CTLOUT  LD      A,C
        CP      'B'
        JR      Z,BELL
        CP      'W'
        JR      Z,WTH
        CP      'C'
        JR      Z,COMP
        CP      'F'
        JR      Z,FORM
        CP      119
        JR      Z,WTHOFF
        CP      99
        JR      Z,CMPOFF
        CP      'E'
        JR      Z,EMPH
        CP      101
        JR      Z,EMPOFF
        CP      'S'
        JR      Z,STK
        CP      115
        JR      Z,STKOFF
        CP      '0'
        JR      Z,RESET
        CP      'G'
        JR      Z,GPHON
        CP      103
        JR      Z,GPHOFF
        CP      'I'
        JR      Z,ITLON
        CP      105
        JR      Z,ITLOFF
        JR      EXIT
BELL    LD      C,7
        JR      EXIT
FORM    LD      C,12
        JR      EXIT
WTH     LD      C,14
        JR      EXIT
COMP    LD      C,15
```

```
        JR      EXIT
WTHOFF  LD      C,148
        JR      EXIT2
CMPOFF  LD      C,146
        JR      EXIT2
EMPH    LD      C,69
        JR      ESC
EMPOFF  LD      C,70
        JR      ESC
STK     LD      C,71
        JR      ESC
STKOFF  LD      C,72
        JR      ESC
ITLON   LD      C,52
        JR      ESC
ITLOFF  LD      C,53
        JR      ESC
GPHON   LD      A,0FFH
        LD      (GPHFLG),A
        LD      A,0
        LD      C,A
        JR      EXIT
GPHOFF  LD      A,0
        LD      (GPHFLG),A
        LD      C,A
        JR      EXIT
RESET   LD      A,146   ;RESET COMP. OFF
        CALL    OUTCHR
        LD      A,148
        CALL    OUTCHR
        LD      A,27
        CALL    OUTCHR
        LD      A,70
        CALL    OUTCHR
        LD      A,27
        CALL    OUTCHR
        LD      A,53
        CALL    OUTCHR
        LD      C,72
ESC     LD      A,27
        CALL    OUTCHR
EXIT    LD      A,0
        LD      (CTLFLG),A
        JP      NORCHR
EXIT2   LD      A,0
        LD      (CTLFLG),A
        JP      CONTIN
OUTCHR  LD      (37E8H),A ; : III  OUTCHR OUT  (0F8H),A
        CALL    CHKPRT
        RET
CHKPRT  LD      A,(37E8H)
        AND     0F0H
        CP      30H
        JR      NZ,CHKPRT
        RET
CTLFLG  DEFB    0
GPHFLG  DEFB    0FFH
        END     START
```

```
*****************
*  MINI-DATABASE  *
*       by        *
*  Roger R. Giler *
*****************
```

Each time that we hear about a database program, we tend to think of a huge program, quite costly, and with numerous mysterious features. But not this one! This program was written in one evening to perform a typical database function. My wife wanted a record of our get-togethers with friends, where she could later retrieve vital information, such as the food that was served, the dress that she wore, how I behaved, if the police had to be called, etc... At the same time, I had been asked by some of you for a better example of the use of the INSTR function in a typical menu.

This program also contains several features that may prove of interest to you. But first things first... What does it do? Look at the menu on line 200 where you can enter any information that you choose, retrieve it, print it, show all the files (names) on the disk. You can also enter a date or a comment that is then picked up with every entry made to the disk during that session. This program differs from other database programin that it makes use of many of the features of Newdos80, and allows the machine do most of the work. Let us now review the listing in detail, discussing special or interesting features as we encounter them.

* LINE 20 : Here we look at the date-time string TIME$ in memory which is 17 bytes long. Just try this. In Basic, type PRINT TIME$ and the screen will show both the date and the time as 08/05/82 23:46:37, for example. If no date was entered in memory, the first two characters will be 00, and this is what we look for. If they are not zero, we assume that the date is valid, use LEFT$ to read the full 8 characters that make up the date, plop them into D$ and hop to the menu at line 200. If we find 00, then we continue to line 30.

* LINE 30 : Here we input the date in the proper format into D$ and at the same time load it into memory by use of the CMD function. This command lets us do operations from Basic, or from a Basic program. Let me elaborate. In DOS I would enter the date by typing DATE and 08/05/82. From Basic, I assign this date in the form of a string variable to Z$ (so that Z$ = "DATE 08/05/82") and then execute the command CMD"Z$". If we were to use a date format that the TRS-80 does not like, the ON ERROR GOTO command takes charge and moves execution to line 900, where, since the error is a No-no # 58, pushes us on to line 910 which recognizes the error and gives us another chance to do it correctly.

* LINE 200 : It is similar to most menus except that the whole

screen is done with a single program line. To get the linefeed, use the DOWN ARROW.

* LINE 210 : Here we use the INKEY$ function that scans the keyboard and put its value into M$. If M$ is blank (M$ = "") we run that line again, if not, we continue to the next line.

* LINE 220 : This is the line where we use the INSTR function to good advantage. I am using this function to give a number which is the relative location of the search string M$ in the searched string. This number is the one that will be used in the ON GOTO function. The first argument of INSTR is the list of the 6 keys that I want to recognize (in both upper and lower case, if applicable). The second argument is M$ which represents the key that has been pressed in line 210. INSTR will then tell us where in string "EeLlPpSsCcQq" the content of M$ found a match. If M$ = S then INSTR returns 7 which if used in the ON GOTO, would send us to the 7th line number listed. Actually, I have added 1, in case INSTR returns with a 0 which is not allowed in the ON GOTO. Therefore we would go to the 8th number, for line 600. If a key other than one of the 6 listed is touched, INSTR returns a zero (for no match) and we are sent back to the first line number listed, namely 200.

* LINE 300 : This is the program line where we select the name of the file for the data. In other words, the file name will be the name of a person. When the name is entered, it is sent to subroutine 800. If it is longer than the 8 allowed for a file name, it is truncated to 8 letters. Variable L is used as a flag which is 0 unless the name has been shortened, in which case L is set to 1.

* LINE 310 : If flag L is set, meaning that the name has been cut, we say so.

* LINE 320 : A sequential file, with the name obtained at 300, and shortened if necessary, is opened with the "E", or Extend function which allows additional data to be added to the end of that file. If there is no file by that name, the situation is remedied at once, as one is opened. The date is displayed and any additional data obtained by LINEINPUT is placed in variable B$.

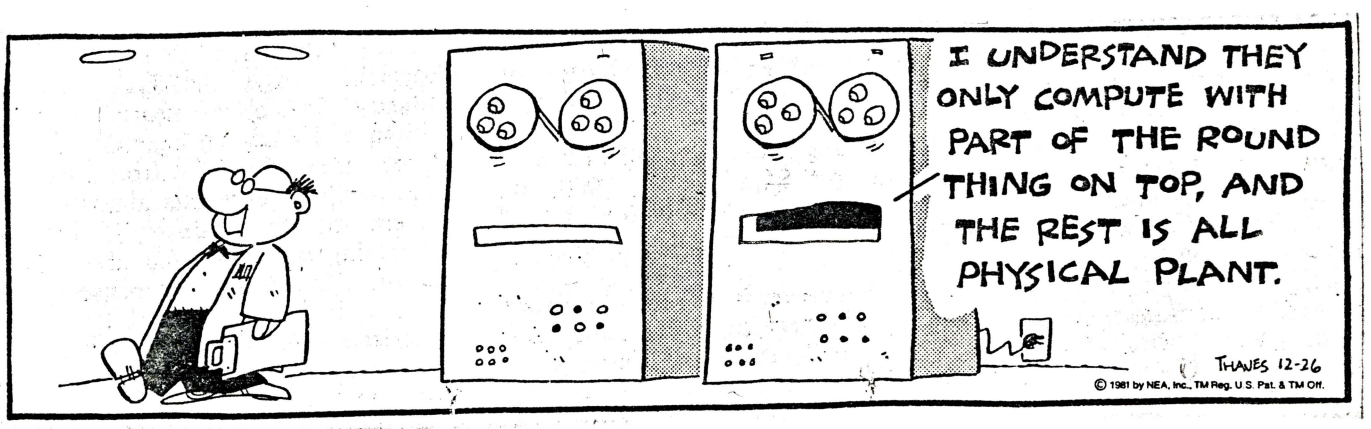
* LINE 330 : If B$ is blank, there is no need to enter anything, so we close the file and go back to the menu.

* LINE 340 : Here we store the date D$, followed by data B$, at the end of the file and close it.

* LINE 410 : To read the information from a file, we ask for the name, shorten it if needed and use the CMD feature again. This time we will use the DOS 'List filespec' command except that we will build it from Basic. We set up Z$ to be "LIST NAME" and execute CMD"Z$". We use a dummy INPUT to stop at the end of the command and to get us back to the menu. Note that if a name that is not on disk is entered, we have just

Voice of the '80 Newsletter September 1982 Vol 3 No 4 Page 17
{}{}{}

```
300 CLS : INPUT "Enter Last name"; N$ : GOSUB 800
310 IF L = 1 THEN PRINT : PRINT "I have changed --> "N$"'s <--
    name to ===> "A$
320 OPEN "E", 1, A$ : PRINT : PRINT "Enter data if any,
    else press <ENTER>" : PRINT : PRINT "===> "D$ " "; :
    LINEINPUT B$
330 IF B$ = "" THEN CLOSE : GOTO 200
340 PRINT# 1, D$; ", "; B$ : CLOSE : GOTO 200
400 CLS
410 N$="" : INPUT "Enter last name"; N$ : GOSUB 800 : B$ =
    "LIST " + A$ : CMD"B$" : INPUT "Push <ENTER>"; Z$ : GOTO
    200
500 CLS
510 INPUT "Enter Last name"; N$ : GOSUB 800 : B$ =
   "PRINT " + A$ : LPRINT A$ : LPRINT : CMD"B$" :GOTO 200
600 CMD"DIR 1" : INPUT "Press <ENTER>"; Z$ : GOTO 200
700 CLS : LINEINPUT "Enter date, any format including blank
    ?"; D$ : GOTO 200
800 L = 0 : IF LEN(N$) > 8 THEN A$ = LEFT$(N$, 8) : L = 1 :
    RETURN
810 A$ = N$ : RETURN
900 IF ERR/2 + 1 = 54 THEN PRINT : PRINT "This Name is not
    on file as entered
    Try again !" : PRINT : PRINT : IF M$ = " L " OR M$ =
    " 1 " THEN RESUME 410 ELSE RESUME 510
910 IF ERR/2 + 1 = 58 THEN PRINT "That is a weird date you
    fed me. Try again !!" : PRINT : PRINT : RESUME 30
999 CLS : END
```



Voice of the '80
Index for September

Using LMOFFSET ..... Page 1
Computer Widow Respite Page 7
MINUTES of August .... Page 9
PRINTER DRIVERS ..... Page 10
MINI-DATABASE ..... Page 15
Cartoon ..... Page 18
Cartoon ..... Page 6
Condensed Programs .. Page 8
ANNOUNCEMENTS ..... Page 10
PRT/CMD Source Code .. Page 12
MINIDB/BAS ..... Page 17

The Voice of the '80 is a publication of the Fairfield County TRS-80 Users Group. 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: Sid Gross VICE PRESIDENT: Pat Todd
SECRETARY: John Webb TREASURER : John Roy
EDITOR : Alan Abrahamson ASSOC. EDITOR : Dennis Palmiero
Telephone: Evenings: 203/866-2670 Days: 203/853-3861

This Newsletter diligently produced and printed by John Krause Jr.

FAIRFIELD COUNTY TRS-80 USERS GROUP
% Alan Abrahamson
10 Richlee Road
Norwalk, CT 06851

1st C L A S S M A I L

TO:

Come to the TRS-80 concert.