MICROPOLIS USERS GROUP

MUG Newsletter #22 - May 1982

****

MICROPOLIS BASIC BUGS

by George Shaw, Shaw Laboratories, Ltd.
17453 Via Valencia, San Lorenzo CA 94580

MUG members might want to be aware of a bug in the RENUM logic which deals with the CLEAR option in the OPEN statement. The problem seems to be that if the clear option is not the first specified, whichever option is specified immediately before the CLEAR will not be renumbered properly. For example:

```
100 OPEN 1 "TEST" CLEAR END 1000 ERROR 2000
```
will renumber, but in
```
100 OPEN 1 "TEST" END 1000 CLEAR ERROR 2000
```
the END 1000 will not renumber. Similarly,
```
100 OPEN 1 "TEST" END 1000 ERROR 2000 CLEAR
```
the ERROR 2000 will not renumber.

I don't really see this as a serious bug, but more of a nuisance if one forgets to specify the CLEAR first.

Another subtle bug I have found is related to the CHAIN statement. The problem deals with string variables and the SIZES statement when chaining. The bug is actually in the SIZES statement.

The SIZES statement is supposed to generate an error if any variables have been used prior to the SIZES statement. If the SIZES statement is used to set the maximum program size for chaining, and the program which contains the SIZES statement is smaller than the maximum, and a string variable was used before the sizes statement, then when the larger program is chained, the string variable will be overwritten by the program. This is because the maximum program size option on the SIZES statement determines where the data for the chained BASIC programs are placed. If a string variable is used before the maximum program size is set, then the string variable will point somewhere inside the program buffer (past the end of the current program but not past the end of the longest program). If this variable is to contain data to be passed to another program, the data the other program receives will not be as expected.

Another bug to be wary of is opening and closing files within FOR-NEXT loops. I am not positive about this one, but I recall writing a program to accumulate the data from several different job tracking files, each file with a sequential job number. The program was coded with a FOR-NEXT loop around the OPEN and CLOSE statements, with appropriate logic to find which job files still existed and to accumulate the data, etc. The program kept generating strange syntax or some sort of errors (it wouldn't run). I believe the error was "MISSING FOR." The program appeared correct (the FOR statement existed and was executed). I changed the FOR-NEXT loop to a loop with a counter and an IF statement and the program worked fine. My recommendation is to stay away from FOR-NEXT loops around OPEN and CLOSE statements.

I have some other information which may be useful for those wishing to increase the speed of MDOS. Not having a completely commented disassembly of MDOS at the time, I searched as best I could to supply this information for one of my A-FORTH customers. One of the things he was interested in was using interrupts in MDOS. I found the memory addresses which I believe to be locations in which to place enable interrupt instructions for the disk routines (they are currently NOPs). These locations are listed below (all addresses are in HEX):

```
    0A04      138B      13BF      1417      143B
```

(Continued in column 8)

BUILDING THE CHEAP COMPUTER, PART I

by Zot Trebor

I have an SSM cpu and video in my system. I bought them because they were cheap. I made them from kits. The kits didn't come from SSM; the cpu came from JADE and the video came from MIKOS. I might have that backwards. Since I've been programming in 8080 assembler language I get things backwards a lot ... or is it that I get a lot of things backwards?

The boards were purchased because of their attractive (cheap) price and the fact that I didn't have a lot of money at the time. I bought the cpu board first and watched old movies on the TV-cum-monitor until I had enough money for the video board. It was sort of fun. I wonder if I can
SAVE "N:1:RAWHIDE"
without a PARM ERROR?

The video monitor is a Hitachi television receiver with a converter by Pickles and Trout. That's really their names; Pickles and Trout. All together I think I paid a hundred bucks for the monitor and it still pulls in Tijuana on a clear cold night. Not exactly DX since I live near San Diego, but I'd like to see an ADM-3 get channel 12 the way I do.

The SSM cpu has space for two 2708's and the SSM people, sure enough, will sell you two 2708's to plug into the board. The ROM's contain the SSM version of an executive or monitor system. See, if you are old and have worked with big computers you call a monitor an excutive. If you're some punk kid who likes games, you call an executive a monitor and a monitor a video. I think it has something to do with the way Intel stores their addresses. I get confused.

The SSM monitor is a piece of #$%@. It expects the video to be addressed at B000, right where I like to store the baseball scores, and it still prints dashes on the screen when you backspace, as though it were a teletype (r). The backspace is important because it is the computer operator's eraser. If you make a mistake you can erase it without filling up the screen with a lot of dashes and back-slashes and sword cuts and stuff like that. The (r) is an expression of fear; if you say 'teletype' and fail to say '(r)' they take away your house and attach your paycheck. If I even hear someone say 'teletype' I put my fingers in my ears and run away. I hate them. To have a video that thinks it is one is anathema. Wormwood. Gall. And other funny words you have to look up in the dictionary.

For a long time I couldn't afford a keyboard. I tried talking to the cpu in Morse code. It was sort of fun. Strange things would happen. I got a QSL from a kid in Indiana and a missle from Vandenburg went off course and blew the #$%@ out of my tomato crop. I decided to get a keyboard.

I bought the cheapest keyboard I could find. The company paid me three dollars to take it. I built a case for it and found new keys for the missing ones and epoxied the cracks in the printer circuit and made hundreds of tiny holes to hold the hundreds of tiny wires that repaired the traces where they were broken.

I built a power supply for it and that was a lot of work as it turned out to be the keyboard from an organ and required compressed air at 5 psi and dry steam a -12 degrees Fahrenheit. After all that work it encoded everything in Bach. The SSM cpu (or maybe it's the video) has a keyboard input port. The port has a 5 vdc supply but no -12 vdc. I went back to my trusty Morse code key and worked a Poly living in an abandoned car near Santa Barabara. Gosh, I like computing!

About this time gold went to eight hundred dollars an ounce and I struck while the iron was hot. By

(Continued in column 9)

FURTHER FORTH

by Richard Newman
Faculty of Medicine, Memorial University
St. John's Newfoundland A1B 3V6 Canada

The FORTH I sent to the library is debugged now, for the most part. The EQU for C0 at the start of FORTH needs to be changed to KBBUF plus 14 from KBBUF+10. Line 446 in FORTHSOR3 needs to be changed from WARM to WARN.

The definition of ENCLOSE, which is on FORTHSOR1 starting at line 292, needs to be rewritten. This rewrite is from FORTH DIMENSIONS which is the publication of the FORTH INTEREST GROUP (FIG) and is therefore in the public domain. The author (William D. Miles) claims that this change is necessary when using Micropolis, which reads and writes 256 byte sectors.

```
            ENCLOSE
            DB      87H
            DB      'E','N','C','L','O','S'
            DB      'E'+80H
            DB      'PFIND-9
 ENCL       DW      $+2
            POP     D
            POP     H
            PUSH    H
            MOV     A,E
            LXI     D,-1
            DCX     H
 ENCL1      INX     H
            INX     D
            CPM     M
            JZ      ENCL1
            PUSH    D
            PUSH    PSW
            MOV     A,M
            ANA     A
            JNZ     ENCL2
            POP     PSW
            INX     D
            PUSH    D
            DCX     D
            PUSH    D
            JMP     NEXT
 ENCL2      POP     PSW
            INX     H
            INX     D
            CPM     M
            JZ      ENCL4
            PUSH    PSW
            MOV     A,M
            ANA     A
            JNZ     ENCL2
            POP     PSW
            PUSH    D
            PUSH    D
            JMP     NEXT
 ENCL4      PUSH    D
            INX     D
            PUSH    D
            JMP     NEXT
```

After entering in the above corrections and getting FORTH up you can install the EDITOR. This is a mini-editor as it does not contain other functions which are included in the full FIG editor. The problem with the FIG editor is that much of the editor requires cursor addressing and therefore must be customized for various terminals of memory mapped video. The full FIG editor is available in the 8080 model of FORTH (screens 87 to 97). The installation of the mini-editor is adapted from a Technote in FORTH DIMENSIONS vol. 3 #2 by Ted Shapin. First initialize a disk and put it aside. Next run FORTH and type in the following:

```
HEX : TEXT HERE C/L 1+ BLANKS WORD HERE
PAD C/L 1+ CMOVE ;

: LINE DUP FFF0 AND 17 ?ERRoROR SCR @
  (LINE) DROP ;

: -MOVE LINE C/L CMOVE UPDATE ;
```

(Continued in column 9)

POLY TO MICROPOLIS CONVERTER

by Marvin Konopick
2949 Hickory St., Alexandria VA 22305

Since purchasing my Micropolis, my problem has been trying to convert my old Polymorphic Assembler program listings to the Micropolis format.

The two systems are vastly apart in operation. The Poly uses an unnumbered line format. The spacing between columns consists of a single tab, whereas the Micropolis, as you know, has the detestful numbered line system and all of its encumberances. Be that as it may, the disk system prevails, regardles of format. During the upgrade from the Poly to the Micropolis, I discovered I had to remove interrupt jumper 'K' on the CPU board. I immediately brought this jumper out on two wires to a front panel switch. As a result I have two operating systems, and can switch from one to the other 'on the go' and enjoy the best of both worlds. This is extremely advantageous when I crash the disk, since I can salvage my RAM by dumping it to tape.

The procedure that I use to convert Assembler Listings from the Poly, is as follows:

1. Load the Poly Assembler
2. Load the Assembler Listing. Be sure to note where it begins (420E) and ends.
3. Use these locations to relocate the listing toward the top end of your RAM. Generally 920E is sufficient for a starting address.
4. Load the converter program shown below. I put this in the extreme top end of RAM.
5. Switch the interrupt switch from Poly to Micropolis and load MDOS/LINEEDIT.
6. After the system has come up in Lineedit, hit reset and modify memory lovations 3F11H to the locaation of the converter, E0 - DF (DFE0). Also enter the beginning of the relocated assembler listing in location DFF9 (POINTER).
7. Warm start the Lineedit at 4B00. Each line fo the Poly listing will appear whenever the spacebar is hit. I originally decided to put in a 7DH as the last character in the listing to be sure I never let it run too far, but that has not been a problem, since my finger tires before the end of the listing. The conversion will automatically remove the tabs, and insert a space in its place.
8. After completion of the transfer, hit reset and restore locations 3F11 to 7B - 27 and warm start again to location 4B00. The listing should be ready to be given the final polish before assembling.

I have had no problems with switching between the two systems, except for an occasional quirk when switching the interrupt switch from the Poly to Micropolis. could cause the system to blow. This happens rarely, but it has happened, so I counter this by hitting reset immediately when MDOS is loaded, and then come up on a warm start to preclude blowing my converted text from RAM, before I have had a chance to SAVE it.

CONVERTOR

This program loads a Poly Assembler Program into Micropolis LINEEDIT format.

The program is located at the end of my RAM at location DFE0. DFE0 must be put into MICROPOLIS Subroutine CILINE to replace the call at 3F10 (1F10). Warmstart the LINEEDIT program and spacebar each line in.

```
DFE0 E5             PUSH  H
DFE1 2AF9DF         LHLD  POINTER ;beginning of the
file.
DFE4 B7             ORA   A
DFE5 7E             MOV   A,M
DFE6 47             MOV   B,A
DFE7 23             INX   H
DFE8 22F9DF         SHLD  POINTER
```

(Continued in column 6)

FORMATTED INPUT ROUTINE

by Jerry Factor
709 No. Palm Drive, Beverly Hills CA 90210

The August, 1981, MUG letter included a routine which provides a winking asterick to prompt entry of a single character. Here's a routine to prompt input of a specific field size, and edit (to a degree) the characters accepted.

The subroutine begins at instruction 600. It expects to be passed I$, which includes two positions for field length, two positions for an ASCII range of acceptable characters, and then any number of other acceptable characters. After receiving a RETURN, the subroutine passes the entered characters back as I$.

You must add your own routines to eliminate more than one decimal point in a number field, conform a field to a required length, reject a month greater than 12, and so on.

For quicker operation, put lines 640, 645, and 650 all on the same line. Crunch it for even more operating speed.

Because RETURN is required to signal a completed entry, I prefer to use the August, 1981, approach for single character entry.

```
005 ! DEMONSTRATION -- FORMATTED INPUT SUBROUTINE
010 !
015 DEF FNC=16RF800+256*PEEK(78)+PEEK(77)
015 ! FNC - Cursor Location
020 A=ASC("*"): B=ASC(" "): U=ASC("_"): I%=79
021 ! I%=Input Byte
025 !
030 T=50: PRINT CHAR$(6):! Clear Screen
035 PRINT "FORMATTED INPUT DEMONSTRATION"
040 !
045 PRINT: PRINT "ENTER Phone No:  ";
050 I$="1009": GOSUB 600: IF I$="" GOTO 155
051 ! Formated Input Subroutine
055 IF LEN(I$)<10 I$=" "+I$: GOTO 55: ! Conform Inp
    ut Length
060 PRINT TAB(T) "("; LEFT$(I$,3);")";MID$(I$,4,3);
    "-";MID$(I$,7)
065 !
070 PRINT: PRINT "ENTER Name:   ";
075 I$=20Az., ": GOSUB 600: IF I$="" GOTO 155
080 PRINT TAB(T) I$
085 !
090 PRINT:PRINT"ENTER Customer Type (ASDF, 1-5): ";
095 I$="0215ASDF": GOSUB 600: IF I$="" GOTO 155
100 !
105 PRINT: PRINT
110 PRINT "ENTER Date (mmdd):  ";
115 I$="0409": GOSUB 600: IF I$="" GOTO 155
120 IF LEN(I$)<4 I$="0"+I$: GOTO 120
125 IF VAL(RIGHT$(I$,2))<1 I$="X": ! Parse Date
130 IF VAL(RIGHT$(I$,2)>31 I$="X"
135 IF VAL(LEFT$(I$,2))< 1 OR VAL(LEFT$(I$,2))> 12
    I$="X"
140 IF I$="X" PRINT CHAR$(13); CHAR$(21): GOTO 110
141 ! Invalid Date
145 PRINT TAB(T) LEFT$(I$,2); "-"; MID$(I$,3)
150 !
155 PRINT: PRINT: PRINT "THIS WAS A FORMATTED INPUT
    DEMONSTRATION"
160 PRINT: PRINT "RETURN=REPEAT: ";
165 I$="00": GOSUB 600: GOTO 30
170 !
175 !
600 ! FORMATED INPUT - I$=LLRRCCC
605 L=VAL(LEFT$(I$,2)): ! L=Allowed Input Length
610 Z$="": ! Z$=Input String
615 C=FNC: ! C=1st Location
620 PRINT REPEAT$("_",L);:POKE(FNC)=B: ! Show Space
    s & Blank Cursor
625 FOR P=0 TO L: ! P=Position Counter
630 W=U: IF P=L W=A: ! W=Wink Character
635 POKE(I%)=0: ! Clear Input Byte
640 POKE(C+P)=W: ! Winking Subroutine
645 FOR Z=1 TO 20: IF PEEK(I%)=0 NEXT Z
650 POKE(C+P)=B:FOR Z=1 TO 10:IF PEEK(I%)=0 NEXT Z:
    GOTO 640
655 Q=PEEK(I%): ! Parse Q (ASCII Of Input)
```

```
660 IF Q=13 I$=Z$: RETURN: ! RTN?
665 IF Q=8 OR Q=127 GOTO 705: ! DEL Or BSP?
670 IF P=L GOTO 635:! Only BSP or RTN In Last Pos'n
675 Q$=CHAR$(Q): ! Parse Q$ (Input Char)
680 IF INDEX(MID$(I$,5),Q$)>0 GOTO690:! Named Char?
685 IF Q$<MID$(I$,3,1) OR Q$>MID$(I$,4,1) GOTO 635
686 ! Char In Range?
690 POKE(P+C)=Q: ! Display New Char
695 Z$=Z$+Q$: ! Append New Char
700 NEXT P: END
705 IF P=0 GOTO 635: ! Parse BSP (No BSP Past 0)
710 IF P=L W=B: ! * In Last Position
715 POKE(C+P)=W: ! Erase Display Pos'n
720 P=P-2: ! Reduce Input Pos'n Count
725 Z$=LEFT$(Z$,LEN(Z$)-1): ! Delete Char From In-$
730 NEXT P: END
```

....

POLY - (Continued from column 4)

```
DFEB FE7D           CPI    7DH  ;don't forget to put
a 7D at the end of file to shut it off.
DFED CA0000         JZ     0000H
DFF0 FE09           CPI    09H  ;convert all tabs to
spaces for the Micropolis.
DFF2 C2F7DF         JNZ    SPACE
DFF5 0620           MVI    B,20H
DFF7 E1     SPACE   POP    H
DFF8 C9             RET
DFF9 0000   POINTER DB     00H,00H ;Put the
beginning of the file here.
DFFB 20202020       DB
```

....

BASIC/Z & BASIC/S

by Buzz Rudow

Don't take the lack of discussion of BASIC/Z as an indication that there's something wrong. There isn't, and it's beautiful. My work load has been tremendous, however, and I just haven't formalized any reviews. Conversion of my programs to run on the new Blackhawk (story below) and its hard disk has been a major task. I've had very few problems, and a great deal of fun. The resultant programs are very fast.

While BASIC/Z & S have almost the same syntax as Micropolis Basic, the capability for sophisticating one's code is huge.

What I mean is, you can transfer your current programs with little effort and they can look just like they did (almost) in Micropolis Basic. Once you get there, you can start putting in all the goodies - like LABELs for GOTOs and GOSUBs, e.g., GOSUB @READDISK, expansion of variable names to be meaningful, formatted input, IF-THEN-ELSE and WHILE statements. I could go on and on, and I will, over the next months.

The point is, it works well and I have no difficulty in recommending BASIC/Z & S to you.

These programs require a signed license to be in my hands before they can be shipped. So, if you're going to buy, write or call me so I can send you a license to sign. If you're curious and aren't on System/z's mailing list, write me for a copy of the BASIC/Z or S brochure.

BASIC/S should be re-released by mid-May. Then the pure MDOSers will have the same powerful capability. The neat thing is, if you ever decide to run in CP/M, you can move your MDOS-BASIC/S software right over. You'll have to buy RUN/Z to execute them, but you won't have to reprogram everything like you do when you translate Micropolis Basic to BASIC-80 or C-BASIC.

....

BLACKHAWK COMPUTERS

by Buzz Rudow

I've acquired another new computer, a Blackhawk. This particular unit has one Micropolis MOD II drive and a Seagate 5Mb hard disk. It is not sold with MDOS, though the floppy controller is Micropolis'. Blackhawk has the right to use MDOS, so I'm trying to get them to configure a system for me. I tried a little bit on mine. I got MDOS up, could do a FILES, but couldn't load BASIC.

The Blackhawk does run CP/M, of course. I've learned a great deal about the operation of CP/M over the last couple months, though I know little about the operating system itself.

This article is to acquaint you with the hardware, though, so let's forget about operating systems for the moment. Using an S-100 bus, 64K of memory, and a 4Mh Z-80, the Blackhawk is similar to the latest Vector Graphic computers. The screen, drives, and all electronics are built into a single housing.

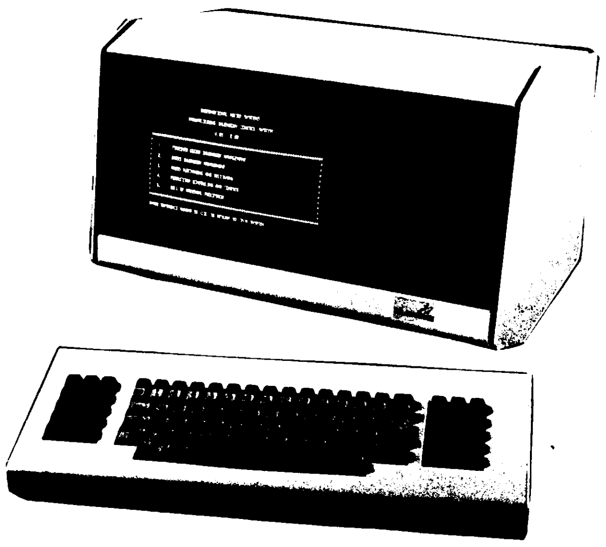
The keyboard is detached and has a separate 15-key pad for cursor control on the left, a 14-key numeric pad on the right, and a row of 14 user-definable keys on the top. This movable keyboard allows the operator to place the keyboard wherever he likes for comfort and convienience. The 24 x 80 display is put on a 12-inch screen.

A 2K ROM contains the monitor routines. An optional programable character generator is available, should you wish to create your own special characters. Sounds like you can make "Invaiders" graphics, etc., though I haven't tried it yet.

There are two versions of the hardware. The Blackhawk III has two Micropolis floppies, the Blackhawk IV either substitutes or adds the 5Mb hard disk. An optional 26Mb hard disk is available for both models. MP/M and CP/NET are also available, allowing a multiuser, multiprocessing system.

I'm a dealer of this hardware, so if you are looking for an additional, compatible, computer to supplement your current Micropolis system - or are looking for an expanded system with hard disk and multiprocessing, give me a ring or drop me a line. I'll send you a brochure and more information.

....



S/W PRICE CHANGES

by Buzz Rudow

Each software vendor seems to have a price structure of his own. System/z's price structure rewards the dealer for accumulated sales. This dealer (me) is therefore also rewarding his customers (you) with a price decrease. To complicate things, however, I've re-thought my pricing scheme after finishing the first three months of operation.

DAMAN will now have a normal price, which will be lower than list in many cases, and will also give MUG member a 5% discount from that. Now, I've heard all the "bad-talk" about 5% discounts - "meaningless, an insult, etc.". Never-the-less, there is no reason for you to pay for the magazine ads. I can get to you guys & gals monthly through the newsletter. The second change is that I will charge for shipping - $5 for the first module, $2.50 for additional modules in the same order. Exceptions to the rule are System/z, Bonjoel and GMS software. These people drop-ship for me.

Buyers outside North America will have to add an additional $7 per module, assuming you want it air-mail.

The new MUG prices for System/z software are:

MDOS

		LIST	MUG
AUTO/EXEC	System Generator	$ 40	34
BASIC/S	Extended BASIC Compiler	345	278
BCMP	Basic Comparison	35	30
BEM	Basic Expansion Module	65	54
CRUNCH	Basic Compactor	35	30
DSM-1	8080/8085 Disk Disassembler	65	54
EDIT/S	Text Editor	45	39
RUN/S	Run-Time Package only	65	54
SORT/A	Hybrid Sort (for Microp. BASIC)	75	63
TR/II	Translator II - BASIC/ASCII	55	47
UTL-1	Disk Utility Package	95	79
XREF	Cross Reference Generator	85	70

CP/M

BASIC/Z	Extented BASIC Compiler	$345	278
RUN/Z	Run-Time Package only	65	54
SORT/B	Hybrid Sort (for BASIC-80)	75	63
UNDELETE	File Recovery	45	39
UNPROTECT	Basic Source Recovery	70	59

Again, these prices include shipping in North America. All System/z software is available in MOD I or II. The MDOS software requires Ver. 4.0; the CP/M software requires Ver. 2.x, and is also available on IBM 8" disks.

....

BUGS -- (Continued from column 1)

Also, at 13F9 there is C3 C0 13 (JMP 13C0H) which should be changed to C3 BF 13 (JMP 13BFH) to ensure interrupts are reenabled.

He also inquired about disabling the retries. There are three pairs of MVI A,xx STA xxxx instructions which set the retry counters for each level of error. The values of xx are currently 3, 4, and 5. These values are at (HEX) 0D82, 0D87, and 0D8C respetively. Changing these values will change the number of retries at each level. The code structure here is the same as the example disk driver code in the disk theory section of the MDOS 4.0 manual.

Another item of interest was the 1 millisecond timer. This is located at 1336 hex, appears to be only entered from this point, and is as follows:

```
1336    PUSH B
        PUSH H
        LHLD 0882H ; 882 contains the addr of the
                   ; controller
```

```
         MOV A,M ; reading status restarts 4 sec
                 ; timer
         ANI 20H ; also allows checking of speed
                 ; jumper W9
         MVI A,60H
         JNZ 1344H
         RLC     ; If jumper, double for 4 mhz
1344     MOVE B,A
1345     MOVE A,B
1346     SUI 1
         ORA A
         JNZ 1346H ; time out 1 ms
         DCX D
         MOV A,E
         ORA D
         JNZ 1345 ; count 1 ms periods
         POP H
         POP B
         RET
```

The above code could be changed to use a hardware timer (possibly using interrupts) or just trimmed for the clock speed on your system.

Also, the memory move routines in MDOS and RES are at the following locations (HEX):

```
    0FCF    0FD8    1251    1B4C    1B55    1B60
```

You may wish to modify some or all of these to Z80 code. There are also a couple of search routines which you may wish to modify also. They are located at (HEX) 0F6A and 2491.

I've played with all of the above items except changing some of the code to Z80 code. If anyone tries this, please let me know. I hope this information is of use.

....

BUILDING -- (continued from column 2)

selling the frames for my glasses I accomplished two things; I became the owner of a Jameco key-board and the world's largest pair of contact lenses.

The Jameco keyboard is a darb. It works. Few things do. It also wants a -12 vdc supply and to be kept clean and dry but what the hey! I fumbled up a -12 vdc supply that works whenever I am home. If I want to go anywhere I have to put the battery back in the truck, but as I can't drive and type at the same time I figured it was a good deal. Now I could talk to the SSM cpu and glare each time I saw those filthy, unmentionable dashes instead of nice neat back-spaces.

Wow! My own cheap computer!

The joys of getting a SYNTAX ERROR that reallys MEANS something. The wonders of a FILE OVERFLOW... I'm sorry, I just get all choked up when I think of it... and when the power supply starts to smoke like this. I'll have to try and finish this next time, when I'll tell you the secret of how to make a bell-less video have its own bell; where Screen Home really lives, and what to do if you catch Control X in your bedroom.

....

FURTHER FORTH (Continued from column 3)

```
      : P 1 TEXT PAD 1+ SWAP -MOVE ;

      DECIMAL
```

Where HEX changes the base to hexidecimal. TEXT is a definition which, when executed, moves text in the input buffer to the dictionary for storage. LINE checks for out of range linenumbers (0 to 15 only acceptable) and gets the address of the disk buffer where the text for that linenumber should go. MOVE actually does the moving of text to the disk buffer and marks the buffer as updated so that it will be written to the disk. P, when preceded by a linenumber, places the text in the disk buffer.

Make sure that when you type this in you leave spaces between the words and characters as shown. FORTH is unlike BASIC in that the language is extensible and therefore definitions are not fixed words so that spaces are necessary to delimit words. Now remove the disk in drive 0 and install the initialized disk. Now if you type 85 LIST, FORTH should list the contents of screen # 85 which, if the disk has been unused before, should contain nothing. Now you can put the mini-editor on disk as in the following example:

```
0 P   (MINI-EDITOR)
1 P  HEX : TEXT HERE C/L ..........
.
.
6 P : P 1 TEXT PAD 1+ SWAP -MOVE ;
7 P ;S
```

Now type FLUSH, which will save your mini-editor on the disk in drive 0. Now whenever you load FORTH all you need to call the EDITOR is to type 85 LOAD. Before using the editor make sure you type DECIMAL to insure that the base is decimal. To edit a screen you must list the screen first by typing the screen # and the LIST. The " ;S " on line 7 is to tell FORTH that it has reached the end of the program to load. Now using your mini-editor you can type in screens 4 and 5, which contain the error messages. Just type 4 LIST and then type in the messages as they appear. Then type 5 LIST and the messages for that screen. After you type in those screens, type FLUSH, which will write them to the disk. If, after you load and use the editor, you want to regain the RAM space it occupies, you simply type FORGET TEXT which will forget everything in the dictionary since TEXT was defined. Typing 85 LOAD of course will recreate the editor.

After loading screens 4 and 5 with error messages, the value of warning should be changed from 0 to 1. This tells FORTH that a disc is present that has error messages in screens 4 and 5. You can do this by changing the 0 in line 66 of FORTHSOR1 to a 1 and then assembling FORTH or, while in FORTH, you can type

```
             HEX 1 1A +ORIGIN !
then type
             HERE .
```

This will tell you the highest point in Forth that need be saved. Then type BYE to get back to MDOS and save FORTH from 2B00 to the value indicated by HERE.

The parameters which are machine dependent are all at the beginning of the source (FORTHSOR1) and are as follows:

```
         ABL=SPACE
         ACR=CARRIAGE RETURN
         ADOT=PERIOD
         BELL=CONTROL G
         BSIN=BACKSPACE CHR (RUBOUT)
         BSOUT=BACKSPACE OUT (CONTROL H)
         LF=LINEFEED
         FF=FORMFEED
         EM=TOP OF MEMORY + 1
         MSCR=NUMBER OF 1024 BYTE SCREENS
         US=USER VARIABLE SPACE
         RTS=RETURN STACK AND TERMINAL BUF SPACE
```

I think I have included everything necessary now to get the 8080 FIG-FORTH model running. Have fun! If I have left anything out or if you find any more bugs let me know.

```
4 LIST
SCR # 4
  0 (ERROR MESSAGES)
  1 EMPTY STACK
  2 DICTIONARY FULL
  3 HAS INCORRECT ADRESS MODE
  4 ISN'T UNIQUE
  5
  6 DISK RANGE ?
  7 FULL STACK
  8 DISC ERROR !
  9
 10
 11
 12
 13
 14
 15 FORTH INTEREST GROUP
OK
5 LIST
SCR # 5
  0 (ERROR MESSAGES)
  1 COMPILATION ONLY, USE IN DEFINITION
  2 EXECUTION ONLY
  3 CONDITIONALS NOT PAIRED
  4 DEFINITION NOT FINISHED
  5 IN PROTECTED DICTIONARY
  6 USE ONLY WHEN LOADING
  7 OFF CURRENT EDITING SCREEN
  8 DECLARE VOCABULARY
  9
 10
 11
 12
 13
 14
 15
OK
```

....

DISK BANKING

Reviewed by Buzz Rudow

DISK BANKING, written by Jerry Lenz, is a set of fourteen programs and three data files which document, track, and allocate banking activities. The system is useful for business or home. It will accept input as shown in your check book and with the use of codes, will sort out the entries by these codes for later uses. It is written in Micropolis BASIC so the user may dump it to study its operation. You can then change it to perform new tasks or to modify its current operation.

Except for the HELP program, the system will easily run in 32K. I'm sure Jerry can break HELP into two modules for those of you with 32K memory. Simply compacting the file might do it.

The program is executed by typing PLOADG"MENU". The first option of the twelve selections on the menu is the HELP file. The menu instructions tell the user to execute this option before running the program any further.

Execution of HELP produces another menu of 14 options. Each option explains some aspect of the system. Selecting item 0 gets you 'General Information'.

Understand that all this is on your screen. You don't have to go hunt up a manual. Any question you have on operation is answered in the HELP program. However, everything in HELP is printable, if you want a hardcopy manual.

The 'General Information' section tells you that before using the system, you must do three things.

OPEN the Data File
Accept or Change the Codes & Titles
Enter a Starting Balance

Opening the data file and entering the starting balance consists of answering the following questions:

1) Are you running a one or two disk system?
2) What's the starting balance?
3) What's the date for the starting balance?
4) What's your console's clear-screen command?
5) What's your printer's form-feed command?

This data is stored on a file and accessed by the rest of the programs. Disk Banking is therefore "self-configuring", a real nice touch.

The codes and their descriptions are specified in DATA statements. As supplied, the data is:

1	Deposits - Receipts
10	Licenses
20	Returns & Allowances
30	Materials
40	Advertising
50	Bank Charges
60	Truck Expenses
61	Gasoline (gallons)
70	Dues & Publications
80	Employee Benefit Programs
90	Insurance
100	Interest
110	Legal & Professional
120	Office Supplies
122	Special Case
130	Business Taxes
140	Postage
150	Rents
160	Repairs
170	Sales Tax Collected
180	Telephone
190	Travel/Entertainment
200	Utilities
210	Salaries/Wages
220	Shop Expenses
230	Other Expenses
240	Proprietors Account
250	Contributions

You can delete, add, or modify any of these by editing the program. Codes may be any 4-digit or less number, descriptors any 30 character or less of text. A couple other unique programming touches are found here. One menu option in the program lists the data section of the program and stops, allowing you to edit. Then, after typing RUN, another option in the menu SAVEs the reconfigured, presently executing, program & data to disk, and goes along its merry way. Whatever you save here doesn't have to be your final thoughts on the subject. New codes may be added at a later time, or unused codes deleted.

Now you're ready to do what we started out to do - track banking activities. We've easily gone through the first 4 of the 14 menu choices.

The program prompts you for required data - 'Type of Transaction', 'Check #', 'Date', 'Issued To', 'Code', and 'Amount'.

You are first asked whether the entry is to be a Deposit, a Withdrawal or an Other. This must be answered by D, W, or X. When a D is entered, the amount entered will show as being deposited to the account. W will withdraw the amount from the account. The X will not affect the bank balance.

The X is used for entries that are neither Deposits nor Withdrawals to the bank account such as items that are paid in cash or items that you want to record. One use of the X could be for keeping a record of all the gasoline you use by using code 61, called 'Gallons gas', and entering the gallons purchased to the Amount prompt.

Code 210 is set-up for Salaries & Wages. When this code is used, another set of prompts will appear, asking for withholding information.

Code 122 is a Special Case Code which is used when checks are written against an account for another client. When this code is used, a prompt will request a second entry which will be added to the name field with a / separating the entries. These two entries will later be separated and properly shown.

The 'Check' prompt will hold up to 5 characters, which can be alpha or numeric.

The 'Issued to' prompt will accept up to 27 characters which can be alpha or numeric. A guide will show the length of the input.

All unspecified deposits should be entered as code #1. Any deposits specified by another code will be added to the bank balance and reflected as a credit to the code specified. Only 2 characters, the day, are needed for the 'Date' prompt. Month and Year have previously been entered.

The 'Code' prompt will allow up to 4 numeric characters. Deposits will automatically enter a numeric 1 in the Code column so all deposits can later be audited correctly.

It is not necessary to enter 00 behind the decimal point if there are no cents in the 'Amount', i.e., $21.00 can be entered as simply 21. Each File entry, when completed, will show the new Bank Balance.

To correct errors, the record number of the entry to be corrected must be used to tell the computer where the data is stored. The record number can be obtained from one of two outputs, either of which can be directed to the console or the printer. When data correction is requested, each field of the record will show sequentially on the screen. Only the erroneous data need be changed. If you are satisfied with the data shown, you press the RETURN key and by-pass the 'good' data.

If you change the monetary amount in any record, you must re-balance the records. This is simply another menu selection.

A menu selection is also provided to show the lastest bank balance. Actually, it shows all of the last data record.

The Quick Look menu option shows the data in the file in compressed form. One can breeze through the files to see what they contain. This form is much faster in showing data than the Sequential Listing as it does not take the time to re-arrange the Data.

The Sequential Listing option prints a formatted listing with the running bank balance for each entry. It also prints a total for all the codes entered.

The Audit Trail option prints a 'sorted' listing of each code's activity. This is the power of the program, or course. Helping balance the checkbook is nice, but what one really wants to know is "how much did I spend on business expenses, gasoline, medical, cloths, or whatever, and which of my checks document those purchases". Very handy at income tax time, very handy to spot trends, to evaluate the use of income, and for planning for future months.

This revision of Disk Banking is considerably improved over the one I used all last year with DAMAN. It uses pointer files to speed the audit trail and has better error trapping. It still has the problem of not being able to insert entries you forgot, but this is easily solved by leaving 4 or 5 blank entries at the end of each month. I 'correct' those with the forgotten data. It also has the 'defect' of not having multiple asset categories. If you have multiple bank accounts, you must either treat them as one, or run multiple files with this program.

Overall, though, the program is well organized, fairly fast, accurate - as far as I can see - and resonably priced. You can either get this from Jerry Lenz (3231 Vineyard Ave., #42, Pleasanton CA 94566 - phone (415) 846-8406) for $75, or you can get it through the MUG for $50 (plus $5 shipping).

Disk Banking isn't a match for DATASMITH's BOOKKEEPING, which has a full set of asset accounts and generates profit & loss statements, etc. But $50 isn't a match for $189, either. If your needs are only for checkbook balancing and expense allocation, or if you don't mind pulling out the totals and doing the profit & loss manually (as I did with DAMAN last year), Disk Banking might be just the thing you need.

....

MICROPOLIS PRODUCTS

I have, in stock, the following Micropolis products:

MDOS Version 4, including BASIC, utilities, and doc-umentaion: $71
Saunders Magnalube lubricant, tube: $5
Head Load Pads, pair: $1

All prices postpaid to North America. Add $7 MDOS, $1 for lub and pads, elsewhere.

....

NEW SOFTWARE

There's a bunch of new software in stock. At some time I'll have to review each of them to see if they are as good as they potentially might be. In each case, I assumed they had a use that wasn't filled by any software I'd personally seen previously. All of the following are CP/M. All prices are postpaid (North America), and include the MUG 5% discount. Elsewhere, add $7.

APPLICATIONS	
NEVADA COBAL	$186
COBAL Applications Book	23
NEVADA PILOT	141
NEVADA EDIT	114
UTILITIES	
DISK DOCTOR	93
DIAGNOSTIC II	93
FILE FIX	84
CATALOG	66
GAMES	
ANALIZA II.(Req. CBASIC2)	47
DUNGEON MASTER	33
NEMESIS	38

ANALIZA II is the "phychiatrist - Eliza" game. The other two are dungeons and dragons type of thing.

....

ACROPOLIS UTILITIES

There have been a lot of questions about what is in the Acropolis Utility Package. They are standard MDOS utilities, not FORTH. The package contains:

RECOVER	For recovering bad sector.
VERIFY	For verifying a disk.
TOKENIZE	For converting any ASCII text file into compressed Micropolis BASIC format.
COMPRESS	For removing spaces and remarks from BASIC programs.
MTEST	For testing memory.
ASCII	For dumping memory in hex and ASCII.
$ASCII	As previous, but does not load into the applications area.
VIEW	Allows the viewing and/or listing on the printer of BASIC programs, data of LINE-EDIT files, with the option to send to disk (this allows the conversion of compressed BASIC program files to ASCII text files).

These programs are available from the MUG for $45.

....

EXPERIMENTORS

Some of the members are experimenting with "add-on" equipment. It would be great if you inform me of your successes & failures with the various products.

The following members would like to talk with people doing the same or similar tasks. I imagine they'll be happy to speak with those of you who are just interested, also.

Houston Instrument's HI PAD digitizing pad and DMP-4 plotter - Stephen Maegerlein, P.O. Box 60, Williams IN 47470, (813) 388-7293.

Digital Research Computers' sound computer board (GI AY3-8910 chips), specifically BASIC routines to drive it - Marlin Weston, 4060 South Garfield, Loveland CO 80537 (303) 669-2054.

....

CLASSIFIED

WANTED: Information on a version of MDOS that has been modified by a company called RecTec. Running it on a Motorola Exordisk III.

Also would appreciate help in building an 8085 based computer from scratch, using DMA and either a WD1771 or a NEC765 floppy controller and HM6116-4 CMOS RAM.

Richard Snell, Sky-Hi Lanes, P.O. Box 3058, Flagstaff Arizona 86003, Phone (602) 774-3511.

....

WANTED: Any information on transferring files between Micropolis MOD II and 8" Morrow DISCUS 2+2. In particular, can a Micropolis add-on be directly attached and controlled by the Morrow DJ/DMA controller?

Vasili Soultoukis, 11883 A Academy Rd., Philadelphia PA 19154, MNET: 70675,1341

....

FOR SALE: CDS Versatile 4.

Paul Thompson, R.D. 2, Philhower Rd., Lebanon NJ 08833

....

WANTED: Any information about adding a numeric keypad (in addition to a current ADM-3A terminal) to the Micropolis system.

Mauricio Gluck, 4510 Pinetree Drive, Miami FL 33140

....

WANTED: Some help in 4 areas:
(1) Has anyone heard of an AMWAY program on any computer system?
(2) Any of the members have knowledge of, or experience with, a dental package? Any of the vendors created a customized system that I might use?
(3) Anyone have a solution for the following problem: Equipment is Explorer 85 (Neutronics), Jaws memory. MDOS seems to work OK. Trying to load BASIC causes the Prompt, the 32 "*"s, then lockup. Disk keeps turning, and knocking. A reset and 4E7 causes "*"s to reprint (usually 32, sometimes 6). System still locked, though disk doesn't turn. Some suspicion of the "hang" being around 1702H.
(4) Anyone have a patch for BASIC so the "wrap" is recognized in the LISTP directive. I want pagination to stay corrent, no matter what length my program lines.

Buzz Rudow, 604 Springwood Cir., Huntsville AL 35803 (205) 883-2621.

....

Published Monthly by the MUG
Subscription rates:
U.S., Canada, Mexico; $18/year: Other, $25/year

FIRST CLASS MAIL
===== ===== ====

FIRST CLASS MAIL
===== ===== ====

==

MICROPOLIS USERS GROUP

Buzz Rudow, Editor
604 Springwood Circle
Huntsville AL 35803
(205) 883-2621

HUNTSVILLE
MAY-4'82
ALA.
U.S. POSTAGE
.020
P.B. METER
1543328

FIRST CLASS MAIL
===== ===== ====