$1.50

dr. dobb's journal of

COMPUTER Calisthenics & Orthodontia

Running Light Without Overbyte

April, 1976 | Box 310, Menlo Park CA 94025 | Volume 1, Number 4

A REFERENCE JOURNAL FOR USERS OF HOME COMPUTERS

In This Issue. . .

MINOL—Tiny BASIC with Strings in 1.75K Bytes

AN OUTSTANDING JOB DONE BY A HIGH SCHOOL JUNIOR

Dear Mr. Warren: **May 1, 1976**

I have a Tiny BASIC program running on my Altair that I think you might be interested in. I call it MINOL (mine-all). It fits in 1.75K memory. Unlike the other Tiny BASIC's, MINOL has a string-handling capability, but only single-byte, integer arithmetic and left-to-right expression evaluation.

Additions to TB include CALL machine-language subroutines, multiple statements on a line (like TBX), and optional "LET" in variable assignments. Memory locations of the form (H,L) can be used interchangably with variables, permitting DIM-like operations.

Sincerely,

Erik T. Mueller **36 Homestead Lane**
Roosevelt NJ 08555

MINOL is an abbreviated form of BASIC with additional features. It has twelve statements: LET, PR, IN, GOTO, IF, CALL, END, NEW, RUN, CLEAR, LIST, and OS.

Variables: A letter from A to Z, or a memory location of the form (H,L), where H is the high address (decimal), and L is the low address. H and L may be expressions.

Number: An integer from 0 to 255.

Expression: A series of terms separated by arithmetic operators.

Terms: Numbers, variables, schars, random.

Schar: A single character enclosed in single quotes. Gives the ASCII value of the character.

Random: "!" (exclamation point) gives a random number between 0 and 255. (Subroutine by Jim Parker.)

Arithmetic Operators: + − * /

Relational Operators (not permitted in expressions): = # < ("less than")

Arithmetic Evaluation: All expressions are evaluated from left to right (no precedence of operations).

Statements: A statement consists of one or more substatements separated by ":" (colon), and terminated by CR. Lines up to 72 characters. Line numbers from 1 to 254. All statements may be used with or without a line number. Statements without a line number are executed immediately. Statements with line numbers are edited into the existing program.

Substatements: **[LET |ϕ] <var> = <expr>** Assigns the value of a variable. The "LET" can be left out if desired.

Ex: LET S = 0
LET (24,0) = P−59
A=B+C*J−198
(25,5)=A*7/B

PR <var-list> [; |ϕ]

<var-list> : Literals, strings, or expressions separated by commas.

Literal: Characters to be printed enclosed in double quotes.

Strings: $(H,L): A series of memory locations starting at H,L which contain characters previously entered.

Expressions: Simple variable or expression.

Ex: PR"YOU SAY YOUR NAME IS",$(10,0)
PRA,B,(6,0),
PR 56+!/A,B
PR

A semicolon at the end of a PR suppresses CRLF. A blank PR produces a CRLF.

PR Format: Numerical values are printed with one leading and trailing space and with all leading zeros suppressed. All strings and literals are printed without leading and trailing spaces. No zone spacing.

GOTO <expr>

Transfers control to the specified statement. GOTO 0 transfers control to beginning of unnumbered statement.

Ex: GOTO A*10
GOTO 78

IF <expr> <relop> <expr>;<statement>

Executes the statement following the ";" (semi-colon) if the specified relation is true. If it is untrue, control is transferred to the next statement on the line (if present).

Ex: IF X=5 ; GOTO 20
IF A='Y' ;PR"SURE, WHY NOT?"
IF A+B*C # !;GOTO 20 : PRA+B*C
IF Y # 6; S=!

IN [<var>|<str>] [,[<var>|<str>]]*

This statement permits two types of data to be entered from the terminal: a) Numeric data; and
b) Alphanumeric data; either a single letter, or a string of n characters.

Using a <var>: The input data is tested. If it is numeric, the number is deposited into the variable. If the data is not a number, the ASCII value of the first character typed is deposited.

Using a <str>: (of the form $(H,L) The inputted characters are deposited into memory sequentially starting at location H,L. 255 is placed in memory after the last character before CR. All spaces inputted are ignored unless enclosed by quotes. Note that (H,L) refers to a single location, but $(H,L) refers to a series of locations beginning at H,L. (H,L) can be used in expressions as a variable, but $(H,L) can only be used in I/O statements (IN, PR).

CALL (H,L)

Calls users subroutine starting at location H,L decimal.

END: Terminates program.

NEW: Deletes all lines of a program.

CLEAR: Sets all variables (A-Z) equal to zero.

RUN: Starts execution of program at lowest numbered statement.

LIST: Lists program in memory.

OS: Transfers control to user's operating system.

Line editing and correction:

Typing X^S deletes the last character typed.
X^L deletes an entire line.
X^C stops executing program.
Prints: BREAK AT LL (LL is the line that was to be executed before the interrupt occurred.)

To delete a line, type the line number followed by CR.

To change a line, type in the line with changes. The new line will replace the old one.

ERROR MESSAGES !ERR L AT XX

1. Label does not exist

2. Input is over 72 characters.
3. Unrecognizable statement type.
4. Illegal variable.
5. Syntax error.
6. Out of memory.

EM MINOL 2.1 SYNTAX Apr. 1976

```
<line>          ::=<number> <statement> cr |<statement> cr
<statement>     ::=<substatement>* : <substatement>
<substatement>  ::=[LET |ϕ] <var> = <expr>
                   PR <expr-list> [;|ϕ]
                   IN <var-str-list>
                   IF <expr><relop><expr> ; <statement>
                   GOTO <expr>
                   CALL <memloc>
                   END
                   RUN
                   LIST
                   NEW
                   CLEAR
                   OS
<number>        ::=<digit>*2 <digit>
<digit>         ::=0|1|. .|8|9
<var>           ::=A|B|. .|Y|Z|<memloc>
<relop>         ::=#|=|<
<expr-list>     ::=[<literal>|<expr>|<str>] [,[<literal>|
                   <expr>|<str>]]*
<var-str-list>  ::=[<var>|<str>] [;[<var>|<str>]]*
<expr>          ::=[<term><aroper>] * <term>
<term>          ::=<var>|<number>| '<schar>' |!
<literal>       ::="<char>*"
<schar>         ::=<char>
<str>           ::=$ <memloc>
<memloc>        ::=(<highadr>,<lowadr>)
<highadr>       ::=<number>
<lowadr>        ::=<number>
<char>          ::=any character except " and cr
```

Notes: < >encloses an element of MINOL
ϕ is the empty set
* repeat limited by length of line
*2 repeat from 0 to 2 times

MINOL

Memory Allocation:
(All locations are split octal)

000 000 - 000 115 I/O Routines, etc.

System Reset:	000	000	061 LXI SP
		001	377
		002	017
		003	317 RST CRLF
		004	303 JMP, MINOL
		005	116
		006	000
CRLF:		010	A subroutine to output a CR followed by a LF.
INPUT:		020	Moves a character from input device to the A register. Parity equals 1. Must output an echo check of the inputted character.
OUTPUT:		040	Outputs character in the A register. Parity equals 1. No registers may be altered.
Must CALL, INT		315	
		363	
		002	

This checks for keyboard interrupt (X^C).

000 116 - 006 377 MINOL Interpreter

000 253 L 261 H	Highest memory location available for MINOL programs.
001 142 L 143 H	Address of user's operating system, or monitor.

All input text is stored at 006 210 - 006 320
Free speace is left for short strings at 006 333 - 006 377
Variables (A-Z) are stored at 005 007 - 005 042

007 000 + Program storage

Executing MINOL:
To start MINOL and initialize program area, EXAMINE 002 350, RUN.
To start without initialization, EXAMINE 000000, RUN.

Dear Jim: **May 24, 1976**

I am enclosing the listing of MINOL—manually typed! There are several features of my program, both positive and negative, that I might point out.

On the plus side, MINOL uses only 1.75K of memory, including the input-output subroutines (although since writing it I see how I can make it even smaller.) Memory locations of the form (H,L) can be used similarly to one- or two-dimensional DIMs in higher BASIC's. Simple input or output strings are possible by specifying a series of memory locations—of the form $(H,L) where H,L is the first location where characters are to be deposited. I am enclosing three programs to illustrate these features. On the negative side, the program is not designed for arithmetic functions, having no grouping of operations, and being limited to a value of 255. The relational operators are restricted to =, #, and <, although > ("greater than") can be done by reversing the logical expressions. Fewer error messages are provided than usual. MINOL is *written completely in machine language* without using IL.

When I can supply MINOL on a cassette I'll let you know. You might like to know that I am in my third year of high school.

Yours truly,
Erik T. Mueller

Britton House
Roosevelt NJ 08555

Additions/changes since the May 1st letter:

Spaces are ignored:
a. During line/statement entry unless enclosed by quotes.
b. When inputting variables.
c. When inputting strings if the L address is zero.

Spaces are accepted:
a. When inputting strings if the L address is non-zero.
b. When enclosed by quotes.

Instead of GOSUB/RET statements, use the following substitute statements to perform the same function:

First initialize the GOSUB stack pointer Y,Z:
2 Y=14:Z=255 (Y and Z are the H,L address of some free space in memory.)

Instead of a GOSUB statement, substitute the

following: LET(Y,Z)=<Return label> :Z=Z-1:GOTO <subroutine label>

Instead of a RET, substitute: Z=Z+1:GOTO(Y,Z)

Free space is left for very short user's strings from 006366 to 006377.

On a directly-executed IN statement, although the data will be correctly stored, an error message may appear after its execution.

The monitor gives a "]" as a prompt. The IN statement gives a "?" unless a sense switch is up.

Three programs in MINOL:

```
]LIST
1Ø PR"GIVE ME A SENTENCE":IN$(14,1)
2Ø PR"STRING TO SEARCH FOR?":IN$(14,101)
21 A=Ø
22 A=A+1:IF(14,A)#255;GOTO22
23 B=Ø
24 B=B+1:IF(14,100+B)#255;GOTO24
3Ø C=1:D=1:S=Ø
4Ø IF(14,D+1ØØ)#(14,C);GOTO7Ø
5Ø D=D+1:C=C+1:IFD<B;GOTO4Ø
6Ø LETS=S+1
65 C=C-1
7Ø LETD=1
8Ø C=C+1:IFC<A;GOTO4Ø
9Ø PR"'";$(14,101);"' OCCURS";S;
96 IFS=1;GOTO1ØØ
97 PR"TIMES IN '";$(14,1);"'":END
100 PR"TIME IN '";$(14,1);"'":END
]RUN
GIVE ME A SENTENCE
? THE BLUE BIRD IN THE BLUE SKY
STRING TO SEARCH FOR?
? BLUE
'BLUE' OCCURS 2 TIMES IN 'THE BLUE BIRD IN THE BLUE SKY'
```

```
]LIST
1Ø "***NUMBER-A NUMBER GUESSING GAME (NUMØ5)
2Ø PR:PR"WHAT IS YOUR NAME";:IN$(14,1)
3Ø X=!:S=Ø:PR"HI,";$(14,1);". WELCOME TO THE GAME OF NUMBER"
4Ø PR"I'M THINKING OF A NUMBER FROM ø TO 255"
5Ø PR"GUESS MY NUMBER!!"
6Ø PR:PR"YOUR GUESS";:ING:S=S+1
65 IFG=X;GOTO9Ø
7Ø IFG<X;PR"TOO SMALL. TRY A BIGGER NUMBER."
8Ø IFX<G;PR"TOO BIG. TRY A SMALLER NUMBER."
85 GOTO6Ø
9Ø PR"THAT'S RIGHT,";$(14,1);"!! YOU GOT IT IN";S;"GUESSES"
1ØØ PR"PLAY AGAIN";:INA:IFA='Y';GOTO3Ø
11Ø PR"OK.....HOPE YOU HAD FUN.":END
```

```
1Ø PR"NAME";:IN$(14,1)
2Ø IF(14,1)='J';IF(14,2)='I';IF(14,3)='M';PR"IT'S JIM!"
3Ø IF(14,1)#'J';PR"IT'S NOT JIM.":GOTO1Ø

]RUN
NAME?ERIK
IT'S NOT JIM.
NAME?JIM
IT'S JIM!
NAME?X^C
BREAK AT 1Ø
]
```

MINOL 2.1 Erik T. Mueller May 1976

```
TAG      ADDRESS I1  I2  I3   MNEMONIC        COMMENTS
MINOL    000116  076 335      MVI A "]"       Output prompt
         000120  347          RST OUT
         000121  315 052 004  CAL,INPTXT      Get input line
         000124  041 210 006  LXI HL:TXT      Point to input text with
         000127  176          MOV A,M           HL
         000130  315 062 005  CAL,CHEKN       Check for label
         000133  322 000 001  JNC,DIRECT      If no label, go execute command
FND      000136  043          INX HL          Point to first non-numeric
         000137  176          MOV A,N           character
         000140  315 062 005  CAL,CHEKN
         000143  332 136 000  JC,FND
         000146  315 256 004  CAL,MKBIN       Convert ASCII label to binary
         000151  021 000 007  LXI DE:PROG     This section edits (inserts,
ZIP      000154  032          LDAX DE           deletes, changes) lines of
         000155  376 215      CPI "CR"          the program
         000157  023          INX DE
         000160  302 154 000  JNZ,ZIP
         000163  032          LDAX DE         Look at line number
         000164  376 377      CPI 377
         000166  312 177 000  JZ,INSRT
         000171  067          STC
         000172  077          CMC
         000173  270          CMP B
         000174  332 154 000  JC,ZIP          Point to line number greater
INSRT    000177  176          MOV A,M           than or equal to entered
         000200  376 215      CPI "CR"          label
         000202  312 345 000  JZ,EKIL         If label alone, delete line
         000205  016 002      MVI C 002
IHR      000207  043          INX HL          Count length of line and add 2
         000210  176          MOV A,M
         000211  014          INR C
         000212  376 215      CPI "CR"
         000214  302 207 000  JNZ,IHR
OHIO     000217  032          LDAX DE         If line entered already exists,
         000220  270          CMP B             first delete the old one,
         000221  302 231 000  JNZ,IBYH          then insert the new one
PZIY     000224  325          PUSH DE
         000225  315 351 000  CAL,KILLINE     Delete old line
         000230  321          POP DE
IBYH     000231  142          MOV H,D         HL points to first location
         000232  153          MOV L,E           where new line will be placed
         000233  325          PUSH DE         Save position in stack
EHR      000234  023          INX DE
         000235  032          LDAX DE
         000236  376 377      CPI 377
         000240  302 234 000  JNZ,EHR          Continue until DE points to
         000243  171          MOV A,C           end of file
         000244  102          MOV B,D         Length of new line in A
         000245  113          MOV C,E
HBY      000246  023          INX DE
         000247  043          INX HL
         000250  365          PUSH PSW
         000251  173          MOV A,E
         000252  376 xxx      CPI xxx         xxx = Low address: limit of
         000254  302 265 000  JNZ,HII           program memory
         000257  172          MOV A,D
         000260  376 xxx      CPI xxx         xxx = High address:memory limit
         000262  312 224 004  JZ,ERR6         Out of memory error
HII      000265  361          POP PSW
         000266  075          DEC A
         000267  302 246 000  JNZ,HBY          Increment until DE points to
UPDT     000272  012          LDAX BC           new end-of-file position,
         000273  022          STAX DE           and HL points to where file
         000274  173          MOV A,E           updating begins
         000275  275          CMP L           BC points to end of file
         000276  302 306 000  JNZ,NHR
         000301  172          MOV A,D
         000302  274          CMP H
NHR      000303  312 313 000  JZ,NET
         000306  013          DCX BC
         000307  033          DCX DE
         000310  303 272 000  JMP,UPDT        Relocate file leaving space for
NET      000313  321          POP DE            new line. Retrieve pointer
STR      000314  041 210 006  LXI HL:TXT
IFD      000317  043          INX HL
         000320  176          MOV A,M
         000321  315 062 005  CAL,CHEKN
         000324  332 317 000  JC,IFD          Point to first non-numeric
         000327  072 120 003  LDA:BIN           character.Put line no. in A
         000332  022          STAX DE         Store line no. in file
NTAT     000333  023          INX DE          Store line text in file
         000334  176          MOV A,M
         000335  022          STAX DE
         000336  043          INX HL
         000337  376 215      CPI "CR"
         000341  302 333 000  JNZ,NTAT
         000344  307          RST RESET       Go back to monitor section
EKIL     000345  315 351 000  CAL,KILLINE     Delete line
         000350  307          RST RESET
KILLINE  000351  032          LDAX DE         Delete a line
         000352  270          CMP B
         000353  300          RNZ             If deleting line that does not
         000354  142          MOV H,D           exist, return
         000355  153          MOV L,E
BBL      000356  043          INX HL
         000357  176          MOV A,M
         000360  376 215      CPI "CR"
         000362  302 356 000  JNZ,BBL          Point to next line
ARK      000365  043          INX HL          Relocate file, deleting line
         000366  176          MOV A,M
         000367  022          STAX DE
         000370  376 377      CPI 377
         000372  310          RZ
         000373  023          INX DE
         000374  303 365 000  JMP,ARK
         000377  000          NOP
DIRECT   001000  317          RST CRLF        Direct execution of a statement
         001001  257          XRA A
         001002  062 121 003  STA:LNE         Set LNE (current line no.)=0
         001005  303 041 001  JMP,EXEC        Execute statement
RUN      001010  041 000 007  LXI HL:PROG     RUN Statement: Start from be-
LPUB     001013  176          MOV A,M           ginning of program. Get next
         001014  376 272      CPI ":"           statement.
         001016  043          INX HL
         001017  312 041 001  JZ,EXEC         If not a new line, go execute
         001022  376 215      CPI "CR"          statement
         001024  302 013 001  JNZ,LPUB
BIB      001027  176          MOV A,M
         001030  376 377      CPI 377
         001032  312 000 000  JZ,RESET        If statement no.=377 (end of
         001035  062 121 003  STA:LNE           program), go back to monitor
         001040  043          INX HL          If not 377, store current line
EXEC     001041  315 363 002  CAL,INT           no. at LNE
         001044  043          INX HL          Check for keyboard program
         001045  176          MOV A,M           interrupt
         001046  376 275      CPI "="         Check for "=" in second column
         001050  312 221 001  JZ,LET            (variable assignment)
         001053  053          DCX HL
         001054  176          MOV A,M
         001055  376 250      CPI "("         If "(" in first column (memory
         001057  312 221 001  JZ,LET            location assignment)go to LET
         001062  376 303      CPI "C"         Check for "C"
         001064  302 106 001  JNZ,GSM          If not, go on
         001067  043          INX HL
         001070  176          MOV A,M
         001071  376 301      CPI "A"
         001073  312 323 002  JZ,CALL         CALL Statement
         001076  376 314      CPI "L"
         001100  312 304 002  JZ,CLR          CLEAR Statement
         001103  303 205 004  JMP,ERR3        If neither, report error
```

```
001106  376 305      CPI "E"          Check for "E" as in END.
001110  312 000 000  JZ,RESET
001113  376 307      CPI "G"
001115  312 007 003  JZ,GOTO
001120  376 242      CPI '"'          Check for " indicating
001122  312 013 001  JZ,LPUB             REM statement
001125  376 316      CPI "N"
001127  312 350 002  JZ,NEW
001132  376 320      CPI "P"
001134  312 327 001  JZ,PR
001137  376 317      CPI "O"
001141  312 LLL hhh  JZ,OS             Address of user's monitor
001144  376 322      CPI "R"
001146  312 010 001  JZ,RUN
001151  376 311      CPI "I"
001153  302 175 001  JNZ,LS
001156  043          INX HL
001157  176          MOV A,M
001160  376 316      CPI "N"
001162  312 116 002  JZ,IN
001165  376 306      CPI "F"
001167  312 324 005  JZ,IF
001172  303 205 004  JMP,ERR3
LS      001175  376 314      CPI "L"
001177  302 205 004  JNZ,ERR3
001202  043          INX HL
001203  176          MOV A,M
001204  376 305      CPI "E"
001206  312 221 001  JZ,LET
001211  376 311      CPI "I"
001213  312 132 005  JZ,LIST
001216  303 205 004  JMP,ERR3
LET     001221  176          MOV A,M          LET Statement executer
001222  315 043 005  CAL,TERM         Find "="
001225  332 217 004  JC,ERR5          Report error if no "=" before
001230  376 275      CPI "="             CR or ":"
001232  043          INX HL
001233  302 221 001  JNZ,LET
001236  303 321 006  JMP,FIX          Transfer expression text in
001241  000 000      NOP                 program text, to expression
001243  043          INX HL              buffer
001244  023          INX DE
001245  322 324 006  JNC,MREN
001250  315 162 003  CAL,EXPR
SERCH   001253  176          MOV A,M          Go back before "="
001254  376 275      CPI "="
001256  053          DCX HL
001257  302 253 001  JNZ,SERCH
001262  176          MOV A,M
001263  315 371 004  CAL,CHEKLTR
001266  322 301 001  JNC,INLET         If not variable, get memory
001271  315 343 004  CAL,GETADR           address
001274  171          MOV A,C
001275  022          STAX DE          Store in variable
001276  303 013 001  JMP,LPUB          Next statement
INLET   001301  376 251      CPI ")"
001303  302 212 004  JNZ,ERR4
JHR     001306  053          DCX HL
001307  176          MOV A,M
001310  376 250      CPI "("
001312  302 306 001  JNZ,JHR
001315  171          MOV A,C
001316  365          PUSH PSW
001317  315 062 006  CAL,VAL          Get memory location in BC
001322  361          POP PSW
001323  002          STAX BC
001324  303 013 001  JMP,LPUB
PR      001327  043          INX HL           PR Statement executer
001330  043          INX HL           Skip assumed characters
001331  176          MOV A,M
001332  315 043 005  CAL,TERM         If blank print, go to CR
001335  332 016 002  JC,DCR
NXTE    001340  376 242      CPI '"'          Check for literal
001342  302 022 002  JNZ,VAR          If not, go on
HR      001345  043          INX HL           Print text until " found
001346  176          MOV A,M
001347  376 342      CPI '"'
001351  312 366 001  JZ,MRENO
001354  315 043 005  CAL,TERM         If terminator before closing
001357  332 217 004  JC,ERR5             quotes, print error
001362  347          RST OUT
001363  303 345 001  JMP,HR
MRENO   001366  043          INX HL
001367  176          MOV A,M
001370  315 043 005  CALL,TERM         If end of statement without
001373  332 016 002  JC,DCR              semicolon ";",go do CR
001376  376 273      CPI ";"
002000  302 217 004  JNZ,ERR5
002003  043          INX HL
002004  176          MOV A,M
002005  315 043 005  CAL,TERM         If term after semicolon, do not
002010  332 017 002  JC,NCR              print CR
002013  303 340 001  JMP,NXTE         Get next thing to print
DCR     002016  317          RST CRLF
NCR     002017  303 013 001  JMP,LPUB
VAR     002022  376 244      CPI "$"
002024  312 076 002  JZ,STR            Check if string
002027  021 151 006  LXI DE:EXP
002032  076 240      MVI A "SP"       Output leading space
002034  347          RST OUT
ER      002035  176          MOV A,M          Transfer expression text from
002036  022          STAX DE             program text to expression
002037  043          INX HL              buffer
002040  023          INX DE
002041  315 043 005  CAL,TERM
002044  332 054 002  JC,HR
002047  376 274      CPI ";"
002051  302 035 002  JNZ,ER
HR      002054  053          DCX HL
002055  033          DCX DE
002056  076 215      MVI A "CR"
002060  022          STAX DE
002061  315 162 003  CAL,EXPR
002064  101          MOV B,C
002065  315 174 005  CAL,PBINBCD       Print expression's value
002070  076 240      MVI A "SP"
002072  347          RST OUT
002073  303 367 001  JMP,MRENO+1
STR     002076  043          INX HL
002077  315 062 006  CAL,VAL          Get start address of string
MRE     002102  012          LDAX BC          in BC. Print string
002103  347          RST OUT
002104  376          CPI
002105  377          377
002106  003          INX BC
002107  302 102 002  JNZ,MRE
002112  043          INX HL
002113  303 367 001  JMP,MREN
IN      002116  043          INX HL           Input statement
002117  333          INP
002120  377          377
002121  376          CPI              If sense switches down,print"?"
002122  000          000
002123  302 134 002  JNZ,EAHR
002126  076 277      MVIA "?"
002130  347          RST OUT
002131  076 240      MVIA "SP"
002133  347          RST OUT
EAHR    002134  176          MOV A,M
002135  315 371 004  CAL,CHECKLET       Check for variable
002140  332 164 002  JC,LVB
002143  376 244      CPI "$"          Check for input string
002145  312 247 002  JZ,STRIN
002150  376 250      CPI "("          Check for single memory
```

```
                002152  302 212 004  JNZ,ERR4        location
                002155  315 150 003  CAL,VALDE       Get location in DE
                002160  345          PUSH HL
                002161  303 170 002  JMP,HS
LVB             002164  345          PUSH HL
                002165  315 343 004  CAL,GETADR      Get address of letter variable.
HS              002170  325          PUSH DE
IHERE           002171  315 052 004  CAL,INPTXT      Input a line
                002174  317          RST CRLF
                002175  041 210 006  LXI HL:TXT
                002200  176          MOV A,M
                002201  315 062 005  CAL,CHEKN       Check for number
                002204  322 224 002  JNC,LETR
FD              002207  043          INX HL
                002210  176          MOV A,M
                002211  315 062 005  CAL,CHEKN
                002214  332 207 002  JC,FD           Point to first non-numeric
                002217  305          PUSH BC           character
                002220  315 256 004  CAL,MKBIN       Convert ASCII input data to
                002223  301          POP BC            binary
LETR            002224  321          POP DE
                002225  022          STAX DE         Put A in variable
CHK             002226  341          POP HL
                002227  043          INX HL
                002230  176          MOV A,M
                002231  376 254      CPI ","         Check for more input variables
                002233  312 116 002  JZ,IN
                002236  315 043 005  CAL,TERM
                002241  332 013 001  JC,LPUB
                002244  303 217 004  JMP,ERR5
STRIN           002247  043          INX HL          Input string
                002250  315 062 006  CAL,VAL         Get first memory location in BC
                002253  345          PUSH HL
                002254  315 360 006  CAL,DMS         Input a line
                002257  317          RST CRLF
                002260  041 210 006  LXI HL:TXT
LD              002263  176          MOV A,M         Store text beginning at spec-
                002264  376 215      CPI "CR"          ified location
                002266  312 276 002  JZ,TE
                002271  002          STAX BC
                002272  003          INX BC
                002273  303 354 006  JMP,CIN
JE              002276  076 377      MVI A 377       Store 377 at end of string
                002300  002          STAX BC
                002301  303 226 002  JMP,CHK
CLR             002304  021 007 005  LXI DE          CLEAR executer.Var storage
LCR             002307  257          XRA A
                002310  022          STAX DE
                002311  023          INX DE
                002312  173          MOV A,E
                002313  376 042      CPI 042         Last variable location
                002315  302 307 002  JNZ,LCR
                002320  303 013 001  JMP,LPUB
CALL            002323  000          NOP             Skip assumed characters
                002324  043          INX HL
                002325  043          INX HL
                002326  043          INX HL
                002327  315 062 006  CAL,VAL         Get address in BE
                002332  345          PUSH HL
                002333  325          PUSH DE
                002334  021 343 002  LXI DE:RET       Load DE with return address
                002337  325          PUSH DE         Push return address into stack
                002340  140          MOV H,B
                002341  151          MOV L,C
                002342  351          PCHL            Jump to user's subroutine
RET             002343  321          POP DE
                002344  341          POP HL
                002345  303          JMP,LPUB
NEW             002350  021 000 007  LXI DE:PROG      New
                002353  076 215      MVI A "CR"      Initialize Program area
                002355  022          STAX DE
                002356  023          INX DE
                002357  076 377      MVI A
                002361  022          STAX DE
                002362  307          RST RESET
INT             002363  365          PUSH PSW        This subroutine checks for
                002364  333 013      INP 013           X^C from keyboard
                002366  376 203      CPI "X^C"
                002370  312 375 002  JZ,BREAK
                002373  361          POP PSW
                002374  311          RET
BREAK           002375  317          RST CRLF
                002376  021 112 003  LXI DE           "BREAK"
                003001  315 360 004  CAL,PRINTXT
                003004  303 240 004  JMP,AT+
GOTO            003007  043          INX HL          GOTO executer
                003010  043          INX HL          Skip assumed characters
                003011  043          INX HL
                003012  043          INX HL
                003013  021 151 006  LXI DE:EXPR      Calculate value of expression
RME             003016  176          MOV A,M
                003017  022          STAX DE
                003020  315 043 005  CAL,TERM
                003023  043          INX HL
                003024  023          INX DE
                003025  322 016 003  JNC,RME
                003030  315 162 003  CAL,EXPRS
                003033  101          MOV B,C
                003034  171          MOV A,C
                003035  376 000      CPI
                003037  302 050 003  JNZ,JUMP
                003042  041 210 005  LXI HL:TXT
                003045  303 000 001  JMP,DIRECT
JUMP            003050  041 000 007  LXI:PROG HL     Goto expression
DUP             003053  176          MOV A,M
                003054  376 215      CPI "CR"
                003056  043          INX HL
                003057  302 053 003  JNZ,DUP
                003062  176          MOV A,M
                003063  376 377      CPI 377
                003065  312 173 004  JZ,ERR1
                003070  270          CMP B
                003071  302 053 003  JNZ,DUP
                003074  303 027 001  JMP,BIB
!ERR            003077  241 305 322  DW "!ER"        Error & Break Texts
                003102  322 240      DW "R "
                003104  377          DB 377
AT              003105  240 301 324  DW " AT"
                003110  240          DB " "
                003111  377          DB 377
BREAK           003112  302 322 305  DW "BRE"
                003115  301 313      DW "AK"
                003117  377          DB 377
BIN             003120  000
LNE             003121  000          Current line no. storage
                003122  000          NOP
ACT             003124  305          PUSH BC         Gets the value of a memory
                003124  353          XCHG              location
                003125  315 062 006  CAL,VAL         Part of the EXPRS subroutine
                003130  353          XCHG              to follow
                003131  012          LDAX BC
                003132  301          POP BC
                003133  107          MOV B,A
                003134  023          INX DE
                003135  303 247 003  JMP,GETNET
NOTEQ           003140  173          MOV A,E         If two expressions are not
                003141  271          CMP C             equal, execute statement
                003142  302 041 001  JNZ,EXEC          (Part of IF statement exe-
                003145  303 013 001  JMP,LPUB          cutor to follow)
VALDE           003150  315 062 006  CAL,VAL
                003153  120          MOV D,B
                003154  131          MOV E,C
                003155  311          RET
                003156  000 000 000  NOP
```

```
                003161  000             NOP
EXPRS           003162  345             PUSH HL         Calculates the value of an
                003163  021 150 006     LXI DE:EXPR-1     expression stored in memory.
                003166  016             MVI C             DE is the cursor
                003167  000             000
RETPT           003170  032             LDAX DE
                003171  315 043 005     CAL,TERM        Check for end of expression
                003174  322 201 003     JNC,GOMOR
                003177  341             POP HL
                003200  311             RET
GOMOR           003201  365             PUSH PSW        Save Operation in stack.
                003202  023             INX DE          Get Term/Factor
                003203  032             LDAX DE
                003204  376 247         CPI "'"         Single character value
                003206  312 271 003     JZ,ASC
                003211  376 250         CPI "C"         Memory location
                003213  312 123 003     JZ,ACT
                003216  376 241         CPI "!"         Random number
                003220  312 000 004     JZ,RND
                003223  315 062 005     CAL,CHEKN
                003226  332 350 003     JC,CONSTANT     Constant (number)
                003231  315 371 004     CAL,CHEKL
                003234  322 217 004     JNC,ERR5
                003237  023             INX DE
IVAR            003240  325             PUSH DE         Variable
                003241  315 343 004     CAL,GETADR
                003244  032             LDAX DE
                003245  107             MOV B,A
                003246  321             POP DE
GETNET          003247  361             POP PSW         Retrieve Operation
                003250  376 253         CPI "+"
                003252  312 277 003     JZ,ADD
                003255  376 255         CPI "-"
                003257  312 305 003     JZ,SUB
                003262  376 252         CPI "*"
                003264  312 313 003     JZ,MULT
                003267  376 257         CPI "/"
                003271  312 326 003     JZ,DIV
                003274  303 217 004     JMP,ERR5
ADD             003277  171             MOV A,C         Add C=C+B
                003300  200             ADD B
                003301  117             MOV C,A
                003302  303 170 003     JMP,RETPT
SUB             003305  171             MOV A,C         C=C-B
                003306  220             SUB B
                003307  117             MOV C,A
                003310  303 170 003     JMP,RETPT
MULT            003313  171             MOV A,C         C=C*B
                003314  005             DCR B
GBK             003315  201             ADD C
                003316  005             DCR B
                003317  320             JNZ,GBK
                003322  117             MOV C,A
                003323  303 170 003     JMP,RETPT
DIV             003326  171             MOV A,C         C=C/B
CTUE            003331  014             INCC
                003332  220             SUB B
                003333  312             JZ,ZER
                003336  332 344 003     JC,MIN
                003341  303 331 003     JMP,CTUE
MIN             003344  015             DCR C
ZER             003345  303 170 003     JMP,RETPT
CONSTANT        003350  023             INX DE
                003351  032             LDAX DE
                003352  315 062 005     CAL,CHEKN
                003355  332 350 003     JC,CONSTANT
                003360  353             XCHG
                003361  315 310 005     CAL,SURE
                003364  353             XCHG
                003365  107             MOV B,A
                003366  303 247 003     JMP,GETNET
ASC             003371  023             INX DE
                003372  032             LDAX DE
                003373  107             MOV B,A
                003374  023             INX DE
                003375  303 134 003     JMP,GETNET
                004000  041 051 004     LXI HL:SH+3     Random Number Generator
                004003  006 010         MVI B 010         by Jim Parker
RTOP            004006  007             RLC
                004007  007             RLC
                004010  007             RLC
                004011  256             XRA M
                004012  027             RAL
                004013  027             RAL
                004014  055             DCR L
                004015  055             DCR L
                004016  055             DCR L
                004017  176             MOV A,M
                004020  027             RAL
                004021  167             MOV M,A
                004022  054             INR L
                004023  176             MOV A,M
                004024  027             RAL
                004025  167             MOV M,A
                004026  054             INR L
                004027  176             MOV A,M
                004030  027             RAL
                004031  167             MOV M,A
                004032  054             INR L
                004033  176             MOV A,M
                004034  027             RAL
                004035  167             MOV M,A
                004036  004             DCR B
                004037  302 006 004     JNZ,RTOP
                004042  107             MOV B,A
                004043  303 134 003     JMP,GETNET
SH              004046  021 231 132     SEED
                004051  351             DATA
INPTX           004052  016 000         MVIC 000        Input a line of 72 characters
INPXT           004054  041 207 006     LXI HL:TEXT-1
INO             004057  327             RST INPUT
HRER            004057  327             MOV B,A
                004061  171             MOV A,C
                004062  376 000         CPI 000
                004064  170             MOV A,B
                004065  302 075 004     JNZ,MID
                004070  376 240         CPI "SP"        Do not accept space if outside
                004072  312 057 004     JZ,INO            quotes
MID             004075  376 242         CPI '"'
                004077  302 117 004     JNZ,GOON
                004102  171             MOV A,C
                004103  376 000         CPI 000
                004105  312 115 004     JZ,MRF
                004110  016 000         MVI C 000
                004112  303 117 004     JMP,GOON
MRF             004115  016 003         MVI C 003
GOON            004117  376 214         CPI X^L         If X^L redo line
                004121  302 133 004     JNZ,HRD
                004124  076 334         MVIA "/"
                004126  347             RST OUT
                004127  317             RST CRLF
                004130  303 052 004     JMP,INPTXT
HRD             004133  376 215         CPI "CR"
                004135  302 144         JNZ,CHM
                004140  303 316 005     JMP,HELP
                004143  000             NOP
CHM             004144  376 223         CPI X^S         If X^S go back a character
                004146  302 160 004     JNZ,CTN
                004151  076 337         MVI A "←"
                004153  347             RST OUT
                004154  053             DCX HL
                004155  303 057 004     JMP,IND
CTN             004160  175             MOV A,L
                004161  376 320         CPI 320         Low top address
```

```
            004163  312 200 004  JZ,ERR2          If over 72 characters, report
            004166  043          INX HL             error
STOR        004167  160          MOV M,B
            004170  303 057 004  JMP,INO
ERR1        004173  006 261      MVI B "1"
            004175  303 226 004  JMP,ERR
ERR2        004200  006 262      MVI B "2"
            004202  303 226 004  JMP,ERR
ERR3        004205  006 263      MVI B "3"
            004207  303 226 004  JMP,ERR
ERR4        004212  006 264      MVI B "4"
            004214  303 226 004  JMP,ERR
ERR5        004217  006 265      MVI B "5"
            004221  303 226 004  JMP,ERR
ERR6        004224  006 266      MVI B "6"
ERR         004226  317          RST CRLF         Print "ERR" message
            004227  021 077 003  LXI DE:ERR       "!ERR"
            004232  315 360 004  CAL,PRINTXT
            004235  170          MOV A,B
            004236  347          RST OUT
            004237  000          NOP
AT+         004240  021 105 003  LXI DE           " AT "
            004243  315 360 004  CAL,PR INTXT
            004246  072 121 003  LDA:STATN
            004251  107          MOV B,A
            004252  315 174 005  CAL,PBINBCD
            004255  307          RST 000
MKBIN       004256  325          PUSH DE          Get value of ASCII Numbers
            004257  053          DCX HL
            004260  176          MOV A,M
            004261  326 260      SUI 260
            004263  107          MOV B,A
            004264  053          DCX HL
            004265  176          MOV A,M
            004266  315 062 005  CAL,CHEKN
            004271  332 303 004  JC,STOC
            004274  016 000      MVI C 0
            004276  036 000      MVI E 0
            004300  303 330 004  JMP,INK2
STOC        004303  326 260      SUI 260
            004305  117          MOV C,A
ENT         004306  053          DCX HL
            004307  176          MOV A,M
            004310  315 062 005  CAL,CHEKN
            004313  332 323 004  JC,STOE
            004316  036 000      MVI E 0
            004320  303 327 004  JMP,INK3
STOE        004323  176          MOV A,M
            004324  326 260      SUI 260
            004326  137          MOV E,A
INK3        004327  043          INX HL
INK2        004330  043          INX HL
            004331  043          INX HL
            004332  315 102 005  CAL,BCDBIN
            004335  170          MOV A,B
            004336  062 120 003  STA:BIN
            004341  321          POP DE
            004342  311          RET
GETADR      004343  345          PUSH HL          Get Address of variable
            004344  021 007 005  LXI DE:VARSTOR
            004347  326 301      SUI 301
            004351  046 000      MVI H 000
            004353  157          MOV L,A
            004354  031          DAD DE
            004355  353          XCHG
            004356  341          POP HL
            004357  311          RET
PRINTXT     004360  032          LDAX DE          Print text pointed to by DE
            004361  376 377      CPI 377
            004363  310          RZ
            004364  347          RST OUT
            004365  023          INX DE
            004366  303 360 004  JMP,PRINTXT
CHEKLTR     004371  067          STC              Check if a character is a
            004372  077          CMC                letter
            004373  376 301      CPI 301
            004375  332 004 005  JC,NOTAP
            005000  067          STC
            005001  376 333      CPI 333
            005003  311          RET
NOTAP       005004  077          CMC
            005005  311          RET
            005006  000          NOP
VARSTOR     005007-005042   000                   Variable storage
TERM        005043  376 215      CPI "CR"         Check for statement terminator
            005045  312 060 005  JZ,YES              (CR or :)
            005050  376 272      CPI ":"
            005052  312 060 005  JZ,YES
            005055  067          STC
            005056  077          CMC
            005057  311          RET
YES         005060  067          STC
            005061  311          RET
CHECKN      005062  067          STC
            005063  077          CMC
            005064  376 260      CPI 260
            005066  332 075 005  JC,NOTA
            005071  067          STC
            005072  376 272      CPI 272
NOT         005074  311          RET
NOTA        005075  077          CMC
            005076  311          RET
            005077  000 000 000  NOP
BCDBIN      005102  076 012      MVI A 012        BCD to BIN subroutine
            005104  200          ADD B
            005105  107          MOV B,A
            005106  015          DCR B
            005107  302 102 005  JNZ,BCDBIN
            005112  000 000 000  NOP
SEC         005115  113          MOV C,E
            005116  173          MOV A,E
            005117  376 377      CPI 377
THI         005121  310          RZ
            005122  076 144      MVI A 144
            005124  200          ADD B
            005125  107          MOV B,A
            005126  015          DCR C
            005127  303 121 005  JMP,THI
LIST        005132  021 001 007  LXI DE:PROG+1    List Command
NEXN        005135  032          LDAX DE
            005136  376 377      CPI 377
            005140  312 013 001  JZ,LPUB
            005143  107          MOV B,A
            005144  315 174 005  CAL,PBINBCD
            005147  076 240      MVI A "SP"
            005151  347          RST OUT
            005152  023          INX DE
            005153  032          LDAX DE
OU          005154  347          RST OUT
            005155  376 215      CPI "CR"
            005157  302 167 005  JNZ,MREN
            005162  023          INX DE
            005163  317          RST CRLF
            005164  303 135 005  JMP,NEXN
MREN        005167  023          INX DE
            005170  032          LDAX DE
            005171  303 154 005  JMP,OU
PBINBCD     005174  305          PUSH BC          Print Binary number
            005175  325          PUSH DE
            005176  026 000      MVI D 000
            005200  016 000      MVI C 000
            005202  170          MOV A  B
IFIR        005203  326 144      SUI 144
            005205  332 214 005  JC,ISEC
```

```
005210  014          INC C
005211  303 203 005  JMP,IFIR
ISEC    005214  006 144      MVI B 144
005216  200          ADD B
005217  107          MOV B,A
005220  076 260      MVI A 260
005222  201          ADD C
005223  376 260      CPI 260
005225  312 271 005  JZ,NP
005230  347          RST OUT
GOM     005231  016 000      MVI C 000
005233  170          MOV A,B
ITHR    005234  326 012      SUI 012
005236  332 245 005  JC,FOR
005241  014          INC C
005242  303 234 005  JMP,ITHR
FOR     005245  006 012      MVI B 012
005247  200          ADD B
005250  107          MOV B,A
005251  076 260      MVI A 260
005253  201          ADD C
005254  376 260      CPI 260
005256  276 005      JZ,INV
IPR     005261  347          RST OUT
DPR     005262  076 260      MVI A 260
005264  200          ADD B
005265  347          RST OUT
005266  321          POP DE
005267  301          POP BC
005270  311          RET
NP      005271  026 001      MVI D 001
005273  303 231 005  JMP,COM
INU     005276  117          MOV C,A
005277  257          XRA A
005300  272          CMP D
005301  171          MOV A,C
005302  302 262 005  JNZ,DPR
005305  303 261 005  JMP,IPR
SURE    005310  305          PUSH BC        Calls MKBIN with saving BC
005311  315 256 004  CAL,MKBIN
005314  301          POP BC
005315  311          RET
HELP    005316  043          INX HL         Adds 377 terminator at end of
005317  160          MOV A,B                 INPUT Text
005320  043          INX HL
005321  066 377      MVI M 377
005323  311          RET
IF      005324  021 151 006  LXI DE:EXPR    IF Statement
NGO     005327  043          INX HL         First expression
005330  176          MOV A,M
005331  376 243      CPI "#"
005333  312 361 005  JZ,COMP
005336  376 275      CPI "="
005340  312 361 005  JZ,COMP
005343  376 274      CPI "<"
005345  312 361 005  JZ,COMP
005350  315 043 005  CAL,TERM
005353  315 334 006  CAL,DIN
005356  303 327 005  JMP,NGO
COMP    005361  365          PUSH PSW
005362  076 215      MVI A "CR"
005364  022          STAX DE
005365  315 162 003  CAL,EXPRS
005370  305          PUSH BC
005371  021 151 006  LXI DE:EXP
NXTVR   005374  043          INX HL         Second expression
005375  176          MOV A,M
005376  376 273      CPI ","
006000  312 016 006  JZ,PHI
006003  315 043 005  CAL,TERM
006006  332 217 004  JC,ERR5
006011  022          STAX DE
006012  023          INX DE
006013  303 374 005  JMP,NXTVR

PHI     006016  043          INX HL
006017  076 215      MVI A "CR"
006021  022          STAX D
006022  315 162 003  CAL,EXPRS
006025  321          POP DE
006026  361          POP PSW
006027  376 243      CPI "A"        Check relational operator
006031  312 140 003  JZ,NOTEQ
006034  376 274      CPI "<"
006036  312 051 006  JZ,LESSTH
EQU     006041  173          MOV A,E        Check if =
006042  271          CMP C
006043  312 041 001 JZ,EXEC
006046  303 013 001  JMP,LPUB
LESSTH  006051  173          MOV A,E        Check if <
006052  271          CMP C
006053  322 013 001  JC,EXEC
006056  303 041 001  JMP,LPUB
006061  000          NOP
VAL     006062  325          PUSH DE        Get address of memory location
006063  021 151 006  LXI DE:EXPR
SHME    006066  043          INX HL
006067  176          MOV A,M
006070  022          STAX DE
006071  315 043 005  CAL,TERM
006074  315 342 006  CAL,PLIN
006077  376 254      CPI ","
006101  302 066 006  JNZ,SMHE
006104  315 347 006  CAL,DCXN
006107  315 162 003  CAL,EXPRS
006112  305          PUSH BC
006113  021 151 006  LXI DE:EXPR
MIG     006116  043          INX HL
006117  176          MOV A,M
006120  022          STAX DE
006121  315 043 005  CAL,TERM
006124  315 342 006  CAL,PLIN
006127  376 251      CPI ")"
006131  302 116 006  JNZ,MIG
006134  315 347 006  CAL,DCXN
006137  315 162 003  CAL,EXPRS
006142  171          MOV A,C
006143  301          POP BC
006144  101          MOV B,C
006145  117          MOV C,A
006146  321          POP DE
006147  311          RET
006150  253          DB "+"
EXPR    006151-006207  000            Expression buffer
TEXT    006210-006320  000            Line buffer (Input text)
FIX     006321  021 151 006  LXI DE:EXP
MREN    006324  176          MOV A,M        Most of the following is
006325  022          STAX DE          to patch up mistakes in
006326  315 043 005  CAL,TERM          the original program
006331  303 242 001  JMP,BACK
DIN     006334  332 217 004  JC,ERR5
006337  022          STAX DE
006340  023          INX DE
006341  311          RET
PLIN    006342  023          INX DE
006343  332 217 004  JC,ERR5
006346  311          RET
DCXN    006347  033          DCX DE
006350  076 215      MVI A "CR"
006352  022          STAX DE
CIN     006354  043          INX HL
006355  303 263 002  JMP,LD
DMS     006360  305          PUSH BC
006361  315 054 004  CAL,INPXT
006364  301          POP BC
006365  311          RET
006366-006377  000            Extra space for user's use.
007000+                        MINOL Programs
```

MINOL Errata & Praise

Dear Jim: July 5, 1976

I have just received a letter from Joseph F. Gaffney listing a zillion errors or typos in the MINOL listing. Below is a list of the corrections that should be made. Apparently, the listing has been published. But I still haven't received the issue or any issues after the third. Please check with the subscription department for me. **ERRORS** (Most of them were pointed out by Joseph F. Gaffney, 321 Lyndhurst Ave., Lyndhurst NJ 07071.)

Changes are underlined.

```
GSM     001106
        002345 303 013 001
ACT     003123
        003211              CPI "("
        003206 312 371 003
        003317 320 315 003
        003327 016 000      MVI C 0
        003333 312 345 003
        003375 303 134 003  JMP, NXGT
        004043 303 134 003  JMP, NXGT
NXGT    003134
        004005 176          MOV A,M
INPTXT  004052
        004135 302 144 004
        004155 303 057 004  JMP, INQ
CHEKN   005062
        005256 312 276 005  JZ, INU
        006027 376 243      CPI "#"
        006353 311          RET
        006053 322 013 001  JC, LPBUB
        006056 303 041 001  JMP, EXEC
        006101 302 066 006  JNZ,SHME
```

Sincerely yours,

Erik T. Mueller
36 Homestead Lane
Roosevelt NJ 08555

Thanks for the errata. Your subscription was entered on May 19th. Issues no. 4 and no. 5 were mailed a week and a half apart, about a month prior to your letter. I *encourage* you to complain to your local congressional reps (complaining to the Post Office appears to be useless). I also mailed an extra copy of the issue in which MINOL appeared, separately.

–JCW

Dear Mr. Warren, July 19, 1976

Erik Mueller's MINOL version of Tiny BASIC in the April issue is fantastic, and I'm really enjoying it! I relocated it to fit with my monitor (a modified 'JAMON' [MITS User's Group]), and it's running with a Model 33 Teletype. Some of the MINOL subroutines are useful in other programs as well, and are easily called (particularly useful is PRINTXT). MINOL is fun, certainly, but it is also very amazing (how can it be so smart and yet so small?).

There were a few typographical errors which were easy to correct. Corrections (at the original addresses) are shown below.

Address	Was	Change to
001/350	342	242
002/050	274	273
002/346	OMITTED	013,001
003/207	271	371
003/317	320	302
003/320	OMITTED	315,003
003/327	OMITTED	016,000
003/334	OMITTED	345,003
004/005	OMITTED	176
004/060	OMITTED	107
004/137	OMITTED	004
005/256	OMITTED	312
005/257	DISPLACED	276,005
006/353	OMITTED	311

As the program stands, the processor will enter an endless loop if you try to divide by zero. This doesn't hurt anything, but it does hang it up. To cure this, you might wish to add the following routine to test for division by zero. It adds Error 7.

```
Change:          003/326    315,000,004   CALL DIV0

DIV0 * MOVAB     004/000    170           MOVE B TO A
       ORAA      004/001    267           SET STATUS
       MOVAC     004/002    171           MOVE C TO A;
                                          STATUS UN-
                                          AFFECTED
       MVIC      004/003    016,000       CLEAR C
       RNZ       004/005    300           RETURN NOT
                                          ZERO
       MVIB      004/006    006,067       ERR '7'
       JMP       004/010    303,226,004   JMP ERR
```

*This is my 'relocated' code. Any convenient locations will do.

Yours truly,

Phillip L. Hansford
6841 Haywood St.
Tujunga CA 91042

EUGENE STORE: THE REAL OREGON COMPUTER CO.

Dear Bob, 4/26/76

Indeed we are running a store and would love it if you mentioned us. The store opened May 8.

Thanks,

John Montgomery
The Real Oregon Computer Co.
205 W 10th
Eugene OR 97401

NEW CLUB CONTACTS: VENTURA COUNTY COMPUTER SOCIETY

VCCS is a Chapter of the Southern California Computer Society. Its mailing address is P.O. Box 525, Port Hueneme, CA 93041. For more direct responses, contact their Secretary, Fred Moeckel, 4240 Harbor Blvd. No.208, Oxnard, CA 93030.