Issue # 81/1

PolyLetter

PUBLISHED BI-MONTHLY January / February

Welcome to a new year of PolyLetter! Several changes have happened recently, the biggest one is our new address:

PolyLetter
1437 Sugarwood Lane
Norcross, GA 30093
(404) 925-2480

Starting with this issue, our organization is changing a bit. Mark Sutherland has turned over publishing responsibility to me, although he is still a Poly owner and remains very interested in PolyLetter. Mark and his wife Winanne will continue to help publish the newsletter. The format of the newsletter will remain much the same as always.

This PolyLetter was typeset on a NEC Spinwriter in my office at Chromatics, Inc. I work as a technical writer in the R & D department. Chromatics produces very-high-resolution color graphics computers and terminals, used for business graphics, process control, animation, etc. Write me for a brochure if you're curious.

This issue we complete Ralph Kenyon's tutorial on assembly language programming for the Poly. Our next PolyLetter will feature Frank Stearns' article on how to "squeeze" a BASIC program for speed and size savings.

Thanks to all of you for supporting PolyLetter through its first year. You can contact me almost any evening at the phone number above, and I'm always happy to talk PolyTalk. Or, if you'd rather talk to my computer, let me know and I can arrange for my modem to answer!

Bob Bybee

*** SYSTEM ENHANCEMENTS ***

The latest version of the operating system is called Exec/93. These are some of its highlights:

Exec has a new command, DLIST, which lists the deleted files in a directory.

File handling in the entire system has been improved for greater speed and reliability. The Editor won't delete your old file if there's no room for a new one on the disk. It has also been made more "friendly" in the event of a full directory. Several other Editor bugs have been corrected in the

BASIC's newest version is "C02 12/12/80." It handles the TwinSystem and has these additional features:

FILE: x,DEL deletes the file on channel x and closes the file.

PEEK(0) and POKE 0,x address the current cursor position, making it easy to blank the cursor (an old PolyLetter trick).

PEEK(1) tells whether the program is being executed at boot time (as an INITIAL file).

PEEK(3) and PEEK(4) return the addresses of the command line and command file buffers, for programs that like to read input in sneaky ways.

PEEK(5) returns 0 on a single-user system, and 1 or 2 on a TwinSystem, depending on which user is running the program.

PEEK(9) returns the value of MEMTOP.

(continued on page 2)

POKE 6,x and POKE 7,x put values into the video screen start and end locations (to protect part of the screen).

Z=CALL(0,"string") closes all files and exits the program to Exec. If "string" is present, it is passed to Exec as a command.

Z=CALL(1) calls the disk input/output routine, Dio.

Many of the CALLs and POKEs above could be done under the old Exec, if you knew the right place to look. But with the TwinSystem, a lot of system addresses have moved around. These new functions in BASIC give you a way to access some system cells without caring whether you're running on a Twin. If you plan to run your programs on an 8813 and on a Twin, or if you sell programs, this will become very important.

*** HARD DISK ANNOUNCEMENT ***

The new 10 Megabyte mass storage system has been formally announced, and deliveries are scheduled to begin this month.

The system may be ordered with only the Hard Disk, or with an 8" floppy included in the same cabinet. The wood cabinet resembles the standard 8813 and 88/MS cabinets. The new operating system, Exec/93 or Exec/90, is required to run the Hard Disk. These versions of Exec do not have any radical changes, and converting existing software to run on them should not be difficult.

There are several advantages in using a Hard Disk system. First, the amount of on-line storage increases tremendously (one Hard Disk holds as much as four [4] pairs of DSDD 8" drives). The time it takes to access the disk is reduced, improving disk-intensive operations like WordMaster and BASIC. The only real disadvantage is that it takes longer to back up a 10 MB disk; but the optional DSDD floppy simplifies this too.

*** MURPHY'S LAWS OF COMPUTERS ***

A program does what you told it to do, not what you thought you told it to do.

After a program has been running six months, the most fatal errors will begin to show up.

Rewriting the REMarks will cause the program to stop functioning.

Adding manpower to a late software project makes it later.

The system will be continuously revised to match the state of the art. No revisions will be noted in the documentation.

Any system that depends on human reliability is not reliable.

Any system designed to reject bad input data will be operated by an ingenious idiot who will find a way to get bad data past the traps.

Backup disks will be stored under refrigerators.

If you put it into memory, you will not remember where you put it.

Never test for an error condition you don't know how to handle.

The reliability of the system is inversely proportional to the importance of (A) the data passing thru it, and (B) the people watching it.

Flow-charting a program takes 50% of the time, writing it takes the other 50%, and debugging it takes the other 50%.

If builders built cities the way programmers write programs, then the first woodpecker that came along would destroy civilization.

Any given program will expand to fill all available memory, or until it outgrows the capability of the programmer who must maintain it. Or both.

*** 1980 POLYLETTER INDEX ***

We had five SUPER issues in 1980! A lot of the articles in them came from you, our readers. And thanks to Paul Hoffman and Charles Thompson for submitting information that was used in this index. It's organized as Issue/Page, for example, ASC is discussed in issue 5, page 2.

*** DISK-OF-THE-MONTH ***

This month we have a few surprises for you!

First of all, a SORT-DEMO program that demonstrates some of the many types of sorts. G. R. Gamble wrote it and it's in BASIC, so you can examine it and learn how to use the sort routines.

Also, a MAZE generator. We understand this one was supplied with Exec/83 in some cases, but if you haven't seen it before you will enjoy it!

Now, the surprises. Charles W. Gross has sent us a disk full of programs that don't work, and has generously donated them for the Disk-Of-The-Month! You can probably get these working with a little effort. He has READABILITY, which computes a measure of the difficulty of a passage of text; GENE, a geneaology (family tree) program; and Home-Inventory, which lets you catalog your personal goods. All are in BASIC.

The Disk-Of-The-Month is provided on a 5" SSSD diskette, with Exec/83 and BASIC C01L. Order from PolyLetter by sending a check for $15 to 1437 Sugarwood Lane, Norcross GA 30093.

*** LETTER-QUALITY FOR SALE ***

I have a Trendata 1000, which is based on the heavy-duty, IBM Selectric 72 I/O typewriter and is housed with its electronics in its own cabinet/table, for sale. The unit has given excellent service for the past 18 months, and has been serviced, on the rare occasions necessary, by the Trendata factory service system, which is present in major American cities, including Atlanta. It may be seen in Atlanta, up and running on an 8813. A printer driver for Wordmaster I is included. A driver for Wordmaster II is available from PolyLetter. Price: $800.00 (originally $1495). Call Stuart Woods at (404) 255-7260.

*** POLY ADS ***

Ads are published as a free service to PolyLetter subscribers.

FOR SALE: 8813 TwinSystem, 3 DSDD drives, 2 terminals, $5000. Call Mark Strong, Metro Com, (215) 985-0606

FOR SALE: 8813-2 with 88 MS SSDD and 56K. Good condition. Sacrifice at $6500. Computer Workshop, (816) 452-3690

FOR SALE: "Master Catalog" program. Establishes and maintains a data base of disk names and files. Requires 32K and 2 drives. $55. John J. Warkentin, 7041 Walnut Avenue, Orangevale CA 95662, (916) 988-8420

FOR SALE: "Little-Ada" compiler - Pascal-like language available for Poly. Price $50, or documentation only for $2. Ralph Kenyon, 145-103 S. Budding Avenue, Virginia Beach VA 23452, (804) 486-4370

*** BASIC SHORTHAND ***
by Frank Stearns, Linear Systems

Ever get tired of adding blank lines, clearing the screen for menus & new displays, etc...?

Instead of doing this:

```
10 REM initialize (DIM's etc...)
99 REM code starts here
100 PRINT CHR$(12)
110 PRINT "* XYZ Co. Data Entry *"
120 PRINT\PRINT\PRINT
130 INPUT "Enter Data: ",A$
```

Try some programmer's shorthand! Write the above as follows:

```
10 REM initialize program
20 V$=CHR$(12)\R$=CHR$(13)
99 REM code starts here
100 PRINT V$,R$,"       * XYZ Co.
    Data Entry *",R$,R$,R$
120 INPUT "Enter Data: ",A$
```

PRINT:(any file channel),R$,R$,R$ etc... works too!

Remember, you can never "wear out" R$ and V$, but you can save space and typing too!

*** COMMAND FILE INTERNALS ***

To our knowledge, the REF information associated with command files is not documented anywhere in Poly manuals. This makes it difficult for assembly language programmers to make use of command files anywhere. Here is a list of the command file associated REFS in Exec/83, along with my best guess of what they do.

CMDF is a one-byte flag that is zero if input is to be taken from the keyboard, and non-zero if a command file is in use.

CBUF is the start of 256 bytes (1 sector) of buffer. When a command file is in use, a sector at a time is read in here and used as input. You can jam characters in here and, by manipulating the flags, make the system think those characters are coming from a command file.

CMDP is a pointer to some character in CBUF, telling where the next command is to be taken from.

CMDN says how many sectors are left to read in the command file. If this is zero, no more disk reads will be done.

CMDA tells where the next disk sector of the command file will be read from (assuming CMDN is not zero).

CMDD is which disk drive to read from.

Following is a sample assembly program that will put two commands into the CBUF area of memory and execute them. Note that CBUF must contain zeros after the last valid character you put there, as this is how the disk files are arranged. This is accomplished by the "DS 256" at the end of the program, and the fact that we move a whole 256 bytes into CBUF regardless of how much we actually want to execute.

Bob Bybee

```
        REFS    SYSTEM
        REF     CMDF
        REF     CBUF
        REF     CMDP
        REF     CMDN
        REF     CMDA
        REF     CMDD

        REF     USER
        REF     Warm

        ORG     USER
        IDNT    $,$

        LXI     H,STUFF ;from here
        LXI     D,CBUF  ; to here
        LXI     B,-256  ; this many
        CALL    MOVE    ; will move
;(MOVE is in the 4.0 monitor)
MOVE    EQU     100H

        MVI     A,0FFH  ;make flag
        STA     CMDF    ;  non-Z

        LXI     H,CBUF  ;point at
        SHLD    CMDP    ; the start

        LXI     H,0     ;no more
        SHLD    CMDN    ; to read

        JMP     Warm

STUFF   DB      'L',13
        DB      'DISPLAY',13

        DS      256

        END
```

The purpose of PolyLetter is
to create a forum of
ideas for users of Poly equipment.
One year (six issues)
subscription, $15.
Editor: Bob Bybee
P o l y L e t t e r
1437 Sugarwood Lane
Norcross, GA 30093
(404) 925-2480
PolyLetter is not affiliated with
PolyMorphic Systems in any way.

PUT ME ON YOUR MAILING LIST:
Name _____
Business _____
Address _____
City/State _____ Z _____
Phone _____
System _____
Printer _____
Uses _____
Future uses _____

*** E x t r a P O L Y a t i o n s ***

By Ralph Kenyon

Greetings and welcome again to the assembly language tutorial corner. Today, I'm going to do a program which involves a very simple data file. I have written this program to use at boot time in order to insure the date on the disk is current.

The program looks for a data file called Date.DT on the system resident drive. It then displays the data in that file and asks if the date is ok. If you say no, it allows you to input the correct date and then writes that date back to disk.

This is a very simple matter in to program in BASIC:

```
10 ON ERROR PRINT CALL(1027) \ REM return to EXEC on error
20 DIM I$(1:1),D$(1:255) \ FILE:6,OPEN,"Date",INOUT
30 INPUT:6,D$ \ PRINT D$ \ INPUT "Is the date ok? :",I$
40 IF I$<>"Y" AND I$<>"y" THEN PRINT CALL(1027)
50 INPUT "What is the correct date? :",D$ \ D$="Date: "+D$
60 FILE:6,POS,1 \ WRITE:6,D$ \ FILE:6,CLOSE \ PRINT CALL(1027)
```

Let's see how we can do this in assembly language. Remember that I said that the comments were the most important part of assembly language programming (and other languages too) so you'd know what you did several months later.

```
        PAGE
; Program name: Today
; Program to change the date on entry
; Date last edited :  September 27, 1980
        REFS    SYSTEM.SY       ;This tells the assembler to open the
                                ;system library file called SYSTEM

;The REF pseudo-opcode tells the assembler to look up an
;item in the system library file which is currently open
        REF     WH0     ;System input "wormhole" (routine to input a
                        ;character from the keyboard)
        REF     WH1     ;System output "wormhole" (routine to output a
                        ;character to the screen)
        REF     Msg     ;System text writer routine which displays
                        ;text thru WH1
        REF     Look    ;Routine to look up a file
        REF     USER    ;Start of user memory
        REF     SYSRES  ;Place to keep track of which drive the system
                        ;currently resides on
        REF     NFDIR   ;Place to keep track of which directory
                        ;is currently in memory
        REF     Fold    ;Routine which converts to upper case
        REF     Err     ;System error processor
        REF     Rlwe    ;Routine to read a line of text into
                        ;a buffer with editing
        REF     Ioret   ;Routine to return from interrupt level
        REF     Dio     ;Routine to do disk input and output
        ORG     USER    ;This program will go into user memory
        IDNT    $,$     ;Load and Start at user memory too
;The routine Look is called with the drive number to search
;in A and with the address of the file name in HL
        LDA SYSRES      ;The file is supposed to be on sysres so load the
                        ;accumulator with the system drive number
        LXI H,Date      ;Date is the address of the file name preceeded
                        ;by the name length.  Look wants the size
```

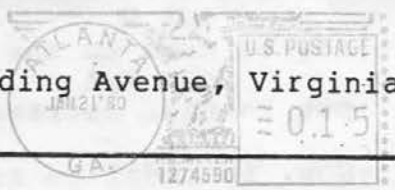
```
                        ;followed by the name and extension the same as
                        ;a file directory entry.

;Now that we've set up properly, call look to look up the file.
        CALL Look
        JC Err          ;Didn't find the file

;Look returns with the carry flag set if it has an error.  If Look
;returns an error, the error code is in DE.  Here we are just going
;to pass the error to Err for processing.  Since Err doesn't return, the
;program is aborted if there's any problem looking up the file Date.DT
;If there was no problem, look returns with the address of the file
;directory entry in DE. We 'fall thru' to here if there was no error.
;Since Look changes what's in A, we have to get back the drive number.
        LDA SYSRES
;Get the drive number and put it in C for Dio
        MOV C,A         ;For Dio
;(Dio requires the drive number to be in C)

;Now that Look found the file, we are going to read it from
;disk.  We have to set up to tell Dio how to do this.  We need
;to tell Dio where to get the stuff, how much, where to put it,
;and whether to read or write.

;Look returns the File Directory Entry address (FDE) in DE.  We're going
;to want to use the 16 bit addition (DAD) so we put that address in
;HL (we can also use MOV to/from Memory)

        XCHG            ;DE has FDE address so move it to HL

;The FDE is in SBUF1 which is the Directory buffer, but we want to get
;data which is past the name.  With one byte for the flags, 4 bytes
;for the name and two bytes for the extension, we need to move up to
;the 7th byte of the FDE.  We do this by using the 16 bit add (DAD)
;by putting 7 in DE and adding it to HL.
        MVI D,Ø
        MVI E,7         ;We know it's name is 4 long
        DAD D

;Now HL has the address of the first data which follows the name, which is
;the starting disk address of the file.  We want to get that data for Dio.
        MOV E,M         ;Get low byte
        INX H           ;Move up to the next one.
        MOV D,M         ;Get hi byte
        INX H           ;Move up to DNS
;The next piece of data in the FDE is the Number of sectors the file
;takes on disk (disk number of sectors or DNS)
        MVI A,1         ;1 sector
        CMP M           ;Supposed to be a 1 sector file
        RNZ
;If the file called Date we looked up does not have only one sector, then
;it's the wrong one so we just return control to Exec.

        INX H
        DCR A           ;Hi byte of size must be Ø
                        ;We had a 1 in A so this makes it a Ø
        CMP M           ;Supposed to be a 1 sector file
        RNZ
        INR A           ;Back to one sector

;We got to here because we found the file and it was only one sector
;long.  We need to setup for Dio.  Dio wants the disk address in HL so,
;we swap HL and DE again.  Remember that we got FDA into DE and it's
```

```
;still there.
        XCHG            ;FDA now in HL
;Dio wants the address of the buffer for reading and writing
;to be in register DE.
        LXI D,BUFF
;We also have to tell Dio which to do by a command in B
        MVI B,1         ;read
;What we are going to do now, is to use the stack for temporary data
;storage. One must be careful with this to make sure the number of
;pushes and pops balance, or the system will crash!

;(If the date has to be changed, we will need all if this data for Dio
;again when it comes time to write, so we will use the stack for a
;temporary storage space.)

        PUSH PSW        ;Save all these
        PUSH B          ;for writing later
        PUSH D
        PUSH H
;We have pushed the registers onto the stack in standard order.  Now,
;we can use the fact that Ioret pops them off the stack in reverse
;order to set up for returning.  Let's push a second copy of HL onto
;the stack, and then we get a the address of Ioret into HL.  After that,
;we swap the top of the stack (our HL) with HL to get HL back and to
;put the address of Ioret on the stack.
        PUSH    H
        LXI     H,Ioret
        XTHL
;When we get here, we have all the stuff for Dio in the
;registers and we have The address of Ioret on the stack.
        CALL    Dio
        JC      Err
;We pass errors directly to the system error handler Err.  If there
;were no errors, Dio will have gotten the one sector from disk and put
;it into the buffer.  Now let's show the operator what was in that buffer.

;Msg requires the address of the text to display in HL, and continues
;to print characters until it hits a zero.
        LXI H,BUFF      ;Print file contents
        CALL Msg
;Now, we've shown what's in the buffer (the old date)
;So, we ask if the date is ok by showing the prompt line.
        LXI H,Prompt
        CALL Msg

;After we've shown the prompt line, we must get the answer
        CALL WHØ        ;get the answer
        CALL WH1        ;echo back what we got
        CALL Fold       ;convert it to upper case
        CPI 'N'         ;see if it's wrong.
;Before we act on the comparison, we setup for it being ok by putting
;a carriage return into the accumulator.  If the operator types anything
;but an "N" or an "n", we will go to the system output wormhole with
;a carriage return.

;When WH1 returns, it will be to that address of Ioret which is
;on the stack, which will pop off 4 data items and return to
;Exec.  Result: back to Exec after printing a carriage return.
        MVI A,13
        JNZ WH1         ;No change
;If the operator types an n or an N then we get thru to here
;and will have to change the data in the file.  Start by
;telling em what we want em to do by printing the Input prompt.
```

```
        LXI D,Input
;Now we set up for Rlwe to do the actual inputting for us.  Rlwe
;requires the address of the text buffer in HL and the number of
;characters to read in C.  Also a flag is put in B to instruct Rlwe
;whether or not to echo the last character.  1 means do not echo.

        LXI H,BUFF+6
        LXI B,1F8H
        CALL Rlwe
;We got to here after the new date has been inputted to the buffer
;by Rlwe.  Rlwe also returns the number of characters inputted in B.
;In this case we want to know if the operator only put in one character
;(the carriage return) to use as a signal not to write the new date.

        DCR B           ;if B is one then no change
;so we jump to WHl and back to Exec
        JZ WHl          ;And return thru Ioret
;Since I use this date file for printing date headings on my papers,
;I want two blank lines after the date.  I am going to put two line
;feeds after the date.  Rlwe returns with the address of the last
;character (the carriage return) in HL, so, we can start moving without
;further setting up HL.  Rlwe also returns with the last character
;inputted in A (the final carriage return).

        MVI C,1Ø        ;Want to put in an LF
;Put the ASCII code for line feed temporarily in register C
;Then put it from C to Memory (replacing that carriage return
;that was read in last)
        MOV M,C         ;two line feeds
        INX H           ;before the CR
;Move up to the next one
        MOV M,C
;and put a line feed here too
        INX H
;now we move up one more and
        MOV M,A
;put in that carriage return which is still in A

;Now, in order to keep our data clean, we've got to sweep up the rest
;of the buffer by putting zeros in every location.  How many zeros?
;We need 256 zeros less the number of characters we have already used
;in the buffer. Since arithmetic on the 8Ø8Ø is Modulo 256, we can
;compute this in only one subtraction.  First, we get low byte of the
;address of the buffer into the accumulator and then subtract the
;address of the current character.

;Example: Suppose the buffer address was 324A and the pointer (HL)
;was at 326D.  By ANDing BUFF with ØFFH, we will end up with 4A in A.
;And with 326D in HL, L has 6D.  Subtracting 6D from 4A results in DD
;in the accunmulator.

        MVI A,BUFF AND ØFFH
        SUB L
        MOV B,A
        XRA A           ;Zero acc
;We put then number of bytes into B and clear A
;And then we loop around putting Ø in the rest of the buffer

LP      INX H           ;Got to fill up sector with Zeros
        MOV M,A
;Put the zero into M
        DCR B
;Count down with B
```

```
        JNZ LP
;If we haven't reached zero yet, we loop again.  When we do reach zero,
;we have swept the rest of the buffer clean of anything left and need to
;set up to write it out to disk.  Remember we stored all that data
;temporarily on the stack, so we can get it back.

        POP H           ;(Ioret)
;We don't need that Ioret address now, but we have to take it off the
;stack anyway.  Now we can get back the data.
        POP H
        POP D
        POP B
        POP PSW
;But we must change the command which was in B to Ø for writing.
;Every thing else is still the same as when we read the disk.

        MVI B,Ø         ;Write
        CALL Dio
        JC Err          ;Pass errors to Emsg
;This time we only need to have a Carriage Return to put the
;cursor on the next line.  We let WHl return to Exec for us.
       MVI A,13
       JMP WHl


Prompt  DB 'Is the date ok? ',Ø
Input   DB 13,'What is today''s date? :',Ø

Date    DB 4,'DateDT'   ;Size of name is 4, extension is DT

BUFF    EQU $           ;Here's where we read and write to from disk

        END
```

(You can write to Ralph at 145-103 South Budding Avenue, Virginia Beach, VA 23452, or call (804) 486-4370.)



PolyLetter

1437 Sugarwood Lane
Norcross, GA 30093
(404) 925-2480

Issue #81/2

PolyLetter

March/April

*** INTERVIEW WITH DONALD MOE ***

Donald Moe has been a Poly 88 user, an 8813 user, a Poly dealer, and is now head of Technical Support at Poly. We spoke to him recently to get some up-to-date news.

PolyLetter: We understand that Richard Eastman is no longer associated with PolyMorphic. What's the status on that, and who's in charge of the company these days?

Don Moe: The new president is Walter Kozinski. PolyMorphic is currently in the process of a merger with American Mini-Systems, a company working with Data General minicomputers. With this merger, and with the TwinSystem and the new Hard Disk, the future for PolyMorphic looks very bright. We're looking forward to a 16-bit project in the near future.

PL: What's the capacity of the Hard Disk, and how is it organized?

DM: The new operating system release, Exec/94, contains a "Volume Manager" structure which allows you to divide up the storage on the Hard Disk into manageable pieces. The Hard Disk can be made to look like 8 different drives. Its total capacity is 32,558 sectors. The same sub-directory system is still used, with up to three levels of sub-directories.

PL: What about backup?

DM: A system with a Hard Disk will always have at least one floppy; this is required since the system can only boot from a floppy. We recommend a DSDD 8" floppy for backup, although it is possible to use a 5" drive. The "new" bit in the directory is used to tell whether a file has been altered since the last backup, so the only files you back up are the ones that have been changed. Also, the largest file you can put on the Hard Disk is determined by the type of backup you are using... when a file is backed up, it must all fit onto a single diskette. If your backup device is a DSDD 8", the largest file you can have is 4924 sectors.

PL: You mentioned Exec/94. How has that changed from Exec/93 and Exec/90?

DM: Exec/94 contains the Volume Manager, so 94 is necessary to run the Hard Disk. We ship Exec/94 with the Hard Disk. Exec/94 also has an updated BASIC, with up to 16 file channels open at once. And, we've fixed a few glitches that have been found in Exec/93 and 90. But there are no other major differences. 93 and 90 are not obsolete by any means: in fact, people are still using Exec/83 and 76.

PL: The operating system has grown so much in recent releases, it won't all fit on a 5" SSSD floppy anymore.

DM: That's true. When we ship Exec/94 for a single density 5" system, it takes up two disks. But, all the files aren't needed for many applications. The new System Programmer's Guide tells you exactly which files are needed for BASIC, for Assembly, and so on. Then, you can "peel out" the ones that aren't needed on a particular disk, and save room that way.

PL: What else is in the new System Programmer's Guide?

DM: The new Guide has many more details on the system than earlier editions did. It's over 200 pages

(Continued on Page 2)

(INTERVIEW - continued from page 1)

long. This time, EVERYTHING is documented: all the data areas, system calls, etc. Plus, all the information pertaining to the TwinSystem is also included. The disk that comes with the Guide has 14 utilities on it: things like SCopy (Super Copy) which copies ANY file; RDB (Relocating DeBugger) for assembly programming; programs to compare files or whole disks, and to copy entire sub-directories; Recover, which aids in reconstructing a damaged directory. There are others, too. And, of course, the old Emedit, Szap, CLEAN, ARISE, and Auth. The new System Programmer's Guide is available from dealers or from PolyMorphic, and the price (with the disk) is $200.

PL: I'm sure you're aware that many Poly users have become dissatisfied... not with the equipment, but with the lack of support and communication from dealers and from PolyMorphic.

DM: Yes, we are aware of the problem. We need to get more dealers out there. Part of the problem is that we don't have a listing of end-users. I would appreciate it if your readers would drop me a post card, giving their name and address, type of system, and a brief description of their application. I don't plan to start an end-user newsletter anytime soon, but I would like to get a list of users so that we can get in touch if we have important technical information to distribute.

PL: If Poly users do need help, who can they turn to?

DM: I'm in charge of all Technical Support for Poly at the moment. Any questions on hardware or software should come to me, and I can refer them to one of our other people if necessary. I can't spend a lot of time on every question, but if someone needs help, It's my job to provide it.

(You can contact Donald Moe at PolyMorphic Systems, 460 Ward Drive, Santa Barbara CA 93111, (805) 967-0468.)

*** POLY USERS ON THE SOURCE ***

One way Poly People can feel more in touch with other users is to get on a time-sharing network, such as the Source. This is a nation-wide computer service that provides electronic mail, interactive communication ("chatting" to other users by typing on your terminal), access to UPI news, games, stocks, and a lot of other services.

These are the Poly people we know of on the Source. If you are on the Source or on MicroNet (a similar service), let us know about it!

CL1543: Bob Measle, a lawyer in Lexington, Kentucky.

TCB203: Frank Stearns, an audio engineer in Cheney, Washington ("one of the ash capitals of the world")

TCC609: Joe Toman and Illini MicroComputer Inc., Ogden, Illinois

TCC611: Jim Kaufman and Bob Zimmerman, Poly hardware hackers in Illinois (lots of homebrew hardware and software)

TCC870: Stuart Woods, a freelance writer in Atlanta

TCD125: Bob Bybee and PolyLetter

TCI127: Ralph Kenyon, ExtraPolyAtor in Virginia Beach, and valued contributor to PolyLetter

*** DON'T CLEAN THOSE DRIVES ***

Don Moe warns us that Shugart 8" disk drives (used in the 88/MS) can be damaged by "cleaning diskettes." Shugart has sent out a memo that using the "cleaning diskettes" can grind down the surface of the disk head, leading to failure. Check with your dealer, or with Poly, for more information.

The GAMES disk is coming!

*** BUGS IN EXEC/93 ***

Several PolyLetter readers have already installed Exec/93 in their systems, and they've run into a few bugs you should be aware of.

The new BASIC doesn't handle disk files well under certain conditions. Using <?> within BASIC is disastrous if the file does not exist: the system will seek forever! As Frank Stearns put it, "For fun, try a direct BASIC statement like 'FILE:4,OPEN, "<?>GARBAGE-FILENAME", INPUT' and watch it tickle your drives into the night. Nifty, huh?" Charles Thompson adds, "If you have three files, JUNK.BS, JUNK.TX, and JUNK.DT, and try to 'FILE:4,OPEN,"JUNK.BS",INOUT', you WILL get JUNK.DT; i.e., it disregards the extention."

The new FORMAT program is well suited to the newer letter-quality printers, such as Qumes and Spinwriters, but it won't do underlines on a Selectric-based printer. The new FORMAT underlines by carriage-returning and then underlining, which causes a blank line when printing with a Selectric.

*** PROCESSING BASIC ERRORS ***
by John J. Warkentin

Here is a trick you can use to improve your applications programs. Normally if an error occurs in a BASIC program, execution stops and you are returned to BASIC. But you can use the BASIC error message overlay, Berr.OV, to report the error and continue execution.

At the beginning of your program, put the line:

```
10 ON ERROR GOTO 10000
```

Then, add this subroutine:

```
10000 REM Error report generator
10010 L=LINE  \REM save line #
10020 E=ERR   \REM save error too
10030 PRINT "Error in line",L,
10040 Z=CALL("Berr",5,0,E,0)
10050 PRINT\PRINT\PRINT
10060 ...etc...
```

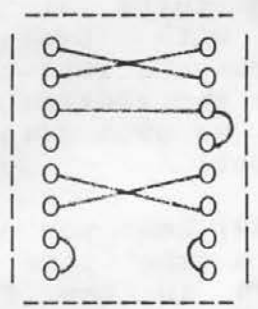
*** NEW POLY PRINTER DRIVER ***

The new operating system release, Exec/93, is supplied with a new version of the printer driver. Printer/42 is designed to interface easily with modern printers which include some kind of internal buffering. This includes printers such as the TI 810 and 820, and most letter quality printers. These printers use hardware handshaking to tell the Poly when to stop sending.

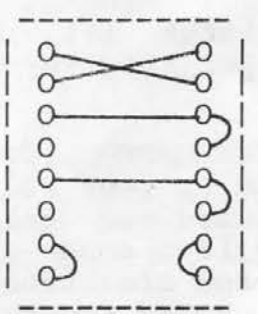
Unfortunately, Printer/42 will not just boot up and run. The new Exec is supplied with an instruction sheet that tells what must be done to your printer interface board (the small card on the inside back wall of the Poly). It involves a couple of minor wiring changes on the header plug, and possibly on the circuit board itself.

If you are comfortable with a soldering iron you will have no trouble following Poly's instruction sheet and making the changes yourself; if not, plan on having your dealer do it for you.

Patty Ryan sent us this note on what the new printer headers have to look like.



For Hardware Handshaking



For Non-Hardware Handshaking

*** SOFTWARE SHOP ***

Programs listed in the Software Shop are written by PolyLetter subscribers and friends. Your ads are welcomed! All software available only on 5" SSSD disks unless noted.

EXEC/90: A few copies are still available. Exec/90 has the new BASIC and WPS, the new WordMaster II menu-driven system. Also included are two new manuals, one for the System 88 and one for WordMaster II. Software is supplied on two SSSD 5" disks, direct from Poly (no second-generation copies). Hurry, they won't last! $75 from PolyLetter.

PRINTER SPOOLER: Lets your printer run while the Poly does other, more productive work for you. This program reserves some RAM as a printer buffer. Using 8K of memory and a 300 baud printer, you can save 5 minutes every time you get a printout. Works on Exec/83 and Printer/36 (included), with all standard printers. Written by Bob Bybee. Price: $54.68

DISASSEMBLER: Written by Ralph Kenyon. Takes a machine language program and turns it back into assembly, with System labels. Includes documentation and support programs for the incredible price of $12.50 (yes, we even give you a disk at that price!)

BASIC EDITOR: Lets you edit a BASIC program with the same flexibilty you've known in the Poly Editor. Move around lines of code, insert and delete characters, use the arrow keys to move the cursor thru your program lines. You'll wonder how you lived without it! Written by Bob Bybee especially for PolyLetter. $45.

DC HAYES MICROMODEM 100 TERMINAL: Make full use of your Poly with a DC Hayes modem card and this program. Includes file send and receive capability, and simultaneous output to printer. Ralph Kenyon wrote this and uses it to send his articles over the phone to PolyLetter. Available for Exec/78 or Exec/83 (specify which), for $85, and includes all necessary instructions.

COPYSELF DATADISKS: Special, minimum operating system designed for backup or data disks. Exec on this disk requires only 3 sectors! Written by Ralph Kenyon, an expert at customizing programs. Buy one at $15, and it will reproduce itself for you.

DISKS-OF-THE-MONTH

PolyLetter still has available some Disks-of-the-Month from previous

(Continued on page 5)

*** IDS PRINTER INTERFACE ***
by Ralph Kenyon
Abstract Systems, Etc.
(804) 486-4370

The Integral Data Systems (IDS) "Briter Writer" IP-125/225, with serial option, can be interfaced to the Poly 8813. Here's how to do it:

The IDS printer supplies EIA level CLEAR-TO-SEND to indicate to the Poly that the printer is ready for data. The CTS signal should be connected whenever data is supplied at speeds greater than 300 baud. This makes the IDS the controlling device. Additionally, the IDS does not send data back to the Poly so there is no need for a data return line.

Wiring the DIP header on the Poly printer card needs a data out path (pin 15 to pin 1), a CTS in path (pin 3 to pin 14) and a clock for the USART (pin 9 to pin 10).

For cable connections see page 17 of Poly's Printer Interface Manual and page 10 of the IDS Manual.

Cable Wiring:

IDS	Purpose	Poly
pin 3	Received data	pin 3
pin 5	Clear-to-send	pin 5
pin 7	Signal ground	pin 7
pin 1	Chassis Ground	pin 1

(SOFTWARE SHOP - continued)

issues. Each is priced at $15.

April 1980: Includes the popular CONTROL-U (lets you print out the contents of the screen at any time); COUNT (counts the number of words in a file); and CALENDAR (prints a calendar for [almost] any year).

August 1980: Includes the utilities Szap (for looking at and changing files ANYWHERE on a disk); POP (for making a system file non-system, or vice-versa); COPY-SUB-DIR (for copying all programs from one subdirectory to another); Cursor (changes the cursor to any character you like); and ROOM (tells how close you are to filling your disk directory).

January 1981: Includes SORT-DEMO (with many kinds of BASIC sorting routines); MAZE (a random maze generator with its own "digital rat"); and the BASIC programs GENE (a family tree program), READABILITY (computes the difficulty of text), and Home-Inventory (keeps your household property in line). These last three programs don't quite work, but we have confidence in your programming ability, so we're selling them anyway.

-- ORDERING INFORMATION --

Please complete the following:

Program	Price
_____	_____
_____	_____
_____	_____
_____	_____
Total:	_____

Name:_____
Address:_____
City, State:_____
Zip:_____ Phone:_____

System:_____ Drives:_____
Exec version:_____ Memory:_____

Send check to PolyLetter, 1437 Sugarwood Lane, Norcross GA 30093. All programs are copyrighted and are not for resale.

*** ANOTHER POLY GROUP LIVES ***

Pat and Roger Lewis, of "AAAA Computer How's" in Los Angeles, are still the leaders of a small group of Poly enthusiasts. We heard that they had disbanded, but a quick phone call to Pat confirms that they are still around... although suffering from the apathy toward Poly that has touched most of us at times.

P & R's group specializes in the Poly 88 cassette systems, as well as the 8813 disk systems. They have a good assortment of programs available and they are a dealer in Poly equipment, at discount prices. My own hybrid North Star/8813 got off the ground with their help. Get in touch with them at 13022 Psomas Way, Los Angeles, CA 90066 and let them know you're still into Poly.

*** POLY-ADS ***

Ads are published as a free service to PolyLetter subscribers.

FOR SALE: System 8813-2 with 88/MS SSDD, 56K. Good condition. Sacrifice at $6500. Computer Workshop Inc., (816) 452-3690.

FOR SALE: 8813-3, 3 SSSD drives, 48K, keyboard, 9" monitor, confidence package, system programmer manual and disk, business software, $4800. Diablo KSR printer, $1900. New 12" Hitachi Monitor, $200. New CPU and Video cards, $250 each. New printer and cassette cards, $50 each. Fine condition, must sell, take best offer. Bill Sullivan, 1137 Kingsbury Ave.., Birmingham AL 35213 (205) 879-8091.

FOR SALE: 8813-3, 48K, monitor, keyboard, printer interface. Diablo dot matrix 210 cps printer with tractor and friction feed. Will take best offer! Barbara Royal, 3513 Linda Circle, Des Moines, IA 50310 (515) 255-3203 nights, 243-2131 ext. 260 days.

The GAMES disk is coming!

WordMaster II -- INDEX

SYSTEM START-UP

WORDMASTER II: AN OVERVIEW OF OPERATING PROCEDURES

STORAGE MANAGEMENT FUNCTIONS

WRITE OR CHANGE A DOCUMENT

PRINTER FUNCTIONS

TYPE A DOCUMENT LITERALLY

FORMAT A DOCUMENT

FORMATTER COMMAND SUMMARY

*** DISK-OF-THE-MONTH ***

The infamous Disk is back again with some more goodies. Three handy programs were contributed by Russ Nobbs, of Rings and Things, Spokane, Washington:

TABBER prints a grid on your printer, to help in lining up output forms, columns of numbers, etc.

PEEK-DUMP dumps memory to the printer, in decimal. You can specify addresses in either decimal or hex, and the output is always decimal. ASCII (character) equivalents are also printed on the side of the page, similar to Szap's format.

FNTIMER contains functions to move the cursor anywhere (a "Print-At" function), and a routine to measure the execution time of part of your program. Find out where the program is wasting most of its time, and then you can optimize that part first.

This next jewel was donated by Jon Wolfert, of JAM Creative Productions in Dallas:

Tran.OV is an overlay you call from a BASIC program. It prints out numbers as words! Great for checks and probably many other applications. Included also are TRAN.DOC which contains the instructions, and TRAN-TEST, a simple BASIC program which gives you a working example of how to use Tran.OV.

The Disk-Of-The-Month is supplied on 5" SSSD, with BASIC CØlL and Exec/83. To order, send a check for $15 to PolyLetter.

*** COMPARING BASIC LOAD TIMES ***
by John J. Warkentin

I recently wrote a program in BASIC that got to be fairly large: 67 sectors of disk space. Its load time, naturally, was quite long. I did some experimenting and time comparisons, the results of which I'd like to share with my fellow Poly users.

First, I reduced the size of the program by stripping out all remarks. This brought the size from 67 to 46 sectors, and cut load time by 1/4.

I next used BASIC's SAVEF command to save a copy of each of the two versions. The resulting savings were significant, and impressive (to me, anyway). The SAVEF/full version now occupied 11 sectors less than it did in text form, and loaded in 1/3 the time. The SAVEF/stripped version occupied about 1/2 the disk space, and took about the same amount of time to load as the SAVEF/full version.

Version	Size	Load Time
	sectors	seconds
Original	67	33
Stripped (no REMs)	46	25
Original, SAVEF	56	13
Stripped, SAVEF	37	12

Apparently, 1/2 to 1/3 of the load time for a non-SAVEF file is spent converting that file to internal token format. The savings in space represented by removing the remarks depends on the programmer, but I don't feel that removing them is worth the effort. The time saved by SAVEF speaks for itself.

(The SAVEF savings should also hold true for a SAVEP'ed file.)

The purpose of PolyLetter is to create a forum of ideas for users of Poly equipment.
One year (six issues) subscription, $15.
Editor: Bob Bybee
P o l y L e t t e r
1437 Sugarwood Lane
Norcross, GA 30093
(404) 925-2480

PUT ME ON YOUR MAILING LIST:
Name _____
Business _____
Address _____
City/State _____ Z _____
Phone _____
System _____
Printer _____
Uses _____
Future uses _____

*** SQUEEZING BASIC ***

by Frank Stearns, Linear Systems
P.O. Box 261 Cheney, WA 99004
(509) 235-6019

How do you get a finished, debugged BASIC program to run faster and minimize RAM and disk space? Use the backslash ("\")! It separates discrete statements and allows multiple statements on the same line, each line may be up to 80 characters in length. There are cautions when one is using multiple statements per line -- more on that in a moment. (NOTE: Read "Optimizing Your BASIC Program" in Poly's BASIC manual. This two-page chapter contains important details not covered here.)

Each line number, in and of itself, takes two bytes of RAM. On disk this can be as many as five bytes. Remember that BASIC uses a 16 bit (two byte) machine address code whereas on disk this 16 bit address becomes a character string, each single character is 8 bits long. The line number "3020" is 16 bits in RAM and 32 bits on disk! Add 8 bits to each if you have a space after the line number. Then added to each line is a "count byte" and a "carriage return".

The line number, count byte, and carraige return of a discrete line can be traded for a single backslash. Of course, there must be a "host line" -- one where there is a line number, count-byte and carriage return -- but nonetheless the storage savings can be dramatic. I have several long programs which would not fit in my 32K machine were it not for the elimination of a lot of line numbers. In a typical "standard" BASIC program at least half the line numbers can be removed, sometimes more. I recently modified a big system version of "Star Trek" to run on Poly. Its original text length was 75 sectors. Its current length, NOT including savings from "fast load" (SAVEF) or "encrypted format" (SAVEP), is 50 sectors.

So how do we do this? First, get a program you know is running just the way it should. The single disadvantage of multiple statement per line programming is debugging. It is tedious following statements that have been mashed together.

We merrily chop away line number after line number, placing the old line contents on the line above using the "\" separator. We've packed each and every line up to the full 80 character limit. The program takes a third less space on disk. Then we try to run it and immediately get errors, the most common being: "I can't find that line". You're cursing me and crying "what happened?"

The tempering factor in multiple statements per line can be any conditional or non-returning branch statement, particularly the IF/THEN statement.

```
10 X=X+1*Q\IF X=A THEN 200\A(X,Q)=X
```

In this example, no matter what the result of the IF, the last statement, A(X,Q)=X, will never be executed. IF/THEN statements are "blind" and "literal." If the condition X=A is NOT true, that's it. The IF process stops immediately and drops to the next line number. And in this case, should the IF be true, we have an immediate jump, and again, the statement A(X,Q)=X is not processed. If A(X,Q)=X must be done each time X<>A, then it should be placed on a separate line as shown below.

```
10 X=X+1*Q\IF X=A THEN 200
20 A(X,Q)=X
```

Or we could use the "ELSE" feature of IF/THEN:

```
10 X=X+1*Q\IF X=A THEN 200 ELSE A(X,Q)=X
```
(Note: using "ELSE" will take slightly more time.)

On the other hand, this "literal" processing of the IF statement can work to our advantage. Let's try a new example:

```
100 IF X$="YES" THEN X=X+1\REM additional statements can go here
```

We can set up a conditional mini-subroutine right on the line thus saving "flagging" variables, convoluted logic, GOSUB/RETURN's, and GOTO's. Of course, we are limited to the 80 character line length. However, this can be expanded with the use of GOSUB.

```
100 IF X$="YES" THEN GOSUB 1000
110 REM execution returned here
```

But how do we avoid a branch-to line number that no longer exists and whose contents are now imbedded in the middle of another line? Well, we don't! But there is an easy testing routine to follow using the TEXT EDITOR when the BASIC program is being handled as text. Using this method we never have to be concerned about puzzling out logic and keeping program flow in our heads or on paper.

Assume while in the Text Editor (not BASIC) you have written the following segment:

```
3020 PRINT X,Y,Z
3030 A=A*(2^3)/A^3
3040 IF X=20 THEN 3200
```

and now wish to compact it to the following:

```
3020 PRINT X,Y,Z\A=A*(2^3)/A^3\IF X=20 THEN 3200
```

Before we place line 3030's statement up behind line 3020's as shown, move the cursor one line up and do a control-F (find text string). Enter the line number we plan to delete, in this case 3030, then hit ESCape. Immediately the cursor should drop back down a line and be just to the right of "3030". This small step is a simple check to be sure you have entered the search line number correctly. Do NOT delete the number at this time. Do a control-B (beginning of TEXT), then a control-C (search for next occurrence of string). If the cursor lands anywhere but behind the actual line number 3030, we CANNOT delete 3030. (Keep hitting control-C to make sure there is not a jump to this line number later in the program. Once at the end hit control-B, then control-C once. This should bring us immediately to the line numbered 3030. Now 3030 could be deleted and the its contents moved to the line above separated by the backslash.

But let's say earlier in the program at line 980 we had:

```
980 IF G=K2 THEN 3030
```

In this case 3030 must be left intact. We move to the next line number, 3040, and check for its occurrence in the same way. Control-F, string, ESCape; Control-B, Control-C.

After searching for more than one occurrence of 3040 and finding only one, the actual line number 3040, we finish compaction for this program segment as follows:

```
980 IF G=K2 THEN 3030
...pgm...
3020 PRINT X,Y,Z
3030 A=A*(2^3)/A^3\IF X=20 THEN 3200
```

There is one caution using the search-string facility of the editor. If the program text is large and all will not fit into memory you will have to make a note of what has been deleted, then do a control-O (half of RAM to disk), then a control-A (bring more from disk to RAM). Or, know that certain jumps will not take execution to opposite ends of the program that might not be in RAM during the edit session. Frequently if the program text won't all fit in memory set aside by the editor, it won't fit in memory set aside by BASIC either. You can come close, however. The editor automatically loads only half of available RAM. So when editing long programs hit control-A a few times before you start searching and deleting. Then do a control-E (move cursor to the end of text in RAM) to see if it's all there. (Or use the Wordmaster II Editor.)

I mentioned earlier the speed of execution factor. Each and every time the BASIC interpreter stumbles into a line number, the processor must do "housekeeping". Anytime you can eliminate a processor chore, or "overhead" as they say in the biz, you reduce execution time. This is of course very true in any heavily used part of the program such as the guts of a FOR/NEXT loop or a subroutine. With the newer versions of BASIC the time reduction results of compaction are more subtle but nonetheless still present. Once you've successfully compacted a program, run the original and the compacted version comparing times. Depending on the nature of the code, you should see a noticeable difference.

An additional method to reduce memory and disk space is to remove all embedded space characters. They are not needed.

For example:

```
10 PRINT\IF X=3 THEN 330
20 MAT A=0\GOSUB 5000\PRINT:4,C,E,F\GOTO 900
```

can be written as:

```
10PRINT\IFX=3THEN330
20MATA=0\GOSUB5000\PRINT:4,C,E,F\GOTO900
```

This is terribly difficult to read, but who cares? Once the program is perfected, the only one reading it will be your computer! (Note the embedded "GOSUB". The return WILL BE properly supported and brought to the file write statement, PRINT:4,C,E,F.)

There is one caution here: Beware of the BASIC commands that are two words: "ON ERROR" or "ON ESCAPE". The interpreter will search unsuccessfully for "ONESCAPE" or "ONERROR", thus generating "Syntax Error".

If you wish to use the global search and substitute feature of the editor to remove spaces, it is wise to use a keyword that is more than just a space -- embed a BASIC command in the keyword. For example, if we wish to remove all the spaces after PRINT statements we would enter the editor, make sure all the program is in RAM, and then strike the following

sequence:

(ESCape):^FPRINT ^[^D(ESCape)

If we wanted to get the space on both sides of the the command "PRINT", we could use a search and substitute sequence as follows:

(ESCape):^F PRINT ^[^W^DPRINT(ESCape)

One might think that using the SAVEF or SAVEP features does space removal for the user automatically. No, it does not. List a SAVEF program while in BASIC. You'll see all the spaces. When encoding, SAVEP appears to leave the SAVEF ASCII string format of the program intact and hence does not remove extra spaces. Also, using the same source program, SAVEF files are the same length as SAVEP files.

PolyLetter

1437 Sugarwood Lane
Norcross, GA 30093

Issue #81/3

PolyLetter

May/June

*** POLY MERGER CALLED OFF ***

The merger we mentioned in our last issue, between PolyMorphic and American Mini-Systems, has been called off. Walter Kozinski, president of PolyMorphic systems until recently, told us that "the necessary agreements could not be reached," so the deal fell through.

Shortly after the merger was cancelled, Mr. Kozinski also left PolyMorphic. He had been acting president of Poly for several months, and had formerly been associated with American Mini-Systems. Don Gallant is now acting president of PolyMorphic.

Now that Poly has released and shipped the Hard Disk (dubbed the 88/HD), their emphasis is on sales. Poly is going to be aiming toward a series of "vertical markets," where an applications package is designed for a particular type of business and then sold to similar businesses around the country.

The vertical market concept was taken because of a common problem among Poly dealers: due to a lack of direction, each dealer was forced to "do his own thing" and sell whatever packages seemed appropriate. No guidelines existed for determining what programs were actually reliable and profitable.

The first vertical market is a package for insurance agency management. This package was written in Santa Barbara. Other packages are expected to follow shortly, as they are written and tested. A package will be installed in at least six sites, and be operating for more than six months, before being considered for a vertical market sales package. A set of guidelines is being prepared by Poly to provide standardization between all vertical market packages.

Because of this orientation toward vertical applications, PolyMorphic will not be making an appearance at the NCC in Chicago this year. Len Danczyk explained that "the audience at the NCC is more oriented toward comparing hardware to hardware," and Poly's current efforts are toward the vertical market software.

We have a good assortment this issue, from many conributors all over the country. I'm constantly delighted at the fine quality of writing PolyLetter receives from our friends! And, let me remind everyone that we gladly accept articles or suggestions for articles, from any source. Your input is what makes PolyLetter shine.

PolyMorphic has been going through more shake-ups recently, with several key people leaving. (See articles on this page and inside.) It's difficult to know what to expect next from Poly, but every time I talk to them, they always have hopeful things to say. PolyLetter will be hanging in there to keep you informed.

In upcoming issues, look for articles on Super Zap (by Mark Sutherland), the Front Panel (by Russ Nobbs), the disk drives in your system (by Frank Stearns), and much more.

The GAMES disk has arrived! See page five for details.

Bob Bybee

*** ASSEMBLY LANGUAGE OP-CODES ***

If you've been keeping up with Ralph Kenyon's series of assembly language tutorials, you're well on your way to becoming an expert assembly programmer. But there are some useful opcodes you won't find in most 8080 programming guides. PolyLetter has compiled this special list for your convenience. (Note: Many of the instructions will only run if your computer is facing west, toward Santa Barbara.)

Disk Op-codes:

ED	Eject Disk
RDR	Reverse Disk Rotation
CWPN	Cover Write-Protect Notch
RID	Read Invalid Data
SNER	Seek Non-Existant Record
SRSD	Seek Record and Scar Disk
WBR	Write Blank Record
WFBU	Write Faulty Backup

I/O Op-codes:

CCS	Chinese Character Set
FSG	Fill Screen with Garbage
IC	Implode CRT
DK	Disconnect Keyboard
III	Inquire and Ignore Input
PSI	Print and Smear Ink
CHPR	Change Printer Ribbon
AFD	Accept Faulty Data
RFD	Refuse Faulty Data

Arithmetic and Logical Op-codes:

CRN	Convert to Roman Numerals
SRZ	Subtract and Reset to Zero
LAS	Load Accumulator Sideways
RSB	Reverse Some Bits
SD	Scatter Data
SMF	Set Most Flags
DSF	Don't Set Flags
CM	Circulate Memory
TDB	Transfer and Drop Bits

Control Op-codes:

TPO	Turn Power Off
TPN	Turn Power On
BTI	Blow Trumpet Immediately
NPN	No Program Needed
PO	Punch Operator
BSO	Branch if Sleepy Operator
BOP	Branch if Operator Present
BOI	Burn Out Indicator
BPO	Branch if Power Off
SPE	Slow Program Execution
HCF	Halt and Catch Fire

*** INCREASED BASIC PRECISION ***

The following code appears in the system overlay Bdir.OV. A simple change of one byte of the overlay, using Szap, allows increased precision of up to 62 digits in certain BASIC functions. This code resides at 2395H in the overlay area (in the fourth sector of Bdir.OV).

```
CD xxxx CALL  PARAM  ;get number
7B      MOV   A,E
FE Ø6   CPI   Ø6H    ;too small?
ØE 32   MVI   C,32H  ;C= error code
DA xxxx JC    ERROR  ;handle error
FE 1B   CPI   1BH    ;too large?
;change the bytes above to read
;"FE 3F" for increased precision.
ØE 33   MVI   C,33H  ;C= error code
D2 xxxx JNC   ERROR  ;handle error
;etc...
```

Not all of the math functions will work well at the increased precision. In particular, the trig functions will never be better than 26 digit precision because the constants built into the algorithms are not precise enough. The following operations have been used with success at 62 digits: +, -, *, /, INPUT, PRINT, PRINT:n, SQRT, ABS, SGN, loops, and comparisons.

(The preceding item was contributed to "Doctor Dobbs' Journal of Computer Calisthenics and Orthodontia," Menlo Park, CA. It was written by John W. McGaw of Las Vegas.)

*** POLY-ADS ***

Ads are published as a free service to PolyLetter subscribers.

WANTED: Low price letter quality printer, in good condition, for use with 8813. Prefer tractor feed. Buy or trade involving 60 cps DECWriter II. John McNally, 7049 Armstrong Road, Goleta CA 93117, (805) 968-4628 after 6 PM west coast time.

*** START/STOP BASIC DISPLAY ***

A well-known problem in BASIC is that it is hard to see a listing of the program, or output, as it screams past your eyes and off the top of the screen. By using this routine you will have a way to pause the output at any point, while listing the program or while it is running. The little-used "Backspace" key on the upper right of the keyboard is used to stop the screen, and any OTHER key will continue the program.

Line 10 builds a small machine code routine in a string (S$). Then the address of S$ is found in 20, and stored into the location called UVEC in 30 and 40. Finally, the "Backspace" character (with ASCII value 8) is stored into the location called UCHR in 50. When the 8813's keyboard handler hears the "Backspace" character and sees that it matches the contents of UCHR, it jumps to the routine whose address is in UVEC. Then our routine (in S$) is executed, which simply waits for another key and returns.

A small warning is in order: if you leave this stuff set up in UCHR and UVEC, and load another program, pressing the "Backspace" key may bomb you into the front panel. If this happens you can TRY to recover by typing SPJ403G. It is better to avoid this problem by disabling UCHR before you execute any other program: do this by POKE 11673,0. You should always do this even if you are going to execute another BASIC program, since the variables (like S$) may not be in the same memory locations after another program is loaded.

```
10 S$=CHR$(33)+CHR$(64)+CHR$(0)+
   CHR$(229)+CHR$(251)+CHR$(195)+
   CHR$(161)+CHR$(3)
20 M=MEM(S$)
30 POKE 11669,M-INT(M/256)*256
40 POKE 11670,M/256
50 POKE 11673,8
60 REM Now let's produce some nice
   output so we can start & stop it.
70 FOR I=1 TO 100
80 PRINT I,SQRT(I)
90 NEXT
```

*** DONALD MOE DEPARTS ***

Donald Moe, head of Technical Support at PolyMorphic Systems, has accepted a position with a company in West Germany. He has resigned from Poly, and will be leaving the U. S. around the end of May. Don will be working with an engineering consulting firm in Munich, dealing with the German automotive industry as well as other industries.

We are all sorry to see Don leaving the Poly arena. His contributions were greatly appreciated by everyone who spoke to him. PolyLetter received many letters commenting on Don's helpfulness. We at PolyLetter join all of you in wishing Donald Moe the best of luck in his new position overseas.

Brian Smith, head of the Poly software department, has also left PolyMorphic. Brian is taking a position locally, in Santa Barbara.

Poly users who have hardware or software problems should contact either Len Danczyk or Frank Anderson. They are currently handling Technical Support for Poly. Their number is (805) 967-0468.

The purpose of PolyLetter is
to create a forum of
ideas for users of Poly equipment.
One year (six issues)
subscription, $15.
Editor: Bob Bybee
P o l y L e t t e r
1437 Sugarwood Lane
Norcross, GA 30093
(404) 925-2480
PolyLetter is not affiliated with
PolyMorphic Systems in any way.

PUT ME ON YOUR MAILING LIST:

Name _____

Business _____

Address _____

City/State _____ Z ____

Phone _____

System _____

Printer _____

Uses _____

Future uses _____

*** DETAILS FROM DALLAS ***
by Charles A. Thompson

RECOVERING FROM A DISASTER. One day you may find that something has gone wrong and you can't do anything in the Editor (it has "locked up"). You need not lose your work. There are at least two solutions -- one is easy.

In later versions of Exec and the Editor, after the Editor hangs up, merely press the LOAD button ("boot" the system). The material you were working on is still in memory, so:

```
$ENABLE
$$GET Edit
$$REENTER
```

Then, enter ESC CTRL-E. The file will be saved to the disk and you can start all over again with your file intact.

The other method is more difficult. Again, boot the system by pressing the LOAD button. Then:

```
$SAVE
From address = 3200
Start address = 0
Load address = 0
# of sectors (1 - 7F) = 7F
File name (1 to 31 characters) =
                        <d>filename
```

Repeat this for the next 7FH and you will have saved all of user memory. Use 8000 (hex) as the "From address" on the second pass. You can then use the Editor to delete extraneous material, merge files, etc., until you have reconstructed your file. 7FH = 127 decimal, so you will need a disk with at least 254 available sectors. Note that in Exec/93, you could use 5500 as a from address instead of 3200, but this may change from one Editor version to another. The safest method is to save all of user memory.

WordMaster II documention is generally complete and well done but here are two significant exceptions.

LINKING FILES. For reasons unclear even at Poly, the {nxt} command was omitted from the WordMaster II manual. It's really useful to link numerous files together for formatting. Syntax is:

```
{nxt <d<nextfilename}
```

and whatever file specified will be automatically loaded and formatted. If you want a new page, be sure {bpg} comes before {nxt}. With {nxt} at the end of each part of your long document, you need only give FORMAT the name of the first, and the rest is automatic.

DIRECT KEYBOARD ENTRY. The <K command is available to do direct keyboard entry into FORMAT. You can enter any format command or text that you could include in a text file. For example,

```
$FORMAT
Give the name of a text file
                    to format: <K
-->{pgstart 24}{pgstop 26}
   {nxt <2<MANUAL}
<2<MANUAL
```

If you want to insert text by hand, it's a bit tedious, but can be done. First, in the file to be formatted, put {nxt} (nothing more) where you want to insert the material, and then end that file at that point. You will get the "Enter name of file to format:" prompt, then enter <K. You'll get the right-pointing arrow and can then enter text or commands. If, for example, you have a letter and want to insert the inside address and salutation, you can do so. You should switch to no-fill (type in {nfil} and RETURN) or end each individual line with {br}. After you have finished your direct entry, enter {nxt <d<filename} where d is the drive number (or ? or #) and filename is the name of the next part of your document. FORMAT will begin to run automatically again.

Include at the top of the continuation file any format commands needed to insure correct formatting after you've been playing with the keyboard. This procedure

(DALLAS - continued from page 4)

will work whether you use WPS (WordMaster II) or use FORMAT directly.

BYPASSING THE PRINTER DRIVER. You may want to send ASCII codes to a printer which you cannot get past the printer-driver (sending just a carriage return with no line feed is an example). It's easy. Just use:

Z=CALL(12288,X,256)

where X is the ASCII code for the character you want to send. Repeat the CALL for each character.

My "Addendum to the Poly Manuals" is now in the 3rd Edition and more than 40 pages long. It includes helpful hints, descriptions of various utility programs, and undocumented Poly capabilities such as those above. It even has information about Exec/94 and BASIC C03 (which allows 16 open data files at any one time, and has other enhancements!)

When the Addendum was 16 pages long, I asked $4.00 to help defray printing and postage costs, but now I must ask for $6.00. Believe me, this is still a "non-profit" venture. My new address is:

Charles A. Thompson, Attorney
2909 Rosedale Avenue
Dallas, Texas 75205

*** ABOUT YOUR SUBSCRIPTION ***

We maintain the PolyLetter subscription list on my home-brew system (part PolyMorphic and part North Star), using a mail list program written by Mark Sutherland. On your mailing label, in addition to the address, you'll see a number like "8203 331". The four-digit number tells when your subscription expires, giving the year and the issue number of the LAST issue you have paid for. The three-digit number is our file number for you. When you renew, or if you have any reason to write about your subscription, it would be helpful if you could copy down these numbers.

*** DISK-OF-THE-MONTH ***

The Disk-Of-The-Month this time is the GAMES disk we've been teasing you about!

PolyLetter received a lot of response to the idea of a GAMES disk, and many people sent in their favorite games to be included. Response was so good that this will be the first of MANY disks containing games. Here are the offerings this month:

BIORHYTHM prints a chart of curves predicting your Physical, Emotional and Intellectual well-being.

BATTLESHIP is the classic encounter between fleets, except that this time the computer tries to sink you.

CHESS plays a darned good game (well, it beat me twice, for whatever that's worth). Good use of Poly graphics.

HANGMAN tests your ability to spell (and your nerves, as the Hangman has his itchy finger on the gallows rope).

SPIRAL and ART are graphics demos that do pretty things to the Poly screen.

BATTLESHIP and BIORHYTHM were converted from North Star BASIC to run on the Poly. (We added some screen graphics to BATTLESHIP to take advantage of Poly's abilities.) CHESS is a demo program compiled by Micro Applications' FORTRAN. SPIRAL was written by Ralph Kenyon, ART by PolyLetter. All are in BASIC except CHESS, which is in machine code.

This fun-filled diskette can be yours at the usual low Disk-Of-The-Month price of only $15. The Disk includes Exec/83 and BASIC C01L. Disks-Of-The-Month are only available on 5" SSSD. Order from PolyLetter, 1437 Sugarwood Lane, Norcross GA 30093.

"PolyMorphic: some say it means 'a dead parrot.'" --Bob Martin, 1979

*** FAST SORTING ***
by Bill Sullivan

Here is a sorting routine that makes use of the MIN, MAX functions in BASIC. It's most useful if you want to sort a small list of elements, say, less than 200 items. This routine prompts you for a number of elements, and then sets up a list of random numbers to be sorted. A timer is included to show how long the sort took. The actual sorting is done in one line, line 60.

```
10 INPUT "Number of elements: ",S
20 DIM A(S),B(S)
30 MAT A=RND(2000)-1000
40 PRINT %#4I \MAT PRINT A,
50 PRINT \T=TIME(0)  \L=MIN(A)-1
60 FOR X=1 TO S \B(X)=MAX(A)
   \A(#)=L \NEXT
70 T=TIME(1)
80 MAT PRINT B, \PRINT \PRINT
90 PRINT %6F3,T/60," seconds."
```

(Editor's note: we use a similar system to sort the PolyLetter mail list, and it works great! --BB)

*** EDITOR LIBRARY FUNCTIONS ***
by Russ Nobbs

The ability of Edit 2.0 and Edit 3.0 (Exec's 90 & 93) to save and load libraries of key definitions goes well beyond the TXdef.ED and BSdef.ED that load automatically when editing files with the extensions .TX and .BS.

The editor will load ANY library file with a format that matches the extension of the file to be edited. Say, for example, you are editing some program documentation text files. If you gave them the extension .DC and created a library file of key assignments named DCdef.ED your keys would be ready to use each time you edited the file.

If you have Ralph Kenyon's DisAssembler (which creates source code files with .AS extensions) you would use ASdef.ED for the key assignment library you create for assembly language purposes.

You can create your own "def" files. When in the editor, define the keys you want to store in a file using the ESC = sequence. Use ESC ? to check that all your definitions are as you want. Then, hit ESC Ctrl-W (for WRITE), and enter a file name such as those suggested above.

If you're editing a file, and realize that the appropriate library of ESC keys is not in memory, you can hit ESC Ctrl-L (for LOAD), and the editor will load whatever file you name.

Remember, these tricks only work on Edit version 2.0 and later versions.

*** YES OR NO? ***
by Jonathan Wolfert
JAM Productions
Dallas, Texas

If you have a BASIC program that asks the user a lot of questions, here are a couple of routines that can speed things up.

For questions that are answered yes or no, you can define a function (we'll call it A1 here) that will work as follows:

```
200 PRINT "Answer yes or no...",
210 X=INP(1) \Y=FNA1(X) \
    ON Y GOTO 210,250,300
250 PRINT "You said YES."
      .
      .
300 PRINT "You said NO."
      .
      .
500 DEF FNA1(Z)
510 IF (Z=89) OR (Z=121) THEN Z=2
    \GOTO 540
520 IF (Z=78) OR (Z=110) THEN Z=3
    \GOTO 540
530 Z=1
540 RETURN Z \FNEND
```

You can refer to this function throughout the program. It lets you answer a yes/no question by simply touching the Y or N key; you don't

(continued on page 7)

*** GREEK ON THE POLY SCREEN ***

You may know that the VTI (Video Terminal Interface) used in System 88's is capable of producing Greek characters. Here is a BASIC subroutine that allows you to print, easily, any of the available Greek characters.

FNG$ is the name of our function. It converts any lower case letters in a string to Greek characters, then prints the string. You can mix normal (upper case) characters in the string and FNG$ will print them unharmed. (A few special characters are used besides lower case, such as the underline character.)

The function is defined in lines 20 thru 70 below. The lines which follow demonstrate some of the uses. First a string of all lower case letters is printed, then printed again by FNG$ (this shows which lower case letters produce which Greek letters). Then a number of examples are displayed, using formulas from an old physics book I found.

It isn't immediately obvious from the program listing, but the results are quite pleasant if you need some math or scientific symbols on your screen. Enjoy!

```
10 DIM X$(1:64),A$(1:64)
20 DEF FNG$(X$)
30 FOR Z9=1 TO LEN(X$)
40 X9=ASC(X$,Z9)
50 IF X9=95 THEN PRINT CHR$(159),
   \GOTO 70
60 IF X9>95 THEN PRINT CHR$(X9+32),
   ELSE PRINT CHR$(X9),
70 NEXT\RETURN ""\FNEND
80 A$="_`abcdefghijklmnopqrstuv
   wxyz{|}~"
90 PRINT A$
100 PRINT FNG$(A$)
110 PRINT\PRINT FNG$("o =),PI,
120 PRINT FNG$("       X =
    p COS(g), Y = p SIN(g)")
130 PRINT\PRINT FNG$("C1 IS
    500 kF AT 50 VOLTS"),
140 PRINT FNG$("      R3 IS
    820x, 1/2 WATT, 1%")
150 PRINT\PRINT "INDUCTIVE
    REACTANCE = j",FNG$("wL"),
160 PRINT FNG$("      y2 ="),
    SQRT(2)
170 PRINT\PRINT "E=h",FNG$
    ("l        C = Fj")
180 PRINT\PRINT FNG$
    ("       z IMPORTANT! {")
190 PRINT\PRINT FNG$("u = ~(1/r)"),
200 PRINT FNG$
    ("        TAN(v) = `/a")
```

(YES/NO - continued from page 6)

even have to hit RETURN. If you hit anything other than Y or N (upper or lower case) nothing happens... so you can't answer incorrectly. FNA1 returns the value 2 for Yes, 3 for No, 1 for anything else.

A variation on this idea follows, for situations where you present the user with a number of choices, and expect the answer as a single digit (1 thru 9). We define FNA2:

```
100 PRINT "Select..." \PRINT
110 PRINT "  1 - Do something"
120 PRINT "  2 - Do something else"
130 PRINT "  3 - Do a third"
140 PRINT \PRINT "Number?  "
150 X=INP(1) \Y=FNA2(X,3)
    \ON Y+1 GOTO 150,200,250,300
    .
    .
200 PRINT "Choice 1 here."
    .
    .
250 etc.
    .
    .
500 DEF FNA2(Z1,Z2)
510 IF (Z1<49) OR (Z1>48+Z2)
     THEN Z1=0 ELSE Z1=Z1-48
520 RETURN Z1 \FNEND
```

The nice thing here is that the second parameter (Z2) sets the upper limit for acceptable answers, 3 in this case. FNA2 returns a value from 1 to Z2 if the appropriate key was hit, otherwise it returns 0. The "ON Y+1 GOTO" statement causes the program to wait for a valid entry.

Ralph Kenyon will be moving shortly, and to help pay for the U-Haul, he has increased the price of the DisAssembler to $25.00 (still a bargain!). It's available from PolyLetter.

*** E x t r a P o l y A t i o n s ***

by Ralph Kenyon, ExtraPolyAtor
Abstract Systems, Etc.

The reader response to the assembly language tutorial was very gratifying. Several people have called and written to say they were able to get the program to run for them.

We heard from Norm Shimmel in Butler, Pennsylvania, who has an 8813. Norm wants to see articles about flags and about writing routines to handle real numbers. Norm... You got one coming up.

Jonathan Wolfert of Dallas Texas, who uses his Poly in a radio commercial production buisness, wants to know about the DEFS, and DEF pseudo-ops as well as what else is in the SYSTEM library file. Jonathan also asked about programs that access disk files. The January issue of PolyLetter had a simple program which does disk access by keeping a date file. I will do a program and an article to keep track of more than one item in a data file in a future article.

Let's answer one of these right here -- The pseudo-opcodes DEFS, and DEF. The pseudo-opcode DEFS is very much like the REFS. REFS opens an existing library file for the assembler to get its input from. DEFS opens a new file which does not exist yet. DEF is the counter-part of REF. REF looks up an item in the library file, while DEF puts one into the open output file. To put this in a table:

MODE	OPEN	ACCESS	
INPUT	REFS	REF	(lookup)
OUTPUT	DEFS	DEF	(enter)

The library file can store labels as well as macros.

Let's go thru saving a macro for a frequently used subroutine. The subroutine I frequently use finds the negative (two's complement) of the contents of register DE, and I call it MinDE (minus DE). The negative of a number in 8080 language is the two's complement form. The two's complement form is computed by first finding the complement of the number and then adding 1.

Example of find the two's complement of 1 (compute -1)

```
In 16 bit binary 1 is:  0000 0000 0000 0001
The complement is:      1111 1111 1111 1110
Adding 1 gives:         1111 1111 1111 1111
```

Note: if we add 1 and -1 we should get 0.

```
    0000 0000 0000 0001
+   1111 1111 1111 1111
    -------------------
1   0000 0000 0000 0000
```

The lone 1 carries out and leaves 0, which is the desired result.

To form the negative, we must complement the number and add 1. In assembly language programming, this may be a very common requirement, so we may want to keep a library routine so we don't have to re-write it every time. The routine with documentation comments looks like:

```
MinDE   PUSH    PSW     ;Save flags and accumulator
        MOV     A,E     ;First we complement the
        CMA             ;high byte
        MOV     E,A
        MOV     A,D     ;Then we complement the
        CMA             ;lo byte
        MOV     D,A
        INX     D       ;Then we add 1
        POP     PSW
        RET
```

Here's how we put that into a library. We must put the routine into a MACRO. Since every character takes up space in the library file, we will want to minimize the comments in the macro.

```
        MINDE   MACRO
#L      PUSH    PSW
        MOV     A,E
        CMA
        MOV     E,A
        MOV     A,D
        CMA
        MOV     D,A
        INX     D
        POP     PSW
        RET
        ENDM
```

MACRO is a pop (pop stands for pseudo-opcode) which creates a master copy in the assembler data. Once a MACRO is defined (as above), it can be re-copied by putting its name in the opcode field thus:

```
Label   MINDE
```

The #L is an instruction to the assembler to use the label on macro in that place.

Macros are "expanded" at execution time. Here we have two examples of calls to the macro. The following text would be written in your source list:

```
MinDE   MINDE           ;first call

DEneg   MINDE           ;second call
```

When you assembled it and asked for a full listing, the following would appear:

```
MinDE   MINDE           ;first call
MinDE   PUSH    PSW
        MOV     A,E
        CMA
        MOV     E,A
        MOV     A,D
        CMA
        MOV     D,A
        INX     D
        POP     PSW
        RET
```

```
DEneg   MINDE           ;second call
DEneg   PUSH    PSW
        MOV     A,E
         .
         .
        etc.
```

Now, we want to save this macro so we don't have to rewrite it for each program. We must open an output library file with the DEFS pop.

```
        DEFS    MY-LIBRARY
```

Then we tell the assembler to save the macro MINDE with the DEF pop:

```
        DEF     MINDE
```

Later, when we write another progrm, we need only use the following code:

```
        REFS    MY-LIBRARY
        REF     MINDE

MinusDE MINDE   ;Remember to call this routine

        END
```

You can put all your frequently used routines into a macro library. It saves re-inventing the wheel each time you write a program.

If any of you out there have questions about assembly language programming, how to do something, or any of the "why did it do that" kind, write or call and we'll try to find out an answer. PS: Is there anyone out there interested in FORTH?

(You can contact Ralph at 145-103 S Budding Ave, Virginia Beach, VA 23452, (804) 486-4370).

Issue # 81/4

PolyLetter

July/August

This month's news from Poly is quite encouraging. Ken Gudis, PolyMorphic's Marketing Manager, says sales have been increasing steadily over the past several months, and Poly is now showing a profit again.

Ken attributes much of this success to the two most recent Poly innovations, the Hard Disk and CP/M. The CP/M offering has caused many people to take a more serious look at Poly. And the fact that you can run either Poly's operating system or CP/M makes it even more attractive... converting to CP/M does not mean you lose the use of your current Poly programs. Conversion to CP/M is a simple process, which can be done either by a dealer or at the factory. (Ken says most of the conversions so far have been done at Poly's factory.)

Poly's CP/M package includes the CP/M editor, assembler, the standard CP/M utilities (including PIP, SUBMIT, DDT and others), and PCOPY (a program to transfer files from CP/M to Poly format and vice versa). A CP/M version of BASIC is not included, but many software vendors offer CP/M BASIC.

The cost is $250 for the CP/M royalties, manuals, and disks. Add $50 for the hardware conversion costs, if Poly does it for you (it involves changes to the CPU, video, and memory cards). And, since CP/M requires 56K, you may have to add another $300 (approximately) for a memory card if your system doesn't have enough memory already. You must also have one or more 5" SSSD drives. Total cost for adding CP/M capability: $300 and up.

Many CP/M systems run 8" disks, but Ken Gudis says that Poly's 5" CP/M has been well received. 5" disks are used by several other CP/M systems, including North Star and Osborne. Ken admits that the 16 x 64 video screen has been a slight drawback since many CP/M systems use a 24 x 80 screen.

PolyLetter continues to grow! This issue we are happy to welcome the first of our overseas subscribers: 1999 Business Systems Ltd. and Biswanger Enterprises Ltd., both in Alberta, Canada; and ELKA-TV and Poly-Data, both in Denmark. We are happy to have you among us.

I've been asked to write a little about the people who read PolyLetter. We currently have about 170 paid subscribers. 70% list a business name on their subcription, and most of the readers I've talked to use their Poly in conjunction with their businesses. The renewal rate has been very satisfying: nearly all of the renewal notices we've sent out have been answered, most with glowing letters about how PolyLetter is a big help. And, we're gaining new subscribers as people find out about us. We get about 5 to 10 new subscribers each month. PolyMorphic has helped us in this regard, by recommending us when their customers ask about user groups.

This month's feature article is by Russ Nobbs, exploring the mysteries of the Front Panel. In future issues, look for more assembly tutorials by Ralph Kenyon, as well as articles by many of our other regular contributors. We try to provide information for the beginning Poly user, as well as the advanced systems programmer. If you have a Poly-related question, pass it on to PolyLetter and we'll try to answer it in an upcoming issue.

Bob Bybee

Looking toward the future, Poly is considering an operating system called OASIS. This operating system would bring many of the same benefits that CP/M has, such as compatibility with other systems and a wide range of software available. It could also turn the Poly into a multi-user system, beyond the present TwinSystem capability. However, OASIS or a similar

(Continued on page 2)

(Continued from page 1)

system would not be available anytime soon. For compatibility with other hardware and software, CP/M is the way to go for the near future.

Poly's Hard Disk is also selling well. With its 10 megabyte capacity, the 88/HD is appealing to dealers who are currently selling large applications programs. A larger capacity 88/HD is expected within a few months, with storage of 35 megabytes.

The 88/HD runs with Exec/94, which includes Volume Manager. Lew Gaiter is the new Applications Software Manager at Poly, and he described some of the abilities of Volume Manager: "With Exec/93 and earlier, you're locked into device numbers. Each drive number from 1 to 7 refers to a specific, fixed, device. Volume Manager lets you change these numbers, by associating a 'logical device' number with a 'physical device' number, or volume number. The 'physical devices' are the old disk drive numbers, 1 to 7, plus the HD. The 'logical devices' are numbers you pick, from 1 to 7, and you decide which number goes to which device. For example, if you had a command file which copied files from drive 1 to drive 2, you could tell the system that these drive numbers correspond to 5" drives, MS drives, or volumes of the Hard Disk.

"Also important, the Volume Manager lets you disconnect a device when you're done with it. Under Exec/94, a single-user system can access up to 7 devices, and a Twin can access up to 9. Volume Manager lets you connect any device number to any drive, use it, and release it after you're through. In this way, a system with floppies and a HD can use just the devices it needs for a particular job."

Lew says the Volume Manager is described in detail in the new System Programmer's Guide, and this information will be incorporated into the regular User's Manual as soon as possible.

Poly's advertising is currently being directed toward small trade journals, aimed at customers for specific applications packages. Ken Gudis explained that "this approach produces more serious inquiries, and more sales, than the broad-range kind of advertising found in larger magazines such as Business Week." The vertical market software packages we mentioned in our last issue are still under development, and Ken says that questionnaires will be going out to dealers soon as part of this effort. Ken commented that Poly's dealers are doing a good job, and mentioned that the foreign dealers are especially aggressive.

*** SUPERPILOT ***

by Tim Scully
Mendocino Microcomputers, Inc.

SUPERPILOT, an extended version of the language PILOT, is available for the Poly 8813. This language is designed to work with string variables and is excellent for creating instructional programs in which the computer teaches the user how to operate the system. It is easy for non-programmers to write instructional software in SUPERPILOT.

SUPERPILOT allows you to call subroutines, jump to labels, read and write disk files, manipulate string variables (whose names can be any length), accept user input and match it against a set of data, do conditional tests, and perform limited numerical computations. If you already have instructional material on disk as a text file, it wouldn't be hard to convert your file into a PILOT program.

Other applications for SUPERPILOT would include tests, questionairres, and general educational uses.

Here is a sample PILOT program using a few of the core instructions: T: (type), A: (accept), M: (match), $NAME (a string variable name), E: (end), and J: (jump).

```
T:Hi.  What's your name?
A:$NAME
T:Glad to meet you, $NAME.  Would
   you like to learn about PILOT?
A:
M:yes,ok,sure,fine
TY:I'm happy to hear that, $NAME.
    I think you'll enjoy PILOT.
JY:*mainprogram
TN:Well, $NAME, it was nice meeting
    you anyway.  Bye.
E:
*mainprogram T: PILOT is... (etc.)
```

For more information on SUPERPILOT, write to Mendocino Microcomputers Inc., 32191 Albion Ridge Road, Albion, CA 95410 or call (707) 937-5001.

*** WORDMASTER II HINT ***

by R. Peter Jackson
Mariposa Corporation

When doing large numbers of short letters, it often becomes a chore to keep re-entering the Format line on the WordMaster II menu. This problem can be solved by creating a file (using WordMaster) in the normal way. You might call the file "NEXT." Type the following into the file:

{bpg}

Be sure to hit the RETURN key after the {bpg} command and then hit ESC Ctrl-E to close the file and write it onto the disk.

Next time you need to print a series of rough draft letters or other text, or if you want to print final copies, use the Format line of the WordMaster menu. After selecting a file to print with the ESC S command, select the NEXT file with the same ESC S command. Then select the next file you want to print, then select the NEXT file again. Continue this process until you have selected all of the files you want to print. Then hit ESC Ctrl-E and the Format program will load. Load the appropriate paper into the printer and push any key to begin printing.

If your letters are more than one page in length, remember to use {rpgn} at the beginning of the letter so that the page numbers will be correct. Also remember to cancel headers when shifting from letter to letter, using {he}{end} at the end of a letter.

*** MORE ON CLEANING MS DRIVES ***

We mentioned in issue 81/2 that Poly had been warned against using "cleaning diskettes" in MS drives. Poly now says that the Innovative Computer Products FD-08 and FD-05 cleaning kits have been approved by Shugart, and are okay to use (following label directions). Contact Poly for more information.

*** POLY - TALK ***

This column is a direct line between those of you who have questions, and anyone who can provide an answer.

I am interested in obtaining software for business expense accounting. I need a program that will allow me to nominate the categories and will provide running totals of expenses. Robert J. Penney, PO Box 363, Casper, Wyoming 82602.

Does anyone have experience with using a parallel printer (such as my Qume) with the Poly serial port? I have heard of a device called the "CS-1" which converts serial to parallel, but I don't know how well it would work. Any suggestions welcomed. Jon Wolfert, JAM Productions, 4631 Insurance Lane, Dallas, TX 75205.

Anyone interested in personal finance programs? I have a series I've developed for my own use, including securities portfolio, income summary, short vs. long-term sale analysis, and total estate summary. I'm willing to sell or trade. Bob Johnson, 5303 Luwana Drive SW, Roanoke, VA 24018.

*** SPOOLER WORKS WITH EXEC/93 ***

We have found a way to make the PolyLetter Printer Spooler work with ALL new versions of Exec, including 90 and 93. If you bought the Spooler for use with Exec/83, you can now upgrade to a newer Exec and still use the Spooler.

The Spooler works in conjunction with the file named "Sio.PS", which was included on your Exec/83 disk, and is also included with the Spooler. The Sio.PS file included on newer versions of Exec is not the same, and the Spooler won't work with it. If you want to use the Spooler with some other Exec, here is the procedure:

1. Delete the file Sio.PS from your new Exec. You will have to use TweakSys to make it a non-system file, before you can delete it.

2. Copy the file Sio.PS from an Exec/83 disk, or the Spooler disk, onto your new Exec. Again, use TweakSys to make it non-system so you can copy it.

3. If you changed the header plug on your printer interface card (inside the Poly cabinet) when you converted to the newer Exec, you will have to put the header back the way it was for Exec/83.

Now, the Spooler will run as it did before under Exec/83.

*** SUPER ZIP (FOR KEYBOARDS) ***

by Frank Stearns
North Hollywood, CA

Want lightning-fast auto-repeat response from your keyboard? All you need to do is change two 10 uF capacitors in the keyboard and you'll be amazed at the difference. (C3 and C4 in KYBD III; C1 and C2 in KYBD II.)

There are two timing functions in auto repeat. First is the delay between when a key is struck and held and when the first repeat begins. The other determines the rate of repeat once the held-state is detected. In my keyboard I have reduced these times by 66%. You can do the same or any use combination that feels good to you. (I changed the two 10 uF caps to 3.3 uF.)

There are things to consider when changing these times. Too fast in that initial response and you'll actually have to worry about the key return spring speed — more than that, how fast you can lift your fingers!

Assume for a moment we're in the Editor. If you're doing multiple deletes, paging (Ctrl-P or Ctrl-N), Ctrl-X's, Ctrl-W's, or any other keystrike that takes processor overhead, watch out! Your repeats may happen faster than the computer can process them, hence you won't see them real-time. You'll fill the type-ahead buffer, and the computer will handle those key-strikes when it gets the chance. You may go too far too fast! Be careful. (Though once you get used to it, it's nice just to use a short burst of DEL's or Ctrl-W's instead of holding the key down. If you overshoot, use Ctrl-U.)

It's interesting to note that different Exec's and versions of the Editor respond much differently. Generally, the older the version, the more sluggish the response.

HOW TO DO IT:

You're looking for the two 10 uF capacitors noted in the first paragraph. They're both on the keyboard PC board, up near the "1" and "2" keys. Each cap is part of the RC time-constant network for the two one-shots (dual; 556 in DIP package). By reducing that time constant, we increase the speed of held-key detection and the rate of repeat. You can change either the R or the C, but since the R values are somewhat tangled up with some other things in the one-shot network, it is simplest to change C. Try 5 uF first. This will halve the times. Feeling adventurous? Try 3.3, or even 2.2. For the fool-hardy, give 1 uf a try. (1/10 the original time.) Remember, you can change those values for the keyboard "feel" you like. Note the "+" sign on the capacitors and be sure to install your replacement cap with the same polarity as the original.

SOFTWARE NOTES: To avoid filling the typeahead buffer accidently, you may want to make frequent use of the system routine "Flush". If you're in Assembler and call Flush, don't forget to "EI" - enable interrupts afterwards! In BASIC, do a "Z=CALL(1054)". BASIC will take care of the interrupts for you. Note that in the Editor, it appears that the system calls Flush after an arrow key has been processed. No matter how short the RC, arrow keys respond "real time", and that "real time" directly depends on the routines that process arrow key strikes. Arrow moves with Edit 3.0 respond much faster than Edit 86 or 80.

*** THE "MOD" FUNCTION ***

by Russ Nobbs

This program demonstrates the MOD (modulo or remainder) function which is not in any copy of Poly's BASIC manual but which works in COlL BASIC.

Y MOD X is (according to HP documentation for the HP-41C) done with Y-(INT(Y/X)*X). This divides Y by X and gives the remainder. If X is zero Poly BASIC gives "Division by zero" error message.

```
100 PRINT CHR$(12)
110 INPUT "Input 2 numbers-->",X,Y
120 A=MOD(Y,X)
130 B=MOD(X,Y)
140 PRINT Y," MOD",X," =",A,
     "  ....syntax is A=MOD(Y,X)"
150 PRINT X," MOD",Y," =",B,
     "  ....syntax is A=MOD(X,Y)"
160 PRINT
170 GOTO 110
```

*** DISK-OF-THE-MONTH ***

The GAMES disk last issue was a great success. Thanks to everyone who ordered it, and thanks also to everyone who sent in their favorite games to be included. We'll be doing that again in future Disk-Of-The-Month offerings. In the meantime, we have a nice selection of utilities this month.

READ.GO simulates the Exec TYPE command, but adds some refinements. You can back up! It accepts Editor-like commands to let you read through the file.

COUNT.GO is version 2.0 of PolyLetter's word-counting program. Multiple files may be counted with one command. You can count words, lines and pages (you specify the number of characters per line or page). And COUNT 2.0 will correctly count a file that's too big to fit into memory.

INPUT.BS is an input subroutine, contributed by Bill Sullivan. It lets you choose what type of input a program will accept, and keeps faulty input out of your program. It has a user-definable flashing cursor, option for numeric input only, and variable input length with underlines to show the limit. Many of the features found only in high-priced input routines!

FLIES.BS was contributed by Russ Nobbs... (that's all I can say about it!)

To order, send a check for $15 to PolyLetter, 1437 Sugarwood Lane, Norcross GA 30093. Disk-Of-The-Month is supplied on 5" SSSD diskette, with Exec/83 and BASIC COIL.

*** POLY -ADS ***

Ads are published as a free service to PolyLetter subscribers.

WANT TO BUY: Used 8813. Software and configuration not important, but low price is! Douglas R. Schirripa, 819 Montgomery Avenue, Apt. B302, Bryn Mawr, PA 19010.

WANT TO BUY: Your extra 8810 or 8813 with keyboard and at least one drive. I need an extra one for home use. Robert L. Schwartz, 906 Main Street, Suite 405, Cincinatti, Ohio 45202.

All of the following programs were written by Ralph Kenyon, and are available through PolyLetter (many at reduced prices):

SELECT: Selects data file records, searching an input data file for a specified character string. Use it to pick out all names beginning with a given letter, or to pick out everyone in a data list with a particular code. $65.

VSELECT: Enhanced version of select which allows variable length fields within each data record. $85.

DIRECTORY REBUILD: Allows rebuilding a crashed directory or adding directory files to recover data past the indexed area. Documentation includes a detailed description of the recovery process using Rebuild and Szap together. $20.

REPLACE: Replaces a program with an updated version having the same name and size (from another drive). Copies the new version on top of the old one. Useful in replacing a crashed program with a good copy from the backup disk. $25.

PRINT-TO-FILE. Takes output to the printer and sends it to a file. Available for Exec/78 and Exec/83 (specify which). It is compatible with WordMaster and allows writing your formatted output to a disk. $25.

COPYSELF DATADISKS: A tailored operating system of only 3 sectors! Will copy itself to another drive and boot. Single density, two or more drives, minimum memory. Ideal for use with non-system packages or data backup files. (Buy one... it reproduces itself.) $20.

DISASSEMBLER: Works on Poly machine language disk files and produces a derived assembly language source file. Output file has SYSTEM labels and may be re-assembled. Requires 32K. $25.

```
10 REM  Convert number to any base.
20 INPUT "What base do
         I convert into: ",B
30 INPUT "Number to convert: ",N
40 C$=""
50 N1=48+MOD(N,B)
60 C$=CHR$(N1+7*(N1>57))+C$
70 N=INT(N/B)
80 IF N>0 THEN 50
90 PRINT C$ \PRINT
100 GOTO 30
```

*** MS ERROR MESSAGES ***
by Russ Nobbs

On my MS system, with two 8" SSDD drives, 03xx errors are often reported as 01xx errors. Many of them, especially the MS disk controller messages, are listed in the manual only as 03xx errors. The 01xx errors make no sense and the user is not led to check out the 8" disk controller. I lost a lot of data for a while until it died entirely and we switched disk controller boards.

According to the old System Programmer's Guide, the routine Look changes a number of 01xx error messages into 03xx (notably 0102 through 0106). Perhaps Poly listed only the 03xx errors for the MS because they expected them all to be filtered through Look. When my disk controller was going out last year, no one at Poly could tell us why we got the 0182 and 0184 errors.

*** PERMUTATIONS ***

If you enjoy working the JUMBLE puzzles in the daily newspaper, try this program. It takes a string and rearranges it into all possible permutations. Thanks to Bob Felts of Atlanta who wrote this for PolyLetter.

```
10 DIM A$(1:72),T$(1:72),A(72)
20 INPUT A$
30 N=LEN(A$)
40 GOSUB 240
45 IF N=1 THEN STOP
50 FOR I=1 TO N-1
60 A(I)=N-I
70 NEXT I
80 I=N-1
90 IF A(I)=0 THEN 140
100 GOSUB 180
110 GOSUB 240
120 A(I)=A(I)-1
130 GOTO 80
140 A(I)=N-I
150 I=I-1
160 IF I<>0 THEN 90
170 STOP
175 REM  reverse part of A$
180 T$=MID$(A$,I,N)
190 A$=MID$(A$,1,I-1)
200 FOR Y1=N-I+1 TO 1 STEP -1
210 A$=A$+MID$(T$,Y1,Y1)
220 NEXT Y1
230 RETURN
240 PRINT A$,CHR$(9),
250 RETURN
```

If this issue looks different, it's because we are using 12-pitch type instead of our usual 10-pitch. Several readers suggested the change, to improve our looks, and also to save on printing and postage costs. We can print six pages worth of information on five pages now. We'll still be using 10-pitch for program listings and other segments requiring special characters, since our 12-pitch print ball doesn't carry all of the standard ASCII characters.

*** NEW OPERATING SYSTEM ***

Because so many users have asked for an operating system of even greater capability than VM, IBM has announces the Virtual Universe Operating System - OS/VU.

Running under OS/VU, the individual user appears to have not merely a machine of his own, but an entire universe of his own, in which he can set up and take down his own programs, databases, system networks, and planetary systems. He need only specify the universe he desires, and the OS/VU system generation program (IEHGOD) does the rest. The program resides in directory GODLIB.DX. The minimum time for this function is 6 days of execution, with 1 day of review. In conjunction with OS/VU, all system utilities have been replaced by one program (PROPHET.GO) which resides in MESSIAH.DX. This program has no control parameters or input, as it knows what you want when you run it.

Naturally, the user must have attained a certain degree of sophistication in the data processing field if an efficient utilization of OS/VU is to be achieved. Frequent calls to non-resident galaxies, for instance, can lead to unexpected delays in processing. Although IBM, through its wholly-owned subsidiary, the United States, is working on a program to upgrade the speed of light and thus reduce the overhead of extraterrestrial and meta-dimensional paging, users must be careful for the present to stay within the laws of physics. IBM will charge an extra fee for violations.

OS/VU will run on any IBM equipped with extended WARP feature. Rental is twenty million dollars per CPU-microsecond.

*** DEALER-WRITTEN SOFTWARE ***

We thought you would be interested in seeing the wide variety of software available for the Poly. The programs listed here were written by Poly dealers, and described in a flyer released by PolyMorphic.

Sorry, but we have no direct knowledge of most of these programs. You should contact the dealers directly for detailed information. Prices shown were also taken from the flyer, and may not be current. In many cases, documentation or sample output is available for a small cost.

Applied Microcomputer Technology, (408) 438-1608: Business order entry system for wholesale/retail, $950. Auto-dial modem system (with modem), $495. Contractors cost accounting system, $100. Newspaper distributor's program, $100. Disassembler, $50.

Advanced Management Systems, (303) 758-1223: Medical computer service system.

Barrett Associates, (305) 678-0172: Manufacturer's representative business system, $550.

Byte Shop, (805) 647-8945: Electronic cash register interface, $469 (discounts available).

Computer Center, (205) 942-8567: Accounts receivable, $500. Accounts payable, $250. General ledger, $500. Inventory, $250. Payroll, $500. Fixed assets accounting, $250. Surveyor's package, $500.

Computer Shop Santa Barbara, (805) 966-2638: Mail list, $200. Quote program, $50.

Contemporary Computers Inc., (602) 969-9110: Insurance data base, $200. Office and business expense account, $100. Job journal, $400. Quotation program, $400. Country club accounts receivable, $500. Country club payroll, $500. Restaurant inventory and modeling, $700. Accounts receivable and general ledger, $400. Business expense account, $100. Commissions earned, $100. Payroll, $500.

Kingmont Enterprises, Inc., (916) 988-8189: Utility package, $150/dealer, $65/single user. Audio control and switcher, call for price.

Micro Applications Corp., (301) 757-5667: FORTRAN, $500.

Microbyte Computers Inc., (604) 873-0133: Canadian payroll. Tax computation. Order entry. Finance. Games. Matrix operations. Search utility. Text editor. Sort routines. EXTENSTAR sort/search package. Disassembler. Hex expression evaluator. Memory test. BASIC utilities. Word processing package.

MicroMart, (605) 338-5592: Star Trek (real-time), $65. Machine language sort utility, $175.

Micro-Systems Design, Santa Barbara CA: Real estate system.

Midwest Computers/Micro Products, (314) 584-3891: Cumulative customer account records, $300. Payroll, $250. Inventory, $250. Microfilm indexing system, $250. Mailing label package, $50.

Omaha Computer Store, (402) 592-3590: House siding inventory control, $500.

Charles Trayser, D.V.M., Fremont, CA: Payroll, $150. General ledger, $150.

21st Century Software, (513) 381-2642: Legal time accounting.

Star Systems, (805) 966-1307: Star data base, $500.

The purpose of PolyLetter is to create a forum of ideas for users of Poly equipment.
One year (six issues) subscription $15 US and Canada, $20 overseas.
Editor: Bob Bybee
P o l y L e t t e r
1437 Sugarwood Lane
Norcross, GA 30093
(404) 925-2480
PolyLetter is not affiliated with PolyMorphic Systems in any way.

\- - -

PUT ME ON YOUR MAILING LIST:

Name _____
Business _____
Address _____
City/State _____ Z _____
Phone _____
System _____
Printer _____
Uses _____
Future uses _____

*** BOMB SHELTERS ***
or, how to escape from Poly's Front Panel

by Russ Nobbs
Rings and Things
Spokane, Washington

Most Polys love to jump into the Front Panel. A static burst, line voltage drop or spike, the call to a routine or disk that isn't where Poly expects it to be and a variety of other events cause Poly to come up with the familiar (if bewildering) Front Panel display.

In most cases, there are simple ways to get back "home" without destroying what you were doing before the jump. All these ways are based on the use of the Front Panel as a machine level debugging tool. The commands used to manipulate the Front Panel also allow us to get out of it with our program, printing, editing or whatever unscathed.

You CAN go home again...

The simplest way to get out of the Front Panel is to type the single character "G". On my system this restarts most operations.

In the editor the screen may be a mess so try the arrow keys to see that you are back in the editor. (If you got back each line will straighten out as you scroll past it.)

In BASIC "G" should give you back your ">" or ">>" prompt. CON should CONtinue your program with your data intact. Or you can type LIST to check out your program. If you jumped from BASIC into the Front Panel but "G" gave you Exec's "$" or "$$" prompt try CON to CONtinue your interrupted program. If you can't continue that way try REENTER followed by CON.

Why it jumps:

I asked veteran PolyUser Ralph Kenyon why Poly was so fond of the Front Panel. His explanation made it sound like Poly's designers had a well thought out reason for it -- so the user could recover from static bursts and the like.

Ralph said that any time Poly reads the hex byte "FF" when it is running it will jump into the Front Panel. FF is machine code for RST 7 (restart to octal location 070 or hex 38) which causes a branch to location 38 in Poly's ROM. This saves the current locations on the stack, and comes up with the Front Panel display.

The FF byte is encountered when Poly finds nothing to read. If you pull out a disk that Poly was ready to go back to or if you attempt to read beyond the top of memory you may cause Poly to think it's reading FF.

(If you have less than 56K of user memory try this: Type ENABLE. Clean out memory with ZAP. Now tell Poly to run the program (that's not there) with START. Poly will race through all the 00 (NOP) bytes until it hits top of memory, will think it reads FF there and jump to the Front Panel. Notice PC is pointing to 1 byte higher than the top of your memory. Typing G keeps you in the Front Panel while incrementing PC 1 byte further each time.)

How it works...

G is the Front Panel command to return to the interrupted program and continue with it. (Just like G in BASIC's WALK.)

L (xxxx) (where xxxx is a hex address) moves the memory window pointer to location xxxx where you can Look at or Load new program or data bytes.

SP sets the Front Panel to modify the program counter (PC) in the 8080 stack.

J tells Poly to expect a 2 byte (Jumbo) hex data word. J is used to load a 4-digit hex number into registers such as the PC, or program counter.

SPJ (xxxx) (the previous two combined) effectively performs a program jump to location xxxx by modifying PC to be xxxx where (hopefully) your program can start again.

When you say "jump", it jumps!

Sometimes the jump to Front Panel loads the program counter and the other registers with strange numbers that do not point back to the area where you were working. These are the times that typing "G" doesn't do anything but flicker the screen. Don't despair! There is still a way.

Poly has arranged most operating system software so that there is a re-entry or warm-start point -- a place where you can jump to and keep everything intact. (Cold-starting starts the software at the beginning by zeroing constants and doing other initialization routines. Warm-start assumes that there is an interrupted process to be completed.)

For BASIC the warm-start address is hex 3203.
For Exec it's hex 403.

The editors seem to be an exception. SPJ 3203 ought to work but it does strange things. My best results come with warm-starting Exec with SPJ 403 (space) G followed by REENTER. This puts me right back in the editor & I can continue as if nothing happened.

Charles Thompson simply hits the LOAD button. The text is usually safe in memory after the system message comes on the screen so recover with:

```
$ENABLE
$$GET Edit
$$REENTER
```

You should then be back in the editor and can enter ESC CTRL-E to save your text to disk and then start over. (If you have a program named INITIAL on the disk this method will not work, since the system will run INITIAL when you press LOAD, and INITIAL will wipe out the Editor data in memory.)

Overlays don't seem to have a reentry point but can often be restarted by getting back to Exec and typing REENTER.

So, if typing "G" doesn't put you back into familiar territory try using the Front Panel command "SPJ", followed by the warm start location you need, followed by a space, and then "G". (Don't worry that some of the characters you type are snatched away from the screen as you type the next one -- that's normal.)

For BASIC use "SPJ 3203" (space bar) "G". You may still need "CON" to CONtinue your program.

These methods work on an 8813 with 2 8" SSDD drives in an MS, and in a system with 5" SSSD drives. They work with BASIC C01, C01L, C02, Exec/78, 79, 83, 90, 93, 94 and both WordMasters. If your system does things in a different manner, or if you have better ways of recovering, please let me know & I'll update this article.

(The Front Panel is not available in Poly's TwinSystem.)

PolyLetter

1437 Sugarwood Lane
Norcross, GA 30093
(404) 925-2480

ATLANTA JUL 20 '91 GA.
U.S. POSTAGE
0.18
PITNEY BOWES 1274590

Issue #81/5

PolyLetter

September/October

NOTES FROM POLY

This column from PolyMorphic Systems is for the purpose of letting PolyLetter readers know what is going on at Poly. We hope you enjoy this column, and we would appreciate your comments and suggestions on items to put into the column.

This issue's column is going to deal mostly with Poly's procedures on support, both hardware and software.

PolyMorphic Systems has maintained a policy of trying to provide good user support along with keeping up with the current trends in the quick-paced computer industry. Unfortunately this brings about some problems, not the least of which is maintaining upward compatibility with older systems.

Poly has met the challenge of upward compatibility fairly well, in that software written in A00 BASIC can be run on C03 BASIC with few changes. Also, hardware purchased four years ago should be compatible with hardware purchased today. Our biggest problem has been, and still is, the FIRMWARE or ROMs. Many Poly users find changing the ROMs every year a bit of an inconvenience, however we should like to note that many other manufacturers require much more extensive changes for a system built in 1979 to run like a system built today.

The people having the most difficulty dealing with the upgrades necessary to run the current software are the people who wait a number of years and then assume Poly has made NO changes in the operating system. The truth of the matter is, we will probably release a new operating system about every six months. The main reason for new operating systems is bug fixes, followed by enhancements or new additions to the OS. The policy for fixing a bug in Exec/75 is to see if it exists in Exec/95. If so, we will fix it; if not a new disk must be purchased. We have to charge for a new disk because of the enhancements, and new features to the system. We only release the LATEST version of the OS so we can more effectively support the operating system.

If you have any suggestions please write them down and send them to us. If you have any questions, please call between 9:00 and 11:00 AM Pacific Time. This is so we have the afternoon to implement the suggestions, and fix the bugs we've been notified about.

PolyMorphic Systems
460 Ward Drive
Santa Barbara, CA 93111
(805) 967-0468

(continued on page 2)

This issue we welcome a new contributor: PolyMorphic Systems will now be writing a column each issue. This issue they discuss Poly's hardware and software support policies, and give us a look at future Poly products. Poly has been very cooperative lately in providing us with good information, and we appreciate it very much.

Another "hot" subject these days is CP/M. This issue we present the first of a series of CP/M reports, written by the people who are using it. If you are using Poly CP/M, please tell us about it so we can pass your experiences on to others.

PolyLetter recently ordered the new System Programmer's Guide from PolyMorphic, and we must agree that it is worth the $250 price tag. We will be reprinting some hints from the Guide from time to time in future PolyLetters. Not only does the new Guide explain everything the old edition did, but this one also includes details of the TwinSystem and Volume Manager operation. It is a MUST for serious assembly language programmers.

A SuperZap article is coming up next issue for SURE!

(NOTES FROM POLY - from page 1)

IN THE QUEUE

The current release of the operating system is Exec/95. BASIC's release number is C04.

Coming up in future releases:

MACRO-85...Poly's Assembler with Relocatable code, INTERNAL and EXTERNAL labels, TITLE, REPT, INCLUDE, and many other features.

EDIT-4.1.0....A New editor designed to have ALL of the current features but will work on ANY size screen, and will also work on a serial terminal.

24x80 CP/M...Poly CP/M designed to work on a serial terminal utilizing the printer interface card.

A Linker/Loader to go with the Relocatable code.

...and further down the road, a NEW file system able to handle more than one output file per volume; and PASCAL!

(CP/M is a trademark of Digital Research, Inc.)

POLY 88 USER'S GROUP

We have received some unfortunate reports concerning the Poly 88 User's Group headed by Roger and Pat Lewis. PolyLetter has had several complaints about their group: orders have not been filled, they have not answered phone calls or letters, and yet they have (on several occasions) cashed a purchaser's check and not shipped any goods.

PolyLetter has written to the Lewis's group and explained the situation. We have given them an opportunity to respond to these reports, but they have not answered our letter.

Until this problem is resolved, PolyLetter must warn mail-order buyers not to do business with Roger and Pat Lewis's Poly 88 User's Group. We are sorry to see this situation develop, and we hope they will take steps to satisfy their customers soon.

If you have any information to contribute concerning this situation, please write to PolyLetter. We will keep you posted.

DIGITAL RESEARCH INFORMATION

Ken Gudis provided us with some information on Digital Research, who created CP/M. The following is reprinted from a Digital Research newsletter.

Who to call at Digital Research:

Marketing Department: (408) 649-3896
Technical Hot Line: (408) 375-6262

We at Digital Research want to give you the best service possible. In our efforts to be of help to you, we would like to describe what kinds of information are available from our different departments, and share our policies regarding contacting DRI [Digital Research, Inc.] by telephone.

DRI does not make recommendations regarding the hardware or software products of any other companies.

The Technical Hot Line is intended for use by registered Digital Research customers to answer specific questions.

We do not provide technical support for a BIOS or XIOS written by another company, unless the caller has attempted to resolve the problem with that company first.

We are happy to provide general information on how to use DRI software, but cannot debug application programs or provide consulting advice on how to write them.

If you have a suspected bug, we can best test it if you send it to us in documented form, without a call to the hotline.

If you are not registered, or if your question is in regard to sales, licensing, distribution and availability of DRI products or compatible application programs, please call the Marketing Department at Digital Research or our foreign representatives.

CP/M CORNER

Now that Poly CP/M is in the field, we plan to present a series of articles on how it works and what it can do for you. This month's article is by Tim Scully, of Mendocino Microcomputers.

The Poly disk operating system expects RAM to begin at address 2000 hex. Most new Polys are now shipped with 56K of RAM from 2000 to FFFF hex, split into two memory boards: a 48K board from 2000 to DFFF, and an 8K board from E000 to FFFF. The address space from 0000 to 1000 is used by the CPU board ROMs (0000-0BFF) and a little RAM (0C00-0FFF). The addresses from 1000 to 2000 are used by the video board and the newer disk controllers.

CP/M expects RAM starting at address 0. The hardware changes made to a Poly which allow it to run CP/M are all designed to accomplish this by "phantoming" all the normal Poly devices below 2000H, and moving the top 8K of RAM from E000 down to 0000. Pin 16 on the S-100 bus becomes "CP/M-", and controls the "phantoming" process. (Phantoming allows two different devices to share the same address, but not at the same time. This new control line on the bus decides which device is being accessed, and which one is a "phantom": not really there.) Executing an IN instruction from port 8 turns CP/M modifications off. An IN from port 12 turns on the CP/M devices. This is done automatically by the routines which operate CP/M.

Since CP/M turns off the Poly video board and the disk controller's dual port memory, CP/M must also turn these things back on every time one of these devices needs to be accessed. This is done each time CP/M sends a character in or out.

Poly CP/M is quite slow in disk accesses. Don Moe said that this is because Poly reads and writes to a disk 1024 bytes at a time, but CP/M only moves 128 bytes at a time. Thus, Poly spends 8 times as much time doing disk accesses as it really should. (This problem may have been corrected by the time you read this.)

As shipped, Poly CP/M is a 44K system. The remaining 12K of RAM are used by CP/M. Poly's CP/M uses more RAM than some other versions of CP/M. If you have 8K of additional RAM, you can reassemble CP/M to give yourself a 52K system.

BOOK REVIEW
By Charles W. Gross

HOME COMPUTERS CAN MAKE YOU RICH, by Joe Weisbecker. Hayden Book Co., Rochelle Park, New Jersey, 119 pages, paperback, $5.95, 1980.

This book takes the position of a person who wants to make money from his hobby. It is dominated by suggestions that one write articles, books, or programs, about subjects suggested by the author. He advises that any money earned should be reinvested in the hobby/business.

Several good ideas are presented about selecting a name for one's business, the stationery used, and selecting hardware. Mr. Weisbecker presents many subjects one could write about, but he leaves it up to the reader to develop the idea and build a story (or program). He suggests that as soon as one has read a couple of books on BASIC, one can sit down and write any kind of program. Well, maybe Mr. Weisbecker can, but I can't.

The author does present several rules which, if followed, would assist in creating a successful business. There are chapters on: Writing For Money, Creating And Selling Programs, Service For Sale, Invent Your Way To Success, and Making Your Money Grow. A chapter on "imagination" addresses the arts, crafts, and novelties that may spring from an interest in computers. The author asks, who will be the one to invent the Pet Rock of the computer age?

Weisbecker covers a great deal of ground. This is a book well worth reading, especially by those of us who would like to earn extra money with our computers.

A POLY DICTIONARY

Interactive System Services announces a dictionary for the Poly 8813. It has over 3000 words and is expandable. The program will automatically check a file, displaying any words not found in its dictionary. It will let you perform corrections as it proceeds through the file. "SISS.RL" is available for single or twin systems, priced from $350. Contact ISS at (805) 964-0062.

S-100 BOARDS IN THE POLY

The Poly 8813 chassis uses a system of signals called the S-100 bus. The name comes from the 100-pin connectors used by the cards. PolyMorphic, North Star, Cromemco, and many other manufacturers produce cards which are designed to plug into the S-100 bus.

HISTORY LESSON: The first reasonably priced small computer was the MITS Altair 8800. In 1975, it sold in kit form for about $400. This included the CPU board (using an 8080), one (1) K of RAM, and a true front panel which used 25 switches and 36 lights. No keyboard or video display was provided, although an enthusiastic hobbyist would probably hook up a noisy ASR-33 Teletype to be a terminal. Disks were out of the question. Even cassette tape was not yet available for data storage.

The Altair 8800 used a 100-pin connector for its bus signals. Another manufacturer, Imsai, also adopted this connector for its computers and boards from the two systems were (almost) compatible. With 100 pins on the bus, and not all of them being used by every board, it was inevitable that some incompatibilities would arise. Still, the S-100 bus became a de facto standard.

Other manufacturers, including PolyMorphic, began producing boards which were compatible with existing S-100 systems. The Poly 88 used a cabinet which would hold five S-100 cards. Since the S-100 was based around the 8080 microprocessor, only manufacturers using the 8080 had any reason for going S-100. Computers using the 6800, 6502, or other processors, developed their own bus structures, but none of these busses is as widely used as the S-100. (When the Z-80 came along, its 8080 compatibility made it an ideal choice for the S-100 bus, and so many Z-80 systems now use S-100 cards.)

The S-100 bus has some disadvantages. Because of their design, S-100 boards will use more power (and produce more heat) than some other systems. This can lead to reliability problems, or failure of the power supply. (The Poly 88 was prone to this, and PolyMorphic eventually added a fan to blow air through the cabinet.) All S-100 cards must use the 100-pin connector, which is more expensive than the 50-pin connector used in (for example) 6800 systems. But, the vast market for S-100 boards has convinced many companies that S-100 is the way to go.

Now, a glance at any computer magazine will show hundreds of ads for S-100 products. Some examples are video boards, serial ports, parallel ports, disk controllers, modems, CPU boards, AC device controllers, and of course, memory boards. All of these boards claim "S-100 Compatibility." But what exactly is S-100 compatibility?

Each manufacturer of S-100 products has made a few "adjustments" to the S-100 scheme, to suit his needs. PolyMorphic, for example, added a real-time clock signal to the bus; this is not present in many other S-100 systems. Problems can arise when one manufacturer assigns his own meaning to a certain bus pin, and some other manufacturer assigns a different meaning to that same pin.

So, when shopping for boards to add to your Poly system, here are a few things to keep in mind:

AN S-100 BOARD IS NOT ALWAYS AN S-100 BOARD. There will be slight differences between manufacturers. A Godbout memory board might work in your Poly. An Imsai memory board might not. Make sure you can return the board if things don't work out well.

HARDWARE REQUIRES SOFTWARE. If you are adding a device such as a serial port or parallel port, something must tell the Poly system how to use it! It is not reasonable to expect that you can buy a board from another manufacturer and immediately have it work in your Poly, without some software written specifically for the board. This will require some assembly language programming ability, and a fair amount of hardware knowledge. The person who writes the software will probably have to be familiar with the Poly operating system. You may be able to do a limited amount of work with the board using PEEKs, POKEs, INPs and OUTs from BASIC.

YOUR SYSTEM MAY NOT WORK EITHER. Sometimes just adding a new board to your Poly will cause the whole system to stop running.

(continued on page 5)

(S-100 - from page 4)

This will happen if the board conflicts with existing system hardware, such as two memory boards assigned to the same address. You should read the instructions carefully to avoid getting into this condition, since it can damage the system (although not usually). Of course, you should NEVER insert or remove boards while the system is turned on.

PolyLetter has done quite a bit of experimenting with different S-100 boards. If you are thinking about adding some boards to your system, we would be happy to discuss the possibilities with you. In some cases we can recommend appropriate software for your application, or help you decide how to produce the necessary programs.

SOFTWARE SHOP

These programs were written by PolyLetter and friends. Programs are available only on 5" SSSD disks.

MODEM KIT: PolyLetter can put you into the modem of your choice. Our Modem Kit includes programs to send and receive text and machine code files, and lets you use your Poly as an intelligent terminal. Hook up to the Source or Micronet using your Poly. Also comes with a cable to let you connect your modem to the Poly printer port with no hardware changes to the Poly. $100.

REMOTE SYSTEM: This is the counterpart to the Modem Kit above. If you have a terminal and want to use it to access your Poly from afar, the Remote System lets you do it. Connect an auto-answer modem to your Poly printer port using the cable (included), and run Remsys.GO. When you dial into the modem, you will be able to run your system remotely. You can run BASIC or other applications programs. Your salesmen can enter their orders while on the road. You can "chat" with the local operator through the modem. New low price! $100.

(Both of the packages above do not include a modem. PolyLetter can obtain a modem for you if you wish. Call for details.)

PRINTER SPOOLER: Originally for Exec/83, can now be used with newer Exec's as well. Spooler allows you to reserve some memory as a printer buffer, so that your printer can be running off your output while you go on to other tasks. Depending on printer speed, you can save over five minutes every time you get a printout. $50.

Programs listed above were written for Exec/83, single-user systems. We expect them to operate on any other single-user system. PolyLetter will work with you to help get programs running, or provide a full refund if you are not satisfied.

NEW SOFTWARE FROM RALPH KENYON

Ralph's company, Abstract Systems Etc., has released these new programs. You can order them from him at 1686 West Main Road, Portsmouth RI 02871, or call (401) 683-0845.

QuickSort and HeapSort are overlays which sort strings in a BASIC program. Both are written in machine code for speed. HeapSort uses a more complex sorting method which gives faster results when an array is almost sorted to begin with. QuickSort: $25. HeapSort: $40.

Little-Ada is a compiler, supporting a subset of the new Department of Defense programming language, "Ada." This is a Pascal-like language. The system disk is supplied with a Little-Ada escape library, a macro library, an error message overlay, and documentation. $50, or documentation only for $2. (Ada is trademarked by the U.S. DoD.)

FORTH is an interpreter for the FORTH language. (See "On The FORTH Day Of..." in this issue.) A demo FORTH application is included, along with details on the Poly implementation of this language. $40.

DisAssembler (version 2.0) supports the lastest versions of Exec, which use macros in the SYSTEM.SY file. Overlay names are correctly handled. A faster sort is included, or you can order a machine-language sort routine with the new DisAssembler. Price $35. With QuickSort, $55. With HeapSort, $70.

(Write to Ralph for a complete catalog of Abstract Systems' software.)

POLY - ADS

Ads are published as a free service to PolyLetter subscribers.

WANTED: Poly hardware and accessories, publications, etc. Don't throw it away until you give me a chance at them! John McNally, 7049 Armstrong Road, Goleta CA 93117, (805) 968-4628.

WANTED: New style keyboard, CRT, 8" drives, or what have you, to upgrade my system. Also want to trade (non-copyrighted) software. Al Levy, PO Box 71, Hicksville, NY 11801, (516) 997-3653. Are there any Poly users in the NY area?

FOR SALE: 8813, 2 drives, 48K, keyboard II, Leedex monitor, software, confidence pack, WordMaster II: $1900. Also Hayes modem and software, $250. Escon IBM conversion kit, $300. Gary Petrowski, 418 Neptune Ave., Encinitas CA 92024, (714) 436-5881.

FOR SALE: TwinSystem 8813, with 2 SSSD drives and DSDD 8" MS. Two videos and two keyboards. Ready to operate, can ship immediately. Asking $9000. Terry Castel, (404) 355-8308.

FOR SALE: SSSD controller, $200. Shugart SA400 drives (used in SSSD systems), $200. SSDD MS unit, $2850. All items in almost-new condition, and I will guarantee them for 90 days. Dr. Charles Trayser, (415) 651-0100.

FOR SALE: Poly 48K memory board with 16K installed, never used, original factory carton. 8813 with CP/M mods. Extra boards, including memory, printer interface, cassette, CPU, video. Software including PLAN, WordMaster, Confidence pack. Will trade or sell. Digital Transactions, (914) 232-7958, or Source mail TCV946.

"As with any system functions, care should be exercised in debugging and experimenting with assembly language programs, especially in the TwinSystem. Causing a system failure while the other user is in the process of packing a disk may be hazardous to your health!"

-- System Programmer's Guide

STOPPING SCROLLING DISPLAYS
by Russ Nobbs

Output from Poly BASIC programs or LISTing of a BASIC program scroll up on the screen faster than most folks can read. There are several solutions.

The newest BASIC (C03 -- found on Exec 94) allows the user to hit any key to stop screen scrolling and to hit any key again to start it again.

The May/June '81 issue of PolyLetter contains a short machine language routine (which is loaded via BASIC) to stop scrolling by pressing the backspace key. This works with Exec 83 thru Exec 90 (and perhaps others). (It uses the user-definable interrupt provided by Poly as UVEC & UCHR.)

Or, with any system Exec and BASIC, you can set up your printer as "Printer Screen" and connect all your output to the "printer". (Use FILE:2,LIST to connect the "printer" and LIST:2 to list the program.) BASIC will stop every 14 lines, put a "." at the bottom of the screen and only continue when you hit a key. Program output that is "printed" to the screen, with PRINT:2, will also be stopped this way.

The only problem this doesn't solve is the screens of data that contain wrapped around lines -- lines longer than 64 characters like many data files. Long lines are truncated by Printer Screen.

Depending on the situation you may get lines cut off at the right edge of the screen (PRINTing a data file with the printer set up as Screen truncates each line) or you may lose a few lines off the top of each screen due to screen wraparound.

Anyone care to write a patch to the operating system that counted both carriage returns as well as printable characters? This would make TYPEing data files much more readable as well as possibly cleaning up the minor problem with Printer Screen.

(See the July/August 1981 issue of PolyLetter for a program called READ, which lets you TYPE a file a screen at a time, but also allows you to back up.)

ON THE FORTH DAY OF...
by Ralph Kenyon

<1<: Pssst, <2<, I think there's a new occupant in <3<.

<2<: Oh, no, we aren't going to be retired by another system version, are we?

<1<: No, I don't think so. This one's not from Mother PolyMorph -- it's not a new sibling.

<2<: Look at that shipping tag. It says: 'Abstract Systems, etc.' Little-Ada, you hear that? I wonder if it's another version.

<1<: Well, we won't have to wait long. The boss just called for Dfn2. He's listing the drive.

```
$L 3
Disk ForthSys has 41 files on it.
345 sectors in use, 0 deleted.
 Size Name
    3 Fmsg.OV
   23 FORTH.GO
    1 TEST.FX
    1 GO.TX
    5 print-forth-messages.AS
    2 print-forth-messages.GO
    8 Fmsg.AS
    4 8-QUEENS.FX
    5 FN.FX
    5 VLIST.TX
    8 DOC.TX
```

<2<: <1<, what do you make of that?

<1<: Well, he probably wants to make sure this new disk doesn't have any fancy self changing command files...

<2<: No, silly, I mean Fmsg.OV and FORTH.GO.

(continued on pages 8-10)

LETTER FORMATTING

Bill Sullivan provided us with this little BASIC program. It will help you set up most of the formatting commands used in a standard letter. (How many times have you formatted a letter, and had to go back to insert some little command? Use this program and avoid high blood pressure.)

Note that the FORMAT commands in here are for the old WordMaster, not WrdM II. Also, you may want to remove the PACK's on line 30... that part caught me by surprise the first time I ran the program.

```
10 DIM A$(5:100),S$(1:75),R$(1:100),
   F$(1:35),F1$(1:35)
20 INPUT"File name: ",F1$ \F$=F1$+
   "1.TX" \FILE:5,OPEN,F$,OUT
30 L$=CHR$(13) \PRINT:5,"PACK",L$,
   "DEL ",F$,L$,"PACK"
40 F1$=F1$+".TX" \PRINT:5,"EDIT ",
   F1$,L$
50 PRINT:5,"{nrj,nfil,lpp 67,
   bm 10, wid 103, lm 10, rm 10,
   npgn,skp 6}"
60 INPUT "Letter date: ",R$
   \PRINT:5,"{ce}",R$,L$,L$,L$
70 FOR X=1 TO 5 \PRINT"Address
   line",X, \INPUT1": ",R$
80 IF R$="" THEN PRINT A$(X), ELSE
   A$(X)=R$
90 PRINT \NEXT \FOR X=1 TO 5
   \PRINT A$(X) \NEXT
100 INPUT"Change? ",R$ \IF R$="Y"
    OR R$="y" THEN 70
120 FOR X=1 TO 5 \PRINT:5,A$(X)
    \NEXT \INPUT"Re: ",R$
130 PRINT:5,L$,"Re: ",R$,L$,L$,
    L$,L$,"Dear ,{fill,skp}",L$
140 PRINT:5,"{nfil,skp 3,ind 45}
    Sincerely,",L$,L$,L$
150 PRINT:5,"William J.@Sullivan,
    III",L$,"{bpg}"
160 FOR X=1 TO 5 \PRINT:5,A$(X)
    \NEXT \FILE:5,CLOSE
170 OUT 0,"BYE"+L$+F$+L$
```

The purpose of PolyLetter is to create a forum of ideas for users of Poly equipment.
One year (six issues) subscription $15 US and Canada, $20 overseas.
Editor: Bob Bybee
P o l y L e t t e r
1437 Sugarwood Lane
Norcross, GA 30093
(404) 925-2480
PolyLetter is not affiliated with PolyMorphic Systems in any way.

PUT ME ON YOUR MAILING LIST:
Name _____
Business _____
Address _____
City/State _____ Z ____
Phone _____
System _____
Printer _____
Uses _____
Future uses _____

<1<: Oh, I thought... Wait a minute, the boss wants Dfn2 again.

Well, now we'll find out, he just gave the system to the newcomer.

```
$EN
$$b 3
(Exec/83)
$
```

<1<: That's one of our older siblings. I wonder what he's picked up.

<2<: Well, we won't have to wait long. The boss asked for...

```
$FORTH
8080 FIG-FORTH 1.1.0 ok
```

<1<: FORTH? -- What's that?

<2<: I've heard of that. It's some kind of new operating system or something....

BASIC: REM Young upstart! Thinks it's a programming language too.

Asmb: ; I've been around since 02. This FORTH fellow's bringing back stuff you interpreters disdain. Imagine! Insisting there's a difference between numerical and literal types...

BASIC: REM shows how much you know! Why, you can't even do DIM.

Asmb: ; I can so! All I have to have is a subroutine to compute the offset....

BASIC: REM Harumph...

<1<: All right you two, get off that old controversy and let's see what this newcomer can do.

<2<: Welcome, FORTH. On behalf of...

<1<: Can the speech, <2<, we're just a friendly bunch around here. Why, the boss doesn't even spell out the commands. FORTH, you'll have to excuse <2<. He gets so much data, he thinks he's a politician. Anyway, glad to have you with us. By the way, are you a program, a language, or a special operating system?

FORTH: (Thanks for the welcome. I'm used to informal routines myself. As for your question, that's a hard one. I guess the answer is 'Yes.') ok

BASIC: REM See! I told you so!

Asmb: ; Don't be so basic.

<1<: Ahem. Don't mind them. They're just jealous that the boss works with us operating systems most of the time. Now, If you are a special operating system, what do you control?

FORTH: (Well, I'm not just an operating system. I have applications too.) ok

<2<: You mean you're good for something?

<1<: He means a special purpose device, like Speech Lab for example.

FORTH: (No, operating systems generally have tasks or functions and device controllers. Languages generally have programs and environments...) ok

FORMAT: { I've got an environment.}

Little-Ada: -- When I grow up I'll have...

Asmb: ; Go back to sleep, that's not what he's talking about...

<1<: Please continue.

FORTH: (I'm both an operating system and a programming language.) ok

<1<: That seems impossible. How does it work?

FORTH: (Well, I work like an interpreter with immediate commands. If you give me data, I push it on my stack. Some commands require data, so I look for it on the stack. The boss has to remember to put the data on the stack first... Wait, the boss is doing some arithmetic.) ok

```
2
ok
3
ok
+
ok
.
5 ok
```

<1<: How come you keep saying 'ok'?

FORTH: (That's to tell the boss his

command was executed and I'm waiting for more. Also, it's rumored to be a joke of the language designers because they think FORTH is 'ok'. Anyway, here, the boss pushed a 2 on the stack, then he pushed a 3 on the stack. When he gave the + command, it looked for two things on top of the stack, took them off, added them, and pushed the result on the stack. The '.' command took the top item off the stack and printed it.) ok

<1<: So you are a language, like BASIC and Asmb. What does FORTH stand for?

FORTH: (Only partially. FORTH is supposed to mean fourth generation software. I combine the characteristics of an operating system, a language and programs all in one.) ok

BASIC: REM Languages are languages and programs are programs.

Asmb: ; You and your data types.

Little-Ada: — I know TYPE ...

<1<: Can you explain how that works for us first, second and third graders .. I mean generation types?

FORTH: (Well, it's all in the : definitions. Suppose the boss wanted to put what he did in a definition, let's call it TEST. He would just type in) ok

```
: TEST 2 3 + . ;
ok
```

(Now, TEST is added to my dictionary. Whenever the boss types in TEST.. Well, I'll show you.) ok

```
TEST
5 ok
```

(See, TEST is now part of the operating system.) ok

<2<: Gee, and when we get an addition to our operating system, it takes Mother PolyMorph months of labor before she gives birth to a fully formed new sibling.

<1<: (Amazed by the veracity of <2<'s comment), Er, ah, aren't you taking chances of the boss really messing up the system?

FORTH: (I give the boss the power to do so. It's his responsibilty to use it wisely.) ok

BASIC: REM Just like POKE...

Asmb: ; Talk about messing with the system...

<1<: Never mind about that. FORTH, most software isn't bug free when it's first written. Yet you allow each, how do you say, 'colon definition' to be an addition to the operating system.

FORTH: (There's more to it than that. I have a FORGET command which cleans stuff out. Also a name can be re-defined. Each time a new colon definition is created, it is compiled at the time of creation. Because the dictionary search is latest first, only the most recent definition is found. So we can re-define something several times, and then only the last definition is used. Previous uses of the old versions aren't changed, since they were compiled with the old version.) ok

Little-Ada: — Hooray! Another compiler!

<1<: I'm not sure I followed all that, but it seems that you end up with a lot of garbage.

FORTH: (Well, during trial and error sessions, yes, but when the boss gets more efficient, he'll use Edit to create the application, just like he does now with BASIC and Asmb programs.) ok

Asmb: ; How else...

BASIC: REM Not ALL programs...

Edit: Edit what?

<1<: How am I going to know when it's a FORTH application?

<3<: That's easy. I know a forth application when the extension .FX is used.

<1<: ...Then... <3<... you've been modified?

<3<: Yup. Custom tailored to run with FORTH. I even have Fmsg.OV, a FORTH message overlay.

<2<: Oh goodie. You hear that Little-Ada? We're not alone anymore.

Little-Ada: -- When I grow up...

<1<: Sometimes I wonder about this place. First, we're the only micro with an operating system, then they finally get on the CP/M bandwagon, and now this. What'll they think of next?

FORTH: (I have it on good authority that the Artificial Intelligence stuff LISP is being worked on.) ok

Little-Ada: -- When I grow up...

<1<: (To no one in particular:) (Groan), How am I ever going to keep order around here?

PolyLetter

1437 Sugarwood Lane
Norcross, GA 30093
(404) 925-2480

Issue #81/6

PolyLetter

November/December

PolyMorphic has moved! Their new address is:

PolyMorphic Systems
5730 Thornwood Drive
Santa Barbara, CA 93117

The new office is only a few blocks from Poly's old home on Ward Drive. Ken Gudis explains that the old lease had run out, and Poly decided to move into an office park, closer to the mainstream of business. The new location also lets Poly consolidate their operation into offices that better suit them, eliminating the wasted space of their old building.

Poly's phone number remains the same, (805) 967-0468. During the moving process, Poly was without phone service for several days. They apologize for any inconvenience this caused, if you tried to call them while the phones were being transferred.

The staff at PolyMorphic is small, but strong. Poly currently employs about 25 people. Despite this (or perhaps because of it?), Poly's sales are doing well. The hard disk systems and CP/M are the newest products, and shipments of these are going out daily. Ken Gudis says that virtually any order can be filled within 30 days. CP/M conversions (adapting your system to run CP/M) typically take only a few days!

To make this quick turn-around possible, Poly asks that you call before sending in any unit for repair or modification. A return authorization number will be assigned to your order, which you should mark on the package. This allows Poly to service your order with the best possible efficiency.

NEW PRODUCTS: Poly's hard disk system expands, with a 35 megabyte drive! Additionally, a new model of the 10 megabyte drive is now available, with better reliability than before. The 88/HD is sold as an add-on to the 8813, and can

(continued on page 2)

OPEN LETTER TO POLY

Russ Nobbs sent us a copy of a long letter to Poly in response to Poly's bug fix and upgrade policy described in the last PolyLetter. Russ argues that Poly should be more willing to replace defective software at no charge. He cites examples of software which would not run on the system with which they were shipped and proposes the following policy on software and customer support.

1) Defective software to be replaced at no charge with a working product if an original disk is returned to the dealer and/or vendor within 90 days of purchase with a description of the problems. (In some cases a patch, supplied in writing, and installed by the dealer or user would suffice.)

If the upgrade is not available the consumer should be told and the 90 day warranty period extended until the upgrade is available. If this is unacceptable to the user a refund should be available.

The upgrade should also include at no charge the revised pages of the manual that reflect the changes in the product.

2) General upgrades of the PolyMorphic operating system or other PolyMorphic software should be offered to bona-fide purchasers at nominal cost. ($25 - $50 range; enough to cover reproduction & handling costs.) These would be available by returning an original disk.

Included in the upgrade would be addenda pages to the manual and specific pages or sections covering the revised material. A complete manual for the upgraded product should be available to users of previous versions at an additional charge. (A charge that should reflect the size of the manual provided.)

3) All software, manuals & upgrades should

(continued on page 2)

(POLY - from page 1)

be purchased with or without an 8" floppy for backup. The disk controller for the hard disk and floppy is included in the purchase price.

Prices are (approximately) $7700 for the 10 megabyte drive without floppy; $9000 for 10 MB with floppy; $12000 for 35 MB with floppy. Call Poly or your dealer for other prices.

Poly plans to announce a new system using a 16-bit microprocessor, in the first quarter of 1982. The power of a 16-bitter will allow tremendously expanded systems. Multi-user systems and multi-system networking will become possible. Details of this new offering from Poly will be published in PolyLetter as the news develops.

Many applications programs and packages, developed by dealers, are now available through a referral service from Poly. Packages include: radio station management, real estate, accounting, manufacturer's rep., and veterinarian's programs. Call Poly for details, and they will put you in touch with the appropriate software vendor.

The purpose of PolyLetter is to create a forum of ideas for users of Poly equipment.
One year (six issues) subscription $15 US and Canada, $20 overseas.
Editor: Bob Bybee
P o l y L e t t e r
1437 Sugarwood Lane
Norcross, GA 30093
(404) 925-2480
PolyLetter is not affiliated with PolyMorphic Systems in any way.

\- - -

PUT ME ON YOUR MAILING LIST:
Name _____
Business _____
Address _____
City/State _____ Z _____
Phone _____
System _____
Printer _____
Uses _____
Future uses _____

(OPEN LETTER - from page 1)

include clearly stated information on what system configurations, ROM versions, or operating systems the software will work.

I understand that PolyMorphic does not currently have a list of their own end-users. But, such a list would help keep track of users qualified to obtain upgrades. It would also allow PolyMorphic Systems to make information on system changes, improvements and upgrades available to users on a regular basis through bulk mailings. A registration card should be included with every piece of hardware or software shipped by PolyMorphic Systems. The card would be completed by the user and returned directly to PolyMorphic Systems. The cards would make building the user data base extremely simple.

I should point out that that mailings to end users are not simply to provide information to the users. Many PolyMorphic users are isolated from active dealers or other users. They often have no idea of what is available to them from the manufacturer.

Your satisfied end user is always a potential customer for system enhancements. Information on new products directed at end users who own your machine is an ideal way to boost sales.

Please consider the software policy changes and the end user mailings. I believe they would improve PolyMorphic Systems' somewhat weak reputation on responsibility to and support of end users.

As much as I like my 8813 and the Poly operating system, I am never sure how to answer the potential purchaser of a Poly who calls with questions including: "How do you like your system?"; "How well supported is it by the dealer and the manufacturer?"; How much software is available for it?"; and "Would you recommend it for my use?"

```
10 PRINT CHR$(12),"SketchPad by Abstract Systems, Etc."
20 PRINT "A program to draw on the system screen."
30 PRINT "RETURN = ON, DELETE = OFF, ESC = Exit."
40 PRINT "Arrow keys to draw (ON) or erase (OFF)."
50 WAIT \ PRINT CHR$(12),
60 PLOT 0,47,0
70 IF W=13 THEN Z=1
80 IF W=127 THEN Z=0
90 IF W=17 THEN Y=MOD(Y+1,48)
100 IF W=18 THEN Y=MOD(Y+47,48)
110 IF W=19 THEN X=MOD(X+1,128)
120 IF W=20 THEN X=MOD(X+127,128)
130 IF W=27 THEN PLOT 0,47,0 \ Z=CALL(1027)
140 IF INP(0)=0 THEN PLOT X,Y,1 \ PLOT X,Y,0 \ GOTO 110
150 W=INP(1) \ PLOT X,Y,Z \ GOTO 70
```

POLY - ADS

Ads are published as a free service to PolyLetter subscribers.

WANTED: CP/M utilities & public domain software to run on 8" SSDD Poly. Will trade and/or supply disks. Will consider purchase of proprietary business software & utilities if guaranteed to run on Poly. Russ Nobbs, Rings & Things, P.O. Box 1753, Spokane, WA, 99210, (509) 624-8565, Source mail TCG256.

WANTED: Spare (low-cost) Keyboard II or III. Ralph Kenyon, 1686 West Main Road, Portsmouth RI 02871, Source mail TCI127.

WANTED: Old PolyMorphic 8810/8813 software and hardware manuals, including kit assembly instructions. J. H. McNally, PO Box 2336, Santa Barbara CA 93118-2336.

WANTED: Information on used Poly 88 equipment. Also need a Poly 88 User's Manual and hardware manuals. Edward Domingo, 2254 Spinnaker Court, Woodbridge VA 22192.

WANTED: Used Keyboard II in good condition. Contact Bob Bybee at PolyLetter.

FOR SALE: Poly 88 with 20K memory, keyboard, cassette and printer interfaces, Sanyo monitor. Excellent condition. $500. Jack Barone, (716) 837-5911.

FOR SALE: Poly 8813 with 2 drives, and IDS printer. I'm moving up to a DEC LSI-11. Martin Tuska, 710 Plummer Drive, Greensboro NC 27410, (919) 299-5442.

Last issue of PolyLetter asked for someone to write a patch to the operating system which prevents the screen from scrolling past lines longer than 64 characters. Ralph Kenyon has done this for the TYPE function and it is available for Exec/93, 94, and 95 (other versions on request). Send Ralph a system disk and $10.00 for this enhancement.

Ralph has also written a custom device driver which will prevent the printer driver from truncating long lines (up to 255 chars) when printing to the screen. (It also scrolls like the new TYPE above). This is also available for $10.00 (Order both of the above together for $15.00.)

MORE ON FREEZING THE SCREEN
by Russ Nobbs

ROM version 81 rev. A (for mixed 5" SSSD and 8" disks) contain slightly revised code for the WH1 screen output wormhole. PolyLetter has published methods for freezing the screen by POKEing a RETurn instruction (201 decimal) at 3108 and defrosting the screen with POKE 3108,205 (a CALL instruction).

The new wormhole normally has a JMP instruction (195 decimal) at that location. POKEing a RETurn continues to freeze the screen but a POKE 3108,195 is needed to return to normal.

Trying to run a program with the original POKE 3108,205 will cause the system to reboot to Exec... a most frustrating event when you expect to be running a program. If you want your program to run with BOTH old and new versions of the ROMs, you should do this:

```
1000 X9=PEEK(3108)  \REM save old contents
1010 POKE 3108,201  \REM freeze it
...and later,
2000 POKE 3108,X9   \REM restore whatever!
```

BUG IN "PACK"

When using command files on a system version prior to Exec/90, PACK may give you some trouble. The new System Programmer's Guide explains that on Exec's before /90, the system would not know if the command file got moved due to a PACK command in the file. This would only cause trouble if:

The command file was more than one sector long,

the command file contained a PACK command somewhere BEFORE the last sector of the file,

and the command file got moved as a result of the PACK.

If all of these conditions were met, the command file could fail somewhere after completing the PACK.

HELPFUL HINTS IN LAYMAN'S LANGUAGE

by Charles A. Thompson, Attorney
2909 Rosedale Avenue, Dallas, TX 75205

This issue, I thought you might find useful some information about how several of the utilities work (especially the more recent ones).

BACKUP.GO is a nifty utility which appeared with the new System Programmer's Guide earlier this year. It allows you to copy automatically each program on a disk or directory which has not been previously backed up. Each time you change a program, the "new" bit in the directory entry is set to indicate that the program is "new" (i.e., hasn't been backed up). Much earlier, Poly distributed FILMS to do file management but FILMS has numerous bugs and it's best not to use it. To use the program, enter BACKUP followed by the source disk (or directory) and the destination disk (or directory). For example:

```
$BACKUP 1 2
```

would copy every "new" file on Drive 1 to Drive 2, while

```
$BACKUP 1<WPS 2<BU<WPS
```

would copy every "new" file in subdirectory 1<WPS to subdirectory 2<BU<WPS. After a file is copied, the "new bit" is cleared by BACKUP.

A nice built in feature allows you to use "*" as the fourth element in a command, which (like SCOPY) tells BACKUP to "copy over" the previous version if one already exists on the destination disk or directory. So,

```
$BACKUP 1 2 *
```

would (a) pick up only the "new" files on Drive 1, (b) look at the Drive 2 directory, and (c) either copy over the existing version if it's already there or create a new copy. For "*" to cause copy over, the two files must be the same length. If they are not the same length, BACKUP will delete the previous version and create a new one at the next available disk space. (Incidentally, if you use just a drive number, BACKUP will backup all "new" files on that drive, including those in subdirectories.)

(NOTE: if you do not use "*", BACKUP will, each time it finds the new file already exists on the destination disk, either delete the old file on the destination disk and put a copy of the new one at the end of the directory or put a copy of the new file with a new name at the end of the directory. You will be prompted to make a choice each time this occurs.)

SETNEW.GO and CLEARNEW.GO work with BACKUP. These allow you (similar to how BIT, POP,TweakSys, etc., work with the "system" bit) to twiddle the "new bit" of directory entries. Use SETNEW to make an entry "new" so BACKUP will copy it. You can use multiple entries on a line, such as

```
$SETNEW Exec Gfid Dfnl
```

IMAGE is a much used feature of Poly Exec, but perhaps you didn't know you can remove the system disk in Drive 1 after entering IMAGE, and insert some other disk. The IMAGE will still be from Drive 1 to Drive 2 (or 3), but the substituted disk will be copied. Be sure to insert the other disk BEFORE answering the question "From Which drive". When you answer the question, IMAGE immediately reads the directory of the disk in that drive to determine how many sectors it will copy. The disk to be IMAGEd must be in the drive at that moment.

After the imaging is completed, you will be prompted to replace the system disk. This also works with Drive 4 on an MS.

It is not necessary to have an initialized disk to IMAGE to (but, in this case, you will be able to use only that portion of the disk which is imaged). In other words, IMAGE duplicates everything on the source disk, including the "overhead" information needed by the system. It images only that portion of the source disk which has programs on it (including deleted files) and then stops (which is why if you use an uninitialized disk as a destination disk, it will work, but you will have some unusable sectors on the end of the new disk unless you happen to be imaging a completely full disk).

Did you know that you can INIT the system disk? In later versions of Exec

(continued on page 5)

(around Exec/83 or so), you may specify that Drive 1 (or whatever the system drive is) contains the disk to be INIT'ed. If you should happen to leave in the system disk, Poly will tell you you have done it, ask if you really want to INIT it, and if you say "Y" you can INIT your system disk. This is mostly useful only in a single drive system (but it also means you'd best be careful lest you do something terminally tacky to your system disk).

COMP-DISK is one of those new utilities which comes with the new System Programmer's Guide. It allows you to compare one entire disk to another (useful after IMAGEing). It compares the disks byte by byte and will tell which bytes are different, if any. Not documented is the fact that, as with IMAGE, you may use it with a two-drive system even if neither disk has "COMP-DISK" on it. As with IMAGE, after you load the program, just remove the system disk, insert the two disks to be compared in the two drives, and answer the questions asked on the screen.

Finally, SCOPY (ver. 2.3, 09/23/80) can be used to do all of the following:

1. Copy any file (including a system file).

2. Copy over any file (including itself) so long as the destination file is the same length as, or longer than, the source file.

3. Do the above repetitively (without reloading SCOPY each time), which greatly speeds copying.

To do multiple copies, load SCOPY, then feed it the names of source files and destination files (you can use a command file to do this). The utility FUTIL uses SCOPY to do its thing.

To copy over existing files, add as the fourth element any printable character ("*" is handy to use, but anything will work with SCOPY). If the new file is longer than the old, an error message will result.

I'll cover more next issue. These items and many others (including all of the new features occurring since Exec/83 and BASIC COOL) are contained in my "Addendum to the PolyMorphic Manuals, 3d Edition (1 May 1981)", available for $6.00.

DISK-OF-THE-MONTH

Due to popular demand, this month the GAMES return! Thanks to Chuck Gross, Jim Holcomb, Russ Nobbs, Doug Schirripa and others who sent in their contributions. 8" versions of the Disk are now available, too! See ordering information below.

SLOT uses some very nice Poly graphics and lets you wager your hard-earned cash. (Would you believe 3 POLYs is a winner?)

BACKGAMMON also does fine graphics. It's a two-player game; the computer is only a spectator.

ARTIL is gunnery practice. An element of random chance is introduced by the wind, so this one isn't too easy.

MOON-LANDER lets you pilot the Lunar Module to a safe landing. (Or a crash-landing... depending on your skill.)

Sex-Appeal quizzes you about your physical attributes and gives you a score. On a scale of 100, how do you rate?

The Disk-Of-The-Month is supplied with Exec/83 and BASIC COOL. Price is $15 for SSSD 5", $20 for SSDD 8". Order from PolyLetter, 1437 Sugarwood Lane, Norcross GA 30093.

PolyLetter has been given a copy of an (unreleased) "Adventure" program for the Poly. It has some bugs, but it is also a lot of fun. We'll try and clean it up a bit for a future issue.

I also have three very useful items for users of WordMaster II. These are a "Table of Contents" (4 pages) and a complete 6-page cross-referenced topical "Index" to the WordMaster II manual, plus an 11-page "Command Summary" explaining every available command in the Formatter (including several not documented by Poly). In alphabetical order, the Summary gives each command and its proper abbreviation, parameter guidance, how it works, and limitations. These documents are attractively printed and easy to use. Price is $6.00 for the set of three items (not available separately). If you get an Addendum at the same time, send $10.00 for all four.

USING SUPER-ZAP

by Bob Bybee

Super Zap, also known as SZAP or Szap, is a very powerful utility program. It can also be very dangerous, and is probably one of the quickest ways to destroy a diskette (short of INIT)! This danger has led many users to be afraid of using Szap at all. In this article, we will describe some of the useful functions of Szap, and emphasize that it CAN be used safely.

Szap allows you to examine memory data or disk data. It displays data in a "page" of 256 bytes. This is equal to one sector of a Poly disk.

To execute Szap, you must first be in ENabled mode, then have a disk with Szap on it. ALWAYS USE A COPY, never an original, un-backed-up disk, when running Szap. You can easily lose the entire disk with one wrong keypress.

```
$EN
$$Szap
```

Szap will display a bewildering list of commands. We will summarize the most important ones here, so don't worry about learning them all at once.

After you get the Szap display, you are ready to begin examining memory. Press

```
:0
```

and RETURN. You will see the first 256 bytes of memory, which are in ROM on the CPU card. Notice the number in the upper right corner: this is the "page number" of the memory currently being displayed. This number is a four-digit hex (base 16) number, and it is also the address of the first byte displayed in the upper left corner. In this first page of memory, the number is 0000, and the upper left byte of the display is what is stored in memory address 0000. The byte to the right of that corresponds to address 0001. Since there are 16 bytes per row of the display, the bytes on the first row are from memory addresses 0000 through 000F. The next row is 0010 through 001F. The bottom row is 00F0 through 00FF.

This display arrangement makes it easy to locate any byte in memory. Suppose we wanted to find the value of the byte in address 0063. Starting at the upper left corner, count down to the seventh row (counting from 0 to 6), and the fourth byte from the left (counting 0 to 3). The byte at that screen position is the value in memory address 0063.

It is often helpful to see the ASCII equivalents of the bytes in the display. For example, the bytes 41 42 43 might be meaningless until you saw that they correspond to the ASCII characters "A B C". Szap can display these ASCII equivalents: press the ESC key. The right side of the display will become a set of characters, a translation of the bytes on the screen. (In page 0000 of the display, you won't see much text, since the ROM is mostly machine code.) To turn off the text display, so you can again see the page number, press ESC again.

To select a different page of memory for the display, you have 3 choices. Press RETURN to advance to the next page. Press LINE FEED to go to the previous page. Or, you can enter a page number directly, as a 4-digit hex number. To see the data in memory addresses 2000 through 20FF, enter:

```
/2000
```

and press RETURN. You are now looking at the first part of the memory area used for system overlays. Turn on the text display with ESC, and notice that the first four characters in this page of memory are the name of an overlay file (a file ending in .OV). This is how the system knows which overlay is in memory.

Until now, we have not tried to use Szap to change memory. Let's start learning how to be destructive...

Select a page of memory the won't be needed: somewhere in higher memory, say around 8000. Type /8000 and press RETURN. Turn on the text display with ESC. Notice the cursor on the first byte of the page: you can move it around with the arrow keys. If you move it out of the current page, Szap will automatically display the next (or previous) page.

Now, with the text display turned on, enter this:

```
41 42 43 44 45
```

(continued on pages 7, 8, 9)

Press a SPACE after each two-digit number. Notice the letters A through E appearing in the text display as you type. Notice the hex numbers you type are showing up in the cursor position, and the cursor moves to the next byte as each byte is entered. You are entering hex data bytes into memory.

Szap can also be used to enter text directly. Enter a single quote ('). Then type in some text, such as

THIS IS A TEST

The words you type are entered into memory, at the cursor position. The text display will reflect each character you type. To end this "text entry" mode, you must hit the ESC key.

Now you have learned how to enter hex numbers and ASCII characters into memory using Szap. One more nifty command before we leave this page of memory: pressing the "Z" key will ZERO the current page, beginning at the cursor and continuing to the end of the page. Try it.

Using Szap to examine and change memory is not too dangerous. Just remember that "user" memory begins at address 3200, and Szap uses several pages of memory while it's running. If you experiment with higher memory, say 8000 and above, you can't do any harm. You should not use Szap to change any memory below 3200, since this area belongs to the operating system... but feel free to examine memory in these areas. If you do accidentally zero something important, you can always re-boot by pressing LOAD. (If you zap something REALLY critical, the system may re-boot without your consent! Again, don't worry. This is harmless.)

Incidentally, try looking at page 0B00. This is the end of the ROMs, and in the ROMs in my system, Bob Martin autographed the last few bytes.

Szap works equally well to change and examine the data on diskettes. The first command you typed in was

:0

This selected "device 0," which is how Szap refers to system memory. You use the same command to select a disk drive. Enter

:1

and press RETURN. Szap will read the first sector of the disk in drive 1. (If you're using an MS and you don't have drive 1, use drive 4 instead.) And remember, we warned you to use a backed-up diskette for this exercise!

The first "page" of a diskette is one-fourth of the directory. Turn on the text display with ESC, and notice the disk name in the first few bytes of the display. The VERY first byte of this page is the directory checksum. It decides whether you get the greeting, "Disk Directory Destroyed!"

Elsewhere on this page, you will see the names of many of the system overlays: Exec, Gfid, Pack, and others. Compare the text display on the right with the hex display on the left. Notice that each filename is preceded by a hex byte, which may show up as a control character or some other character in the text display. This byte is called the "flag byte," and each file has one. It contains the bits which tell how long a filename is, and whether it is a system file, deleted file, new file, or some combination. Notice that each filename is immediately followed by its extention, 2 characters such as "OV" for overlays.

After the extention, the file directory entry has four important pieces of data. You will find that Szap is a very handy tool (and just about the ONLY tool) for altering these items. They are the File Disk Address (FDA), the file's Disk Number of Sectors (DNS), the Load Address (LA), and Start Address (SA). All of this information is displayed when you LIST a directory in ENabled mode. The DNS, or file length, is displayed in decimal by LIST. The other things are displayed in hex.

Each of these items is two bytes long. The lower, or least significant byte, comes first. The upper byte comes second. (The 8080 likes this byte-reversed storage.)

The FDA tells where the file begins on the disk. If the FDA is 04 00, then the file begins at sector 0004. This would be the

first file on the disk, since the directory occupies sectors 0 thru 3.

The DNS tells the file length. If the DNS is 07 00, the file is 7 sectors long. Remember that all these numbers are hexadecimal.

The LA tells where the file should be loaded into memory. Overlays (files whose extention is OV) are always loaded at 2000, so the Szap display shows 00 20. Likewise, the SA tells where to jump to begin execution of the file. The SA for most programs is equal to the LA.

Use the RETURN key to move to the next sector of the disk. This is the second sector of the directory. By the time you get to the second or third sector, you should be seeing files like BASIC, Asmb, FORMAT, and probably some of your own files.

You can move directly to any sector of the disk just like you moved to any page of memory. To look at sector 12 (hex), you would type

```
/12
```

and press RETURN. But there's another way to get to a particular location of the disk. The "I", or "indirect" command, also works when examining memory; but it really comes in handy when working with a disk. Pick a file from the directory, preferably a text file so you will recognize it. Find the FDA, which immediately follows the extention. Using the arrow keys, move the cursor to the FIRST byte of the FDA. Now press "I". The "indirect" command effectively says, "read the hex number from under the cursor, and move to that page." The cursor must be on the first byte of the number, not the second byte.

If you followed the instructions above, you should now have a display of the first sector of the file you chose. Note that the page is mostly normal text, with a few RETURN characters mixed in. RETURN has the hex value 0D, and shows up in the text display area as a squiggly "E". Press RETURN a few times to walk through the file. Notice that the end of the file is filled out with 00 bytes, which show up as the Greek "alpha" character in the text area.

One of my favorite uses for Szap is to make quick changes to a large file, without the need for editing the file. Editing requires deleting the old file, creating a new one, and possibly PACKing the disk... a nuisance if all you need to do is change one byte in the middle of the file! You can use Szap to enter text or hex data into a disk sector just as you did in memory. Since most files have a little space at the end of the last sector, you can also add a few characters to the end of a file with Szap.

The changes you make will appear on the screen immediately, but they DO NOT go onto the disk immediately. Szap only writes to the disk when you exit the program or display a different sector.

This gives you a chance to recover. If you accidentally hit "Z" while looking at a disk sector, the sector will be zeroed ON THE SCREEN. But the ruined sector will not go to the disk until you select another sector with RETURN, LINE FEED, the "I" command, or the "/" command, or if you exit.

Another good use for Szap is to "fool" the operating system. EDIT will only let you edit a normal text file: it refuses to edit any file whose SA and LA are not zero. If you have a BASIC data file containing fixed-length records, EDIT won't let you edit it. But, you can copy it, and use Szap to alter the directory so that the SA and LA are zero. Then, EDIT to your heart's content! I mainly use this trick to examine a data file, not modify it, since the edited file may not be suitable for BASIC anymore.

Szap is also useful for recovering a "destroyed" disk directory... depending on what did the destroying. The procedure is quite complex, and really should be the subject of a different article. I will note that Szap is smart enough to create a new checksum whenever it modifies the main directory of a disk. It DOESN'T know when you have altered a subdirectory, however, so you have to force Szap to create a new checksum. Do this by viewing the first sector of the directory and entering CTRL-C. Then write that sector back to disk. (Several utilities are available which assist in rebuilding or altering a disk directory. These include RECOVER.GO by Poly, and Rebuild.GO by Ralph Kenyon.)

EXITING SZAP: There are three ways to leave Szap. The normal way is

CTRL-E

This writes changes to the disk, if necessary, and clears the screen. This is the "clean" way to get out of Szap. The second way is

CTRL-Y

which, as usual, aborts the program. If you changed something in a disk sector but you DON'T want to actually write it back to the disk, use CTRL-Y. (The third way to exit Szap is to press the Load button!)

I have found Szap to be an invaluable tool for examining files and memory contents. And, in all my fiddling with Szap and the rest of the Poly system, I've only managed to destroy one or two disks. No tool can be as powerful as Szap without opening the door to a few risks, but common sense (i.e., BACKUPS!) will prevent any real damage from occurring. Let me know how you make out.

BB

HOW TO ERASE YOURSELF

This program was written by Bob Felts for PolyLetter. When executed, it erases ALL of memory, including itself. (In the Poly, this causes the system to re-boot.) You might use this program after editing a confidential file. Or it could be used in a custom "Auth" overlay, after determining that an unauthorized user is trying to access the system.

```
        ORG     3200H
        IDNT    $,$
        DI
        LXI     D,0
        LXI     H,LABEL
        LXI     SP,LABEL
LABEL   PUSH    D
        PCHL
        END
```

REENTRY MADE EASY

by Ralph Kenyon
Abstract Systems, Etc.

Many PolyMorphic programs take arguments on the command line. Arise, the old DLIST, TweakSys (also know as POP, Reset, etc.), SCOPY are examples. There are many others. One thing that can be done with these programs is to restart the program with new arguments. Suppose you were using the TweakSys program...

```
$TweakSys BASIC,Asmb (RETURN)
BASIC.GO        now system file.
Asmb.GO         now system file.
$
```

If you now key-in the following:

```
$START BASIC (RETURN)
BASIC.GO        now un-system file.
$
```

Notice that the program did its thing without having to be reloaded. Of course the shorthand

```
$ST BASIC (RETURN)
BASIC.GO        now system file.
$
```

works just as well.

(Cmdf Abort) Recovery

Have you ever hit CONTROL-Y while in the middle of executing a command file and then wish you had not?

If you are in Exec do the following:

$ENABLE	Enable the system
$$(CTRL-Z)	Enter front panel
L2D88 (SPACE)	Locate flag CMDF
FFG	Set non-zero, Go

If you are in BASIC and you have the >> prompt do the following:

$EXEC	Go to Exec
$$ENABLE	Enable the system
$$CON	Return to BASIC
>>CON	Resume program
(CTRL-Z)	Interrupt quickly
L2D88 (RETURN)	Locate CMDF
FFG	Set non-zero, Go

(If the BASIC Program has an ON ESCAPE active this won't work.)

PolyLetter

1437 Sugarwood Lane
Norcross, GA 30093
(404) 925-2480

20