PORT FE

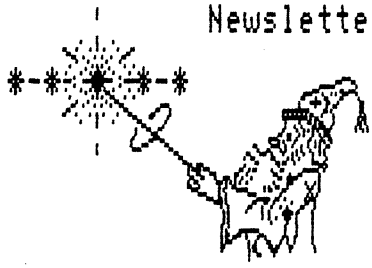
SORCERERS USERS' GROUP

(Toronto)

P.O. Box 1173 Sta. 'B'
Downsview, Ontario,
Canada. M3H 5V6

SORCERER

Newsletter



The Toronto Sorcerer Users' Group was founded in the Spring of 1979, a handful of willing and eager to learn members.

This newsletter shall at all times keep in mind the goal at its conception. To spread the seeds of knowledge.

Articles printed in this newsletter shall be free for all Sorcerer Users' groups to reprint or comment on as they see fit.

Articles submitted for this newsletter must be in no later than the beginning of the 1st of every month.

Oct./Nov. 1982 ISSUE

TABLE OF CONTENTS

GENERAL INTEREST

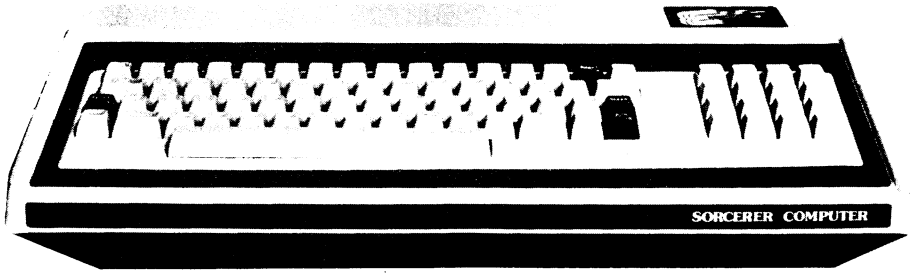
CP/M RELATED

MEETING PLACE

Location : Bathurst Heights Library - 7:00 PM 3170 Bathurst St.

Thur. Oct. 14
Thur. Nov. 18 Wed. Dec. 15

One block north of Lawrence on the west side of Bathurst.



The Prez Zez.

Well this month we have our elections, hope to see a good turnout this time. The new excecutive will be announced in the December issue of PORT FE.

---------- ---------- ----------

Let me apologize for the September issue being 'less than desirable', the mess was a combined effort of my printer ribbon "dying" and the printer having some clown doing the collating. We have included in this issue the first page for 'all' ,who knows there may even be someone out there who got that one. As for me, I was missing the first and ninth pages. That was ok for me because all I had to do was just dump them to the printer again, but what about everyone else ??... If anyone wishes to get a copy of any of the pages that they are missing just write and let me know, and we shall include that page with your next copy of the newsletter.

---------- ---------- ----------

Update ------> Re: Club policy

With reference to the recent inquiry sheet that was in PORT FE, the results are as follows:

1/ The CLUB NAME shall remain as is.

2/ PORT FE shall include more CP/M related material.

3/ Everyone wanted to leave things very much the same.

---------- ---------- ----------

PARTS FOR YOUR EXIDY

We have run across another source for some of the expansion boards for the Sorcerer. Note the following two names. The first I don't have an address for.

1/ South Valley Electronics John Parnell 408-727-0906
(They may be out of Exidy parts at this time,
if they get more who knows)

2/ Genius , (Byron Wagner)
6232 Santa Monica Blvd. Hollywood CA.90038 -- 213-462-1206

I understand that there was one source that was dumping disk drives for about $100 to $200 U.S. --" beware some of them are not even running "--

========== ========== ========

ON THE SUBJECT OF HORSE SHOES

One of our members (Tom Gottweis) was a very fortunate young man this past month. Tom walked into one of the local electronic supplier stores in town that deals in surplus parts and new gear as well. He ran across a little item that he had been wanting for (an eternity) well ever since he got his first disk drive that is. Thats right another drive, well Tom really couldn't afford one of the Exidy add one drives. Anyway he inquired as to the price, (this we have all done I'm sure, at one time or another) the sale clerk told him to make him an offer. Well Tom didn't want to insult the guy with an offer which he could afford so he said " I don't know , tell me how much you want for it " (Tom could only afford about $50 or $60 dollars).

At this point in time the sales clerk would not venture to say whether or not the disk drive would work or not. Tom looked over the drive controller card and saw that one of the capacitor leads was broken off, but also he noticed that the power hookup position didn't even have a socket soldered in place, suggesting that the drive never had power hooked to it. The blue resistance bridge was also missing.

The drive was an identical one to what he had in his Exidy FDS disk unit, a Micropolis 5.25 only this one was DOUBLE SIDED, ok so he waited for the clerks response..... Finally, he said "give me $ 70 for it and it's yours" could TOM be hearing things ???. He quickly got out $ 70 and handed it to the clerk.

Now then, he thought one of two things, (1) he had $ 70 worth of spare parts or (2) maybe a second (long sought drive), with 600K storage. He phoned me quite promptly, and inquired about bringing it over and checking it out. He was over within 30 minutes. I looked at the drive after he gingerly removed it from a plastic bag. Looked ok, from all physical points of view, with the exception of that broken capacitor. I replaced the .47 uf. Tantalum capacitor, next the power supply receptacle for the control card.

HORSE SHOES continued.....

Well we were missing that, so I knew I had some of the Molex 4 & 5 pin type connectors at the shop. So we hopped into my car and drove to the shop. Fifteen minutes later we had a four pin receptacle and a jumper with two plug ends which mate with the second power supply receptacle on the Exidy disk & power supply board. Back to my place where we were eager to test the new drive out.

We plugged in the new drive in place of the one that he had in his FDS disk unit and hoped for the best. plug in the controller cable and screwed it into place.

Now for the moment of truth...

We put in a disk and turned on the Sorcerer and the FDS drive. So far so good, ok now GO DF00 (cr).....

We didn't hear the familiar click, whir, clunk --- looked up at the screen and there was the CP/M sign on message. It worked, hurray.

Well we had to try it again to see why it was so silent. Sure enough it worked just fine. This made Tom's day for sure, the next day he got himself another control cable with two drive connectors, and the following day went to one of his other friends places and made up an expansion collar for adding the second drive right above the old drive. He decided to use the new drive as his "A" and the old drive as his "B", why not, it was a lot more quiet anyway. The old drive used to come up with BDOS errors once in a while, might as well have the new drive as "A" and give it a good work out. Next how about the other side ?? Well after trying a format on a new diskette, we checked it with STAT and as expected EXIDY flunked again. The format nor the configured CP/M 1.4 was not smart enough to recognize the second side. The disk controller used in the FDS drives can be used to read & write on both sides so now Tom is engaged in getting that other 300K on the other side of his new drive.

Tom just happened to be in the right place at the right time.

As the old saying goes, " A little sun must shine into one's life sometime"

Bargain hunters beware - Tom was wearing "horse shoes" when he entered the store.

by: H.A. Lautenbach

==========

QUIC-N-EASY

Next month I hope I will be able to bring you a review of QUIC-N-EASY by Standard Microsystems, Inc. I have brought it up on the Sorcerer with the help of EXMON2. Without this new monitor you could forget about it's existence.

From a brief glimpse at the structuring of the program which the magazines have reviewed, they state that it is better than the "LAST ONE" and much easier to use. From my trials (and errors!!!) I do agree with them.

There are two versions available, an everybody-can-use version and a professional programmers version. The difference between the two is that in the first version a lot of the things are very automatic, and the second (PRO version) it's not automatic. (this is the version I have)

The QUIC-N-EASY was written in Z80 machine language and 'C'. It has one runtime module and a file builder module. Also there is an install for configuring the program to various terminals. Of course like many of the newer programs written, there is not an EXIDY configuration file.

Since I cannot give you an indepth review at this time, I will outline the main functions (purpose) though. The program is mainly intended for the business world where file handling and data access is so important. It's power lies in the ability to let the user outline a screen of displayed information and define the fields afterward. The QUIC-N-EASY program then takes the information directly off the screen. The way one enters the information is very simple indeed, just cursor around the screen and type away. You put the information in the exact position (how you want it to look). Then cursor to the start of the field where you either want input or you want to display data to appear and hit a control key, up comes a Field Definition Menu and all you have to do is fill in the different questions that are asked. If the field contains an input operation then you can set up a PROCEDURE for that field to screen input or whatever. The documentation for this could be a little more detailed with more examples, but what can you expect with only having had a couple of hours with the program. It has and supports it's own language for all PROCEDURES that are set up.

Next month I will expand on the functions just a little more and maybe even know how to use it by then. Looks very good so far. Oh by the way this program only requires 48k to run.

by: H.A. Lautenbach

'C' LIBRARY ROUTINES by Dereck Gomes

First of all, allow me to apologize for the re-use of the third installment on the last submission; it was done by mistake. I don't know how it happened but it cost me a lot of work, since I also found out that I had somehow erased all my copies of the fourth installment. I had to write a 'C' program to recover the erased directory entry for this file on my disk. However, here it is.

Our fourth installment of 'C' library routines finds us looking at a method of obtaining input from the keyboard in a much more efficient manner than scanning the entire keyboard at every pass. Now we can check for only those keys we are interested in. It will be assumed that you understand the concept of using a "header file" that will substitute the values of the parameters listed below that correspond to the definitions of the keys.

Please note that some keys (the punctuation keys) have been spelled out as they mean something to 'C' and so cannot be used with their symbols. Listed first is the Header file that will contain the key parameter definitions. This is done in three columns to conserve space but of course you will use only one column at run time. These definitions consist of two parameters for each key; the first is the key column number and the second is the line number as shown in the Sorcerer 1 Technical Manual keyboard schematic.

This routine does not echo the character to the screen. Therefore it gives the programmer ultimate flexible control of applications.

HEADER FILE:

```
#define   _STOP      0,0   #define   _GRAPHIC   0,1   #define   _CTRL      0,2
#define   _SHIFTLOCK 0,3   #define   _SHIFT     0,4   #define   _CLEAR     1,0
#define   _REPEAT    1,1   #define   _SPACE     1,2   #define   _SKIP      1,3
#define   _ESC       1,4   #define   _X         2,0   #define   _Z         2,1
#define   _A         2,2   #define   _Q         2,3   #define   _1         2,4
#define   _C         3,0   #define   _D         3,1   #define   _S         3,2
#define   _W         3,3   #define   _2         3,4   #define   _F         4,0
#define   _R         4,1   #define   _E         4,2   #define   _4         4,3
#define   _3         4,4   #define   _B         5,0   #define   _V         5,1
#define   _G         5,2   #define   _T         5,3   #define   _5         5,4
#define   _M         6,0   #define   _N         6,1   #define   _H         6,2
#define   _Y         6,3   #define   _6         6,4   #define   _K         7,0
#define   _I         7,1   #define   _J         7,2   #define   _U         7,3
#define   _7         7,4   #define   _COMMA     8,0   #define   _L         8,1
#define   _O         8,2   #define   _9         8,3   #define   _8         8,4
#define   _SLASH     9,0   #define   _PERIOD    9,1   #define   _SEMICOLON 9,2
#define   _P         9,3   #define   _0         9,4   #define   _BACKSLASH 10,0
#define   _AT        10,1  #define   _RBRACKET  10,2  #define   _LBRACKET  10,3
#define   _COLON     10,4  #define   __         11,0  #define   _RETURN    11,1
#define   _LINEFEED  11,2  #define   _POWER     11,3  #define   _MINUS     11,4
#define   _KPLUS     12,0  #define   _KTIMES    12,1  #define   _KDIVIDE   12,2
#define   _KMINUS    12,3  #define   _K0        13,0  #define   _K1        13,1
#define   _K4        13,2  #define   _K8        13,3  #define   _K7        13,4
#define   _KPERIOD   14,0  #define   _K2        14,1  #define   _K5        14,2
#define   _K6        14,3  #define   _K9        14,4  #define   _KEQUALS   15,3
#define   _K3        15,4
```

ROUTINE #1:

COMMENTS. This routine scans the keyboard for a specific key. It does not wait for a key to be pressed but simply returns TRUE if the requested key is pressed.
CALLED ROUTINES: Outp, and inp. (BDS C Package routines).
USAGE EXAMPLE: if (scankey(_H)) putdma(4,5,"H key pressed");

```
scankey(outval, inbit)
        int outval, inbit;
{
        int bitpos;  bitpos=0x01;

        outp( 0xfe, outval );
        return( !(inp(0xfe) & (bitpos << inbit)) );
}
```

Dr. Dijkstra - thoughts to ponder on.

Programming is one of the most difficult branches of applied mathematics; the poorer mathematicians had better remain pure mathematicians.

The tools we use have a profound (and devious!) influence on our thinking habits, and, therfore, on our thinking abilities.

FORTRAN, "the infantile disorder", by now nearly 20 years old, is hoplessly inadequate for whatever computer application you have in mind today; it is now too clumsy, too risky, and too expensive to use.

```
/* LSTSET by Tom Gottweis */

MPISET: PROC OPTIONS(MAIN);

/*****************************************************************
*  This program is designed to set any software selectable      *
*  functions that a particular printer may support.             *
*  You will need the file DIOMOD.DCL and also PLIDIO.REL        *
*  in order to successfully compile and link  this program.     *
*  This program does not use sysin or sysprint in order to      *
*  it short, a similar effort written without the direct        *
*  CP/M calls was 14k long.                                     *
*   As the program is now set up, it will set the parameters    *
*  for an MPI 88G printer. It now allows a maximum of 30        *
*  commands and 30 separate form lengths.                       *
*  In order to set the proper form length for the MPI, a        *
*  bias must be added for the required selection in the form    *
*  length menu, currently it is 48. The commands are sent to    *
*  the MPI as 2 character ESCAPE sequences, these will need     *
*  to be changed to suit the individual printers that this      *
*  program will control. If you don't have PL/I, I don't        *
*  expect there should be very much difficulty in modifying     *
*  the com file with SID as the data areas are clearly laid     *
*  out by the compiler.                                         *
*  PLEASE NOTE, this program ,due to the depth of nesting of    *
*  the if then else structure, needs to be compiled with the    *
*  $F switch set in order to utilize the full compiler stack.   *
*                                                               *
*                                  Tom Gottweis                 *
*                                  4662 KKingston Rd. #115      *
*                                  West Hill, Ont.              *
*                                  M1E 4Y7                      *
*****************************************************************/
%INCLUDE 'DIOMOD.DCL';
%REPLACE maxline by 64;
%REPLACE numfunc by 18;
%REPLACE numpage by 16;
        DCL
        (c,i,j) fixed binary,
        1 d static ,
         2 maxsize bit(8) init('2'b4),
         2 inchars char(2) var,
        prfn(1:30) STATIC CHAR(64) VAR INITIAL(''),
        formln(1:30) static char(15) var initial('');

        prfn(1)='1)Carriage return.^M^J';
        prfn(2)='2)Line feed.^M^J';
        prfn(3)='3)Form feed.^M^J';
        prfn(4)='4)Start elongated printing.^M^J';
        prfn(5)='5)Stop elongated printing.^M^J';
        prfn(6)='6)Set unidirectional printing.^M^J';
        prfn(7)='7)Set bidirectional printing.^M^J';
        prfn(8)='8)Disable perforation skip.^M^J';
        prfn(9)='9)Enable perforation skip.^M^J';
        prfn(10)='10)Set graphics mode on.^M^J';
        prfn(11)='11)Set 6 lines per inch.^M^J';
        prfn(12)='12)Set 8 lines per inch.^M^J';
        prfn(13)='13)Set serif font.(graphics 75 dpi)^M^J';
        prfn(14)='14)Set 80 column font.(graphics 50 dpi)^M^J';
        prfn(15)='15)Set 96 column font.(graphics 60 dpi)^M^J';
        prfn(16)='16)Set 132 column font.(graphics 80 dpi)^M^J';
        prfn(17)='17)Set form length.^M^J';
        prfn(18)='18)RETURN TO CP/M^M^J';

        formln(1)='1)0"^M^J';
        formln(2)='2)3.0"^M^J';
        formln(3)='3)3.5"^M^J';
        formln(4)='4)4.0"^M^J';
        formln(5)='5)4.5"^M^J';
        formln(6)='6)5.0"^M^J';
        formln(7)='7)5.5"^M^J';
        formln(8)='8)6.0"^M^J';
        formln(9)='9)7.0"^M^J';
        formln(10)='10)8.0"^M^J';
        formln(11)='11)8.5"^M^J';
        formln(12)='12)9.5"^M^J';
        formln(13)='13)10"^M^J';
        formln(14)='14)11"^M^J';
        formln(15)='15)12"^M^J';
        formln(16)='16)14"^J^M';
```

```
      do while ('1'b);
         call conout('^L');
         do i=1 to numfunc;
              do j=1 to length(prfn(i));
                 call conout(substr(prfn(i),j,1));
              end;
         end;
         call rdbuf(addr(d));
         c=binary(inchars);

         if c=1 then call setlpt(13,00);
    else if c=2 then call setlpt(10,00);
    else if c=3 then call setlpt(12,00);
    else if c=4 then call setlpt(27,14);
    else if c=5 then call setlpt(27,15);
    else if c=6 then call setlpt(27,26);
    else if c=7 then call setlpt(27,25);
    else if c=8 then call setlpt(27,22);
    else if c=9 then call setlpt(27,21);
    else if c=10 then call setlpt(27,23);
    else if c=11 then call setlpt(27,20);
    else if c=12 then call setlpt(27,19);
    else if c=13 then call setlpt(27,30);
    else if c=14 then call setlpt(27,28);
    else if c=15 then call setlpt(27,29);
    else if c=16 then call setlpt(27,31);
    else if c=18 then call reboot();
    else if c=17 then
           do;
              call conout('^L');
              do j=1 to numpage;
                   do i=1 to length(formln(j));
                      call conout(substr(formln(j),i,1));
                   end;
              end;
              call rdbuf(addr(d));
              c=binary(inchars)+48;
              call setlpt(27,c);
           end;
    end;

setlpt:
      proc(contval,setval);
      dcl (contval,setval) fixed binary;
      call wrlst(ascii(contval));
      call wrlst(ascii(setval));
return;
end;

end MPISET;
```

==========

Simple Patching of Autost.com

This is just a simple note on how to patch the program AUTOST.COM to customize it to your systems needs. First off I will assume you have a basic knowledge of how the DDT program works. If you don't have this knowledge, read the chapter on DDT in the "CP/M Primer" by Murtha and Waite.

At any rate, use DDT to make the following changes to AUTOST.COM to customize it to your personal needs. Autost.com is located starting at 0100H. When you exit DDT you will type the command "SAVE 10 AUTOST.COM" to save the binary image of your patched version.

There are two possible changes to make in Autost.com that I have found (if you find more let me know!!). The first is the program that Autost.com will kick you into after it runs. The second is the sign-on message that appears after the program starts on auto-boot. To change the program name that Autost.com jumps to use DDT to go to the address location 0169H and merely place the CP/M filename of the program that you want to run here. For example, if you wanted to have Autost.com throw you into XDIR.COM after it boots up, you would use DDT to patch in "XDIR" beginning at location 0169H. To change the message that appears on the screen at sign-on (right after the "Big 0") then patch in your message starting at location 0180H. You will have only 20 bytes to work with and you can not go over without destroying the program!!!!! The characters that you see above what appears to be the bulk of the program are the graphics that appear on the screen as the "Big 0". Have fun creating your own custom disks and using Autost.com!!!

Byron A. McKay

```
;-----------------------------------------------
; AUTOCPM.ASM   version 1.0              10/12/81
;
; copyright 1981 by Thom Hogan
;
;              Basically Speaking Press
;              401 Fulton
;              Palo Alto, CA  94301
;
; This program loads and executes MBASIC.COM   (arbitrary file name)
;
;-----------------------------------------------
;
; EQUATES
;
clear       equ     12h     ;clear screen
escape      equ     1bh     ;escape character
home        equ     11h     ;home cursor
pbuff       equ     9       ;BDOS print buffer
bdos        equ     5       ;location of BDOS
cr          equ     0dh     ;carriage return
lf          equ     0ah     ;line feed
;
; START OF PROGRAM
;
            org     0100h
;
            lhld    01
            mvi     l,00
            mov     a,h
            sui     16h
            mov     h,a
            shld    ccp
            lxi     d,logo
            lxi     b,2047
over:       ldax    d
            sui     65
            stax    d
            inx     d
            dcx     b
            mov     a,b
            ora     a
            jnz     over
start:
            lxi     d,startgr       ;point to startgraphics
            call    print
            lxi     d,logo          ;point to message
            call    print           ;display it
            lxi     d,stopgr        ;point to endofgraphics
            call    print           ;display it
            lxi     d,endmes        ;point to loadmessage
            call    print
            lxi     d,filename      ;point to filename
            lxi     b,10            ;set counter
move:
            lhld    ccp
            mvi     l,07
            call    again           ;move it
            lhld    ccp
            mvi     l,88h
            mvi     a,08h           ;lsb of ccp pointer
            mov     m,a             ;put it in place
            lhld    ccp
            mvi     l,89h
            mov     a,h
            mov     m,a
            lhld    ccp
            pchl                    ;execute cold start
again:
            ldax    d               ;get byte to move
            mov     m,a             ;move it
            inx     h               ;increment location
            inx     d               ;increment location
            dcx     b               ;decrement counter
            mov     a,b             ;get counter in a
            ora     c               ;check if done
            jnz     again           ;...not done
            ret                     ;...done
print:
            mvi     c,pbuff         ;get proper call in c
            jmp     bdos            ;do it
```

```
;
; STORAGE AREA
;
ccp:        ds      2
filename:   db      04,'MBASIC',0,0,0,0,0
;                   ^    ^       ^
;              length  command  must be followed by
;           of command  string   zeroes to work properly
;
; note: if command string is longer than 8 characters,
;       you must change "lxi b,10" to "lxi b,length+2"
;
startgr:    db      clear,'$'
stopgr:     db      home,'$'
endmes:     db      cr,lf,lf,lf,9,'Loading MBASIC & CP/M...',cr,lf,lf,'$'
logo.       db      'YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaaKWWWWWWWWWWWWWWWWWWWWI'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY'
            db      'Iaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaKWWWWWWZaaaaaaaaaaVWWWWW'
            db      'WIaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY'
            db      'YYaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaWWWWWWaaaaaaaaaaaaaWWWWW'
            db      'WWaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY'
            db      'YYaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaWWWWWWaaaaaaaaaaaaaWWWWW'
            db      'WWaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY'
            db      'YYaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaWWWWWWaaaaaaaaaaaaaWWWWW'
            db      'WWaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY'
            db      'YYaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaVWWWWIaaaaaaaaaaaaKWWWWW'
            db      'WZaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY'
            db      'Zaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaaVWWWWWWWWWWWWWWWWWWWWZ'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
            db      'aaaaaaaaaaaaaaaaaaaaaaaaaaaa$'
end
```

==

A NEW List-routine for BASIC

The routine will print the reserved words with their First letter as a capital and the rest in lower case, so RESTORE will be printed as Restore. The routine is relocatable and can be put anywhere in free memory.

When the routine is loaded, enter SE O=xxxx where xxxx is the first address of the routine (i.e. 0000 in the listing below).

If you want to send your BASIC list's to your printer, then change at 0038: C3 93 E9 (Centronics) etc.

```
0000. CD A2 E1 E5 D5 F5 3A 4C 01 FE 99 20 28 21 00 00
0010: 39 D1 D1 D1 D1 D1 F9 EB 2B 7E FE 80 38 17 F1 FE
0020: 41 38 13 F5 FD 7E 6E BD 28 06 7D FD 77 6E 18 05
0030: F1 C6 20 18 01 F1 D1 E1 C3 1B E0 00 00 00 00 00
```

from. Per Danielsson (Sweden) P.S. Library tape on the way.

=====================================

Sorcerer Technical manual correction.

For those of you with Sorcerer Technical Manuals, there are 2 (two) errors in the hex to decimal conversion table at the back.

The hex equivalent of decimal 224 should be 'E' not 'D'

and '29672' should be '28672'

==========

ROM PAC BASIC BUGS

by: Dennis Wong

FIXING SOME OF THE BUGS INSIDE YOUR ROM BASIC

There are quite a number of known bugs inside your BASIC rom pac which were caused by 'careless' mistakes. These bugs have prompted many toolkits from various sources. Of course, in addition to removing the bugs, these kits also made available many desirable utilities.

However, after examining the disassembled listing of the BASIC rom pac, one would tend to think that the ultimate solution to a bug-free BASIC rom pac is to re-write the entire BASIC PAC.

In order to debug the BASIC rom pac the writer has chosen to relocate the entire basic to 5000H.

The following illustrate some of the bugs and the ways the writer used to fix them:-

1. PAUSE LISTING

In fact, BASIC has a provision to interrupt a listing process by checking the ESC/runstop key. If any of these keys are detected , BASIC will jump into a loop which requires any key to be pressed befor listing process resumes.

The solution is to change the byte in C701H to 1BH.

2. REDO FROM START

The reason this message never appeared on your screen is simple. On the way to printing this message, a stupid syntax check forces BASIC to print SN ERROR and break out of the INPUT loop. Putting four NOPs (00H) beginning CA39H will give you the proper ERROR MESSAGE.

3. PREMATURE CARRIAGE RETURN

The premature CR bug has plagued anyone who wants to use the quotation marks to send graphic patterns to the screen using the arrow keys to make up the desired pattern. The reason for this bug is that BASIC will bypass all CONTROL CHARACTERS, sending them to the monitor SEND routine.

At C596H, enter JP C,XXYY (DA YY XX where XXYY is the address of the subroutine we want to add).

AT XXYY - enter the following codes:-

```
FE 7F          CP DELETE
28 10          JR Z,L1-$
FE 08          CP BACKSPACE
28 0C          JR,L1-$
FE 01          CP LEFTARROW
28 08          JR Z,L1-$
FE 13          CP   RIGHTARROW
CA 99 C5       JP Z,C599H
C3 A8 C5       JP C5A8H
3A 8E 01   L1: LD A,(018EH)
3D             DEC A
32 8E 01       LD (018EH),A
C3 A8 C5       JP C5A8H
```

4. BUFFER OVERFLOW CHECK & CLEAR SCREEN HANDLING

The input routine of BASIC does not check for buffer overflow, but it does keep track of the number of characters that are being entered. For the sake of simplicity, the writer has directed the flow of BASIC to OM ERROR when activated by the buffer overflow condition.
After clearing the screen, the pointers will be reset and BASIC will jumpp to WARM START.
At C564H, enter CALL AABB (CD BB AA where AABB is the address of the subroutine we want to add).

At AABB , enter the following:-

```
FE 0C          CP CLEARSCREEN
28 07          JR Z,CLS-$
78             LD A,B
FE 40          CP BUFFERSIZE
C0             RET NZ
C3 09 C3       JP OM ERROR
CD 1B E0  CLS: CALL VIDEO
CD BA C9       CALL CRLF
C3 66 53       JP WARM   (SHOULD BE C3 66 C0) SHOWN IN RELOCATED POSITION
```

5. BREAK DURING INPUT

Another undocumented feature of ROM PAC BASIC is the break during input command which should be ^U.

FURTHER ENHANCEMENTS OF THE RELOCATED ROM BASIC

After relocating the ROM BASIC, you are free to experiment with any new ideas you have in mind.
We can make use of the free memory at the higher end of BASIC to add new COMMANDS/FUNCTIONS to the ROM BASIC. To achieve this we must relocate the TOKENS TABLE (C0F6H - C1E0H TOKEN 80H - C6H). Additional TOKENS can be tagged to the end using the format compatible with the TOKENIZATION subroutine (i.e. the first character of the keyword must have MSB set and the end marker of the TOKEN table is 80H. There are two locations that need to be changed along with the new token table:-

1. At C49CH, enter NEW TOKEN TABLE ADDRESS MINUS ONE
2. At C613H, enter NEW TOKEN TABLE ADDRESS

After relocating the ORIGINAL TOKEN TABLE, we can make use of the free space for tagging new FUNCTIONS to the VERB ACTION TABLE (C0C6H - C0F5H TOKEN AFH - C6H). So that our new keyword will have token value from C7H to FFH.

There is an error trapping routine in the execution driver (address C6BAH) which will send all token values greater than 9DH (new) to the SN ERROR routine. This part of the BASIC will have to be modified to handle the new token values.

In the expression evaluator, there is another routine to trap for the token values C4H,C5H and C6H (LEFT$,RIGHT$,MID$) because these are not unary functions. If we are to add new unary functions to BASIC then this part will also have to be modified.

The relatively small input buffer can be expanded to 254 characters (the last byte is reserved for a null) with little effort, by changing addresses C3D3H,C46EH,C522H,C53BH and C9BDH to the proper new input buffer address. Please note that C522H and C9BCH should contain the new input buffer address minus one. Furthermore, the buffer overflow check will have to be modified to reflect this change. Address C018H will have to be changed to the last byte address of the input buffer.

COMMENT

In the above discussion, the writer has purposely used the original BASIC ROM addresses. The reader should change all the addresses to their respective relocated addresses.
It is very educational to study the disassembled listing of the BASIC ROM because you will learn more about the powerful Z80 CPU. A good reference book on the subject is titled "MICROSOFT BASIC DECODED" BY James Farvour.
The writer believes that many fellow members are working on similar projects, and would be grateful if they too would share their findings with their fellow members.

EXMON2 - Exidy CP/M Changes by: H.A. Lautenbach
==========

We are still getting feedback of various programs that require changes, which make EXMON2 compatible with older software. Most of the changes are very minor in nature, some are just a little longer. In the following you will find the changes for Exidy CP/M,Exidy Sysgen,Exidy Copydisk and for those of you that have TD1.COM, the changes for that as well.

EXIDY CP/M SYSTEM IMAGE COMPATIBILITY CHANGES FOR EXMON2
(modify a system image file i.e. CPM48.COM and then you can SYSGEN the new image to a new disk A)SYSGEN CPM48.COM as an example)

CALL/JMP	addresses to be changed	ROUTINE	CHANGE TO
E1BA	22D5,22DB,22E7,22F3,22FD,235B,262C,2830	MSGOUT:	E1CF
E2ED	22E1,22ED,22F7	HCHOUT:	E206
E205	2300,234F,235E,270B	CRLF:	E1F8
E997	2713 (REMOVE PUSH AF PRIOR TO ENTERING THIS PRINTER CALL ROUTINE)	CENDR1:	E033
E9B1	282A	VIDINT:	E920

EXIDY CP/M SYSGEN.COM PROGRAM CHANGE FOR EXMON2

CALL/JMP	addresses to be changed	ROUTINE	CHANGE TO
E1E8	35B,368	ADDOUT:	E201

EXIDY CP/M COPYDISK.COM PROGRAM CHANGE FOR EXMON2

CALL/JMP	addresses to be changed	ROUTINE	CHANGE TO
E1BA	12F,1D5,1DF,1ED,1FB,207,21C,224,2E1,2FD 87F,885,891,89D,8A8,8BF,9C3,9D3	MSGOUT:	E1CF
E1ED	1F1,30D,318,88B,897,8A2	HCHOUT:	E206
E9E8	2AE	REC:	E96F
E9CC	2C5	WCUR:	E941
E205	8AB,8C2,9B7	CRLF:	E1F8

TD1.COM CHANGES FOR EXMON2

E1BA	106,113,131,137,152,176,193,1AB,1CD,1F1 20F,22D,284,2EA,305,371,3A3,3CD,413	MSGOUT:	E1CF
E13A	230	LINEIN:	E15F
E22F	32F	SCANHL:	E240
E264	33B	NAMFND:	E2C1
E232	347	SCANLT:	E243
E23D	34E,359	CONV:	E29B
E225	352	SCAN:	E233
E205	35D,8A3,8C2,923,942	CRLF:	E1F8
E28A	360	MOTRON:	E2E7
E71B	364	GETHED:	E7B7
E4CA	36E	FNDMSG:	E4C1
E6DE	374,3D0	HEDPRT:	E77A
E759	3A6	TAPWT:	E7EA
E6A9	3B0	BLKADJ:	E76A
E2DA	3B5	TAPEIN:	E354
E1D4	3B8	FINISH:	E1E8
E734	3DD	SKIPFL:	E7D0
E74E	3BF	CKCRC:	E365
E9E8	8C9,949	REC:	E96F
E2AF	3C7	MTROFF:	E30C
E9CC	8E0	WCUR:	E941
E453	3CA	FILHD:	E495

Dr.Dijkstra - thoughts to ponder.

PL/I --"the fatal disease"-- belongs more to the problem set than to the solution set.

It is practically impossible to teach good programming to students that have had a prior exposure to BASIC: as potential programmers they are mentally mutilated beyond hope of regeneration.

DATE: April 13,1982
PRODUCT: WordStar
CATEGORY: CORRECTION
Revision: All
PROBELM: ETX/ACK & XON/XOFF protocols through CP/M LIST DEVICE
REF. USER4, LISINP:, XON/XOFF, ETX/ACK, LST:, PRINTER

Description:

This bug only affects printers which are sent output characters through the standard CP/M list (LST:) device and the users wishes to implement the XON/OFF or ETX/ACK protocols.

The list device driver in USER 4 includes a patchable item called LISINP: which is necessary to provide a routine to retrieve the ACK character from the printer.

Unfortunately this routine is never called by WordStar. The fix is as follows:

```
A>SID WSOVLY1.OVR<CR>
  SID VERS 1.4
  NEXT   PC   END
  8600  0100  BCFF
  #S739B<CR>
  739B  FA CA<CR>
  739C  25 .<CR>
  #S7992<CR>
  7993  25 .<CR>
  #G0<CR>
A>SAVE 133 WSOVLY1.OVR<CR>      EAS
```

```
;DATE:     April 13,1982
;
;PRODUCT:  WordStar
;CATEGORY: MODIFICATION
;Revision: 3.0 only
;Subject:  SYSTEM FILE DISPLAY SUPPRESSION IN WS DIRECTORY
;
;  THIS PATCH WILL SUPPRESS DISPLAY OF ANY FILES ASSIGNED TO BE SYSTEM FILES
;  UNDER CP/M 2.XX (HAVEN'T TESTED MP/M) WITH THE COMMAND:
;
;               STAT FN.FT $SYS
;
;               PATCH IS FOR WORDSTAR 3.00      EAS  02/03/82
;
;    THE PATCH REPRESENTS 31 BYTES OF CODE IN THE USER PATCH AREA (USER1,
;    MORPAT:) ORGANIZED TO EXIST AT THE HIGHEST LOCATION AVAILABLE SO THAT
;    CURRENTLY EXISTING PATCHES WHICH DO NOT EXTEND PAST LOCATION 33CH WILL
;    NOT BE AFFECTED BY THE INSTALLATION OF THIS SYSTEM FILE SUPPRESSION.
;
;                              Enjoy,
;                                    Evan
;
DELENTX EQU     2C28H
DELENTY EQU     2C2BH
DTAB    EQU     2C69H
DENTPT  EQU     36D1H
DOSNX   EQU     2C8AH
NDTST   EQU     2BF6H

        ORG     33DH
;
MORPAT: PUSH    D
        INX     D
        LDAX    D
        ANI     80H
        POP     D

        JNZ     NODISP
        LXI     H,DTAB

        JMP     DELENTY

NODISP: POP     H
        POP     B
        XCHG
        LHLD    DENTPT
        XCHG
        PUSH    H
        CALL    DOSNX
        POP     H
        POP     D
        JMP     NDTST
;
        ORG     DELENTX
;
        JMP     MORPAT
        END
```

Some more changes follow on the next few pages.

We can thank the local BBS here in Toronto for supplying these changes to Wordstar. I'm sure that many of you can use some of these.

```
;DATE:      April 13,1982
;
;PRODUCT:  WordStar
;CATEGORY. MODIFICATION
;Revision: 3.0 only
;Subject.  Formatted output from printer driver
;
*************************************************
*       PRINT TO DISK WITH WORDSTAR 3.0         *
*************************************************
;
; THIS PROGRAM TAKES WHAT WORDSTAR SENDS TO THE LIST DEVICE
; TO A FILE CALLED FILEOUT.  THIS MEANS YOU CAN PRINT FILES
; DIRECTLY TO THE DISK AND THEN PRINT THEM LATER USING PIP
; OR AN EQUIVALENT.  ALSO GOOD FOR SEEING EXACTLY WHAT WORDSTAR
; DOES FOR ANY SPECIFIC PRINTER INSTALLATION.  INSTALL
; WORDSTAR FOR THE PRINTER OF YOUR CHOICE, NO PROTOCOL,
; AND CP/M "LST:" DEVICE DRIVER.  THIS PROGRAM USES
; PBGMEM, SO BE SURE TO SAVE ENOUGH PAGES TO HOLD THE
; PATCHES.
;
**************************
*       EQUATES          *
**************************
;
DEFBUF  EQU     0080H           ; DEFAULT BUFFER USED FOR FILE I/O
BDOS    EQU     5               ; LOCATION OF BDOS JUMP VECTOR
CLOSEF  EQU     16              ; CLOSE FILE FUNCTION
DELETEF EQU     19              ; DELETE FILE FUNCTION
MAKEF   EQU     22              ; MAKE FILE FUNCTION
WRITEF  EQU     21              ; WRITE SEQUENTIAL
SETDMA  EQU     26              ; SET DMA ADDRESS
INISUB  EQU     02A4H           ; ADDRESS OF INITIALIZATION SUBR IN WS 3.0
UNISUB  EQU     02A7H           ;    "     "  DEINITIALIZATION SUBR IN 3.0
PRINIT  EQU     070DH           ;    "     "  PRINTER INITIALIZATION SUBR
PRFINI  EQU     0710H           ;    "     "  "        DEINITIALIZATION SUBR
MORPAT  EQU     02E0H           ;    "     "  EXTRA PATCHING AREA IN ROOT
LISEND  EQU     071DH           ;    "     "  LST: PRINTER DRIVER IN WS
PBGMEM  EQU     035CH           ;    "     "  POINTER TO FREE MEMORY
MEM     EQU     7900H           ; WHERE THE ABOVE POINTER POINTS TO INITIALLY
;
        ORG     INISUB
        JMP     INITA           ; CHANGE BDOS VECTOR TO GO TO OUR ROUTINE
                                ; WHICH CHECKS FOR WS CHANGING DMA ADDRESS
                                ; SO WE CAN RESET IT AFTER WE WRITE OUR FILE
                                ; TO THE DISK.
;
        ORG     UNISUB
        JMP     DEINITA         ; CHANGE BDOS VECTOR BACK TO WHAT IT WAS
                                ; ORIGINALLY.
;
        ORG     PRINIT
        JMP     INITB           ; DELETE THE FILE IF IT EXISTS, AND CREATE
                                ; A NEW ONE.
;
        ORG     PRFINI
        JMP     DEINITB         ; WRITE LAST SECTOR TO, AND CLOSE FILEOUT
;
        ORG     LISEND
        JMP     PFILE           ; PRINT FILE TO DISK
;
        ORG MORPAT
INITA:  LHLD    BDOS+1          ; GET BDOS VECTOR
        SHLD    CBDOS+1         ; STORE IT FOR USE AFTER CHECKING FOR
                                ; CHANGE DMA ADDRESS
        LXI     H,TRANS         ; POINT TO TRANSLATE ROUTINE
        SHLD    BDOS+1          ; MAKE 6 & 7 POINT THERE SO
                                ; THAT WE CAN DO TRANSLATION
        RET                     ; AND RETURN
;
INITB:  MVI     C,DELETEF       ; DELETE "FILEOUT"
        LXI     D,FILE          ; IF IT EXISTS
        CALL    BDOS            ; NOTE, NO ERROR CHECKING
        MVI     C,MAKEF         ; CREATE THE FILE ON CURRENTLY LOGGED DRIVE
        LXI     D,FILE
        CALL    BDOS
        INR     A               ; IF NO DIRECTORY SPACE
        JZ      EXIT            ; RETURN TO OPERATING SYSTEM
        RET                     ; ELSE DONE
;
```

```
DEINITA:LHLD    CBDOS+1         ; POINT TO ORIGINAL JMP ADDRESS
        SHLD    BDOS+1          ; STORE IT BACK
        RET
DEINITB:LXI     D,DEFBUF        ; POINT TO BUFFER
        MVI     C,SETDMA        ; CHANGE DMA
        CALL    CBDOS
        LXI     D,FILE          ; POINT TO FCB
        MVI     C,WRITEF        ; WRITE IT OUT
        CALL    CBDOS
        ORA     A               ; ERROR IF NON-ZERO
        JNZ     EXIT            ; RETURN TO OPERATING SYSTEM
        LXI     H,DEFBUF-1      ; POINT TO 80H
        SHLD    COUNT           ; STORE NEW COUNT
        LHLD    NEWDMA          ; GET WS'S DMA ADDRESS
        XCHG                    ; PUT IN DE
        MVI     C,SETDMA        ; TO SET DMA AND
        CALL    CBDOS           ; CHANGE IT BACK
        MVI     C,CLOSEF        ; CLOSE THE FILE
        LXI     D,FILE
        CALL    CBDOS
        INR     A               ; CHANGED DISKS ON US?
        JZ      EXIT            ; IF SO, REBOOT
        RET                     ; ELSE DONE
TRANS:  PUSH    PSW             ; SAVE ACCUMULATOR
        MOV     A,C             ; GET BDOS FUNCTION NUMBER
        CPI     SETDMA          ; IF SETTING DMA, WE WANT TO KNOW WHERE TO
        JZ      STODMA          ;
        POP     PSW             ; ELSE RESTORE ACC
CBDOS:  JMP     0000H           ; AND JMP TO THE BDOS (ADDRESS PUT IN AT INITA)
STODMA: XCHG                    ; GET NEW DMA IN HL
        SHLD    NEWDMA          ; STORE IT FOR FUTURE USE
        XCHG                    ; RESTORE DE & HL TO ORIGINAL VALUES
        POP     PSW             ; RESTORE ACCUMULATOR
        JMP     CBDOS           ; AND JUMP TO BDOS
NEWDMA: DW      0000H           ; THIS IS WHERE WS'S DMA IS (PUT IN FROM
                                ; STODMA ROUTINE ABOVE
        ORG     PBGMEM
        DW      LEN + MEM       ; STORE NEW POINTER TO END OF MEMORY AT PBGMEM
        ORG     MEM             ; START WHERE PBGMEM ORIGINALLY POINTED TO
                                ; BECAUSE THERE'S NOT ENOUGH SPACE TO DO
                                ; ALL THIS IN MORPAT
;
; ACTUAL ROUTINE TO PRINT FILE TO DISK
;
PFILE:  LHLD    COUNT           ; GET COUNT
        INX     H               ; INCREMENT IT
        MOV     B,A             ; SAVE CHARACTER BEING PRINTED
        MOV     A,H             ; GET HI ORDER COUNT
        CPI     1               ; GONE TO 100H?
        JZ      WRITIT          ; YES, TIME TO WRITE A SECTOR
        SHLD    COUNT           ; NO, STORE NEW COUNT
        MOV     M,B             ; STORE CHARACTER
        RET                     ; AND RETURN
WRITIT: PUSH    B               ; SAVE LAST CHARACTER
        LXI     D,DEFBUF        ; POINT TO BUFFER
        MVI     C,SETDMA        ; CHANGE DMA
        CALL    CBDOS
        LXI     D,FILE          ; POINT TO FCB
        MVI     C,WRITEF        ; WRITE IT OUT
        CALL    CBDOS
        ORA     A               ; ERROR IF NON-ZERO
        JNZ     EXIT            ; RETURN TO OPERATING SYSTEM
        LXI     H,DEFBUF-1      ; POINT TO 80H
        SHLD    COUNT           ; STORE NEW COUNT
        LHLD    NEWDMA          ; GET WS'S DMA ADDRESS
        XCHG                    ; PUT IN DE
        MVI     C,SETDMA        ; TO SET DMA AND
        CALL    CBDOS           ; CHANGE IT BACK
        POP     B               ; GET LAST CHARACTER
        MOV     A,B             ; INTO A
        JMP     PFILE           ; AND START AGAIN
;
EXIT:   CALL    DEINITA         ; RESTORE BDOS VECTOR
        JMP     0000            ; AND REBOOT
COUNT:  DW      80H-1           ; INITIALIZE TO BEGINNING - 1
FILE:   DB      0
        DB      'FILEOUT ','    '; FCB FOR OUTPUT FILE
        DB      0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0
        DW      -1              ; TO MARK END
LEN     EQU     $-PFILE
END
```

MICROSOFT NEWSLETTER
MARCH-APRIL 1982

How do Microsoft Products Interface with CP/M Disk Files? We've had a lot of requests for more information regarding the manner in which Microsoft products function with CP/M when dealing with files. In the next few months we'll attempt to explain in detail how we deal with various types of files. First, a few ground rules: We're going to discuss version 2.2 and later of CP/M. Older versions (or the so called "look-alikes", including MP/M,) may behave differently! We'll begin with a review of CP/M's disk allocation schemes. While most of the disk file access routines are up to the manufacturer of the particular system you're using; there are some basic facts to keep in mind. The most important is that CP/M ALWAYS deals with files as collections of 128 byte records. CP/M ALLOCATES files in larger blocks, which may range from 256 bytes to 16K bytes. This "block" is the smallest unit of disk space that CP/M will allocate for any file. Finally, there is a unit of storage called an "extent", which is a multiple of the minimum allocation. For example, with a 1K allocation, the extent size might be 16K. When you run the CP/M program STAT (type STAT *.*), the program will tell you the size of each of the files on your disk, in terms of the number of records, the number of extents, and the size of the file in bytes (always a multiple of the minimum allocation block). CP/M handles the "mapping" of physical sectors to logical 128 byte records through a "blocking/deblocking" algorithim, which can be treated as a "black box" for all but the most esoteric applications. Finally, when we start trying to compute how much space we have remaining on a given disk; we're also limited by the number of "directory entries"; since the CP/M system allocates a fixed space on the disk for the directory. To obtain information about the physical characterisics of your particular disk system you can use the STAT DSK: command. I hope this provides a good introduction to the CP/M file system. Next month, we'll discuss sequential files and how they are handled by MBASIC, FORTRAN, and COBOL.

==========

Microshell is a package that simulates the Unix shell on a CP/M system, with such features as redirected input (BDOS calls 1, 6, 10, and 11), redirected output (BDOS calls 2, 6, and 9), shell scripts and pipes. Microshell completely replaces the CCP of the CP/M, so it also does all those commands, plus a few extensions, like 32 user areas, automatic search of 3 disks for .COM files, and an integrated SUBMIT and XSUB facility. The user can also define his/her own prompts, upper/lower case filenames, echoing commands during the scripts (.SUB files), any control characters within the scripts, and logging in disks without rebooting.

Pipes, in case you dont know, are nifty things. You can redirect the output of the DIR command to a file, like DIR *.* >FILES and get a file called FILES that has all your disk files on it. Do this twice a few days apart and use SSED on the two files and you get a list of the files that are new or missing between the two times. More elaborate are programs that intentionally massage the data, like sort the names into alphabetic sequence, or eliminate duplicates, and you get very fancy things. For example, to print your directory,
just enter DIR | PIP LST:=CON:

It does have disadvantages, such as wiping out an extra 9K from your system, and there are no UNIX-like filters, such as CAT or SRT. I am not a real UNIX expert, but Microshell sure makes CP/M a lot more user friendly like all those UNIX freaks keep complaining about. Microshell does work with all Digital Research programs except MOVCPM, and with all other programs I have been able to test, including BYE, MODEM, SD, XD, Wordstar, Microsoft Basic, CBASIC, EBASIC, and dozens of the Public Domain CP/M programs. The principle exceptions are those programs that overlay the CCP and BDOS. In that case, Microshell just goes away and you are back to regular CP/M.

If you need that extra 9K that Microshell gobbles up, there is a script given that will run a program under regular CP/M and then re-instate Microshell when done. The script files cannot be nested, but Microshell itself can be to simulate the same thing. The only thing lacking is lots of freebie filters for the users, and some of the filters available on UNIX should be written for us CP/M folks anyway, like a good general sort.

Microshell is sold by New Generation Systems for $150 and I think it is worth the price for anyone but the confirmed hacker that has completely dis-assembled his CCP anyway and re-wrote MAC in 1978. Contact New Generation Systems at 2153 Golf Course Dr, Reston VA 22091 or see recent ads in Microsystems or BYTE.

Chuck Weingart

Subject: CP/M 2.2 patch for backspace From: LEWIS MOSELEY

New patch to CP/M V2.2 BDOS. Causes both <DEL> and <BS> to be treated like a <BS>, i.e, to echo a <BS> instead of the character. Patch this into your BIOS source code:

```
        ORG     CCP+0A1BH
        JMP     CCP+0A07H
        DB      0,0,0,0,0,0,0,0
```

NOPs to keep it pretty. Thats all there is to it!

==========

FLOPPY DISK WEAR.

Well Hi;

It has come to my attention that some of you newcomers to this great area of human endeavour have not yet got the word on floppy disks. Care, feeding, and brand selection.

First a word about myself. I have been piloting a micro since 1976 and had a mini-floppy since '77, 8" since last year. I have worked in the microcomputer business since 1976 and have handled (ordered, sold, shipped, used, etc.) in excess of 50,000 floppy disks of all types. I have spoken to and yelled at and been cursed by Rep's from every major brand (and most of the minors) of floppys made.

Conclusions

For every floppy disk drive, there are brands of diskettes made that will not work right, the same for brands of computers, operators, locations, etc. If 99 people think brand X is wonderful, the next user thinks they are the worst thing since "Heavens Gate". If 500 people have tried brand "V" and been burned, there are 50 who think they're wonderful. There ain't no accountin' for taste.

Howsomever, after having felt 50,000 plus diskettes carrying the 3M/SCOTCH lable slide through my fingertips, I can honestly say that I had no more than 100 diskettes come back defective. Run that through your HP-97s' and figure a percentage of returns and you get a mighty small number. (we won't talk about the 10 cases that where labled soft sector when they were really 16 secter... the disks where OK, just the lables were bad)

Instructions; If you got to keep cleaning your heads, change brands! If you haven't cleaned the machine since '77 and it still works, Don't Fool With It.

Keep'em Rotatin', Doc Watson.

==========

From: Frank J. Wancho modified by H.A. Lautenbach Oct./82

For those of you trying to use UNSPOOL and need to implement a LSTSTAT in their USER portion of their CBIOS, the following code is what I wrote and use for the CP/M for N* from N* in Z80 code. It can easily be adapted to whatever printer port assignments you have Note that it also handles XON/XOFF flow control from the printer - and was the other reason I needed to write it in the first place... (the code in lower case is what was added)

```
;
;INPUT FROM SECONDARY (LEFT) SERIAL PORT
CINR:   IN      A,(0FDH)        ;GET STATUS FOR LEFT SERIAL PORT
        AND     0               ;RCVR READY ?
        JR      Z,CINR          ;NO, LOOP TILL RECEIVED
cinr1:  IN      A,(0FCH)        ;GET CHAR FROM DATA PORT
        AND     7FH
        RET
;
;OUTPUT TO SECONDARY (LEFT) SERIAL PORT (PRINTER)
COUTR:  ld      a,(xonwait)     ; Check if XON pending
        or      a
        jr      z,coutr2        ; Zero if it is not pending
coutr1: call    cinr            ; Wait for character
        cp      11H             ; Is it ^Q?
        jr      nz,coutr1       ; No, loop
        xor     a               ; Make a zero
        ld      (xonwait),a     ; Yes, clear flag
coutr2: IN      A,(0FDH)        ;GET USART STATUS
        AND     1               ;IS TX READY?
        JR      Z,COUTR2        ;LOOP TIL READY
        LD      A,C             ;BYTE FROM C TO A
        OUT     (0FCH),A        ;SEND BYTE VIA DATA PORT
        in      a,(0FDH)        ; Get status
        and     0
```

```
        ret     z               ; Nothing pending
        call    cinr1           ; Otherwise, get character
        cp      13H             ; Is it a ^S?
        ret     nz              ; No, just return
        ld      a,0ffh
        ld      (xonwait),a     ; Yes, set flag
        ret
;
; XON/OFF IMPLEMENTATION OF PRINTER TEST
PRTTST: ld      a,0             ; Check if XON wait is active
xonwait equ     $-1
        or      a               ; Zero means NOT WAITING
        jr      z,prtst1        ; Skip rest if not waiting
        in      a,(0FDH)
        and     0               ; Check for RCVR Ready
        ret     z               ; Return NOT READY
        call    cinr1           ; Otherwise, get the character
        cp      11h             ; Is it a ^Q?
        ld      a,0             ; Pre-load zero (NOT READY)
        ret     nz              ; Not ^Q, thus still NOT READY
        ld      (xonwait),a     ; Clear flag
prtst1: in      a,(0FEH)
        and     1
        ret     z               ; XMIT buffer not empty yet
        ld      a,0ffh          ; Load READY
        ret
;
```

==========

GUIKIES Version: 3 Editor: Roy Lipscomb, Logic Associates, Chicago

ITEM: Using the I command of DDT to read files from disks other than A.

PROCEDURE: Translate the desired drive's name into its alphabetic sequence number; for instance, "1" for "A", "2" for "B", and so on. Then perform the following sequence of commands to install the file's name and drive number, and read the file into RAM:

```
-Ifilenam.typ<cr> (desired file's name)
-S5C<cr>
005C 00 x<cr> ("x" = the above sequence number)
005D yy .<cr> ("yy" = don't care)
-R
```

SOURCE: Hal Walchli, Pittsburg Area Computer Club Newsletter, June 17, 1981. (Also seen in other newsletters.)

==========

ITEM: Fix for bug in SUBMIT.COM (2.X only)
DESCRIPTION: SUBMIT will not accept pseudo-control characters.

DIAGNOSIS: Create a file X.SUB consisting of one line -- PIP X.SUB=X.SUB[S^Z]

where "^" and "Z" are separate characters. Then issue the command SUBMIT X. The faulty version of SUBMIT will display the message "Error on line 001 Invalid Control Character," then exit to CP/M.

FIX: Execute the following procedure:

```
A>DDT SUBMIT.COM<cr>
DDT VERS 2.2
NEXT  PC
0600 0100
-S442<CR>
0442 61 41<CR>
0443 32 .<CR>
-^C
A>SAVE 5 SUBMIT.COM<CR>
```

RETEST: Issue the command SUBMIT X. No error should occur, and X.SUB should remain unchanged.

SOURCE: Dave Marschall of AMSI.

==========

ITEM: Counting records in a file.

PROCEDURE: PIP CON:=filename.typ[ND6]

EFFECT: The "virtual" record numbers for each record will be output to the screen. The last number will be the record count of the file. (Records are assumed to be terminated by a carriage return.)

EXTENSION: Instead of CON:, you can substitute any valid CP/M filename, and the "count up" will be entered into that file instead. (Numbers under 6 digits are preceded by blank fill.) This type of file can be useful during program testing.

This option reveals a slight anomaly in PIP: if a line begins with tabs, those tabs will not be deleted but will be retained and appended to the line number.

SOURCE: Roy Lipscomb, Logic Associates

========== ==========

ITEM: Execute any commands by pressing only 2 keys.

REQUIREMENT: Public domain program SYNONYM.COM, version 2 or later, available on various RCPMs.

PROCEDURE: Select an unused alphanumeric, or other character allowable as part of a file name. As an example, let's take "A". Let "command" be a command-plus-parameters. Issue this command:

SYNONYM A command

Now whenever you wish to execute "command", just A and carriage return.

EXTENSION1: To create a single name for a sequence of several commands, 1) place the commands into a .SUB file (eg, TESTIT.SUB), and 2) follow the above procedure to create a name for the command "SUBMIT TESTIT".

EXTENSION2: Better yet, use /.COM or SUPERSUB.COM to eliminate the need for a separate .SUB file. That way, all the commands to be executed can be coded as part of "command" above.

SOURCE: Roy Lipscomb, Logic Associates

========== ==========

Dr. Dijkstra - revalations of the mind.

The use of COBOL cripples the mind; its teaching should, therefore, be regarded as a criminal offense.

APL is a mistake, carried through to perfection. It is the language of the future for the programming techniques of the past: it creates a new generation of coding bums.

The problems of business administration in general and data base management in particular are much too difficult for people that think in compounded with sloppy English.

About the use of language: it is impossible to sharpen a pencil with a blunt axe. It is equally vain to try to do it with ten blunt axes instead.

Besides a mathematical inclination, an exceptionally good mastery of one's native tongue is the most vital asset of a competent programmer.

Many companies that have made themselves dependent on (and in doing so have sold their soul to the devil) will collapse under the sheer weight of the unmastered complexity of their data processing systems.

We can found no scientific discipline, nor a healthy profession, on the technical mistakes of the Department of Defense and, mainly, one computer manufacturer.

The use of anthropomorphic terminology when dealing with computing systems is a symptom of professional immaturity.

By claiming that they can contribute to software engineering, the soft scientists make themselves even more ridiculous. (Not less dangerous, alas!) In spite of its name, software engineering requires (cruelly) hard science for its support.

In the good old days physists repeated each other's experiments, just to be sure. Today they stick to FORTRAN, so that they can share each other's programs, bugs included.

Projects promoting programming in "natural language" are intrinsically doomed to fail.

EXMON2

A NEW MONITOR FOR THE SORCERER

At last, here's a revised version of Exidy's operating system that has full terminal functions. EXMON2 will increase the flexibility of your Sorcerer computer and open the door to many excellent software applications that require special terminal features.

These New EXMON2 routines can be used directly from your Sorcerer keyboard and are easy to use in BASIC programs:

- **Direct cursor positioning**
- **Clear to end of line**
- **Clear to end of screen**
- **Delete line**
- **Text highlight on/off**
- **Reverse ASCII characters**

EXMON2 has many additional features that give your Exidy Sorcerer even greater flexibility:

- **Define and *reserve* top of screen**
- **Search memory for hex or ASCII string**
- **Parallel *and* serial printer drivers *with* or *without* line feeds**
- **A *properly* working serial port (for modems, etc.)**
- **A *f-a-s-t* keyboard status routine that's CP/M compatible**
- **And there's *more* . . .**

Each EXMON2 set comes complete with two burnt-in, fully-tested Eproms and an accompanying user's manual with easy-to-understand installation instructions.

1 set (2 Eproms & manual) . . . **$65.00** U.S. + postage*

Group discounts { (5 to 9 sets) . . . **$55.25** U.S. + postage* / (10 or more) . . . **$52.00** U.S. + postage*

***Add $5.00** U.S. with the first set, **$3.50** U.S. with **each additional** set for postage and handling.

Please make cheques payable to:
H.A. LAUTENBACH, and mail to: **P.O. BOX 1173, STN. B, DOWNSVIEW, ONTARIO, CANADA M3H 5V6**

TO ORDER EXMON2:

_____ Please send more information on EXMON2.

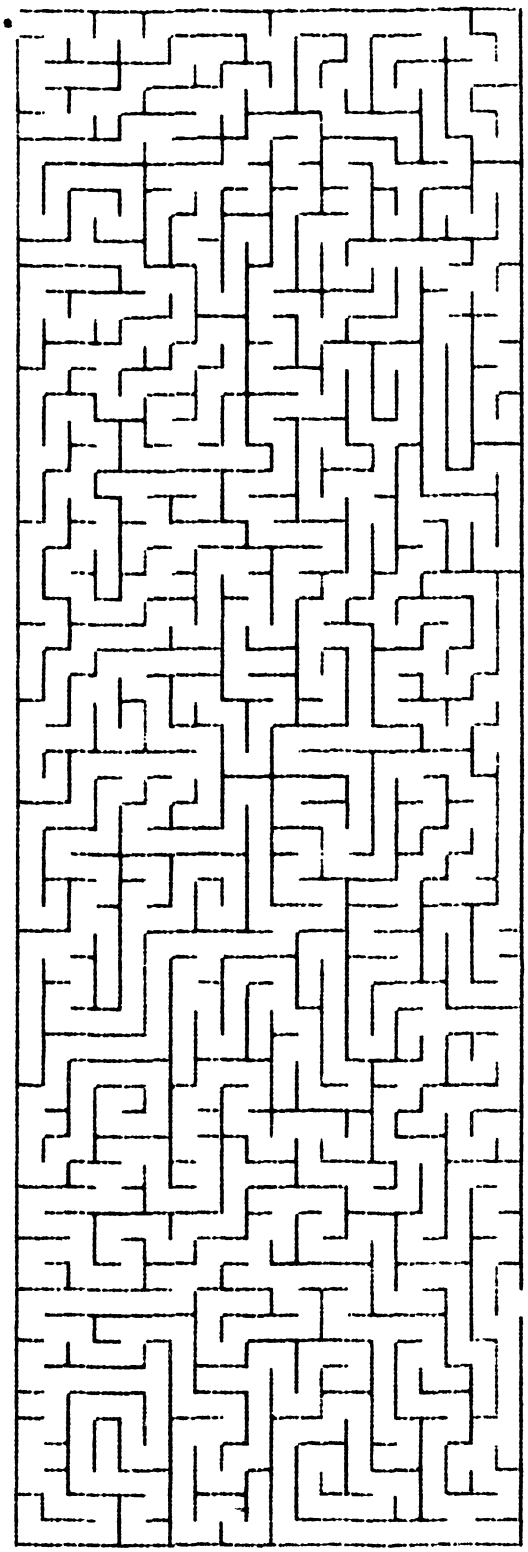
I have enclosed my certified cheque/money order for $_____; please send me ____ EXMON2 set(s).

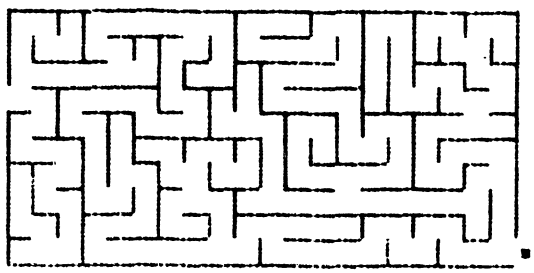
Company/Club _____

Name _____

Address _____

_____





SORCERER USERS' GROUP (TORONTO)

Membership Application Form — Covering Jan. to Dec.1983

Membership to the group is not restricted to the TORONTO area. All persons willing to participate are invited to join.

As a member of the Sorcerer Users' Group (Toronto), I enclose the annual membership fee and agree to the following Terms.

1. That I will not, without the authorization of the board of directors, represent myself or take any action as agent, or representative or become spokesperson of the group.

2. That I will not use any software obtained from the SUGT library for any commercial purpose or financial gain. The library shall be available to me should I wish to obtain programs donated by other members. These programs shall not be distributed without the owners consent and/or the consent of the board of excecutive officers.

3. That I have the right to vote for the officers and directors of the organization at the annual general meeting.

4. That any breach of the above conditions and any other restrictions that the Officers of the Club may invoke in the future on my part may result in suspension or termination of my membership without refund.

Annual Membership Rates: (Jan - Dec)

Canadian - $15.00 Cdn - PLUS $6.00 Postage
U.S.& Foreign $15.00 (U.S Funds) PLUS $10.00 Postage

Payable to - SORCERER USERS' GROUP (TORONTO) - by Cheque or Money Orders.

The SUGT program library is available to all members in the following manner.

You may send $8.00 + $1.50 postage for each volume as they become available and we shall supply the cassette/s. Program cassettes shall be sent via Air Mail.
All issues of PORT FE shall be mailed first class, in the case of non local issues, they are mailed via Air Mail. Past issues of PORT FE are only available for the current calendar year. Contact the editor, he will advise the amount of payment for previous issues.

NAME(print):.....

ADDRESS:.....

CITY:.....

POSTAL CODE:.....

TELEPHONE: Res..... Bus.....

Payments enclosed (membership):..... Library tape/s.....Vol 1 or 2

Signature:.....

Please list the type of equipment you are using etc...

Sorcerer size: 8... 16... 32... 48... other..... S100... Graph board.....

Disk system - Micropolis..... Discus.... Exidy.... other... Size.....

Other Equipment .....
.....
.....
.....
.....

If you belong to any other Sorcerer Users' Group please list it below.
.....