PORT FE

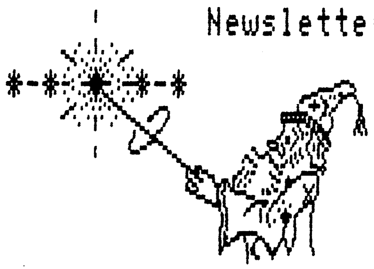
SORCERERS USERS' GROUP

(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.

April 1982 ISSUE

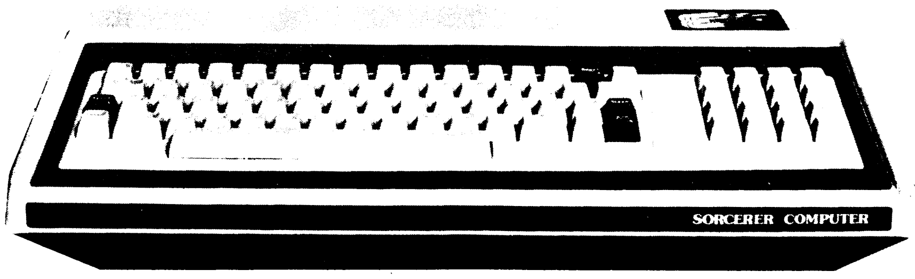
TABLE OF CONTENTS

MEETING PLACE

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

April : Wed. 14th. May : Thurs. 13th. June : Wed. 16th.

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



'C' LIBRARY ROUTINES   by Dereck Gomes

This column will hopefully be a monthly part of this newsletter. It is intended to provide a forum for 'C' programming language users' exchange of 'C' library routines. It is assumed that users have the BDS C package or at least access to the library routines that come with that package. If anyone has some 'C' routines that they can share with the rest of us then please contact me at 493-1270. The submissions must be in source listings and follow the general format shown below. I reserve the right to make cosmetic changes to the style of the routine. Please note that the length of the name of a variable bears no relation to the amount of bytes used in the final CRL or COM file; only in the source listing. Therefore, don't be cryptic in naming your variables: call a spade a spade. Also, since comments are removed prior to compilation, comment liberally. I will endeavor to include about two routines every month. Here, now, are examples of typical routines. Study their format.

ROUTINE #1:

COMMENTS: This routine puts a null terminated string pointed to by pointer <str> onto the screen at location column <col> and line <line>.
CALLED ROUTINES: None.
USAGE EXAMPLE: putdma(12, 5, "This is a string");

```
putdma( line, col, str )   /* put line on screen */

        int  line, col;
        char *str;

{
        char *spot;

        spot = 0xf080 + line * 64 + col;
        while (*str)
             *spot++ = *str++;
}
```

ROUTINE #2:

COMMENTS: This routine puts a character <c> onto the screen at location column <col> and line <line-no>.
CALLED ROUTINES: None.
USAGE EXAMPLE: putdmc(12, 5, c);   where c is char. variable

```
putdmc( line_no, col, c )   /* put character on screen */

        int  line_no, col;
        char c;
{
        char *video;

        video = 0xf080 + line_no * 64 + col;
        *video = c;
}
```

TWO SHORT PROGRAMS FROM SOUTH OF THE BORDER

Jean Pickett of Hamiltom New Jersey sent us these two programs written in XITAN BASIC and they work as is.

```
110 PRINT CHR$(12):F=INT(RND(1)*64+128):FOR I=0 TO 15 STEP .1:X=INT(RND(1)
*I*2):Y=INT(RND(1)*I):POKE &HF080+(32+X)+64*(14-Y),P:POKE &HF080+(31-X)
+64*(14-Y),F:POKE &HF080+(31-X)+64*(15+T),P:FOKE &HF080+(32+Y)+64*(15+Y),F:
NEXT I:GOTO 110

120 PRINT CHR$(12):FOR I=0 TO 15 STEP .1:X=INT(RND(1)*I*2):P=INT(RND(1)*64+128)
:Y=INT(RND(1)*I):POKE &HF080+(32+X)+64*(14-Y),P:POKE &HF080+(31-X)+64*(14-Y),P:
POKE &HF080+(31-X)+64*(15+Y),P:POKE &HF080+(32+X)+64*(15+Y),F:NEXT I:GOTO 120
```

The only change that was required was to change POKE &F080 to POKE &HF080

Thankyou Jean for the programs.

Converting Rom-Pac Basic Programs to Compiled MBASIC Files

In my collection of Rom-Pac programs are a few gems that stand out amongst the rest but are too slow to really be enjoyed. One example is TREK7 by Paul Tan. It is a great Basic Star Trek but not as fast as some of the machine language versions.

I ventured to convert this program into an ASCII file for MBASIC. From this I could edit the file then compile it into machine language using Basic Compiler from Microsoft. Using the following information you should be able to convert any Sorcerer basic program into a .COM file.

First we must get the tokenized basic source onto disk in an ASCII format:

- Turn off the S100 expansion unit - this disables the disk controller boot ROM from memory leaving it open for the basic ROMPAC. If you own a Micropolis drive at BC00 then this is unnecessary.
- Insert the basic ROMPAC , turn on the computer and load the program from tape which you want converted. Now 'POKE 322,255' while in the command mode so the basic will not insert carriage returns at the end of a line.
- At this time you should check the program for packed graphic strings - these are strings that contain graphic characters. You must translate the string to CHR$(x) format.

```
ex- A$="graphic A graphic B"        --> Sorcerer basic
   \
    \__ A$=CHR$(154)+CHR$(155)      --> BASCOM
```

Insert a blank tape into the recorder. Enter the monitor with 'BYE'. Type 'SE O=S' then 'PP' to return to basic.

You won't be able to see anything on the screen at this moment because all output is going to the tape. Now turn on the tape recorder and type 'LIST'. Listen to the output of the computer (anyway you can) while the recorder is taping the program. When you hear the single tone signalling it is done, hit Control 'Z' on the keyboard. This sends an EOF byte needed later when PIP is loading the program to disk. Turn off the tape recorder. The program is now in ASCII format on the tape.

Turn off the computer and remove the ROMPAC, turn on the S100 expansion unit and boot up CP/M. Now do the following:

```
A>SID PIP.COM

#GE003  - exit to the monitor
```

```
>EN 103

0103: C3 10 01/          ---->> Set input routine to tape Input

>EN 110

0110: CD 0F E0 32 09 01 C9/ -->> this gets 1 byte off tape.
                                 and places it at 09H

>GO 0                    ---->> return to CP/M

A>SAVE 28 PIP1.COM       ---->> You MUST save it on disk.
A>PIP1
*
```

Now PIP has a patch in it to allow input from an external tape device. We can buffer the program on the tape (which is a text file) into memory and finally onto disk. When PIP finds an EOF (CTRL Z) it writes the remainder of the buffer to disk and closes the file.

This may seem a difficult method of conversion but it did save me much time writing a routine to convert the tokenized program to a text file and saving it on disk. This way the basic , the monitor and PIP do all the work.

To start the loading, first rewind the tape to the start of the save. Now type the following line and when the drive head unloads, press the play button on the recorder. The process is automatic after this and you take control when the '*' re-appears.

```
*PROGRAM.ASC=INP:[B]      - loads program from tape to disk
*^C                       - returns to CP/M
A>
```

Now load Wordstar or a similar editor. Give it the filename of the program just loaded from tape. It is your job to clean up the file. Take out the 'LIST', 'READY' and any other trash picked up at the start or end of the program.

The Basic Compiler will only work with files that have spaces after the basic command keyword. Most Pac-Basic programs are compacted and need spaces added. A simple method to correct this is to use the 'Find and Replace' command of Wordstar or equivalent.

```
ex- FORI=1TO200STEP2:NEXTI ==>> FOR I=1 TO 200 STEP 2:NEXT I

 ^QA FIND? TO    REPLACE WITH? _TO_   OPTIONS? NG
     FIND? FOR   REPLACE WITH? FOR_   OPTIONS? NG
     FIND? STEP  REPLACE WITH? _STEP_ OPTIONS? NG
     FIND? NEXT  REPLACE WITH? NEXT_  OPTIONS? NG
```

Do this for all the keywords associated with your program. It won't take more than 15-20 minutes. There is one drawback - things other than keywords can be changed ex.- STOP can be changed to STO P with the 'TO' change. A better idea would be to use only the 'G' option which changes all occurences in the file but asks the user if the specific keyword is to be changed to not. This would make conversion time much longer.

While we have the file in an editor, a few inconsistencies between the SYNTAX of both basics must be cleared up. Make the following changes to the Rom-Pac file if they show up in the program (I can't guarantee this will solve all your problems - check the MBASIC reference manual for further info).

- Machine language subroutines in zero page will have to be relocated to high memory or just located after the end of the .COM file. This address can be found after the program has been linked with L80 - just re-edit the file to poke the routine a few hundred bytes after this address.

- the CLEAR statement will have to be removed as it is not implemented in BASCOM.

- include 'WIDTH 64' at the start of the program (not always needed).

- the USR argument and the pokes to 260 and 261 must be changed to ---)

```
   10 DEF USR0 = ADDRESS OF ROUTINE IN AN INTEGER VALUE
   20 A=USR0(X)
```

- TAB(X) does not work in the same manner as Pac Basic - it spaces over 'X' characters irrelevant of the present cursor position.

- any machine routines used to get dynamic input from the keyboard can be changed to 'A$=INKEY$'

Exit the editor when all changes can be made and load MBASIC 5.2 Then load the newly converted file with 'LOAD "PROGRAM.ASC"'. If it doesn't load properly then re-edit it until no errors occur. Now run it - edit the file in basic until it works properly then 'SAVE "PROGRAM.ASC",A' and exit MBASIC with 'SYSTEM'.

Now we will start to compile the basic file:

```
A>BASCOM =PROGRAM.ASC/Z
A>L80 PROGRAM,PROGRAM/N/E
A>PROGRAM       ---))  Try  out .COM version to see  it  will
work.
```

Probably the program won't work the first time and may need many changes to the source file. The overall effect is great and worth all the time put into a conversion. The BASIC Compiler had been very in-active until I found how to do the conversion. If I only had more time to convert those Basic files!

##

ONLY IN CANADA YOU SAY

It seems that if anything to do with the Sorcerer is going to be corrected it is done by the ingenious owners of Sorcerers in Canada. Specifically speaking of the **CP/M 2.2** supplied for the Sorcerer running Micropolis Mod II disk drives. The product supplied by a Mentzer Electronics has a BIOS in it written by some poor under educated soul who does'nt even know that the Sorcerer has a built in Centronics Driver and is of the firm opinion that everbody will run only software he prescribes has a serious deficiency. With any luck he will eventually learn that a programer does not force his views on a user BUT writes the software to allow the user to make the best use of his tools.

Listed is a long awaited fix to this individuals BIOS that allows CP/M to perform the way it was designed to operate. Some additional Bells and Whistles (fixes) has increased the Sorcerer's ability considerably.

```
 ;   NOTE:  Please make sure that the BIOS that you are working with has the  same
 ;          beginning. The rest will by all probability be the same.
 ;           The  JUMP table also has some revisions so read and make the changes
            very carefully.
```

The following changes were made by Walter Blady and Frank Aylesworth

CHANGES TO THE PROGRAM "MBIOS32E.ASM"

```
;         MBIOS - DRIVERS FOR CP/M V2.2 AND MICROPOLIS DISC
;         COPYRIGHT 1979, BRUCE R RATOFF, ISELIN, NJ
;         LAST REV 08/06/80

FALSE   EQU     0                   ;BASIC DEFINITIONS
TRUE    EQU     NOT FALSE

SYSSIZE EQU     48                  ;SET TO YOUR SYSTEM'S MEMORY SIZE IN K
RELOC   EQU     FALSE

MODII   EQU      TRUE               ;SET TRUE FOR 4 77-TRACK MOD II DRIVES

VERS    EQU     22
MSIZE   EQU     SYSSIZE             ;NOMINAL MEMORY SIZE
CBASE   EQU     MSIZE*1024-2000H;CBASE DERIVED FROM NOMINAL SIZE
PATCH   EQU     CBASE+1600H

CPMB    EQU     CBASE                       ;START OF CP/M
BDOS    EQU     CBASE+806H                  ;START OF BDOS
LOGDSK  EQU     4                           ;WHERE CURRENT DISK IS KEPT
CPBUFF  EQU     80H                         ;DEFAULT DISK BUFFER
ERRMAX  EQU     5                           ;MAX RETRIES ON READ ERROR
IOBYTE  EQU     3                           ;CP/M IOBYTE ADDRESS
CLEAN   EQU     0CH                         ;CLEAR SCREEN CHAR.

;'INTIOB' SPECIFIES THE DEFAULT I/O ASSIGNMENT AT COLD BOOT.
;'IOBYTE' IS DIVIDED INTO 6 FIELDS, OF 2 BYTES EACH.
;THE ASSIGNMENTS ARE...
;(bits 0,1)
;CONSOLE - 0=TTY DEVICE (TTY:)
;        - 1=CRT (CRT:)
;        - 2=BATCH:READER AS INPUT AND LIST DEV.AS OUTPUT (BAT:)
;        - 3=USER DEFINED CONSOLE (UC1:)
;
;(bits 2,3)
;READER  - 0=TTY (TTY:)
;        - 1=HIGH SPEED READER (PTR:)
;        - 2=USER DEFINED READER 1 (UR1:)
;        - 3=USER DEFINED READER 2 (UR2:)
;
;(bits 4,5)
;PUNCH   - 0=TTY (TTY:)
;        - 1=HIGH SPEED PUNCH (PTP.)
;        - 2=USER DEFINED PUNCH 1 (UP1:)
;        - 3=USER DEFINED PUNCH 2 (UP2:)
;
;(bits 6,7)
;LIST    - 0=LIST IS TTY (TTY:)
;        - 1=LIST IS CRT (CRT:)
;        - 2=LIST IS LINE PRINTER (LPT:)
;        - 3=LIST IS USER DEFINED LIST DEVICE (UL1:)

INTIOB  EQU     0       ;INITIAL IOBYTE VALUE - (PRESENTLY SET
                                                ;FOR TTY:)
MODIIMAX EQU    (77-2)*32/16-1              ;NUMBER OF BLOCKS PER MOD II DRIVE

NSECTS  EQU     (PATCH-CPMB)/256            ;NUMBER OF SECTORS IN CCP+BDOS

DSTEP   EQU     60H     ;STEP IN COMMAND
DWEN    EQU     80H     ;WRITE ENABLE COMMAND
SLTD    EQU     4       ;SELECTED STATUS FLAG
```

```
DSEL     EQU      20H      ;SELECT COMMAND
DREADY   EQU      20H      ;READY STATUS FLAG
TK0      EQU      8        ;TRACK 0 STATUS FLAG
WPT      EQU      10H      ;WRITE PROTECT STATUS FLAG

;
         JMP      BOOT              ;COLD BOOT ENTRY
WBOOTE:  JMP      WBOT              ;WARM BOOT ENTRY
         JMP      CONST             ;CONSOLE STATUS
CI:      JMP      CONIN             ;CONSOLE INPUT
CO:      JMP      CONOUT            ;CONSOLE OUTPUT
         JMP      LIST              ;WRITE TO LIST DEVICE
         JMP      PUNCH             ;WRITE TO PUNCH DEVICE
         JMP      READER            ;READ FROM READER DEVICE
         JMP      CPHOME            ;PERFORM LOGICAL HOME
         JMP      CPSLCT            ;PERFORM LOGICAL SELECT,SET DENSITY,RETURN DPB
         JMP      CPSTRK            ;PERFORM LOGICAL SEEK
         JMP      CPSSEC            ;SET LOGICAL SECTOR #
         JMP      CPSDMA            ;SET DISK I/O ADDRESS
         JMP      CPREAD            ;READ A CP/M PSEUDO-SECTOR
         JMP      CPWRIT            ;WRITE A CP/M PSEUDO-SECTOR
         JMP      LISTST            ;TEST LIST OUTPUT READY
         JMP      SECTRAN           ;TRANSLATE CP/M SECTOR #

; ======================================================================
;    ALL DATA THAT IS INCLUDED BETWEEN THESE SECTIONS REMAINS THE SAME
; ======================================================================
-------------------- EXIDY CONSOLE SECTION -----------------

THE FOLLOWING ROUTINES USE THE IOBYTE TO DECODE THE
ADDRESS TO JUMP TO FOR THE PROPER FUNCTION.  THE ADDRESS
TABLES POINT TO THE PROPER ROUTINES.  IF YOU ADD OR MOVE
ROUTINES AROUND, THE CHANGES MUST BE REFLECTED IN THESE TABLES
\


EXDYIN   EQU      0E009H
EXDYOT   EQU      0E00CH
EXPARL   EQU      0E997H
COLPT    EQU      EXPARL
OUTAPE   EQU      0E012H
INTAPE   EQU      0E00FH

;EXIDY CONSOLE STATUS

CSTTY:   PUSH     BC
         LD       B,0FH
ON1:     LD       A,B
         OUT      (0FEH),A
         IN       A,(0FEH)
         AND      1FH
         CP       1FH
         JR       NZ,ON2
         DJNZ     ON1
ON2:     POP      BC
         LD       A,00
         RET      Z
         DEC      A
         RET
```

```
PNCH1.  RRA
        RRA
        JR      READR1

;ADDRESS FOR LIST DEVICE

LIST:   LD      HL,LTBLE
LIST1:  LD      A,(IOBYTE)
        RRA
        RRA
        JR      PNCH1

;ADDRESS FOR LIST DEVICE STATUS

LISTST: LD      HL,LSTBLE
        JR      LIST1

;THE FOLLOWING TABLES HOLD THE ADDRESSES OF THE VARIOUS
;DEVICE ROUTINES AND THEIR RELATED STATUS ROUTINES
;CONSOLE INPUT TABLE-

CITBLE: DW      CITTY
        DW      CICRT
        DW      READER  ;DEPENDS ON READER ROUTINE
        DW      CIUC1

;CONSOLE OUTPUT TABLE-

COTBLE: DW      COTTY
        DW      COCRT
        DW      LOUT
        DW      COUC1

;LIST DEVICE TABLE-

LTBLE:  DW      COTTY
        DW      COCRT
        DW      LOUT    ;EXIDY PARAL.OUTPUT
        DW      COUL1

;PUNCH DEVICE TABLE-

PTBLE:  DW      COTTY
        DW      COPTR
        DW      COUP1
        DW      COUP2

;READER DEVICE TABLE-

RTBLE:  DW      CITTY
        DW      CIPTR
        DW      CIUR1
        DW      CIUR2

;CONSOLE STATUS TABLE-

CSTBLE: DW      CSTTY
        DW      CSCRT
        DW      CSREAD
        DW      CSUC1
```

```
;READER DEVICE STATUS TABLE-

CSRTBL: DW      CSTTY
        DW      CSPTR
        DW      CSUR1
        DW      CSUR2

;LIST DEVICE STATUS TABLE-

LSTBLE: DW      READY   ;CONSOLE ALWAYS READY
        DW      READY   ;LIST DEV.ALWAYS READY
        DW      READY   ;DUMMY
        DW      READY   ;DUMMY

READY:  XOR     A
        JR      READY1

,DUMMY OUPUT ROUTINES-

COCRT   EQU     $       ;OUTPUT TO CRT
COUC1   EQU     $       ;OUTPUT TO USER CONSOLE 1
COPTR   EQU     $       ;OUTPUT TO PAPER TAPE PUNCH
COUP1   EQU     $       ;OUTPUT TO USER PUNCH 1
COUP2   EQU     $       ;OUTPUT TO USER PUNCH 2
COUL1   EQU     $       ;OUTPUT TO USER LIST DEV.1

        JP      COTTY

;DUMMY INPUT ROUTINES-

CICRT   EQU     $       ;INPUT FROM CRT
CIUC1   EQU     $       ;INPUT FROM USER CONSOLE 1

        JP      CITTY

CIUR1   EQU     $       ;INPUT FROM USER READER 1
CIUR2   EQU     $       ;INPUT FROM USER READER 2
CIPTR   EQU     $       ;INPUT FROM PAPER TAPE READER

        LD      A,1AH
        RET

;DUMMY STATUS ROUTINES-DUMMY, ALWAYS RETURN 'READY'

CSUR1   EQU     $       ;STATUS OF USER READER 1
CSUR2   EQU     $       ;STATUS OF USER READER 2
CSPTR   EQU     $       ;STATUS FOR PAPER TAPE READER
CSUC1   EQU     $       ;STATUS OF USER CONSOLE 1
CSCRT   EQU     $       ;STATUS FROM CRT

        XOR     A
        JR      READY1

STAT:   LD      A,0
        RET     Z
READY1: DEC     A
        RET

;CONSOLE INITIALIZE ROUTINE-CLEARS SCREEN AND SETS THE IOBYTE
;FOR DEFAULT CONSOLE-SEE INTIOB AT START OF SOURCE CODE
```

```
IOINIT: LD      C,CLEAN
        LD      A,INTIOB
        LD      (IOBYTE),A
        JP      CO

FIRST$FREE      EQU     $

; THESE AREAS ARE USED FOR WORKING STORAGE BY THE BIOS AND BDOS.
, NOTE THAT 'CONADR' IS INITIALIZED BY THE COLD BOOT AND MUST BE
; THE FIRST TWO BYTES OF THE WORKING STORAGE AREA.

BEGDAT  EQU     $

CONADR: DS      2       ;POINTER TO MICROPOLIS CONTROLLER
TRKTBL: DS      4       ;CURRENT POSITION OF EACH DRIVE
DTRACK: DS      1       ;DESIRED TRACK
DSECT:  DS      1       ;DESIRED SECTOR
UNIT:   DS      1       ;DESIRED UNIT
DMAAD:  DS      2       ;DATA TRANSFER ADDRESS
ERRCNT: DS      1       ;NUMBER OF ERRORS ON CURRENT OPERATION
SECTMP: DS      1       ;TEMP USED BY WARM BOOT
CCHAR:  DS      1       ;LAST CONSOLE INPUT CHARACTER
LCHAR:  DS      1       ;LAST CONSOLE OUTPUT CHARACTER

RDBUFF: DS      268     ;DE-BLOCKING BUFFER FOR READS
WRBUFF: DS      268     ;EN-BLOCKING BUFFER FOR WRITES

DIRBUF: DS      128     ;BDOS'S BUFFER FOR DIRECTORY ACCESSES
CSV0:   DS      32      ;DIRECTORY CHECKSUM VECTOR
ALV0:   DS      19      ;ALLOCATION VECTOR
CSV1:   DS      32      ;DIRECTORY CHECKSUM VECTOR
ALV1:   DS      19      ;ALLOCATION VECTOR
CSV2:   DS      32      ;DIRECTORY CHECKSUM VECTOR
ALV2:   DS      19      ;ALLOCATION VECTOR
CSV3:   DS      32      ;DIRECTORY CHECKSUM VECTOR
ALV3:   DS      19      ;ALLOCATION VECTOR

ENDDAT  EQU     $
DATSIZ  EQU     ENDDAT-BEGDAT
        END
```

Use M80 and L80 to compile and link the new BIOS; and then create a 48K system image. If you have SID great if not follow the following steps to put your new system together·

1. Compile the new BIOS using M80,
2. Link the REL file with L80 to produce a COM file,
3. DDT or SID(if possible) the new system image(48K),
4. When DDT or SID prompt appears type

```
        -Inewbios.com(cr)
        -R1F00
        -^C
A>SAVE 40 CPMXX.COM
```

You are now ready to sysgen your new system.
Now the new image will work like regular CP/M so on with the whistles;
If you are using SID then type what appears on the right if not then use DDT and type whats on the left. The addresses are as they will appear in either SID or DDT.

```
 UNDER DDT                              UNDER SID(A COMMAND)
ADDRESS OF CODE

1402    FE7F                            CP 7F
1404    2011                            JR NZ,1417
1406    78                              LD A,B
1407    B7                              OR A
1408    28E5                            JR NZ,13EF
140A    3E08                            LD A,08
140C    05                              DEC B
140D    2B                              DEC HL
140E    C5                              PUSH BC
140F    E5                              PUSH HL
1410    4F                              LD C,A
1411    CD54BA                          CALL 0BA54 (0d654 in 55K)
1414    C3AFAA                          JP 0AAAF   (0C6AF IN 55K)
1417    FE08                            CP 08
1419    200B                            JR NZ,1426
141B    78                              LD A,B
141C    B7                              OR A
141D    CAEFA9                          JP Z,0A9EF (0C5EF in 55K)
1420    7E                              LD A,(HL)
1421    05                              DEC B
1422    2B                              DEC HL
1423    C3A9C6                          JP 0AAA9    (0C6A9 in 55K)
```

With this routine SHIFT RUB does its thing
CRTL H now echo's the deleted character.

Change address 1333 from a 13 to 1B this changes CRTL S to ESC

Change address 13F4 from 1F to FF to allow CP/M to use EXIDY Graphics.

The whole BIOS is listed but I must point out that the major changes occur from the Exidy Console section routine so typing is not as much.

It is hoped that you will find this routine will make your EXIDY perform more like a computer as well you will be able to use Software packages that employ CRTL characters ie:MODEM7, Wordstar etc.

Frank Aylesworth

The Prez Zez

As many of you may be aware these changes to the CP/M 2.2 versions that have been offered for sale have not been running to the best ability that would be expected. Hopefully these changes outlined will be in the best interest for all of those people with CP/M 2.2.
We did not have enough room to print the entire BIOS and if any of you are encountering any difficulty implementing these changes we will can send to you an entire BIOS listing for a nominal handling charge of $6.00 ,this is to cover copying and postage.
We have not had the room for our usual technical tips ie; PIO & SIO related material. This shall be continued in the following issues of PORT FE.

Considerable ground has been covered since the last topic of the PIO interface to the Sorcerer. The basics of SIO will enable a Sorcerer owner to hook up a serial terminal with data exchange (so far tested) 19,200 baud. The circuit diagrams will be published in the forthcoming issues also. Since this can get rather complicated it will be taken in stages.

The Prez

SORCERER USERS' GROUP (TORONTO)

<u>Membership</u> <u>Application</u> <u>Form</u> Covering Jan. to Dec.1982

Membership to the group is <u>not</u> 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 spokes-person 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 excecu-tive 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 <u>Cheque</u> or <u>Money</u> Orders.

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

You may send $6.00 + $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.
....