PORT FE

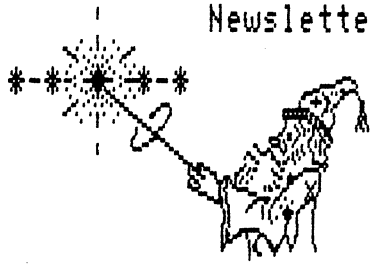
SORCERERS USERS' GROUP

(Toronto)

P.O. Box 1173 Sta. 'B'
Downsview, Ontario,
Canada. M3H 5Y6

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.

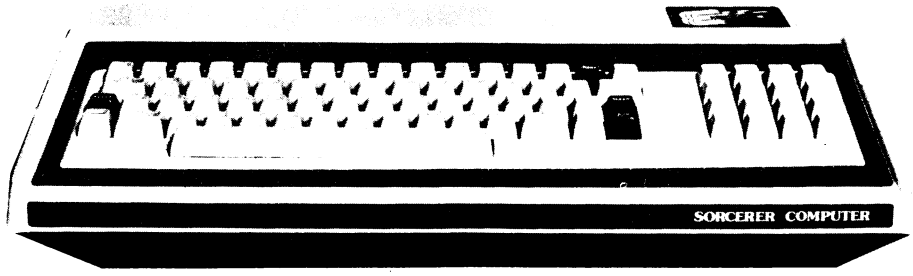
December 1982 ISSUE

TABLE OF CONTENTS

GENERAL INTEREST

MEETING PLACE

somewhere at somebody's house sometime



FINAL ISSUE

FINAL ISSUE

FINAL ISSUE

FINAL ISSUE

FINAL ISSUE

FINAL ISSUE

FINAL ISSUE

FINAL ISSUE

FINAL ISSUE

FINAL ISSUE

FINAL ISSUE

FINAL ISSUE

FINAL ISSUE

FINAL ISSUE

The Prez Zez;

Well it had to come some time..... For those of you who never attended one of our meetings, I shall try and put some of our logic and reasoning together for you.

At our Dec. 15/82 meeting, with the number of members present totaling (7), it was to have been our elections. There were NO nominations and no one wanted the responsibility of the excecutive positions. One of the most evaded positions was that of PORT FE editor. Without the continuation of PORT FE and the local support, we were doomed.

Many of our local members within the last two years have sold thier Sorcerers and gone to better (hopefully) other computers. Some have made quantum jumps to 16 bit processors, and others have moved not up but rather just sideways to machines that had more in the way of support software. Lets not forget also that many of the Sorcerer owners when trouble arrived could not find service and this I feel was a major contributing factor to the way people felt.

We had talked about the eventual dwindling of members for some time and I vowed that we would not just drift and dwindle to nothing. It was evident for some time also that articles from members were becomming very infrequent and that the other clubs were experiencing the same sort of thing. Well for our own group this in a way didn't matter, because most were interested in more and more exposure to CP/M related programming. Same I believe holds true for most of us that have disk systems. There are a relatively small group that do not have disks by this time and they probably never will at the rate of program development for the Sorcerer is concerned. Many have changed because of the fear of service and also because they wanted to stay in the main stream of computer development. When one sees other clubs emerging and growing very swiftly, all within one year, one begins to wonder whether they are right and we are doing something wrong.

Most felt that it would be a shame to dissolve the group for fear again of future support. Well we had agreed that we would still continue to meet at a less frequent times and with the number turning out for the meetings it was just not justified for the rental of the premises, and we could do with meeting at a home of any one of the people that still remained interested.

I pointed out that in a way the group had fulfilled its function, from the time that it started to this very day and that we shouldn't feel too much remorse in its breaking up. We can still be of service to those that remain in other lands by contributing any articles that we feel should be brought to the attention of others by sending them in to some of the other newsletters. I would probably continue to keep most of the local people that I personally know informed as to what's going on, so that if anyone wanted to get help it would still be available. The objective was to be a unified body with a common interest. The common interest was still there but not the volunteers.

Volunteers is what we needed the most, because it was almost always the same people that did the work, maybe shuffle the jobs around a little, but never new faces to take on the responsiblities. Many probably don't care what happens but let this be a reminder for those that belong to other clubs, they need not only your membership funds to keep things going but more important they need your personal and physical help. If you want to have a club then put your hand up, the more people that put their hands up the easier the work becomes and also the feeling of support and ties become even stronger. Everyone has at one time or another cursed because the support of the manufacturer was a poor show, well the same holds true for any organization. We have had a hard road to travel for the past four years. Reflect back to the time the Sorcerer was first introduced and you compare that with say the Apple, no contest, well this shows you that it really doesn't matter what type of computer it is but how fast support is developed for it, that really determines whether or not it will be around and for how long. In our times of swift changes, we should all be prepared for what is to come. The micros that you see here today, gone tommorrow !!! Thats the way with fast changing technology.

Many will try to hold onto things because of one reason or another. For others it is a monetary consideration, and too expensive to change. Let me give some advice to most of you that are considering changing for some other type of computer. I have been an amateur radio operator for some years and there is a favourite saying "keep the stock rotating", this virtually means change your equipment on a regular basis, and it holds true for any type of electronic gear or whatever you have. Unless that piece of gear can't be replaced or there is nothing better available. We have found that you don't lose as much in the long run if say you traded your 1982 model for the 1983 model. But if you held onto it for too long of a time frame the technological advancements would make your equipment value decrease on a logrithmic scale (slowly at first and more and more every month thereafter).

This some of you will argue doesn't apply to computers. In a way it should be treated a little different, for example, you just don't learn everything that you want within let's say one year, and you're right, but don't forget about the world around you because it won't stand still while you take your time learning, and subseqently you have to face up to that fact that you can't keep pace with it all.

For those of you who have the Sorcerer as the first micro computer ever, it seems unfair that you should suffer the consequence of the let's say (" RAPID TRANSIT") effect that is occuring with the computer industry as a whole. For many get very disheartened to hear of the 'better' and more sophisticated micros that are emerging every day. What to do ? you ask yourself ? Can a painter predict what his collegues will do next ? NO ! and neither can anyone predict what the fast electronic world will come up with next and when? So now it comes down to the facts, let us consider the situation and some of the possible alternatives that are left open for us.

1. Advantages
2. Disadvantages
3. Courses left open for us
4. Why you shouldn't change
5. Why you should change
6. Life cycle of the Machine
7. Epilogue

1. Advantages that you have could be grouped into various fields.
One of the most significant advantages is that you have some software that is of personal use to you. The variety of programming in many ways cannot be matched by some of the other micros. If you have children, then this could have been one of the reasons that you got it in the first place. If you have disks and access to the CP/M environment, you may in many ways not care what happens. You have a limited but very useful range of graphic capability as can be seen from the varied games available. Nearly all the languages are now available, some of which you can run and some of which your system lacks the memory to run. But for the most part there is more that you CAN run than what you can't run. If you have the S100 expansion unit then this I would consider an advantage, because many S100 cards will fit and support is out there for those wishing to experiment.

2. Some of you may be running into problems related to languages that cannot be run for whatever the reason. Memory has always been a problem, some programs require a minimum system configuration of 56K and for some that is a problem. If you don't have an S100 expansion unit then I would say it's a point against you. Expansion of any system is a must. The cost incurred in expanding your persent system to include such features as colour and graphics may be of prime concern to some of you and without the technical knowledge to do it yourself, support is dwindling. Scanned keyboards are a deterrent to the speed of the computer and if the video is also tied in then the rate of processing information decreases.

3. Courses that are left open are as follows and for the most part probably evident to many of you already. Keep the machine for whatever purpose you are using it for now, or sell it. I firmly believe that many of us have found very useful programs that have helped us in a variety of different ways. If the usefulness of the computer has helped perform some task that was either too complex or to mundane a task, then it's of some help in the future to you. You could also look at the hardware end of things and try to bring it up to date, but I would caution you on that aspect 'it will be cheaper in the long run to buy another', some things can be updated and other things are better off not to even be attempted.

4. Why you shouldn't change will be very evident to those of you that have the Sorcerer doing something of value. If it is performing business functions at a speed acceptable right now to you then don't consider it. If you are still learning the various languages that are of interest then it still performs a valuable function, or if you wish to have your children learn, it is valuable item for them. Consider the SOFTWARE that may NOT be transportable to the new computer.
Evaluate the situation very carefully. The losses may end up greater than you realize or that are evident at the moment.

5. Why you should change will be very obvious to those people that have a function that cannot be done with the current computer. This will seem to emerge in various degrees and at various times. The best way to handle the situation will be to determine whether or not this will become more and more prevalent in the future and then make your decisions based on the facts at hand. For some this will be a very obvious thing, for others it will need a lot of soul searching, if not just the pocket book. Many will try to hold out for as long as possible, this may be a mistake, depending on how you look at his/her situation.

6. The Life Cycle of the computer is not hard to determine, once one begins to understand the nature of the animal that you are dealing with. To some the life cycle may merely be that it 'stopped working'. This is not a valid life cycle but a malfunction that could be corrected. The matter of whether or not this terminates the life cycle is 'will it cost just as much to fix it as a new machine will cost - or close to it. Take for example the Apple II, there are now Apple look alikes on the market. An Apple normally sells for around the $1700 range here in Canada. Should a person spend $1000 just to replace the main board or should he spend $750 for a complete new Apple look alike? What would you do? The answer is obvious. The same type of thought holds true for any type of computer. There are now Sorcerers (probably with problems) selling for as low as $50.00 yes Fifty dollars. Now ask yourself whether or not this does something for the image of the computer or not. For $50 you can get spare parts at the very least. Many of our members have more than one computer, why ? because they are trying to keep up with todays world or just the advancement of our age.

7. Epilogue - Because of todays swiftly changing technology there are many things that are going to be forced on you, obsolescence is just one of those things. In the world of computers you will find that the computer will continue to undergo many changes. Be prepared to move as swiftly as the industry moves, keep up to-date on whats going on and try to keep one step ahead of "losing your shirt" on your investment - "the computer".

==========

A Personal Note

I would like to thank all those people that have helped 'keep us informed' and have spent long hours working with me on PORT FE. Many have sent in articles on a regular basis - special thanks to those people for they deserve it.

For the friendships that I have had the oportunity to experience
I'm sure will remain mutual for a long time to come.

==========

RETURN OF 1983 MEMBERSHIP FUNDS

Your 1983 membership payments will be returned as soon as possible. If you have sent them in, for many of you will have found them included with this issue and anyone that has any questions, or not received their membership funds back please write and let me know. This however does NOT apply to the 1982 membership listings.

REFLECTIONS & ACHIEVEMENTS

Many wonder why things must come to an end. Just remember that all good things must end sometime. I think back to the beginning when we got our Sorcerers and remember some of the good things that have come out of our group.

a) Graphic games
b) Restore xxxx (xxxx is a line number)
c) Clear xxxx,aabb (aabb is top of Basic Ram)
d) All the articles on Tape problems.
e) A true modem program SMODEMX
f) TRS801 TRS 80 level II basic C/W disk commands.
g) Pacbasic relocated and c/w disk commands.
h) SIO Serial interface for the Sorcerer.
i) PIO Parallel interface for the Sorcerer.
j) CP/M Speed up BIOS modifications.
k) EXMON2 new monitor for the Sorcerer.

This is just to name a few of the great things that have highlighted the existence of our group. We have had the fortune to have many talented people in our small group. Yes small group. We started out to have about 20 - 30 local members and we dwindled down to something like 7 - 12 attending the meetings every month.

I would hope that everyone has enjoyed themselves in the past. Many times Sorcerers were brought to the meetings, but I still remember the time we had FOUR systems up and running at a meeting this year. That was the most number we ever had.

We all had fun watching the games being demonstrated and finding out who had some new software that had never been run on the Sorcerer before. I guess the fellows with the disk systems had about the most fun of all. Problems and all, yes many had BIOS problems and one by one these were cleared up. Then came the memory expansion period and everybody was in the act. I know a few that have other computers and one who has interfaced his Sorcerer with an Elf and a 6809 and an Amsi and has also interfaced it with a TMS 9918 video chip and will probably within the next few months also hook up a 68000 to it as well. Now you tell me whether or not that's called expansion and keeping up with the times. Yes there are some who are never satisfied. Just remember that all this takes time. If you have the time to devote then you can stay ahead of the game. Now programming is another thing, it just eats the time away and sometimes you wonder were it went and also what you have to show for it.

If you're happy with what you've got that's all that counts.

I wish happiness for all those members in far away places that I've not had the chance to meet in person.

Have a Happy New Year and I wish you all prosperity.

==

Update to 'Tutorial on Standard Rom-Pac Basic - Part I'

Please make the following changes on page 2 of the June/July 1982 issue of Port-FE:

0117-0139 Pseudo-random number data
01C3 Internal storage used for floating point prinout and multiplication.

I must thank William A. Pointer of Cambridge, ONT. for these updates which now makes the article complete.

<u>Tutorial on Standard Rom-Pac Basic - Part II</u>

In the June/July issue of Port-FE I explained the make-up of the Rom-Pac in a general sense and am now going to apply these subroutines into basic command extensions.

With most basic interpreters, there is a vector point from which you may insert a jump to an 'extension interpreter'. This would allow you to add new commands to the basic's vocabulary. Such an example is DISK BASIC for the TRS-80 which is just an added section of code to the Level II cassette basic. When Microsoft configured ROMPAC basic, they were very kind in not including one of these vectors!

After fruitless searching inside the interpreter I found a pseudo vector point (?!?) - it deals with the INP/OUT commands. Whenever basic executes an 'INP' or 'OUT' command, it places the port number in the Basic work area and jumps to the appropriate routines at 0106H or 013EH (outside the ROMPAC). The following is the actual code for the two I/O commands to give you a better understanding about what I'm going to explain.

INPUT: (A=INP(255))

```
                ORG  0D24AH     ; code found in Rom-Pac
D24A: CD 90 D2  CALL GETVALUE   ; get output port number
      32 3F 01  LD   (013F),A   ; store it in BWA input routine
      CD 3E 01  CALL 013EH      ; call input port routine
      C3 1A CF  JP   STOREINPUT ; store A in basic variable

                   ORG  013EH   ; found in Basic work area
** 013E: DB (00)    IN   A,(00) ; get value from port  **
** 0140: C9         RET         ; and return to ROMPAC **
```

OUTPUT: (OUT 255,0)

```
D256: CD 8D D2  CALL GETVALUE   ; get port number
      32 07 01  LD   (0107H),A  ; store in MWA input routine
      CD 7A C5  CALL SYNTAX     ; check for comma
      2C        DEFB ","        ; comma to be compared to (HL)
      CD 8D D2  CALL GETVALUE   ; get value to be output
      C3 06 01  JP   0106H      ; jump to output routine in BWA

** 0106:  D3 (00)   OUT 00,A       ; send value to port    **
** 0108:  C9        RET            ; return to OUT routine **
```

---) The (00) value in the Basic Work Area is modified by the BASIC INP/OUT commands before the routine is executed.

This is just what we want!. Now we can jump out of basic into ram, intercept the INP or OUT commands and jump to our own commands. To do this, place a JUMP (C3) at 0106H and 013EH in the I/O routines. Then we set 0108H and/or 0140H to the most significant byte of our new command's memory address. What about 0107H and 013FH? Well, remember, whenever INP is called, the input port number is placed at 013FH. And the output port for the OUT command is placed at 0107H. So all we have to do is place the least significant byte of the jump in the variable argument of INP(XX) or OUT XX,##. Basic will now put this value at 107H or 13FH, jump to one of the I/O routines but instead of doing I/O , it will jump to our new routine. Let me show you in an actual routine.

```
0106 or 013E: C3        JP instruction
0107 or 013F: (00)      LSB of jump which is placed here
                        by OUT routine
0108 or 0140: ##        MSB of jump which is poked here
                        by user.
```

The following program demonstrates the use of the OUT command. It will send the value in LETTER to the screen through the monitor's OUTPUT vector at E01B.

```
10 POKE 262,195       ; set first 0106 to a jump(0C3H)
20  POKE 264,224      ; set MSB of jump to a jump into monitor
30   A=65             ; send out ASCII character  set through
40   OUT 27,A         ; monitor OUTPUT vector at E01BH
50  A=A+1: IFA=78 THEN END  ; and do rest
60 GOTO 40
```

The following program is a dynamic INKEY$ routine (get keypress from keyboard). Most people usually had to set up a USR argument to achieve this, but now can use one INP command. The argument on the left side of the equal sign receives the character pressed.

```
10 POKE 318,195      ; set 013E to a jump
20  POKE 320,224     ; and set MSB of jump to monitor jump table
30   CHAR=INP(24)      ; get value from keyboard
40  PRINTCHR$(CHAR);   ; and print char. received
50 GOTO 30             ; continue
```

Another plus is that we can call any address in the current page addressed without rePOKEing the MSB of the jump address, unlike the USR command which can only call one address at a time. Example- if we make the MSB point to the monitor jump table (E0XXH), then we can call any monitor routine via the table -) A=INP(24) gets a keypress from keyboard (E018), A=INP(00) does a monitor cold start(E000) and A=INP(3) does a warm start (E003).

Passing values back and forth between basic and machine language is more complicated with the USR. But with the INP/OUT mode, the A register either contains the value from the OUT command (OUT 255,10 - jumps to 255H with A contains 0AH) or the INP command (CALL 0E018H /RET/ Value in A passed onto INP command and stores in argument on left side of equal sign - VALUE=INP(255)).

Register and stack values when INP/OUT invoked:

Lets get to know more about the unseen aspects of the I/O commands. When the OUT command has jumped to our routine the registers and stack look like this:

```
A      = value to send to output port - 'A' value of OUT port,X
BC,DE  = no special contents.
HL     = Basic text pointer (should point to last byte of :
         OUT PORT,ARGUMENT)

STACK+0  = Return to basic command interpreter
STACK+2  = 0000 - signifies end of basic stack
```

And the following are the stack and register values of the INP command at the point when it has jumped to our command.

```
A = value that WILL be sent back to basic with the value
    of the input port number located at 013FH. Any value
    can be returned with our routines, and will be
    placed in the floating point variable on the left
    side of the equal sign.

BC,DE,HL = no special contents.

STACK+0 = Return address to INP routine in ROMPAC
STACK+2 = Return address FUNCTION interpreter
STACK+4 = basic text pointer - points to last bracket in
          'A=INP(0)'
```

We now have a way of adding commands onto basic. I have already demonstated an easier way to add a INKEY$ command and use the A register value passing method. But what if we were to pass 16 bit arguments and not 8 bit? No problem... This is where I will use the routines I described in the June/July issue of PORT FE. Lets write a PRINT@ command. This is a TRS80 function which allows you to move the cursor to any location on the screen (0-1919). The format will be: 'OUT 0,0,XX'

The OUT is the command to jump to 0106H where our new command jump is. The first '0' tells to jump to 0000H (264 must be 0 as well). The second zero is a dummy variable - it is not used here. The XX is the 16 bit screen location.

```
10 POKE 264,0:POKE 262,195:      ; Set up OUT command to jump
                                   to 0000
20 DEF FNA(X)=INT(X*RND(2)+1)
30 OUT 0,0,FNA(1919)             ; move cursor to random spot
40 PRINTCHR$(32+FNA(30));
50 GOTO 30
```

-) after entering the above program, get into the monitor and enter this machine language program. Return to basic and type RUN.

```
0000: E5            PUSH HL          ; save BASIC text pointer
      CD A2 E1      CALL FINDMWA     ; point IY to MWA
      CD E8 E9      CALL REPCURS     ; take cursor off screen
      E1            POP  HL          ; get BASIC pointer
      23            INC  HL          ; inc. over ',' in line
      CD 7F CB      CALL TESTNUM     ; test for numeric
                                     ; argument - TM ERROR if not
      CD D0 C7      CALL GET16BIT    ; get 16 bit argument
                                     ; from BASIC text pointed to
                                     ; by HL, put value in DE and
                                     ; return
      7A            LD   A,D         ; get MSB of XX argument
      E6 07         AND  07H         ; limit argument to 07FFH
      57            LD   D,A         ; save it
      7B            LD    A,E        ; and get LSB of XX argument
      E6 C0         AND  0C0H        ; limit value of XX to 1919D
      FD 77 68      LD   (IY+068H),A ; store away cursor offset
      FD 72 69      LD   (IY+069H),D
      7B            LD   A,E         ; get column of cursor
      E6 3F         AND  03FH        ; limit to 0-3FH columns
      FD 77 6A      LD   (IY+06A),A  ; save in MWA
      E5            PUSH HL          ; save basic text pointer
      CD CC E9      CALL MOVECURSOR  ; and move to new cursor
                                     ; location
      E1            POP  HL          ; get back basic pointer
      C9            RET              ; return to OUT command
```

A more involved INP routine will now be discussed. These involve special routines in the PAC. This example will simulate the VARPTR command of the TRS80. It finds the address in memory of a variable, or string and puts this value (16 bit) in the basic variable on the left side of the equal sign. See my column in the June/July issue of PORT FE for a description of the VARPTR routine inside the PAC (CD54)

```
Format:  XX=INP(00),AB    ; to find location of variable AB
         XX=INP(00),AB$   ; to find location of AB string
```

This can be used to put a machine language routine inside a string at the very top of memory instead of in zero page. The machine program is then protected until basic performs a 'CLEAR' statement which will wipe out the storage string.

Example:

```
10 REM THIS DEMO PROGRAM SHOWS HOW TO PUT A RELOCATABLE
20 REM MACHINE LANGUAGE PROGRAM INSIDE A STRING LOCATED
30 REM AT THE VERY TOP OF MEMORY.
40 :
50 REM THE DEMO PROGRAM WILL JUST FLASH ASTERISKS THEN
60 REM SPACES ON THE SCREEN UNTIL CONTROL C IS PRESSED.
70 :
80 CLEAR100: REM ALWAYS OPEN UP SPACE FOR THE PROGRAM
85 :
90 POKE 318,195 :REM POKE IN NEW VECTOR FOR 'INP' COMMAND
100 POKE 320,0:  REM SO IT WILL JUMP TO 0000H
110 :
115 REM NOW POKE IN THE 'VARPTR' ROUTINE WHICH FINDS THE MEM.
116 REM LOCATION OF THE SPECIFIED STRING'S ADDRESS POINTER
117 :
120 FOR I=0TO20:READ A:POKEI,A:NEXT I
121 :
122 REM *** IMPORTANT ***
123 :
124 REM WE MUST EXPAND B$ WITH ANY DATA SO IT WILL BE THE
125 REM CORRECT SIZE AT THE TOP OF MEMORY FOR OUR ROUTINE.
126 REM BEFORE DOING THIS, B$ WOULD TAKE UP 0 BYTES OF
127 REM MEMORY - AFTERWARDS IF WOULD LEGALLY TAKE UP 43
128 REM BYTES OF ''RESERVED'' MEMORY. NOTE: THE MAXIMUM
129 REM VALUE OF 'I' WILL VARY WITH THE LENGTH OF YOUR
130 REM MACHINE LANGUAGE PROGRAM
131 :
135 FORI=1TO43:B$=B$+" ":NEXT
136 :
140 XX=INP(00),B$: REM CALL VARPTR ROUTINE AND FIND ADDRESS
145 REM                 OF STRING POINTER FOR B$
146 :
147 REM NOW GET THE TRUE ADDRESS OF B$ VIA THE STRING POINTER
148 REM IN LOWER MEMORY
149 :
150 LOC=PEEK(XX+2)+PEEK(XX+3)*256
152 :
153 REM AND POKE THE LOCATION OF B$ INTO THE USR JUMP
154 :
155 POKE260,PEEK(XX+2):POKE261,PEEK(XX+3)
160 :
170 REM NOW POKE IN OUR DEMO PROGRAM INTO THE STRING
180 :
190 FOR I=0TO41:READ A:POKE LOC+I,A:NEXT I
200 :
210 A=USR(0): REM AND NOW EXECUTE DEMO PROGRAM
390 :
400 DATA 225,225,227,205,122,197,44,205,205,198,43,205,84
410 DATA 205,122,83,227,229,195,12,207
490 :
500 DATA 62,32,33,128,240,17,129,240,1,127,7,119,245,237
510 DATA 176,205,21,224,194,0,01,33,255,255,43,124,181,32
520 DATA 251,241,254,32,40,4,62,32,24,220,62,42,24,216
530 END
```

The following is the assembly code of the data statements found in lines 400 and 410 of the demo program above. When called, it will find the address in memory of a variable or string and return this value in the 'XX' basic variable specified.

```
0000:E1          POP  HL          ; destroy return to INP command
                                  ; since we want to return a 16
                                  ; bit value and not the 8 bit
                                  ; value that INP expects.

     E1          POP  HL          ; get return to interpreter
     E3          EX   (SP),HL     ; HL= basic text pointer
     CD 7A C5    CALL SYNTAX      ; check for a comma ',' after
                                  ; the XX=INP(00)
     2C          DEFB ","         ; value to be checked
     CD CD C6    CALL #SCAN       ; jump over spaces in basic
                                  ; text - or jump to location of
                                  ; variable argument in command
     2B          DEC  HL          ; dec. basic text pointer for ret
     CD 54 CD    CALL VARPTR      ; go get variable's mem. loc.
     7A          LD   A,D         ; A=LSB of variable loc.
     53          LD   D,E         ; D= MSB of variable loc.
     E3          EX   (SP),HL     ; save basic text pointer
     E5          PUSH HL          ; save interpreter return addr
     C3 0C CF    JP   STOREVALUE  ; store variable addr found
                                  ; in D,A in the floating point
                                  ; basic variable in command.
```

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

This now concludes the final issue of PORT FE

Let me finish by saying that it has given me pleasure to be able to help so many Sorcerer owners throughout the world. My personal thanks again to all those that have helped in the past.

Keep up the good work

Good Luck and God Bless

Yours Sincerely

H.A. Lautenbach CSW,IES,VE3IMB