SORCERER'S APPRENTICE

INTERNATIONAL COMPUTER USERS GROUP/NEWSLETTER

PAGE 49
VOLUME 4
NUMBER 3
April 15, 1982

Price $3.00

IN THIS ISSUE -

DUEL - A Dogfight in space

by Don Gottwald

At last a game for two players. Now you can play against another individual instead of against the machine. Duel is an addictive game, once you start you can't quit.

The game is supplied on a good quality cassette in a Norelco case. It loaded the very first time without any CRC errors - which is unique on my system. You must use the Monitor command LO to load the program (do not LOG). When the program is loaded, type GO 100 and you will be treated to a superb graphic display of two "X-wing" fighters shooting at each other. Then the copyright message is displayed. Because of the 0100 starting address, this program can easily be put on disk to run as a CP/M .COM file and so you can enjoy the speed of your disk system rather than having to resort to tape.

To start the game you must choose the ship speed, rotate speed, missile speed and firing rate. The choices are from 1 to 4. You'll have to compromise with your partner, since the selection is for both ships. The background is outer space with a different star map each time the game is started over. If you don't like your selection of speeds above, then just hit RETURN at any time and start all over. If you do like the selection, then press the letter 'G' and the game starts. The fighters can be distinguished from one another quite readily. To play the game, the left fighter is controlled by the keyboard letters (Z), (X), (C), (V) and the right fighter is controlled by the cursor pad symbols (+), (0), (.), (=). You can rotate right (Z & +), rotate left (X & 0), forward speed (C & .) and fire (V & =). While initially awkward, this arrangement quickly proved comfortable.

Each of the fighters are initialized with 200 units of fuel and 200 missiles. A running tally in the upper corners reflect usage. A point is scored each time you score a hit on your opponent. When a fighter is hit by a missile it explodes realistically and then reappears randomly in another portion of the battlezone. Watch out for probes and comets. If one of them hits your ship you blow up and lose five points. If you hit one with a missile you'll score five points.

The realism involved in the game is astounding, you really feel in control of the fighter. This is a very exciting game and a must for any gaming enthusiast. My younger son prefers this game over the Atari games and he is an expert at these things. For the $20.00 you'll spend many hours of enjoyment. It is well worth double that price.

Emiliano DeLaurentiis is an educational psychologist who has done a great deal of work analyzing the suitability of program designs. He was going to review this program but was unable to do so due to other commitments. He did pass along some of his own impressions, however. He too found this program captivating and very well executed. The keyboard was easy to use and the graphics were superb, even better than the Atari version. He found the graphics so well done that he would like to see the author write an article on how this was achieved.

We agreed that the game would be even nicer if it had sound effects - but that is such a small item you'll never even notice. I'm hooked and I usually don't go in for games.

ODDS AND ENDS

by Ralph LaFlamme

I would like to thank Don Gottwald again. He put this issue together for me while I was out of town.

I'm going to re-print the comment column in issue #34 of the Australian newsletter, S.C.U.A. It seems appropriate to echo Ian Macmillan's frustrations here since it reflects our situation also.

The only purpose of this newsletter is to inform, instruct and amaze, and the only way it is kept going is by people contributing articles. If interest ever lapses to the point where there is nothing to publish, the newsletter, and our group, will have run its course, and I can go back to writing programs for fun and profit.

I occasionally hear from members who want to see more of this and less of that, but don't realize that I only edit the thing using the best I've been sent...I cannot and will not write it all.

Mind you, I am a bit biased about what articles get first go. They have to be informative, instructive, or amazing. I only rarely find room for programs as such; most listings are either utilities, or illustrate some technique.

So keep those contest articles rolling in...it looks as if we may have to generate a Bumber Issue to cope with some of the long, long articles though. Beginners articles, by, for, and both are especially welcome.

Ian Macmillan

We have reciprocal exchange priviledges with the following newsletters:

E.S.C.
32 Watchyard Lane
Nr. Liverpool
ENGLAND L37 3JU

S.C.U.A.
P.O. Box 144
Doncaster, Vic. 3108
AUSTRALIA

PORT FE
P.O. Box 1173 Sta. 'B'
Downsview, Ontario,
CANADA

Roger Hagan Associates is offering for sale a TI-59 program on magnetic strip called Decimal-Hex conversion. It handles conversions both ways and can display the binary equivalent. It sells for $10.

Roger also encourages owners of Spreadsheet programs to send him their original copy for an update provided they received their copy prior to May 1982.

I would like to remind everyone that this newsletter is done on a VOLUNTEER basis and as such must be fit in with the the more mundane but necessary realities of life such as family and jobs. The published deadlines for release of issues are intended as a guidelines only. Equipment failures, difficulty in getting articles in on time, etc. all affect the timelyness with which an issue is released. Please keep this in mind. We are not McGraw-Hill.

ROM PAC NOTEBOOK NO. 2

by J. de Rivaz B.Sc.(Eng.)

Introduction:

This note follows on from last month's article, (Vol. IV, #1) which mentioned a no wait keyboard scan routine. The routine to be discussed was used in the game "Cryonics Society Organizer" published in Sorcerer's Apprentice Volume 3 no 6 page 122 and 123. Readers who don't have this issue can obtain one from the publishers, for details see the back page. When the Inp(24) method of getting a single key entry is used, the statement X=Inp(24) gives in X a value which is the ASCII code of the key being pressed when the statement is called. However, if a key is pressed and held down, action ceases until it is released. This allows players to cheat during a real time game. Also, it means that in games where one presses keys on the pad to get movement on the screen, the repeat key has to be used to get continuous movement, or the pad keys repeatedly pressed up and down to get the extended movements.

In "Cryonics Society Organizer" action takes place continuously which may be modified by the player holding down keys on the numeric pad. In fact, the point of the game is that the player has to determine from the graphic display whether he is pressing the right key. (Each number attracts a person of some specific characteristic to join the society. If there aren't that many people with the characteristic represented by the number pressed present in the population, then the player would be better off by trying another number.) If by holding down a key all action stopped, the game would be impossible to implement.

A Further note on Inp(24):

The Inp function in BASIC works by calling a short routine in the RAM work area. This is:

```
        DB XX       IN A,X  ;X is argument of BASIC statement
        C9          RET
```
This can be modified by poking to:
```
        C3 XX YY    JP YY XX; X is argument of BASIC statement
```

YY is the high byte of the jump command, in the case of calls to the monitor it is 0E0H which is 244 decimal. The argument is the low byte, and in the case of a call to the keyboard scan it is 018H, or 24 in decimal. Therefore to call the keyboard sean from the monitor, Poke 318,195:Poke 320,224 and call Inp(24). A RET at the end of the routine called causes BASIC to execute the next statement. It is believed that this curious arrangement is due to the fact that the ROM PAC was written for 8080 code, which doesn't have an IN function. However it is a useful quick way of calling simple routine that returns a value in the accumulator which is passed back to BASIC. The Out function in BASIC can be similarly used to pass material to a routine.

This method can also be used to call a machine code program written in the spare memory from 0000 to 00FFH. In this case, poke 230 with 0 and make the argument of the Inp statement the low byte of the start of your program. This is the method used in "Cryonics Society Organizer" to call the no wait keyboard scanning routine.

Note on the Sorcerer's keyboard circuit:

There is no separate circuitry for scanning the Sorcerer's keyboard. It is done by the monitor. A value is sent to port FE of zero to F, and the result is read off from this port. The bits are tested to see which key is pressed. The values are as follows:

```
                                                          Bit
Value sent: 0      1    2 3 4 5 6 7 8 9 A   B C   D E    F    set
   shift   ESC   1 2 3 5 6 7 8 0 :  - none 7 9   3       4
   sh/lk   TAB   Q W 4 T Y U Q P [      -  8 6   =       3
   CNTRL   SPACE A S E G H J O ; ]  LF  /  4 5 none      2
   RPT     GRAPH Z D R V N I L . @ RET  *  1 2 none      1
   R/STP   CLEAR X C F B N K , / \ RUB  +  0 . none      0
   <          TYPEWRITER KEYBOARD      ><  PAD    >
```

Thus if 2 is sent to port FE and the port then read and bit 0 is found to be set, then key X has been pressed.

The routine:

A table is filled with the results required from each key press on the pad. The . is 10 and the = 11, so that two extra digits are available. A "Scan" routine increases IX until a key press is detected, and if so the value pointed to by IX is returned in A. As zero is a wanted result, no press is indicated by a return of 255. A listing of this routine is attached to the article. The numbers to poke in to from a BASIC program are to be found in "Cryonics Society Organizer". To run the subroutine, Poke 318,195:Poke 320,0 and call Inp(0).

In various graphics management programs sold by my firm, pictures can be drawn in either high or medium resolution on the screen from the pad. In order to rub out an incorrectly drawn line, all that need be done is to alter the shift configuration. In fact, this routine provides five shifts.

Key pressed	Location 40H
none	0
shift	1
shift/lock	2
CTRL	3
Graphic	4

continued on page 68

Better than BASIC

C/80 increases the efficiency of your Sorcerer, at ten times the speed of BASIC!

Triangle Systems **C/80** provides you with the speed of a compiler and the power of structured programming. Now, you can run programs up to ten times faster than before, with a lot less debugging. **C/80** offers you a valuable programming tool at a very reasonable price.

C/80 Supports:

- Character and integer types
- Pointers and arrays
- String constants
- All C math and logic
- Full function recursion
- All C control statements
- I/O redirection
- Standard C I/O library
- Dynamic storage allocation
- C preprocessor statements

$49. includes

C/80: compiler and library, **CASM**: absolute assembler, Sample **C/80** programs: file compression utility, file comparison utility, WP PAC file conversion. Triangle Systems includes a tutorial introduction to **C/80**.

C/80 needs at least 40K of RAM and either Exidy or Micropolis C/PM.

C.O.D. orders are accepted in the United States. Or, send check or money order, including $3.00 domestic, $8.00 overseas for shipping and handling. Add 5.5% sales tax in Ohio. Specify hardware configuration and software format. (1200 baud cassette, Exidy C/PM or Micropolis C/PM.)

Trianglesystems

1690 West Lane Avenue • Columbus, Ohio 43221 **614/486-3527**

COMPUTER SYSTEM EVOLUTION

by Eric Zorawowicz

Much of this story will sound familiar to all of you. I would imagine the evolution of any computer system is similar. However, each system has its little differences and idiosyncrasies and mine is no exception.

It all started in September 1979 with a Sorcerer I and a Leedex Video 100 monitor. Since I already had a portable cassette player, I was in business. It soon became obvious, that to do any serious programming, some hard copy was necessary. It was just too much trouble to keep listing parts of the program on screen. I got a Centronics 730 printer, which at the time looked like a good printer for the money. Since then so many improvements have been made in the printer field. There are several printers now available for half the price and twice the capabilities. But I can always say I've had mine to use for two years.

All the early problems with cassette tape have been well documented in newsletters and suffice it to say, I ran into most of the problems. It got very frustrating working with cassette tape so I started looking around for disk drives. I decided on the VISTA V-200 dual 5-1/4" drives which connect directly to the Sorcerer's bus and do not need the S-100 expansion unit. The only real problem is that these disk drives are not compatible with anyone else. Has anyone tried reading a soft-sectored diskette on a VISTA drive? These drives are memory mapped I/O driven and use ROM's in the address range D800-DFFF. Since I wanted to use the BASIC ROM PAC with the disk drives, I had to change two of the three ROM's in the disk controller to move it to the address range B800-BFFF. Now I had a CP/M disk operating system and could write BASIC programs and save them on disk.

It turns out that once you have a disk operating system, you find out that there are programs that need more than a 32K system to run, especially since CP/M is taking up part of your 32K. About this time the March 1, 1981 issue of Sorcerers Apprentice arrived showing how to upgrade a Sorcerer I to 48K. Well, this sounded great, and after designing a little extra logic, I used this technique to upgrade to a 46K system. An odd size, but it was necessary to logically eliminate the last 2K since that is where the disk controller ROM's reside.

Since then I have added an acoustic modem and use a CP/M-NET system to get public domain software. I have also used the Sorcerer based RCPM but find the cost prohibitive when calling from California, even on weekends. I wonder if there is any interest in paying a little more in membership dues to allow a toll-free number.

The latest addition to my system is a speech synthesizer, which I have interfaced to the parallel port and have programmed to talk in BASIC, C, and Assembly language. The next step is to rewrite some of my programs to have the Sorcerer talk to me while playing a game.

What's next? Well, that depends a lot on what the industry offers and how it can be adapted to the Sorcerer. Speech recognition, graphics printing, and home appliance control are already available and might be interesting to play with. There is no doubt that the Sorcerer will give years of enjoyment yet.

ZETU - CASSETTE BASED ASSEMBLER FOR THE SORCERER

by Carol E. Brown

I have been using ZETU - a cassette based assembler package which I purchased from SYSTEM SOFTWARE in Australia. It is so far superior to the EXIDY DEVELOPMENT PAC and so easy to use that it is great fun. The capabilities for string storage and reserved blocks of memory (within your program) are just the beginning. All editor commands are one control character. The SEARCH command positions the cursor on each occurrence of the search string for you to do whatever you want to it. The editor is character oriented so that anything from a single character to an entire line can be inserted when making corrections.

There are utilities for printing out source files and assemblies on your printer. Source files are easily merged, saved, loaded.. object files easily saved .. all this without worrying about vectors. At any time during an edit session, you can check out how much free space is left.

ZETU is really a bargain - fun and only $29.95. It far outperforms the development pac even with the extension.

<<< CLASSIFIED ADS >>>

$1/line $1/line

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

Sorcerer Model II, 48k, Zenith green 12" monitor, WP PAC, Centronics 737 printer -- $1000.00. Call Wim Plaat at 614-486-3527.

RANDOM I/O

by Don Gottwald

J. A. McNabb of P. O. Box 2, Trenton, Nova Scotia, Canada, B0K 1X0 is a Ham (VE1DZ) who would like to contact other Hams interested in using a Teletype with the Sorcerer. He has built a terminal unit (QST-Dec 80) and is looking for a program to make it work.

Mike Ward of 788 Holbrook Place, Sunnyvale, CA 94087 is looking for a WP PAC from someone who may have upgraded to a disk based word processor. If you have one you no longer use - send a note to Mike.

Richard Stone of Everett, WA replies to F. C. Creed's problem of system crashes with the following advice:

The system crashes appear to be a case of weak memory chips. They don't store data correctly until they are warmer. If the program is in RAM, the best way to test for this would be to do a long memory test just as the machine is turned on, in the morning when it is still cold. However if the program is in ROM (WP Pac) then either the ROM itself or the connection between the ROM and the computer is acting up. Make sure the contacts are clean and the Pac is inserted correctly. The only way to solve the sporadic memory problem is to find the weak memory chips (painstaking) and replace them.

The DEV Pac should be usable with a disk system. One must make sure that the memory space used by the disk system (for the BIOS, etc.) is not overwritten by the DEV Pac. Perhaps the best way would be to change HIMEM (stored in F000,F001) to protect the space you need. This should cause the DEV Pac to adjust the various buffers accordingly. Object code should be diskable if you ORG it at 0100 then load it as a COM file.

To Dr. Matheson, Richard replies: There is no disk system running CP/M at a reasonable price. The NEC printer will print bi-directionally only if it was designed to do so - it should state so in your manual. If it is one of the new 'smart' printers, it could in theory be re-programmed (in a new ROM) but that would be a formidable task.

The question about writing a BIOS for reading hard sector disks with the Exidy controller is not answerable without additional information. If the controller and the support hardware are capable of dealing with the index pulses arriving once per sector instead of once per revolution, it could theoretically be done.

The Sorcerer CANNOT be run at 4Mhz (even with a Z80A and a change of the clock crystal) since the clock controls not only the processor, but the cassette/serial I/O speed, the memory refresh rate, and most importantly, the video control timing signals. Speeding up the clock would cause the video output to be all bent and twisted. The monitor/TV could in theory be adjusted to operate at this higher speed, but I doubt that many monitors/TV's would have the approximate 60% adjustment range needed, thus requiring extensive reworking of the horizontal and vertical circuitry.

I. W. Metzger of Minnesota writes that a Radio Shack Model CTR-80 cassette recorder in combination with a simple VU meter gives him 99% reliability at 1200 baud. He also uses 90 minute TDK cassettes for WP files with no difficulty. Terry Walters Power/cassette monitor (SA 3.8) was a super idea - works great. He is also experiencing difficulty with keyboard bounce when in the monitor or Basic. WP Pac seems to ignore them. Anyone with a cure?

TRS-80 LEVEL II CASSETTE BASIC

by Robert Lansdale Jr.
18 Ashfield Drive
Etobicoke, ON Canada M9C 4T6

This is NOT an emulator but the real TRS-80 Level II BASIC. I have modified it extensively for the Sorcerer Input/Output such as 1200 baud cassette I/O and changes to the screen.

I would recommend that you buy the Radio Shack Level II BASIC reference manual from Radio Shack as it would give a greater description of the functions of this BASIC.

If you do not want to build the Radio Shack tape loading circuit (Fig. 1 on page 66), then send the tape to me and I will return it to you as a 1200 baud Sorcerer formatted tape. Any TRS-80 model I, Level II games can be run under this modified BASIC with little or no modifications needed.

RADIO SHACK BASIC INTERPRETER CHANGES:

In order for any R/S program to run on the Sorcerer, the interpreter has been modified and new commands added to simulate R/S memory mapped hardware. A number of utility commands are also included to make the I/O to the outside world easier; such as game paddle input.

You must have a 32K or greater Sorcerer to run this BASIC with any programs.

These are the interpreter changes and extensions:

- -to load in the TRS-80 BASIC, remove the Standard BASIC ROMPAC and 'LOG' or 'LO' then 'GO 0000'
- -press RETURN to 'MEMORY SIZE' question when BASIC is coldstarted. This gives you the same amount of memory as a 16K TRS-80 computer if you have a 32K Sorcerer.
- -the BASIC's coldstart address is '0000'
- -to return to BASIC if you leave to go into the MONITOR, type 'PP' just as in Standard BASIC
- -the monitor work area and the monitor stack have been moved down to 3B01H to leave the top of memory free
- -the monitor's prompt has been changed to a '*' so as not to be confused with R/S BASIC's '>'
- -to enter the Sorcerer's monitor type 'BYE' as in Standard BASIC
- -any machine language routines may be POKEd between 3B70-3B99
- -press 'ESC' (SEL) key to break the program execution BUT control B for break during a LIST or if the computer is in the INPUT mode (ex. - INPUT AB$) or in command mode
- -shift 4 on the numeric keypad deletes the entire line you just typed in
- -shift RUB is the backspace key now
- -LLIST and LPRINT are set for a MX=80 printer
- -graphic characters 192-255 don't print as tab spaces but as normal Sorcerer graphic characters
- -the Sorcerer graphic character set normally found from FC00 to FDFF have been moved to FE00 to FFFF
- -shift 8 on the numeric keypad is used to get out of any edit sub-command
- -CHR$(23) converts the screen to 32CPL mode
- -CLOAD? is still the R/S tape verify command
- -BASIC FOR loops and the entire program runs faster on a Sorcerer, so you may have to re-adjust timing loops
- -no BASIC keywords are allowed in the CSAVE filename, ex. CSAVE SINGLE has the word 'SIN' which is a keyword
- -Exidy Standard ROMPAC BASIC programs are not compatible with TRS-80 BASIC because of the difference in tokens
- -single keystroke input is allowed - these are the most commonly used:

FOR	= Graphic 2
NEXT	= " 8
DATA	= " 9
GOTO	= skip key
IF	= W
GOSUB	= R
EDIT	= F
BYE	= @
LIST	= Numeric keypad 9
POKE	= " " -
THEN	= Graphic shift ':'

A serial printer can be driven by placing the address of the serial printer driver routine at 4026H and 4027H

NEW COMMANDS

Memory between 3000H and 3FFFH is used as memory mapped I/O on the TRS-80, but was free in the Sorcerer, so I put it to good use and added the following commands. These are changes to resident commands (as in PEEK and POKE) and ones to imitate memory mapped keyboard (as an example).

-PEEK(X)	- If X is between 15360 and 16383, which are the screen addresses of the TRS-80, then X will be automatically changed to the Sorcerer video locations of F400H to F7FFH. Therefore no changes will have to be made to programs accessing the TRS-80 screen.
-POKE(X,Y)	- The same rules for the value of X as described in PEEK(X) apply.
-PEEK(14337)	to PEEK(14464) This is the memory mapped keyboard of the TRS-80. In place of this, I have added the 'KEY(X)' command. First find what character is to be found at the PEEK location (look in the TRS-80 BASIC manual on page C/1), then look up the character on the following chart:

To see if the key was pressed use:

```
IF (KEY(COLUMN#) AND ROW#)=0 THEN PRINT
"KEY HIT"
As an example, to see if 'X' was pressed:
IF (KEY(2) AND 1)=0 THEN PRINT "X pressed"
If no key was pressed, the result would have been 31
```

CHART

ROW NUMBER

1	2	4	8	16	COLUMN NO.
STOP	GRAPHIC	CONTROL	SHIFT LOCK	SHIFT	0
CLEAR	REPEAT	SPACE	SKIP	SEL	1
X	Z	A	Q	1	2
C	D	S	W	2	3
F	R	E	4	3	4
B	V	G	T	5	5
M	N	H	Y	6	6
K	I	J	U	7	7
,	L	O	9	8	8
/	.	;	P	0	9
`	@	]	[	:	10
_	<CR>	LF		-	11 These keys are on numeric keypad.
+	*	/	-		12
0	1	4	8	7	13
.	2	5	6	9	14
			=	3	15

JOYSTICK INPUT

Refer to February 1979 Kilobaud, page 135 for the hardware joystick interface. Attach the output of the timers to bits 0-3 of the parallel input port. Attach the TRIGGER line to output bit 3 of the parallel port.

To read the value of the timers, use this command:

```
A--PADL(B*256+C)
```

```
Where B=0 for a 2 MHz clock
       =1 for a 3.16 MHz clock (speed up mod.)
      C=1 for joystick number 1
       =2 for joystick number 2
```

=3 for joystick number 3
=4 for joystick number 4

Upon return - A=0 if fire button was pressed on unit #1 or unit #3; 0 is the value for that timer (0-255)

ODD COMMANDS

NOISE -where 0<X<32767
-This creates complex sound effects through the parallel port
-See SOUND(X) for the sound hardware circuit

TEXT(0) -This makes the screen 16 lines long (Normal R/S format, BASIC is brought up in this mode)

TEXT(1) -This converts the display to 30 lines long (the PRINT@ position maximum is now 1919 and SET(X,Y) can have values of 0-127, 0-89)

SAVE -This is the R/S format for CSAVE

LOAD -This is the R/S format for CLOAD

CLOAD NAME UNIT# - Sorcerer Tape format LOAD
CSAVE NAME UNIT# - Sorcerer Tape format SAVE

BYE -go to Sorcerer monitor - Return to BASIC with 'PP'

OLD -Restores a NEWed program, or a BASIC program after a coldstart

SOUND(X)-where 0<X<255
-this outputs a tone through the parallel port bit 2
-attach the sound circuit of Figure 1

CALL(ADDRESS,ARGUMENT) - This command is easier to use than the USR command. If the ADDRESS >32767 then the ADDRESS must be changed to ADDRESS - 65536. Upon calling the routine, DE=ARGUMENT and A=LSB of ARGUMENT. One use of this is to convert a decimal number to hex directly by calling a monitor subroutine:

CALL(&HE1E8,32767) -- will print 7FFF on the screen

NUMERIC CONSTANTS - Different number bases can be used in this version of the interpreter.

Example - PRINT &HE1E8,&O1762,&D65520

&H = Hex
&D = Decimal
&O = Octal

APEND - This joins two BASIC programs together in memory. One program must be in memory and the other on tape. The lines of the two programs must not interfere with each other (no two line numbers are the same)

```
>APEND
LOAD AT: XXXX
*LO NAME UNIT# XXXX
*PP
READY
>APEND
LOAD AT: XXXX ---don't worry about this message
*PP
READY
```

INPUT#-UNIT# - These work exactly as in the manual
PRINT#-UNIT# - but have been modified to work on the Sorcerer cassette I/O at 1200 baud.

****

PAC BASIC TO TRS-80 DISK BASIC

PAC BASIC:

Sorcerer Standard BASIC has been relocated to 8000H with disk I/O residing from 7A59H to 7FFFH. It is completely bug free (I hope!) and will only work with 48k or greater Sorcerers with CP/M. It will also work with disk sytems other than DISCUS 2D.

All PAC BASIC programs that have any machine language routines that call PAC BASIC ROM routines must be modified s that the addresses are relocated from C000H-DFFFH to 8000H-9FFFH. All others will work exactly the same as in ROM PAC BASIC.

Another potential problem are programs that were saved on tape starting at 01D5H. An example would be that of a small machine language program residing at 0000H and saved as part of the BASIC program using the monitor SA command. This small program will have to be converted to data statements and POKEd into zero page when the program is loaded. Only BASIC programs starting at 01D5H can be saved on disk. This is because CP/M's zero page must be intact when the disk I/O commands are used in BASIC.

COMMANDS:

The filename given with the following commands conform to CP/M CCP's filename set-up.

DIR"A:FILENAME.TYP" - this prints the directory of drive A, 'A:' is optional, FILENAME.TYP is optional, if you change a disk in a drive between then do a 'DIR' to initialize that drive.

DSAVE"A:FILENAME" - this saves a BASIC program with filetype 'BAS' on disk A 'A:' is optional

DLOAD"A:FILENAME" - this loads a PAC BASIC program of type 'BAS' from disk into memory 'A:' is optional

ERRORS:

FF Error	- file not found error do a 'DIR' to initialize the disk or check to see if the program is really on the disk
DF Error	- disk full error
NB Error	- not PAC BASIC filetype 'BAS' program

CP/M COMPATIBILITY

CP/M and PAC BASIC reside in memory at the same time.

CP/M Zero Page	0000 - 00FFH
BASIC Work Area	0100 - 01D5H
BASIC Text Area	01D5 - 7A58H
PAC BASIC	7A59 - 9B80H
CP/M	9D00 - BFFFH

To return to CP/M from BASIC:

--- BYE & GO 0000 or BYE & GO D000

This will only work if the BASIC program has not destroyed zero page. It might be saver to GO D000. This will not destroy PAC BASIC. You can ERA, SAVE, and DIR in CP/M then return to the monitor and 'PP' back to PAC BASIC without destroying anything.

****

TRS-80 BASIC

Radio Shack BASIC has been modified so the PAC BASIC routines will be able to reside inside the BASIC itself. The conflict between zero page RST's of the R/S BASIC and CP/M zero page needs have been overcome by a sharing system.

CP/M and TRS-80 reside in memory at the same time but do not interfere with each other. A minimum of 32k is needed to run TRS-80 and CP/M at the same time and 48k is needed to load and run R/S programs off the disk.

ERRORS:

The only error different than for PAC BASIC is:

NT Error - not TRS-80 filetype 'TRS' program

COMMANDS:

The only difference between the two BASIC's is:

CPM - this will return directly to CP/M's >A
DSAVE"A:FILENAME" - same as PAC BASIC except filetype is 'TRS'

OLD -This UN-NEWs a BASIC program after a cold start or if the program was NEWed. See the following text for its use.

CP/M COMPATIBILITY

To return to CP/M without destroying the BASIC, type 'CPM'. In CP/M you may DIR, SAVE, and ERA a program. One other possibility is to 'STAT' a program but you must reload TRS80 BASIC if you do.

To return to BASIC from CP/M, go to the monitor and type 'PP'. BASIC will then be warm started.

It is possible to go to CP/M, do a STAT, reload TRS80 BASIC, and recover the BASIC program that was there before by typing 'OLD', which 'un-NEWs' a newed or cleared out BASIC program.

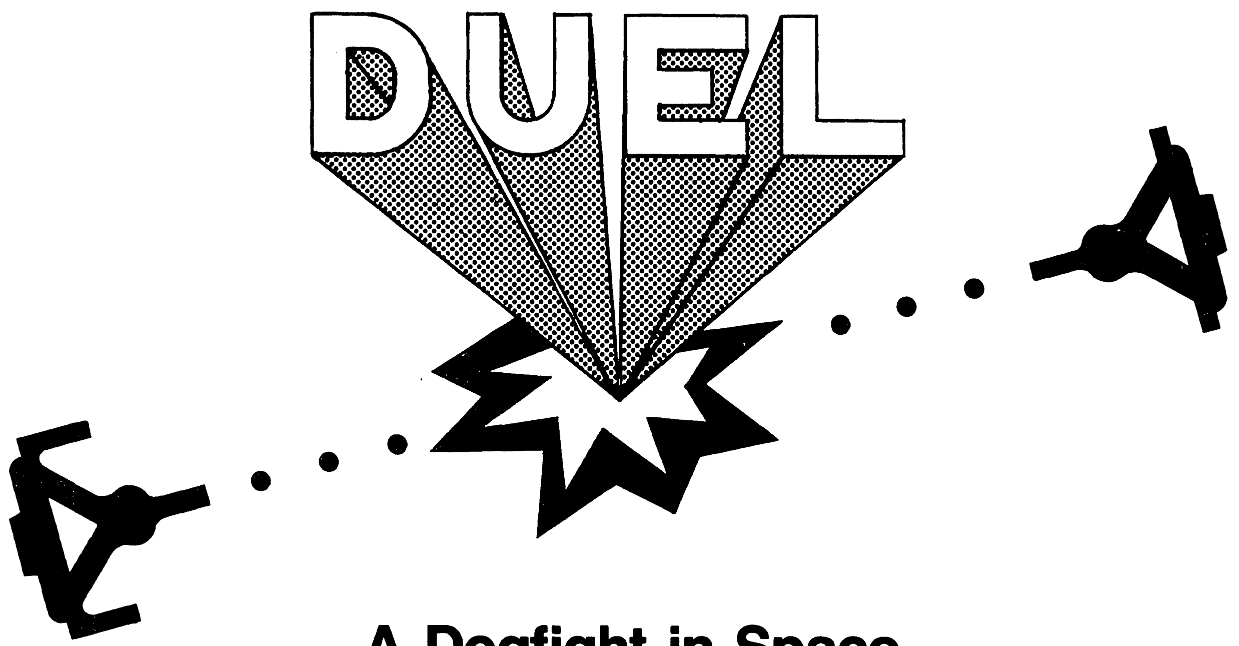
If any of the disk commands do not work, go to the monitor and cold start CP/M with a GO D000, return to the monitor and 'PP' for both of the BASIC's. Now the commands will work.

MEMORY MAP:

```
0000 - 0007H -CP/M warm boot and BDOS entry point.
0008 - 005DH -TRS80 BASIC RST's & system entry points.
005E - 00FFH -CP/M FCB's and disk buffer.
0100 - 3A60H -TRS80 BASIC.
3A61 - 3FFFH -Disk I/O routines.
4000 - 42E9H -BASIC system usage.
42EA - 9CFFH -BASIC text area.
```

****

FOR THE EXIDY SORCERER™



A Dogfight in Space

- a real-time graphics game for two players
- written in machine language for the Exidy Sorcerer™
- graphics characters continually redrawn for smooth, high-resolution movement
- each ship realistically accelerates, rotates, and fires
- 16K required
- $20.00 for cassette (includes shipping) MasterCard VISA

We think you'll like it!

DAYSPRING

COMPUTER ENTERPRISES

P.O. BOX 1910 • EUGENE, OREGON 97440 • (503) 689-7409

STRUCTURED BASIC

A Review by Tom Bassett

"Another Basic? How (yawn) interesting."

Such was my reaction to reading the first ads for a new Basic compiler, written by Topaz Programming and distributed by Micro-Ap, Inc. Its name: S-Basic.

Having spent some time exploring the mysteries and experiencing the frustrations of standard Basic, I was ready to broaden my horizons. Magazine articles by software guru's had been extolling the virtues of the structured languages; Pascal, PL/I, C, Algol, and others. This seemed to be the way to go. So who needed another Basic?

Then I read a review of S-Basic by Bill Burton in the March, April, and May 1981 issues of Lifelines magazine. It seemed that S-Basic was much more than just another Basic. I decided to give it a try. An order to Lifeboat ($295) brought the disk and documentation in short order.

Do I like It?

I've been using S-Basic for some months now, and have written several programs and translated several others from standard Basic and from Pascal. I have written an extensive Personal Accounts Payable System in S-Basic; I don't think I could have written it as successfully in a standard Basic. The language has become my "standard" language. I'm hooked on structured programming. In short, I really like S-Basic.

This review will necessarily be a brief one; for a fairly thorough exploration of the language, see the articles by Burton in Lifelines. If your appetite is whetted, order the documentation manual. It is not a teaching manual, but a programmer who has used standard Basic successfully will have little difficulty understanding the documentation.

A Teaching Language

In my opinion the major feature of S-Basic is that it affords the Basic programmer an excellent means of making the transition from unstructured languages to the structured discipline. It does this by means of being almost a combination of Basic, Pascal, and PL/I.

For example, a program written in standard Basic will, with some edits, compile and run under S-Basic. The major edits will be in two areas: all variables in S-Basic must be declared before use; and no multiple statements per line are allowed (they can be placed within a BEGIN - END block, where they are treated as one logical line).

At the same time, a program written in Pascal will, with some edits, compile and run under S-Basic. Here the major edits will be of syntax.

A PL/I program would require major editing, but the logic and structure of the program could be followed.

So here is the real strength of S-Basic as a teaching language. The Basic programmer can write his program in a style with which he is familiar, and then experiment with the constructs of a structured language to accomplish the same result. All within the same program. And, as he learns to use the features of S-Basic, he will find himself possessed of an extremely powerful language.

Language Structure

So, what does this accomplish? What's so great about a structured language?

Let's look at an example. In the Personal Accounts Payable System which I wrote in S-Basic, there are 19 separate programs. In all of these programs, I probably used only a total of a dozen REM statements to clarify the program logic; no more were needed. Perhaps only a dozen GOTO statements were used, and no GOSUBs. The programs are almost completely self-documenting; they read like a book. One starts reading at the top, and proceeds to the end of the program.

It's easy to see why the experts prefer "top-down" programming; it's much, much easier to read and understand the thing a month after you've written it. And certainly much easier to debug it!

S-Basic, like other structured languages, is replete with control constructions such as REPEAT - UNTIL, WHILE - DO, CASE - OF, and BEGIN - END. If the Basic programmer writes a FOR - NEXT loop, a bit of experience with the language will enable him to rewrite the loop as a REPEAT - UNTIL or a WHILE - DO. If the programmer is familiar with the Basic ON X GOTO or IF X THEN GOTO constructs, a bit of experimentation will enable him to write them as CASE - OF constructs.

Once these techniques are familiar to the programmer, he will find himself writing structured programs as a matter of course.

No Line Numbers!

And - glory of glories - no line numbers! Unless you need them, of course, in which case S-Basic allows you to be descriptive. As long as it begins with a numeral, any alphanumeric expression will serve as a line number. For example, 1.HERE.-IS.WHERE.THE.PROGRAM.BEGINS is a valid line number in S-Basic. My primary usage of line numbers has been as destinations for the ON ERROR GOTO error-trapping feature of the language.

Variables can be up to 128 characters in length, and, since variables are declared as to name and type, it is no longer necessary to use the $ to distinguish string variables. For example, BEGINNING.DATE can be a valid string variable; if it was declared at the beginning of the program as VAR BEGINNING.DATE = STRING:8, then memory space is reserved in the variable storage area of the program for an eight-byte variable; this is the maximum length it can obtain.

Since S-Basic is a compiler, a variable name of 128 characters takes up no more memory in the .COM file than a variable name of two characters.

Variable Types

S-Basic allows several variable types. The usual string, integer, and binary floating-point single precision (called "Real") variables are supported. In addition, a double-precision Real (14 digits of precision) is supported, along with a fixed-point BCD numeric with 11 digits, three of which are to the right of the decimal. Only two decimal digits of the Fixed variable will print, because the third is used to support automatic rounding.

A variable type called, interchangeably, Char or Byte is just that; one eight-bit byte. This may be an ASCII character (LET X = 'S') or a numeric value (LET X = 17H, or LET X = 23).

Any of these variables may be declared as COMMON for convenience in chaining program overlays; as VARIABLE for normal use; or as BASED for placement in memory wherever the programmer wants them.

A LOCATION statement, similar to the VARPTR in standard Basic, allows the programmer to find the memory location of, for example, the input buffer of a disk file, and then base the variables contained in each disk record right on top of the input buffer. So a statement such as READ #1,-RECORD.NUMBER, where RECORD.-NUMBER is an integer variable, will read the record RECORD.NUMBER of the file that has been opened as #1, and the program variables contained in that record immediately assume the values of the variable fields of the disk record.

This allows very fast file I/O, since no assignments must be made to program variables after a disk read or before a disk write. Arrays may be based in this manner also.

Another use of the BASED variable might be to base an array of type Char right on top of a string variable, so that the string could be easily parsed by just reading the array.

Binary and BCD Arithmetic

The availability of both binary and BCD numeric variables is an example of the flexibility of S-Basic. If the programmer wants high-speed number-crunching, he will use the binary variables. If he's writing a financial program in which his variables represent dollars and cents, he will use the BCD variable type. If you've ever experienced the frustration of manipulating a bunch of numbers in a binary floating-point format, and a calculation which you know should result in a value of 0 results instead in a value of .0000001, you'll appreciate the BCD arithmetic of S-Basic.

Procedures & Functions

A major feature of S-Basic is the availability of Procedures and Functions. A Procedure is much like a subroutine in standard Basic, except it is called by name instead of by line number. The big difference is that values can be passed to local variables within the Procedure, and other local variables can be declared within the Procedure.

What's a local variable? Simply a variable that has meaning within the Procedure or Function, but not within the main program. For example, the value of a main-program variable named TITLE can be passed to a Procedure local variable named NAME. NAME is manipulated within the Procedure, but at the conclusion of Procedure execution the variable TITLE is unchanged.

These local variables may also be declared within a BEGIN - END block of code. Any variable declared prior to a Procedure, a Function, or a BEGIN - END block is global to the local code. Procedures, Functions, and BEGIN - ENDs may be nested to any depth, with corresponding global-local relationships of the variables.

A Function is pretty much the same as a Procedure except that it returns a result, whereas the Procedure does not.

Functions can call themselves; recursive programming is supported. Functions can call other Functions or Procedures, or Procedures can call other Procedures or Functions.

Expression Type

One feature of S-Basic that takes a bit of getting used to, but which is a boon to the programmer once he understands it, is the concept of expression type. It is best illustrated by an example.

Suppose we want to parse our BEGINNING.DATE variable, and set an integer variable to the value of the day of the month. Our date variable is a string, and contains the value 03/25/82. We want to set the integer variable DAY.OF.MONTH to the date value of 25. In standard Basic we would write:

```
LET DAY.OF.MONTH=VAL-
(MID$(BEGINNING.DATE,4,2))
```

using the function VAL to convert a string value to a numeric value.

But in S-Basic the same expression can be written:

```
LET DAY.OF.MONTH = MID$-
(BEGINNING.DATE,4,2)
```

without using the VAL function. This is because the expression type is set by the first variable encountered on the left of the expression, and the variables on the right of the expression are converted to that type. The conversion is automatic. Very handy. Also, very dangerous if you don't pay attention to expression type!

Goodies

S-Basic is full of goodies. For example, the function TEXT allows the programmer to create formatted screen displays without all the PRINTs and TABs necessary in standard Basic. For example:

```
TEXT 0,*
Select:
1...Do this
2...Do that
3...Quit
*
```

will print everything between the delimiting characters "*" just as it appears on the screen! No quotes are needed.

Standard Basic insists on printing a question mark as a prompt for an input statement whether you want it or not. With S-Basic, the programmer has a choice of four input statements. INPUT is the same as always. INPUT1 prints the question mark but does not do a carriage return. INPUT2 omits the question mark but prints the CR. INPUT3 omits both the question mark and the CR.

INPUT statements default to the console, but another device may be specified. INPUT #0 is the console; #1 and #2 are dummy devices; #3 is the CP/M reader device; #4 will input console status; and #5 is a "keyin" device that will return 255 if a key is down or 0 if not, with the carry bit set in the PSW. Thus, an input can be done to see if a key has been pressed.

The standard Basic INKEYS function is emulated by inputting to a variable of type Char or Byte; no carriage return is needed after the key is pressed.

Here's a real goodie. Input channel #9 is used when an S-Basic file is loaded with a second file name, or if the second file name is placed in the FCB at 5CH by the program. INPUT #9 can be used to "read" the specified file. The following code will read a text file and "dump" it to the console (1AH is the end-of-file marker):

```
VAR LETTER = CHAR
REPEAT
  BEGIN
    INPUT #9;LETTER
    PRINT LETTER;
  END
UNTIL LETTER = 1AH
```

A PRINT statement, without an operand, directs printing to the console; so does a PRINT #0. A PRINT #1 statement directs printing to the CP/M list device. An integer variable can be used instead of the number, so that a PRINT #DEVICE statement prints on either the console or the printer, depending on the value of DEVICE. A neat solution to the PRINT/LPRINT problem in other Basics.

S-Basic does not include a text editor. A source file must be created and edited by the CP/M ED editor, or any of the standard text editors. Spellbinder makes a beautiful source file editor for S-Basic.

Using the Compiler

S-Basic is a one-pass compiler. It is called from CP/M with the command SBASIC FILENAME.NNN, where NNN is used the same as in the CP/M assembler. The extension .BAX, for example, will read the source file from the B: disk, send the .COM file to the A: disk, and list the .PRN file on the screen. No assembly or linkage is necessary after the compilation. The .COM file may be called from the CP/M command prompt just as with any other .COM file. No run-time package is necessary.

But a compiler is more difficult to use than an interpreter, right? Well, yes and no. After using the S-Basic compiler for some months, I have developed a routine that seems to work for me. Immediately after keying in the code, I compile the source file with the extension .BZX. This directs the compiler to read the source file from the B: disk, generate no .COM file, and send the listing to the screen. Even a big program, say 800 lines, will list in 3-4 minutes, and I immediately see all my keyboarding errors.

Error messages are easy to spot, and the location of the error within the offending line is pinpointed by the error-checking routine. I can hit the run-stop key while I note the error (I am using Brian Lewis's CBIOS for CP/M 2.2), and then continue to the end of the listing. The source file can then immediately be edited, and another compilation listing made.

Once the thing will compile on the screen without errors I do a full compilation, generating a .COM file for testing. Here I use the extension .BAY; the Y directs the program listing to the printer.

The full compilation does take a while: that 800-line source file I mentioned above might take 10-12 minutes to generate a .COM file. But by this time I'm sure it will compile without error, except perhaps errors in logic that the compiler can't trap. For these errors, S-Basic is equipped with full trace and error message generation capabilities. But once the programmer has a working program, he can set a toggle ($LINES) within the source code that disables this error-tracing function, and save about seven bytes per line of source code and increase execution speed.

Chaining

S-Basic supports full program chaining with Common variables, machine language calls, and, in addition, has a neat little EXECUTE statement with which the programmer can call a machine language .COM file, and, upon its completion, call another S-Basic .COM file. This entails a use of the CP/M Submit facility from within the S-Basic program. Very neat.

The CALL statement will call a machine language routine; and, if desired, will pass values to the CPU registers and recover the register values upon completion of the machine language code. This code can be included within the S-Basic .COM file by means that are too lengthy to explain here, or the S-Basic .COM program can load the machine language file into a character array at run-time, thus accomplishing the same thing.

Niceties

Other niceties are present in the language; one can toggle the screen echo of typed characters on and off, for example -- useful in inputting a secret password with the whole world watching.

Rather than using REM statements for each line of commentary, one can use the COMMENT - END block structure and type anything he wishes within those commands -- the compiler will ignore all of it.

If the entry of a control-C from the keyboard would endanger file integrity (by exiting the program before a file is closed), the programmer can toggle CONTROL.C.TRAP OFF (or back ON).

The compiler itself incorporates a $INCLUDE directive with which one can insert and compile one of a number of routines or procedures from a library file. If the routine is lengthy, and the programmer doesn't want it to list in the compiled listing, he can toggle the listing off and on with $PRINT.

I mentioned at the beginning that this would be a brief look at S-Basic, and I seem to have written a book. But I have really only skimmed the surface of S-Basic features; Burton, in his Lifelines review, went into much greater depth but still didn't discuss all of the language's features in three magazine articles.

And, the Faults

So, a quick closing look at the faults of S-Basic.

continued on page 59

ROM PAC NOTEBOOK NO. 3

by J. de Rivaz B.Sc.(Eng.)

In this issue I propose to discuss a method of creating WP files from text produced by the Dev Pac and the BASIC Pac, and a method of saving the numerical array area of BASIC programs using the monitor routines.

WP Files:

If one is writing an article on computing, such as this one, it may be necessary to add output or data from BASIC or the assembler. The procedure is as follows:

1. Select a memory area for your store. In the routine it is chosen at 3000H, but it can be anywhere large enough to hold the file.
2. Enter the short routine given below. If you are using BASIC set the output to line 9, not the line 8, which is the input point from the Dev Pac. If entering the routine elsewhere, modify the load addresses relating to the store to wherever you have put your store. Don't forget that the low byte is first, high byte second.
3. Set the Sorcerer's output to the input of the routine.
4. Run the program so as to fill your file. Remember that if you re-run the program, the file will fill from where it left off. Therefore if you don't want this, re-set it.
5. Save the file onto a cassette.
6. Switch off, insert your WP pac.
7. Create text longer than your file. I suggest type in a few lines of asterisks, put then in the H file, and the use command U to call them into the main file in blocks. If you are inserting the new text at the end of some existing text, create the dummy text here. You can even insert it within existing text!
8. Put the cursor where you want the new text to start, and use command X to go to the monitor. Dump 748 and 749 to find the cursor address, to use as your start address. If you are at the start of text, then use address 080FH as your start address.
9. Load the text you have saved in 5 above, using the start address obtained in 8 above. Go to WP warm start 0C003H
10. Delete surplus dummy text from your WP file.

The WP file doesn't like 0AHs (line feeds). Therefore this routine omits them. There is no test for whether the file is overwriting anything it shouldn't. Therefore use it with care.

Now I'll use it to add the routine below!

```
 ADDR  OBJECT     ST

                  0001 ;
                  0002 ;Subroutine to allow Dev Pac
                  0003 ;to create WP text in a file
                  0004 ;after assembly.
                  0005 ;
 >E01B            0006 VIDEO   EQU    0E01BH
                  0007         ORG    0BD00H  ;after DPX48
'BD00  7A         0008 INPUT   LD     A,D     ;Dev Pac has data in D
'BD01  FE0A       0009         CP     0AH     ;we don't want
'BD03  D5         0010         PUSH   DE
'BD04  280A       0011         JR     Z,NOSAVE-$
'BD06  ED5B15BD'  0012         LD     DE,(STORE) ;position counter for file
'BD0A  12         0013         LD     (DE),A
'BD0B  13         0014         INC    DE
'BD0C  ED5315BD'  0015         LD     (STORE),DE
'BD10  CD1BE0     0016 NOSAVE  CALL   VIDEO
'BD13  D1         0017         POP    DE
'BD14  C9         0018         RET
'BD15  0030       0019 STORE   DEFW   03000H  ;start of text file
```

... And there is the proof that it works!

Saving arrays from BASIC.

If you use the BASIC Pac's method of saving arrays, it simply doesn't work reliably enough for serious applications. One way of making the array load is to record a header of 32H zeros before the array starts. This is achieved by poking in the short routine give on page 17 of the Sorcerer Technical manual:

```
Poke         Hex                 Instr           Comment
6,1          06 01               LD B,1
205,36,224   CD 24 E0            CALL CMOTON     Turns on unit 1
6,50         06 32               LD B,32H        want 50 zeros
62,0         3E 00          LOOP LD A,0
205,18,224   CD 12 E0            CALL OUTAPE
16,249       10 F9               DJNZ LOOP-$
201          C9                  RET
```

Enter this as a USR routine and call it before the statement "CSAVE*".

However, this method still doesn't prevent errors being loaded. There is no CRC check with the Rom Pac's array load.

In the next issue, I shall be presenting a routine that saves the entire array area using the monitor routine. If an error is found on loading, instead of just stopping in the usual defeatist manner, the routine goes on to load the next file. Therefore making two savings of a file, even if there is an error on loading the first copy, the program automatically tries to load the second. You can save as many copies as you wish depending on the reliability of your cassette system.

Watch out for this column in the next issue of Sorcerer's Apprentice.

DISK NOTES

by Bryan Lewis

This month I want to bring to your attention some good public domain software. Most of it comes from the CP/M User Group or the SIG/M Group; some of them I wrote or revised for the Sorcerer. There's so much stuff available from the user groups that it's hard to keep track of what's good and what works for us. I've sent all of the programs to the Apprentice for posting on the Bulletin Board. There's a lot of it, though; you might prefer to send a disk to the Apprentice, instead of using the phone.

First four games, taken from the CP/M Group Volume 48, which is a sampler of the BDS C language. (The COM files don't require C to run). They don't use a lot of graphics, but they're well written versions of some classics. I'm borrowing the comments from those on the disk by Ward Christensen.

1. MM.COM: So you want to play a simple game of letter guessing, eh? Try this one. But watch your ego: it can be deflated. MasterMind is a "simple" game, that makes you think. The computer generates a random "word" consisting of 4 letters from A-F. You simply "guess" 4 letters at a time, and MM tells you how many are "hits" i.e. the right letter in the right spot, and how many are misses. Thus you deduce the missing pattern. The game goes a bit slow, however (at least at 2MHz). Why? MM is computing how many possibilities are left, based on the clues it has given you. When this number reaches "1", it says: "You should have it by now".

It becomes a real challenge to see how few times you can keep that message from coming out, and is a "real thrill" to "beat it" - especially a couple times in a row. However, having it "know" YOU should "know", but you "missed" catching on for, say 5 turns, makes you feel like a real dummy.

2. OTHELLO.COM: Have you the patience to beat this one?

3. STONE.COM: You get to specify how "hard" the computer works to beat you, and if you let it work a while, it's nearly un-beatable.

4. TTT.COM: Hmmm, what could be new in a Tic Tac Toe game? Well, brains, and wit for two things. Ex: it puts its "X", you put your "o", it thinks a while, and says "I've got ya". If it thinks for a while, and after several pieces are on the board, doesn't say "I've got ya", then you MAY be on your way to a rare win, or more likely a "cat" game. Play it and see.

5. TABIFY.COM. A non-game from the same disk. A nice utility to delete spaces from a file, inserting tabs where appropriate, based on the CP/M convention. This is handy for compacting a Spellbinder file, since Spellbinder expands all tabs to strings of 8 spaces. That can mean a significant expansion in the size of an ASM source file.

6. LIFE.COM and LIFE.DOC. This one came from Joseph R. Power; the assembler code was published in his Tsunami newsletter. I entered it, revised the shape of the little men, and saved it on disk. It's the old game of Life, first published in Scientific American in 1970. The Sorcerer allows full-screen editing for setting up the positions, and a lot higher speed than pencil and paper. See LIFE.DOC for rules.

7. SEARCH.COM is my utility to search through memory for an arbitrary sequence of bytes. After you run SEARCH, you will be asked

continued on page 59

to enter the sequence; enter hexadecimal values, up to 16 of them, with a question mark for any byte that you don't care about (a wild card). The source was published in the Apprentice of October 1980. This one is ready to run, with a couple of bugs fixed. Note: this doesn't need disks to run.

8. ASCIIFY.WPM is a word processing macro for Spellbinder. I wrote it so I could print out C programs. C uses a lot of characters that mean special things to Spellbinder, like curly braces and vertical lines. Hence a simple Print command won't work quite right. This macro goes through the text and enhances all those characters, so that Spellbinder does not recognize them as special. It will work for any file (C or not) that uses the reserved symbols.

9. SURVEY.ASM and SURVEY.COM are a neat little program to report the usage of your system's assets. It will display how much space is used and left over on your disk, which I/O ports are active, and how each kilobyte of your memory is used: RAM for the transient program area, RAM occupied by CP/M, ROM, or unused RAM -- it thinks the Sorcerer's video and graphics RAM are unused. This kind of information might even be useful to you if you're a dealer, configuring software for many different machines.

10. MSPEED1.COM and MSPEED2-.COM are the CP/M Group's SPEED-.COM, modified by me for Micropolis. MSPEED1 is for CP/M version 1.4, while MSPEED2 is for 2.2. But what's SPEED, did you say? It modifies CP/M's disk access routines, to buffer a whole track at a time in memory, not just 128 or 256 bytes. If you're doing something that involves a lot of disk activity, like assembling or compiling, this will save you **lots** of time, since writing a whole track to disk is faster than several sector writes. I measured a factor of two speed-up, when doing assembly. The disadvantage is a loss of memory: storing a track from a Micropolis disk can eat up 4K. If you specify all the options (buffered seeks, reads, and writes), you can use up 13K. That's not usually a serious loss when you're assembling or compiling. It sure is neat to give a DIR command and see an instant response without the disk clicking. (The directory is in memory!)

To use it, just type MSPEED1 (or 2), and you're set. For the rest of the instructions, read the two manuals, SMAN.PRT and FMAN.PRT. (Note: The first time you use MSPEED2, you may get a BAD SECTOR message. I don't know why, but just type CTRL-B and you'll be ok).

11. MENU.COM and MENU.ASM are an automatic menu generation utility for CP/M. Just type MENU, and you'll get a numbered table of the COM files on the disk. Just enter one of the numbers, and that file will be executed. This was written up in Creative Computing in December 1979, but I had to massage it to make it work on the Sorcerer. It will also create a menu of BASIC files, if you change a couple of options in MENU.ASM. Study the source code if you want to see how to poke commands into CP/M's command buffer for automatic execution.

12. MODEM7.COM (and MODEM7-.DOC for instructions) is the latest and greatest of the CP/M modem programs. You can do everything that you could with PLINK or any of the other modem-like programs, but more easily. For example, you can capture incoming characters onto disk, without leaving the program or even leaving terminal mode; just type CTRL-Y while you're on line. Similarly, you can start sending a file from disk just by pressing CTRL-T while you're on line.

Other niceties: you can display the disk directory without leaving the program. You can send multiple files to another computer, using a batch transmission mode, without having to sit and type in each new name.

You don't have to modify the source code to use this one; I've already done it. It should run as is, on a Sorcerer with a "fixed" serial port, i.e., one with a hardwired port, or Version 1.1 ROM's. You also don't have to use the SETMODEM and SETTAPE commands (see the December Apprentice, pp. 169-170); I've put the port initialization into the program.

The revisions are for an acoustic modem on the Sorcerer's serial port. If you have a PMMI modem board (you lucky devil), then get the original program, which came configured for the PMMI. It allows dialing, changing baud rates, and disconnecting, all from the keyboard. I've labelled that file M7PMMI.COM.

If you want to see how I made the revisions, or just want to learn how the program works, look at MODEM7.ASM and MODEM7.SET. Correction: one feature MODEM7 doesn't have is the trigger character capability of EXLINK. I've never needed that feature, but Bob Hageman has found a use for it, for sending bulletin board messages.

****

S-BASIC cont'd

Most of us expect a compiled language, since it runs in machine code, to be lightning-fast. S-Basic is fast (example: a do-nothing FOR-NEXT loop requires an operand of 4000 to kill about one second of time) but not as fast as a functionally-equivalent assembly program. This is not really a fault, just a fact of life with compilers.

The same applies to the size of the object code module. S-Basic .COM files are not compact; I estimate a bare minimum of 4k in overhead run-time code for a simple program, and some of the 800-liners I've written compiled to a good 30k of object code. Again, a fact of life with compilers, not a fault of S-Basic.

Error messages are another thing, however. It takes a bunch of getting used to the cascade of error messages that can follow a simple typo in a source file compilation listing. Once the programmer gets the feel of the logic, however, this is less of a handicap.

And, finally, documentation. As I mentioned earlier, the S-Basic manual is not a teaching document; it assumes a fair knowledge of programming. But, even for one who has hacked around in Exidy Basic for a couple of years, it is readily understandable. At least after a few readings.

But this fault-finding is really nit-picking; S-Basic is, in my opinion, a major language, and really is the only language for one attempting the transition from Basic to a structured language. I really like it, as you can probably tell from the tenor of this review.

And "ah, the delights of structure!" Like caviar or pickles, you can't appreciate it until you try it!

****

IN THE PUBLIC DOMAIN

by Bruce Blakeslee
CP/M - Micropolis Librarian
906 Crestwood Road - West
Westfield, N.J. 07090

As I have agreed to take on the job of software librarian for CP/M in, (at the moment) the Micropolis format, I felt it would be of use to other readers of the Sorcerer's Apprentice for me to write a few articles about what software is available in the public domain. As very few of us have the resources to purchase all the software we would like, it becomes increasingly important to know what is available at little cost.

At the present time there are two major CP/M user groups which have placed the majority of software in the public domain. Between them, at the time of this writing, they have 112 disks of programs. Each disk has about 200-250K of software and between them this comes out to about 20 megabytes of software. In reality, for us as Sorcerer users this is not quite as great as it sounds. Much of the software is made up of modifications and updates of earlier releases of the same program. Many of the programs are hardware dependent and so of little direct use to us. However, a significant amount is useful directly, some need to be modified to be of direct use and still others are useful as a tutorial on software development. All in all,the CP/M and SIG/M user group disks are a worthwhile investment and a great deal of fun to dig into.

CP/MUG, the older of the two, began the flood of public domain software. Their disks date from 1977 and run through the most recent releases only a few months ago. At this point they have 54 disks released and, I imagine have more in the works. This group of disks is truly a historic overview of public domain software for microcomputing. They have released the source code of Tiny Basic and an early version of Tarbell Basic. They have released the source for STOIC and ALGOL-M. If you would like to learn assembly language development and the writing of interpreters and compilers this is the place to look. It is rich!

SIG/MUG is the new kid on the block. It has been in existance for a little over a year now and has released 58 disks ofsoftware. It grew out of the ACGNJ (Amateur Computer Group of New Jersey) and has taken off like a shot. At this point I feel that this is the most dynamic of the two groups. The software they have released has been everything from the original Adventure in both source and *COM files, to the RIBBS software for a complete bulletin board system. They recently released the complete Musicraft System in three disks. This had been sold on the open market until recently for something on the order of $100.00. In addition they have released an 8 disk set of the Yale Catalog of Bright Stars, (For astronomers this is a fantastic treasure), a complete hard disk BIOS, a program to UNprotect MBASIC protected programs (the P option) and several disks of Pascal-Z programs.

I have only touched the tip of the programs available. There is so much here that I will discuss the two user group's disks in future issues of the Apprentice. If you would like a full summary of the programs available on the disks I will be glad to provide you (at a cost outlined below) with a disk containing the catalog of both user groups. I will also provide anyone with copies of the user group disks as I have a complete set.

The cost to you is as follows:

continued on page 67

REVIEW OF CADAS 2.00

by Emiliano DeLaurentiis

What is a database system? It is not explained in the documentation for **CADAS 2.00** so before you buy it, let me explain what you may be missing.

In its simplest form a database system simply allows the storage and retrieval of information. In a relational database system such as CADAS, the information is stored as a series of rows and columns. Each row is called a record, and each record is composed of a number of fields. A record would be, for example, a customer in a list of customers. The fields in your record of a customer would be specific data on this customer, such as the customer's name, address and telephone number. The following diagram represents just such a database.

```
                          FIELDS

        RECORDS      Name    Address    Telephone

                     --------------------------
        Customer #1  V   V          V          V
                     V   V          V          V
        Customer #2  V   V          V          V
                     V   V          V          V
        Customer #3  V   V          V          V
                     V   V          V          V
        Customer #4  V   V          V          V
                     --------------------------
```

Besides allowing the storage of information most data base systems will allow the manipulation of this data in one fashion or another. For example, the system may allow the sorting of records on the basis of one of the fields. It may also allow you to select and search for some of te data on the basis of some fields. For example, you may sort according to name, or according to postal code. This kind of manipulation is of the simplest type.

More sophisticated data base systems will also allow you to write programs that use the same data base. For example, one could set up a data base of customer accounts. An accounts receivable program, written in the data base system's language, would allow you to update each customer's accounts to reflect when they have paid you. Similarly, a mailing label program, which would also be written in the database's language could select only the customer addresses from the same data base and print out mailing labels, ignoring the accounting information. Without going into any more detail, it is clear that the level of sophistication of the system will influence its flexibility.

Going back to **CADAS 2.00**, it represents the second system described, without the benefit of a programming language, or the ability to use the system from other languages such as BASIC. It allows you to store ad retrieve data from a cassette or disk system, to print it, to add or delete data, to sort it, and produce totals of certain fields (assuming values in that field are numeric).

CADAS operates with the file of records entirely from RAM memory. Therefore if RAM is exceeded, then no more data can be added to the data base. As long as one does not exceed the memory limitations, then the system itself restricts the user to 750 records, and up to 9 fields per record. Each field can be no longer than 56 characters.

Some of the very positive aspects of CADAS are that it is very user friendly and it is very simple to learn and use. On the other hand, more work could have been done in improving the scrolling of the screen, and in displaying th data base on the screen or printer. For example, a better database system would display the information as a series of rows and columns (since it is conceptually stored in that fashion) and it would allow screen editing of the information.

For comparism purposes, I would like to describe General Business System (GBS) from Quality Software. It is a database system which has a programming language to program specific applications. It is not limited to RAM memory. In fact each file can contain 1 million bytes. In this system one may have 65,536 records and 48 fields per record. It is also very user friendly and extremely flexible.

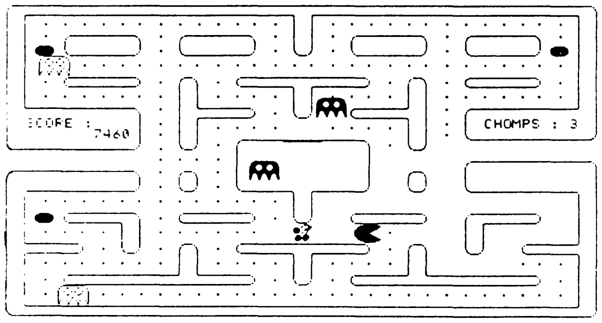
This comparism is not meat to degrade or sensationalize either system, but rather to place the systems in proper perspective. A system such as CADAS, at a cost of $29.95, can hardly be a loss. On the other hand, if your data base will easily exceed 39k of RAM (in a 48k system in cassette mode), or 34k of RAM (in disk mode), then CADAS may never be useful to you. A system as powerful as GBS costs $700.00, but will allow you to use it for a myriad of applications. It is a new programming language in itself. The costs and benefits of either system must be weighed by your requirements, your budget, and your commitment to learn a more complicated and sophisticated database system.

ENSIGN SOFTWARE

FREE CATALOGUE

CHOMP $19.95

is an absolute must for every game enthusiast. It is better then the Pac-Man arcade game where you are being chased through a maze of alleys by four Monsters who will eat you if they can catch you.



DISASSEMBLER $17.95

is a Z-80 machine language two-pass disassembler whose output format is directly compatible with the Development Pac. The Z-80 assembly language source (input to assembler) listing can be sent to Video, Cassette or Printer. The cassette file produced is a source file for the Editor/Assembler and can be read directly into the editor of the Development Pac. The disassembler has a displacement function which allows any program residing anywhere in memory to be decoded, whether it is at its normal address or has been moved to be decoded.

PRODUCT	PRICE
OVERSEAS POSTAGE: ADD $3 NO POSTAGE CHARGE IN USA & CANADA	
VISA MasterCard TOTAL	

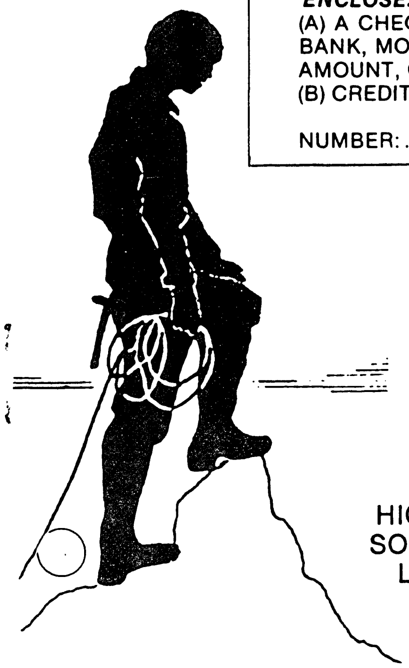
ENCLOSE:
(A) A CHECK IN U.S. DOLLARS DRAWN ON A U.S. BANK, MONEY ORDER OR CASH FOR THE ABOVE AMOUNT, OR
(B) CREDIT CARD, EXPIRATION DATE__________

NUMBER:....................................

MAIL YOUR ORDER TO:

ENSIGN SOFTWARE
2312 N. COLE RD., SUITE E
BOISE, IDAHO 83704 U.S.A.

PHONE: (208) 378-8086

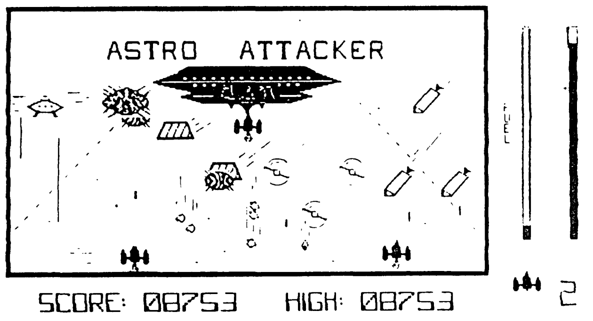


HIGH QUALITY SOFWARE AT A LOW PRICE

THE NEW ARRINGTON SOFTWARE SERVICE

ASTRO ATTACKER $21.95

is similar to the arcade game called "ASTRO BLASTER". This action game for the Sorcerer is far superior to all other Sorcerer games because of its high resolution graphics, sound, variety and playability. Astro Attacker graphics are extremely advanced. The display is of the console inside your astro fighter craft. In your console window you see the enemy ships placed against a background of continuously moving stars. Gauges also indicate the amount of fuel remaining and the temperature of your lazer cannons. If you fire too frequently you can overheat the lazers, or if you move recklessly you may run out of fuel. Your challenge is to survive and destroy the Spinners, the Lazer Ships, the Rockets, the Flame Throwers, and the Meteor shower. Docking with the mother ship is crucial to survival as this restores your shield strength and fuel, and cools your lazer cannon. With each succeeding level of play, survival becomes more difficult as the enemy ships attack with greater frequency and quickness. Superb sound too.



DATABASE SYSTEM II $29.95

is a RAM-based general purpose database system for handling alphanumeric data. It is written in Z-80 machine language and is suitable for use in 32K or 48K Sorcerer microcomputers. Files may be stored on cassette or on disk under the CP/M 1.4 or later operating system. This database is useable by both cassette based systems and/or disk based systems.

Commands available in functional groupings are:
A: File Definition: CREATE
B: File Input/Output: LOAD, MERGE, SAVE
C: File Alteration: ADD, DELETE, EDIT, SORT
D: File Listing: LIST, REPORT, TOTAL
E: System Parameters: PRINTER, SPACE, TABSET
F: Program Exit: CPM, MONITOR

Allowance is made for up to 750 records which may consist of 1-9 fields. Each field is given a name by the user when creating the database and this is used as an aid in manipulating the file. The space available for record storage is approximately 9K less than the available RAM in cassette mode and 14K less than CP/M system size in disk mode. A field may contain up to 56 characters. It is best to divide the record into small fields which are useful for sorting and searching. For example, a file of names and addresses, the fields could be:

NAME, STREET, CITY, and PHONE.

The software is sent on cassette tape, but is easily transfered to your CP/M disk. Use the Monitor >LO command, boot your disk, and then type A> SAVE 27 DATABASE.COM.

```
=========================================================

  THE HAGAN SPREADSHEET  Copyright c 1981 by Roger Hagan
 Associates, 1019 Belmont Pl. E., Seattle, WA  98102  USA
         continued from page 44, previous issue)

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


*** ------ Save arrays subroutine.  First 3 bytes = ramwriter
974 DATA 2,3,201,213,229,245,197,253,54,63,105,253,54,64,0,237,
    91,185,1,1,154
978 DATA 191,205,232,225,3,237,91,187,1,205,232,225,253,54,63,2
    7,253,54,64,224,193,241,225
***
*** ------ Load arrays subroutine starts with 3rd byte below --
980 DATA 209,201,213,229,245,197,253,54,63,105,253,54,64,0,237,
    91,185,1,1,150
983 DATA 191,205,232,225,253,54,63,27,253,54,64,224,193,241,
    225,209,201
***
*** -----------------Cassette Motor #1 on -----------------
985 DATA 6,1,195,36,224
***
***  --------------- Sheet parameter queries ------------------
***
995 IF P=0 THEN PRINT CHR$(12)
1000 PRINT:INPUT "HOW MANY COLUMNS";COLS
1010 PRINT:INPUT "HOW MANY ROWS";ROWS
1015 PRINT:INPUT "Press 'T' to load data from tape";Q$
1017 IF Q$="T" THEN 12500
1020 PRINT:PRINT"NAME THE COLUMNS..."
1030 FOR I= 1 TO COLS
1040 PRINT "   "I" - ";
1050 INPUT CN$(I)
1052 IFLEN(CN$(I))<3 THEN CN$(I)=CN$(I)+"   "
1055 CN$(I)=STR$(I)+")"+CN$(I)
1060 NEXT
1065 GOSUB 1070:GOTO 2015
1070 PRINT:PRINT"NAME THE ROWS..."
1080 FOR I= 1 TO ROWS
1090 PRINT I":";
1095 HL$=RN$(I)
1097 RN$(I)=""
2000 INPUT RN$(I)
2005 IF RN$(I)="" THEN RN$(I)=HL$:GOTO 2010
2007 IF RN$(I)="\" THEN RN$(I)=HL$:GOTO 2013
2009 RN$(I)=STR$(I)+")"+RN$(I)
2010 NEXT
2013 RETURN
2015 FG=0
2020 PRINT:PRINT"Completed.  Here is the upper part of the sheet."
***
***  ---------------"M" Write the sheet with move window option
***
2030 SC=1
2031 SR=1:CC=20:C=1:R=1:GOSUB 10000
2035 GOSUB 10200
2036 CH=2+J-1
***  --------------- Input data where cursor is ---------------
***
2100 PR=R:FOR R=PR TO ROWS
2105 IF R-SR>SZ THEN 2160
2106 IN$=""
2107 OUT CC,CH
2110 INPUT IN$
2115 GOSUB 100
2130 CH=CH+J
2140 NEXT
2145 GOSUB 600    ***  Next column
2150 GOSUB 10000
2155 GOTO 2100
***  --------------- Input to lower sector --------------------
***
2160 MR=R+SZ-1:SR=R:CH=2+J-1
2165 IF MR>ROWS THEN SR=ROWS-SZ:CH=2+J-1+(R-SR)*J
2170 GOSUB 10000
2180 GOTO 2100
***
***  ---------------"H" Draw Help menu ------------------------
***
***  Actual border characters are, clockwise from top left,
***  Graphic Y,C,U,X,H,V,G,Z.  Some print no substitute here.
***
2895 F2=0:IF CC>39 THEN F2=CC:CC=0
2900 OUT CC+10,CR-1:PRINT "))))))))))))))))))))))"
2910 OUT CC+10,CR  :PRINT "' USE CAPITAL LETTER ("
2920 OUT CC+10,CR+1:PRINT "'Move to new sector  ("
2930 OUT CC+10,CR+2:PRINT "'Back up the column  ("
2940 OUT CC+10,CR+3:PRINT "'Jump to amortization("
2950 OUT CC+10,CR+4:PRINT "'Quarter sector toggl("
```

continued on page 67

MISCELLANEOUS NIBBLES

by Don Gottwald

Sorry it took so long to put this issue together. It seems that Ralph and I had more than our share of equipment failures and other priorities, which unfortunately didn't allow us to be on time. We hope to be back on track by issue #5.

We now have two new programs; one written expressly for the Sorcerer, and the other a program in S-BASIC (compiled) for CP/M systems. The first is a Sorcerer Telecommunications System (STS) program written by Jonathan Burnette (Bits & Bytes) and is available on cassette for $30.00. It is beyond the scope of this article to list but a few of the features of STS. It is written entirely in Z-80 assembly language and requires no ROM PACs. STS will run on any size Sorcerer. Upon initial loading of STS and whenever option '1' is selected from the MASTER MENU, the COMMUNICATIONS PA RAMETER MENU is displayed. Several options have been preselected (shown in inverse graphic characters), for the convenience of working with remotely operated CP/M systems (RCPM's). All options may be changed to suit the users needs. Here is a brief description of the MASTER MENU:

1. SET PARAMETERS - (Sets communications parameters such as FULL or HALF DUPLEX, 300 or 1200 BAUD, 1 or 2 STOP BITS, 7 or 8 BIT WORD, EVEN, ODD or NO PARITY, AUTO LINEFEED ON or OFF, ECHO ON or OFF, DISPLAY CONTROL CODES ON or OFF, DISPLAY ERRORS ON or OFF. Any parameter can be permanently or temporarily changed).
2. TERMINAL MODE
3. SAVE TO TAPE
4. LOAD FROM TAPE
5. BLOCK FILE TRANSFER
6. ASCII FILE TRANSFER
7. DISPLAY DIRECTORY

Each of the selections on the menu are fully documented (12 pages of hardcopy) and very easy to read and follow. It is now very easy to access any RCPM and either upload or download to the systems. This program has many more features than you'll find on any other Smart Terminal program presently available.

The second program is called PAPS, which stands for Personal Accounts Payable System and was written by Thomas Bassett (SA Advertising Manager) in S-BASIC. S-BASIC (see review elsewhere in this issue) is a compiled BASIC. You do not need anything other than the program and a CP/M system to run this extremely useful program. This is not just an accounts payable program, but a full fledged personal accounting system, with enough capability to run a small professional business. It comes on three diskettes (Micropolis Mod II format only at this time) and costs just $50.00. We compared it to a Radio Shack program costing twice as much and found this program to be much more professional and easy to use. Several people already have this program and now won't do without it. We hope to have a review for you in a future issue.

****

4TH TIP

by Tim Huang, FORTH Editor

The Screen Editor - Part 4

Continuing the theme of the last issue, we are going to continue, in more detail, to build our Screen Editor. This column will deal with the EDITing portion.

Most Editors, including fig-FORTH's line editor and Quality Software's tape version, only cover this aspect (edit functions) in their Screen Editor. They do not include anything discussed in the last column (i.e. Command functions). Following the algorithm laid out in this series of columns, I don't see any reason why the Editor can't be enlarged into a full blown Word Processor!

```
: EDITING
      INSTRUCTION
         BEGIN
           GET
           LIST.SCR
           BEGIN
             KEY
             CASE
               xx   OF   CURSOR
ENDOF
               yy   OF   ESC
ENDOF
               zz   OF   QUITTING
ENDOF
              BELL
            UNTIL
          ?CONTINUE
          UNTIL ;
```

The EDITING needs to start from INSTRUCTION so that even the novice can understand what key to use. The program then goes into a double BEGIN-UNTIL loop. The outer one takes care of getting the right screen displayed and checking to see if the user wants to continue editing other screens. The inner loop is the one which does the editing of a particular screen. This part should contain several important functions:

1. Sensing the keyboard.
2. If one of the function keys is pressed, performing the corresponding instruction, such as positioning the cursor, and deleting or inserting a character.
3. If a non-functional key is pressed, then updating the video and disk RAM buffers to accept the text character. An important thing to remember here is to map the video display into the disk buffer RAM so that it always contains updated results.
4. In order to keep this inner loop simple, using the ESC key to indicate all two key strokes commands. That is, once the ESC key is pressed, the program will take action in accordance with the next key pressed. With this arrangement, (which in fact is a CASE nested in another CASE), we can have the ESC CASE doing quite a lot of dirty work. I think the following would be nice to have included in ESC CASE: Delete a character; delete a word; erase a line; insert a character; insert a word; insert a line, foreward +/- n lines from the cursor; jump a word (both directions), clear the screen; erase the cursor line; erase +/- n lines from cursor line; etc.

There you have the whole outlined precursor of a Word Processor. As I said at the beginning, this is just the framework to which you can add whatever you think is necessary for your convenience. There certainly are many functions that come in very handy. The following are some examples:

1. Set up certain switches so that the Editor can accept control-characters (for macro and/or special printer controls) in the text, but display them in reverse video.
2. Set up certain switches so that a screen can be loaded selectively, i.e. parts of a screen word behave differently depending on the status of the switches during compilation, such as:

   ```
   :(H;IMMEDIATE
   :);IMMEDIATE
   and :(H 41 WORD;IMMEDIATE
   ```

   These two should exist in an early screen. Then, in a later screen we can use the '(H' and ')' to carry out hardware simulations such as: : TEST ... ... (H GOOD) ... ... ;. If the second definition of '(H' gets used, the compiler ignores the word 'GOOD', but if the other definition is chosen, then the '(H' acts like a do-nothing word and 'GOOD' becomes an active word.
3. Write a set of documenting screens, so that you can document your FORTH programs.
4. Include a Command mode function to allow listing screens (3 to a page) continuously and without printed line numbers.
5. Write a set of useful printing routines.
6. Support Multiple column listings.

***   ***   ***

The "AND SO FORTH" Package

I would now like to announce the release of my new FORTH package, **And So FORTH.** This is a Z-80 fig-FORTH version customized for the Sorcerer I or II. From here on I'll mainly refer to this version. This does not, however, exclude me from discussing other related topics if it appears within my reach.

The specifications for this package are as follows:

1. It is a customized fig-FORTH for the Z80 CPU, particularly for the Sorcerer I or II. Should the upcoming Sorcerer III be using a Z80B, then this package will be modified so that it will work with the new machine. All the fig's standard words are included. Most of the 79-Standard words are also included. However, this is not, repeat **not**, a 79-Standard version. (There is still some disagreement about the definition of some 79-standard words. This package will be updated when it is appropriate to do so.) Many basic Sorcerer related I/O words are also included.
2. It runs under either Mentzer's CP/M 2.2 or Lifeboat CP/M version 1.4. The Exidy version is under development. Vista or other formats will be provided if demand warrants it.
3. It requires the following hardware configuration:
   Sorcerer I or II with at least 32K of RAM
   Micropolis Mod II drive (2 preferred, 1 ok)
   (Soft sector format will be available shortly)
4. The whole package includes:
   A: FORTH disk:
      FORTHnnn.COM - Command files for different memory sizes.
      BnnFORTH.COM - Expanded (Big) FORTH command files for different memory sizes. (nn = 32 or 48)
   B: Screen Disk I:
      Our first Screen disk comes with the package and includes more than 200 screens of various program source code such as the Screen Editor, 8080 Assembler, TRACEr, DIS-(forth), data base, music routines, utilities, etc. Other Screen disks are under development and will be release in near the future. As a matter of fact, Screen Disk II, which is about 70% done, will include: CP/M file interface, routines for the Digitalker board (speech synthesizer), and other useful programs.
   C: And So FORTH:
      This is not merely a skimpy user's manual directing the user to hit this and that key. It is a very comprehensive book describing in detail the internal parts of fig-FORTH. It contains 20 chapters, 5 appendices, and takes up more than 300 pages of 8 X 11 with the 12 character/inch printing. It is intended for users to learn and understand why and how to use FORTH.
5. This package costs $120 (US dollars). The book alone costs $25. This includes postage, handling, and current update notes for 1 year). You can purchase this package from the following places: Timothy Huang, 9529 N.E. Gertz Circle, Portland, OR 97211 - tel. (503) 289-9135; Roger Hagan, 109 Belmont Pl., Seattle, WA 98102 - (206) 394-5034; Arrington Software Service, 9522 Linstock, Boise, ID 83704 - tel. (208) 377-1938.

Since several FORTH books have recently become available, I would like to spend some time updating the reading material listed in my first column. The first book I would like to discuss is entitled "Starting FORTH" by Leo Brodie of FORTH, Inc. ($16 soft, $20 hard cover). This is an excellent book for the beginner who wants to start cold turkey. I(ll vouch for that also. -RL) It was well written, authoritative, and with a good sense of humor. Note, however, that some of its poly-FORTH words are different from fig-FORTH's. For example, the word ' (tick) in fig is an IMMEDIATE word but not in poly. I am in the process of noting the differences between two versions and will include these in a future column.

The second book, "And So FORTH", assumes the reader knows the difference between ROM, RAM, Monitor, video, etc., particularly as it relates to the Sorcerer. The book is divided into four parts. Part I deals with the FORTH philosophy, history, and programming tools. This is the **know-why** section.

Part II gives detailed information about fig-FORTH. This is the **know-how** section. Part III is the Sorcerer section. It details how to boot up and use this package with **your** machine. Part IV includes an improved line oriented editor, definitions of each word of the basic FORTH (i.e., FORTHnnn.COM), in Ascii order and functional classifications. It also includes important information about Screen disk I and additional references.

With FORTH, it seems preferable to me to understand the "why" before the "how". This book, therefore, was organized to reflect this philosophy.

The last book I will consider, "Invitation to FORTH", I would **not** recommend. Not only is the printing lousy, but so is the contents.

For most of us, reading books may be the only way to learn FORTH. It may not be the most effective way, however. To those who prefer to learn it in a more academic environment, please contact me for further details. I have taught FORTH twice to beginners with good results. After taking my comprehensive 20 hours crash course, you should feel fairly comfortable with the basic philosophy of FORTH as well as programming in FORTH.

Until next time, may FORTH be with you.

SPEECH SYNTHESIS

by Eric Zorawicz

Wouldn't you like to have the Sorcerer talk to you while playing a game or running your favorite program? If that sounds interesting, you might try the Sweet Talker Phonetic Speech Synthesizer as I have. The Sweet Talker uses the VOTRAX SC-01 speech synthesizer chip and is sold by Micromint, Inc. for $139.00 A/T. They also offer a power supply for $35.00 which has all the needed voltages. With a speaker and a cable you are ready to interface it to the Sorcerer's parallel port as follows:

SORCERER PIN #		SWEET TALKER PIN
1	Ground	12,D
4	Strobe	21
5	Inflection	16,20
6	Output Bit 5	19
7	Output Bit 4	15
19	Output Bit 3	18
18	Output Bit 2	14
17	Output Bit 1	17
16	Output Bit 0	13
25	Busy	8

Since the Sorcerer handles the Strobe with software using output data bit 7, you may want to add a pulse extender on the Strobe line as shown in Reference 1. However, this problem can be dealt with by writing a little extra software. The Sweet Talker can say just about anything since it speaks in phonemes. Let's suppose you want to say 'abort'. Here is the BASIC program with the subroutine needed to output to the Sweet Talker:

```
10 V$="2N44+*?"
20 GOSUB 1000
```

SUBROUTINE:

```
1000 REM ** VOUT SUBROUTINE **
1010 FOR I=1 TO LEN(V$)
1020 B$=MID$(V$,I,1)
1030 J=ASC(B$) AND 63
1040 J80=J OR 128
1050 K=INP(255) AND 128
1060 IF K<>0 THEN 1050
1070 OUT 255,J
1080 OUT 255,J80
1090 NEXT I
1100 RETURN
```

Along with the Sweet Talker comes a very handy dictionary for quick access to phoneme sequences for approximately 1400 words. Words not in the dictionary do take some trial and error to get them to sound right. One nice feature is that two different voices can be used by changing the inflection bit which is output data bit 6 and can easily be programmed. The Sweet Talker actually has four voices but the Sorcerer only has one bit available to use for inflection.

The Sweet Talker is fun and gives your computer a little more character and personality. I'm sure you'll come up with many ideas once you start using it.

REFERENCES

1. Ciarcia, Steve. "Build a Unlimited-Vocabulary Speech Synthesizer", BYTE, September 1981, page 38.

2. Ciarcia, Steve. "Build a Computerized Weather Station", BYTE, February 1982, page 38.

RESTORING LOST LINK ADDRESSES

byJames R. Canning, 64 Walnut St., E.Providence, RI 02914)

Here is a copy of an assembly program I wrote to back up the 'TOSCA' (Tape Output SCAnner) program in the 'May' issue of the combined SA-S.U.N. The latter program allowed me to salvage a large and long-lost BASIC program I had "saved" on cheap tape before I learned better. Having put the tape away in the hope that something like this was possible, I used the 'TOSCA' program to get back what I could, as soon as I received the issue with this program.

Restoring the link addresses, which went off right at the start proved to be so laborious and accident-prone that I wrote this 'FXLNX' program. This program has the much less demanding requirement that only the end-of-lines nulls, and the three nulls marking end-of-file, be in their proper place - or at least at one's best guess. Execution of FXLNX will then recompute all the link addresses based on the nulls and stash the EOF address in the proper place so that a BASIC listing can be produced immediately, and further cleanup begun.

```
ADDR     OBJECT     ST
                    0001         ORG   0H
'0000    0E00       0002 FXLNX   LD    C,0
'0002    DD21D301   0003         LD    IX,01D3H
'0006    21D501     0004         LD    HL,01D5H
'0009    DD23       0005 LOOP    INC   IX
'000B    0C         0006         INC   C
'000C    DD7E00     0007         LD    A,(IX+0)
'000F    FE00       0008         CP    0
'0011    20F6       0009         JR    NZ,LOOP-$
'0013    CD2900'    0010         CALL  EOFCK
'0016    79         0011         LD    A,C
'0017    FE06       0012         CP    6
'0019    38EE       0013         JR    C,LOOP-$
'001B    DDE5       0014         PUSH  IX
'001D    D1         0015         POP   DE
'001E    13         0016         INC   DE
'001F    D5         0017         PUSH  DE
'0020    73         0018         LD    (HL),E
'0021    23         0019         INC   HL
'0022    72         0020         LD    (HL),D
'0023    E1         0021         POP   HL
'0024    0E00       0022         LD    C,0
'0026    C30900'    0023         JP    LOOP
'0029    DDB601     0024 EOFCK   OR    (IX+1)
'002C    DDB602     0025         OR    (IX+2)
'002F    C0         0026         RET   NZ
'0030    DDE5       0027         PUSH  IX
'0032    D1         0028         POP   DE
'0033    73         0029         LD    (HL),E
'0034    23         0030         INC   HL
'0035    72         0031         LD    (HL),D
'0036    DD23       0032         INC   IX
'0038    DD23       0033         INC   IX
'003A    DD23       0034         INC   IX
'003C    DD22B701   0035         LD    (01B7H),IX
'0040    DD22B901   0036         LD    (01B9H),IX
'0044    DD22BB01   0037         LD    (01BBH),IX
'0048    E1         0038         POP   HL
'0049    C9         0039         RET

ERRORS=0000
EOFCK               0029 FXLNX       0000 LOOP   0009
```

This program is used after a read-all tape load via the TOSCA Tape Output SCAnner program of Mr. J. Burns, San Francisco, CA. in the 'May' (first combined) issue of SA/S.U.N. The end-of-line nulls and three nulls marking end-of-file must be checked out and restored as best possible (DON'T forget the latter!) then FXLNX can be executed and a BASIC listing done immediately. IX points thru the BASIC file and HL holds the address in which to stash (IX) when i is found to be a new link address. BASIC lines of the type nnn : are accepted via the limiting value of 6 in the counter C; anything less is rejected as litter.

JOYSTICK INTERFACES FOR THE SORCERER

by Walt Hendrickson, 2313 W. 181st St., Torrance, CA 90504)

I noticed in the OCT-NOV 1980 issue of SORCERER'S APPRENTICE an invitation for articles on joystick interfaces. I have interfaced two (2) different joystick units to my Sorcerer. Both of these are the 'switch' type (i.e. units that consist of four microswitches triggered by a control handle to give 8 distinct directional outputs).

The two joysticks used are: 1) the Interact Entertainment Controller; and 2) the Atari Game Controller. These units are available from the sources listed below. To connect either of these units to the Sorcerer's parallel input port requires only a short cable adapter. The cable schematic for connecting one Interact controller to the Sorcerer is shown in Fig. 1. To connect two such units, use the schematic shown in Fig. 2. The connectors (DB-25p and DE-9p) can be obtained at most computer stores. The Atari unit uses the same connector, but a different pinout.

Shown in listing 1 is a short program for reading 1 joystick. Because there is only one 8 bit input port on the Sorcerer, two joysticks with 'fire' buttons must also use the handshaking lines on both the input and output ports to obtain the needed 10 bits. I have adapted several 'real time' games to joystick control, and find the enjoyment is greatly enhanced by the 'feel' of the joystick vs. keyboard control.

LISTING 1: One controller + fire button

This routine will read one joystick and return with the Zero flag set if there is no joystick command, or with the directional data in register A if a command is present. The directional data is as follows:

Data in A	Command
80H	Fire button is pushed
80-8AH	Fire button is pushed with direction also
00	no commands
01	Left (west) command
02	Right (east) command
04	Up (north) command
08H	Down (south) command
06H	Northeast command
0AH	Southeast command
09H	Southwest command
05H	Northwest command

```
------------------------------------------
0000 DB FF   JOY: IN 0FFH INPUT PORT DATA
0002 2F           CMA     REVERSE PATTERN
0003 B7           ORA A   SET FLAGS
0004 C8           RZ      NO CMD RETURN WITH A=00H
0005 E6 8F   ANI  8FH     UNUSED BITS
0007 C9           RET     RETURN TO CALLER
```

LISTING 2: Two controllers plus fire buttons

This routine reads the 10 data bits for two joysticks and exits with 'fire' button data in the high 2 bits of register B and the direction data in register A.

Data in B	Command
80H	'fire' button 1 pressed
40H	'fire' button 2 pressed
C0H	both buttons pressed

Data in A Command

This is the same as for 1 controller, with controller #2 on the high nibble and controller #1 on the low nibble.

```
------------------------------------------
0000 DB FE  JOY2: IN 0FEH     INPUT 'FIRE' BUTTONS
0002 E6 C0        ANI 0C0H    MASK UNUSED BITS
0004 47           LD B,A      'FIRE' DATA TO B
0005 D3 FF        OUT 0FFH    CLEAR OUTPUT FLAG
0007 DB FF        IN 0FFH     INPUT DIRECTION
0009 2F           CMA         FIX DATA
000A C9           RET         RETURN
------------------------------------------
```

INTERACT CONTROLLER	ATARI CONTROLLER
from: Micro Video Corp.	from: Compumart
P.O. Box 7357	P.O. Box 568
204 E. Washington	270 Third St.
Ann Arbor, MI 48107	Cambridge, MA 02139
Price:$17.95 ea	Price:$19.95/pair
$34.95 pr.	

REVIEW OF SUPER DISASSEMBLER

by Emiliano DeLaurentiis

Super Disassembler by the Global Software Network is a very useful and efficient disassembler. It will decode into Z80 mnemonics any section of your computer's memory. It will disassemble code and display it on the video, or print it on your serial or parallel printer, or even output the source to a cassette.

I must confess that I do not program much in assembly language, so I cannot attest to the professionalism of the program. My preference for this disassembler over others that I have seen is the speed that it operates at, and most importantly, the fact that it creates source code which is fully compatible with the Development PAC. I can now use this disassembler as a learning tool. I can decode existing routines, and then by using the Development PAC, make systematic changes to the routines to explore its effect. Being a lazy person, I prefer not having to key in streams of source code if I can avoid it.

This disassembler has a few other features which I have not yet used. The are: **dump** memory (which is similar to the monitor routine), **exclude** certain memory locations from being decoded, **add** a displacement to the decoded output, **exit**, and **limit** the label field size when outputting the source code.

I am certain that you will discover this assembler to be a very useful product.

****

LOADING ML & BASIC PROGRAMS TOGETHER

by Jack MacGrath

If you are sick of POKEing a ML program through BASIC, try this:

1. Find the end of the BASIC in bytes 01B7-8
2. Choose a start address 16 bytes down from 01B7-8
3. With MO command, move the ML code behind the address you chose in #2.
4. 16 bytes in front of the ML code, insert following program:

```
01 xx yy LD BC   xx yy   ; Length of ML program
11 aa bb LD DE   bb aa   ; Destination address
21 ee cc LD HL   cc ee   ; Address from #2
ED B0    LD IR           ; Move ML up in memory
C3 6B C0 JP C06B         ; Warm start in BASIC
```

5. In address 100-102, goes a jump to the loader:

```
C3 zz kk    JP kkzz      ; Address of boot
```

6. SA from 0100 to the end of the ML code.
7. >LOG automatically loads BASIC and ML, and put the ML in high memory.

Another way to save ML and BASIC if they are close to each other in memory, is to load all the programs you wish to have in memory, then SA from start to end. All the programs will load together with the CLOAD command and run. As long as you call the routines through their properly POKEd addresses in Basic, this will work fine.

****

ASTRONOMY PROGRAMS

JSATS- Displays configurations of Jupiter's satellites for any date and time or series of dates and times, N or S at top. **$10**

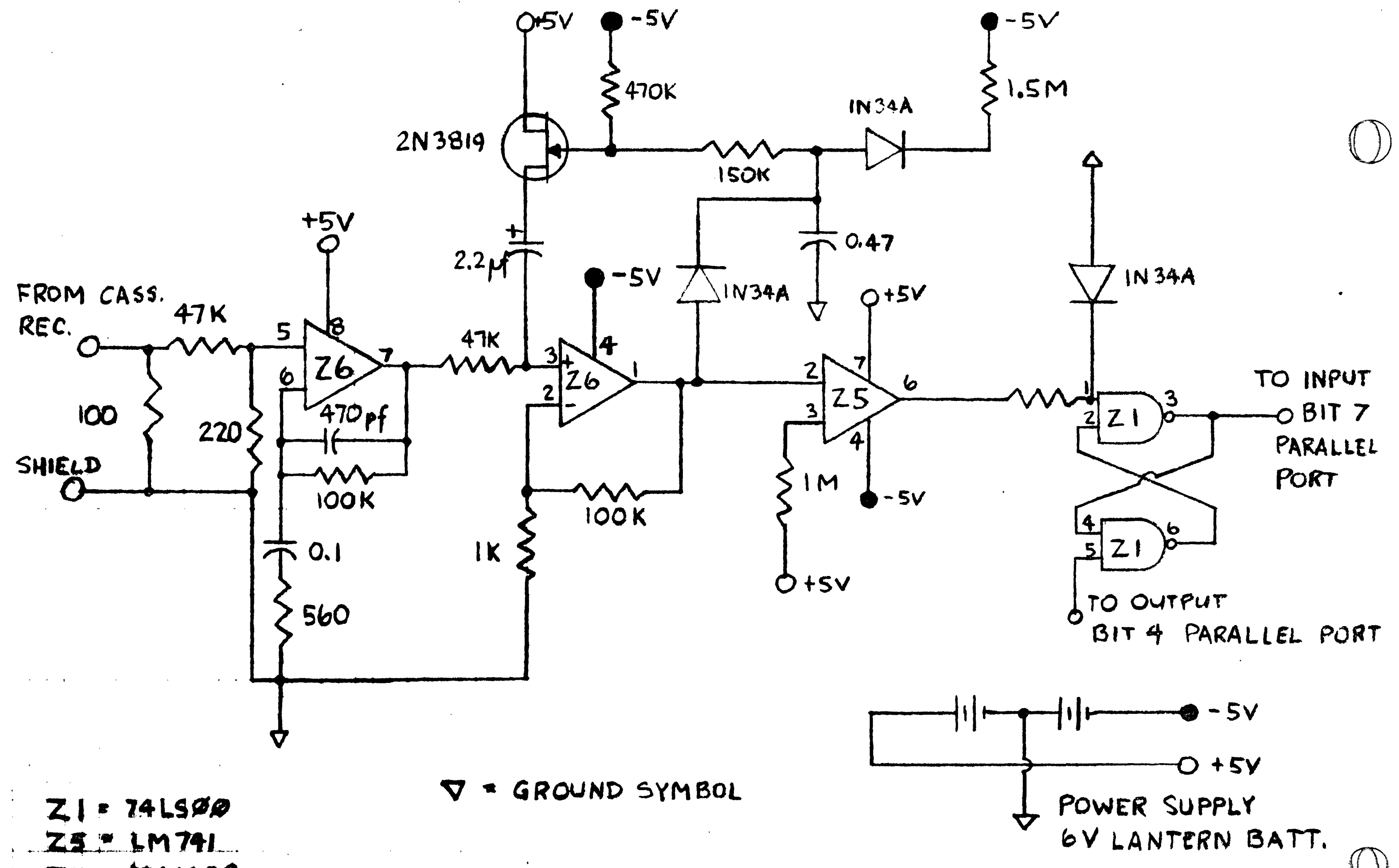
ECLIP- Gives date and magnitude of next umbral eclipse of Moon, starting any year and continuing for as long as requested. **$5**

PLTTN- Ask for any planet or Sun on any date and program selects and displays a star map and plots planet plus any others and Moon if in same region. With or without RA and Dec grid, and plots a series for selected time intervals. Identifies stars. Indicates phase of Moon. **$20**

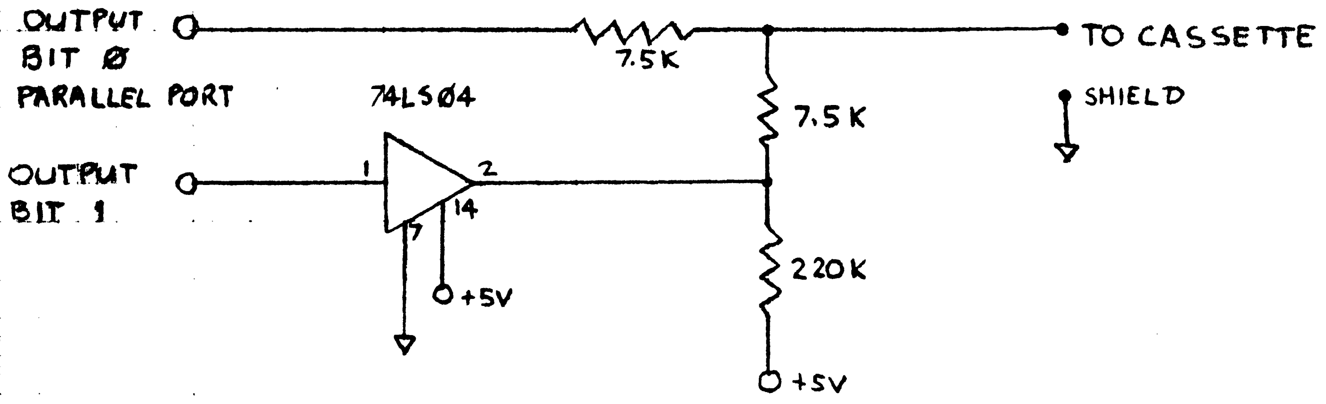
RADEC- Gives RA and DEC for planets, any date. **$15**
RADCM for Moon. **$10**
SKYPN- Plots stars, planets, Sun, and Moon visible above horizon at any time and date in Northern or Southern Hemisphere to 85 deg. lat. **$25**

BOOK of listings of **20 astronomical programs** with photos of screen displays **$25** plus $3.50 postage and handling ($7 overseas).

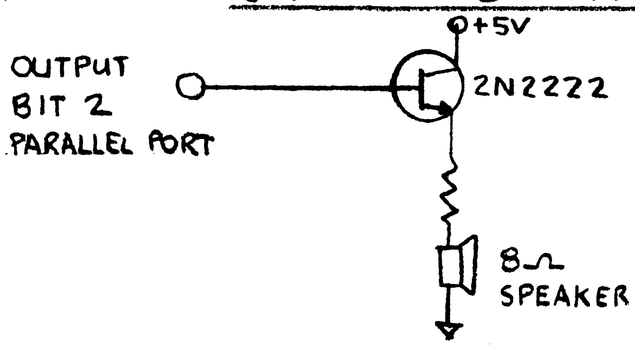
(A self-addressed envelope for details. Available for Astrologers also.)



CASSETTE LOAD CIRCUIT



CASSETTE SAVE CIRCUIT



SOUND CIRCUIT

Spreadsheet continued from page 62

```
2952 OUT CC+10,CR+5:PRINT "'Write page again    ("
2954 OUT CC+10,CR+6:PRINT "'Last column back    ("
2956 OUT CC+10,CR+7:PRINT "'Next column         ("
2958 OUT CC+10,CR+8:PRINT "'Goto a specific slot("
2960 OUT CC+10,CR+9:PRINT "'Relationship def'n  ("
2963 OUT CC+10,CR+10:PRINT"'Set/clear row totals("
2966 OUT CC+10,CR+11:PRINT"'Column input again  ("
2969 OUT CC+10,CR+12:PRINT"'Print toggle on/off ("
2970 OUT CC+10,CR+13:PRINT"'Kalkulate (yes, K)  ("
2971 OUT CC+10,CR+14:PRINT"'Alter row names"to \("
2972 OUT CC+10,CR+15:PRINT"'Xpand display on/off("
2973 OUT CC+10,CR+16:PRINT"'Tape save and load  ("
2975 OUT CC+10,CR+17:PRINT"'Extend value across ("
2976 OUT CC+10,CR+18:PRINT"'Formulas display    ("
2977 OUT CC+10,CR+19:PRINT"'Viz-a-plot          ("
2980 OUT CC+10,CR+20:PRINT"********************"
2982 OUT CC+20,CR+20:INPUT IN$
2983 IF IN$="" THEN 2982
2984 IF F2<>0 THEN CC=F2
2985 GOTO 120
3005 IN$="H":GOTO 114
***
***  ---------------"B" Back up column -----------------------
***
3100 CH=CH-J:R=R-1
3105 IF CH<2 OR R<1 THEN CH=2:R=SR:GOTO 2895
3110 OUT CC,CH
3130 INPUT IN$
3140 GOSUB 100
3145 IF IN$<>"" THEN 2100:REM  Input data
3160 GOTO 3100
***
***  ---------------"W" Write the sheet ----------------------
***
3200 GOSUB 10000
3205 CH=(R-SR+1)*J+1
3210 GOTO 2100: REM  Input data
***
***  -------------- Write the sheet subroutine ---------------
***
10000 PN=20:REM                          Print column names
10005 IF P=0 THEN PRINT CHR$(12)
10010 FOR I=SC TO SC+3
10012 IF I>COLS THEN 10030
10015 PRINT TAB(PN);
10020 PRINT CN$(I);
10025 PN=PN+10
10030 NEXT
10035 PN=20:REM                          Print row names
10038 RB=R
10039 IF J=2 THEN PRINT
10040 PRINT:FOR RA=SR TO SR +SZ
10045 IF RA>ROWS THEN RA=RB:RETURN
10050 PRINT RN$(RA);
10055 FOR CA=SC TO SC+3:REM            Print variables by column
***
10057 IF CA>COLS THEN 10080:REM (\ $ Formatter if printing /)
10058 IF P=1 THEN PRINTTAB(PN+8+FNJ(V(CA,RA)));V(CA,RA);:GOTO 10070
10059 PRINT TAB(PN);
10060 PRINT V(CA,RA);
10070 PN=PN+10
10080 NEXT CA
10090 PRINT
10095 IF J=2 THEN PRINT
10100 PN=20
10110 NEXT RA
10115 RA=RB
10120 RETURN
*** --------------- Print divider and Move choices -----------
10200 PRINT"================================================";
10202 PRINT"=========="
10204 PRINT"To see more of the sheet, you may move Right,";
10206 PRINT "Left, Down, or Up."
10210 PRINT "To Proceed with entries, type 'P'."
10215 INPUT ">>>  R,L,D,U, or P...";Q$
10220 IF Q$="P" THEN RETURN
10230 IF Q$="L" THEN 10500
10240 IF Q$="D" THEN 11000
10250 IF Q$="U" THEN 11500
10260 IF Q$<>"R" THEN 10200
***     All these but "P" loop back to 10215: P does the return
***
***  -------------"R" Move right one (or 1/2) 4 col sector ---
***
10270 IF FG=1 THEN 11600
10272 SC=SC+4:IF SC+3>COLS THEN SC=COLS-3
10275 C=SC:CH=2+J-1:GOSUB 10000:GOTO 10215
***
```

continued in next issue

continued from page 59

If you send the media to me formatted (Micropolis only) $3.00/disk plus $1.50 postage.

If I must purchase the media and format the disks $8.00/disk plus $1.50 postage.

Make checks payable to Bruce Blakeslee and allow 4 to 6 weeks for checks to clear and the mail to deliver.

In either case, $2.00 of the cost of each disk is returned to the Sorcerer's Apprentice to support the newsletter and the bulletin board.

If you would like the user group disks in a format other than Micropolis-CP/M you can order them from the following sources:

8" SS/SD -- SIG/M
P.O. Box 97
Iselin, N.J. 08830

(SIG/M has made a decision to support ONLY 8" SS/SD disks. They will no longer support Micropolis, North Star, etc., because of the cost of multiple libraries.)

All formats are carried by:
Lifeboat Associates
1651 Third Avenue
New York, N.Y. 10028
(CP/MUG disks for sure. I am not sure about the SIG/MUG disks. Check with them on the price.)

*I would hope that if you are going to buy the disks in Micropolis format that you would buy from us and support the newsletter and the bulletin board system.

****

100H = 1024D ! ! ! ! ! !

by Rich Franzen

I have accomplished easily, freely, and quickly the above results with my system, which consists of a 48k Sorcerer II and one Exidy 77 track soft-sectored drive. I also have the 1.1 monitor ROMs (EXMON), which are not necessary for the below idea, but make it slightly more convenient. Basically all I did was move my disk boot from BF00-BFFF to EF00-EFFF.

"But that space belongs to EXMON!" you say to me. I say that you are right, but it is the part of EXMON concerned only with remembering the standard Exidy graphic characters, and only the last half of these characters, to boot! Specifically, addresses EDFE-EFFF define this character set, which usually means that over 200H bytes in the Sorcerer are duplicated, once in EXMON, and second (where it is actually used) in addresses FC00-FDFF of high RAM, used by the Video section of the Sorcerer to define what ASCII characters 128-191 look like; once the information is in this RAM, whatever the top of EXMON says is immaterial--almost. If you are not concerned with these characters, there is no problem. If you do use these standard graphic characters, then following my suggestion will pose a small inconvenience, since this character set must then be loaded from disk (or tape). Also note that only characters "GRAPHIC K" thru "(keypad) GRAPHIC =" are affected.

The first step of my suggestion involves saving the standard graphics on disk. Boot your disk, then give control to EXMON. Using EXMON, do the following:

```
MO FC00 FDFF 110
EN 100
100: 01 00 02
103: 11 00 FC
```

continued on page 68

After your BASIC routine has called X=Inp(0), it can then call Y=Peek(64) and see which shift is being used.

Conclusion:

This subroutine provides the opportunity for writers of real time games to provide for vigorous action. Serious applications also can benefit from the time saved from repeated key presses or use of the repeat key.

Appendix: The routine:

```
ADDR  OBJECT    ST
                0001 ;
                0002 ;Numeric keypad scanning routine
                0003 ;
                0004 ;returns with number in Accumulator
                0005 ;
                0006 ;ALSO if the following keys are
                0007 ;pressed, 40H is loaded as follows:
                0008 ;        none    0
                0009 ;        SHIFT   1
                0010 ;        S/lock  2
                0011 ;        CTRL    3
                0012 ;        Graphic 4
                0013 ;
                0014 ;March 1981
                0015 ;
'0000 0EFE      0016 START  LD    C,0FEH  ;port no
'0002 160D      0017        LD    D,0DH   ;column no
'0004 DD213100' 0018        LD    IX,TABLE
'0008 CD2300'   0019 CONT   CALL  SCAN
'000B 3008      0020        JR    NC,SHIFT-$
                0021 ;            got a key - check shifts
'000D 14        0022        INC   D       ;next column
'000E CB62      0023        BIT   4,D     ;see if 10H
'0010 28F6      0024        JR    Z,CONT-$
                0025 ;here if no key pressed
'0012 3EFF      0026        LD    A,0FFH  ;flag NOKEY
'0014 C9        0027        RET
'0015 1600      0028 SHIFT  LD    D,0     ;shift col.
'0017 DD6600    0029        LD    H,(IX)  ;no. found
'001A CD2300'   0030        CALL  SCAN
'001D 78        0031        LD    A,B
'001E 324000'   0032        LD    (STORE),A
'0021 7C        0033        LD    A,H
'0022 C9        0034        RET
                0035 ;SCAN subroutine.
                0036 ;C=port no:d=column no:IX increased
                0037 ;with each bit tested. E used
                0038 ;returns with B one to four
'0023 ED51      0039 SCAN   OUT   (C),D
'0025 ED58      0040        IN    E,(C)
'0027 0605      0041        LD    B,5          ;counter
'0029 CB3B      0042 LOOP   SRL   E
'002B D0        0043        RET   NC
'002C DD23      0044        INC   IX
'002E 10F9      0045        DJNZ  LOOP-$
'0030 C9        0046        RET
                0047 ;
'0031 00        0048 TABLE  DEFB  0
'0032 01        0049        DEFB  1
'0033 04        0050        DEFB  4
'0034 08        0051        DEFB  8
'0035 07        0052        DEFB  7
'0036 0A        0053        DEFB  0AH
'0037 02        0054        DEFB  2
'0038 05        0055        DEFB  5
'0039 06        0056        DEFB  6
'003A 09        0057        DEFB  9
'003B 0000      0058        DEFW  0
'003D 00        0059        DEFB  0
'003E 0B        0060        DEFB  0BH
'003F 03        0061        DEFB  3
'0040 00        0062 STORE  DEFB  0
                0063 ;
                0064 ;TEST
                0065 ;
 >E1ED          0066 ASCOUT EQU   0E1EDH
 >E205          0067 CRLF   EQU   0E205H
 >E015          0068 QUICK  EQU   0E015H
'0041 CD0000'   0069 TEST   CALL  START
'0044 CDEDE1    0070        CALL  ASCOUT
'0047 CD05E2    0071        CALL  CRLF
'004A CD15E0    0072        CALL  QUICK
'004D 28F2      0073        JR    Z,TEST-$
'004F C9        0074        RET
```

****

continued from page 67

```
106: 21 10 01
109: ED B0
10B: C3 00 00 /
GO 0
```

This will give CP/M control, and you have set up a simple up-load for the graphics. Now simply type "SAVE 3 EXGRAF.COM", which gives you the future ability to load them when necessary by typing "EXGRAF".

"O.K., how do you move the boot program anyway?" TURN OFF YOUR COMPUTER AND DISK DRIVE. I unplugged my drive from both the wall and the Sorcerer and brought it to a well lighted, dry, clean spot. You will need a Phillips screwdriver and the confidence to perform some very light surgery inside the drive.

Disconnect both the electrical cord and ribbon cable from the rear of the drive. On each side of the drive, remove the three rearmost Phillips screws. Lift off the top of the case and slide rearward the back panel (this panel is the power supply, and it is still wired in place, so treat it gently). The disk controller board can now be seen at the bottom of the case, along with two sets of DIP switches. I have no idea what the four pole DIP switch, labelled SW2, does, but that is not the one you have to mess with. You are going to change the eight pole switch, labelled SW1. This switch locates the page of memory which the booting program will over-write. It should be already factory set to all eight poles off except #2, which corresponds to the page starting at BF00. Slide #2 to "off" and #4 to "on". This was the minor surgery--doing this starts the boot program at EF00. Screw the rear panel back in place and then do the same to the top of the case. Note that if your drive is still under warranty, you have just voided that warranty, but I believe this only lasts ninety days anyway. The choice is yours.

I started out by saying that 100H=1024D. This is choice you can now make. You have just regained for yourself the whole 48k of your RAM by freeing this top page (BF00-BFFF). If CP/M does not have control, this is useable by you in any way you want. With Exidy's CP/M you have two options. You know that due to the CACHE BIOS the maximum CP/M configuration you could have previously was 47k, which left four cache buffers between the top of CP/M and the bottom of EXMON's work area. With a 47k CP/M you now have five buffers, which can increase overall speed on disk-intensive tasks. This is your first option. Your second option is to follow my promise by doing the "MOVCPM 48 *" program followed by "SYSGEN". This will give you 1024D additional bytes in the "transient program area". In other words, by freeing the 100H bytes of RAM you have just given your CP/M environment 400H (1024D) more bytes to play with! Of course you would only have one cache buffer, which will slow down disk accessing, but only to the level of most of the CP/M world, which does not ordinarily use the CACHE technique. For most of my purposes, it's 48k!

****

SORCERER'S APPRENTICE
P.O. Box 33
Madison Heights, MI 48071

!!! JOIN NOW !!!

To become a 1982 member of the **Sorcerer's Apprentice** User's Group and receive Vol. IV of the **SORCERER'S APPRENTICE** Newsletter, return this completed application along with payment.

NAME:

Title: Mr. Miss Mrs. Ms Dr. or ________

First name: ________________

Middle initial: ________________

Last name: ________________

Business Name (if a business):

____

ADDRESS:

Number, Street and Apt. No.

____

City: ________________

State or Province: ________________

Zip/Postal code: ________________

Country: ________________

PHONE:

Home: (________)________________

Business: (________)________________

If you do NOT wish the above released, sign here:

____

NETWORKS:

Source ID: ________________

MicroNET ID: ________________

The following information will be held in confidence:

Which of the following do you have?
(circle where applicable):

SORCERER:

Number of units (if more than one): ________

Model: I or II

RAM memory: 8K 16K 32K 48K >48K

EXPANSION:

Exidy S-100 Expansion Unit: Yes No

Other expansion unit: ________________

Exidy expansion cable: Yes No

Cards used in expansion unit:

____

PRINTER:

Type: ________________

DISK SYSTEM:

Type: ________________

CP/M:

Exidy 1.4 2.2

Lifeboat 1.4 Mentzer 2.2

Other DOS: ________________

BASIC: ________________

MODEM:

Type: ________________

PERIPHERALS:

____

PERSONAL:

Age: ________________

Occupation: ________________

How do you rate yourself as a computerist?

>HARDWARE: Beginner Intermediate Expert

>SOFTWARE: Beginner Intermediate Expert

Is your interest:

Hardware Software Both

Is your application:

Business Personal Both

Use a separate sheet of paper, if you don't have enough room to comment below.

PLEASE USE SEPARATE PAPER FOR YOUR QUESTIONS.

EXIDY:

If you have had any dealings with Exidy Systems, describe their nature and outcome:

____

____

FEEDBACK:

What types of software interest you most?

____

____

List the columns or articles you like the most.

____

____

List the columns or articles you like the least.

____

____

What topics/articles would you most like covered?

____

____

What comments have you about the Newsletter?

____

____

BACK ISSUES:

ARESCO Source (issues 1-5)	SOLD OUT
S.U.N. Volume II	SOLD OUT
Sorcerer's Apprentice Vol I (1-7)	SOLD OUT
Sorcerer's Apprentice Vol II (1-5)	SOLD OUT
Sorcerer's Apprentice Vol III (1-8) @ $12:	$________
Sorcerer's Apprentice Vol III @ $2.50 each:	$________
Overseas orders add $1/issue or $4/Vol:	$________

1982 MEMBERSHIP - VOLUME IV:

U.S.A. - Third Class postage @ $18:	$________
U.S.A. - First Class (in an envelope) @ $24:	$________
Canada & Mexico - First Class @ $24:	$________
All others - Airmail @ $32:	$________
Single issue - USA, Canada & Mexico @ $3:	$________
- All others airmail @ $4:	$________
TOTAL	$________

Make checks or money orders (only in US funds drawn on a US bank) payable to: **SORCERER'S APPRENTICE.**

****

TERCENTENNIAL TECHNICAL
Video and Data Communications
Technical Services

****

For quality Service on the following equipment:

EXIDY SORCERER COMPUTERS - EXPANSION INTERFACE & BOX
RADIO SHACK...all models - ZENITH/HEATH - ATARI
LEEDEX, SANYO ELECTROHOME........and other brand monitors
MICROPOLIS, VISTA SHUGART, PERCOM, MPI...and other drives
MODEMS, DOT MATRIX PRINTERS..................most brands
Memory upgrades and new Monitor 1.1 PROM's available.

* *

Services available on most home entertainment electronics:

* *

SOLID STATE TELEVISION foreign and domestic brands.Sony etc.
VIDEO RECORDERS---VIDEO CAMERAS---VIDEO MONITORS
STEREO RECEIVERS---AUDIO RECORDERS---RECORD CHANGERS
VIDEO SWITCHERS---TIME/DATE GENERATORS---VIDEO DISC SYSTEMS
ATARI VIDEO GAMES and others

****

For services or consultations on the above units, call or write:
Jack MacGrath.....**TERCENTENNIAL TECHNICAL**
Tech Lic #8228 P.O. Box 5
70 Tercentennial Drive
Billerica MA 01821
phone after 6:00 pm EDT (617)667-8272

(This is now a part time business, but with YOUR help, I hope to soon make it FULL time and render faster service!)

****

ME mentzer electronics

590 South Hill Boulevard, Daly City, California 94014
(415) 584-3402

****

CP/M Catalog program, good for cataloguing your CP/M disks	$ 75.00
dBASE II Relational Database Management Program	$595.00
Exidy 1.1 Monitor ROMS	$ 45.00
SPELLBINDER Word Processor Now also for the Exidy 77 track soft sectored drives.	$395.00
SPELLCHECK Dictionary program to work with SPELLBINDER	$295.00
CP/M 2.2 For the Exidy with Micropolis hard sector drives only. (CP/M is a trade-mark of Digital Research)	$190.00

We have Godbout Electronics, and Morrow Designs hardware.
Check with us for all your hardware needs.

****

MASTER CARD and VISA on orders of $50.00 or more.
Shipping will be added to all orders.
California Sales Tax added for CA residents

BINARY

by David Cooke, President of SAT TRAK INTERNATIONAL

The following program will read the keyboard and display the ASCII character, the ASCII code and the BINARY equivalent

```
          ;*** BINARY *** - A PROGRAM TO READ KEYBOARD AND DISPLAY
          ;            1. THE ASCII CHARACTER
          ;            2. THE ASCII CODE
          ;            3. THE BINARY EQUIVALENT
          ;
0100            ORG 0100H
          ;
0100  31FF01 BINARY  LXI SP,STACK ;INITIALIZE TO 03FFH
0103  CD05E2 MORE    CALL CRLF
0106  CD05E2         CALL CRLF
0109  CDD1EA WAIT    CALL QUICKCK ;LOOK FOR 'CTL C'
010C  C203E0         JNZ WARM     ;DO WARM BOOT
010F  CD1CEB         CALL GETKEY  ;GET CHR FROM KEYBRD
0112  CA0901         JZ WAIT      ;NO CHR YET
0115  F5             PUSH PSW     ;SAVE 3 COPIES
0116  F5             PUSH PSW
0117  F5             PUSH PSW
0118  FE20           CPI ' '      ;WAS IT 20H OR LESS
011A  FA2801         JM NOSHOW    ;YES, THEN DON'T PRINT IT
011D  CD0CE0         CALL PRINT   ;NO, THEN TO VIDEO
0120  3E20           MVI A,' '
0122  CD0CE0         CALL PRINT   ;NOW PRINT A SPACE
0125  C33001         JMP ASCII    ;AND SKIP NOSHOW
0128  3E20   NOSHOW  MVI A,' '
012A  CD0CE0         CALL PRINT
012D  CD0CE0         CALL PRINT   ;TWO SPACES FOR CTL CHR
0130  F1     ASCII   POP PSW      ;GET CHR BACK
0131  E6F0           ANI 0F0H     ;MASK OFF LOW NIBBLE
0133  0F             RRC
0134  0F             RRC
0135  0F             RRC
0136  0F             RRC          ;SHIFT IT DOWN 4 BITS
0137  CD7201         CALL CONVRT  ;GET ASCII CODE
013A  CD0CE0         CALL PRINT
013D  F1             POP PSW      ;GET ANOTHER COPY BACK
013E  E60F           ANI 0FH      ;MASK OFF HI NIBBLE
0140  CD7201         CALL CONVRT  ;GET ASCII CODE
0143  CD0CE0         CALL PRINT
0146  3E20           MVI A,' '
0148  CD0CE0         CALL PRINT   ;LEAVE ANOTHER SPACE
014B  1608           MVI D,8      ;TO COUNT 8 BITS
014D  0E04   NIBBLE  MVI C,4      ;TO COUNT EACH 4 BITS
014F  F1     SHIFT   POP PSW      ;GET IT BACK AGAIN
0150  07             RLC          ;TO CHECK EACH BIT
0151  F5             PUSH PSW
0152  DA5A01         JC ONE
0155  3E30           MVI A,'0'    ;NO CARRY THEN IT WAS A 0
0157  C35C01         JMP NXTBIT
015A  3E31   ONE     MVI A,'1'
015C  CD0CE0 NXTBIT  CALL PRINT
015F  15             DCR D
0160  CA0301         JZ MORE      ;DONE ALL 8 BITS
0163  0D             DCR C
0164  CA6A01         JZ SPACE     ;DONE 1ST 4 BITS, LEAVE SPC
0167  C34F01         JMP SHIFT
016A  3E20   SPACE   MVI A,' '
016C  CD0CE0         CALL PRINT
016F  C34F01         JMP SHIFT
          ;
0172  C630   CONVRT  ADI 30H      ;ADD OFFSET TO GET NO.'S
0174  FE3A           CPI 3AH      ;CHECK IF STILL IN NUMBERS
0176  F27A01         JP LETTER    ;NO, MUST BE A LETTER (A-F)
0179  C9             RET
017A  C607   LETTER  ADI 7H       ;ADD OFFSET TO GET LETTER
017C  C9             RET
          ;
E00C  =      PRINT   EQU 0E00CH
E003  =      WARM    EQU 0E003H
01FF  =      STACK   EQU 01FFH
E205  =      CRLF    EQU 0E205H
EAD1  =      QUICKCK EQU 0EAD1H
EB1C  =      GETKEY  EQU 0EB1CH
          ;
017D                 END
```

RANDOM I/O cont'd

We have had several requests for schematics on Texas Instruments' Speak & Spell. Anyone have any idea on how to obtain these? Please let us know - we'll publish any sources in this column.

REPRINTED FROM S.C.U.A. #34, March 1982 - John Harcourt reports that if you have a BASIC ROM PAC upgraded to revision B and you have an Exidy FDS unit, you may find that programs written with this version will not convert to disk BASIC using Exidy's ROM2DSK.COM program which is supplied on their EXBASIC disk. Programs written on the standard BASIC ROM PAC run on revision B without any problems.

I just heard about a place which has some very good bargains for Sorcerer users. Things like a Leedex 100-80 green monitor in a metal case with space for a disk drive - $99.95. 32k Sorcerer model I for $499.00. Also available are 48k motherboards, (if you want to upgrade your model I to a model II send them your old motherboard along with $175.00 and they will send you a 48k model II motherboard) S-100 adaptor boards for the disk drives (limited quantities), some keyboards and power supplies, plastic Sorcerer housings, single and dual drives (no need for the S-100 box with these), singles are priced at $575.00 and doubles are under a $1000.00. They've got a whole bunch of well priced Sorcerer goodies. I think they must have bought most of Exidy Systems' spares when they were holding their moving sale. Call or write to:

John Parnell
South Valley Electronics
2110 E Walsh Avenue
Santa Clara, CA 95056
(408)727-0906

Roger Coe of Box 95, Navajo, New Mexico 87328, would like to know where he can obtain a good analysis of the WP PAC control area. His wish list for the WP PAC also includes the following:
Either more than one holding buffer, or a holding buffer you can add to without unloading first. Longer words witout having to break them up. A command to make a whole line, paragraph, or file all CAP'S or all lowercase. If anyone can help with any of the above items, please contact Roger at the above address.

J. M. Solga, of Fairfax, VA recently purchased an Anadex DP8000 printer for which he would like interfacing information, since he received no manual with it. Anyone who has successfully interfaced this type of printer, is encouraged to share this information with J. M. and others.

Now you can upgrade your Sorcerer I or II to 56K of RAM. Using the new 6116 RAM chips in an EPROM PAC you can now have a system of up to 56k. We hope to have an article about this in the near future.

****

ERROR...ERROR...ERROR...

In Volume IV, issue 2, we had an article on page 41 about 'Graphics for the MX-80. Instruction 'c' should read: **EN 10.** Sorry about that.

Members of the **Sorcerer's Apprentice User's Group** are entitled to 8 issues of the group's Newsletter, the **SORCERER'S APPRENTICE**; the services of the library; access to its on-line CP/M based Computer Bulletin Board Service; other services as they become available.

MEMBERSHIP RATES for 1982: USA - bulk postage - $18, 1st class postage in an envelope - $24; Canada & Mexico - $24; single issues $3; all others - air mail - $32, single issues $4.

BACK ISSUES:	Sorcerer's Apprentice Vol III (1-8)	$ 12
	S.A. Vol. III (per issue)	$2.50

PROGRAMS:	Personal Accounts Payable System on disk (3 diskettes, Micropolis Mod II format)	$50.00
	Telecommunications Program (Cassette)	$30.00
	CP/M Public domain programs (see Vol. IV, issue #2, page 32 for details)	

Overseas orders for back issues add $4 per volume or $1 per issue to cover additional air mail postage and handling.

Make checks or money orders (only in US funds drawn on a US bank) payable to: **SORCERER'S APPRENTICE.**

Commercial advertisers, please contact us for advertising rates. **Non-commercial classified ads** are accepted at the rate of $1 per 35-column line or part-line.

Newsworthy items may be submitted via the MiniCBBS on the Sorcerer-based RCPM at (313) 535-9186, the SOURCE (TCF656), or MicroNET (70150,365), on Word Processor cassettes or CP/M Word Processor/Editor files on Micropolis Mod II hard-sectored diskettes (any of these preferred) or hardcopy. Magnetic media returned upon request. Hardcopy will be returned if requested and accompanied by SASE.

SEND ALL CORRESPONDENCE TO:

SORCERER'S APPRENTICE
P.O. Box 33
Madison Heights, Michigan 48071
U.S.A.

BULK RATE
U.S. POSTAGE
PAID
WYANDOTTE
MICHIGAN 48192
PERMIT NO. 23

RETURN AND FORWARDING POSTAGE GUARANTEED