

# SORCERER'S APPRENTICE

PAGE 1

VOLUME 4

NUMBER 1

Jan. 15, 1982

INTERNATIONAL COMPUTER USERS GROUP/NEWSLETTER

Copyright (C) 1982 by Sorcerer's Apprentice - All rights reserved Price \$3.00

## IN THIS ISSUE -

RANDOM I/O.....	1	PROTECT EDIT7.....	11
THE OFFICE SORCERER.....	2	DEVELOPMENT PAC EXTENSION...	14
STRINGY FLOPPY REVIEW.....	2	SERIAL PORT REVISION.....	15
'MESAG' METHOD.....	5	RENEWAL FORM.....	17
RCMPAC NOTEBOOK 1.....	5	SERIAL I/O W/ PARALLEL PORT.	18
4TH TIP.....	7	DUST NOTES.....	18
PROGRAMMING HINTS.....	8	SAVING STRING ARRAYS EXCAS..	19
DUSTINGS.....	10	WP PAC & MDOS TIPS.....	20
BITS & BYTES.....	10	HARDWARE NOTES.....	22
APPRENTICE PORT.....	11	PASCAL PORT.....	23

## RANDOM I/O

by Don Gottwald

Ralph went on vacation - so yours truly is responsible for this issue. Please don't judge too harshly - it's my first attempt at the whole issue.

**News from Exidy:** Exidy Systems Inc. is a wholly owned subsidiary of Biotech Capital Investments. Paul Terrell has left Exidy Systems Inc. and so far noone has been named to succeed him as President. It is a distinct possibility that Exidy Systems Inc. will shift its corporate headquarters and manufacturing facilities to the Dallas, Texas area. Watch for announcements of new generation products from them in the near future.

M/T Microsystems Pascal will work in a 56K Exidy Sorcerer. To get 56K of memory you now have a choice of either a **RAMPAC** from Weston Microtechnology in Ireland or perform the modification by Ed Mentzer (see SA Vol. 3, No. 6, page 108). The Exidy **AUTOBOOT PAC** has the new static memory chips, which are pin for pin compatible with the 2716 EPROM. The PAC contains 6K of RAM and 2K of ROM for the 'autoboot' function. Weston's RAMPAC requires only the addition of one wire in order for it to function.

F. C. Creed of RR #1, Hubbards, N.S., Canada, B0J 1T0 has a problem with the WP PAC. For the first hour of operation it misbehaves, sometimes locking up the computer and at other times it fills the screen with garbage. Any ideas from anyone? He would also like to know how he can use his Development PAC with the Micropolis Disk System. He can save Source code on disk, but not Object code. If anyone can help - please write to him or, better yet, write an article for everyone's benefit.

Jack MacGrath issues this warning to people not too familiar with electronics: Know how the system works - be familiar with electronic troubleshooting methods - use fine handtools such as a low wattage soldering pencil, solder sucker etc. - know how to read schematics and be very careful -- one slip and --- presto -- a large repair bill and aggravation for the experienced technician.

Dr. Douglas Matheson of the College of the Pacific, Stockton, California 95211 would like to find Sorcerer owners in his area. He is also a ham radio operator (WA6WRR) and would like to communicate with other hams who own Sorcerers. Where can he purchase a Disk System with CP/M for a reasonable price? His department is on a limited budget. If anyone can help - please let him know. Also, how can he make his NEC5530 printer print bi-directionally? He has literally hundreds of questions - with noone to turn to. (NOTE: We are quite willing and able to provide our readers with the answers to their questions - BUT if you don't write in with specific questions - we won't know about your problems and consequently won't be able to help you. So let's hear from you. We will answer most questions through the newsletter - some may be referred to experts in the field who may answer you individually. ED).

continued on page 16

THE OFFICE SORCERER

by Roger Hagan, Business Editor

Here is my technique for getting a Basic listing edited in the Word Processor back into Microsoft Basic or EXBASIC. The addresses for Exidy EXBASIC are higher because it is larger. The file in the Word Processor must begin with "GO 100" and conclude with:

```
DEF USR0=&HE003
? USR(0)
SE I=K
```

(in EXBASIC "BYE" replaces these two lines) then saved to disk as a Word Processor disk file. First, though, carriage returns at the ends of lines which are not really the ends of basic lines must be removed, and a space must be added at the end of PRINT lines which wrap around to the head of the continuing next line.

Then load MBASIC/EXBASIC, and get back to the Exidy monitor. I get back to CP/M from MBASIC with SYSTEM and then type M, because I modified my CP/M to give me a jump to monitor when an invalid disk drive is called. From EXBASIC just type BYE. In monitor, preserve the first two pages of Basic by MOVing 100 2FF 5B00, or for EXBASIC, 6600. Re-enter CP/M and use PUT.COM (available from the Apprentice BBS or CPMUG disk 15) to PUT Program.WPE at 5D00 X (for non-auto execute). With EXBASIC, make it 6A00 X. Then bring in BASWR.COM, which is listed below, with the command MBASWR or EXBASWR. It leaves you in the monitor. Now use SE I=AE. The carriage return starts execution. The program whizzes into BASIC, and error messages may fly by. Try to note where they are for troubleshooting. The keyboard is locked out until you end up in monitor.

BASWR, when executed, moves itself to the CP/M default buffer at 80H, then moves the two pages of Basic previously protected from the smashing of PUT, back into their home at 100-2FF. At AE in the default buffer is the "read file" routine which makes each character of the ASCII file at 5D00 (6A00 for EXBASIC) available to Basic as if typed in via your having changed the monitor's input vector. At the end of the listing it reads commands which send MBASIC back to the monitor (MBASIC, unlike EXBASIC, has no 'BYE' command), where the final SE I=K is read, returning control to the keyboard. The ASCII file from your Word Processor must be 'PUT' way above the top of Basic because the Basic program it builds will gradually overtake it. The starting point may be lowered to suit the length of your program. The address change must be reflected in bytes 128-129 of BASWR.COM, which is given here for MBASIC:

```
100: 21 0E 01 11 A0 00 01 1C 00 ED B0 C3 A0 00 21 00
110: 5B 11 00 01 01 00 02 ED B0 C3 03 E0 E5 2A BA 00
120: 7E 23 22 BA 00 B7 E1 C9 00 5D
```

The EXBASIC version should have 66 at 110 and 6A at 129.

Roger Hagan Associates
109 Belmont Place, Seattle, WA 98102 Tel. (206)394-5034

CASSETTE ASTRONOMY PROGRAMS FOR EXIDY SORCERER

- MARS - (\$10) Distance and angular diam. of Mars for any date; date and details of next opposition following any date.
MVEN - (\$10) Phase, distance and angular diam. of Mercury and Venus for any date and next elongation after any date.
MERVE - (\$10) Graphical display of Mercury and Venus relative to Sun for series of time intervals; distances, etc. for any date.
PRISE - (\$10) Risings or settings of Mercury and Venus before or after the Sun for any location and date.
RISES - (\$10) Times of rising, transit, and settings for any planet, Sun, or Moon for any location and date.
SSTAR - (\$20) Dates, radiant, etc. of annual meteor showers and graphical display for selected month.
SASE for details on these and other astro programs to:

Eric Burgess, F.R.A.S.
13361 Frati Lane, Sebastopol, CA 95472
(Overseas add \$2.00 for Airmail, each order)

<<< CLASSIFIED ADS >>>

I am interested in used equipment for the Sorcerer and also a Sorcerer Model II. Must be convertible to 220 VAC, 50HZ. Please send information and price (if possible including shipping by surface) to: Knut Nilsen, Box 2 Hovseter, Oslo 7, Norway
\*\*\*\*\* COMPLETE SYSTEM -- 48K Sorcerer, S-100 Unit, 630K Micropolis dual disk system, 9" Sanyo monitor, dual cassette system, BASIC, W/P, and Development PACs, plus lots of software. Call: Roger Gough at 201-549-8600 days, 201-544-0977 nights/weekends

STRINGY FLOPPY FOR THE SORCERER

Review by Ian Duddy

ASP MICROCOMPUTERS' implementation of EXATRON'S STRINGY FLOPPY DRIVE has been in almost continuous operation on the eight Sorcerers used in the Geology Department, University of Melbourne (Australia), for the last six months. The Sorcerers are used mainly for word processing and also for data collection and manipulation. The Stringy Floppy (SF) has been nothing short of remarkable in both operation and convenience when compared with both cassette and disk systems, and cannot be beaten at the price.

The standard ASP system consists of the following:

- 1. The ASP controller unit, housed in a small plastic case, comes with its own plug pack power supply and provisions for two drive units. The controller plugs into the 50 pin S100 socket.
2. An EXATRON STRINGY FLOPPY DRIVE.
3. Two new monitor ROMS implement new SF commands and as a bonus correct all known Sorcerer monitor bugs. (The new monitor is 10 bytes larger than the original and as such some existing user programs may require modifications that are outlined in the users manual).
4. 10 blank SF wafers.
5. A wafer containing data I/O programs for 8, 16, 32 and 48 K machines, a test program and a comprehensive memory test program to replace the standard routine.
6. A comprehensive user manual.

Data is recorded digitally on endless loop wafers at approximately 9000 baud! - that's right, about 8 times faster than the fast cassette operation and infinitely more reliable. What's more, the read/write operations are completely under computer control which means no hassles like those with the cassette volume and tone controls. Up to 127 sequential files may be stored on the wafers which range in capacity from 5 to 75 Kbytes (5 to 75 feet).

If the SF is plugged in, I/O is automatically set to the SF drives. I/O is changed between SF and cassette by issuing the new monitor commands: SE M=C (cassette) and SE M=S (stringy floppy). Other SE commands are rationalized and a new command, CE, has been implemented to certify wafers (erase and measure their length).

When writing Monitor or Basic files the SF automatically writes and confirms the files by making two passes of the loop without

## THE HIGH COST OF HIGH SPEED

Until recently, to realise the power of the Sorcerer you required an S100 Expansion Unit plus floppy disk drive and controller. Now Exidy's direct plug-in drive provides a less expensive alternative. But the investment is still significant, and who wants only ONE drive!

There had to be a cheaper method of high speed storage.

So as the Australian Distributor for the various versions of the Exatron Stringy Floppy, we decided to design a special version for the Sorcerer.

### STRINGY FLOPPY

STRINGY FLOPPY is a high speed digital tape transport system for storing programs and data entirely under the control of the computer. There are no controls to adjust, just one light indicating the drive is in operation and another indicating that data is being written onto wafer. Drives are individually packaged and connected to the Controller via a narrow ribbon cable.

### WAFERS - NOT CASSETTES

STRINGY FLOPPY stores data on specially designed "Wafers" about the size of a credit card and 5mm. thick. These contain an endless loop of special chromium dioxide tape ranging in length from 1.6 to 23 metres. Their low mass means they operate reliably at high speed.

Removal of a reflective label protects them against accidental overwriting of data.

The special tape and true digital recording technique (like floppy disks) means reliability. STRINGY FLOPPY does not use fluctuating audio tone like cassettes.

### LOW MAINTENANCE DRIVE UNIT

The STRINGY FLOPPY Wafer has the pressure roller for the capstan built in. So the capstan and the record head are fixed in the die-cast aluminium drive.

No adjustments required.

The Wafer just slides into the Drive and "clicks" home.

No other mechanical motion is required. And because Wafers contain endless loops, the tape always travels in the same direction.

There is no need for rewind capability. Standard speed is 25 cm/second, and fast forward (25% faster) is implemented where appropriate to speed throughput.

Simplicity means reliability.

### RECORDING SPEED DENSITY

STRINGY FLOPPY records at 8500 baud (850 characters per second). 28 times faster than the Sorcerer's lower tape rate, 7 times faster than its higher tape rate. On a 23 metre Wafer you can fit approximately 75K.

### CONTROLLER

The compact STRINGY FLOPPY Controller plugs directly into the 50 way connector on the back of the Sorcerer. If you already have accessories plugged into this connector, a 2 for 1 bus extender is available for \$A35. The Controller will handle one or two STRINGY FLOPPY Drives.

# STRINGY FLOPPY FOR SORCERER

Power to the Controller is provided by a plug pack. STRINGY FLOPPY Drives draw their power from the Controller. In operation the Controller turns off the Sorcerer's random access memory between 46K and 48K to accommodate a 2K read only memory (ROM) in the Controller. This ROM contains firmware routines to integrate the Sorcerer to the STRINGY FLOPPY.

### MONITOR

Also included with STRINGY FLOPPY are two further ROMs to replace the Monitor program in the Sorcerer. Known bugs in the Monitor are corrected and routines included to allow computer input/output to be directed to tape or STRINGY FLOPPY using the SET Command. If STRINGY FLOPPY is plugged in at power up I/O is set initially to STRINGY FLOPPY. Others sell Monitors without the original bugs for around \$100. Ours is included in the price of STRINGY FLOPPY.



### MONITOR/BASIC

STRINGY FLOPPY may be used from either the Monitor or Basic. Wafers may be certified, programs saved or loaded. Up to 127 files may be stored on a Wafer (subject to capacity).

### EXTRA SOFTWARE INCLUDED

To make room for our Monitor enhancements, we had to leave out the Sorcerer's memory test. So with each STRINGY FLOPPY we include a much more comprehensive Memory Test on Wafer. In addition, to enhance Sorcerer Basic we include a Data I/O Program that allows high speed storage of String Arrays on Wafers. Full instructions and a demonstration program are included.

### WORD PROCESSOR PAC

As in the case of disk systems used with the Sorcerer, a patch program must be loaded from STRINGY FLOPPY to assign Exidy's Word Processor Pac input/output to

STRINGY FLOPPY. This takes only a few seconds. The patch program is \$A50. Remember to order it if you use the Pac with your Sorcerer.

### PRICE

To be honest it couldn't be done for the price of the Tandy version. After all you get a replacement Monitor correcting known bugs, implementation of the fast forward capability of the Drive, a Data I/O capability, comprehensive Memory Test, higher data density, AND a separate Controller design allowing low cost addition of a second Drive. So although the initial cost may be higher, the cost of a two Drive system is substantially lower.

Including an initial supply of 5 Wafers STRINGY FLOPPY for the Sorcerer is \$A339 (Australian Dollars). And the extra Drive is \$A157. If you order as a two Drive system at \$A496 we'll include an extra 5 Wafers on us. Add \$A10 for surface mail from Australia to the USA, or \$A39 for airmail (both including insurance). Extra Wafers are available within the USA from Exatron Inc.

If you need more detailed information, Manuals are available separately for \$A20 including air mail postage.

A 6 month limited warranty applies to Drive and Controller.

All prices are Australian Dollars, and Bank Drafts must be in Australian Dollars.

Orders may also be placed on your VISA or MASTERCARD card by forwarding signed order quoting full name, address, Card Number, and Expiry Date.

### ASP MICROCOMPUTERS

ASP is an electronic distributor and design company based in a suburb of Melbourne in beautiful Australia. Having experienced the problems of inadequate attention from overseas suppliers ourselves we are particularly mindful of our obligations to our customers. We aim to establish a substantial export market for our products.

Note: Sorcerer and Word Processor Pac are products, and no doubt trademarks of, Exidy Inc. Stringy Floppy is a Trademark of Exatron Inc.

# ASP

## MICROCOMPUTERS



Telephone:  
(03) 2118855 2118344  
797 DANDENONG ROAD,  
EAST MALVERN 3145.  
VICTORIA AUSTRALIA.

**FOR THE EXIDY SORCERER™**

# DUEL



## **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)  

We think you'll like it!

**DAYSPRING**  
COMPUTER ENTERPRISES

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

## ROM PAC NOTEBOOK NO. 1

by J. de Rivaz B.Sc.(Eng)  
West Towan House, Porthtowan  
Truro, Cornwall TR4 8AX  
United Kingdom.

### Introduction:

These notes are intended to enable newcomers and old hands alike to make the best use of their ROM PACs. Most of the work will be on the BASIC PAC, but we will from time to time consider the Word Processor and Development PACs. In this first article, various ideas not in the Exidy manuals will be summarized, together with references. In future, such topics as saving arrays from BASIC using the monitor routines, no wait key polling and machine code generation of medium resolution graphics will be discussed.

I invite readers with ideas they would like to see included to write in with them, and also for those with queries to write in with these also. If I don't know of a solution, then the problem can still be mentioned in the column in case another reader knows the answer.

I am also anxious to communicate with any reader interested in Immortality or Cryonics with a view to using home computing as a means to contact other likeminded people. It is not my intention to discuss these matters in this column, though.

### Word Processor ROM PAC:

Saving with the monitor:

The file saving methods employed by the Word Processor PAC (WP) are clumsy to use, and I have not found them to be reliable. Therefore I have invented a way of saving files with the monitor. It can be done by hand, by exiting to the monitor with Command X, dumping 074AH and 074BH, and saving from 0740H to the quantity held at 074AH. Where the number at 074AH is XX and that at 074BH is YY, enter SAVE TEXT 0740 YYXX. This is because the Z80 stores the high byte second in a sixteen bit word. I have produced a little routine that saves files this way from an unused command. It also takes the first five letters after the first asterisk in the text as the file name, else names the file "TEXT". This will appear in a future issue.

This text can be loaded by exiting to the monitor and loading in the normal way. The jump to WP warm start (0C003) and find your text safely loaded. If you have been sent text from a WP whose parameters have been changed for another printer, and you are using an initialization routine with your printer, then you may need to re-enter via the initialization routine.

The use of monitor loading is quick and has the good reliability of monitor loads. The only loss is the inability to spool through vast amounts of text from one cassette machine to the other. However, it is not impossible to organize one's text into chapters so that this facility is not needed. If you are using it a lot, then you may well be better off using discs anyway.

**BASIC ROM PAC:** Line Length: Poke 322 N, where N is line length. This is useful when printing a listing on a printer whose line is set for more than 64 characters.

cont'd on pg. 6

## THE MESAG METHOD

by William A. Heitman

I was very interested in the program by Bob Stafford which was included in Volume 1, Number 5 of Sorcerer's Apprentice. The problem of the Development Pac's lack of message handling capabilities is a very real one for those of us who are attempting to use minimum facilities until we understand more about what features we do want in a computer.

I enlarged Bob's program a bit to include a search routine to find the message file location of an executable program. I find it more convenient to assemble a program, link it where I want it, and then run this program to add the messages. That way I save the program and message file at the same time without the need of creating a message file, saving it and reloading it at the end of the assembled program and then saving the entire program.

This program uses all the features of Bob's program (which I have repeated below) with a few exceptions:

The MESAG-method is used to allow the programmer to input the starting search address.

The load address can be entered to facilitate the search or if <CR> is entered, the search begins at location 1000H.

The 'RUB' command is used without the shift (It is not possible to 'RUB' out more than the message file or to enter more than 65 characters per line)

My thanks to Bob Stafford for the idea and the work he did before I got to it. Here is the program ---

A machine language program that will search the Exidy Sorcerer memory to locate the message location identified by a 'MESAG DEFW 0303' statement at the end of the source program. This will allow the programmer to insert any length message or messages to be read by a subroutine within the source program and will inform the programmer of the lower boundary of the program to be saved using the monitor.

To use this program, proceed as follows:

>GO nnnn (or LOG program) ; where nnnn is the load address  
Enter load address of program (or <CR>): :MSG #1

Message buffer located :MSG #2  
enter <CR> and begin entering messages

The following value should be used to save  
the executable program and MESAG file

End address

>SE X=nnnn (nnnn=load address)  
>SA MESAG nnnn xxxx (xxxx=end address)

In the event the program search cannot locate the coded value '0303', an additional message will inform the programmer of this fact and will return to the monitor.

The MESAG program cannot locate the message buffer : MSG #4

The coding of this program takes advantage of some of the input and output routines of the monitor.

If <CR> is entered instead of the LOAD address to the first query, the program search begins at 1000H.

Notes on using this routine:

This program stores the message file at the end of the assembled, linked program. It searches for the value '0303' (ETXETX) and places messages at that location. The most logical place for these values is the last statement in the source code. However, it can be located anywhere if the programmer is very careful about the message length.

At the end of each message, enter the 'TAB' key. This will both, insert a null in storage (to indicate end-of-message) and clear the screen for the next message.

When the programmer is finished entering messages, (after the last 'TAB') press the 'ESC' key and the end address of the file will be displayed at the top of the screen as a four digit hexadecimal number. This number will be used to save the entire program (see how-to instructions).

To erase the previous character, use the 'RUB' (no shift) key. This will erase the character in the file and if it is on the current line of the video display, it will also be erased there.

continued on next page

When testing this program, it will be easier to use an abbreviated test MESAG file instead of entering the longer version found at the end of the source code. The following entry is suggested:

>EN 1DB6

1DB6: 00 31 00 32 00 33 00 34 00 03 03/

This will indicate a '1' if message 1 is accessed; a '2' if message 2 is accessed, etc.

```

*****
*          ***** ASSEMBLY MESAG PROGRAM *****          *
*****
;
; CANNED MONITOR ROUTINES
;
MONTR EQU 0E003H
VIDEO EQU 0E01BH
LININ EQU 0E13AH
SNDLN EQU 0E1BAH
ADRES EQU 0E1E8H
PCRLF EQU 0E205H
PARSE EQU 0E225H
BI2HX EQU 0E23DH
KEYED EQU 0EB1CH
;
CALL CLEAR ; CLEAR SCREEN
ID B,1 ; LOAD 'START ADDRESS' MESSAGE NUMBER
CALL WRIMG ; WRITE MESSAGE #1 TO SCREEN
CALL LININ ; GET PROGRAM START ADDRESS
PUSH HY ; CHANGE START ADDRESS LOCATION
POP HL ; FOR CANNED ROUTINE CALL
CALL PARSE ; GET ALL FOUR INPUT CHARACTERS
JR Z,ASIGN-§ ; IF NO INPUT - ASSIGN VALUE
CALL BI2HX ; CONVERT TO HEX
JR CNTLE-§ ; DO NOT ASSIGN VALLE
ID DE,1000H ; ASSIGN 1000H TO START ADDRESS
EX DE,HL ; MOVE START ADDRESS TO REGISTER HL
;
; ***** FIND SEARCH STRING *****
;
ID BC,8000H ; LOAD NUMBER OF SEARCH LOCATIONS
ID A,03H ; LOAD CHARACTER 'ETX'
FNDBT CPIX ; COMPARE BYTE AGAINST MEMORY CONTENTS
JR Z,SECD-§ ; IF FOUND - CHECK FOR SECOND BYTE
ID B,4 ; IF NOT - LOAD 'NOT FOUND' MESSAGE NUMBER
CALL WRIMG ; WRITE MESSAGE #4 TO SCREEN
CALL MONTR ; RETURN TO MONITOR
SECD CP (HL) ; IS IT A SECOND BYTE MATCH?
JR Z,LDMSG-§ ; IF IT IS - START ENTERING MESSAGE
JR FNDBT-§ ; IF NOT - SEARCH FROM CURRENT POSITION
;
; ***** INPUT MESSAGE *****
;
LDMSG DEC HL ; DECREMENT MEMORY LOCATION
DEC HL ; TO OVERLAY SEARCH BYTES
CALL CLEAR ; CLEAR SCREEN
PUSH HL ; STORE CURRENT MEMORY LOCATION
REPET ID B,2 ; LOAD 'ENTER MESSAGE' MESSAGE NUMBER
INPUT CALL KEYED ; GET CHARACTER FROM KEYBOARD
JR Z,INPUT-§ ; WAIT FOR INPUT
CP 0DH ; IS CHARACTER A 'CR'?
JR NZ,REPET-§ ; IF NOT - NEED TO KEY 'CR' TO CONTINUE
POP HL ; RESTORE CURRENT MEMORY LOCATION
PUSH HL ; SAVE CURRENT MEMORY LOCATION
NEWMG INC HL ; INCREMENT CURRENT MEMORY LOCATION
ID (HL),0 ; LOAD NULL IN FIRST MESSAGE LOCATION
POP DE ; CLEAR STACK - HOUSEKEEPING
PUSH HL ; STORE CURRENT MEMORY LOCATION
ID B,64 ; LOAD CRT CHARACTER-PER-LINE COUNT
MSGIN CALL CLEAR ; CLEAR SCREEN
CALL KEYED ; GET CHARACTER FROM KEYBOARD
JR Z,MSGIN-§ ; WAIT FOR CHARACTER
CP 27 ; IS CHARACTER 'ESCAPE'?
JR Z,GCOUT-§ ; IF IT IS - GET OUT
CP 11 ; IS CHARACTER 'TAB'
JR Z,NEWMG-§ ; IF IT IS - START NEXT MESSAGE
CP 95 ; IS CHARACTER A 'RUB'
JR NZ,STRGE-§ ; IF NOT - WRITE CHARACTER TO STORE AND SCREEN
POP DE ; RESTORE LINE-START MEMORY LOCATION
PUSH DE ; SAVE IT FOR NEXT TIME
PUSH HL ; SAVE CURRENT MEMORY LOCATION
SBC HL,DE ; SUBTRACT START FROM CURRENT LOCATION
POP HL ; RESTORE CURRENT MEMORY LOCATION
JR C,MSGIN-§ ; IF MINUS VALLE - DON'T ERASE

```

continued on next page

BASIC ROM PAC: Printer output: Poke Monitor Work Area (MWA) plus 63 with low byte and plus 64 with high byte of printer driver routine.

BASIC ROM PAC: "INKEY": Poke 318,195; Poke 320,224. The statement "A=Inp(24)" then gives the ASCII code of any key pressed in A. If a BASIC line is of the form "1224 A=INP(24):If A=0 Then 1224", then it loops until an input is received. Alternatively, particularly in games programs, it can be arranged so that a program reacts only if a key is pressed else it runs on to something else. If however, a key is held down, everything stops until it is released. A routine that gets over this problem will be detailed in a future issue. It is used in the game "Cryonics Society Organiser" published in SA3.6, September 1981, but only some of its features are employed in this game.

BASIC ROM PAC: "RANDOMIZE": Continuing on from the above idea, if the statement were to read "1234 A=RND(1):A=Inp(24):If A=0 Then 1234" the RND function is called every time the line loops. As there will be a variable time whilst the operator responds to the loop, the RND is effectively shuffled or randomized. [Reference - for further reading on the reason why Inp(24) works, see SPEC 10 p3 - (Now ESC, European Sorcerer Club newsletter 32, Watchyard Lane, Formby, Liverpool L37 3JU, U.K.)]

Finally, these introductory notes on the ROM PACs wouldn't be complete without mentioning that the Development PAC is several orders of magnitude easier to use if you get Quality Software's Development PAC extension, DPX. This allows rapid switching between editor and assembler and easy saving of files on cassette using monitor routines, as well as a number of useful editing functions. (\$29.95 Address 6660 Reseda Boulevard, Suite 202, Reseda, CA 91335)

Similarly the BASIC PAC is enormously improved by using one of the Editors on the market. The one by Ray Bannon, handled by my own firm, costs #12 and has a number of useful editing functions and commands. For details see advertisement in SA vol 3 no 3, or write. Others include the SPEC/Grimshaw Toolkit which is available from a number of suppliers including North American Software for about \$30, two from Arrington Software for \$10 and \$22, and the rather different "System 3" from System Software and others for about \$A30. I have not seen System 3, but it uses the same commands as the word processor. For further details, see ESC18 p 11, where it is reviewed. It was generally liked except for the fact that one has to call up specific lines to edit them. I would doubt whether it really matters which editor you get, but get one and you'll find your BASIC very much easier to write. The cost is not a big percentage on the price of your Sorcerer!

The Word Processor PAC is designed to work with costly printers that the average user is unlikely to be able to afford. There are a number of printer adaptors available for different printers. One for the Centronics 737 is dumped as part of an article on the Centronics 737 in ESC14, and others are available from suppliers of printers. <>

4th TIP

by Tim Huang, Forth Editor

The Screen Editor - Part 2

In the last issue (#7) I used top-down programming design to lay the foundation for the Screen Editor. All of our designing should be approached in this fashion. You should also keep the term 'structured programming' in mind as we work with FORTH, since it forces you to be structured in your approach.

After we define the high level WORD concepts, we then descend to the next level, repeating this process until we reach the lowest level, in which all the WORDS do the heavy, dirty work. You can then treat yourself to the joys of keying in your routine from the bottom up.

At this point one must be very careful to resist a common programmers' malady known as 'keyboard syndrome'. This affliction is characterized by an uncontrollable urge to begin banging away at the keyboard. Control of this scourge is evidenced by returning to the drawing board. We must now decide what tools are needed to accomplish our goals.

Referring back to the diagram on page 151 of issue #7, we see several unusual symbols which we've never seen before in a flow chart. The diagram is not really a flowchart, but a 'D-chart'. The vertical line, with the word INSTRUCTIONS on it, is the symbol for the direct route through the program. The intersecting point, which should have shown a circle with an X tiling it, is the symbol for a loop which ended with the large dot. The last of the symbols are the Y and inverted Y which stand for: IF (true) ELSE (false) THEN. As you can see, it is possible to literally translate the D-chart directly into FORTH words. For further details about D-charts refer to Mr. Kim Harris' article on this topic in Forth Dimensions Vol. 1, NO. 3, pages 30-32.

There was one set of symbols which I did not mention above. They were the IF-ELSE-THEN. But I did. They were nested IF- ELSE-THEN. We just created a new symbol for it. Under the multiple choice situation we can write the program as:

```
(flag) IF A
      IF B
      IF C ELSE D THEN
      ELSE F THEN
      ELSE E
      THEN
```

Notice that the number of IF's should be paired with the same number of THEN's. It would be nice if we didn't have to type so many IF-ELSE-THEN's. This is possible with the CASE statement.

The CASE statement leaves you a choice. It makes the whole program much easier to read. I can never keep track of more than 2 or 3 IF-ELSE-THEN's.

There are many ways to implement CASE. In fact each CASE would be designed for its case. But I think in general, we can categorize the CASE to two cases:

```
DEC HL      ; POINT TO LAST CHARACTER
INC B       ; INCREMENT CRT CHARACTER-PER-LINE COUNT
INC B       ; INCREMENT COUNT AGAIN
ID A,8      ; LOAD ERASE CHARACTER
CALL VIDEO  ; WRITE ERASE CHARACTER TO SCREEN
JR MSGIN-§  ; GET NEXT INPUT FROM KEYBOARD
INC HL      ; INCREMENT CURRENT MEMORY LOCATION
ID (HL),A   ; STORE CHARACTER
CALL VIDEO  ; WRITE CHARACTER TO SCREEN
DJNZ CRCHK-§ ; IF NOT 65 CHARACTERS - CHECK FOR 'CR'
ID A,13     ; IF IT IS - LOAD 'CR' CHARACTER
JR STRGE-§  ; WRITE 'CR' TO SCREEN AND STORE
CRCHK CP 13  ; IS CHARACTER A 'CR'?
JR NZ,MSGIN-§ ; IF NOT - CONTINUE
SUB 3       ; PUT 'LF' IN REGISTER A
ID B,65     ; LOAD CRT CHARACTER-PER-LINE COUNT
JR STRGE-§  ; WRITE 'LF' TO SCREEN AND STORE
```

```
***** CLOSING ROUTINE *****
GCOUT ID (HL),0 ; CLOSE FILE WITH NULL
      PUSH HL   ; SAVE MEMORY LOCATION
      ID B,3    ; LOAD 'SAVE' MESSAGE NUMBER
      CALL WRIMG ; WRITE MESSAGE #3 TO SCREEN
      CALL PCRLF ; WRITE 'CR' AND 'LF' TO SCREEN
      CALL PCRLF ; MOVE DOWN ANOTHER LINE
      POP HL    ; RESTORE LAST MEMORY POSITION
      CALL ADRES ; WRITE LAST POSITION TO SCREEN
      CALL PCRLF ; DOUBLE SPACE VIDEO OUTPUT
      CALL MONTR ; RETURN TO MONITOR
```

```
***** PROGRAM SUBROUTINES *****
CLEAR ID A,12   ; LOAD ERASE CHARACTER
      CALL VIDEO ; WRITE ERASE CHARACTER TO SCREEN
      RET       ; RETURN TO HOST PROGRAM
```

```
WRIMG ID HL,MESAG ; GET MESSAGE FILE LOCATION
      XCR A       ; ZERO OUT REGISTER A
NXTBT CP (HL)     ; COMPARE MEMORY LOCATION
      INC HL      ; INCREMENT TO NEXT LOCATION
      JR NZ,NXTBT-§ ; FIND END OF MESSAGE
      DJNZ NXTBT-§ ; FIND END OF SPECIFIC MESSAGE
      CALL SNDLN  ; WRITE MESSAGE TO SCREEN
      RET       ; RETURN TO HOST PROGRAM
```

```
***** PROGRAM CONSTANTS *****
MESAG DEFW 0303H ; SYMBOL FOR END OF EXECUTABLE PROGRAM
***** END OF PROGRAM *****
```

After assembling and linking this program in its final form, use the monitor 'EN' command to enter the following MESAG file:

```
>EN 1DB6
MSG #1
1DB6: 00 45 6E 74 65 72 20 6C 6F 61 64 20 61 64 64 72 65 <CR>
1DC7: 73 73 20 6F 66 20 70 72 6F 67 72 61 6D 20 28 6F 72 <CR>
1DD8: 20 43 52 29 20 3A 20 <CR>
MSG #2
1DDF: 00 0D 0A 4D 65 73 73 61 67 65 20 62 75 66 66 65 72 <CR>
1DF0: 20 6C 6F 63 61 74 65 64 0D 0A 20 20 20 65 6E 74 <CR>
1E01: 65 72 20 28 43 52 29 20 61 6E 64 20 62 65 67 69 6E <CR>
1E12: 20 65 6E 74 65 72 69 6E 67 20 6D 65 73 73 61 67 65 <CR>
1E23: 73 2E 20 <CR>
MSG #3
1E26: 00 0D 0A 54 68 65 20 66 6F 6C 6C 6F 77 69 6E 67 20 <CR>
1E37: 76 61 6C 75 65 20 73 68 6F 75 6C 64 20 62 65 20 75 <CR>
1E48: 73 65 64 20 74 6F 20 73 61 76 65 0D 0A 20 20 20 <CR>
1E59: 74 68 65 20 65 78 65 63 75 74 61 62 6C 65 10 70 72 <CR>
1E6A: 6F 67 72 61 6D 20 61 6E 64 20 4D 45 53 41 47 20 66 <CR>
1E7B: 69 6C 65 2E 20 <CR>
MSG #4
1E80: 00 0D 0A 54 68 65 20 4D 45 53 41 47 20 70 72 6F 67 <CR>
1E91: 72 61 6D 20 63 61 6E 6E 6F 74 20 6C 6F 63 61 74 65 <CR>
1E2A: 20 74 68 65 20 6D 65 73 73 61 67 65 20 62 75 66 66 <CR>
1EB3: 65 72 2E 00 /<CR>
```

\*\*\*\*\*

cont'd on page 14

## PROGRAMMING HINT

by Carl R. Manning

### SORCERER VERTICAL TAB

Construct the following string in the beginning of your BASIC program:

```
10 VT$=CHR$(10):REM line feed
20 FOR L=1 TO 5:VT$=VT$+VT$:NEXT L:REM make a string of them
30 VT$=CHR$(17)+VT$:REM add Home at the beginning
```

Now whenever you want a vertical tab you can use a statement similar to the following:

```
50 PRINT LEFT$(VT$,10);TAB(30)"MENU";LEFT$(VT$,20)
```

Notice that "LEFT\$(VT\$,1)" puts you at the top line and LEFT\$(VT\$,30)" puts you at the bottom line. Computer count from zero does not apply here.

You can extend this principle to make a "no erase" horizontal tab as follows:

```
10 HT$=CHR$(19):FOR L=1 TO 6:HT$=HT$+HT$
20 NEXT:HT$=CHR$(13)+HT$
30 FOR L=1 TO 5
40 : READ CATEGORY$
50 : PRINT LEFT$(VT$,L*2+5);CATEGORY$
60 NEXT L
70 FOR L=1 TO 5
80 : PRINT LEFT$(VT$,L*2+5);LEFT$(HT$,32);:INPUT C(L)
90 NEXT L
100 DATA ...
```

Personally, I like to see the cursor only when the computer is waiting for me to do something. Therefore I often make and use the following "hide cursor" string:

```
10 HC$=CHR$(13)+CHR$(1):REM carriage return plus left arrow
50 PRINT CHR$(12);LEFT$(VT$,15);TAB(25)"Sorting...";HC$
60 REM sort ...
```

Be sure to include the ";" after HC\$; otherwise the cursor will pop out again. Two final notes: Be sure to CLEAR enough string space if you are working in Standard Basic. If you are working in a high powered Disk Basic, you might consider DEFining VTAB and HTAB as string functions.

\* \* \* \* \*

### SORCERER REPAIRS

\*\*\*\*\*

Is your beloved Sorcerer down? Having too many CRC errors? Would you like more memory? Just want some help or advice? I am a professional technician, able to competently service the following units at reasonable rates and promptness. All repairs are fully guaranteed for 90 days!

Exidy Sorcerer I & II  
Dot Matrix Printers  
Exidy Expansion Box

Video Monitors  
Acoustic Modems  
All ROM Pacs

And, SOCN, Disc Drives and S-100 Cards!

=====

Send all inquiries to me, or call me after 6:00 pm EST; I will be happy to answer all questions.

Jack MacGrath  
70 Tercentennial Drive, P. O. Box 5  
Billerica, MA 01821 Tel. (617)667-8272

\*\*\*\*\*



# C FOR YOUR SORCERER

Based on Ron Cain's small-C, C/80 was written by Walt Bilofsky of the Software Toolworks. Triangle Systems distributes C/80 for the Sorcerer with an enhanced tutorial introduction to C/80. C/80 needs at least 40K of RAM and either Exidy or Micropolis C/PM.

C/80 gives you the power and efficiency of structured programming. Programs written in C/80 run up to 10 times faster than BASIC and require less debugging.

**\$49.00**

**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

**C/80 includes:**  
 C/80: compiler and library  
 CASM: absolute assembler  
 Sample C/80 programs:  
 file compression utility  
 file comparison utility  
 WP Pac file conversion

Ask your Exidy dealer for Triangle System products or order direct: specify hardware configuration and software format (1200 baud cassette, Exidy CPM, or Micropolis CPM), add \$3.00 domestic, \$8.00 overseas shipping and handling, 5.5% sales tax in Ohio, and send check or money order in US funds to:



**TRIANGLE SYSTEMS**

P.O. Box 44026, Columbus, Ohio 43204

**614/486-3527**

**mentzer electronics**

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

\*\*\*\*\*

CP/M Catalog program, good for cataloging your CP/M disks	\$ 75.00
dBASE II Relational Database Management Program	\$595.00
Exidy 1.1 Monitor RCMS	\$ 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 Exidy, 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

## DUSTINGS FROM THE LIBRARY

by Robert Hageman, Librarian and Sysop

Let us resume our discussion of the CP/M communication programs with MODEM.

MODEM was originally written by Ward Christensen (Ward's best known program is CBBS). Since he released it to the public domain, MODEM has undergone almost constant revision. Assembly switches have been added for non-standard CP/Ms and various modem boards and serial modems. The original purpose remains, provide reliable universal communication between any two CP/M systems. One revision is important to our group, XMODEM by Petersen is a direct descendant of MODEM. XMODEM is the remote version for use by callers to an RCPM, it's revision history is similar to MODEM's. These two programs provide direct, speedy, correct, transfer communication between the "home or business" computer and the "public" RCPM systems (quotes are due to public RCPMs being, most often, public use of a home or business computer).

The most often heard question about MODEM; "Now that I've got it, what do I do with it?". First thing, put Smart Terminal (ST) into storage. You will never need ST as long as you operate CP/M and have the MODEM program. Now you can go software shopping on the RCPMs, ALL the software is there for the price of a phone call.

To operate Modem, set up one of the following commands; MODEM TO.300 for an S-100 modem board  
MODEM T for serial modem  
T= terminal mode, O= originate mode, and .300= 300 baud. You need not give a <CR> just yet. Contact the RCPM or other system of your choice. When the carrier tone is heard, enter a <CR>. Then wait for Modem to sign-on, seat the phone in the cradle if using a serial modem, hang up the phone if using an S-100 board. Type <CR>s until the remote system signs on. (Hint: most remote systems are using a program which scans for the baud rate. Beginning at 110 and progressing through 300, 450 and 600 this program looks at incoming data until it recognizes either a <CR> or linefeed). There is no need to beat out <CR>s at 100 per minute, as a matter of fact, the response of these remotes should occur on one of the first four <CR>s sent. Therefore typing the <CR>s in a smooth unhurried manner can act as a diagnostic procedure in that no response after four <CR>s generally means problems and after eight means "There are problems!".

Ok, we have contact with the RCPM. Locate the files of interest to you and you are ready to transfer them. Tell the remote system; XMODEM S FILENAME.TYP; and wait for Xmodem to sign-on and tell you the file is open and the program is ready to transfer the file. Enter a control-E on your end. This returns you to YOUR system, enter one of the following;

MODEM ROT.300 FILENAME.TYP<CR> for S-100 modem board  
MODEM ROT.300 d:FILENAME.TYP<CR>  
MODEM RT FILENAME.TYP<CR> for serial modem  
MODEM RT d:FILENAME.TYP<CR>  
R= receive a file, O= originate mode, T= return to terminal mode after transfer, .300= 300 baud, d:= optional drive to save file to, and FILENAME.TYP= what to call the incoming file.

At this point the two programs will establish handshaking and the transfer will be done automatically. If any are interested in the exact manner of the transfer, please see MODEMPRO.DOC on our system. Modempro explains the modem protocol use by these programs.

Modem will report "TRANSFER COMPLETE" and the remote system will again prompt you for commands. You may also send files to a remote system by using; XMODEM R FILENAME.TYP and MODEM SOT.300 d:FILENAME.TYP. The transfer proceeds as did the previous case.

So much for RCPM transfers. You may also use Modem to communicate with any other system also using modem. The only differences in the commands are those options necessary to set up one computer as "The Computer" and the other as "The Terminal", by agreement with the other party;

Computer:  
MODEM EA.300, MODEM RAE.300 or MODEM SAE.300  
/\ -  
to computer to terminal normal  
communication transfer transfer  
Terminal; -  
/\

MODEM TO.300, MODEM SOT.300 or MODEM ROT.300  
E= echo mode, A= answer mode and S= send a file. Serial modems will not need to use the A, O or .300 sub-options, as answer or originate is handled by a switch on the modem and the baud rate is standard.

READ THE DOC FILE BEFORE YOU ATTEMPT TO USE MODEM, it saves time and aggravation in the long run. Next issue we'll continue with the CP/M communication programs.

## BITS & BYTES

by Jonathan Burnett

HAPPY NEW YEAR everyone. Those of you that have used (or attempted to use) EXIDY's Z80 DEVELOPMENT PAC are probably experienced in the problems I discussed in the last issue. That is, the constant modification of the I/O vectors just to go from one module to another is truly a headache, and tends to make one wonder if the 'end really justifies the means'!

Well, as I said in the last issue, help is on the way...in fact its already here: read the article on page 14, which provides us with a little utility in machine code that eases our problems immensely!

Once you have keyed this program in and saved it to tape, you can really speed up your development process:

From DDT enter: E E003 [cr]  
From the MONITOR enter: LOG DEVX [cr]

After it is loaded, you will return to DDT, but with a different outlook on life! No longer will you have to set those I/O vectors manually. You now can do all the things we discussed in the last issue by the following steps:

1. TYPE: CNTL-E (1st time entry to EDITOR)
2. Key in your assembly language program.
3. TYPE: CNTL-D (Returns to DDT)
4. TYPE: CNTL-A (Assembles your source file)  
(NOTE: During assembly the RUN/STOP key will pause the listing so you can catch the error messages. The space bar will continue the listing. CNTL-C aborts the assembly and returns to DDT.)
5. TYPE: CNTL-R (Re-enters the EDITOR to correct any errors)
6. Now repeat steps 3 thru 5 until you are satisfied with the results.

This saves you many steps and a lot of grief if you're as prone to making mistakes when setting the vectors as I am. Though not perfect or as automatic as you might like it, but the price is right!

For those of you who prefer a bit more (an understatement), have I got a DEAL for you QUALITY SOFTWARE markets a DEVELOPMENT PAC EXTENSION product similar to the one in this issue. It is written by Don Ursem, and goes a light year or two beyond our freebies! I personally use it exclusively and have done so for over a year and a half with much enjoyment! Here is what you can expect for the \$29.95 that QS charges:

From DDT enter: E E003 [cr]  
From MONITOR enter: LOG DPX48 [cr] (There are different versions for 8K, 16K, & 32K SORCERERS, plus they supply you with a load module version, so you can relocate it to any location you desire!) After it loads and displays a sign-on message, you will be in DDT mode. The DEVELOPMENT PAC will now function normally as before, but you will have approximately 22 additional commands at your disposal. You will now get spoiled very quickly! The most time consuming aspects of using the PAC is when using the EDITOR to key and edit your source code file. With DPX, you enter the EDITOR by typing: #ED [cr]. This is the only command you need to enter the EDITOR, (unless you wish to reinitialize the edit buffer, in which case you would use the old E;ED). Gone FOREVER will be the accidental destruction of your source file by forgetting to use the editor restart command! The excellent documentation accompanying DPX also explains how to recover from an intentional destruction of your source file.

Once you are in the EDITOR, you'll have 12 new commands to enjoy. You can position your line pointer backwards now, with the #U command. It will position the line pointer UP the number of lines you specify. You can also position the line pointer with the #F command. #F allows you to FIND a string, starting at the current position of the line pointer to the end of file. It will then point to that line. The #U command will also allow you to search for a string, however it only searches backwards from the current line pointer position. Speaking of line pointers, that mysterious and invisible marvel, the #LI command will display its value and the line whenever you want!

You'll really appreciate the next two commands: #C and #QC. The #C is a global string find and replace command. You specify which string you wish to find, and the string you wish to replace it with. (The replace function is automatic, without operator intervention). You can also specify how many lines you wish it to search. #QC performs essentially the same function as #C, but when you press enter, it finds the 1st string, and prompts you for a 'Y' to proceed with the change or any other key to continue the search with no changes made. Pressing the ESCAPE key to aborts the function. (NOTE: both commands work in the downward direction, toward the end of the file.

continued on page 20

## APPRENTICE PORT

by Don Gottwald, President

In the last issue (#7) we covered some of the terminology used in assembly language coding. This time we will follow a complete cycle of instructions from fetching of the instruction from memory, to decoding by the instruction register (IR) and finally to execution of the instruction. Refer to the diagram for clarity.

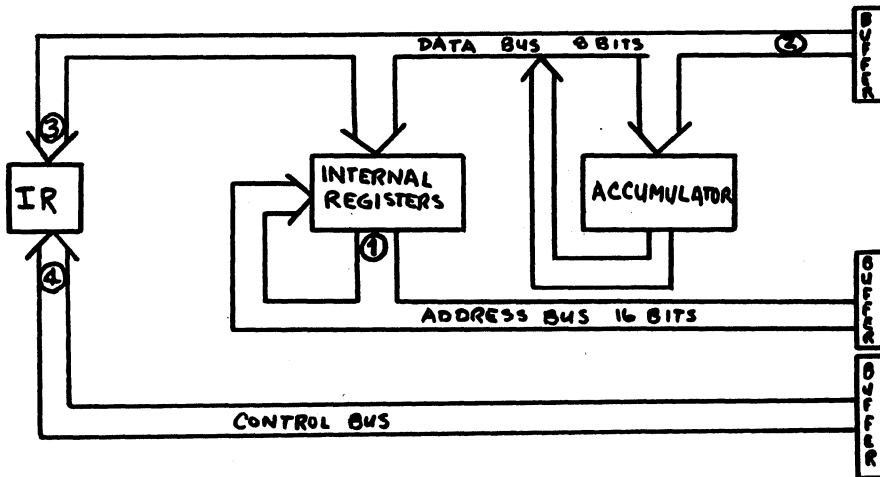
The memory chip can be of any type - ROM, RAM etc. Memory chips store instructions and data. Some additional terms have to be defined. All instructions are executed in three phases: **FETCH**, **DECODE**, **EXECUTE**. Each of these phases require several clock cycles. The Z-80 executes each phase in one or more machine cycles. The shortest machine cycle lasts three clock cycles. Accessing the memory requires four clock cycles. Since each instruction must be fetched from memory the fastest instruction requires four clock cycles.

The **FETCH** phase of an instruction is implemented during the first three states of a machine cycle. These three states (clock cycles) are necessary for all instructions of the microprocessor, since all instructions are in memory and must be fetched prior to decoding and execution.

The first clock cycle (state) presents the address of the next instruction to the memory. This address is contained in the **program counter register (PC)**. The contents of the PC are placed on the address bus. The memory address decoder will decode this address to select the proper memory location. Several hundred nanoseconds later the selected memory location's contents becomes available at the output of the memory which is connected to the data bus. This is clock cycle number 2. While the memory is being read (clock cycle #2) the contents of the program counter (PC) are incremented by one. At the end of clock cycle 2, the contents of the memory are available and can be transferred within the microprocessor to the **instruction register (IR)**. This is clock cycle #3. The instruction which has been read out of memory is deposited on the data bus and transferred into the instruction register (IR), where it will be decoded. During the fourth clock cycle the instruction deposited during clock cycle 3 is decoded by the control unit of the microprocessor. The control unit then generates the correct sequence of internal and external signals for the execution of the specified instruction.

There is a short decoding delay followed by execution of the instruction, the length of which depends on the nature of the specified instruction. Some instructions will execute entirely within the MPU, while other instructions will fetch or deposit data from or into memory. This is why the various instructions of the MPU require various lengths of time to execute. This duration is expressed in number of clock cycles.

During the execution of a program, instructions are in sequence from memory. There is an automatic mechanism within the MPU which fetches instructions in sequence. This is called the **Program Counter (PC)**. Whenever the contents of the PC register are deposited on the address bus, an incrementer attached to it automatically increments the contents and writes it back into the PC register. If the PC contained the value '0' in the beginning it would be output to the address bus. Then the contents would be incremented and the value '1' written back into the PC. The next time the PC would be used, it would point to the correct instruction at the next memory address.



The numbers in the diagram refer to the clock cycles (states).

It is difficult to write this column without some specific input from you, so please, let me know what you would like to see in this column. I would appreciate any comments and or suggestions to be sent to the Sorcerer's Apprentice, ATTN: Apprentice Port. Thank you.

## PROTECT YOUR EDITOR

Howard Arrington's Edit7 editor for Exidy Standard Basic is a good one. With Edit7 loaded at 7000H, and input set to 7000H (SET I=7000), the user can enjoy the benefits of a true screen editor plus a very useful renumbering routine and, with the optional cross-reference routine, a means of generating a variable and program flow cross-reference table. A very nice package.

Just one problem. Edit7 resides in Basic territory; in a 32K Sorcerer, the machine-language editor at 7000H leaves some room for Basic string storage from 7EFFH downward. But what happens if, when the program runs, Basic strings overwrite the editor? Just as you might expect, a complete blitz of both the Basic program and the editor.

Of course, you can reset input to the keyboard monitor routine before you run the program (SET I=K). But then the editor must be reloaded before you again set input to the editor; forget to reload before SET I=7000, and another hang-up results.

Joseph Power and Earl Youngs of East Lansing, Michigan have discovered an undocumented feature of the CLEAR command in Exidy Basic. With the format CLEAR N,n (where N is the decimal number of bytes to reserve for string space, and n is the decimal limit of memory that Basic will be allowed to use), the Arrington editor can be protected from being overwritten by Basic strings.

Simply by using this CLEAR command as the first statement in your program, with the parameters CLEAR N,28671, the editor will be protected.

However, this leaves a bunch of memory above the editor, and below 7EFFH, unused. To make more memory available, it would be useful to relocate the editor higher in memory, to just below the Monitor stack. One could write a program to do the relocating, but perhaps it's just as easy to do it the hard way, with the Monitor ENTER command. To do so, follow the instructions:

1. From the Monitor, with the Edit7 and Cross-Reference program in memory at 7000H, type: MO 7000 75A0 7900.
2. Enter the following bytes at the addresses indicated:

7939:7E	793E:7A	7950:7A
7961:7A	79A6:79	79B3:79
79D8:79	79E1:79	79FB:7A
79FE:7A	7A05:79	7A34:79
7A3A:7B	7A40:7B	7A43:7B
7A46:7B	7A4E:7B	7A53:7A
7A78:7B	7A7E:7B	7A82:7B
7A85:7B	7A9F:7B	7AA6:7B
7AB1:7B	7AB8:7B	7AC8:7B
7ACD:7B	7ADB:7B	7AE2:7A
7AE5:7A	7AEC:7B	7B0B:79
7B16:7B	7B1A:7B	7B1D:7B
7B21:7B	7B37:7B	7C06:7C
7C10:79	7C13:7E	7C16:79
7C64:7C	7C6C:7C	7C72:7C
7C75:7C	7C7D:7C	7C82:7D
7C89:7C	7CED:7C	7D00:7C
7D04:7C	7D07:7C	7D13:7C
7D1C:7C	7D24:7C	7D37:7C
7D3E:7C	7D43:7C	7D4E:7C
7D54:7C	7D65:7C	7D6C:7C
7D77:7C	7D7A:7C	7D7E:7C
7D82:7C	7D8B:7C	7D92:7C
7D9A:7C	7D9F:7C	7DA3:7C
7DAD:7C	7DB6:7C	7DB9:7C
7DC6:7C	7DCD:7E	7DD1:7C
7DE7:7E	7E5F:7E	7E6B:7E
7E78:7E	7E91:7A	7E96:7C
7E99:79	7E9D:79	

3. With these changes made, SA EDIT7 7900 7EA0 to tape. Then, when you load the tape (don't LOG this one!) SET I=7900, write a first line in your Basic program CLEAR N,30975, and you're in business. <>

# ARRINGTON SOFTWARE SERVICE

9522 Linstock, Boise, IDAHO 83704

**SORCERY BREWS** is a manual of programming tricks specific to the Sorcerer. This ready reference of valuable examples simplifies programming efforts and improves both professional appearance and performance. The manual has chapters on programming Tips, Basic's Commands and Functions, Keyboard, Video, Joysticks, Sound, the Monitor, Machine Language Interfacing and Routines, I/O Drivers, Cassette Tape, a source listing of an Editor for Basic, CP/M, Word Processor, Development PAC, Plotting, tables and forms. Best of all, we are very proud of the extensive Basic ROMPAC Map which identifies dozens of useful ROMPAC routines and how they work. Customers have been hungry to have this kind of information collected together and concisely presented in a single manual. I've gleaned my mind of every clever and useful piece of information I know about the Sorcerer. Everyone is sure to discover something they will treasure. Believe me, **YOU'LL USE THIS MANUAL!** It's over 100 pages long and stuffed with hundreds of 'brews'. **\$16.95**

**CHOMP** is an absolute must for every game enthusiast. It is better than 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. As you maneuver around to keep out of the way of the Monsters, you score points by eating small dots found in the alleys. If you eat one of the large dots, you become mightier than the Monsters in that you can now chase, catch and eat them. You score big for each monster you chomp, but be careful as they soon return to their normal rolls as the aggressors. The game is a superb display of high resolution graphics. The program is 100% machine language for speed and ease of use by all of our customers, disk based systems included. This program is fantastic. **WE LOVE IT!!** Move over Galaxians -- you have to share some of the spot light for the elite with this new game entry. The game includes fantastic sound effects and joystick or keyboard control. It has action, excitement, suspense and requires strategy. **\$19.95**

All orders are in the return mail within 3 days. Software is recorded at both 300 and 1200 baud and is guaranteed. We seek to have your approval and satisfaction. We will try to answer questions and be of service in every possible way.

**KEY:** B-Basic, M-Machine code, U-Utility, G-Game, S-Sound, J-Joystick or Keyboard, H-Hardware, E-Education, F-Music File

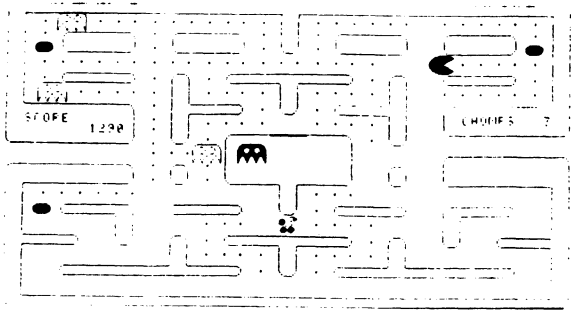
**HOWARD ARRINGTON**

**PHONE: (208)377-1938 Daytime or Evenings**

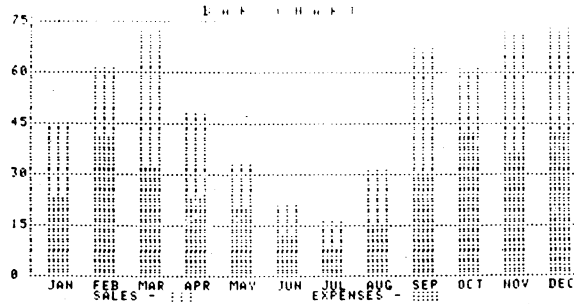
[ ] MUSIC SYSTEM I	\$40.00 BMUSHEF	SORCERER SIZE>>> 16K 32K 48K
[ ] PIANO PLAYER	\$15.00 MF	
[ ] MUSIC SYSTEM II	\$59.95 BMUSHEF	RETURN ADDRESS:
[ ] FINLANDIA FANTASIA	\$10.00 FF	
[ ] BOOGIE & ELEANOR	\$10.00 FF	
[ ] 'JESU' & ODE TO JOY	\$10.00 FF	
[ ] STRAUSS WALTZES	\$ 5.00 F	
[ ] HAYDN SERENADE	\$ 5.00 F	
[ ] MOZART RONDO	\$10.00 F	[ ] CHOMP
[ ] WILLIAM TELL OVERT.	\$ 5.00 F	[ ] JUKEBOX COMPLETE
[ ] MOCKINGBIRD	\$ 5.00 F	[ ] JUKEBOX (no board)
[ ] STING & MAPLE RAG	\$10.00 F	[ ] ARTILLERY
[ ] BACH'S BOUREE IN C	\$ 5.00 F	[ ] CUBES
[ ] CHESS 'BRUCE'	\$17.95 MG	[ ] DATABASE SYS II
[ ] GRAPHICS PACK I	\$25.95 BMU	[ ] M.CODE TUTORIAL
[ ] GRAPHICS PACK II	\$25.95 BMU	[ ] SCREEN GENIE
[ ] DISASSEMBLER	\$17.95 MU	[ ] CASSETTE FILES
[ ] CROSS REFERENCE	\$14.95 MU	[ ] SPACETREK 32K
[ ] MUSICAL HORSE RACE	\$ 7.95 BMGS	[ ] BLACKJACK
[ ] JAIL BREAKOUT	\$ 7.95 MGS	[ ] QUBIC
[ ] EDITOR FOR BASIC	\$10.00 MU	[ ] OTHELLO
[ ] QUICK EDIT	\$10.00 MU	[ ] CONCENTRATION
[ ] MILITARY ENCOUNTER	\$14.95 BMG	[ ] CIRCUS
[ ] 2716 EPROM BURNER	\$49.95 MUH	[ ] MISSILE DEFENSE
[ ] DOUBLE PORT BOARD	\$ 7.95 H	[ ] SUPERX EDITOR
[ ] DOUBLE PORT COMPLETE	\$24.95 H	[ ] DYBUG TOOL
[ ] CHARACTER GENERATOR	\$10.00 BU	[ ] SCREEN SYSTEM
[ ] GALAXIANS	\$19.95 MGSJ	[ ] JOYSTICK PAIR
[ ] SPACE INVADERS	\$17.95 BMGSJ	\$39.95 (+\$6.00 OVERSEAS POSTAGE)
[ ] SORCERY BREWS	\$16.95 (+\$5.00 OVERSEAS POSTAGE)	

(BREWS IS SENT 4TH CLASS POSTAGE, ADD \$3.00 FOR FIRST CLASS)

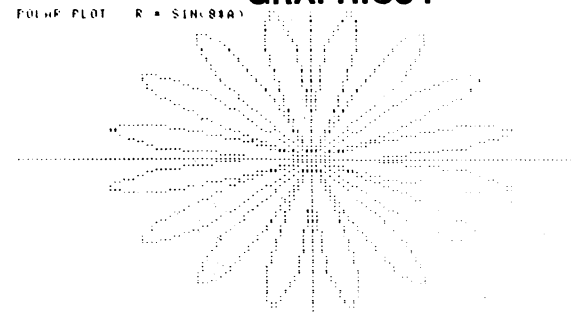
# CHOMP



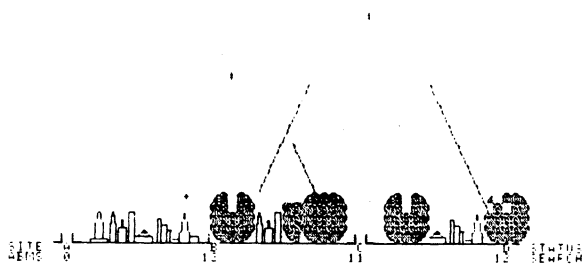
# GRAPHICS I



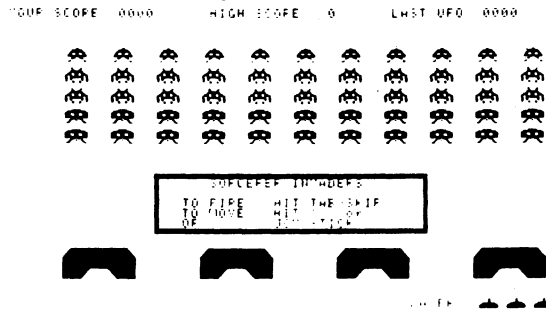
# GRAPHICS I



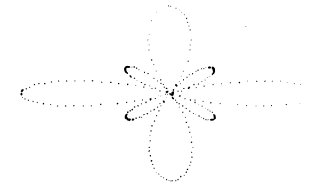
# MISSILE DEFENSE



# INVADERS



# GRAPHIC II



# MUSIC SYSTEM

COPYRIGHT © 1980

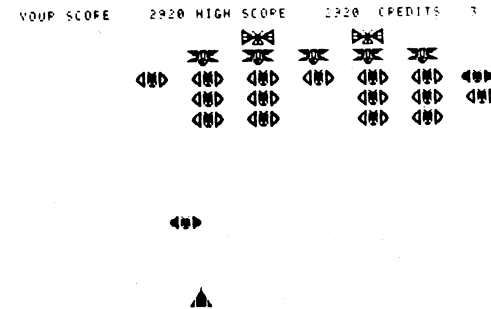
VOICE 1 2 3 4  
SET # 321 OF 440  
TEMPO 140

STEP	VOICE	LEAD	KEY
PLAY	COPI	LOAD	SAVE
PLAY	ENTP	SET	SET#
PPAD	TRNS	EXIT	FILE

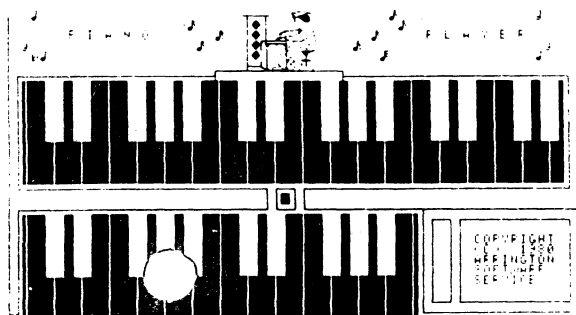
# SORCERY BREWS

This manual is a treasury of programming tricks that are specific to the Exidy Sorcerer computer, although much is applicable to other microcomputers that employ either Microsoft Basic or a Z80 microprocessor. With this ready reference of valuable examples at your fingertips, your programming efforts will be greatly simplified, and your programs will be more professional in both appearance and performance. Using this manual will unleash the hidden powers of your Sorcerer. You will graduate from being an apprentice to being a full wizard as you study and use the brews concocted by masters of the Sorcerer.

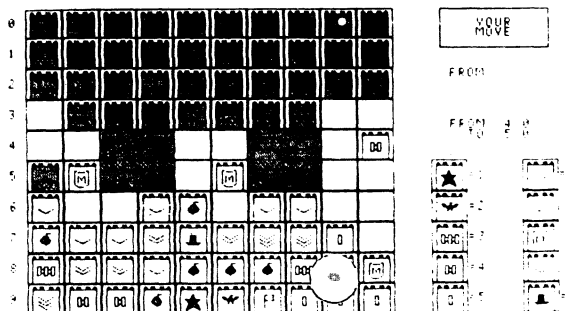
# GALAXIANS



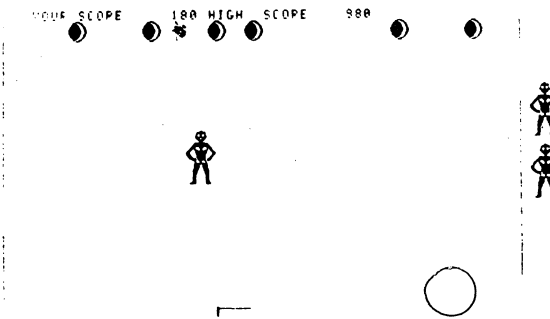
# PIANO PLAYER



# MILITARY ENCOUNTER



# CIRCUS



**DEVELOPMENT PAC EXTENSION**

by Anonymous

As with everyone, the first day I received my computer, I could not wait to program EVERYTHING. As I became accustomed to my system, I realized that an Assembler would be a great asset. So, I wrote one in Basic; it required over six months to complete (without a printer, any development is rough.) Assembling only 70 to 80 lines a minute, it wasn't fast, but it was accurate. Then, I scraped up enough to buy the Development Pac - what a difference! Assembly was exceedingly fast, and in an overall comparison, Exidy won. However, there are several problems with their package. The worst problem is the constant harassment of changing I/O vectors everytime one jumps into editor. Other problems include an assembler which is so fast that the screen merrily scrolls by at Warp 6 (leaving you to wonder just where that ONE error occurred!) and a tape storage/retrieval speed that barely beats archaic systems (such as writing the source file in a notebook.) Of course, don't forget that INIR generates the wrong bytes.

All of these problems can be solved (I have). Since this is not a thick magazine, I have provided a trimmed down version of my Dev Pac extension which should be of use to even the casual user.

FIRST, type the program below in Monitor:

```

ADDR  0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
0000: 18 60 CD 18 E0 28 FB FE 18 CA 03 E0 FE 05 28 3F
0010: FE 12 28 44 FE 0E 28 11 FE 04 28 28 FE 5F 20 02
0020: 3E 7F FE 01 CA 9B CD 57 C9 21 00 00 22 20 F0 22
0030: 28 F0 21 02 00 22 1E F0 21 F5 C5 22 24 F0 21 0A
0040: C6 22 22 F0 21 24 C6 22 26 F0 F5 F1 C3 FA DF 21
0050: 1F C6 22 26 F0 C3 A9 CA 21 1F C6 22 26 F0 C3 B0
0060: CA C9 F5 CD 15 E0 28 11 FE 03 28 DF CD 18 E0 28
0070: FB FE 03 28 D6 FE 20 20 F3 F1 C3 4E C5
    
```

NEXT, type:

```

>SE X=29
>SA DEVX 0 7C ; storage for later use
    
```

You are now ready to use it.

After turning on the power with the Dev Pac in your unit, type the following:

```

.E E003 ; This gets you into Monitor
>DEVX
    
```

After loading, the system will return to DDT80 (.). A list of the commands available to you are included below.

COMMAND	ACTION
cntl-A	Assemble
cntl-E	Enter editor (first time)
cntl-R	Restart editor (Both editor entries set :SI = :BI.)
cntl-D	Exit editor (Jump to DDT80) (Sets :SI = :BO)
cntl-X	Jump to Monitor
cntl-N	Normal. This is also the entry point which sets the following vectors: CI 0002 CO 0000 OI :AO OO :AI SI :BO SO 0000
run/stop	During assembly or edit listing, will pause until the space bar is hit.
cntl-C	Aborts assembly and returns to DDT80.

I hope you enjoy this little tool in your assembly endeavors. For speed, machine language is the ONLY way to go!

\*\*\*\*\*

**HERRATA:** Issue 3.7, Page 148.....

Line 40 of the short BASIC program reads:

```
40 PRINT "1";
```

It should read:

```
40 PRINT CHR$(177) : REM (equivalent to graphic/shift '-' on keypad).
```

\*\*\*\*\*

**A. SEQUENTIAL CASE** -- access to the elements within the CASE with an index which should be continuous.  
**B. RANDOM CASE** -- the accessing conditional figures do not have to be continuous, such as input from the keyboard.

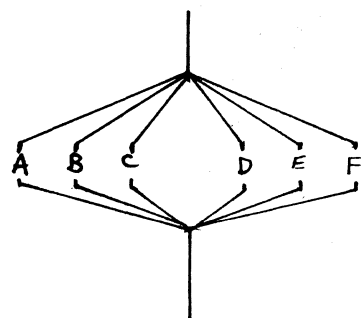
If you don't have the sequential CASE in your vocabulary, please read Mr. Kim Harris' article (Byte, August 1980, page 164). If you want some random CASEs, the FORTH DIMENSIONS dedicated a whole issue (Vol. II, No. 3) to the CASE contest. Or, why not write your own CASE in case you don't like their cases?

I think I'm going to use Dr. C. E. Baker's random CASE for our example here.

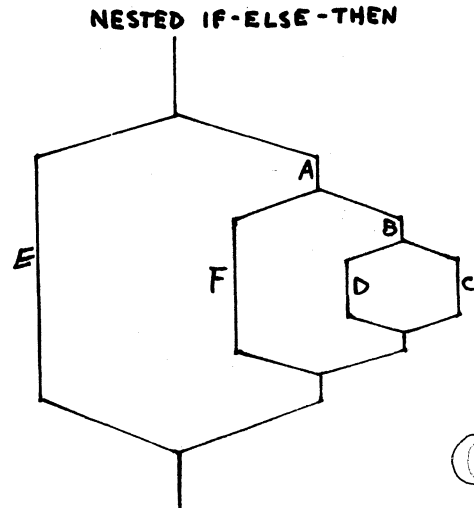
```

:CASE ?COMP CSP @ SP 4 ; IMMEDIATE
:CF ?PAIRS COMPILE OVER COMPILE =
  COMPILE OBRANCH HERE 0 ,
COMPILE
  DROP 5 ; IMMEDIATE
:ENDCF 5 ?PAIRS COMPILE BRANCH
  HERE 0 ,
:ENDCASE 4 ?PAIRS COMPILE DROP
  BEGIN
  SP@ CSP @ = 0=
  WHILE
  2 [COMPILE] ENDIF
  REPEAT
  CSP ! ; IMMEDIATE
    
```

Well, this article is getting too long.....Until then, may the FORTH be with you. <>



CASE SYMBOL



## SERIAL PORT REVISIONS

by Bryan Lewis

The Sorcerer I RS232 serial port has caused many a headache to its owner. Let's look at why and what can be done to solve its problems.

The Sorcerer gets double duty out of its single UART chip. Through some auxiliary circuitry, the UART output is either amplified to 12 volts as required for RS232 interfacing, or else, left at 5 volts for cassette operation. (There is probably more to it than that. I'm not a hardware whiz. Anyway, there are two modes the serial port can operate in). The choice of mode is made by port FE's high bit, which is nice since you can change it from software. Executing `OUT FE,80` (all in hex) sets RS232 mode. `OUT FE,0` sets cassette mode. (The other bits on the port specify baud rate, etc.). Unfortunately, the writer of the Exidy Monitor software erred. Everytime he accessed the keyboard (which is also on port FE -- another instance of double duty), he inadvertently cleared the high bit. So the port goes back to cassette mode; not good if you're driving a serial printer.

To fix it, you can find the circuit in the Sorcerer I chassis that corresponds to that selector bit. That's chip 8B-9. (Look at the Technical Manual schematics, if you have them). You can that high (run a wire to +5 volts), or route a wire through a switch first

if you want to be able to return to cassette operation at some time (you probably do). Even better, install the following simple modification, invented by Berney Zawrotny and me. (see diagram below).

Just one trace to cut, one wire to ground, and one resistor to +5 volts. This makes the selector bit depend on the state of the cassette motor transistors. When the Monitor wants to initiate cassette operation, it turns on the cassette motor and the serial port is automatically put into cassette mode. At all other times the port stays in RS232 mode. No manual switches to worry about. Clever, eh?

Second way to fix the bug...modify the Monitor keyboard routine so that it remembers to put the selector bit back where it found it. That's what the new Exidy ROM's do for you. By the way, that's how other serial-port-related software operates, such as the Smart Terminal program. It includes its own keyboard routine, a copy of the one in the Monitor except that the serial port status is preserved.

Refer to the diagram for the modifications to the Sorcerer I serial interface to automatically select cassette mode when needed, and to stay in RS232 mode otherwise. Changes are pointed out and drawn in dotted lines.

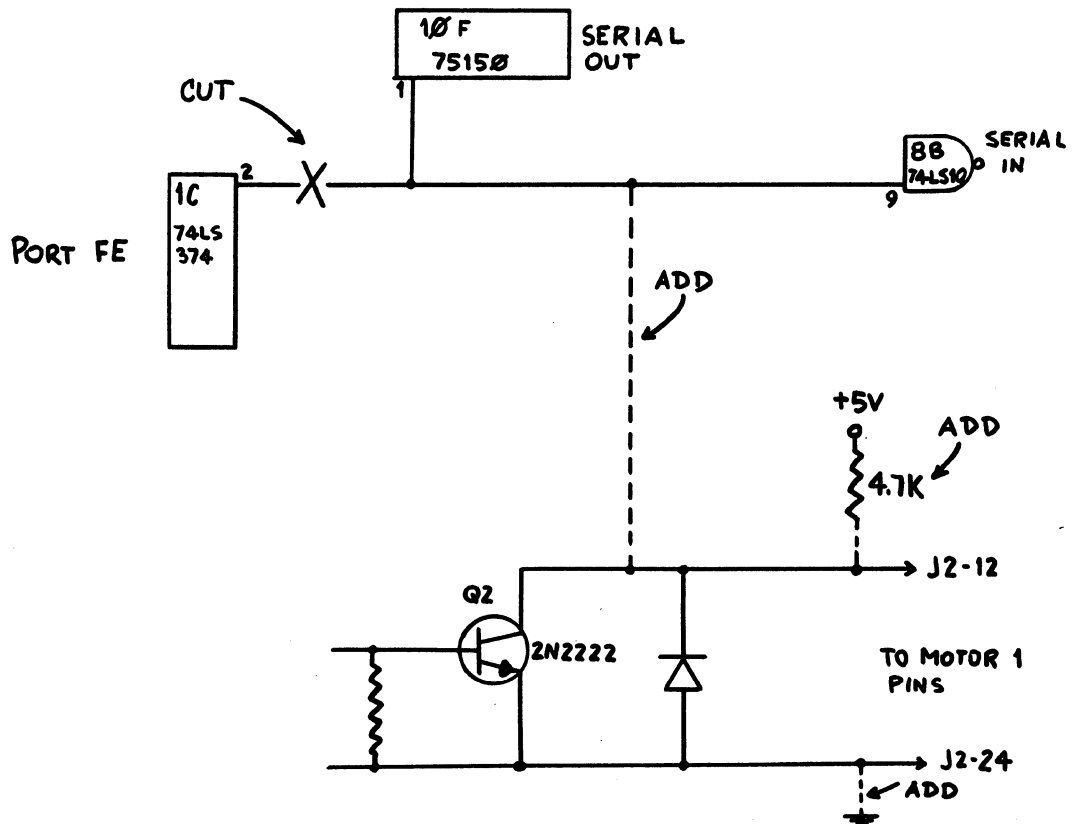
**Old method:** Bit 7 of port FE (1C-2) was connected to the serial input and output circuits 8B and 10F. That bit selected the mode: high --> RS232, low --> cassette. The cassette motor transistor had nothing to do with mode, but it was turned on by the Monitor software during cassette mode.

**New method:** Bit 7 of port FE is unused. The motor transistor, when on, connects the serial circuits 8B and 10F to ground. (Low --> cassette mode). At all other times, the signal is held at +5V, so the default mode of operation is RS232.

If dual cassettes are used, just duplicate the changes. Both transistors will be connected to the same point - in parallel. If either cassette is used, the selection signal will be pulled low, as desired.

There is one case I know of where this doesn't work. The Scott Adams Adventures contain their own cassette routines for saving games. They don't call the Monitor's routines, and the motor transistor never gets turned on; the interface (with above mod's made) stays in RS232 mode. If this is a problem for you, then you'll have to add the manual toggle switch.

If you have any questions - drop me a line c/o Sorcerer's Apprentice.



Harry L. S. Jones of Houston, Texas, informs us that Vista disk systems use hard 10 sector diskettes instead of soft-sector as reported on page 155 of SA Vol. 3, No. 8. He is a dealer for a number of S-100 manufacturers as well as Vista. He is offering a 15% discount to Sorcerer Apprentice members. His address is: 5802B Clarewood, Houston Texas, 77081.

There are a number of people interested in software dealing with amateur radio and electrical engineering. If you know of the existence of such software, please let us know so we can pass this information along.

Bartlett Enterprises, 1277 Airport Boulevard, San Jose, CA 95110, (408)295-5868 is an Exidy dealer.

In response to a reader request for the models of tape recorders which work well with the Sorcerer, here is what Steven Smouse of Texas replies:

Sony TC-62 (about \$42.00) CSAVES & CLOADS 95% of the time with software recorded from the Exidy. Will work only about 50% of the time with purchased software.

Sounuesign #7636 gives excellent result 98% of the time. CLOAD's 80% of purchased software on the first try, 100% for the second or third try. (about \$46.00). Tapes have a lot to do with the quality of recording and playback. Do not use the long playing tapes - they are wound very tight and cause a lot of problems. C-60 should be the longest playing tape you use. Steven recommends Radio Shack low noise, high output tapes in the red or purple packages (\$1.59 to \$2.59).

David Cooke of SAT TRAK Int'l, Treasure Island, Route 1, Box 18, Kingston, ON Canada K7L 4V1 informs us that any member who has an old version of TRAK (Satellite tracking software, now exclusively distributed by Quality Software of Reseda, CA) can obtain an update for \$2.00 if you'll send the old tape back to him at the above address.

Does anyone have a disassembled listing of Exidy's disk BIOS? What software mod's are necessary to read double sided disks with Exidy's SS controller card? Is it possible to write a BIOS that would allow the SS disk (77 track) to read/write to the Micropolis MOD II hard sector disks (77 track)? Does anyone know how to enable EBASIC to read sequential ASCII files without recognizing the comma as a delimiter? Can the Sorcerer run at 4MHZ by replacing the Z80 with a Z-80A?

Trelen Wilson of 2826 Rutland Ave., Des Moines, IA 50311 would like to find Sorcerer owners in his area.

Has anyone tried the 'Last Memory' RAM board? A member in Valley Forge, PA is having difficulties with his and would like to talk to others who have used this particular memory board to see if a fix can be found.

Another member reports that the Panasonic Slimline recorder is very good. Doesn't have motor control, but it works at 300 and 1200 baud without a problem.

A. S. Marland of Paris, France reports a bug in Roth's Z-80 Relocator. at location 504E '0C' should be changed to '03'. The new Monitor 1.1 ROM's from Exidy have a bug in the TEst program. OK still means there is a zero at the indicated bit location, while BAD still means there is a one. No indication of what the bits ought to have been. Put another way, the bad address is compared with 00 during printout, even though it was compared to something else when the fault was actually detected. POP BC (to recover B), PUSH BC (to put C back on stack) needs to be inserted between E963 and E964.

\*\*\*\*\*



**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) @ \$8: \$ \_\_\_\_\_  
~~SUN Volume II @ \$10: \$ NA~~  
~~Sorcerer's Apprentice Vol I (1-7) @ \$10: \$ NA~~  
Sorcerer's Apprentice Vol II (1-5) @ \$10: \$ \_\_\_\_\_  
Sorcerer's Apprentice Vol III (1-8) @ \$12: \$ \_\_\_\_\_  
Sorcerer's Apprentice Vol I-III @ \$2 per issue: \$ \_\_\_\_\_  
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.**

**SERIAL I/O DRIVER USING  
SINGLE BITS OF THE PARALLEL I/O PORT**

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

**DUST NOTES**

from the System

```

0100      0010 *-----*
0100      0020 *  SORCERER READ AND PRINT ROUTINE *
0100      0030 *  USING 1 BIT OF THE PARALLEL I/O *
0100      0040 *  PORTS. 1/30/81 W. HENDRICKSON *
0100      0050 *  BAUD RATE CONSTANTS ARE: *
0100      0070 *  BAUD RATE  CONSTANT *
0100      0080 *  110 -----0B&H *
0100      0090 *  300 ----- 42H *
0100      0100 *  500 ----- 28H *
0100      0110 *  600 ----- 20H *
0100      0120 *  1200 ----- 38H *
0100      0130 *  2400 ----- 19H *
0100      0140 *  4800 ----- 09H *
0100      0150 *  § = ALTERNATE DELAY CIRCUIT *
0100      0160 *-----*
0100      0180 *----- READ AND ECHO ROUTINE -----*
0100 C5      0200 RDPK:      PUSH B      WE USE THESE
0101 D5      0210          PUSH D
0102 DB FF   0220 RDPK1:      IN 0FFH      GET INPUT BIT
0104 0F      0230          RRC *          PUT INTO CARRY
0105 DA 02 01 0240          JC RDPK1      NO START BIT YET
0108 CD 5D 01 0250          CALL HALF    DELAY 1/2 BIT TIME
010B 3E 00   0260          MVI A,0      ECHO START
010D D3 FF   0270          OUT 0FFH
010F 16 80   0280          MVI D,80H   SET FLAG BIT
0111 CD 5A 01 0290 RDPK2:      CALL WHOLE  DELAY 1 BIT TIME
0114 DB FF   0300          IN 0FFH      SAMPLE INPUT
0116 E6 01   0310          ANI 01H     MASK FOR BIT 1
0118 D3 FF   0320          OUT 0FFH     ECHO
011A 1F      0330          RAR
011B 7A      0340          MOV A,D      GET DATA
011C 1F      0350          RAR *          ROTATE FOR NEXT BIT
011D 57      0360          MOV D,A      SAVE NEW PATTERN
011E D2 11 01 0370          JNC RDPK2    LOOP FOR 8 BITS
0121 CD 5A 01 0380 THE END:      CALL WHOLE  LAST BIT
0124 3E 01   0390          MVI A,01H   SET STOP BIT
0126 D3 FF   0400          OUT 0FFH
0128 CD 5A 01 0410          CALL WHOLE  SEND 1 STOP
012B 7A      0420          MOV A,D      DATA TO REGISTER A
012C D1      0430          POP D
012D C1      0440          POP B
012E C9      0450          RET *          ASCII DATA IN REG A
012F C5      0470 *----- SEND ASCII DATA IN REGISTER A -----*
0130 D5      0490 PRINT:      PUSH B      WE USE THESE
0131 F5      0500          PUSH.D
0132 57      0510          PUSH PSW
0133 3E 00   0520          MOV D,A      SAVE DATA IN D
0135 D3 FF   0530          MVI A,0
0137 0E 08   0540          OUT 0FFH    SEND START BIT
0139 CD 5A 01 0550 PRINT1:      MVI C,8      BIT COUNTER
013C 3E 01   0560          CALL WHOLE  WAIT 1 BIT TIME
013E A2      0570          MVI A,01H
013F D3 FF   0580          ANA D
0141 1F      0590          OUT 0FFH    MASK DATA
0142 7A      0600          RAR          SEND DATA BIT
0143 1F      0610          MOV A,D      GET NEXT BIT READY
0144 57      0620          RAR
0145 0D      0630          MOV D,A      SAVE NEW DATA
0146 C2 39 01 0640          DCR C        DECREMENT COUNTER
0149 CD 5A 01 0650          JNZ PRINT1  LOOP FOR 8 BITS
014C 3E 01   0660          CALL WHOLE
014E D3 FF   0670          MVI A,01
0150 CD 57 01 0680          OUT 0FFH    STOP BIT
0153 F1      0690          CALL TWCBIT SEND 2 STOPS
0154 D1      0700          POP PSW
0155 C1      0710          POP D
0156 C9      0720          POP B
0157 C5      0730          RET *          DONE 1
0157 CD 5A 01 0750 *----- DELAY ROUTINE -----*
015A CD 5D 01 0770 TWCBIT:      CALL WHOLE
015D 06 42   0780 WHOLE:      CALL HALF
015F E3      0790          MVI B,BAUD RATE BALD RATE CONSTANT
0160 E3      0800 DELAY:      XIHL
0161 05      0810          XIHL
0162 C2 5F 01 0820          DCR B
0165 C9      0830          JNZ DELAY
0166          0840          RET
0166          0860 BALDRATE EQU 42H
SYMB 2
RDPK      0100 : RDPK1      0102 : RDPK2      0111
THEEND    0121 : PRINT      012F : PRINT1     0139
TWCBIT    0157 : WHOLE      015A : HALF      015D
DELAY     015F : BALDRATE  0042
    
```

The Quality Software Smart Terminal is NOT capable of downloading OBJ files. OBJ files are machine executable code from CP/M.COM files of the same name. As such they contain bytes having values greater than 7F hex. The Smart Terminal program strips the high bit from each received byte reducing each to a value of 7F or less.

IF YOU ARE A CP/M OPERATOR, download MBOOT3.ASM and assemble it for your system. Then use it to get MODEMxxx.ASM. Assemble the Modem program and use it to pick up PLINK.ASM and SETMODEM.ASM/SETTAPE.ASM if you need them. Setmodem and Settape are used with serial modems as port initializing routines. Modem and Plink are now your communications programs (you no longer need Smart Terminal). IF YOU DO NOT HAVE CP/M, most OBJ files will be of no use to you as most need CP/M to run.

I have repeatedly been asked: "Are you really using a Sorcerer for the remote CP/M system?". Yes, a Sorcerer is really being used.

Hardware: Sorcerer II, S-100 Expansion Unit, Micropolis Mod. II, PMMI Modem, 4K RAM Board, Special S-100 EPROM board, Leedex 12" video monitor.

Software: CP/M 2.21, PMMIBYE (adapted to EPROM), CHAT Ver. 1.51 (a slightly modified 1.5), HELP (edited for this system), MLIST Ver. 5.0 (used as TYPE), SD Version not reporting sizes, used as DIR), SD (version reporting sizes, used as SD), XMODEM Ver. 4.2, NEWBALD BYE (a jump routine to PMMIBYE in EPROM), MINICBBS (a custom program written for Keith Petersen by Ward Christensen. The program is only to those who have purchased CBBS from Ward Christensen).

All software, except CP/M, BYE, and MINICBBS can be had by downloading from one of the large RCPM systems. Try Technical CBBS at (313) 846-6127.

BasicE consists of not one but two different programs. BasicE is the compiler module and RunE is the runtime interpreter. BasicE expects to be loaded with the commandline; BASIC FILENAME, where the filetype is BAS. BasicE then creates FILENAME.INT. RunE will load with the commandline; RUNE FILENAME, where the filetype is INT. These programs do not accept input from the console except during a program run that calls for such input (you don't type the programs in via the console).

We have the complete set of the newest versions on the B: Documentation is available from:

Computermart of New Jersey  
501 Route 27  
Iselin, NJ 08830  
The price is \$5.50 plus shipping (about \$1.00). <>

## SAVING STRING ARRAYS IN EXTENDED CASSETTE BASIC

by R.L. Henne

I have had Exidy's Extended Cassette BASIC (EXCAS) up and running for several months now, and found it to be a most capable language. However, like the ROMPAC BASIC, EXCAS has no CSAVE\*/CLOAD\* capability for string arrays. [This is because string arrays do not occupy contiguous memory locations.] This deficiency is somewhat easier to overcome in EXCAS than in ROMPAC BASIC due to the presence of integer arrays and the VARPTR function.

The EXCAS manual contains a program for CSAVE\*/CLOAD\* of string arrays which packs three characters per 4-byte single-precision floating-point word, for 75% efficiency. The following program packs two characters per two-byte integer word, providing the speediest I/O possible under the circumstances. More importantly, though, it gives S.A. readers a chance to see some of the features of EXCAS in action.

First, you will notice that the apostrophe will suffice for REM, making comments much more readable. The use of "%" after a variable or array name defines it as an integer. EXCAS also has single- and double-precision floating-point types. The OPTION BASE 1 statement in line 1000 allows array subscripts to start with 1 instead of 0. (A victory for the mathematicians over us computer scientist types.)

Note also the ease with which loops may be indented for easier reading without the need for colons. The ERASE command in line 5050 deletes obsolete arrays, which may then be re-dimensioned. Critical to this program is the VARPTR command, which gives the address in memory of a variable.

Other features of EXCAS which provide interesting capabilities include: Hex and Octal constants and functions, Error trapping and simulation, Editing within program lines, Call and expanded USR functions for access to machine language, CLEAR with the capability to set aside memory for machine language subroutines, CLOAD? for verification of CSAVEs, CURSOR addressing, plus LINEINPUT, IF/THEN with ELSE, WHILE/WEND, PRINT USING, RENUM, Program tracing (TRON/TROFF), and lots of other goodies!

One last point of interest - since my printer requires a special software driver, it looked as though I would be unable to take advantage of EXCAS's special LPRINT and LLIST capabilities (which use the Monitor's Centronics driver or a serial driver in EXCAS) without a hardware mod. A bit of detective work led me to find the following code at location 1FC4:

```
1FC4: F5 C3 97 E9
```

This code saves the AF register-pair and jumps to location E997 in the Centronics driver. [The F5 or PUSH AF takes the place of the one at E993, which is followed by a CALL VIDEO, which EXCAS is skipping around.] Therefore, if a special printer driver is required, change the jump address in locations 1FC6-7 to point to it, and make sure it does a POP AF before returning.

As far as I can tell at this point, there is space at 4BC0 for later patches to EXCAS. I put my driver in this area, made the change at 1FC6-7, and re-saved EXCAS. Now, when I >LOG EXCAS, my printer is ready for immediate use, without having to load the driver separately! (I originally tried 4B50, and discovered the hard way that the input buffer took that space on long program lines!)

Happy Programming!!

### PROGRAM LISTING:

```

98 *****
99
100 This program demonstrates a way to CSAVE*/CLOAD*
110 a string array indirectly by packing two bytes
120 (characters) into each element of an integer array.
130
140 The A$ array contains 30 elements of 10 characters.
150 The B% array is the integer array containing 5
160 integers per element of A$.
170 (If any element of A$ is longer than 10 characters,
180 it will be truncated. Longer A$ strings would
190 require enlarging the dimensions of B% and the
200 loop indices.)
210
220 *****
230
240 Lines 1000 - 1200 decide whether we'll read or write
250
260
270
280
290
300
310
320
330
340
350
360
370
380
390
400
410
420
430
440
450
460
470
480
490
500
510
520
530
540
550
560
570
580
590
600
610
620
630
640
650
660
670
680
690
700
710
720
730
740
750
760
770
780
790
800
810
820
830
840
850
860
870
880
890
900
910
920
930
940
950
960
970
980
990
END

```

## RESTORING DELETED WORD-PRO-PAC TEXTS

by Jim Cooper, Box 73, Paramus, N.J. 07652

Always start your text with at least two carriage returns. This way, if you delete the text for some reason and then want to restore it, you need only do the following:

- A. Use X command to enter the Monitor
- B. DUmp 800 900
- C. Use ENter command to change data at locations 80F thru 81E as follows:

```
ADDR: 0 1 2 3 4 5 6 7 8 9 A B C D E F
800:                                     0D
810: 0D 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 0D/
```

D. Return to WORD-PRO with the PP command, and you should be in the EDIT mode with the original text showing in full. Only difference will be that one of the first few lines will be filled with spaces and a carriage return.

## SAVING WORD-PRO-PAC TEXT ON MICROPOLIS MDOS DISKS

by Jim Cooper

It is difficult to save WORD PROCESSOR text on CP/M disks because of CP/M saves memory only from 0100H up (with no indicated way of changing that). Also, the CP/M OS conflicts with memory used by the WORD PROCESSOR PAC. MDOS also conflicts with memory used by WORD PROCESSOR, but allows text to be moved and saved in free memory space.

### TO SAVE TEXT ON DISK --

1. Enter text for WPF b-e-f-o-r-e booting MDOS.
2. Use X command to enter Monitor.
3. Use DUmp 800 FFFF and watch for end of WPF text (identified by at least one string of 03 followed by 15 0E, then blank memory - ie: 00 FF 00 FF, etc.)
4. Determine memory location of end of text. (If greater than cannot be saved by this procedure). Let ABCD=Top of file address.
5. Use MOve 0 ABCD 4000 to move WPF into upper memory.
6. Now, boot the MDOS system.
7. In MDOS, save text as follows --  
SAVE "Unit:XXXXX/ABCD" 4000 DBCD 0

where XXXXX=Filename (5 Char. or less)  
ABCD =Hex address of end of orig. WPF text  
DBCD =4000H + ABCD (you add in hexadecimal values)

Using /ABCD as part of the file name will help when loading the file back at a later time.

8. Use FILES command to confirm that the program was added to the disk directory.

### TO LOAD WORD PROCESSOR FILE FROM MDOS DISK --

by Jim Cooper

1. Power up Sorcerer with WPF PAC inserted.
2. Use X command to enter Monitor.
3. Boot MDOS.
4. Type LOAD "XXXXX/ABCD" 4000
5. Still in MDOS, type EXEC E003  
This will set you back to Monitor with warm-start (no memory changes).
6. In Monitor, type MOve 4000 DBCD 0  
This must be done in Monitor, not in MDOS command, as the WPF text will overlay parts of MDOS and destroy the MDOS system. WPF text has now been moved into place for WPF PAC.
7. Type, GO C003  
This sets you into edit mode of the WPF with a warm start, and your text should appear on the screen.
8. Go to WPF COMMAND mode, type E, and check that all of your text is really present. Command T will set you back to the top of your text and you can continue as if you started in WPF mode. <>

One of the real BIGGIES is the #ST command. It will show you the STATUS of the edit buffer, by displaying the following addresses: START OF TEXT, END OF TEXT, TOP OF THE EDIT BUFFER, LINE POINTER and how many bytes are free in the edit buffer. Now you can use the MONITOR's Save command to write your source file to tape. Use the START and END addresses shown by the #ST command. Finally your source code files will have labels and can be more quickly saved and reloaded. The #SA command will permit you to save any number of lines of your source file, from the current line pointer position to the end of file, or other end address you specify, using the standard MONITOR format, with a label name of your choosing. This one is ideal for saving a sub-routine for use in other programs. You could then merge it into another program with the #LO command. This command will load, by filename, a MONITOR file into your current source file, starting at the line number you specify. It will insert the new file without overlaying any of your existing source text. If you start to run out of room in your edit buffer, the #SQ command can save the day. It will compress all but essential spaces from the source file. This can amount to a 33% savings! The last commands permit you to rapidly jump to other modules, automatically changing the I/O vectors. #DD takes you to the DDT mode. #MO exits you to the MONITOR (where you can return to the EDITOR by typing #ED). #AS also puts you in the DDT mode, but first sets up the vectors for assembling source code. You would then enter the normal assembler command E :AS to begin the assembly. There are also three commands for controlling the display. #PR will PRINT all displays to a RS-232 printer at 300 baud, using a built-in driver. If you have a CENTRONICS compatible printer, you can patch this command to the MONITOR's drivers (the book tells you how!). #PV will display everything on the video screen at 1200 baud. And finally the #PH command activates the PRINT HALT capability. This works with either the #PR or #PV mode of display. Whenever anything is printing on either the printer or the video screen, you can pause the listing by pressing any key once. Pressing any key a second time will resume the listing. If after pausing a listing, you press the ESCAPE key you will immediately exit to the MONITOR. By issuing the #PV command, you can effectively disable the print halt option.

Since all the I/O vectors are automatically set for you by the #ED and #AS commands, you will not ordinarily have to manually set anything whenever you use the RAM buffers for development work. If however you need to use tape drive(s) for assembling VERY large files, or for generating LINK EDIT modules, then all you have to do just before you execute the assembler, is use the DDT vector set command to alter any of the vectors already set, to the ones you need or prefer. If after using these NON-STANDARD vectors, you find a need to return to the RAM buffer settings, then the command #RA, will do it for you automatically.

Starting with the next issue, I will attempt to systematically describe the various components that make up the Z80 assembly language. If you have any particular questions or comments regarding this column, the PAC, or assembly language in general, please write the editor of this newsletter or leave me a note on our RCPM/CBBS. I'll try to answer your questions thru the newsletter or thru the CBBS whichever you prefer. <>

# EXIDY 56K

Just plug in our new RAMPAC and a 48K Exidy Sorcerer becomes a 56K Super Machine. The disk user can now run other languages such as COBOL and PASCAL. Larger business software can also be executed.

**Special introductory offer: \$159.95 including p&p**

\* \* \* \* \*

WOULD YOU LIKE ABOUT 50% MORE SPACE FOR YOUR BASIC PROGRAMS?

WOULD YOU LIKE TO UTILIZE THOSE MEMORY HUNGRY LANGUAGES?

Now, with the WESTON EXMEM, you can increase your memory size from 32K to 48K. This revolutionary board fits inside the Exidy case. The cost for a fully assembled and tested 8K expansion memory unit is \$115.00 plus \$5.00 for postage.

Easy to follow full - fitting instructions are included.

\* \* \* \* \*

## <<<< CUSTOM CHARACTER SETS >>>>

Allows an alternative set of characters for special applications. The extra set is switch selectable. There are several sets already available. Each set consists of 96 characters.

- British - This includes the pound sign
- French - The Franc sign, and the three French accents
- German - All the German character set
- Math/Greek - All the math symbols and full Greek alphabet
- Legal - This includes Copyright and Registered notices

These sets are designed to work with standard Daisy printwheels. The user is provided with the original character set and one alternative set.

Custom character sets can be made upon request at a charge of \$1.80 per character.

Fitting is easy and requires NO soldering.

The price is only \$41.49 plus \$4.00 for Airmail postage and packing. Also available for the Apple, Pet, and Tandy. Please specify which character set is required when ordering.

All prices are in U.S. dollars. (Exidy is a trademark of Exidy Systems, Inc.; Apple is a trademark of Apple, Inc.; Pet is a trademark of Commodore).

Trade inquiries are Welcome - Telephone Dublin 803429

Send all inquiries and orders to:

**WESTON MICROTECHNOLOGY LIMITED**  
WESTON HOUSE  
12 Alam Road  
Monkstown, Co. Dublin  
IRELAND

## HARDWARE NOTES

by Russell Frew, Hardware Editor

One of the most frustrating situations you can encounter is to have a perfectly working system that suddenly decides not to boot your disk. Strangely enough, the next day when you power-up, everything works like a champ.

Immediately, everyone begins to wonder about power line drop-outs, spikes, bus noise and all the other classic causes for intermittent disk failures. Whether you have an Expansion Box or not, there is another culprit that you should look at after you have eliminated power drop-outs.

Exidy needed an octal line driver (8 bit wide transceiver) to sit on the bus of the Sorcerer I to protect the CPU from bus conflicts outside the main system. This activity could come from the Expansion Box or in the case of the Vista drives, the disk controller itself. The chip that Exidy selected is rather an odd one. It is a DM8304 transceiver. There are 7 of them in the Sorcerer I. Unfortunately their reliability over time seems to be rather poor. When they die, they don't always fail in a catastrophic manner. Sometimes only one bit fails. To make matters worse, frequently this stuck bit can get unstuck as the system warms up or after repeated RESETS. This can make troubleshooting extremely difficult.

If your disk fails to respond to your >GO XXXX command, try dumping the bootstrap PROM. For example with a Micropolis drive system using the boot address BC00H, you would give the command >DU BC00 BCFF. This will put the contents of the PROM on your screen. Compare this with your documentation or better yet, a printed copy of the PROM made previously when the system was functioning properly. If the address of the dump is not correct then the problem could be one of the two 8304's that buffer the address lines in and out of the computer. If the data is off in one or more bits then the problem may be in one of the two that sit on the data bus. Hopefully one of the first two areas is your problem because the last option is more difficult to identify. The last 8304 sits on 8 of the CPU's signal lines.

Having done the PROM dump several times to confirm that the same bits stick or drop out each time you are faced with one additional problem. If you have an Expansion Box you might have guessed that Exidy used the same 8304's on the other end of the Expansion Bus cable. Because both chips are drivers it's hard to tell which chip is driving the line and which is sinking the current. If you disconnect the expansion cable you quickly find that all signals disappear on both ends of the system so there is no easy answer there. Assuming that most users don't have the digital instruments necessary to carry this search much farther, let's choose the less elegant solution. If we have the problem isolated to two IC's, let's go with the statistics. The greatest number of failures of 8304's I've encountered have been in the computer not the Expansion Box. Exidy seems to be of the same opinion. The gamble then is to replace the suspect 8304 in the computer first. The address IC's are located at 3H & 4H; data at 2H; and control signals at 5F. There is an additional 8304 located at 1H which buffers the CPU. Unless the system has totally crashed, stay away from that IC.

Carefully unsolder the suspect IC and replace it with a socket. Save yourself a lot of trouble and don't skip the socket. Removal of any IC is always difficult and you will probably have to reconstruct several traces after you finish. Because half of these chips are located under the ROM Pac tray, you most likely will have the main logic board out of the chassis. I would recommend that you rebuild these traces on the bottom of the logic board with 30 AWG wire wrap wire. It is much easier and will result in far less problems. Replace only one IC at a time and then try the system. If the problem remains the same, all that's left is the IC in the Expansion Box. I hope the odds are with you!

The 8304 is not an easy chip to buy. I managed to locate it at Advanced Computer Products, Santa Ana, CA. Their phone number is (714) 558-8813. There are several other sources in Byte magazine.

user intervention. In addition, a fast forward mode, which is about 25% faster than read/write speed, operates in certain circumstances. For example, if file 2 is asked for and it finds that it is in a higher numbered file, the unit will enter fast forward mode until it reaches the beginning of the tape where normal speed is resumed until file 2 is found.

The Word Processor Pac is interfaced to the SF using a short patch program that loads into low RAM (it does not seem to interfere with the Exidy print driver that also loads into low RAM). The patch enables all of the standard WP Pac commands to be used and makes the SF transparent to the user. The one disadvantage of this method is that the full 48K of memory cannot be dumped on a single wafer because of the way the files are stored in 256 byte blocks with intervening blanks. This means only about 30K can fit on a 75 K wafer. A slightly more complicated recording procedure, described in the SF newsletter number 2, shows how to record WP files as monitor files so that wafers can be used more efficiently.

In use, the SF has been extremely reliable with only an infinitesimally small number of write errors detected by the verify loop (verify error displayed). A few parity errors are detected while reading, but these are usually overcome by cleaning the easily accessible head and drive spindle, with alcohol. When certifying new wafers, cleaning should be performed frequently.

I suppose all this sounds to good to be true. Well, there were a few initial problems with faulty wafers that were too wide to go in the drive mouths due to poor plastic welding by the manufacturer, as well as a number of wafers would not certify correctly. It is rare for a wafer to fail after certification, although apparently random failures may occur with any magnetic media. All faulty wafers were replaced immediately, without question. Also, initially, wafers recorded on one drive were not always readable on other drives. This was apparently due to the variable setup (particularly head alignment) of bare drives purchased by ASP from EXATRON. Rigorous alignment procedures undertaken by ASP now mean that any wafer may be read by all drives. The only mechanical failure was a motor, but again this was replaced without question.

When considered overall and in the light of unreliable cassettes and expensive and more complicated disk systems, the ASP Stringy Floppy implementation is very fast, simple to use and very reliable. Above all the Stringy Floppy is a very reasonably priced storage alternative that should be considered by all Sorcerer owners.

PASCAL PORT

by Daniel Conde

Continuing on the discussion of memory allocation in Pascal, we will discuss the proper statements needed to declare and use create objects. These objects need to be pointed to by POINTERS, which contain addresses, but since Pascal is a strongly typed language, we need to direct Pascal as to what sort of object a pointer will point to, as follows:

```

link = car;
car =
    RECORD
    passengers: integer;
    car_no: integer;
    capac: integer;
    next: link
    END;
    
```

Notice that the declaration for 'link' referred to 'car', BEFORE 'car' was even declared. This 'forward' referencing may seem strange, but that is the only way pointers are declared in Pascal. The RECORD 'car' is rather straightforward, with 3 integer fields and a link to the next car. Assuming that we keep only one chain of cars around in the program, we need to declare some variables to reference them. (Remember, the TYPE declarations simply define objects, and do not allocate any).

```

VAR
    head_ptr, tmp: link;
    
```

Here, we have actually set aside two pointers to reference the series of cars to be created later. The first one to point to the first car, and the second one as a handy temporary pointer to move down the series of cars to locate a specific one. To create new instances of 'car', Pascal's built-in function, NEW is required. The NEW function takes as its argument a pointer to the new object to be created. In the 'C' language ad PL/1, the equivalent function calls would be 'alloc'. Be sure that you do not call NEW with a pointer that points to something you want to save, since it will now point to the new object (probably filled with garbage), and your old object is lost forever, unless another pointer had saved its value.

To create a 'head-car', we will have this statement in Pascal:

```
new(head_ptr);
```

It is a good idea to explicitly initialize the fields in the 'car' record to known values:

```

head_ptr.passengers:=0;
head_ptr.car_no :=0;
head_ptr.capac :=0;
head_ptr.link :=nil;
    
```

Please take note of the last statement. The value 'nil' is a special value meant to represent 'nothing', or ground, if you wish. We are not allowed to access an object pointed to by 'nil'. Also of note in the above statements is the word 'car' is never mentioned. Since 'head\_ptr' is known to point to a 'car', Pascal is able to deduce where the fields come from.

Until next time, please think about how successive values of 'car' may be linked to the head of the chain by repeated calls to NEW>

\*\*\*\*\*

**ERRATA:** In last month's Bits & Bytes column the following lines should be changed:

```

LD HL,0FE80H should read: LD HL,0F080H
LD DE,0FE81H should read: LD DE,0F081H
    
```

We hope this hasn't caused anyone a lot of headaches.

\*\*\*\*\*

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.

<b>BACK ISSUES:</b>	ARESCO Source (issues 1-5)	\$ 8
	S.U.N. Volume I	SOLD OUT
	S.U.N. Volume II	\$10
	Sorcerer's Apprentice Vol I (1-7)	\$10
	Sorcerer's Apprentice Vol II (1-5)	\$10
	Sorcerer's Apprentice Vol III (1-8)	\$12
	S.A. Vol's I thru III (per issue)	\$ 2

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

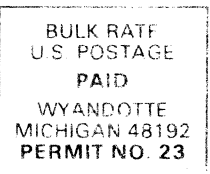
Make checks or money orders (only in US funds drawn on a US bank) payable to: **SORCERER'S APPRENTICE**.

**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.



RETURN AND FORWARDING POSTAGE GUARANTEED