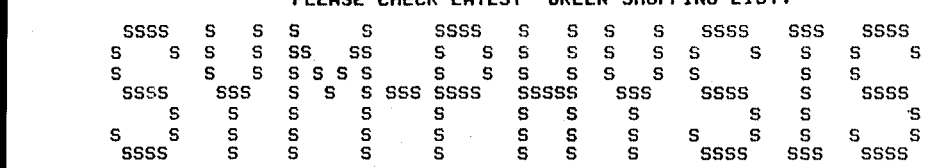
EDITED AND REPRINTED MAY 1981
MANY PRICES QUOTED IN THIS ISSUE ARE OBSOLETE,
PLEASE CHECK LATEST 'GREEN SHOPPING LIST.



THE SYM-1 USERS' GROUP NEWSLETTER

INTRODUCTORY ISSUE - SEPTEMBER 1979

SYM-PHYSIS is a bimonthly publication of the SYM Users' Group, P. O. Box 315, Chico, CA. 95927. SYM-PHYSIS and the SYM Users' Group (SUG) are in no way associated with Synertek Systems Corporation (SSC), and SSC has no responsibility for the contents of SYM-PHYSIS. SYM is a registered trademark of SSC. SYM-PHYSIS, from the Greek, means the state of growing together, to make grow, to bring forth.

Editor/Publisher: H. R. "Lux" Luxenberg
Business/Circulation: Jean Luxenberg

SUBSCRIPTION RATES:

USA/Canada $9.00 for a volume of 6 issues; overseas $12.50. Make checks payable in US dollars to "SYM Users' Group," P. O. Box 315, Chico, CA 95927, Telephone (916) 895-8751.

WHO, WHERE, WHY, WHAT, WHEN, HOW:

We (that's an editorial "we") teach computer science courses at California State University at Chico (about 90 miles north of Sacramento). The CSUC microcomputer laboratory facility was based on ten 6800-based single board micros. While these were adequate for an introduction to the subject, their versatility was far less than KIM's, whose hard copy outputs via an ASR 33 TTY could be submitted to an instructor for help in debugging, and for grading purposes.

More advanced students were using my personal KIM, for which I had developed mountains of utility, and other, software, and which had been expanded to include graphical (oscilloscope) and musical interfaces. My original KIM became dedicated, by default, to student use, and had to be replaced by a newer system, with more memory and I/O capability. One big problem with the KIM setup was the limited port capability. I had to settle for 6 bit D/A and A/D, rather than the 8 bits I wanted for music and graphics. Besides, KIM had to be depowered to permit interchange of my two output boards, and this slowed up demonstrations. I had learned earlier, the hard way, by burning up one of the port pins in a 6530 chip, not to change accessory boards with power on. My hardware requirements were certainly met by my new VIM (Versatile Interface Module, SYM's original name), but it took several months to convert the most important software from KIM to SYM. Some is still not yet converted.

I am certain that the existence of the KIM-1/6502 USER NOTES, now greatly expanded into the 6502 USER NOTES, and from which The First Book of KIM was born, and Robert Tripp's PLEASE package, and Peter Jennings' MICROCHESS helped to sell a lot of KIMs. Tripp's early work with the 6502 has expanded into the monthly publication MICRO. In the "early" days nearly every article in both the NOTES and MICRO were of help to a KIM user. Now only a small portion of the material is helpful to the SYM beginner (but very helpful). The remainder is of value to the advanced SYMMER; some of my best programs have been adapted from programs originally developed for APPLE, PET, KIM, OSI, etc. Now that I have RAE available, it will be easy to adapt 6800 programs, too. The "different" 6800 instructions can be treated as macros in RAE.

I believe there is a definite need for a SYM users information interchange. I am constantly impressed with just how powerful the SYM is with just the on-board expansion and an added terminal (which need not be that expensive); I would like to learn more from other users, and pass on some of what I have learned to others. That is one reason for SYM-PHYSIS. Another reason is to provide to beginners and old-timers alike a source of, and a directory to, immediately usable SYM software.

~~Enclosed in this introductory issue of SYM-PHYSIS is a subscription form for your use. We will hold all checks received for subscriptions until enough have been received to reach the break-even point. If the decision is GO, Volume 1, Number 1, dated Jan-Feb 1980, will be in your hands by December 15, 1979, in time for a Christmas gift!~~ Each issue will be at least the equivalent of 20 single spaced typewritten pages. That's a lot of software and documentation for the price!

BOOK RECOMMENDATIONS:

Camp, R. C., T. A. Smay, C. J. Triska: "Microcomputer Systems Principles-Featuring the 6502/KIM," Matrix Publishers, Inc., Portland, OR, 1978.

Camp, R. C., T. A. Smay, C. J. Triska: "Microprocessor Systems Engineering," Matrix Publishers, Inc., Portland, OR, 1979 (a revised version of the above, featuring the AIM).

Foster, Caxton C.: "Programming a Microcomputer-6502," Addison Wesley Publishing Co., Reading, MA, 1978.

Zaks, Rodnay: "Programming the 6502," SYBEX, Berkeley, CA, 1978.

Zaks, Rodnay: "6502 Applications Book," SYBEX, Berkeley, CA, 1979.

SYM-1 TAPE DIRECTORY

Here are three different types of listings for a modified version of the program of the same name by John Gierytic, in MICRO #8, pages 35-37. The original version used the 7-segment displays, was for the original SUPERMON, and was interrupted only by RESET. The revised version uses the terminal for display, is for SUPERMON Version 2, and can be interrupted by the terminal break key being held down until the end of the SYNCH search. It even starts and stops the recorder under computer control. It is fully relocatable and is callable as a subroutine or as a USR function from SYM BASIC or TINY BASIC and returns after terminal break. Those without a terminal may update the original version by noting the changed addresses for Version 2. There was also a typo in the original version. The correct address for SCAND should be 8906 (not 890B). It's handy to be able to review the ID numbers to refresh your memory; also since RAE, SYM BASIC, and TINY BASIC use different starting addresses for their files, you can even identify the type of file from its starting address.

```
>ASSEMBLE LIST     DIRECTORY - Source Code Version

                   0001             .OS
                   0010             .BA $F00
                   0020
                   0030 ;        DATA LOCATION DECLARATIONS
                   0040
                   0050 ID          .DE $9A
                   0060 SAL         .DE $9B
                   0070 SAH         .DE $9C
                   0080 EAL         .DE $9D
                   0090 EAH         .DE $9E
                   0100 MODE        .DE $FD
                   0130
                   0140 LATCHL      .DE $A004
                   0150
                   0160 ;        SYM MONITOR ROUTINES USED
                   0161
                   0170 ;     THOSE MARKED *NEW* ARE SUPERMON
                   0180 ;          VERSION 2 ADDRESSES
                   0189
                   0190 OUTBYT      .DE $82FA     ;PRINT A AS 2 HEX DIGITS
                   0200 COMMA       .DE $833A     ;PRINT A COMMA
                   0210 SPACE       .DE $8342     ;PRINT A SPACE
                   0220 CRLF        .DE $834D     ;PRINT <CR> <LF>
                   0230 OUTCHR      .DE $8A47     ;PRINT ASCII FROM A
                   0240 TSTAT       .DE $8B3C     ;CARRY SET IF BRK KEY DOWN
                   0250 ACCESS      .DE $8B86     ;UNWRITE PROTECT SYS RAM
                   0255 EX10        .DE $8D4E     ;*NEW*
                   0260 SYNC        .DE $8D52     ;*NEW*
                   0270 START       .DE $8DA9     ;*NEW*
                   0280 RDCHTX      .DE $8DE1     ;*NEW*
                   0290 RDBYTH      .DE $8DE5     ;*NEW*
                   0300 RDBYTX      .DE $8E26     ;*NEW*
                   0310
```



```
                   0320
                   0321 ;        BEGIN MAINLINE
                   0322
0F00- 20 86 8B     0330 BEGIN       JSR ACCESS
0F03- A0 80        0340             LDY #$80      ;SELECT HI SPEED MODE
0F05- 20 A9 8D     0350 NXT.FILE    JSR START     ;INIT TAPE ROUTINES
0F08- A9 1F        0360             LDA #$1F      ;SET UP TIMER
0F0A- 8D 04 A0     0370             STA LATCHL
0F0D- 20 52 8D     0380 FIND        JSR SYNC      ;SEARCH TAPE FOR RECORD
0F10- 20 E1 8D     0390 READ        JSR RDCHTX    ;GET CHARACTER
0F13- C9 2A        0400             CMP #'*
0F15- F0 06        0410             BEQ TEST
0F17- C9 16        0420             CMP #$16      ;SYNC CHARACTER?
0F19- D0 F2        0430             BNE FIND
0F1B- F0 F3        0440             BEQ READ
0F1D- A5 FD        0450 TEST        LDA *MODE
0F1F- 29 BF        0460             AND #$BF
0F21- 85 FD        0470             STA *MODE
0F23- 20 26 8E     0480             JSR RDBYTX
0F26- 85 9A        0490             STA *ID
0F28- 20 26 8E     0500             JSR RDBYTX    ;GET SAL FROM TAPE
0F2B- 85 9B        0510             STA *SAL
0F2D- 20 26 8E     0520             JSR RDBYTX    ;GET SAH FROM TAPE
0F30- 85 9C        0530             STA *SAH
0F32- 20 E5 8D     0540             JSR RDBYTH    ;GET EAL
0F35- 85 9D        0550             STA *EAL
0F37- 20 E5 8D     0560             JSR RDBYTH    ;GET EAH
0F3A- 85 9E        0570             STA *EAH
0F3C- 20 4D 83     0580             JSR CRLF
0F3F- A9 2E        0590             LDA #'.
0F41- 20 47 8A     0600             JSR OUTCHR
0F44- A9 53        0610             LDA #'S
0F46- 20 47 8A     0620             JSR OUTCHR
0F49- A9 32        0630             LDA #'2
0F4B- 20 47 8A     0640             JSR OUTCHR
0F4E- 20 42 83     0650             JSR SPACE
0F51- A5 9A        0660             LDA *ID       ;PRINT THE
0F53- 20 FA 82     0670             JSR OUTBYT    ;  ID NUMBER
0F56- 20 3A 83     0680             JSR COMMA
0F59- A5 9C        0690             LDA *SAH      ;PRINT
0F5B- 20 FA 82     0700             JSR OUTBYT    ;  THE
0F5E- A5 9B        0710             LDA *SAL      ;    STARTING
0F60- 20 FA 82     0720             JSR OUTBYT    ;      ADDRESS
0F63- 20 3A 83     0730             JSR COMMA
0F66- C6 9D        0740             DEC *EAL      ;DECREMENT
0F68- A5 9D        0750             LDA *EAL      ;  THE
0F6A- C9 FF        0760             CMP #$FF      ;    ENDING
0F6C- D0 02        0770             BNE CONT      ;      ADDRESS
0F6E- C6 9E        0780             DEC *EAH
0F70- A5 9E        0790 CONT        LDA *EAH      ;PRINT
0F72- 20 FA 82     0800             JSR OUTBYT    ;  THE
0F75- A5 9D        0810             LDA *EAL      ;    ENDING
0F77- 20 FA 82     0820             JSR OUTBYT    ;      ADDRESS
0F7A- 20 4D 83     0830             JSR CRLF
```

```
0F7D- 20 3C 8B  0840            JSR TSTAT     ;BREAK KEY DOWN?
0F80- B0 02     0850            BCS RETURN    ;IF SO, STOP
0F82- 90 81     0860            BCC NXT.FILE
0F84- 4C 4E 8D  0870 RETURN     JMP EX10      ;STOP TAPE AND RETURN
                0880            .EN
```

LABEL FILE: [/ = EXTERNAL]

```
/ID=009A                /SAL=009B               /SAH=009C
/EAL=009D               /EAH=009E               /MODE=00FD
/LATCHL=A004            /OUTBYT=82FA            /COMMA=833A
/SPACE=8342             /CRLF=834D              /OUTCHR=8A47
/TSTAT=8B3C             /ACCESS=8B86            /EX10=8D4E
/SYNC=8D52              /START=8DA9             /RDCHTX=8DE1
/RDBYTH=8DE5            /RDBYTX=8E26            BEGIN=0F00
NXT.FILE=0F05           FIND=0F0D               READ=0F10
TEST=0F1D               CONT=0F70               RETURN=0F84
```

DIRECTORY - Disassembled Version

```
0F00-  20 86 8B   JSR  8B86
0F03-  A0 80      LDY  #80
0F05-  20 A9 8D   JSR  8DA9
0F08-  A9 1F      LDA  #1F
0F0A-  8D 04 A0   STA  A004
0F0D-  20 52 8D   JSR  8D52
0F10-  20 E1 8D   JSR  8DE1
0F13-  C9 2A      CMP  #2A
0F15-  F0 06      BEQ  0F1D
0F17-  C9 16      CMP  #16
0F19-  D0 F2      BNE  0F0D
0F1B-  F0 F3      BEQ  0F10
0F1D-  A5 FD      LDA  FD
0F1F-  29 BF      AND  #BF
0F21-  85 FD      STA  FD
0F23-  20 26 8E   JSR  8E26
0F26-  85 9A      STA  9A
0F28-  20 26 8E   JSR  8E26
0F2B-  85 9B      STA  9B
0F2D-  20 26 8E   JSR  8E26
0F30-  85 9C      STA  9C
0F32-  20 E5 8D   JSR  8DE5
0F35-  85 9D      STA  9D
0F37-  20 E5 8D   JSR  8DE5
0F3A-  85 9E      STA  9E
0F3C-  20 4D 83   JSR  834D
0F3F-  A9 2E      LDA  #2E
0F41-  20 47 8A   JSR  8A47
0F44-  A9 53      LDA  #53
0F46-  20 47 8A   JSR  8A47
0F49-  A9 32      LDA  #32
0F4B-  20 47 8A   JSR  8A47
0F4E-  20 42 83   JSR  8342
0F51-  A5 9A      LDA  9A
0F53-  20 FA 82   JSR  82FA
0F56-  20 3A 83   JSR  833A
0F59-  A5 9C      LDA  9C
0F5B-  20 FA 82   JSR  82FA
0F5E-  A5 9B      LDA  9B
0F60-  20 FA 82   JSR  82FA
0F63-  20 3A 83   JSR  833A
0F66-  C6 9D      DEC  9D
0F68-  A5 9D      LDA  9D
0F6A-  C9 FF      CMP  #FF
0F6C-  D0 02      BNE  0F70
0F6E-  C6 9E      DEC  9E
0F70-  A5 9E      LDA  9E
0F72-  20 FA 82   JSR  82FA
0F75-  A5 9D      LDA  9D
0F77-  20 FA 82   JSR  82FA
0F7A-  20 4D 83   JSR  834D
0F7D-  20 3C 8B   JSR  8B3C
0F80-  B0 02      BCS  0F84
0F82-  90 81      BCC  0F05
0F84-  4C 4E 8D   JMP  8D4E
```

DIRECTORY - ObJect Code

```
.V 0F00,0F86 with Checksum
0F00 20 86 8B A0 80 20 A9 8D,A7
0F08 A9 1F 8D 04 A0 20 52 8D,9F
0F10 20 E1 8D C9 2A F0 06 C9,DF
0F18 16 D0 F2 F0 F3 A5 FD 29,65
0F20 BF 85 FD 20 26 8E 85 9A,99
0F28 20 26 8E 85 9B 20 26 8E,61
0F30 85 9C 20 E5 8D 85 9D 20,56
0F38 E5 8D 85 9E 20 4D 83 A9,84
0F40 2E 20 47 8A A9 53 20 47,06
0F48 8A A9 32 20 47 8A 20 42,BE
0F50 83 A5 9A 20 FA 82 20 3A,76
0F58 83 A5 9C 20 FA 82 A5 9B,16
0F60 20 FA 82 20 3A 83 C6 9D,F2
0F68 A5 9D C9 FF D0 02 C6 9E,32
0F70 A5 9E 20 FA 82 A5 9D 20,73
0F78 FA 82 20 4D 83 20 3C 8B,C6
0F80 B0 02 90 81 4C 4E 8D,B0
 40B0
```


SOFTWARE RECOMMENDATION: The SYM/KIM Appendix to The First Book of KIM

For beginning SYMMERS, or those with absolutely no extensions to their SYMs, I strongly recommend "The SYM/KIM Appendix to 'The First Book of KIM'," by Robert A. Peck, P. O. Box 2231, Sunnyvale, CA 94087.(The FBOK was edited by F. J. Butterfield.) Mr. Peck has provided complete SYM modifications for all of the programs in the Games and Diversions section of FBOK except CLOCK and TIMER (which are better handled using the the 6522 on-board timers), and MUSIC BOX (which does require a cassette with monitor feature or an external speaker). When I first got my KIM, my favorite computer recreations were CRAPS, BLACKJACK and WUMPUS from FBOK. Thanks to the "Appendix," I now have WUMPUS and BLACKJACK back again on the SYM. One day soon, I'll have my KIM dump all of its FBOK programs in KIM format rather than Butterfields' "Hypertape" (see FBOK for this), and get them running on SYM thanks to the Appendix. Price for the Appendix alone is* for the Appendix plus FBOK is for the FBOK alone is California sales tax, post paid, from Mr. Peck. All programs will run with either SUPERMON or SUPERMON Version 2. *SEE BELOW

For a general discussion of KIM/SYM conversions see "The First Book of KIM-on a SYM" by Nicholas Vrtis, MICRO #14, pages 35-37.

For a discussion of the use of the 6522 Timer, see "SYM-1 6522-Based Timer" by John Giervic, MICRO #11, pages 31-32.

For an alternate set of modifications for the FBOK WUMPUS program and for mods to FBOK MUSIC BOX see "Wumpus and Music Box Mods for SYM," by Jim Adams, 6502 USER NOTES #14, page 20.

RAE NOTES:

For the past several months two of us here have been working with a pre-release version of Synertek's Resident Assembler Editor, RAE. Our version was on cassette; the production versions will be available in both a single 8K ROM chip, and a pair of 4K ROMs. The specs for RAE have been widely publicized by Synertek Systems, so we won't repeat them here. RAE is a full features assembler, with macro capability, conditional assembly features, and a relocating loader patch in RAM. It has been a real pleasure to give up "hand-assembly", not because it was so hard to enter the initial version of a program, but because by the time you made a half-dozen or so modifications, involving deletions, insertions, relocations, etc., the original documentation is hopelessly out of date. Some of the programs which you will see in issues of SYM-PHYSIS do not exist anywhere in source code form, and are available only in the dis-assembled form. Incidentally, we are working on a "symbolic disassembler" for SYM which will insert (non-mnemonic) labels for variables and referenced lines. This will permit easy mods; can't estimate its completion date as yet! As an example of the text editing capabilities of RAE, we present this edition of SYM-PHYSIS.We will have some suggestions on how to use RAE more effectively in our next issue, when more of you have installed RAE in your system.

* APPENDIX $4.75, FBOK 11.00,
APPENDIX WITH FBOK $14.50
FOR OVERSEAS DELIVERY ADD $4.00 U.S.FUNDS FOR FBOK OR FBOK & APPENDIX
OR FOR APPENDIX ALONE ADD $1.50 U.S.FUNDS
CALIFORNIA SALES TAX IS APPLICABLE TO CALIF. RESIDENTS ONLY.

RAE NOTES (continued):

The pseudo-ops used in RAE-1 differ from those used in the MOS Technology/System 65 assemblers. As an aid in converting source codes from one format to another, the following example illustrates how the listing for SUPERMON Version 2 would look in RAE format. Compare it with the listing given in the third printing (June 1979) of the SYM-1 Reference Manual. There are two other differences: L,LABEL and H,LABEL in RAE (see lines 340 and 360 in "Terminal Control Patch for BASIC") translate to <LABEL and >LABEL, respectively.

ASSEMBLE LIST

```
                0010 ;EXAMPLE TO ILLUSTRATE HOW
                0020 ;SUPERMON VERSION 2 WOULD
                0030 ;BE ASSEMBLED USING RAE
                0040 ;
                0050            .BA $A600
A600-           0060 SCPBUF     .DS $20
                0070 RAM        .DI =
A620-           0080 JTABLE     .DS $10
A630-           0090 TAPDEL     .DS 1
A631-           0100 KMBDRY     .DS 1
                0110 ;LEAVE A GAP HERE FOR CON-
                0120 ;VENIENCE IN ILLUSTRATION
                0130            .BA $A63D
A63D-           0140 SCRD       .DS 1
                0150 RC         .DI SCRD
A63E-           0160 SCRE       .DS 1
A63F-           0170 SCRF       .DS 1
A640-           0180 DISBUF     .DS 5
A645-           0190 RDIG       .DS 1
A646-           0200            .DS 3
A649-           0210 PARN       .DS 1
                0220 ;LEAVE A GAP
                0230            .BA $A680
                0240 PADA       .DE $A400
                0250 PBDA       .DE $A402
                0260            .BA $8000
8000- 4C 7C 8B  0270 MONITR     JMP MONENT
8003- 20 FF 80  0280 WARM       JSR GETCOM
8006- 20 4A 81  0290            JSR DISPAT
8009- 20 71 81  0300            JSR ERMSG
800C- 4C 03 80  0310            JMP WARM
                0320 ;LEAVE A GAP
                0330            .BA $8147
8147- FF FF FF  0340            .BY $FF $FF $FF
814A- C9 0D     0350 DISPAT     CMP #$0D
                0360 ;LEAVE A GAP
                0370 ;ZERO PAGE ADDRESSING
                0380            .BA $829F
829F- 86 FF     0390            STX *$FF
                0400 ;LEAVE A LONG GAP
                0410            .BA $8FA0
```



```
                0420 DFTBLK     .DI =
8FA0- 00 C0     0430            .SE $C000
8FA2- A7 8B     0440            .SI TTY
8FA4- 64 8B     0450            .SI NEWDEV
8FA6- 00 00     0460            .SE $0000
8FA8- 00 02     0470            .SE $0200
                0480 ;DEFINE SELECTED INTERNAL
                0490 ;LABELS TO PERMIT ASSEMBLY
                0500 ;OF THIS EXAMPLE
                0510 MONENT     .DI $8B7C
                0520 GETCOM     .DI $80FF
                0530 ERMSG      .DI $8171
                0540 TTY        .DI $8BA7
                0550 NEWDEV     .DI $8B64
                0560 ;END OF EXAMPLE
                0570 ;NOTE THAT THE "RELATIVE
                0580 ;ADDRESSES" GIVEN AFTER
                0590 ;THE ERROR COUNT BELOW ARE
                0600 ;RELATIVE TO THE LAST .BA
                0610 ;IN THE SOURCE CODE
                0620            .EN
```

LABEL FILE: [/ = EXTERNAL]

```
SCPBUF=A600     RAM=A620        JTABLE=A620
TAPDEL=A630     KMBDRY=A631     SCRD=A63D
RC=A63D         SCRE=A63E       SCRF=A63F
DISBUF=A640     RDIG=A645       PARN=A649
/PADA=A400      /PBDA=A402      MONITR=8000
WARM=8003       DISPAT=814A     DFTBLK=8FA0
MONENT=8B7C     GETCOM=80FF     ERMSG=8171
TTY=8BA7        NEWDEV=8B64
//0000,8FAA,8FAA
```

Note that the SUAM II Version of the 6502 assembler used by Mr. Vrtis for his HEX PROGRAM VERIFY PROGRAM uses a still different set of pseudo-opcodes. It is left as an exercise for the reader to provide the necessary translation for SUAM II!

SOFTWARE RECOMMENDATIONS: TINY BASIC

We find Tom Pittman's TINY BASIC for the 6502 extremely useful in SYM, even though we also have SYM BASIC (BAS-1). Because TINY is in RAM (0200-0AC6) it is easy to set at and customize. TINY BASIC is available from~~*Itty Bitty Computers, P. O. Box 23189, San Jose, CA, 95153, for $5.00. We recommend you also get the TB Experimenter's Kit at the same time, for an additional $10.00 (Californians add $.65 for sales tax). We here we can make arrangements with Mr. Pittman to distribute a SYM readable cassette with enhancements, since he offers only a punched paper tape version.~~

~~*~~ the SYM USERS' GROUP with cassette.

See Issue #2-27

MERGE, RENUMBER, and FIND for SYM:

In "Inside PET BASIC," by Jim Butterfield, MICRO #8, pages 39-41, appear three useful utilities for PET. These are UNLIST (a procedure for merging programs), FIND, and RESEQUENCE (actually RENUMBER).

UNLIST is not even indirectly adaptable to SYM because it makes use of certain commands and procedures not found in SYM BASIC, e.g., OPEN, CLOSE, CMD, PRINT #, etc. In the next issue we will publish one of several of the MERGE, DELETE, APPEND, etc., programs we have worked out for SYM. We have tested these out (and they work fine!) as extensions of the Terminal Control Patch published elsewhere in this issue; we still want to try out and compare three alternate approaches to patching. One is to call through USR. Two is to trap the SAVE and LOAD calls (this will permit us to "name" our files (a la PET). Three is to trap the SN and FC errors (if we can) to permit us to add new commands to SYM BASIC. Haven't found out how to do this latter; we're still trying to implement the GET instruction which is available as a token.

RESEQUENCE (RENUMBER) and FIND, however, are easily modified for SYM, by changing the tokens, and the PEEK AND POKE locations, and correcting the typographical errors in lines 60010, 60240, and 60250 in the original. We have modified RENUMBER to ask for the start and step values (together with a few other minor mods), and relocated FIND. Until you have MERGE available these must be entered from tape previous to starting any program development, and, until you have DELETE available, you must, on a line-by-line basis, wipe these out or they will be SAVEd with your final program.

To use RENUMBER, enter GOTO 60000 as a direct command. It should be noted that in addition to GET mentioned above, GO is also a reserved word, i.e., a token. SYM BASIC will treat GO TO and GOTO as equivalent, although the first is stored as C5 20 9E, and the second as 88. FIND and RENUMBER will not; RENUMBER will work correctly only with GOTO in the BASIC program. RENUMBER is very slow; and should eventually be replaced by a machine language version (see "The Ultimate PET Renumber," by Don Rindsberg, MICRO #11, pages 37-47, for an example of how this might be done). We haven't pried sufficiently into SYM BASIC to do so, however, nor do we personally intend to do so. Our approach will be to pass the programs to RAE, which is written in machine language, and patch RAE to do the editing, resequencing, etc.

Because RENUMBER is so slow on long programs, and appears to be doing nothing, we added the instruction at 60185 so we could be kept posted on its progress. This should be deleted for a hard copy terminal to save paper.

To use FIND to locate any string, enter the string at line 0 with no space, e.g., 0COS, 0POKE, 0 Q$, etc., then enter GOTO 60500 as a direct command. If you wish to be able to find all references to a variable, for example, E%, it is necessary to have it enclosed by spaces everywhere it is used in the program, and also in the dummy instruction at 0. Otherwise you will have too many "false-alarms."

SYM BASIC RENUMBER/FIND

```
60000 CLEAR:INPUT"Start, Step?";M1,IN:PRINT:PRINT:REM RESEQ 8/5/79
60010 T=0:DIMV%(99),W%(99):GOSUB60160:FORR=1TO1E3:GOSUB60210
60020 IFGTHENGOSUB60090:NEXTR
60030 GOSUB60160:FORR=1TO1E3:N=INT(M/256):POKEA-1,M-N*256
60040 POKEA,N:V=L:GOSUB60070:W%(J)=M:GOSUB60170:IFGTHENNEXTR
60050 GOSUB60160:FORR=1TO1E3:GOSUB60210:IFGTHENGOSUB60110:NEXTR
60060 PRINT"Finished":END
60070 J=0:IFT<>0THENFORJ=1TOT:IFV%(J)<>VTHENNEXTJ:J=0
60080 RETURN
60090 IFV<>0THENGOSUB60070:IFJ=0THENT=T+1:V%(T)=V
60100 RETURN
60110 GOSUB60070:IFJ=0THENRETURN
60120 W=W%(J):IFW=0THENPRINT"Insert ???? in line";L:RETURN
60130 FORD=ATOB+1STEP-1:X=INT(W/10):Y=W-10*X+48:IFW=0THENY=32
60140 POKED,Y:W=X:NEXTD:IFW=0THENRETURN
60150 PRINT"Insert";W%(J);"in line";L:RETURN
60160 F=513:M=M1-IN
60170 A=F:M=M+IN
60180 F=PEEK(A)+PEEK(A+1)*256:L=PEEK(A+2)+PEEK(A+3)*256:A=A+3:G=L<6E4
60185 PRINTL,G
60190 RETURN
60200 S=0
60210 V=0:A=A+1:B=A:C=PEEK(A):IFC=0THENGOSUB60170:ONG+2GOTO60210,60190
60220 IFC<>136ANDC<>140ANDC<>161ANDC<>SGOTO60200
60230 A=A+1:C=PEEK(A)-48:IFC=-16GOTO60230
60240 IFC>=0ANDC<=9THENV=V*10+C:GOTO60230
60250 S=44:A=A-1:RETURN
60500 A=513:X=PEEK(517):REM FIND 4/6/79
60510 FORK=A+4TOA+83
60520 P=PEEK(K):IFP=XTHENGOSUB60570
60530 IFP<>0THENNEXTK
60540 A=PEEK(A)+PEEK(A+1)*256:Z=PEEK(A+2)+PEEK(A+3)*256
60550 IFA<>0ANDZ<6E4THEN60510
60560 PRINT"Finished":END
60570 FORL=1TO80:Y=PEEK(517+L):IFY<>0THEN60590
60580 PRINTZ:RETURN
60590 IFY=PEEK(K+L)THENNEXTL
60600 RETURN
```

PERIODICAL RECOMMENDATIONS:

Here are some periodicals which will be especially helpful to SYMMERS:

MICRO-The Magazine of the APPLE, KIM, PET and OTHER 6502 Systems; P.O. Box 6502, Chelmsford, MA 01824. Monthly, $15.00 per year/12 issues.

6502 USER NOTES:* ~~P. O. Box 33093, N. Royalton, OH 44133. Bimonthly, $13.00 per volume/6 issues.~~

*No longer published, but back issues still available. See Issue #2-2 under Periodical Recommendations.

HEX PROGRAM VERIFY PROGRAM

Did you ever find a handy program in a newsletter or magazine, key it in, and wonder how many keying mistakes you are going to have to find? I purchased the hex dump version of Tom Pittman's "TINY BASIC", and was faced with over 3K of code to enter, and no check digits to tell me if I got it in right. I decided to borrow a technique used in most data entry shops and verify the code by keying the same thing twice. The theory is that if you key the same thing twice, chances are that you won't make the same mistake twice in the same place. The theory works. I entered TINY in,verified it (catching a half dozen mistakes) and it worked. The process doesn't take twice as long to do, since you can go a little faster because you know that mistakes will be caught later.

The program logic is relatively simple. It displays the starting address on a line, and accepts input for the number of bytes specified at $02, without displaying them first. The value entered is compared to the existing data, and if they are equal the current address is incremented, and either another byte is accepted, or a new line is started. The only tricky parts come if a non-hex digit is entered, or the two values don't match. If the non-hex character is a carriage return, INBYTE sets the equal flag, and the program starts over on a new line. This is handy if your listing happens to have unequal length lines for some reason. If the non-hex wasn't a carriage return, then it was just a typo, and the program outputs one or two backspaces depending upon whether the first or second character was the non-hex. The puts the cursor back to the start of the entry for that byte.

When the input byte and the existing byte don't match, there really is no way for the program to know which is the correct one. The solution is the beep the beeper, tell you what the existing value is, and accept a replacement value from the terminal. The existing value is displayed to help resolve the difference, in case you are off a column or row in you listing. The cursor is also backspaced so that all the keying occurs in the same place on the screen. This keeps the screen looking like the listing. If any non-hex character is entered as part of the replacement byte, the program figures you and/or it are confused, and starts over on the same byte.

If you don't have a CRT that responds to the backspace character, you may want to NOP most of the code that does the backspacing. I would suggest that you at least output some sort of character so that you can look back and see what happened.

Operation of the program is simple. Locations $00 and $01 are used to contain the current address of the program that is being verified. Do an SD command to store the starting address there, and start the verify program. It is completely relocatable, so it can be put most anywhere that the program to be verified isn't. As I mentioned earlier, location $02 contains the number of bytes input before a new line is started. The purpose of the new line is to display a reference address so you can check where you are at in the listing.

```
SUAM - V2.1 (03/79)                    6502 ASSEMBLER - SUAM II VERSION

 LOC    ----OBJECT----   STMT          H E X  P R O G R A M  V E R I F Y  P R O G R A M

0000   00  02            00004 CURAD   DCA   2,$200          CURRENT ADDRESS AREA
0002   10                00005 PERLINE DC    1,16            BYTES PER LINE

0003   20  4D  83        00007 NEWLINE JSR   CRLF            START OF A NEW LINE
0006   A6  01            00008         LDX   CURAD+1         GET CURRENT ADDRESS
0008   A5  00            00009         LDA   CURAD
000A   20  F4  82        00010         JSR   OUTXAH          AND OUTPUT IT
000D   A6  02            00011         LDX   PERLINE         SET # OF BYTES/LINE

000F   20  42  83        00013 GETLOOP JSR   SPACE           LEADING SPACE OF READABILITY
0012   20  D9  81        00014 GETCHR  JSR   INBYTE          GET 2 HEX DIGITS = 1 BYTE
0015   90  0E            00015         BCC   TWOHEX          BRANCH IF BOTH WERE HEX
0017   F0  EA            00016 NOTHEX  BEQ   NEWLINE        C/R MEANS HE WANTS TO START NEW LINE
0019   A9  08            00017         LDA   #$08            ELSE MOVE CURSOR BACK
001B   50  03            00018         BVC   *+5             ONLY ONCE IF 1ST DIGIT WAS NON-HEX
001D   20  47  8A        00019         JSR   OUTCHR          TWICE IF IT WAS THE SECOND DIGIT
0020   20  47  8A        00020         JSR   OUTCHR
0023   D0  EF            00021         BNE   GETCHR          UNCONDITIONAL - GET ANOTHER TRY

0025   A0  00            00023 TWOHEX  LDY   #0              SET 'Y' TO ZERO
0027   D1  00            00024         CMP   (CURAD),Y       COMPARE INPUT AGAINST EXISTING
0029   F0  1F            00025         BEQ   NEXTONE         EQUAL IS SUPER
002B   20  72  8B        00026         JSR   BEEP            ELSE BEEP THE BEEPER FROGIE
002E   A9  08            00027         LDA   #$08            AND BACKUP 2 PLACES
0030   20  47  8A        00028         JSR   OUTCHR
0033   20  47  8A        00029         JSR   OUTCHR
0036   B1  00            00030         LDA   (CURAD),Y       GET EXISTING VALUE
0038   20  FA  82        00031         JSR   OUTBYT          OUTPUT IT AS HEX VALUES
003B   A9  08            00032         LDA   #$08            HERE WE GO BACKING UP AGAIN
003D   20  47  8A        00033         JSR   OUTCHR          SO HE WILL OVERTYPE
0040   20  47  8A        00034         JSR   OUTCHR
0043   20  D9  81        00035         JSR   INBYTE          NOW GET THE CORRECT VALUE
0046   B0  CF            00036         BCS   NOTHEX          CONFUSED -- START THIS BYTE AGAIN
0048   91  00            00037         STA   (CURAD),Y       OTHERWISE STORE THE RIGHT VALUE

004A   E6  00            00039 NEXTONE INC   CURAD          BUMP CURRENT ADDRESS
004C   D0  02            00040         BNE   *+4
004E   E6  01            00041         INC   CURAD+1
0050   CA                00042         DEX
0051   F0  B0            00043         BEQ   NEWLINE         COUNT THE BYTES ON THIS LINE
0053   D0  BA            00044         BNE   GETLOOP         START NEW LINE IF DONE
                                                             ELSE GO GET NEXT BYTE INPUT
                         00046 ********************************************************************
                         00047 * SYM-1 SYSTEM MONITOR ADDRESS REFERENCES                          *
                         00048 ********************************************************************
81D9                     00049 INBYTE  EQU   $81D9           INPUT 2 HEX DIGITS AS ONE BYTE
82F4                     00050 OUTXAH  EQU   $82F4           OUTPUT X & A AS 4 HEX DIGITS
82FA                     00051 OUTBYT  EQU   $82FA           OUTPUT A AS 2 HEX DIGITS
8342                     00052 SPACE   EQU   $8342           OUPUT ONE SPACE
834D                     00053 CRLF    EQU   $834D           OUTPUT CARRIAGE RETURN / LINE FEED
8972                     00054 BEEP    EQU   $8972           BEEP THE ONBOARD BEEPER
8A47                     00055 OUTCHR  EQU   $8A47           OUTPUT A AS ASCII CHARACTER
END OF PASS 2-ERRORS=    00000 ********************************************************************
 LOC    ----OBJECT----   STMT  SYMBOL TABLE PRINT

8972                     00054 BEEP
8342                     00053 CRLF
0000                     00004 CURAD
0012                     00014 GETCHR
000F                     00013 GETLOOP
81D9                     00049 INBYTE
0003                     00007 NEWLINE
004A                     00039 NEXTONE
0017                     00016 NOTHEX
82FA                     00051 OUTBYT
8A47                     00055 OUTCHR
82F4                     00050 OUTXAH
0002                     00005 PERLINE
8342                     00052 SPACE
0025                     00023 TWOHEX
```



```
**********  HEX PROGRAM VERIFY PROGRAM  **********
:150000000210204D53A601A5002DF482A6022D428320D9810700
:150015900EF0EAA9085003204784A20478AD0EDAA000D100F0D9A6
:15002A1F207289A9082047BA20478AB1C020FA82A90820470771
:15003FBA20478A20D981B0CF910DE6000002E601CAF0B00000B32
:010054BA010F
```

The above program contributed by Nicholas J. Vrtis, 5863 Pinetree S.E., Kentwood, MI 49508 (self-addressed stamped envelope with all correspondence).

DISASSEMBLER

To use DISASSEMBLER enter SAL and SAH of the program to be disassembled at $F0, $F1. At $F2 enter $3F for 66 line/page printing terminals, or $16 for 24 line/screen CRT terminals. Start DISASSEMBLER at $2000. After each halt, restart with a G and RETURN. To relocate the program a whole number of pages change only the high order addresses 20 and 21 to the new page numbers. The extra AAs are merely space fillers.

This version of DISASSEMBLER for the SYM is based upon the KIM version in 6502 USER NOTES #14, page 4, by Bob Kurtz and Eric Rehnke, which in turn was based upon an earlier version by Steve Wozniak and Allen Baum, published in Doctor Dobbs' Journal, Sept 1976. Page zero addresses do not conflict with SYM BASIC.

```
.M F0
00F0,24,00
00F1,21,20
00F2,FF,
.G 2000
```

```
2000-  20 06 20   JSR  2006
2003-  20 35 80   JSR  8035
2006-  A5 F2      LDA  F2
2008-  85 F8      STA  F8
200A-  20 1A 20   JSR  201A
200D-  20 F5 20   JSR  20F5
2010-  85 F0      STA  F0
2012-  84 F1      STY  F1
2014-  C6 F8      DEC  F8
2016-  D0 F2      BNE  200A
2018-  F0 E9      BEQ  2003
201A-  20 DB 20   JSR  20DB
201D-  A1 F0      LDA  (F0,X)
201F-  A8         TAY
2020-  4A         LSR  A
2021-  90 0B      BCC  202E
2023-  4A         LSR  A
2024-  B0 17      BCS  203D
2026-  C9 22      CMP  #22
2028-  F0 13      BEQ  203D
202A-  29 07      AND  #07
202C-  09 80      ORA  #80
202E-  4A         LSR  A
202F-  AA         TAX
2030-  BD 22 21   LDA  2122,X
2033-  B0 04      BCS  2039
2035-  4A         LSR  A
2036-  4A         LSR  A
2037-  4A         LSR  A
2038-  4A         LSR  A
2039-  29 0F      AND  #0F
203B-  D0 04      BNE  2041
203D-  A0 80      LDY  #80
203F-  A9 00      LDA  #00
2041-  AA         TAX
2042-  BD 66 21   LDA  2166,X
2045-  85 F3      STA  F3
2047-  29 03      AND  #03
2049-  85 F4      STA  F4
204B-  98         TYA
204C-  29 8F      AND  #8F
204E-  AA         TAX
204F-  98         TYA
2050-  A0 03      LDY  #03
2052-  E0 8A      CPX  #8A
2054-  F0 0B      BEQ  2061
2056-  4A         LSR  A
2057-  90 08      BCC  2061
2059-  4A         LSR  A
205A-  4A         LSR  A
205B-  09 20      ORA  #20
205D-  88         DEY
205E-  D0 FA      BNE  205A
2060-  C8         INY
2061-  88         DEY
2062-  D0 F2      BNE  2056
2064-  48         PHA
2065-  B1 F0      LDA  (F0),Y
2067-  20 0C 21   JSR  210C
206A-  A2 01      LDX  #01
206C-  20 EC 20   JSR  20EC
206F-  C4 F4      CPY  F4
2071-  C8         INY
2072-  90 F1      BCC  2065
2074-  A2 03      LDX  #03
2076-  C0 04      CPY  #04
2078-  90 F2      BCC  206C
207A-  68         PLA
207B-  A8         TAY
207C-  B9 80 21   LDA  2180,Y
207F-  85 F5      STA  F5
2081-  B9 C0 21   LDA  21C0,Y
2084-  85 F6      STA  F6
2086-  A9 00      LDA  #00
2088-  A0 05      LDY  #05
```



```
208A-  06 F6      ASL  F6
208C-  26 F5      ROL  F5
208E-  2A         ROL  A
208F-  88         DEY
2090-  D0 F8      BNE  208A
2092-  69 3F      ADC  #3F
2094-  20 04 21   JSR  2104
2097-  CA         DEX
2098-  D0 EC      BNE  2086
209A-  20 EA 20   JSR  20EA
209D-  A2 06      LDX  #06
209F-  E0 03      CPX  #03
20A1-  D0 12      BNE  20B5
20A3-  A4 F4      LDY  F4
20A5-  F0 0E      BEQ  20B5
20A7-  A5 F3      LDA  F3
20A9-  C9 E8      CMP  #E8
20AB-  B1 F0      LDA  (F0),Y
20AD-  B0 1C      BCS  20CB
20AF-  20 0C 21   JSR  210C
20B2-  88         DEY
20B3-  D0 F2      BNE  20A7
20B5-  06 F3      ASL  F3
20B7-  90 0E      BCC  20C7
20B9-  BD 73 21   LDA  2173,X
20BC-  20 04 21   JSR  2104
20BF-  BD 79 21   LDA  2179,X
20C2-  F0 03      BEQ  20C7
20C4-  20 04 21   JSR  2104
20C7-  CA         DEX
20C8-  D0 D5      BNE  209F
20CA-  60         RTS
20CB-  20 F8 20   JSR  20F8
20CE-  AA         TAX
20CF-  E8         INX
20D0-  D0 01      BNE  20D3
20D2-  C8         INY
20D3-  98         TYA
20D4-  20 0C 21   JSR  210C
20D7-  8A         TXA
20D8-  4C 0C 21   JMP  210C
20DB-  20 4D 83   JSR  834D
20DE-  A5 F1      LDA  F1
20E0-  A6 F0      LDX  F0
20E2-  20 D4 20   JSR  20D4
20E5-  A9 2D      LDA  #2D
20E7-  20 04 21   JSR  2104
20EA-  A2 03      LDX  #03
20EC-  A9 20      LDA  #20
20EE-  20 04 21   JSR  2104
20F1-  CA         DEX
20F2-  D0 F8      BNE  20EC
20F4-  60         RTS
20F5-  A5 F4      LDA  F4
20F7-  38         SEC
20F8-  A4 F1      LDY  F1
20FA-  AA         TAX
20FB-  10 01      BPL  20FE
20FD-  88         DEY
20FE-  65 F0      ADC  F0
2100-  90 01      BCC  2103
2102-  C8         INY
2103-  60         RTS
2104-  84 F7      STY  F7
2106-  20 47 8A   JSR  8A47
2109-  A4 F7      LDY  F7
210B-  60         RTS
210C-  84 F7      STY  F7
210E-  20 FA 82   JSR  82FA
2111-  A4 F7      LDY  F7
2113-  20 3C 8B   JSR  8B3C
2116-  90 01      BCC  2119
2118-  00         BRK
2119-  60         RTS
211A-  AA         TAX
211B-  AA         TAX
211C-  AA         TAX
211D-  AA         TAX
211E-  AA         TAX
211F-  AA         TAX
2120 AA AA 40 02 45 03 D0 08,B6
2128 40 09 30 22 45 33 D0 08,A1
2130 40 09 40 02 45 33 D0 08,7C
2138 40 09 40 02 45 B3 D0 08,D7
2140 40 09 00 22 44 33 D0 8C,15
2148 44 00 11 22 44 33 D0 8C,5F
2150 44 9A 10 22 44 33 D0 08,BE
2158 40 09 10 22 44 33 D0 08,88
2160 40 09 62 13 78 A9 00 21,88
2168 01 02 00 80 59 4D 11 12,D4
2170 06 4A 05 1D 2C 29 2C 23,EA
2178 28 41 59 00 58 00 00 00,04
2180 1C 8A 1C 23 5D 8B 1B A1,8D
2188 9D 8A 1D 23 9D 8B 1D A1,DA
2190 00 29 19 AE 69 A8 19 23,17
2198 24 53 1B 23 24 53 19 A1,FD
21A0 00 1A 5B 5B A5 69 24 24,23
21A8 AE AE A8 AD 29 00 7C 00,79
21B0 15 9C 6D 9C A5 69 29 53,BD
21B8 84 13 34 11 A5 69 23 A0,6A
21C0 D8 62 5A 48 26 62 94 88,EA
21C8 54 44 C8 54 68 44 E8 94,C6
21D0 00 B4 08 84 74 B4 28 6E,C4
21D8 74 F4 CC 4A 72 F2 A4 8A,D4
21E0 00 AA A2 A2 74 74 74 72,90
21E8 44 68 B2 32 B2 00 22 00,F4
21F0 1A 1A 26 26 72 72 88 C8,A8
21F8 C4 CA 26 48 44 44 A2 C8,96
```

RECORDER NOTES:

SUPERMON Version 2 (on a ROM chip marked 02-0012-B) seems to have solved most of our recorder problems. Three of us local SYMMERS can now exchange programs on cassette with no problems, other than perhaps a minor adjustment of the volume control. With the exchange ROM comes a set of three resistors, a capacitor and a jumper wire to upgrade our SYMS to the current production model. Haven't yet installed my sets, but two of the SYMMERS have; they tell me they can now read others' cassettes without even a volume control adjustment. This increased readback reliability will make it really feasible to distribute SYM software on cassette. (more about this below.)

Have not yet tried the corrected KIM format, even though I also have a KIM. Because of earlier problems with the KIM format I transferred my earlier software from KIM to SYM via a borrowed ASR 33 TTY using punched paper tape (ugh!). It is still not possible to read KIM tapes which record over $FE, $FF because this destroys the current value of BUFADL, BUFADH. Reading in over the stack area is no problem, however, unless you need the data stored there.

When we begin to distribute software on cassette another feature of SUPERMON Version 2 will be very helpful. Instead of dead leader, SY1.1 generates 6 seconds of SYNCH, which can be increased up to over 6 minutes(!). This will provide plenty of time to adjust volume controls. I would prefer to distribute software in the high speed format, reserving the slower KIM format for exchange with AIM and KIM systems. With my KIM, "Hypertape" was used exclusively; SYM will not accept these tapes, of course. Since there is now an easy way (see page 6) to convert the same programs from the "First Book of KIM" for SYM use, I will reactivate my old KIM, re-record the games in KIM format and enter them into SYM, make the mods and re-enjoy them.

And now a couple of hints: It's kind of reassuring to monitor the tape read visually or aurally, especially if you are having tape read problems. A scope can be used at the Extension Connector, E-X, but it is far simpler to permanently connect a transistor radio earphone across Application Connector pins A-L and A-1 and listen.

Had an extra earphone, so I plugged it into the earphone jack of the write recorder. Now I can monitor both S2 and L2 (or SAVE and LOAD, as they are called in BASIC, or PUT and GET, as they are called in RAE) by ear. It's reassuring to hear the data go by. I have learned to recognize many programs by their sound. I use blocks of AA, rather than 00 or EA to fill unused memory areas. This was originally done so I could more easily locate unused areas with a Verify; it also turns out that AA has a more musical sound!

If you are using two recorders, one for read and one for write (which works beautifully with RAE), beware of ground loops! I am using a single AC adaptor to power both recorders, so they are effectively grounded together at the adaptor. Until the ground lead from the Audio In jack at the read recorder was disconnected, the ground loop hum made operation impossible. You may have similar problems with a single recorder when

both the record and playback jacks are connected simultaneously. If so, try removing the ground lead to the playback jack.

The most cost effective recorder we have found is the Sanyo Recorder, Model 1530A. It lists for under $25.00, and every few months our two local super drugstores have them on sale for under $20.00. We have 6 of them in use here and plan to get many more.

We were curious to see if SYM would accept second generation tapes, duped from one recorder to another, so we bought a Radio Shack attenuating patch cord (42-2152) to make the tests. Results were excellent; SYM read the duped tapes as easily as originals, with only one stipulation. Because of the common AC adaptor problem mentioned above (on the scope it was observed that each recorder reflected a sawtooth waveform back into the power supply), it was necessary to run one recorder on its internal batteries. Hence, either use separate AC adaptors or modify the attenuating patch cord to eliminate the ground connection to the read jack. The reason for this experiment was to check out the feasibility of mass production of SYM tapes by audio means. Of course, for low volume production it is very simple to write a short program using .E (Execute) to make multiple dumps directly from SYM in both .S2 and .S1 formats on tapes for distribution.

QUESTIONS FROM THE EDITOR ????

We have published the program DIRECTORY in three formats - source code (RAE), disassembled form, and object code with check sum, so that you (and we) could compare the amount of space required to publish each format. While the source code is the most useful form, it does take lots of space compared to the object code. It looks like in the very near future (like next issue, in fact) we will have more programs to publish than we will have room for. One answer, the expensive one, is to go for more pages. The worst answer is to print only object code. My own feeling is to publish the disassembled form, wherever possible, to get more programs in, since the disassembled form is much more helpful for relocation purposes than is the object code. Of course if you have the disassembler available(you should, because it's published in this issue) we could save space by printing only object code. You could then enter and disassemble, then use Butterfield's RELOCATE from FBOK (or modify by "hand"), and SYM's Block Move to get the program where you want it. So we ask you the following questions: Would you prefer more programs, but in disassembled format or full source code at the expense of fewer programs? Would you want to purchase at a fixed price per page, full source listings for selected programs? Would you want to purchase (your cassette or mine) source code (RAE format) or object code on tape? (When RAE is more widely available source code will be the first class way to go.) What would you be willing to pay for programs on paper (per page)? For a readable cassette? Would you prefer a monthly? Please let us hear from you and give us your input.

A USEFUL EXTENSION FOR SYM BASIC

The followins program is an extension of an original program due to Carl Moser, by Thomas Gettys, 535 W. 12th. St., Chico, CA 95926, and correspondence and questions should be directed to him (self-addressed stamped envelope, please). Mr. Moser supplied the original program and other ideas in personal correspondence and telephone conversations. Thanks to the insights which he provided, we have been able to extend SYM BASIC in numerous ways, including APPEND, DELETE, "named" files etc. The idea of providing an input line buffer to BASIC can obviously be extended to a full page buffer for more elaborate editing. As an extension to APPEND one can pass BASIC programs to RAE for editing, including resequencing, commenting, etc., and to SYM BASIC for execution. The I/O vectoring capability in SUPERMON is a very powerful tool for these kinds of tasks.

A TERMINAL CONTROL PATCH FOR SYM BASIC

The terminal control patch is intended to be used with the SYM ROM BASIC and a CRT-type terminal, providing upgraded line editing features and other conveniences, such as automatically linking the trigonometric package to BASIC and allowing for an easy exit to the SYM monitor.

Deleting single characters with the rubout (or delete) key makes for a very unreadable display; using the "@" character to cancel a line is unconventional and thus hard to remember, and exiting to SUPERMON with X=USR(&"8035",0) is impossible! These aspects give one the impression of poor quality software, when actually the SYM BASIC is of high quality. Such considerations motivated the design of the terminal control patch to eliminate this illusion by improving the man-machine interface.

Control Functions

CONTROL C	Exit to the SYM monitor. Return to BASIC via the monitor G command.
CONTROL H	Delete the previous character. The character is erased from the screen.
CONTROL X	Cancel the current line. The entire line will be removed from the screen.

Additional Features

*Cassette SAVE/LOAD functions remain operational after exiting and re-entering BASIC.

*Automatic linkage to the trigonometric package.

*The memory size default is user defined.

*Enables lower case input to BASIC.

*Hex strings can be specified with a "$".

HOW TO USE THE TERMINAL CONTROL PATCH

Enter the object code from the listing provided. In order to relocate this code you need only change the addresses found in the macro expansions. The code is located so that the trig package exactly fits behind it (0EC7-0FFF, assuming you have 4K of memory on your SYM).

Cold start BASIC via the monitor command "G 0DE0". If you enter a carriage return in response to the "MEMORY SIZE" query then the numbers (in ASCII) in memory locations 0EC1-0EC5 will be passed to BASIC. This is to provide space for the terminal patch automatically. If you should hit the RESET key for some reason, warm start BASIC via the monitor command "G 0E94".

ASSEMBLE LIST

```
0010            .BA $0DE0
0020
0030 ;*************************************************
0040 ;*                                               *
0050 ;*    TERMINAL CONTROL PATCH FOR THE SYM-1 BASIC  *
0060 ;*                                               *
0070 ;*           COPYRIGHT 1979 BY C. MOSER          *
0080 ;*               ALL RIGHTS RESERVED             *
0090 ;*                                               *
0100 ;*************************************************
0110 ;
0120 ;
0130 ;FUNCTIONS
0140 ;---------
0150 ;  CTRL C       GO TO SYM MONITOR
0160 ;  CTRL H       DELETE LAST CHARACTER
0170 ;  CTRL X       DELETE CURRENT LINE
0180 ;
0190 ;EXTRAS
0200 ;------
0210 ;
0220 ;  PATCHES IN THE TRIG PACKAGE
0230 ;  MEMORY SIZE DEFAULT IS USER CHOSEN
0240 ;  ALLOWS LOWER CASE INPUT TO SYM BASIC
0250 ;  ALLOWS HEX STRINGS TO BE DESIGNATED WITH A "$"
0260 ;  CASSETTE OPERATIONS WORK AFTER EXIT AND RE-ENTER
0270 ;
0280 ;*************************************************
```

```
                0310 ;       DOUBLE STORE MACRO DEFINITION
                0320
                0330 !!!DS      .MD (DATA ADDRS) ;PUT DATA IN ADDRS
                0340            LDA #L,DATA
                0350            STA ADDRS
                0360            LDA #H,DATA
                0370            STA ADDRS+1
                0380            .ME
                0390
                0400
                0410 ;       ADDRESS DECLARATIONS
                0420
                0430 TRIG.PATCH .DE $00C4    ;TRIG PATCH LOCATION
                0440 BUFFER     .DE $0135    ;INPUT BUFFER
                0450 TRIG.START .DE $3F68    ;ENTRY TO TRIG PACKAGE
                0460 BASIC.COLD .DE $C000    ;COLD START TO BAS-1
                0470 BASIC.WARM .DE $C27E    ;WARM START TO BAS-1
                0480 OUTVEC     .DE $A663    ;OUTPUT TRANSFER VECTOR
                0490 INVEC      .DE $A660    ;INPUT TRANSFER VECTOR
                0500 RESXAF     .DE $8A3E    ;RESTORE ALL BUT A AND F
                0510 INTCHR     .DE $8A58    ;INPUT CHARACTER ROUTINE
                0520 ACCESS     .DE $8B86    ;UNWRITE PROTECT SYS RAM
                0530
                0540
                0550
                0560 ;       BEGIN MAINLINE
                0570
0DE0- 20 86 8B  0580 TCP.START  JSR ACCESS
0DE3- A9 00     0590            LDA #0
0DE5- 8D BF 0E  0600            STA NUMBER   ;FLAG CRT BUFFER CLEAR
                0610            DS (INITIAL INVEC+1)
0DF2- 4C 00 C0  0620            JMP BASIC.COLD
                0630
                0640
                0650 ;       PATCH IN TRIG PACKAGE AND SET MEMORY
                0660 ;       SIZE DEFAULT TO PROTEC THIS PROGRAM
                0670
                0680 INITIAL    DS (TRIG.START TRIG.PATCH)
                0690
0DFF- 20 44 0E  0700            JSR GET.LINE ;INPUT MEMORY SIZE
0E02- C0 01     0710            CPY #1       ;<CR> ONLY?
0E04- D0 0C     0720            BNE TCP      ;IF NOT, SKIP DEFAULT
0E06- 88        0730            DEY          ;ELSE USE TABLE VALUES
0E07- B9 C1 0E  0740 LOOP       LDA TABLE,Y  ;GET NEXT CHARACTER
0E0A- 99 35 01  0750            STA BUFFER,Y ;PUT IT IN BUFFER
0E0D- C8        0760            INY
0E0E- C0 06     0770            CPY #6       ;LAST CHARACTER?
0E10- 90 F5     0780            BCC LOOP     ;IF NOT, DO NEXT ONE
                0790
                0800 TCP        DS (GETCHR INVEC+1)
0E1C- 68        0810            PLA
0E1D- 68        0820            PLA
0E1E- 18        0830            CLC
0E1F- 90 0A     0840            BCC LINE+3
```



```
                0870 ;       BASIC CALL FOR CHARACTERS ENTRY POINT
                0880
0E21- 68        0890 GETCHR     PLA
0E22- 68        0900            PLA
0E23- AD BF 0E  0910            LDA NUMBER   ;CRT BUFFER EMPTY?
0E26- D0 0B     0920            BNE CHAR     ;BRANCH IF NOT
                0930
                0940 ;       GET A NEW LINE
                0950
0E28- 20 44 0E  0960 LINE       JSR GET.LINE
0E2B- 8C BF 0E  0970            STY NUMBER   ;SAVE # CHARS. INPUTTED
0E2E- A9 00     0980            LDA #0
0E30- 8D C0 0E  0990            STA INDEX    ;POINT TO START OF BUFFER
                1000
                1010 ;       PASS BASIC NEXT CHARACTER
                1020
0E33- AC C0 0E  1030 CHAR       LDY INDEX
0E36- B9 35 01  1040            LDA BUFFER,Y
0E39- EE C0 0E  1050            INC INDEX    ;UPDATE BUFFER POINTER
0E3C- CE BF 0E  1060            DEC NUMBER   ;UPDATE COUNT REMAINING
0E3F- C9 0D     1070            CMP #$D
0E41- 4C 3E 8A  1080            JMP RESXAF   ;RETURN TO BASIC
                1090
                1100 ;       GET NEXT LINE, CHECK FOR EDIT
                1110 ;       CONTROL CODES AND HEX STRINGS
                1120
0E44- A0 00     1130 GET.LINE   LDY #0
0E46- 20 B9 0E  1140 LP.GET     JSR INPUT    ;GET A CHARACTER
0E49- 99 35 01  1150            STA BUFFER,Y
0E4C- C8        1160            INY          ;UPDATE BUFFER POINTER
0E4D- C9 0D     1170            CMP #$D      ;IS IT A <CR>?
0E4F- D0 01     1180            BNE HEX.CHECK
0E51- 60        1190            RTS          ;IF SO, RETURN
                1200
0E52- C9 24     1210 HEX.CHECK  CMP #'$      ;IS IT A $?
0E54- D0 12     1220            BNE ^H.CHECK
0E56- 20 B9 0E  1230            JSR INPUT    ;IF SO GET NEXT CHARACTER
0E59- C9 22     1240            CMP #'"      ;IS IT A "?
0E5B- D0 EC     1250            BNE LP.GET+3 ;IF NOT, NOT A HEX STRING
0E5D- 99 35 01  1260            STA BUFFER,Y ;OTHERWISE SAVE "
0E60- A9 26     1270            LDA #'&      ;BASIC HEX CHARACTER
0E62- 99 34 01  1280            STA BUFFER-1,Y
0E65- C8        1290            INY          ;BUMP BUFFER POINTER
0E66- 10 DE     1300            BPL LP.GET   ;CONTINUE PROCESSING
                1310
0E68- C9 08     1320 ^H.CHECK   CMP #8       ;IS IT A CTRL H?
0E6A- D0 0A     1330            BNE ^X.CHECK
0E6C- 88        1340            DEY          ;IF SO BACK UP POINTER
0E6D- 88        1350            DEY          ;   FOR CTRL H AND CHAR.
0E6E- 30 D4     1360            BMI GET.LINE
0E70- 20 AE 0E  1370            JSR SP.BS    ;ERASE CHARACTER
0E73- 18        1380            CLC
0E74- 90 D0     1390            BCC LP.GET   ;GO GET NEXT CHARACTER
```

```
0E76- C9 18     1410 ^X.CHECK   CMP #$18      ;IS IT A CTRL X?
0E78- D0 09     1420            BNE ^C.CHECK
0E7A- 88        1430 CLEAR.LINE DEY          ;REMOVE LINE FROM SCREEN
0E7B- F0 C7     1440            BEQ GET.LINE
0E7D- 20 A9 0E  1450            JSR BS.SP.BS
0E80- 18        1460            CLC
0E81- 90 F7     1470            BCC CLEAR.LINE
                1480
0E83- C9 03     1490 ^C.CHECK   CMP #3       ;IS IT A CTRL C?
0E85- D0 BF     1500            BNE LP.GET   ;IF NOT, CONTINUE INPUT
                1510            DS (INTCHR INVEC+1)
0E91- 00        1520            BRK          ;IF SO, RESET INVEC
0E92- EA        1530            NOP          ;   AND BREAK TO MONITOR
0E93- EA        1540            NOP
                1550
                1560 ;       ENTRY POINT AFTER A CTRL C
                1570
0E94- A9 00     1580 ^C.ENTRY   LDA #0
0E96- 8D BF 0E  1590            STA NUMBER   ;FLAG CRT BUFFER EMPTY
0E99- 20 86 8B  1600            JSR ACCESS
                1610            DS (GETCHR INVEC+1)
0EA6- 4C 7E C2  1620            JMP BASIC.WARM
                1630
                1640
                1650 ;       TCP SUPPORT ROUTINES
                1660
0EA9- A9 08     1670 BS.SP.BS   LDA #8       ;PRINT SPACE,BKSPACE,SPACE
0EAB- 20 63 A6  1680            JSR OUTVEC
                1690
0EAE- A9 20     1700 SP.BS      LDA #' '     ;OUTPUT BACKSPACE, SPACE
0EB0- 20 63 A6  1710            JSR OUTVEC
0EB3- A9 08     1720            LDA #8
0EB5- 20 63 A6  1730            JSR OUTVEC
0EB8- 60        1740            RTS
                1750
0EB9- 20 58 8A  1760 INPUT      JSR INTCHR   ;INPUT A CHARACTER
0EBC- 29 7F     1770            AND #$7F
0EBE- 60        1780            RTS
                1790
                1800
                1810 ;       STORAGE AND DATA DECLARATIONS
                1820
0EBF-           1830 NUMBER     .DS 1        ;NO. OF CHARS. IN BUFFER
0EC0-           1840 INDEX      .DS 1        ;INDEX INTO BUFFER
0EC1- 30 33 35  1850 TABLE      .BY '03552'  ;MEMORY SIZE DEFAULT DATA
0EC4- 35 32
0EC6- 0D        1860            .BY 13       ;CARRIAGE RETURN
                1870
                1880 END.PGM    .EN
```

(several null lines were deleted to save space)


```
LABEL FILE:  [ / = EXTERNAL ]

/TRIG.PATCH=00C4      /BUFFER=0135          /TRIG.START=3F68
/BASIC.COLD=C000      /BASIC.WARM=C27E      /OUTVEC=A663
/INVEC=A660           /RESXAF=8A3E          /INTCHR=8A58
/ACCESS=8B86          TCP.START=0DE0        DATA=0E21
ADDRS=A661            INITIAL=0DF5          LOOP=0E07
TCP=0E12              GETCHR=0E21           LINE=0E28
CHAR=0E33             GET.LINE=0E44         LP.GET=0E46
HEX.CHECK=0E52        ^H.CHECK=0E68         ^X.CHECK=0E76
CLEAR.LINE=0E7A       ^C.CHECK=0E83         ^C.ENTRY=0E94
BS.SP.BS=0EA9         SP.BS=0EAE            INPUT=0EB9
NUMBER=0EBF           INDEX=0EC0            TABLE=0EC1
END.PGM=0EC7
//0000,0EC7,0EC7
```

SUPERMON Version 2 (SY1.1):
--------- ------- - -------

Some of the audio features of SY1.1 are described in the section RECORDER NOTES. A few of the other features are described here. Since hitting the Reset key reinitializes system RAM to default values it's nice to avoid the use of Reset if your program requires non-default values in system RAM.

SY1.1 provides two helps in this matter. The CR key on the hex pad will abort a tape load if you change your mind, or if you have put in a non-existent ID number (or, in RAE, forgotten an EOF marker). Also TSTAT at $8B3C now checks for Break key down on both the 20ma loop (TTY) and the RS-232 (CRT) terminals. I have rewritten most of my programs which have infinite loops (graphic and audio) to include a JSR TSTAT with appropriate action on return to permit stopping with terminal Break, rather than Reset.

The display of the cassette ID number is nice, since you may have forgotten what numbers you have used on the tape. It will also help you to identify a tape prepared by RAE. RAE uses ID-00 for all its records, so the absence of ID numbers indicates a RAE file.

Since Version 2 is so inexpensive* ~~($15.00~~ + 6% California sales tax, ~~shipping free in the US from either Sunrise Systems or~~ us, including ~~the latest edition of the SYM-1 Reference Manual, June 1979)~~ we will not support programs written in Version 1. The program DIRECTORY, for example, is valid only for Version 2. While most of the changes affect only the cassette portion of SUPERMON, we will feel free, for example, to use subroutines from that portion in non-cassette programs. For example JSR SYNC5 at $8D76 could be used (preceded by PHA PHA if necessary to cancel the PLA PLA in USRREQ) to permit the CR hex pad key to serve as a Break key in infinite loops. Of course if this were used, an extra JSR STTC at $8DBB would be needed to retoggle the tape deck control.

*$16.00 U.S.FUNDS FOR US/CANADA
 $17.00 U.S.FUNDS FOR EUROPE AIR MAIL DELIVERY
 $18.00 U.S.FUNDS FOR ASIA/PACIFIC AIR MAIL DELIVERY

**the MON 1.1 ENHANCEMENT MANUAL.

SOFTWARE RECOMMENDATION: A 2K Symbolic Assembler for the 6502

The only time during my professional career when I actually programmed for a living was during the period 1949-1952, for the SWAC (NBS Western Automatic Computer) and a series of magnetic drum based airborne computers. This was long before the days of assemblers, so we assembled by hand. We didn't call it hand-assembly, we called it coding. My very first experience with an assembler, and it was a very pleasant learning experience, was with "A 2K Symbolic Assembler for the 6502." This was developed by Robert Ford Denison, RD5 Teeter Road, Ithaca, NY 14850, for the KIM, and is adaptable to SYM by merely changing from the KIM to SYM equivalents of CRLF, OUTCH (OUTCHR), GETCH (INCHR), and OUTSP (SPACE).

In a brief note in the Aug-Sep 1978 issue of MICRO #6, Bob offered a free "sneak preview" of the assembler to get user feedback on the documentation. I have been using the 2KSA since last October, until I obtained RAE (the SSC 8K assembler), and one of my students is using it now on his SYM. The program and its documentation are models of what programs and documentation should be, and I recommend this program highly to the low budget beginner for a number of reasons.

First, by studying the documentation, the beginning assembly language programmer will learn much about clean, structured programs and their documentation, and how to write assembly language programs.

Second, the 2K program, even in a 4K SYM, leaves a lot of room for the applications program. If the SYM is expanded to 8K with the W7AAY board mentioned elsewhere in this issue, the 2KSA can be either left where it is (0200-09FF) or moved up out of the way to the very top of the RAM. It can also be PROMmed or EPROMmed to fit into the otherwise unused 2K block at F000-F7FF. Explicit detailed instructions on relocation and modification are given in the manual.

Third, the user does not even need an elaborate terminal. The author describes in his Appendix A: "An Inexpensive I/O System," how to interface a Qwerty keyboard for under $35.00, and gives the software for driving the KIM segment displays. This software can be easily modified for SYM, but one of my students went this one better, by using the scope driver program given in the SYM-1 manual as the basis for using a scope as his output terminal. The manual even shows how to modify the 2KSA to match a 32 column width terminal; this is just the right length to match the 32 byte scope buffer provided by SUPERMON. Of course, a "real" terminal is much nicer.

The 2KSA does not use the standard MOS Technology mnemonics; rather it uses the same mnemonics used in MICRO, which put the addressing mode information in the opcode rather than with the operand. For me, this is no problem; once I start thinking with either set of mnemonics, I continue without thinking further about it. I plan to have three SYM systems, two fully expanded with RAE in ROM, one for school, one for home and one in which the ROM sockets are for special applications. In this special purpose SYM I will use the 2KSA in RAM (or possibly ROM) for applications program development.

Copies of the 2KSA manual may be obtained from ~~either Mr. Dennison at the above address, or from~~ the SYM Users' Group for ~~$9.00 plus $1.00~~ for first class postage. Cassette tapes of "2KSA" are available from SUG in combined SYM/KIM dumps, for ~~$5.00~~ postpaid. California addresses please add 6% sales tax. SEE BELOW

HARDWARE RECOMMENDATION: W7AAY 4K Memory Expansion Board

Two of us have added the W7AAY 4K memory expansion boards to our systems, and will be adding them to two more systems soon. We already had 4K RAM on-board, and the SSC MBC016 16K RAM off-board. These latter are normally addressable only at 8K boundaries, and we needed 4K more to fill the gap at 1000-1FFF. The W7AAY boards are almost on-board, in that the board is mounted on the SYM-1 board itself in the logo area. The board holds ten 2114 chips, two of which were removed from sockets U12 and U19. The board connects to SYM with two 16 wire jumpers with 16 pin DIP pluss going to the now empty sockets at U12 and U19 and four added wires to pick up the $\overline{10}$, $\overline{14}$, $\overline{18}$ and $\overline{1C}$ signals, either from chip U1, pins 7, 9, 10, and 11, or from the applications connector pins A-F, A-H, A-K, and A-J (a jumper is needed at HH, 41). We tried both methods, and both work well. I placed a piece of thick cardboard over the logo area and the expansion board rests on the cardboard, supported in place by the jumper cables. I like to think of having 8K on-board RAM. The boards, plus instructions, are available from John Blalock, ~~3054 West Evans Drive, Phoenix, AZ 85025, for $5.00~~ and a stamped, self-addressed envelope. *P.O. BOX 39356, PHOENIX, ARIZONA 85069. PRICE $8.00

See "SYMple Memory Expansion," by John M. Blalock, MICRO #15, pages 42-43, for a description of the board. See, also, "Another KIM-1 Expansion," by John M. Blalock, Kilobaud MICROCOMPUTING #33 (September 1979), pages 130-133, for ideas on how to add 24K to SYM-1. Now, wouldn't it be nice if Mr. Blalock would design a similar extension board to hold all of the EPROMS we would like to insert into the one available socket at U23!

QUESTIONS TO THE EDITOR ????

We don't have any questions from readers this issue (only our own), but we propose the following. If you have questions we can answer easily, we will; if you address and stamp a return envelope. We will publish those of general interest. If we at SYM-PHYSIS can't answer your questions we will put your questions in our SYM-pathy column and ask our readers to come up with a solution. We know that the SYM documentation does not answer all of our (the SYM users) requirements, but we hope SYM-PHYSIS will.

SOFTWARE/HARDWARE DISTRIBUTION:

If you have SYM software and/or hardware you wish to market, contact us. We can use RAE-1 to prepare the manuals and listings, and can advertise and distribute through the SYM Users' Group.

*MANUAL $11.00 ... UNDS US/CANADA
PRICES OBSOLETE UNDS AIR ... FOR CURRENT PRICES
TAPE ... PACIFIC
SEE LATEST "GREEN SHOPPING LIST" ... CANADA
... US. FUNDS AIR MAIL TO EUROPE
$ 6.75 US. FUNDS AIR MAIL TO ASIA/PACIFIC

LETTERS TO THE EDITOR:

Here is our very first letter to the editor. Mr. Vrtis explains in his last paragraph, better than we can, the reason for SYM-PHYSIS.

August 14, 1979

Nicholas J. Vrtis
5863 Pinetree S.E.
Kentwood, Mi. 49508

Dear Dr. Luxenberg:

I hope that the enclosed article will be of some use to you for the SYM Users' Newsletter. I would have gotten back to you sooner, but your letter arrived while I was on vacation.

I don't know if this article is the type of stuff you wanted for the Newsletter, but I figured that if you wanted to get it out some time around mid-September, I had better just send it along with my answer that I would like to contribute.

I will be looking forward to the Newsletter. The problem with writing for MICRO is that they only include one SYM article per issue usually, and when it is mine, I've already read it. It will be nice to read a number of articles about the SYM.

Sincerely,

Nick

P.S. The Verify of $00-$59 is 2331, and the dump at the end is 5P 0,59 with XXXRC = $15

FORMAT FOR SUBMISSIONS:

All typewritten or computer generated manuscripts, listings, etc., submitted will be pasted up within a 10" by 15 1/2" (horizontal) area on 11" by 17" sheets and reduced 70% for printing. Some materials will be Xerox reduced 64% prior to pasteup to save space if legibility permits. See Nicholas J. Vrtis' excellent article, "Hex Program Verify Program," for an example of how your material will look after this treatment.

EDITOR'S NOTES:

*One of the local SYMMERS had some problems reading with SUPERMON Version 2 tapes saved with the original SUPERMON. Version 2 initializes some of its cassette parameters into RAM where you can set at them and modify them. By changing the value of HSBDRY in $A632 according to the procedure described in Technical Note No.72-SSC, April 1979, he was able to read and convert his tapes. Two others of the local group did not have this problem.

*Synertek Systems Corporation has published a series of Technical Notes during the past year or so. Nine of them, as listed below have been bound into one volume. This volume is available from the SYM User's Group at ~~$3.00 plus $1.00~~ for first class postage (Californians please add 6% sales tax).

TN 22	SYM-1	Input/Output Pins Utilization
TN 35	SYM-1	Updates for "VIM" Reference Manual
TN 49	SYM-1	Sample Programs
TN 50	SYM-1	Display Routine
TN 52	SYM-1	Time Delay Using 6532 Timer
TN 53		Trigonometric Functions for Synertek BASIC
TN 54	SYM-1	Power-Up to User ROM
TN 59	SYM-1	Modification for Inputting Lower Case Characters
TN 72		Cassette Data Reading Using SYM-1 High Speed Format

*David B. Schaechter, 14053 Fenton Lane, Sylmar, CA 91342, has available OTHELLO for the 1K SYM-1. I have seen a copy of his source code; but have not yet received the cassette object code. I suggested to Dr. Schaechter that, for people unfamiliar with the game, he also include a description of the rules of the game. Contact him directly for price and other information.

*We will be teaching a week-end course called "Microprocessor Fundamentals," for the University of California at Davis, January 25-27,1980. The course fee is $450.00, and each student will receive a SYM-1 and power supply plus some software goodies. For a bulletin on this course, contact University Extension, UCD, Davis, CA 95616.

COMING ATTRACTIONS:

We have a number of graphic and audio packages for future issues; publication of these depends on how large a portion of our subscribers have added the necessary D/A and A/D interfaces. We also have more utility packages, depending on the interest shown. There is no way of telling what kinds of contributions you readers will be sending in, so we too will be surprised by the next issue. See you then... Lux

Synertek Systems
CORPORATION

150 S. WOLFE RD. • SUNNYVALE, CA 94086 • TELEPHONE (408) 988-5689
P. O. Box 552 • SANTA CLARA, CA 95052 • TWX: 910-338-0135

October 1979

Dear SYM Owner:

We are happy to announce the beginning of a new publication devoted solely to SYM users, the "SYM-Physis." The newsletter is published by Dr. H. R. Luxenberg of the California State University at Chico. Dr. Luxenberg, who is a professor of computer science, has wide experience with microcomputers and is especially knowledgable in the SYM system. We at Synertek Systems are happy to see him publish this newsletter.

"SYM-Physis" promises to be of great value to all SYM users, whether your emphasis is hobby or OEM. Problems, programs, games, etc., will be discussed and will be of interest to many of you. Of course, articles by you the user are requested for publication.

Subscriptions to the newsletter can be obtained by completing the form on the last page of the newsletter. Make all checks payable to SYM Users' Group.

Happy reading and good luck to "SYM-Physis."

Sincerely yours,

SYNERTEK SYSTEMS
Marketing Department

BULK RATE
U.S. Postage
PAID
Chico, CA 95927
Permit # 430

TIME VALUE PRINTED MATTER

SYM-PHYSIS
SYM-1 Users' Group
P.O. Box 315
Chico, CA 95927

Address Correction Requested