THE USER'S GROUP OFFICE WILL BE CLOSED DEC. 20-24, 1982

SYM-PHYSIS

THE SYM USERS' GROUP NEWSLETTER

DOUBLE ISSUE
VOLUME III, NUMBER 3 (ISSUE NO. 13) - AUTUMN 1982 (JUL/AUG/SEP)
VOLUME III, NUMBER 4 (ISSUE NO. 14) - WINTER 1982 (OCT/NOV/DEC)

SYM-PHYSIS is a quarterly publication of the SYM Users' Group, P. O. Box 319, 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*
Office Staff: *Joyce Arnovick, Denny Hall*

SUBSCRIPTION RATES: (Volume III, 1982, Issues 11 - 14)

USA/Canada - $10.50 for a volume of four issues. Elsewhere - $14.00. Make checks payable in US dollars to "SYM Users' Group", P. O. Box 319, Chico, CA 95927, Telephone (916) 895-8751.

BACK ISSUES ARE STILL AVAILABLE AS FOLLOWS:

Issue 0, the Introductory Issue (1979), and Issues 1 through 6 (Volume I, 1980), are available, as a package, for $12.00, US/Canada, and $16.00, First Class/Airmail, elsewhere.

Issues 7 through 10 (Volume II, 1981), are available for $10.50, US/Canada, and $14.00, First Class/Airmail, elsewhere.

RAM-B'LINGS

This "double-issue" marks the end of our third full year of publication (add a half-year, if you count the introductory issue). With the final issue of each volume, we must decide, each year, if we should try for still another. We seem to be living our lives on a one-year-at-a-time basis! We wondered why the quarterly publication deadlines seemed so much more difficult to meet than the bi-monthly publication schedule, and we finally figured it out.

Our university, California State University, Chico, is on a semester, rather than on a quarterly basis. Since the newsletter preparation cycle must be meshed in with our teaching schedule, which is actually trimestrial in nature, if the summer period is taken into account, either three issues/year, or six issues/year, would be much more commensurate with our three cycle/year teaching load than the four issues per year we have been trying for during the past two years.

We feel that preparing one issue each semester, and a third during the shorter summer session, when the teaching load is lighter, would make for a much more sensibly distributed workload. We will go on, then, with Volume 4, on a thriceannually basis, with Spring, Summer, and Fall issues (numbers 15, 16, and 17, respectively).

Each of the three issues will be some 52 pages, instead of the current 40 pages, so that Volume 4 will contain very nearly the same amount of text as the current volume. Unfortunately for California subscribers, however, any periodical published less frequently than quarterly is not

(continued to page 13/14-48)

FORTH IN ROM/EPROM - A PRELIMINARY REPORT

Jerry Larsen of Synertek Systems sent us a 2732 EPROM containing a preliminary version of the object code for a 4K FORTH in ROM which Synertek is planning to release for the SYM-1/SYM-2. He asked us to give the chip a good workout, and to report any problems back to him, together with any comments or suggestions we might care to make. Here are some extracts from his letter:

> We plan to supply a copy of Brodie's book (Starting Forth) with the chip as a tutorial text along with a (better) copy of the glossary included with this chip. The program itself is a subset of the Forth-79 Standard. Word definitions therefore follow the Standard, Brodie's book and then fig-Forth in that order. This will lead to some differences from the Forth by Jack Brown. The major exception to the Standard, besides some words which were omitted, is that we do not support double length numbers. It was not felt necessary since this Forth is intended as a control system language for the SYM-1. If needed, the user can write his own double length routines and a new definition for NUMBER. The code field address for the new NUMBER is then placed in memory at $CC-$CD and double length numbers will be available. We plan application notes on this and other extended features such as disk I/O.

We installed the EPROM on a new 4K RAM SYM-1 in socket U21, after first modifying the jumpers to conform to a 2732 at $C000-$CFFF (FORTH overlays the lower half of BAS-1), and began our checkout. Incidentally, the 2732 differs sufficiently in its pinout from the 2332/2532 ROM/EPROM pair that it cannot be used in a socket jumpered for the lower half of BAS-1 (if you have installed the earlier version of BAS-1, which came in a pair of 2332 ROMs).

Reproduced below is a printout of the listing produced by the FORTH word VLIST (short for VOCABULARY LIST):

```
FORTH  X.2
COPYRIGHT 1982 SYNERTEK CORP.
VLIST
  4 TAS   3 MON   3 GET   3 PUT   6 ACC   4 LOA   1 P     1 L
  4 WIP   4 LIN   4 FIL   3 L/S   5 VLI  10 VOC   5 DOE   7 <BU
  1 ?     2 U.    1 .     2 .R    3 (.)   2 #S    1 #     4 SIG
  2 #>    2 <#    3 PAD   4 HOL   6 SPA   3 MIN   3 MAX   3 ABS
  2 */    5 */M   3 MOD   1 /     4 /MO   1 *     4 ?DU   9 IMM
  6 FOR   5 ABO   4 QUI  11 DEF   5 FOR   7 LIT   9 [CO   1 ]
  1 [     9 INT   1 '     6 NUM   6 CRE   1 (     4 WOR   1
  5 QUE   6 EXP   4 ELS   5 WHI   2 IF    6 REP   5 AGA   5 UNT
  5 +LO   4 LOO   2 DO    5 BEG   4 THE   7 DEC   3 HEX   2 ."
  4 TYP   5 COU   7 COM   5 SPA   2 BL    3 ROT   1 >     1 <
  1 =     1 -     2 C,    1 ,     5 ALL   4 HER   2 2+    2 1+
  2 BS    3 TOP   4 BUF   4 BAS   5 STA   3 BLK   3 >IN   7 CUR
  7 CON   1 H     1 2     1 1     1 0     8 VAR   8 CON   5 ;CO
  1 ;     1 :     9 ?TE   2 CR    4 EMI   3 KEY   5 CMO   7 EXE
  6 (FI   5 DIG   5 U/M   2 U*    3 XOR   3 AND   6 NEG   1 +
  2 0<    2 0=    2 -R    5 LEA   4 EXI   1 I     2 R>    2 >R
  4 SWA   4 DRO   4 OVE   3 DUP   3 SP@   2 +!    2 C!    1 !
  2 C@    1 @   OK
```

As you can see, the "dictionary" stores each of the words in an abbreviated form requiring exactly four bytes for each. The format consists of one hex byte containing the length (number of characters) of the entry followed by the first three ASCII characters of the word. Short names are padded to three characters with trailing spaces. The current trend in implementing FORTH is to provide for variable length names (unabridged). A VLIST then omits the length digit and all of the words are spelled out in full, although not so neatly tabulated.

While we do prefer the variable length word names as providing greater recognition capability, we are willing to give them up for the sake of getting more important capabilities into the 4K allocated. Besides this FORTH is intended for control applications, and who uses VLIST in a control application?

The major difference between this FORTH and the fig-FORTH and the 79-STANDARD models is in the omission of the double precision capability. This was a reasonable compromise, since 16 bits is more than adequate for any analog process control.

As noted earlier, we would much rather see a VLIST in which the words are spelled out in full. For example, it took us a few minutes to figure out that 4 TAS stood for TASK, as in FORGET TASK. TASK can actually be forgotten, i.e., deleted from the vocabulary. Incidentally, TASK is copied from ROM into RAM in page 02 in order to mark both the starting point of the dictionary and the starting point of the user space; it is otherwise essentially equivalent to the 6502 NOP. All default values are copied down to page 00, as are two vectors which may be changed to permit easier expansion to the full 79-STANDARD model.

We found one very obvious "bug" by inspecting the original VLISTing (this has been corrected in the version above). Fixing the bug required changing one byte in the object code, but since we did not have our 2732 EPROM burner finished, and since we prefer to work from RAM anyway, at least for software still in the development stage, we decided to relocate the object code at $9000, using Dessaintes' Disassembler.

Now, FORTH is a "threaded" language, which means that the "compiled" form consists of "strings" (into, or onto, which the words are "threaded"). Actually, each word is assigned a 16-bit (two-byte) vector; it is these vectors which are the "beads" on the strings. Furthermore, only a very small portion of the FORTH "implementer" (we deliberately avoid the use of the terms "compiler" or "interpreter" here), need be written in the "native" machine language (ML). Once a few FORTH words are defined in ML the rest of the words may be defined in terms of these, with only occasional requirements for additional ML sections. This means that the majority of the FORTH implementer is written in FORTH itself, sort of on a bootstrap principle.

Thus, it turned out that less than 20% of the 4K object code was written in ML, the remainder consisting of vectors and isolated one or two byte "literals" and ASCII encoded messages. The disassembler created gibberish for this portion, but since we had some a priori knowledge of FORTH's structure, it was omly a matter of many hours of dog-work to come up with a reasonably complete source code. Since the FORTH words are precisely defined in an accompanying glossary, the source code is almost self commenting.

We hope that Synertek will see fit to provide the source code with the release package, or authorize its independent publication, since we feel that one very good way to really understand how to use FORTH is to see how it builds itself up from a very simple nucleus.

The following extract from the (uncopyrighted) FORTH-79 Standard, available from the FORTH INTEREST GROUP, P. O. Box 1105, San Carlos, CA 94070, is reproduced for the convenience of those who may wish to compare the VLIST above against the standard:

10. REQUIRED WORD SET

The words of the Required Word Set are grouped to show like characteristics. No implementation requirements should be inferred from this grouping.

Nucleus Words

! * */ */MOD + +! +loop - / /MOD 0< 0= 0> 1+ 1- 2+ 2- < = > >R ?DUP @ ABS AND begin C! C@ colon CMOVE constant create D+ D< DEPTH DNEGATE do does> DROP DUP else EXECUTE EXIT FILL I if J LEAVE literal loop MAX MIN MOD MOVE NEGATE NOT OR OVER PICK R> R@ repeat ROLL ROT semicolon SWAP then U* U/ U< until variable while XOR

(note that the lower case entries refer to just the run-time code corresponding to a compiling word.)

Interpreter Words

#> #S ' (-TRAILING . 79-STANDARD <# >IN ? ABORT BASE BLK CONTEXT CONVERT COUNT CR CURRENT DECIMAL EMIT EXPECT FIND FORTH HERE HOLD KEY PAD QUERY QUIT SIGN SPACE SPACES TYPE U. WORD

Compiler Words

+LOOP , ." : ; ALLOT BEGIN COMPILE CONSTANT CREATE DEFINITIONS DO DOES> ELSE FORGET IF IMMEDIATE LITERAL LOOP REPEAT STATE THEN UNTIL VARIABLE VOCABULARY WHILE [[COMPILE]

Device Words

BLOCK BUFFER EMPTY-BUFFERS LIST LOAD SAVE-BUFFERS SCR UPDATE

MORE ON FORTH

If we could have but one higher level language for our SYM-1, or for any other system, for that matter (see below), our choice would be FORTH. Here are some of our reasons:

First, what FORTH provides, in essence, is a STANDARDIZED set of macros to supplement the natural machine language of the host computer. This means that if your applications programs are written wholly in terms of these macros (i. e., FORTH words), they are 100% transportable between systems, independent of the nature of the host computer! (One exception, of course, is time-dependent programs, such as music applications, unless the programs include allowances for differing clock rates, etc.)

Second, FORTH is the easiest higher level language to implement on any microcomputer, especially after you have implemented it on your first one, or have disassembled a working version for any particular microprocessor. More on this below.

Third, because of its "threaded" structure, FORTH is nearly as fast as machine language itself, and requires far less memory than any other higher level language. Furthermore it is infinitely extensible; you can add as many new words as desired, organizing them into separate VOCABULARY groups for different applications, if you wish.

It is convenient to extend FORTH to include an ASSEMBLER vocabulary, so that ML programs may be incorporated into applications programs where maximum speed is required. More sophisticated editing capabilities may be added by incorporating any one of the EDITOR vocabularies appearing in the open literature (much FORTH material is in the public domain). Thus FORTH can include a Resident Assembler Editor (RAE), if desired.

Forth(!), FORTH customarily treats any supplementary mass storage as virtual memory, so that very little RAM is actually required for even the most elaborate development systems. A 32K SYM-1 with a pair of floppies, any size, should handle just about any control application that can be assigned to a microprocessor system.

(continued to page 13/14-30)

```
                        0010 ; "SKELETONIZED" FORTH FOR DEMONSTRATION PURPOSES
                        0020
                        0030            .BA $9000
                        0040
                        0050 ; NOTICE THE INITIAL PORTION IS MAINLY MACHINE LANGUAGE
                        0060
                        0070 ; FETCH, "@", AND STORE, "!", ARE THE ROOT "PRIMITIVES"
                        0080
9000- 01                0090 FETCH.     .BY $01
9001- 40 20 20          0100            .BY '@ ' $20
9004- 00 00             0110            .BY $00 $00  ;END OF DICTIONARY MARKER
9006- 08 90             0120 FETCH      .SI FETCHX
                        0130
9008- B5 01             0140 FETCHX     LDA *$01,X
900A- 95 FF             0150            STA *$FF,X
900C- A1 FF             0160            LDA ($FF,X)
900E- 48                0170            PHA
900F- F6 FF             0180            INC *$FF,X
9011- D0 02             0190            BNE =+3
                        0200
9013- F6 00             0210            INC *$00,X
9015- A1 FF             0220            LDA ($FF,X)
9017- 4C 48 90          0230            JMP ENTER
                        0240
901A- 01                0250 STORE.     .BY $01      ;NUMBER OF CHARACTERS IN WORD
901B- 21 20 20          0260            .BY '! ' $20 ;FIRST THREE CHARACTERS OF WORD
901E- 00 90             0270            .SI FETCH.   ;POINTER TO NEXT WORD
9020- 22 90             0280 STORE      .SI STOREX   ;POINTER TO MACHINE LANGUAGE
                        0290
9022- B5 01             0300 STOREX     LDA *$01,X
9024- 95 FF             0310            STA *$FF,X
9026- B5 03             0320            LDA *$03,X
9028- 81 FF             0330            STA ($FF,X)
902A- F6 FF             0340            INC *$FF,X
902C- D0 02             0350            BNE =+3
                        0360
902E- F6 00             0370            INC *$00,X
9030- B5 02             0380            LDA *$02,X
9032- 81 FF             0390            STA ($FF,X)
                        0400
                        0410 ;  STACK MANAGEMENT UTILITIES
                        0420
9034- E8                0430 POPTWO     INX
9035- E8                0440            INX
9036- E8                0450 POPONE.N   INX
9037- E8                0460            INX
9038- D0 13             0470            BNE NEXT     ;ALWAYS
                        0480
903A- E8                0490 POPONE.E   INX
903B- E8                0500            INX
903C- D0 0A             0510            BNE ENTER    ;ALWAYS
                        0520
903E- 48                0530 NOPUSHENT  PHA
903F- A9 00             0540            LDA #$00
9041- F0 05             0550            BEQ ENTER    ;ALWAYS
                        0560
9043- 48                0570 PUSHENT    PHA
9044- A9 00             0580            LDA #$00
9046- CA                0590 PUSH       DEX
9047- CA                0600            DEX
9048- 95 00             0610 ENTER      STA *$00,X
904A- 68                0620            PLA
904B- 95 01             0630            STA *$01,X
904D- A5 DA             0640 NEXT       LDA *$DA
904F- 18                0650 NEXT1      CLC
9050- 69 02             0660            ADC #$02
```



```
9052- 85 DA             0670            STA *$DA
9054- 90 02             0680            BCC =+3
                        0690
9056- E6 DB             0700            INC *$DB
9058- A0 01             0710            LDY #$01
905A- B1 DA             0720            LDA ($DA),Y
905C- 85 DE             0730            STA *$DE
905E- 88                0740            DEY
905F- B1 DA             0750            LDA ($DA),Y
9061- 85 DD             0760            STA *$DD
9063- 4C DC 00          0770            JMP $DC
                        0780
                        0790 ;    END OF UTILITIES
                        0800
9066- 03                0810 DUP.       .BY $03
9067- 44 55 50          0820            .BY 'DUP'
906A- 1A 90             0830            .SI STORE.
906C- 6E 90             0840 DUP        .SI DUPX
                        0850
906E- B5 01             0860 DUPX       LDA *$01,X
9070- 48                0870            PHA
9071- B5 00             0880            LDA *$00,X
9073- 4C 46 90          0890            JMP PUSH
                        0900
9076- 04                0910 OVER.      .BY $04
9077- 4F 56 45          0920            .BY 'OVE'
907A- 66 90             0930            .SI DUP.
907C- 7E 90             0940 OVER       .SI OVERX
                        0950
907E- B5 03             0960 OVERX      LDA *$03,X
9080- 48                0970            PHA
9081- B5 02             0980            LDA *$02,X
9083- 4C 46 90          0990            JMP PUSH
                        1000
9086- 04                1010 DROP.      .BY $04
9087- 44 52 4F          1020            .BY 'DRO'
908A- 76 90             1030            .SI OVER.
908C- 36 90             1040 DROP       .SI POPONE.N
                        1050
908E- 04                1060 SWAP.      .BY $04
908F- 53 57 41          1070            .BY 'SWA'
9092- 86 90             1080            .SI DROP.
9094- 96 90             1090 SWAP       .SI SWAPX
                        1100
9096- B5 03             1110 SWAPX      LDA *$03,X
9098- 48                1120            PHA
9099- B5 01             1130            LDA *$01,X
909B- 95 03             1140            STA *$03,X
909D- B5 02             1150            LDA *$02,X
909F- B4 00             1160            LDY *$00,X
90A1- 94 02             1170            STY *$02,X
90A3- 4C 48 90          1180            JMP ENTER
                        1190
90A6- 04                1200 EXIT.      .BY $04
90A7- 45 58 49          1210            .BY 'EXI'
90AA- 8E 90             1220            .SI SWAP.
                        1230
                        1240 DOSEMICOLN ;(ALTERNATE NAME FOR EXIT)
                        1250
90AC- AE 90             1260 EXIT       .SI EXITX
                        1270
90AE- 68                1280 EXITX      PLA
90AF- 85 DB             1290            STA *$DB
90B1- 68                1300            PLA
90B2- 4C 4F 90          1310            JMP NEXT1
```

```
                    1320
90B5- 02            1330 ZEROEQ.     .BY $02
90B6- 30 3D 20      1340             .BY '0= '
90B9- A6 90         1350             .SI EXIT.
90BB- BD 90         1360 ZEROEQ      .SI ZEROEQX
                    1370
90BD- B5 00         1380 ZEROEQX     LDA #$00,X
90BF- 15 01         1390             ORA #$01,X
90C1- D0 01         1400             BNE =+2
                    1410
90C3- C8            1420             INY
90C4- 98            1430             TYA
90C5- 4C 3E 90      1440             JMP NOPUSHENT
                    1450
90C8- 02            1460 ZEROLESS.   .BY $02
90C9- 30 3C 20      1470             .BY '0<
90CC- B5 90         1480             .SI ZEROEQ.
90CE- D0 90         1490 ZEROLESS    .SI ZEROLESSX
                    1500
90D0- B5 00         1510 ZEROLESSX   LDA #$00,X
90D2- 29 80         1520             AND #$80
90D4- 0A            1530             ASL A
90D5- 2A            1540             ROL A
90D6- 4C 3E 90      1550             JMP NOPUSHENT
                    1560
                    1570
90D9- 06            1580 NEGATE.     .BY $06
90DA- 4E 45 47      1590             .BY 'NEG'
90DD- C8            1600             .BY ZEROLESS.
90DE- E0 90         1610 NEGATE      .SI NEGATEX
                    1620
90E0- 38            1630 NEGATEX     SEC
90E1- 98            1640             TYA
90E2- F5 01         1650             SBC #$01,X
90E4- 48            1660             PHA
90E5- 98            1670             TYA
90E6- F5 00         1680             SBC #$00,X
90E8- 4C 48 90      1690             JMP ENTER
                    1700
90EB- 01            1710 PLUS.       .BY $01
90EC- 2B 20 20      1720             .BY '+ ' $20
90EF- D9 90         1730             .SI NEGATE.
90F1- F3 90         1740 PLUS        .SI PLUSX
                    1750
90F3- 18            1760 PLUSX       CLC
90F4- B5 01         1770             LDA #$01,X
90F6- 75 03         1780             ADC #$03,X
90F8- 48            1790             PHA
90F9- B5 00         1800             LDA #$00,X
90FB- 75 02         1810             ADC #$02,X
90FD- 4C 3A 90      1820             JMP POPONE.E
                    1830
                    1840 ; NOTE THAT NOT ALL FORTH WORDS NEED BE IN THE DICTIONARY
                    1850
9100- 02 91         1860 BRANCH      .SI BRANCHX
                    1870
9102- A0 02         1880 BRANCHX     LDY #$02
9104- B1 DA         1890             LDA ($DA),Y
9106- A0 00         1900             LDY #$00
9108- C9 00         1910             CMP #$00
910A- 10 01         1920             BPL =+2
                    1930
910C- 88            1940             DEY
910D- 18            1950             CLC
910E- 65 DA         1960             ADC #$DA
9110- 85 DA         1970             STA #$DA
```



```
9112- 98            1980             TYA
9113- 65 DB         1990             ADC #$DB
9115- 85 DB         2000             STA #$DB
9117- 4C 4D 90      2010             JMP NEXT
                    2020
                    2030
911A- 1C 91         2040 ZBRANCH     .SI ZBRANCHX
                    2050
911C- E8            2060 ZBRANCHX    INX
911D- E8            2070             INX
911E- B5 FE         2080             LDA #$FE,X
9120- 15 FF         2090             ORA #$FF,X
                    2100 ;           BNE FIXUP
                    2110
9122- F0 DE         2120             BEQ BRANCHX  ;ALWAYS
                    2130
9124- A5 DA         2140 DOCOLON     LDA #$DA
9126- 48            2150             PHA
9127- A5 DB         2160             LDA #$DB
9129- 48            2170             PHA
912A- A5 DE         2180             LDA #$DE
912C- 85 DB         2190             STA #$DB
912E- A5 DD         2200             LDA #$DD
9130- 4C 4F 90      2210             JMP NEXT1
                    2220
                    2230 ; NOTICE THE FINAL PORTION IS MAINLY "COMPILED" FORTH
                    2240 ; STRUCTURE IS DOCOLON WORD1 WORD2 ... WORDN DOSEMICOLON
                    2250
9133- 01            2260 EQUAL.      .BY $01
9134- 3D 20 20      2270             .BY '= ' $20
9137- EB 90         2280             .SI PLUS.
9139- 24 91         2290 EQUAL       .SI DOCOLON
913B- 63 91         2300             .SI MINUS
913D- BB 90         2310             .SI ZEROEQ
913F- AC 90         2320             .SI DOSEMICOLN
                    2330
9141- 01            2340 LESS.       .BY $01
9142- 3C 20 20      2350             .BY '< ' $20
9145- 33 91         2360             .SI EQUAL.
9147- 24 91         2370 LESS        .SI DOCOLON
9149- 63 91         2380             .SI MINUS
914B- CE 90         2390             .SI ZEROLESS
914D- AC 90         2400             .SI DOSEMICOLN
                    2410
914F- 01            2420 GREATER.    .BY $01
9150- 3E 20 20      2430             .BY '> ' $20
9153- 41 91         2440             .SI LESS.
9155- 24 91         2450 GREATER     .SI DOCOLON
9157- 94 90         2460             .SI SWAP
9159- 47 91         2470             .SI LESS
915B- AC 90         2480             .SI DOSEMICOLN
                    2490
                    2500
915D- 01            2510 MINUS.      .BY $01
915E- 2D 20 20      2520             .BY '- ' $20
9161- 4F 91         2530             .SI GREATER.
9163- 24 91         2540 MINUS       .SI DOCOLON
9165- DE 90         2550             .SI NEGATE
9167- F1 90         2560             .SI PLUS
9169- AC 90         2570             .SI DOSEMICOLN
                    2580
                    2590
916B- 03            2600 ABS.        .BY $03
916C- 41 42 53      2610             .BY 'ABS'
916F- 5D 91         2620             .SI MINUS.
9171- 24 91         2630 ABS         .SI DOCOLON
```

```
9173- 6C 90      2640            .SI DUP
9175- CE 90      2650            .SI ZEROLESS
9177- 1A 91      2660            .SI ZBRANCH
9179- 03         2670            .BY ABS1-=
                 2680
917A- DE 90      2690            .SI NEGATE
917C- AC 90      2700 ABS1       .SI DOSEMICOLN
                 2710
917E- 03         2720 MAX.       .BY $03
917F- 4D 41 58   2730            .BY 'MAX'
9182- 6B 91      2740            .SI ABS.
                 2750
9184- 24 91      2760 MAX        .SI DOCOLON
9186- 7C 90      2770            .SI OVER
9188- 7C 90      2780            .SI OVER
918A- 47 91      2790            .SI LESS
918C- 1A 91      2800            .SI ZBRANCH
918E- 03         2810            .BY MAX1-=
                 2820
918F- 94 90      2830            .SI SWAP
9191- 8C 90      2840 MAX1       .SI DROP
9193- AC 90      2850            .SI DOSEMICOLN
                 2860
9195- 03         2870 MIN.       .BY $03
9196- 4D 49 4E   2880            .BY 'MIN'
9199- 7E 91      2890            .SI MAX.
919B- 24 91      2900 MIN        .SI DOCOLON
919D- 7C 90      2910            .SI OVER
919F- 7C 90      2920            .SI OVER
91A1- 55 91      2930            .SI GREATER
91A3- 1A 91      2940            .SI ZBRANCH
91A5- 03         2950            .BY MIN1-=
                 2960
91A6- 94 90      2970            .SI SWAP
91A8- 8C 90      2980 MIN1       .SI DROP
91AA- AC 90      2990            .SI DOSEMICOLN
                 3000            .EN
```

ADJUSTABLE REAL TIME (SWISS) CLOCK - ERNST SCHUMACHER

Here is one of the finest clock programs we've ever seen for an almost unexpanded SYM-1. We say almost, because the program spills over four bytes beyond the first 1K of RAM. Of course, we could cheat a little and put some of the program into page one, and still .S2 and .L2 it in one segment. We don't approve of reading cassette dumps back in over the stack area, however, and certainly reading cassette dumps back in over the top of page zero is not possible, so page zero is out for multipage saves and loads.

We could not figure a way to trim away the four bytes. But, once you have added the additional 1K of RAM, there should be lots of room left to bring the SYM-1 up to the performance level of the inexpensive digital watches which also include a calendar, an audible, independently setable alarm, and a stopwatch/timer combo!

Many of our non-computer oriented friends, and even some of our computer science students, find it difficult to believe that digital timepieces are really general purpose microcomputers (or should they be called nanocomputers?) which are programmed in almost exactly the same way as the larger computers to which they are more accustomed. Showing them how the SYM can be programmed to do the same job, even though at much greater cost, and letting them look at a listing of the program could prove very instructive.


```
0010
0020 ;                                  Bern, 23 jul 1982
0030 ;Dear Lux:
0035
0040 ;Here is another SYM-clock. It works with the HKB but can
0050 ;be changed to CRT as indicated in SYMPHYSIS. To a Swiss,
0060 ;nostalgic over a once active watch industry, a clock
0070 ;must be regulateable and setable while it runs and to
0080 ;the limits of the precision of the given oscillator.
0090 ;That's in this program. A regulation to +/- 1 us/s gives
0100 ;not more than +/- 1 s in 11 days or +/- 30 s a year, much
0110 ;better than most 'quartz-watches' available today. This
0120 ;should be so, since the quartz in the SYM costs about as
0130 ;much as a complete digital watch.  You can set the
0140 ;flicker-free (!) keyboard display for hours, minutes,
0150 ;seconds, and 1/20 second without stopping the clock.
0155
0160 ;To change the display, press the keys
0165
0170 ;       4  5  for + or -  1/20 second  [reg $F7 not displ.]
0180 ;       6  7  for + or -  seconds      [reg $F6]
0190 ;       8  9  for + or -  minutes      [reg $F5]
0200 ;       A  B  for + or -  hours        [reg $F4]
0205
0210 ;When pressed continuously the digits whizz up or down
0220 ;through their ranges with correct over or underflow into
0230 ;the next digits. The 1/20 seconds are not displayed but
0240 ;can be examined in register $F7 after pressing 01 which
0250 ;brings the monitor back in while the clock ticks on. To
0260 ;jump back into the display, type G 0.
0265
0270 ;My SYM persistently shows a precision better than 1 sec
0280 ;in 11 days if it is not exposed to temperature changes
0290 ;of more than +/- 3 deg. centigrade for several days.
0300 ;+/- 1 us/s regulation is by pressing either 03 or 02
0310 ;on the HKB. The changes are not displayed but can be
0320 ;examined in regs. $F2 and $F3 [MIKSEC, LNIB].
0325
0330 ;The set-display interpreter is from lines 1470-2670.
0340 ;The clock regulation is explained from 3040-3650.
0350 ;Thanks for all you do !      Ernst Schumacher
0360
0370            .OS
0380            .LS
0390 START      .DE $0200
0400            .BA START
0405            .MC $9000
0410
0420 ;*************************************************************
0430 ;*                                                           *
0440 ;*        R E A L   T I M E   C L O C K  for SYM-1           *
0450 ;*                                                           *
0460 ;*     Clock can be regulated to 1 usec; display on          *
0470 ;*     SYM Hex-keyboard is HH.MM.SS; it can be set           *
0480 ;*     +/- .05 s, +/- 1 s, +/- 1 min, and +/- 1 hour         *
0490 ;*               while the clock is running.                 *
0500 ;*          E.S. 23 jul 1982    CH-3000 Bern 9               *
0510 ;*                                                           *
0520 ;*************************************************************
0530
0540
0550 ;          DEFINITIONS
0560
0570 IRQVEC     .DE $A67E
0580 DISBUF     .DE $A640
```

```
                    0590 DISBUF2    .DE $A641
                    0600 DISBUF4    .DE $A643
                    0610 ACCESS     .DE $8BB6
                    0620 SCAND      .DE $8906
                    0630 LRNKEY     .DE $892C
                    0640 MIKSEC     .DE $00F2
                    0650 LNIB       .DE $00F3
                    0660 HOUR       .DE $00F4
                    0670 MIN        .DE $00F5
                    0680 SEC        .DE $00F6
                    0690 COUNT      .DE $00F7
                    0700 NIBASC     .DE $8309
                    0710 ASCIM1     .DE $8BEE
                    0720 SEGSM1     .DE $8C28
                    0730 T1LL       .DE $A006
                    0740 T1CH       .DE $A005
                    0750 ACR        .DE $A00B
                    0760 IER        .DE $A00E
                    0770 ZERO       .DE $0000
                    0780
                    0790 ;          INITIALIZATION
                    0800
0200- 20 98 03      0810 CSTART     JSR INICLCK  ;init clock-routine
0203- 20 B6 8B      0820 WSTART     JSR ACCESS   ;come here after clck stops
0206- A9 B7         0830            LDA #L,CLOCK         ;set interrupt
0208- 8D 7E A6      0840            STA IRQVEC
020B- A9 03         0850            LDA #H,CLOCK         ;vector for clock
020D- 8D 7F A6      0860            STA IRQVEC+1
0210- A9 4C         0870            LDA #$4C     ;put a JMP at
0212- 85 00         0880            STA *ZERO    ; zero location for
0214- A9 03         0890            LDA #L,WSTART        ; finding WSTART by
0216- 85 01         0900            STA *ZERO+1  ; G 0
0218- A9 02         0910            LDA #H,WSTART        ; after a call to
021A- 85 02         0920            STA *ZERO+2  ; monitor
021C- D8            0930            CLD
                    0940 ;           start display from right to left
021D- A0 00         0950 LOOP1      LDY #00      ;initial pointer into DISBUF
021F- A5 F4         0960            LDA *HOUR
0221- 20 78 03      0970            JSR OUTBT
0224- A5 F5         0980            LDA *MIN
0226- 20 78 03      0990            JSR OUTBT
0229- A5 F6         1000            LDA *SEC
022B- 20 78 03      1010            JSR OUTBT
022E- AD 41 A6      1020            LDA DISBUF2
0231- 09 80         1030            ORA #$80     ;set period
0233- 8D 41 A6      1040            STA DISBUF2
0236- AD 43 A6      1050            LDA DISBUF4
0239- 09 80         1060            ORA #$80     ;set second period
023B- 8D 43 A6      1070            STA DISBUF4
023E- 20 06 89      1080            JSR SCAND
                    1090
                    1100 ;          DISPLAY ROUTINE
                    1110
0241- A0 04         1120 LOOP       LDY #4       ;display routine HKB, flicker-free
0243- A9 14         1130            LDA #$14     ;updates display only when digits
0245- C5 F7         1140            CMP *COUNT   ; change; writes seconds first
0247- D0 2B         1150            BNE LIGHT
0249- A5 F6         1160            LDA *SEC
024B- 20 78 03      1170            JSR OUTBT
024E- A0 02         1180            LDY #$02
0250- A5 F6         1190            LDA *SEC
0252- D0 20         1200            BNE LIGHT
0254- A5 F5         1210            LDA *MIN
0256- 20 78 03      1220            JSR OUTBT
```



```
0259- AD 43 A6      1230            LDA DISBUF4
025C- 09 80         1240            ORA #$80
025E- 8D 43 A6      1250            STA DISBUF4
0261- A0 00         1260            LDY #00
0263- A5 F5         1270            LDA *MIN
0265- D0 0D         1280            BNE LIGHT
0267- A5 F4         1290            LDA *HOUR
0269- 20 78 03      1300            JSR OUTBT
026C- AD 41 A6      1310            LDA DISBUF2
026F- 09 80         1320            ORA #$80
0271- 8D 41 A6      1330            STA DISBUF2
0274- 20 06 89      1340 LIGHT      JSR SCAND    ;scan the display and watch
0277- F0 C8         1350            BEQ LOOP     ; for key down ? no, continue
0279- 20 2C 89      1360            JSR LRNKEY   ;yes. Wait with
027C- A2 20         1370            LDX #$20     ; debounce loop to
027E- A0 FF         1380 LP1        LDY #$FF     ; prevent multiple operations.
0280- EA            1390 LP2        NOP  ;         It is 82 ms, long enough
0281- EA            1400            NOP  ;         to make sure that one clock-
0282- EA            1410            NOP  ;         cycle has gone before new
0283- 88            1420            DEY  ;         changes are made.
0284- D0 FA         1430            BNE LP2
0286- CA            1440            DEX
0287- D0 F5         1450            BNE LP1      ;end debounce loop
                    1460
                    1470 ;          CMD INTERPRETER HKB
                    1480
0289- C9 31         1490            CMP #$31     ;key 01: SYM-1 monitor warm entry
028B- F0 3A         1500            BEQ WARM
028D- C9 32         1510            CMP #$32     ;key 02: clock faster -1 usec/sec
028F- F0 39         1520            BEQ FASTER
0291- C9 33         1530            CMP #$33     ;key 03: clock slower +1 usec/sec
0293- F0 45         1540            BEQ SLOWER
0295- C9 34         1550            CMP #$34     ;key 04: set display +1/20 sec
0297- F0 51         1560            BEQ PLUS20
0299- C9 35         1570            CMP #$35     ;key 05: set display -1/20 sec
029B- F0 58         1580            BEQ MINUS20
029D- C9 36         1590            CMP #$36     ;key 06: set display +1 sec
029F- F0 65         1600            BEQ PLUSEC
02A1- C9 37         1610            CMP #$37     ;key 07: set display -1 sec
02A3- D0 03         1620            BNE PLUMN
02A5- 4C 38 03      1630            JMP MINUSEC
                    1635
02A8- C9 38         1640 PLUMN      CMP #$38     ;key 08: set display +1 min
02AA- D0 03         1650            BNE MINMN
02AC- 4C 68 03      1660            JMP PLUMIN
                    1665
02AF- C9 39         1670 MINMN      CMP #$39     ;key 09: set display -1 min
02B1- D0 03         1680            BNE PLUHR
02B3- 4C 6C 03      1690            JMP MINMIN
                    1695
02B6- C9 41         1700 PLUHR      CMP #$41     ;key 0A: set display +1 hour
02B8- D0 03         1710            BNE MINHR
02BA- 4C 70 03      1720            JMP PLUHOR
                    1725
02BD- C9 42         1730 MINHR      CMP #$42     ;key 0B: set display -1 hour
02BF- D0 03         1740            BNE LOP      ;for all other keys depressed
02C1- 4C 74 03      1750            JMP MINHOR
                    1755
02C4- 4C 41 02      1760 LOP        JMP LOOP
                    1765
02C7- 4C 03 80      1770 WARM       JMP $8003    ;back to the display by .G 0
                    1775
02CA- A5 F2         1780 FASTER     LDA *MIKSEC  ;set clock 1 us/s faster
02CC- D0 06         1790            BNE LOPQ     ;go do it
```

```
02CE- A9 14     1800            LDA #$14     ;to prevent underflow
02D0- 85 F2     1810            STA *MIKSEC  ; load jiffy-count and
02D2- C6 F3     1820            DEC *LNIB    ; adjust LNIB, keeping the time
02D4- C6 F2     1830 LOPQ       DEC *MIKSEC  ;now regulate clock
02D6- 10 EC     1840            BPL LOP      ;normally; but an interrupt could
02D8- 30 0C     1850            BMI LOPP     ; have lowered MIKSEC to FF !
02DA- A5 F2     1860 SLOWER     LDA *MIKSEC  ;set clock 1 us/s slower
02DC- C9 14     1870            CMP #$14     ;is it below max jiffy ?
02DE- 90 06     1880            BCC LOPP     ;yes, go on as usual
02E0- A9 00     1890            LDA #0       ;no, reduce it to 0
02E2- 85 F2     1900            STA *MIKSEC  ; put it there and adjust
02E4- E6 F3     1910            INC *LNIB    ; LNIB, which is equivalent time
02E6- E6 F2     1920 LOPP       INC *MIKSEC  ;one more in 20 jiffies with +1 us
02E8- 10 DA     1930            BPL LOP      ;always
02EA- C6 F7     1940 PLUS20     DEC *COUNT   ;make it one shorter
02EC- D0 D6     1950            BNE LOP      ; and continue
02EE- A9 14     1960            LDA #$14     ;spec.treatment if zero
02F0- 85 F7     1970            STA *COUNT   ; full jiffy but
02F2- 4C 06 03  1980            JMP PLUSEC   ; one sec more
                1985
02F5- A5 F7     1990 MINUS20    LDA *COUNT
02F7- C9 14     2000            CMP #$14
02F9- F0 04     2010            BEQ LOPN     ;spec. treatment
02FB- E6 F7     2020            INC *COUNT   ;make jiffy 1 more to count to 0
02FD- D0 C5     2030            BNE LOP      ; and go back
02FF- A9 01     2040 LOPN       LDA #$1      ;one more, but at
0301- 85 F7     2050            STA *COUNT
0303- 4C 38 03  2060            JMP MINUSEC  ;one sec less
                2065
0306- F8        2070 PLUSEC     SED
0307- A9 01     2080            LDA #01
0309- 18        2090            CLC
030A- 65 F6     2100            ADC *SEC
030C- 85 F6     2110            STA *SEC
030E- C9 60     2120            CMP #$60
0310- D0 22     2130            BNE EXIT
0312- A9 00     2140            LDA #0
0314- 85 F6     2150            STA *SEC
0316- A9 01     2160 PLUSMN     LDA #01
0318- 18        2170            CLC
0319- 65 F5     2180            ADC *MIN
031B- 85 F5     2190            STA *MIN
031D- C9 60     2200            CMP #$60
031F- D0 13     2210            BNE EXIT
0321- A9 00     2220            LDA #0
0323- 85 F5     2230            STA *MIN
0325- A9 01     2240 PLUSHR     LDA #01
0327- 18        2250            CLC
0328- 65 F4     2260            ADC *HOUR
032A- 85 F4     2270            STA *HOUR
032C- C9 24     2280            CMP #$24
032E- D0 04     2290            BNE EXIT
0330- A9 00     2300            LDA #0
0332- 85 F4     2310            STA *HOUR
0334- D8        2320 EXIT       CLD
0335- 4C 1D 02  2330            JMP LOOP1
                2335
0338- F8        2340 MINUSEC    SED
0339- A5 F6     2350            LDA *SEC
033B- 38        2360            SEC
033C- E9 01     2370            SBC #01
033E- 85 F6     2380            STA *SEC
0340- C9 99     2390            CMP #$99
0342- D0 F0     2400            BNE EXIT
0344- A9 59     2410            LDA #$59
```



```
0346- 85 F6     2420            STA *SEC
0348- A5 F5     2430 MINIMN     LDA *MIN
034A- 38        2440            SEC
034B- E9 01     2450            SBC #01
034D- 85 F5     2460            STA *MIN
034F- C9 99     2470            CMP #$99
0351- D0 E1     2480            BNE EXIT
0353- A9 59     2490            LDA #$59
0355- 85 F5     2500            STA *MIN
0357- A5 F4     2510 MINIHR     LDA *HOUR
0359- 38        2520            SEC
035A- E9 01     2530            SBC #01
035C- 85 F4     2540            STA *HOUR
035E- C9 99     2550            CMP #$99
0360- D0 D2     2560            BNE EXIT
0362- A9 23     2570            LDA #$23
0364- 85 F4     2580            STA *HOUR
0366- D0 CC     2590            BNE EXIT
0368- F8        2600 PLUMIN     SED
0369- 4C 16 03  2610            JMP PLUSMN
                2615
036C- F8        2620 MINMIN     SED
036D- 4C 48 03  2630            JMP MINIMN
                2635
0370- F8        2640 PLUHOR     SED
0371- 4C 25 03  2650            JMP PLUSHR
                2655
0374- F8        2660 MINHOR     SED
0375- 4C 57 03  2670            JMP MINIHR
                2680
0378- 48        2690 OUTBT      PHA  ;save display byte
0379- 4A        2700            LSR A
037A- 4A        2710            LSR A
037B- 4A        2720            LSR A
037C- 4A        2730            LSR A
037D- 20 81 03  2740            JSR NBASO1
0380- 68        2750            PLA
0381- 20 09 83  2760 NBASO1     JSR NIBASC
0384- A2 0A     2770 OUTDS      LDX #$0A
0386- DD EE 8B  2780 OUD2       CMP ASCIM1,X          ;  in ASCII-table
0389- F0 05     2790            BEQ GETSGS
038B- CA        2800            DEX
038C- D0 F8     2810            BNE OUD2
038E- F0 07     2820            BEQ EXITOT
0390- BD 28 8C  2830 GETSGS     LDA SEGSM1,X          ;segment-table for numbers
0393- 99 40 A6  2840            STA DISBUF,Y
0396- C8        2850            INY  ;bump pointer into DISBUF
0397- 60        2860 EXITOT     RTS
                2870
0398- A9 10     2880 INICLCK    LDA #$10      ;init counter for start
039A- 85 F7     2890            STA *COUNT
039C- 8D 0B A0  2900            STA ACR       ;set bits 7,6 low in aux.ctr.reg.
039F- A9 C0     2910            LDA #$C0      ;set bits 7,6 high in
03A1- 8D 0E A0  2920            STA IER       ; interrupt enable reg. timer1
03A4- A9 31     2930            LDA #$31      ;init. low. nib. of timer1 and
03A6- 85 F3     2940            STA *LNIB     ;save [assume 1.000000 MHz qrtz]
03A8- A9 0A     2950            LDA #$0A      ;init. midway between
03AA- 85 F2     2960            STA *MIKSEC   ; 0 and 14 hex
03AC- A9 24     2970            LDA #$24      ;1st loop of timer1 shorter
03AE- 8D 06 A0  2980            STA T1LL
03B1- A9 C3     2990            LDA #$C3      ; and start w. hi nib. of
03B3- 8D 05 A0  3000            STA T1CH      ; timer1 for 49957 usec
03B6- 60        3010            RTS
```

(continued to page 13/14-29)

```
                    0010            ; *** 2758/2716/2732 ***
                    0020            ; EPROM PROGRAMMER FOR SYM-1
                    0030            ; BY PETER G. FONG SAM
                    0040            ; AND PAUL L. BEAUPRE
                    0050
                    0060            ; SINGLE LETTER COMMANDS ARE USED.
                    0070            ; TYPE IN LETTER COMMANDS FOLLOWED
                    0080            ; BY EPROM TYPE, I.E. 2716, AND THEN
                    0090            ; MEMORY STARTING ADDRESS, FOLLOWED BY
                    0100            ; MEMORY ENDING ADDRESS AND THEN FOLLOWED
                    0110            ; BY A CR. ALL ENTRIES ARE TO BE SEPARATED
                    0120            ; BY COMMAS AS PER THE SYM-1 ENTRY MODE
                    0130
                    0140            ; >>> COMMANDS <<<
                    0150
                    0160            ; B = BLANK TEST
                    0170            ; C = COPY EPROM TO MEMORY SPECIFIED
                    0180            ; L = LIST EPROM BY LINES SPECIFIED
                    0190            ; P = PROGRAM EPROM FROM MEMORY SPECIFIED
                    0200            ; V = VERIFY CONTENTS OF EPROM TO MEMORY
                    0210            ;     LOCATIONS SPECIFIED
                    0220
                    0230            ; RETURN KEY = RETURN TO MONITOR
                    0240            ; BREAK = BREAK FROM LIST OR PROGRAM ONLY
                    0250
                    0260 ACCESS     .DE $8B86
                    0270 CRLF       .DE $834D
                    0280 EPROM      .DE $A646
                    0290 ERMSG      .DE $8171
                    0300 INCHAR     .DE $8A1B
                    0310 INSTAT     .DE $8386
                    0320 LSTCOM     .DE $A657
                    0330 MONITR     .DE $8000
                    0340 OUTBYT     .DE $82FA
                    0350 OUTCHR     .DE $8A47
                    0360 OUTQM      .DE $8320
                    0370 PAD        .DE $A001
                    0380 PADD       .DE $A003
                    0390 PARNR      .DE $A649
                    0400 PBD        .DE $A000
                    0410 PBDD       .DE $A002
                    0420 P1H        .DE $A64F
                    0430 P1L        .DE $A64E
                    0440 P2H        .DE $A64D
                    0450 P2L        .DE $A64C
                    0460 P2SCR      .DE $829C
                    0470 P3H        .DE $A64B
                    0480 P3L        .DE $A64A
                    0490 SIZE       .DE $A647
                    0500 SPACE      .DE $8342
                    0510 STATUS     .DE $A407
                    0520 STOCOM     .DE $8120
                    0530
                    0540 TEMP1      .DE $FE
                    0550 TEMP2      .DE $FF
                    0560
                    0570 T1024      .DE $A41F
                    0580
                    0590            .BA $9000     ; OR WHEREVER
                    0600            .OS
                    0610
9000- 20 86 8B      0620 START      JSR ACCESS
9003- A9 FF         0630 RESET      LDA #$FF
9005- 8D 02 A0      0640            STA PBDD
9008- A9 A0         0650            LDA #$A0
900A- 8D 00 A0      0660            STA PBD
```



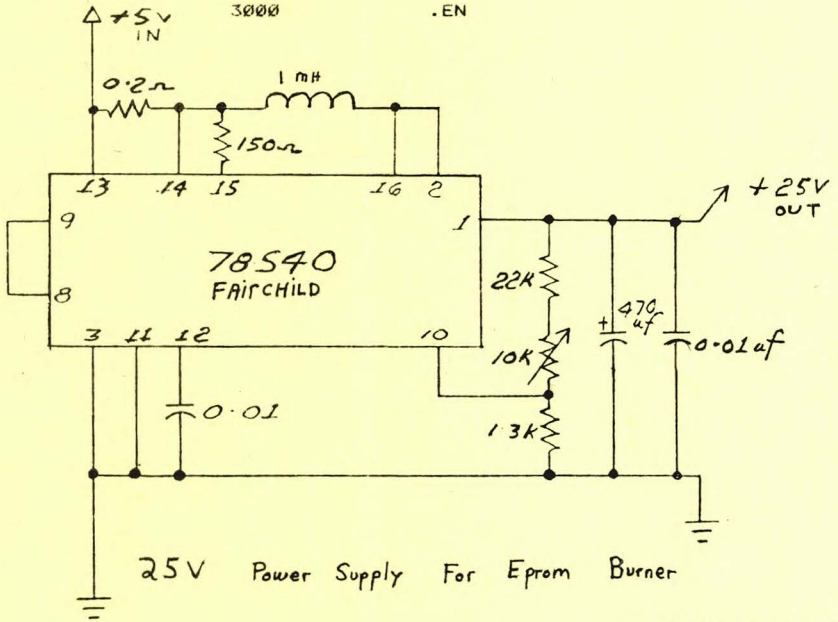
```
900D- 20 4D 83      0670 PROMPT     JSR CRLF
9010- A9 2A         0680            LDA #'*
9012- 20 47 8A      0690            JSR OUTCHR
9015- 20 20 83      0700            JSR OUTQM
9018- 20 42 83      0710            JSR SPACE
901B- 20 1B 8A      0720 INCOM      JSR INCHAR
901E- C9 0D         0730            CMP #$0D
9020- D0 03         0740            BNE OKCOM
9022- 4C 00 80      0750            JMP MONITR
                    0760
9025- 20 20 81      0770 OKCOM      JSR STOCOM
9028- C9 0D         0780            CMP #$0D
902A- D0 39         0790            BNE OUTERR
902C- AD 49 A6      0800            LDA PARNR
902F- 0A            0810            ASL A
9030- A8            0820            TAY
9031- B9 49 A6      0830            LDA PARNR,Y
9034- C9 27         0840            CMP #$27
9036- D0 2D         0850            BNE OUTERR
9038- 88            0860            DEY
9039- B9 49 A6      0870            LDA PARNR,Y
903C- C9 16         0880            CMP #$16
903E- D0 0C         0890            BNE SIZE4K
9040- A9 08         0900 SIZE2K     LDA #8
9042- 8D 47 A6      0910            STA SIZE
9045- A9 00         0920            LDA #0
9047- 8D 46 A6      0930            STA EPROM
904A- F0 1F         0940            BEQ CHECK
904C- C9 32         0950 SIZE4K     CMP #$32
904E- D0 0A         0960            BNE SIZE1K
9050- A9 10         0970            LDA #$10
9052- 8D 46 A6      0980            STA EPROM
9055- 8D 47 A6      0990            STA SIZE
9058- D0 11         1000            BNE CHECK
905A- C9 58         1010 SIZE1K     CMP #$58
905C- D0 07         1020            BNE OUTERR
905E- A9 04         1030            LDA #4  ".
9060- 8D 47 A6      1040            STA SIZE
9063- D0 E0         1050            BNE SIZE2K+5
9065- 20 73 81      1060 OUTERR     JSR ERMSG+2
9068- 4C 0D 90      1070            JMP PROMPT
                    1080
906B- 20 9C 82      1090 CHECK      JSR P2SCR
906E- AD 49 A6      1100            LDA PARNR
9071- C9 01         1110            CMP #1
9073- D0 09         1120            BNE THREE
9075- AD 57 A6      1130            LDA LSTCOM
9078- C9 42         1140            CMP #'B
907A- D0 E9         1150            BNE OUTERR
907C- F0 22         1160            BEQ BLANK
907E- C9 03         1170 THREE      CMP #3
9080- D0 E3         1180            BNE OUTERR
9082- AD 57 A6      1190            LDA LSTCOM
9085- C9 43         1200            CMP #'C
9087- D0 02         1210            BNE LISTPR
9089- F0 61         1220            BEQ COPY
908B- C9 4C         1230 LISTPR     CMP #'L
908D- D0 03         1240            BNE PROG
908F- 4C 0D 91      1250            JMP LIST
                    1260
9092- C9 50         1270 PROG       CMP #'P
9094- D0 03         1280            BNE VER
9096- 4C 50 91      1290            JMP PROGRM
9099- C9 56         1300 VER        CMP #'V
909B- D0 C8         1310            BNE OUTERR
909D- 4C 89 91      1320            JMP VERIFY
```

```
                    1330
                    1340            ; BLANK TEST
                    1350
90A0- A9 00         1360 BLANK      LDA #0
90A2- 8D 03 A0      1370            STA PADD
90A5- A8            1380            TAY
90A6- AD 46 A6      1390            LDA EPROM
90A9- 8D 00 A0      1400            STA PBD
90AC- 20 CD 91      1410            JSR DELAY     ;ALLOWS RELAYS TO SETTLE
90AF- A9 FF         1420            LDA #$FF
90B1- CD 01 A0      1430 CHKBYT     CMP PAD
90B4- D0 1F         1440            BNE ERROR
90B6- EE 00 A0      1450            INC PBD
90B9- CE 00 A0      1460            DEC PBD
90BC- C8            1470            INY
90BD- D0 F2         1480            BNE CHKBYT
90BF- E8            1490            INX
90C0- EC 47 A6      1500            CPX SIZE
90C3- D0 EC         1510            BNE CHKBYT
90C5- 20 4D 83      1520 DONE       JSR CRLF
90C8- A9 4F         1530            LDA #'O
90CA- 20 47 8A      1540            JSR OUTCHR
90CD- A9 4B         1550            LDA #'K
90CF- 20 47 8A      1560            JSR OUTCHR
90D2- 4C 08 90      1570            JMP RESET+5
                    1580
90D5- 20 4D 83      1590 ERROR      JSR CRLF
90D8- 8A            1600            TXA
90D9- 20 FA 82      1610            JSR OUTBYT
90DC- 98            1620            TYA
90DD- 20 FA 82      1630            JSR OUTBYT
90E0- 20 42 83      1640            JSR SPACE
90E3- AD 01 A0      1650            LDA PAD
90E6- 20 FA 82      1660            JSR OUTBYT
90E9- 4C 08 90      1670            JMP RESET+5
                    1680
                    1690            ; COPY
                    1700
90EC- A9 00         1710 COPY       LDA #0
90EE- 8D 03 A0      1720            STA PADD
90F1- AD 46 A6      1730            LDA EPROM
90F4- 8D 00 A0      1740            STA PBD
90F7- 20 CD 91      1750            JSR DELAY
90FA- AD 01 A0      1760 GETCHR     LDA PAD
90FD- 81 FE         1770            STA (TEMP1,X)
90FF- EE 00 A0      1780            INC PBD
9102- CE 00 A0      1790            DEC PBD
9105- 20 DA 91      1800            JSR COMPAR
9108- 90 F0         1810            BCC GETCHR
910A- 4C C5 90      1820            JMP DONE
                    1830
                    1840            ; LIST
                    1850
910D- A9 00         1860 LIST       LDA #0
910F- 8D 03 A0      1870            STA PADD
9112- AD 46 A6      1880            LDA EPROM
9115- 8D 00 A0      1890            STA PBD
9118- 20 CD 91      1900            JSR DELAY
911B- 20 F4 91      1910 NEWLIN     JSR LOOK
911E- 20 4D 83      1920            JSR CRLF
9121- A0 00         1930            LDY #0
9123- AD FF 00      1940            LDA TEMP2
9126- 20 FA 82      1950            JSR OUTBYT
9129- AD FE 00      1960            LDA TEMP1
912C- 20 FA 82      1970            JSR OUTBYT
912F- 20 42 83      1980            JSR SPACE
```



```
9132- 20 42 83      1990 DATA       JSR SPACE
9135- AD 01 A0      2000            LDA PAD
9138- 20 FA 82      2010            JSR OUTBYT
913B- EE 00 A0      2020            INC PBD
913E- CE 00 A0      2030            DEC PBD
9141- 20 DA 91      2040            JSR COMPAR
9144- 90 03         2050            BCC CKCNTR
9146- 4C 08 90      2060            JMP RESET+5
                    2070
9149- C8            2080 CKCNTR     INY
914A- C0 10         2090            CPY #$10
914C- D0 E4         2100            BNE DATA
914E- F0 CB         2110            BEQ NEWLIN
                    2120
                    2130            ; PROGRAM
                    2140
9150- A9 FF         2150 PROGRM     LDA #$FF
9152- 8D 03 A0      2160            STA PADD
9155- AD 46 A6      2170            LDA EPROM
9158- F0 07         2180            BEQ NOT4K
915A- A9 1A         2190            LDA #$1A
915C- 8D 00 A0      2200            STA PBD
915F- D0 05         2210            BNE GO
9161- A9 0C         2220 NOT4K      LDA #$C
9163- 8D 00 A0      2230            STA PBD
9166- 20 CD 91      2240 GO         JSR DELAY
9169- 20 F4 91      2250 BURN       JSR LOOK
916C- A1 FE         2260            LDA (TEMP1,X)
916E- 8D 01 A0      2270            STA PAD
9171- EA            2280            NOP
9172- EA            2290            NOP
9173- CE 00 A0      2300            DEC PBD
9176- 20 CD 91      2310 TIMOUT     JSR DELAY
9179- EE 00 A0      2320            INC PBD
917C- 20 DA 91      2330            JSR COMPAR
917F- 90 E8         2340            BCC BURN
9181- A9 80         2350            LDA #$80
9183- 8D 00 A0      2360            STA PBD
9186- 20 9C 82      2370            JSR P2SCR
                    2380
                    2390            ; VERIFY
                    2400
9189- A9 00         2410 VERIFY     LDA #0
918B- 8D 03 A0      2420            STA PADD
918E- AD 46 A6      2430            LDA EPROM
9191- 8D 00 A0      2440            STA PBD
9194- 20 CD 91      2450            JSR DELAY
9197- AD 01 A0      2460 NEXBYT     LDA PAD
919A- C1 FE         2470            CMP (TEMP1,X)
919C- D0 0E         2480            BNE ERRPTR
919E- EE 00 A0      2490            INC PBD
91A1- CE 00 A0      2500            DEC PBD
91A4- 20 DA 91      2510            JSR COMPAR
91A7- 90 EE         2520            BCC NEXBYT
91A9- 4C C5 90      2530            JMP DONE
                    2540
                    2550            ; DISPLAY ERROR AS MEMORY LOCATION,
                    2560            ; MEMORY DATA, EPROM CONTENTS
                    2570
91AC- 20 4D 83      2580 ERRPTR     JSR CRLF
91AF- A5 FF         2590            LDA *TEMP2
91B1- 20 FA 82      2600            JSR OUTBYT
91B4- A5 FE         2610            LDA *TEMP1
91B6- 20 FA 82      2620            JSR OUTBYT
91B9- 20 42 83      2630            JSR SPACE
91BC- A1 FE         2640            LDA (TEMP1,X)
```

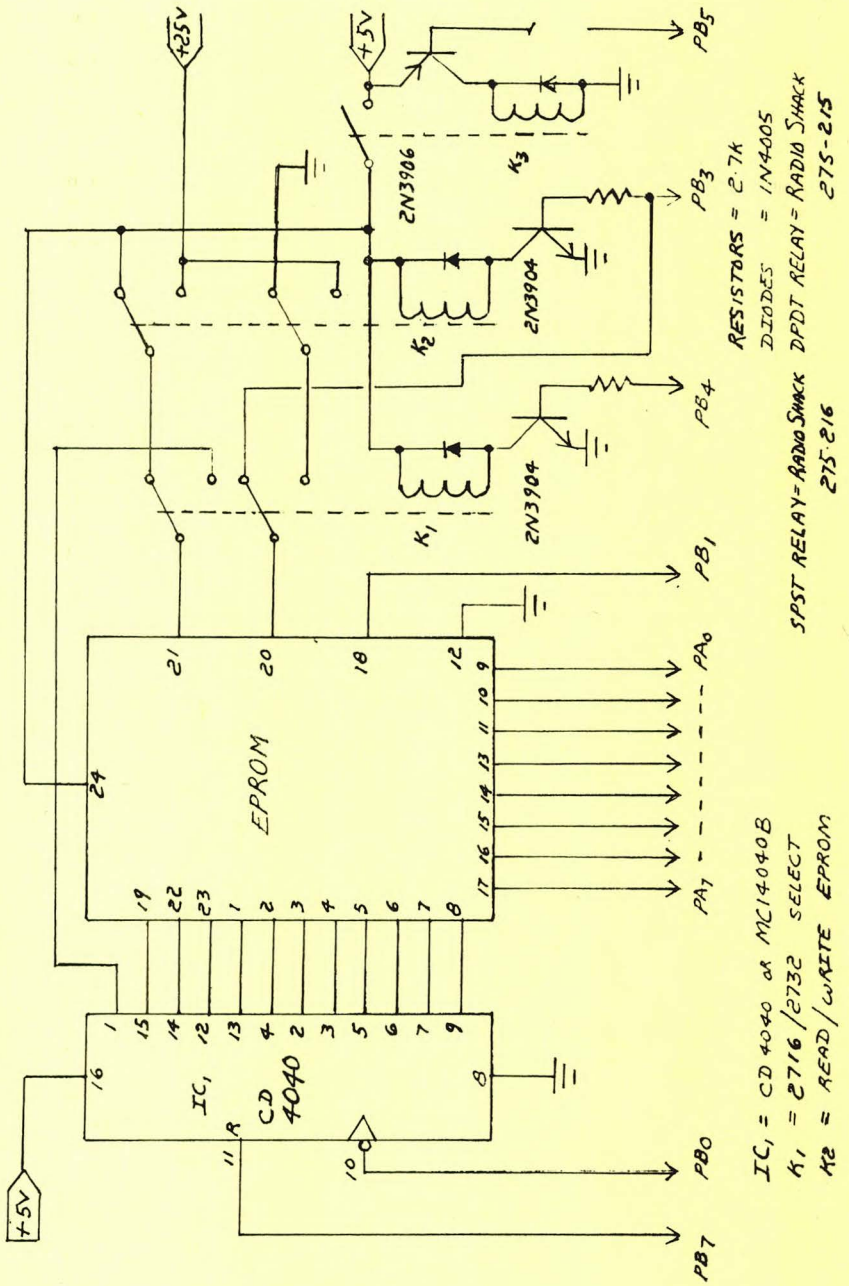
```
91BE- 20 FA 82  2650            JSR OUTBYT
91C1- 20 42 83  2660            JSR SPACE
91C4- AD 01 A0  2670            LDA PAD
91C7- 20 FA 82  2680            JSR OUTBYT
91CA- 4C 08 90  2690            JMP RESET+5
                2700
91CD- A9 2F     2710 DELAY      LDA #$2F DELAY FOR 50 MS
91CF- 8D 1F A4  2720            STA T1024
91D2- AD 07 A4  2730            LDA STATUS
91D5- 10 FB     2740            BPL DELAY+5
91D7- A2 00     2750            LDX #0
91D9- 60        2760            RTS
                2770
91DA- A5 FE     2780 COMPAR     LDA *TEMP1
91DC- CD 4A A6  2790            CMP P3L
91DF- F0 08     2800            BEQ TESTHI
91E1- E6 FE     2810 UPLOW      INC *TEMP1
91E3- D0 0D     2820            BNE OUT
91E5- E6 FF     2830            INC *TEMP2
91E7- D0 09     2840            BNE OUT
91E9- A5 FF     2850 TESTHI     LDA *TEMP2
91EB- CD 4B A6  2860            CMP P3H
91EE- D0 F1     2870            BNE UPLOW
91F0- 38        2880            SEC
91F1- 60        2890            RTS
                2900
91F2- 18        2910 OUT        CLC
91F3- 60        2920            RTS
                2930
91F4- 20 86 83  2940 LOOK       JSR INSTAT
91F7- B0 01     2950            BCS CONT
91F9- 60        2960            RTS
                2970
91FA- 4C 08 90  2980 CONT       JMP RESET+5
                2990
                3000            .EN
```



25V Power Supply For Eprom Burner

SYM-1 2758 · 2716 · 2732 Eprom Programmer

All Ports Are Hooked up To U-25 on SYM



RESISTORS = 2.7K
DIODES = 1N4005
SPST RELAY = RADIO SHACK 275-216
DPDT RELAY = RADIO SHACK 275-215

IC1 = CD 4040 or MC14040B
K1 = 2716/2732 SELECT
K2 = READ/WRITE EPROM

ASCII TEXT FILE FOR JEFF LAVIN'S PRINTER PROGRAM: "SUSAN" (ABRIDGED)

```
     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0200 0D 0A 0D 0A 0D 0A 0D 0A 20 20 20 20 20 20 20 20,5C  ........
0210 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,5C
0220 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,5C
0230 20 2E 2E 20 20 20 2E 2E 2E 0D 0A 20 20 20 20 20,79   ..   .....
0240 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,79
0250 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,79
0260 20 41 49 4D 4D 59 49 41 4D 4D 59 59 49 49 2E 2E,DA   AIMMYIAMMYYII..
0270 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20,B1  ..
0280 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,B1
0290 20 20 20 20 20 20 20 20 41 49 48 4D 4D 4D 48 4A 49,25         AIHMMMHJI
02A0 4D 4D 49 4C 4C 53 49 49 41 0D 0A 20 20 20 20 20 20,7D  MMILLSIIA..
02B0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,7D
02C0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 41 49,C7                AI
02D0 48 50 50 2F 3F 2F 20 2F 24 24 2F 50 56 59 4D 4A,A8  HPP/?/ /$$/PVYMJ
02E0 48 49 49 41 2E 0D 0A 20 20 20 20 20 20 20 20 20,28  HIIA...
02F0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,28

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0300 20 20 20 20 20 20 20 20 2E 41 50 2F 2F 24 20 3F,C8          .AP//$ ?
0310 20 2F 2F 24 24 2F 20 2F 2F 2F 2F 56 4D 4D 48 49,1A   //$$/ ////VMMHI
0320 41 2E 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20,20  A...
0330 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,20
0340 20 20 20 20 41 2F 2F 2F 24 2F 24 3F 3F 2F 2F 2F,F0      A///$/$??///
0350 24 20 2F 2F 2F 2F 2F 2F 3F 2F 2F 56 4D 48 49 41,60  $ //////?//VMHIA
0360 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20,37  ..
0370 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,37
0380 20 41 2F 2F 2F 24 2F 24 2F 3F 2F 2F 2F 2F 20 2F,15   A///$/$/?//// /
0390 20 2F 2F 2F 2F 3F 2F 2F 2F 3F 2F 56 48 48 41 20,72   ////?///?/VHHA
03A0 2E 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20,57  ...
03B0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,57
03C0 20 41 2F 2F 2F 2F 24 2F 24 3F 3F 2F 2F 2F 20 2F,45   A////$/$??/// /
03D0 2F 2F 2F 2F 2F 2F 3F 2F 2F 3F 2F 2F 3F 56 4D 4D,C8  //////?//?//?VMM
03E0 41 20 2E 0D 0A 20 20 20 20 20 20 20 20 20 20 20,CE  A ...
03F0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,CE

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0400 20 20 20 2F 2F 2F 2F 24 2F 24 20 3F 3F 2F 2F 2F,8C    ////$/$ ??///
0410 20 2F 2F 2F 2F 2F 2F 3F 2F 2F 2F 2B 2F 3F 2F 2F,89   //////?///+/?//
0420 56 4D 48 49 59 4D 2E 2E 0D 0A 20 20 20 20 20 20,96  VMHIYM....
0430 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,96
0440 20 20 20 20 20 20 20 41 2F 2F 2F 2F 24 2F 41 4D,54        A////$/AM
0450 4D 4D 4D 4D 4D 48 48 48 41 2F 2F 3F 2F 2F 2F 3F,57  MMMMMHHHHA//?///?
0460 2F 2F 3F 2F 2F 2E 49 59 48 4D 4D 4D 4D 2E 0D 0A,E3  //?//.IYHMMMM...
0470 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,E3
0480 20 20 20 20 20 20 20 20 20 20 20 20 2E 48 2F 2F,37            .H//
0490 2F 2F 41 4D 4D 4D 48 48 48 4D 4D 4D 4D 48 48 48,A1  //AMMMHHHMMMMMHH
04A0 48 41 2F 2F 3F 2F 2F 3F 2F 2F 3F 24 56 48 48 48,53  HA//?//?//?$VHHH
04B0 59 59 49 2E 0D 0A 20 20 20 20 20 20 20 20 20 20,D3  YYI...
04C0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,D3
04D0 20 20 41 48 2F 2F 41 4D 4D 48 4D 48 48 4D 48 4D,DC    AH//AMMHMHHHMH
04E0 4D 4D 48 48 48 48 48 48 49 59 2F 2F 3F 2F 2F 3F,02  MMHHHHHHHIY//?//?
04F0 24 2F 2F 4D 48 48 48 4D 48 49 2E 0D 0A 20 20 20,2C  $//MHHHMHI...

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0500 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,2C
0510 20 20 20 20 20 20 20 20 20 20 20 2F 2F 41 48 4D 48,E8           //AHMH
0520 4D 49 49 4D 48 48 49 4D 4D 48 4D 59 48 4D 49,A0  MIIMHHIMMMHMYHMI
0530 49 4D 49 2F 2F 3F 24 2F 2F 2F 4D 4D 4D 48 48 49,8D  IMI//?$///MMMHHI
0540 49 49 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20,B6  II..
0550 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,B6
0560 20 2F 41 48 48 49 48 4D 49 48 48 48 4D 4D,00   /AHHIHMIHIHHHMM
0570 4D 4D 4D 48 48 48 4D 48 4D 4D 41 2F 24 2F 2F 2F,0F  MMMHHHHMHMA/$///
```



```
0580 4D 4D 4D 4D 48 4D 48 49 49 2E 0D 0A 20 20 20 20,77  MMMMHMHII...
0590 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,77
05A0 20 20 20 20 20 20 20 20 2E 4D 4D 59 59 48 49 49,CB          .MMYYHII
05B0 48 49 49 49 49 48 4D 4D 4D 4D 4D 4D 4D 48 48 4D,77  HIIIIHMMMMMMMHHM
05C0 4D 4D 4D 4D 4D 2F 2F 48 4D 4D 4D 48 4D 48 48 49,F3  MMMMM//HMMMHMHHI
05D0 49 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20,F3  I..
05E0 20 20 20 20 20 20 20 20 20 20 20 2E 41 48 48 48 48,C2            .AHHHH
05F0 49 4D 4D 4D 4D 3A 49 3A 3A 3A 3A 49 49 4D 4D 4D,23  IMMMM:I::::IIMMM

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0600 4D 4D 59 59 4D 4D 4D 4D 4D 4D 4D 4D 48 4D 2F 4D,E8  MMYYMMMMMMMMHM/M
0610 4D 48 48 48 48 4D 4D 48 0D 0A 20 20 20 20 20 20,E6  MHHHHMH..
0620 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,E6
0630 41 49 4A 4D 4D 48 2F 49 3A 4D 49 22 22 4D 4D 3A,FC  AIJMMH/I:MI""MM:
0640 2E 2E 3A 3A 3B 3B 41 59 4D 4D 4D 4D 3A 3A 3A 4D,0B  ..::;;AYMMMM:::M
0650 4D 4D 4D 2F 4D 4D 4D 59 48 49 48 48 49 2E 2E 0D,34  MMM/MMMYHIHHI...
0660 0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,1E  .
0670 20 20 20 20 20 20 41 4D 48 49 48 49 49 4D 4D 48,B9        AMHIHIIMMH
0680 49 49 20 49 22 4D 4D 4D 3A 20 2E 3A 3A 49 22 20,44  II I"MMM: .::I"
0690 4D 56 41 3A 3A 3A 2E 4D 4D 4D 4D 4D 59 4D 4D,C5  MVA:::.MMMMMMMYMM
06A0 48 49 49 49 49 49 2F 0D 0A 20 20 20 20 20 20 20,A0  HIIIII/..
06B0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 41 49,EA                AI
06C0 49 48 48 48 48 4D 4D 4D 4D 4D 49 49 20 22 56 2E,2C  IHHHHMMMMMII "V.
06D0 22 2E 20 3A 3A 3B 49 49 3B 3A 3A 3A 2E 2E 2E 4D,9D  ". :;;II;;:::...M
06E0 4D 4D 4D 4D 4D 4D 4D 48 48 49 49 49 49 49 0D,0F  MMMMMMMMHHIIIII.
06F0 0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,F9  .

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0700 20 20 20 20 20 20 56 49 48 48 49 48 48 4D 4D 4D,A8        VIHHIHHMMM
0710 4D 4D 49 27 20 27 2E 2E 27 20 20 3A 3A 3B 3B 3B,E1  MMI' '..'  :;;;;
0720 3A 3A 2E 2E 2E 3A 48 4D 4D 4D 4D 4D 4D 4D 4D,16  ::...:HMMMMMMMMM
0730 48 48 48 49 49 49 2E 0D 0A 20 20 20 20 20 20 20,EE  HHHIII...
0740 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,EE
0750 20 48 48 48 48 4D 4D 4D 4D 4D 27 20 20 2E 2E 2E,A0   HHHHMMMMM'  ...
0760 20 20 20 20 3A 3A 3A 3B 3B 3A 3A 3A 3A 3A 3A 56,4D,23     ::::::::::VM
0770 4D 4D 4D 4D 48 48 4D 4D 48 48 49 49 49 49 49 49,C7  MMMMHHMMHHIIIIII
0780 49 2E 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20,D5  I...
0790 20 20 20 20 20 20 20 20 20 20 20 20 48 49 48 4D,7B              HIHM
07A0 4D 4D 4D 4D 4D 41 20 20 20 2E 2E 3A 20 27 3A 3A,EE  MMMMMA   ..: ':::
07B0 3A 49 49 3A 3A 3A 3A 3A 3A 3A 3A 4D 4D 4D 4D 4D,0B  :II:::::::::MMMMM
07C0 48 48 48 48 48 49 49 48 48 48 48 48 2E 0D 0A 20,D2  HHHHHIIHHHHH...
07D0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,D2
07E0 20 20 20 20 20 20 20 48 48 49 48 4D 4D 4D 4D 4D,54          HHIHMMMMM
07F0 4D 3A 20 20 2E 3A 3A 27 22 3A 49 59 49 3A 49 49,F7  M:  .::'":IYI:II

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0800 3A 3A 3A 3A 3A 3A 4D 4D 4D 4D 4D 4D 4D 48 4D 4D,50  ::::::MMMMMMMHMM
0810 48 48 48 48 49 49 49 49 0D 0A 20 20 20 20 20 20,6B  HHHHIIII..
0820 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,6B
0830 20 20 48 48 48 48 48 4D 4D 4D 4D 41 20 20 3A,4F    HHHHHMMMMMA  :
0840 49 2E 20 20 2E 3A 3A 3A 3A 3A 4A 49 3A 3A 3A 3A,D1  I.  .:::::JI::::
0850 49 4D 49 4D 48 4D 4D 4D 48 49 48 4D 4D 4D 49 49,7E  IMIMHMMMHIHMMMII
0860 49 49 49 0D 0A 20 20 20 20 20 20 20 20 20 20 20,D0  III..
0870 20 20 20 20 20 20 20 20 20 20 20 20 20 56 48,2E                VH
0880 48 49 4D 4D 4D 4D 4D 4D 41 20 3A 3A 59 48 22 22,47  HIMMMMMMA ::YH""
0890 22 22 27 4D 50 3A 3A 3A 2E 2E 3A 4D 4D 59 4D 4D,20  ""'MP:::..:MMYMM
08A0 4D 4D 4D 4D 48 48 4D 4D 48 49 49 49 49 0D 0A 20,21  MMMMHHMMHIIII..
08B0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,21
08C0 20 20 20 20 20 20 20 20 48 48 48 48 49 59 4D 4D,7D          HHHHIYMM
08D0 4D 4D 4D 41 20 2E 2E 56 49 50 48 48 49 22 2E 3A,73  MMMA ..VIPHHI".:
08E0 3A 3A 3A 3A 41 4D 4D 4D 4D 4D 4D 4D 4D 48 49,E2  ::::AMMMMMMMMHI
08F0 4D 4D 3A 3A 3A 2E 2E 0D 0A 20 20 20 20 20 20 20,7D  MM:::....
```

```
     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0900 20 20 20 20 20 20 20 20 20 20 20 20 20 20 2E 3A,A5                .:
0910 48 4D 4D 4D 48 59 59 48 4D 4D 4D 56 4D 24 41 2E,33  HMMMHYYHMMMVM$A.
0920 20 20 20 22 22 3A 3A 3A 3A 3A 3A 3A 41 4D 4D 4D,95    "":::::::AMMM
0930 59 59 4D 4D 4D 4D 4D 4D 4D 4D 3A 3A 2E 2E 3A,06  YYMMMMMMMMMM::..:
0940 2E 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20,EB  ...
0950 20 20 20 20 20 20 20 20 20 48 56 20 41 48 4D 4D,EC            HV AHMM
0960 48 59 4D 4D 41 2F 56 4D 2F 24 3A 20 20 20 20 2E,75  HYMMA/VM/$:    .
0970 2E 3A 3A 3A 3A 49 49 4D 4D 4D 4D 4D 4D 4D 4D 4D,D2  .:::::IIMMMMMMMMMM
0980 4D 4D 4D 56 49 49 3A 3A 3A 2E 20 3A 2E 0D 0A 20,3C  MMMVII:::. :...
0990 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,3C
09A0 20 20 20 20 20 20 48 48 48 48 49 48 4D 4D 59 59 4D,26      HHHHIHMMYYM
09B0 4D 2F 2F 20 24 24 3A 3A 2E 2E 3A 3A 3A 3A 49 49,83  M// $$:::..::::II
09C0 49 49 49 49 4D 4D 4D 4D 4D 4D 4D 4D 4D 4D 49,3F  IIIIMMMMMMMMMMMI
09D0 49 49 49 3A 3A 3A 3A 2E 3A 0D 0A 20 20 20 20 20,21  III:::::.:..
09E0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,21
09F0 20 20 20 20 20 48 48 48 4D 2F 2F 48 4D 24 24 20,41       HHHM//HM$$

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0A00 2F 20 4D 3A 3A 3A 2E 2E 2E 3A 3A 3A 3A 49 49 49,D8  / M:::...::::III
0A10 49 59 59 59 59 59 59 48 48 49 49 49 49 49 49 3A,B7  IYYYYYYHHIIIIII:
0A20 3A 3A 3A 3A 3A 0D 0A 20 20 20 20 20 20 20 20 20,10  :::::..
0A30 20 20 20 20 20 20 20 20 20 20 20 20 20 20 2E 3A,38                .:
0A40 3A 3A 59 4D 2F 2F 56 53 4D 58 24 24 24 20 2F 49,02  ::YM//VSMX$$$ /I
0A50 49 3A 3A 2E 2E 2E 3A 3A 3A 3A 49 49 49 49 49 3A,D8  I:...:::IIIII:
0A60 2E 49 49 49 49 49 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A,B7  .IIII:::::::::
0A70 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20,8E  ..
0A80 20 20 20 20 20 20 2E 3A 3A 3A 2E 3A 3A 3A 2F 59,8E        .:::.:::/Y
0A90 2F 41 58 2F 2F 2F 2F 2F 24 4D 4D 49 49 49 3A 3A,4E  /AX/////$MMIII::
0AA0 3A 49 49 49 49 49 49 49 49 49 49 2E 49 3A 3A 3A,87  :IIIIIIIII.I:::
0AB0 3A 3A 3A 3A 3A 3A 3A 3A 49 49 0D 0A 20 20 20 20,80  ::::::::II..
0AC0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 2E 3A,A8                .:
0AD0 2E 2E 2E 3A 3A 3A 3A 2F 56 59 58 2F 2F 2F 2F 2F,3B  ...:::/VYX/////
0AE0 2F 48 48 2F 48 49 49 3A 3A 3A 2E 3A 3A 3A 3A 3A,01  /HH/HII:::.:::::
0AF0 3A 3A 3A 3A 3A 2E 2E 3A 3A 3A 3A 3A 3A 3A 3A 3A,89  :::::..:::::::::

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0B00 3A 49 49 49 3A 0D 0A 20 20 20 20 20 20 20 20 20,0F  :III:..
0B10 20 20 20 20 20 20 20 20 2E 2E 2E 2E 2E 3A 3A 3A,A3                .....:::
0B20 3A 2F 2F 2F 2F 2F 2F 2F 2F 2F 2F 2F 49 49 2F 2F,D2  :////////////II//
0B30 49 49 49 49 49 49 3A 3A 2E 3A 3A 3A 3A 3A 3A 3A,C0  IIIIII:.:.:::::::
0B40 3A 2E 3A 3A 3A 3A 3A 3A 49 3A 3A 3A 49 49 49 3A,90  :.:::::::I:::III:
0B50 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20,67  ..
0B60 20 20 2E 2E 2E 2E 2E 3A 3A 3A 49 3A 2F 2F 2F 2F,7A    .....:::I:////
0B70 2F 2F 2F 2F 2F 2F 2F 58 2F 2F 2F 49 49 49 49 49,15  ///////X///IIIII
0B80 49 3A 2E 2E 3A 3A 3A 3A 3A 3A 3A 2E 2E 3A 3A,94  I:..:::::::::..::
0B90 3A 3A 49 48 3A 3A 3A 3A 49 49 3A 0D 0A 20 20 20,C4  ::IH::::II:..
0BA0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 2E 2E,E0                ..
0BB0 20 3A 3A 3A 3A 49 49 2F 2F 2F 2F 2F 2F 2F 2F 2F,21   ::::II/////////
0BC0 2F 58 2F 2F 2F 2F 3A 3A 3A 49 49 49 3A 2E 2E 3A,BD  /X////:::III:..:
0BD0 3A 3A 3A 3A 3A 3A 3A 2E 2E 3A 3A 3A 3A 49 4D 3A,67  :::::::..::::IM:
0BE0 3A 3A 3A 49 49 3A 0D 0A 20 20 20 20 20 20 20 20,F8  :::II:..
0BF0 20 20 20 20 20 20 20 20 2E 2E 20 2E 3A 3A 3A 49,99          .. .:::I

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0C00 49 49 49 49 49 3A 2F 2F 2F 2F 2F 2F 58 2F 2F 2F,3F  IIIII://///X///
0C10 2F 3A 3A 3A 3A 49 49 3A 2E 3A 3A 3A 3A 3A 3A 3A,E6  /::::II:.:::::::
0C20 3A 3A 2E 2E 3A 3A 3A 3A 49 4D 3A 3A 3A 3A 3A 49 49,AE  ::..::::IM::::II
0C30 3A 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20,9F  :..
0C40 20 20 20 2E 2E 20 2E 2E 2E 3A 3A 3A 49 49 48 49,D6    .. ...:::IIHI
0C50 49 49 3A 3A 3A 22 2F 2F 2F 2F 2F 49 3A 3A 3A 3A,54  II:::"/////I:::::
0C60 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 2E 3A,E8  ::::::::::::::.:
0C70 3A 3A 3A 49 4D 3A 3A 3A 3A 3A 49 3A 0D 0A 20 20,28  :::IM:::::I:..
```



```
0C80 20 20 20 20 20 20 20 20 20 20 20 20 20 20 2E 2E,44                ..
0C90 2E 2E 3A 3A 3A 3A 3A 49 49 49 49 49 3A 3A 3A 3A,17  ..::::::IIIII::::
0CA0 2E 2E 3A 3A 3A 2F 2F 3A 3A 3A 3A 3A 3A 3A 3A 3A,89  ..:::/::::::::::
0CB0 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 2E 2E 3A 3A 3A 49 49,2F  ::::::::::..:::II
0CC0 3A 3A 3A 3A 3A 3A 3A 0D 0A 20 20 20 20 20 20 20,BC  :::::::..
0CD0 20 20 20 20 20 20 20 20 20 20 2E 2E 2E 2E 2E 3A 3A,36          .....::
0CE0 3A 3A 49 49 49 3A 3A 3A 3A 3A 2E 2E 3A 3A 3A,FA  ::IIII:::::..:::
0CF0 3A 3A 3A 3A 3A 3A 3A 2E 2E 3A 3A 3A 3A 3A 2E 3A 3A,76  :::::::..::::::.::

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0D00 3A 3A 3A 3A 3A 3A 2E 2E 3A 3A 49 49 3A 3A 3A 3A 3A,1C  ::::::..::II:::::
0D10 3A 3A 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20,27  ::..
0D20 20 20 20 20 2E 2E 2E 2E 3A 3A 3A 3A 49 49 49 49,6B      ....::::IIII
0D30 4D 22 3A 3A 3A 3A 2E 2E 3A 3A 3A 3A 3A 3A 3A 3A,EE  M":::::..::::::::
0D40 2E 2E 2E 2E 3A 3A 3A 2E 2E 2E 2E 3A 3A 3A 3A 3A,2E  ....:::....:::::
0D50 2E 2E 3A 3A 49 49 3A 2E 2E 3A 3A 3A 3A 0D 0A 20,45  ..::II:...:::::..
0D60 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 3A,5F                :
0D70 2E 2E 2E 3A 3A 3A 3A 3A 49 49 49 4D 3A 3A 3A 3A,1B  ...:::::IIIM::::
0D80 22 2E 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 2E 2E 2E,73  ".::::::::::::...
0D90 2E 2E 2E 2E 2E 3A 3A 3A 3A 3A 3A 3A 2E 2E 3A 3A 48,CD  .....:::::::..::H
0DA0 49 3A 2E 2E 3A 3A 3A 3A 0D 0A 20 20 20 20 20 20,6B  I:..::::..
0DB0 20 20 20 20 20 20 20 20 20 20 3A 3A 2E 2E 3A 3A,EF            ::..::
0DC0 3A 3A 3A 3A 3A 3A 49 4D 3A 3A 3A 22 2E 2E 3A 3A 3A,81  ::::::IM:::"..:::
0DD0 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 2E 2E 2E 2E 2E 2E 2E,CD  ::::::::::.......
0DE0 3A 3A 3A 3A 3A 3A 2E 20 2E 3A 48 4D 49 3A 2E 2E,53  ::::::. .:HMI:..
0DF0 3A 3A 3A 0D 0A 20 20 20 20 20 20 20 20 20 20 20,78  :::..

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0E00 20 20 20 20 20 3A 3A 2E 3A 3A 3A 3A 3A 3A 3A 3A,8A        ::.:::::::::
0E10 49 4D 49 3A 3A 20 20 2E 2E 3A 3A 3A 3A 3A 3A 3A,0F  IMI::  ..::::::::
0E20 3A 3A 3A 3A 3A 3A 2E 2E 2E 3A 3A 3A 3A 3A 3A 3A,8B  ::::::...:::::::::
0E30 2E 2E 20 2E 2E 56 4D 49 3A 2E 2E 3A 3A 3A 0D 0A,AA  .. ..VMI:..:::...
0E40 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,AA
0E50 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 49 49 3A 20,4E  ::::::::::::II:
0E60 20 20 20 2E 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 49 49 3A,B2     .::::::::::II:
0E70 3A 2E 2E 2E 3A 3A 3A 3A 3A 3A 2E 2E 20 20 2E 2E,CA  :...::::::..  ..
0E80 2E 4D 49 3A 2E 2E 3A 3A 3A 0D 0A 20 20 20 20 20,89  .MI:..:::..
0E90 20 20 20 20 20 20 20 20 20 20 20 3A 3A 3A 3A 3A,0B           :::::
0EA0 3A 3A 3A 3A 3A 3A 3A 49 48 2E 2E 20 20 20 20 27 3A,35  ::::::IH..    ':
0EB0 3A 3A 3A 3A 3A 3A 3A 3A 49 49 49 49 3A 3A 2E 2E 2E,ED  :::::::IIII::...
0EC0 3A 3A 3A 3A 27 20 20 20 2E 20 2E 2E 4D 49 3A 2E,04  ::::'   . ..MI:.
0ED0 2E 2E 3A 3A 0D 0A 20 20 20 20 20 20 20 20 20 20,2B  ..::..
0EE0 20 20 20 20 20 20 3A 2E 2E 2E 3A 3A 3A 3A 3A 3A,0B        :...::::::
0EF0 3A 3A 4D 3A 27 20 20 20 20 20 27 3A 3A 3A 3A 3A,16  ::M:'     ':::::

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0F00 3A 49 49 49 49 49 3A 3A 2E 2E 3A 3A 3A 3A 3A 27,D6  :IIIII::..::::::'
0F10 20 20 2E 20 2E 20 2E 56 49 3A 3A 2E 2E 3A 3A 0D,D0    . . .VI::..::.
0F20 0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20 2E,C8  .                .
0F30 49 2E 2E 2E 3A 3A 3A 3A 3A 3A 3A 3A 49 3A 20 20,2E  I...::::::::I:
0F40 20 20 20 20 2E 20 3A 3A 3A 3A 49 49 49 49 49 49,9A      . ::::IIIIII
0F50 49 3A 3A 3A 3A 3A 49 3A 27 2E 20 20 2E 20 2E 2E,C7  I:::::I:'.  . ..
0F60 3A 56 20 49 3A 2E 2E 3A 3A 3A 0D 0A 20 20 20 20,9B  :V I:..:::..
0F70 20 20 20 20 20 20 20 2E 3A 3A 3A 3A 49 2E 2E 3A,70         .::::I..:
0F80 3A 3A 3A 3A 3A 3A 3A 49 3A 20 20 20 20 20 20 20,69  :::::::I:
0F90 20 2E 3A 3A 49 49 49 49 48 48 48 48 48 49 49 3A,83    .::IIIIHHHHHII:
0FA0 3A 3A 20 20 20 20 41 4D 49 20 2E 56 20 20 49 3A,B5  ::     AMI .V  I:
0FB0 2E 2E 3A 3A 3A 0D 0A 20 20 20 20 20 20 20 20 20,F6  ..:::..
0FC0 2E 3A 3A 3A 2E 2E 3A 49 3A 2E 2E 3A 3A 3A 3A 3A,69  .:::..:I:..:::::
0FD0 3A 3A 49 49 20 41 49 2E 20 20 20 2E 20 2E 3A 3A,97  ::II AI.   . .::
0FE0 49 48 48 48 49 3A 3A 49 49 3A 3A 27 20 20 20 20,08  IHHHI:II::'
0FF0 20 22 57 56 2E 56 27 20 20 49 3A 3A 3A 3A 3A 3A,87   "WV.V'  I::::::
```

```
     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
1000 0D 0A 20 20 20 20 20 20 2E 3A 3A 2E 2E 2E 2E 2E,E6  ..       .::......
1010 2E 2E 49 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 49 48 4D 2E,A1  ..I:::::::::IHM.
1020 56 4D 41 2E 2E 2E 3A 41 48 49 49 48 48 48 49 49,CE  VMA...:AHIIHHHII
1030 3A 3A 2E 20 2E 2E 20 20 2E 2E 3A 3A 2E 2E 3A 56,E8  ::. ..  ..::..:V
1040 20 20 20 20 3A 3A 3A 3A 2E 2E 3A 0D 0A 20 20 20,5D      ::::..:..
1050 20 20 2E 3A 3A 3A 2E 2E 2E 2E 2E 2E 2E 2E 49 3A,6C    .:::.........I:
1060 3A 3A 3A 3A 3A 3A 3A 49 49 49 4D 4D 4D 4D 4D 4D,AB  :::::::IIIMMMMMMM
1070 48 48 48 48 48 2E 2E 2E 2E 3A 3A 49 49 49 3A 3A,BE  HHHHH....::III::
1080 3A 3A 3A 3A 3A 49 49 49 49 56 20 20 20 20 20 3A,04  :::::IIIIV     :
1090 3A 3A 2E 3A 3A 3A 0D 0A 20 20 20 20 2E 3A 3A 3A,C7  ::.:::..    .:::
10A0 3A 3A 3A 3A 3A 3A 3A 2E 2E 49 3A 3A 3A 3A 3A 3A,5E  :::::::..I::::::
10B0 3A 3A 3A 49 49 49 4D 4D 4D 4D 4D 48 48 48 48 48,D0  :::IIIMMMMMHHHHH
10C0 48 48 48 48 49 49 49 49 48 48 48 4D 4D 4D 48 4D,6D  HHHHIIIIHHHMMMMHM
10D0 4D 48 48 56 20 20 20 20 20 20 3A 3A 3A 3A 2E 3A,AB  MHHV      ::::.:
10E0 3A 0D 0A 20 20 20 2E 20 3A 3A 3A 3A 3A 3A 3A 3A,7A  :..    . ::::::::
10F0 3A 3A 3A 3A 3A 49 3A 3A 3A 3A 3A 3A 3A 3A 3A 49,38  :::::I:::::::::I

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
1100 49 41 4D 4D 4D 4D 48 48 48 48 48 48 48 48 49 49,C8  IAMMMMHHHHHHHHII
1110 49 49 49 49 49 48 48 48 48 48 48 48 56 20 20 0B,0B  IIIIIHHHHHHHHV
1120 20 20 20 20 20 2E 2E 3A 2E 2E 2E 3A 0D 0A 20 20,5C    ..:...:..
1130 2E 3A 20 2E 2E 2E 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A,B2  .: ...:::::::::
1140 49 3A 3A 3A 3A 3A 3A 3A 3A 3A 49 49 3A 3A 56 4D,AE  I:::::::::II::VM
1150 4D 4D 4D 48 48 48 48 48 48 48 49 49 49 49 48,42  MMMHHHHHHHHIIIIH
1160 48 48 48 48 48 48 48 48 22 20 20 20 20 20 20 20,84  HHHHHHHH"
1170 20 3A 3A 2E 2E 2E 3A 0D 0A 20 20 3A 3A 2E 20 2E,23   :::...:..  ::. .
1180 3A 3A 3B 3B 3B 3B 3B 3A 3A 3A 3A 49 3A 3A 2E 3A,CB  :::;;;;;;;::I:::.:
1190 3A 3A 3A 3A 49 49 49 3A 3A 49 56 4D 4D 4D 4D 4D,22  ::::III::IVMMMMMM
11A0 4D 4D 48 48 48 48 49 49 49 49 49 49 49 48 48 48,B3  MMHHHHHIIIIIIIHHH
11B0 48 56 20 20 20 20 20 20 20 20 20 20 20 2E 2E 3A,47  HV             ..:
11C0 2E 2E 0D 0A 20 2E 3A 3A 3A 2E 20 27 3A 3A 3A 3B,14  .... .:::. ':::;
11D0 3B 3B 3A 3A 3A 3A 49 3A 3A 2E 2E 3A 3A 3A 3A 3A,AD  ;;:::I::..:::::
11E0 49 49 3A 3A 3A 3A 3A 3A 3A 56 4D 4D 4D 4D 48 48,EF  II:::::::VMMMMHH
11F0 48 48 48 48 49 49 49 48 4D 4D 4D 22 20 20 20 20,BB  HHHHIIIHMMM"

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
1200 20 20 20 20 20 20 20 20 20 2E 2E 20 2E 0D 0A 20,BC           .. ...
1210 3A 3A 3A 3A 3A 2E 20 20 20 20 27 3A 3A 3B 3B 3A,D7  :::::.    ':;;;;:
1220 3A 3A 49 2E 2E 2E 3A 3A 3A 3A 49 49 49 49 49 49,BC  ::I...:::::IIIIII
1230 49 49 3A 3A 3A 3A 56 48 4D 4D 4D 4D 48 48 48 48,28  II:::::VHMMMMHHHH
1240 49 48 4D 4D 22 20 20 20 20 20 20 20 20 20 20 20,D5  IHMM"
1250 20 20 20 20 27 2E 2E 0D 0A 20 3A 3A 3A 3A 3A 3A,6B      '.... ::::::
1260 2E 20 20 20 20 20 20 27 3A 3B 3B 3B 3A 3A 49 3A 2E,70  .      ':;;;;::I:.
1270 2E 3A 3A 3A 3A 3A 49 49 49 49 4D 3A 3A 3A 3A 3A,53  .:::::IIIIM:::::
1280 3A 49 49 4D 4D 4D 4D 4D 4D 4D 4D 4D 4D 22 20 20,83  :IIMMMMMMMMMM"
1290 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 2E,91                .
12A0 2E 2E 0D 0A 20 3A 3A 3A 3A 2E 2E 3A 3A 3A 2E 20,64  .... :::::..:::.
12B0 20 20 20 20 3A 3A 3A 3A 49 2E 20 3A 3A 3A 3A 3A,85      ::::I. :::::
12C0 49 49 49 27 3A 3A 3A 3A 3A 49 49 48 48 4D 4D 4D,B2  III':::::IIHHMMM
12D0 4D 4D 4D 4D 4D 4D 22 2E 20 20 20 20 20 20 20 20,D0  MMMMMM".
12E0 20 20 20 20 20 20 20 20 20 20 27 3A 2E 0D 0A 20,D6            ':...
12F0 3A 3A 3A 2E 2E 2E 3A 3A 3A 3A 49 3A 3A 20 20 20,13  :::...::::I::

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
1300 22 3A 3A 49 3A 20 2E 3A 3A 3A 49 49 49 2E 3A,A5  "::I: .:::::III.:
1310 3A 3A 3A 3A 49 48 48 4D 4D 4D 4D 4D 4D 4D 4D,1B  ::::IHHMMMMMMMMM
1320 56 3A 2E 20 20 20 20 20 20 20 20 20 20 20 20,79  V:.
1330 20 20 20 20 20 20 3A 3A 0D 0A 20 3A 3A 3A 2E 2E,EE        ::.. :::..
1340 2E 2E 3A 3A 3A 49 49 49 3A 20 20 20 22 3A 49 3A,4C  ..:::III:   ":I:
1350 2E 20 3A 3A 3A 3A 49 49 49 3A 3A 3A 3A 49 49 49,20  . ::::III::::III
1360 48 48 48 4D 4D 4D 4D 4D 4D 56 3A 3A 3A 20 20,57  HHHMMMMMMMV:::
1370 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,57
```



```
1380 20 3A 3A 0D 0A 20 3A 3A 3A 3A 2E 2E 2E 3A 3A 3A,42   ::.. ::::...:::
1390 3B 49 49 49 48 20 20 20 20 3A 49 3A 20 27 3A 3A,98  ;IIIH    :I: ':::
13A0 3A 49 49 49 49 3A 49 49 48 48 48 4D 4D 4D 4D 4D,1B  :IIII:IIHHHMMMMM
13B0 4D 48 48 48 49 49 3A 3A 3A 20 20 20 20 20 20 20,60  MHHHII:::
13C0 20 20 20 20 20 20 20 20 20 20 20 20 20 27 3A 0D,58              ':..
13D0 20 3A 3A 3A 3A 2E 2E 2E 3A 3A 3A 3B 3B 49 49,DA    :::::...::::;;II
13E0 4D 2E 3A 2E 20 49 3A 3A 2E 2E 3A 3A 3A 3A 49 49,70  M.:. I::..::::II
13F0 48 48 48 48 4D 4D 4D 4D 4D 4D 4D 48 48 48 49 49,15  HHHHMMMMMMMHHHII

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
1400 3A 3A 3A 3A 20 20 20 20 20 20 20 20 20 20 20 20,7D  ::::
1410 20 20 20 20 20 20 20 20 27 0D 0A 20 3A 3A 3A 3A,C3          '.. ::::
1420 2E 2E 2E 3A 3A 3A 3A 3B 3B 49 4D 4D 57 48 4D,B4  ...:::::;;IMMWHM
1430 49 3A 3A 2E 2E 3A 3A 3A 3A 49 49 49 48 48 48 4D,B5  I::..::::IIIHHHM
1440 4D 4D 4D 4D 48 48 48 48 49 49 3A 3A 3A 3A 3A 20,DD  MMMMHHHHII:::::
1450 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,DD
1460 20 20 20 20 2E 0D 0A 20 3A 3A 3A 3A 3A 2E 2E 2E 3A,88     ... :::::...:
1470 3A 3A 3A 3B 49 49 48 4D 4D 4D 4D 4D 3A 3A 2E 2E,9C  :::;IIHMMMMM::..
1480 2E 3A 3A 3A 3A 49 49 48 48 48 48 48 48 48 48 48,CC  .:::::IIHHHHHHHHH
1490 48 48 48 49 49 3A 3A 3A 3A 27 0D 0A 20 3A 3A 3A,2A  HHHII:::::'.. :::
14A0 3A 3A 2E 2E 3B 3A 3A 3A 3A 3B 3B 49 49 48 4D 4D,07  ::..;:::::;;IIHMM
14B0 4D 4D 4D 3A 3A 2E 2E 2E 3A 3A 3A 3A 59 48 48,F7  MMM::...::::::YHH
14C0 48 48 48 4D 4D 4D 48 48 49 49 49 3A 3A 3A 2E 0D,0A  HHHMMMHHIII:::..
14D0 0A 20 3A 3A 3A 3A 3A 2E 2E 2E 3A 3A 3A 3A 3B 3B,3E  . :::::...::::;;
14E0 49 49 48 4D 4D 4D 4D 49 3A 2E 2E 2E 3A 3A 3A,54  IIHMMMMMI:...:::
14F0 3A 49 20 3A 22 22 22 22 3A 3A 3A 49 49 49 49 3A,C5  :I :"""":::IIII:

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
1500 3A 3A 3A 0D 0A 20 3A 3A 3A 3A 3A 3A 2E 2E 3A 3A,D6  :::.. ::::::..::
1510 3A 3A 3B 3B 3B 49 49 48 4D 4D 4D 4D 4D 3A 3A 2E,F8  :::;;;IIHMMMMM::.
1520 2E 3A 3A 3A 3A 3A 49 3A 3A 3A 3A 2E 2E 2E 20 20,43  .::::::I::::...
1530 20 2E 20 2E 3A 3A 27 0D 0A 20 27 3A 3A 3A 3A 3A,FA   . .::'.. ':::::
1540 2E 2E 3A 3A 3A 3A 3B 3B 3B 49 49 49 48 4D 4D 4D,F9  ..::::::;;;IIIHMMM
1550 56 49 3A 2E 2E 3A 2E 3A 3A 3A 49 3A 3A 3A 3A 3A,AF  VI:..:.:::I:::::
1560 3A 3B 3B 3B 3B 3A 3A 3A 3A 0D 0A 20 20 27 3A 3A,AF  :;;;;;::::..  ':::
1570 3A 3A 3A 3A 3A 3A 3A 3B 3B 3B 49 49 49 49 4D,A1  ::::::::::;;;IIIIM
1580 4D 49 3A 3A 49 3A 3A 2E 2E 3A 3A 3A 49 3A 3A,69  MI::I::..::::I::
1590 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 0D 0A 20 20 20,5E  ::::::::::::..
15A0 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3B 3B 3B 3B 3B 49,12  ::::::::::::::::I
15B0 49 4D 48 3B 3A 3A 3A 49 3A 3A 2E 2E 3A 3A 3A 49,E9  IMH;:::I::..:::I
15C0 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 3A 0D 0A 20 20,F8  ::::::::::::..
15D0 20 27 3A 3A 3A 3A 3A 3A 3A 3A 3A 3B 3B 3B 3B 3B,70   ':::::::::::;;;;;
15E0 3B 49 4D 3B 3A 3A 3A 3A 49 3A 2E 3A 2E 2E 3A 3A,1F  ;IM;::::I:.:..::
15F0 31 49 2E 2E 3A 3A 3A 3A 3A 3A 3A 3A 27 0D 0A 20,23  1I..:::::::::'..

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
1600 20 20 20 3A 3A 3A 3A 3A 3A 3A 3A 3A 3B 3B 3B 3B,79     ::::::::::;;;;
1610 3B 3B 49 4D 3A 3A 3A 3A 3A 3A 49 3A 3A 2E 2E 2E,28  ;;IM::::::I:::...
1620 3A 3A 49 3A 3A 3A 3A 3A 3A 3A 3A 3B 27 0D 0A 20,4E  ::I::::::::;'..
1630 20 20 20 27 3A 3A 3A 3A 3A 3A 3A 3A 3A 3B 3B 3B,90     ':::::::::::;;;
1640 3B 49 4D 49 3A 3A 3A 3A 3A 3A 3A 49 2E 3A 2E 2E,4D  ;IMI:::::::I.:...
1650 3A 3A 2E 49 2E 3A 3A 3A 3A 3A 3A 3B 3B 0D 0A 20,55  ::.I.:::::::;..
1660 20 20 20 3A 3A 3A 3A 3A 3A 3A 3A 3B 3B 3B 3B 3B,AC     ::::::::::::;;;;
1670 49 4D 3B 3A 3A 2E 3A 3A 3A 3A 3A 49 49 3A 2E 3A,75  IM;::.:::::II:.:
1680 3A 2E 2E 3A 3A 3A 3A 3A 3A 3B 27 0D 0A 20 20 20,40  :..:::::::;'..
1690 20 20 3A 3A 3A 3A 3A 3A 3A 3A 3A 3B 3B 3B 49 49,CD    ::::::::::::;;II
16A0 4D 3A 3A 2E 2E 3A 3A 3A 3A 3A 49 49 3A 2E 2E 3A,6E  M:::..::::::II:..:
16B0 3A 2E 49 3A 3A 3A 3B 3B 0D 0A 20 20 20 20 20 27,21  :.I:::::;;..      '
16C0 3A 3A 3A 3A 3A 3A 3A 3A 3A 3B 3B 49 49 4D 3A 3A,F4  ::::::::::;;IIM::
16D0 2E 2E 2E 3A 3A 3A 3A 3A 3A 49 49 3A 2E 2E 3A 2E 3A,6A  ...:::::::II:..:.:
16E0 3B 3B 3B 3B 27 0D 0A 20 20 20 20 20 20 3A 3A 3A,02  ;;;;'..       :::
16F0 3A 3A 3A 3A 3A 3A 3A 3B 3B 49 48 3A 3A 2E 2E 2E 3A,9D  ::::::::;;IH::...:
```

```
     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
1700 3A 3A 3A 3A 3A 49 49 49 3A 2E 2E 3A 2E 49 3B 3B,57  :::::III:..:.I;;
1710 56 0D 0A 20 20 20 20 20 20 27 3A 3A 3A 3A 3A 3A,07  V..      ':::::::
1720 3A 3A 3B 3B 49 48 3A 3A 2E 2E 2E 3A 3A 3A 3A 3A,A2  ::;;IH::...:::::
1730 3A 3B 49 49 49 2E 3A 3A 3A 2E 49 56 0D 0A 20 20,F2  :;III.:::.IV..
1740 20 20 20 20 20 3A 3A 3A 3A 3A 3A 3A 3A 3A 3B 49,20        ::::::::::;I
1750 56 3A 3A 2E 2E 2E 3A 3A 3A 3A 3A 3A 3A 49 49,D6  V:::...:::::::::II
1760 49 2E 3A 3A 3A 49 2E 0D 0A 20 20 20 20 20 20 20,69  I.:::I...
1770 27 3A 3A 3A 3A 3A 3A 3A 3B 3B 48 3A 3A 2E 2E 2E,E2  ':::::::::;;H:::...
1780 3A 3A 3A 3A 3A 3A 3A 3B 3B 4A 49 3A 2E 2E 27 3A,78  ::::::::;;JI:..':
1790 3A 41 0D 0A 20 20 20 20 20 20 20 3B 3A 3A 3A,F3  :A..         ;:::
17A0 3A 3A 3A 3A 3B 3B 49 3A 3A 2E 2E 2E 2E 3A 3A 3A,74  ::::;;I:::....::::
17B0 3A 3A 3A 3B 3B 49 49 2E 2E 2E 3A 3A 2E 49 20 20,DF  :::;;II...:::.I
17C0 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,DF
17D0 20 20 20 20 20 20 20 20 20 20 20 2F 0D 0A 20,C5                /..
17E0 20 20 20 20 20 20 20 20 3A 3A 3A 3A 3A 3A 3A 3B 48,BE          ::::::::;H
17F0 3A 3A 2E 2E 2E 2E 3A 3A 3A 3A 3A 3A 3A 3A 3B 3B,30  ::....::::::::;;

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
1800 49 3A 2E 2E 3A 27 3A 3A 3A 2E 20 20 20 20 20 20,0C  I:..:':::.
1810 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,0C
1820 20 20 20 20 20 20 20 3A 0D 0A 20 20 20 20 20 20,FD         :..
1830 20 20 20 3A 3A 3A 3A 3A 3B 3B 49 3A 3A 2E 2E 2E,3C     :::::;;I:::...
1840 3A 3A 3A 3A 3A 3A 3A 3A 3A 3B 49 49 3A 2E 2E 27,D0  ::::::::::;II:..'
1850 3A 3A 3A 3A 49 20 20 20 20 2E 3A 3A 3A 2E 2E 2E,E7  ::::I    .:::...
1860 2E 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20,F5  .
1870 20 2E 3A 0D 0A 20 20 20 20 20 20 20 20 20 3A 3A,28   .:..          ::
1880 3A 3A 3A 3A 49 3A 3A 2E 2E 2E 2E 3A 3A 3A 3A 3A,A7  ::::I::....:::::
1890 3A 3A 3A 3A 3B 49 3A 2E 2E 2E 2E 3A 3A 49 3A 49,45  :::::I:....::I:I
18A0 2E 3A 49 3A 3A 3A 49 2E 3A 3A 3A 3A 3A 3A 3A,EB  .:I:::I.:::::::
18B0 3A 3A 2E 20 20 2E 2E 20 20 20 2E 3A 49 49 0D 0A,9A  ::.  ..   .:II..
18C0 20 20 20 20 20 20 20 20 20 27 3A 3A 3A 3A 3A 49,4C          ':::::I
18D0 3A 3A 2E 2E 3A 3A 3A 3A 3A 3A 3A 3A 3B 3B 3B 49,E6  ::..:::::::::;;;I
18E0 3A 27 2E 3A 3A 49 49 48 49 3A 49 2E 3A 49 49 49,D2  :'.::IIHI:I.:III
18F0 49 49 49 49 49 49 49 49 49 49 49 49 22 22 2F 4D,FE  IIIIIIIIIIII""/M

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
1900 4D 4D 2D 2E 3A 4A 49 48 4D 0D 0A 20 20 20 20 20,0C  MM-.:JIHM..
1910 20 20 20 20 27 3A 3A 3A 3A 49 3A 3A 2E 2E 2E 3A,1C      ':::::I:::...:
1920 3A 3A 3A 3A 3A 3A 3A 3A 3B 3B 49 22 3A 3A 3A 49,C4  ::::::::;;I":::I
1930 49 56 56 49 3A 49 2E 48 48 49 3A 3A 3A 3A 3A 3A,E8  IVVI:I.HHI::::::
1940 49 49 2F 4D 4D 4D 4D 4D 4D 4D 4D 4D 2E 2E 50,57  II/MMMMMMMMMM..P
1950 50 50 50 3A 0D 0A 20 20 20 20 20 20 20 20 20 20,D8  PPP:..
1960 3A 3A 3A 3A 49 3A 3A 2E 2E 2E 3A 3A 3A 3A 3A 3A,63  ::::I::...::::::
1970 3A 3A 3B 3B 3B 49 59 3A 3A 49 56 4D 4D 41 3A 3A,8C  :::;;;IY:::IVMMA::
1980 3A 49 48 3A 3A 49 41 3A 3A 3A 41 48 48 4D 4D 4D,BB  :IH::IA:::AHHMMM
1990 56 58 58 58 58 58 56 58 58 49 49 3A 3A 3A 3A 0D,56  VXXXXXVXXII:::::.
19A0 0A 20 20 20 20 20 20 20 20 20 20 20 3A 3A 3A 3A 49,D1  .           ::::I
19B0 3A 27 2E 2E 2E 3A 3A 3A 3A 3A 3A 3A 3A 3A 3B 3B,3C  :'...:::::::::;;
19C0 49 3A 3A 56 4D 48 48 49 49 49 3A 22 49 48 3A 3F,6D  I::VMHHIII:"IH:?
19D0 48 48 41 3A 48 2F 2F 2E 20 58 58 58 58 58 58 58,D4  HHA:H//. XXXXXXX
19E0 2F 22 22 3A 49 3A 3A 3A 49 49 0D 0A 20 20 20 20,A1  /"":I:::II..
19F0 20 20 20 20 20 20 3A 3A 3A 49 3A 3A 2E 2E 2E 3A,90        :::I::...:

     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
1A00 3A 3A 3A 3A 3A 3A 3A 3A 3B 3B 22 2E 3A 2F 41 49,19  :::::::::;;".:/AI
1A10 3A 3A 48 48 48 41 2E 3A 48 3F 48 48 48 48 2F,2C  ::HHHA.:H?HHHHH/
1A20 24 58 58 58 22 24 24 58 58 2E 58 2F 2F 2E 41 49 3A,F0  $XX"$$XX.X//.AI:
1A30 3A 3A 3A 3A 3A 0D 0A 20 20 20 20 20 20 20 20 20,49  :::::..
1A40 20 3A 3A 3A 49 3A 3A 2E 2E 2E 3A 3A 3A 3A 3A 3A,BA   :::I::...::::::
1A50 3A 3A 3A 3B 49 3A 2F 41 49 49 3A 48 22 2E 22 2E,4A  :::;I:/AII:H".".
1A60 22 2E 22 3F 48 48 48 48 48 2F 2F 2F 2F 2F 24 27,99  "."?HHHHH/////$'
1A70 2F 2F 2F 2F 22 2F 2F 2F 41 3A 3A 3A 3A 3A 3A 3A,DB  ////"///A:::::::
```

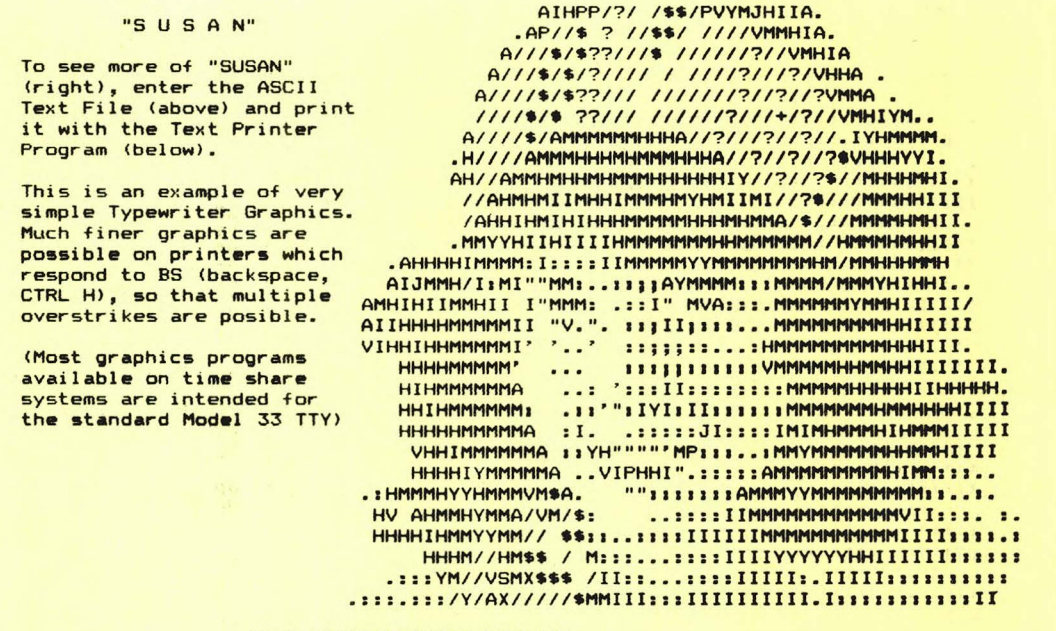
REPLACE THE LAST THREE BYTES WITH $0D (CR), $0A (LF), $04 (EOT)

"S U S A N"

To see more of "SUSAN" (right), enter the ASCII Text File (above) and print it with the Text Printer Program (below).

This is an example of very simple Typewriter Graphics. Much finer graphics are possible on printers which respond to BS (backspace, CTRL H), so that multiple overstrikes are posible.

(Most graphics programs available on time share systems are intended for the standard Model 33 TTY)



```
0010 ;TEXT PRINTER PROGRAM
0020 ;ADAPTED BY LUX FROM
0030 ;LAVIN'S MODEM PROGRAM
0040
0050            .BA $0100
0060            .MC $9000
0070            .OS
0080
0090 EOT        .DE $04
0100
0110 POINTER    .DE $00FE
0120 TEXTSTART  .DE $0200     ;OR WHEREVER
0130
0140 INCCMP     .DE $82B2
0150 INSTAT     .DE $8386
0160 OUTCHR     .DE $8A47
0170
0100- A9 02     0180 INIT       LDA #H,TEXTSTART
0102- A0 00     0190            LDY #L,TEXTSTART
0104- 85 FF     0200            STA *POINTER+1
0106- 84 FE     0210            STY *POINTER
                0220
0108- A0 00     0230 PRINTIT    LDY #$00
010A- B1 FE     0240            LDA (*FE),Y
010C- C9 04     0250            CMP #EOT      ;END OF TEXT MARKER (CTRL D)
010E- F0 0B     0260            BEQ EXIT
                0270
0110- 20 47 8A  0280            JSR OUTCHR
0113- 20 B2 82  0290            JSR INCCMP
0116- 20 86 83  0300            JSR INSTAT    ;CARRY SET IF BRK KEY HELD DOWN
0119- 90 ED     0310            BCC PRINTIT
                0320
011B- 60        0330 EXIT       RTS :         ;BACK TO CALLER
                0340
                0350            .EN
```

```
                3020
                3030      (SWISS) CLOCK (continued from page 13/14-14)
                3040 ;Here follows the interrupt driven clock routine. The
                3050 ;us/s regulation is accomplished with the time spent
                3060 ;in the interrupt service before the start of
                3070 ;timer1: Interrupts have to occur every 50000 us, with
                3080 ;20 'jiffies' to the second. If the timerload is
                3090 ;changed by +/- 1 this  gives a change of +/- 20 us/s,
                3100 ;much larger than the precision of the clock. We get
                3110 ;a resolution of +/- 1 us by making a number A of 20
                3120 ;jiffies each 1 us longer than the remaining 20-A.
                3130 ;The size of 0<A<=20 is in MIKSEC and can be changed
                3140 ;by pressing 02 or 03 on the HKB. It over- or under-
                3150 ;flows into LNIB of the timer1-load so that a
                3160 ;continuous regulation is possible. This allows to
                3170 ;correct deviations of the quartz from 1.000000 MHz,
                3180 ;which is essential if the clock is supposed to run
                3190 ;for many days or weeks.
                3200
                3210 ;           INTERRUPT SERVICE
                3220
03B7- 08        3230 CLOCK       PHP  ;save    [from here to start of timer1
03B8- 48        3240             PHA  ; status [30 or 31 usec depending on
03B9- F8        3250             SED  ;        [byte in MIKSEC ]
03BA- A5 F3     3260             LDA *LNIB    ;+/- 1 if over/underflo of MIKSEC
03BC- 8D 06 A0  3270             STA T1LL
03BF- A5 F7     3280             LDA *COUNT
03C1- C5 F2     3290             CMP *MIKSEC  ;adjusts usec in loop !
03C3- B0 01     3300             BCS CONT     ;branch if count >= MIKSEC
03C5- EA        3310             NOP  ; otherwise not, which is 1 usec longer !
03C6- A9 C3     3320 CONT        LDA #$C3     ;hi nib of timer1
03C8- 8D 05 A0  3330             STA T1CH     ;start timer1
03CB- C6 F7     3340             DEC *COUNT   ;  for 49970+30[+1] usecs
03CD- D0 31     3350             BNE EXITC
03CF- A9 14     3360             LDA #$14     ;restore jiffy-counter
03D1- 85 F7     3370             STA *COUNT
03D3- A9 01     3380             LDA #01      ;now, one of the usual routines
03D5- 18        3390             CLC
03D6- 65 F6     3400             ADC *SEC     ; for updating the display-
03D8- 85 F6     3410             STA *SEC     ; registers
03DA- C9 60     3420             CMP #$60
03DC- D0 22     3430             BNE EXITC
03DE- A9 00     3440             LDA #00
03E0- 85 F6     3450             STA *SEC
03E2- A9 01     3460             LDA #01
03E4- 18        3470             CLC
03E5- 65 F5     3480             ADC *MIN
03E7- 85 F5     3490             STA *MIN
03E9- C9 60     3500             CMP #$60
03EB- D0 13     3510             BNE EXITC
03ED- A9 00     3520             LDA #00
03EF- 85 F5     3530             STA *MIN
03F1- A9 01     3540             LDA #01
03F3- 18        3550             CLC
03F4- 65 F4     3560             ADC *HOUR
03F6- 85 F4     3570             STA *HOUR
03F8- C9 24     3580             CMP #$24
03FA- D0 04     3590             BNE EXITC
03FC- A9 00     3600             LDA #00
03FE- 85 F4     3610             STA *HOUR
0400- D8        3620 EXITC       CLD  ;make sure its hex again
0401- 68        3630             PLA  ;restore status
0402- 28        3640             PLP
0403- 40        3650             RTI
                3655
                3660             .EN
```


MORE ON FORTH (continued from page 13/14-4)

Incidentally, FORTH is a stack oriented system, making use of both the host system stack (for its "return" stack) and its own stack, which is implemented in page 00 for the 6502, using the X register as its stack pointer. Since stacks are, by their very nature, LIFO (last-in, first-out), Reverse Polish format for both the arithmetic and the language syntax is the inherent way to go. After a little practice, the Reverse Polish Notation (RPN) becomes almost a natural way of logically ordering ideas. There is a close similarity between RPN and the German grammatical structure for complex sentences, in which the verbs from each clause are all "stacked" together at the end of the sentence, LIFO!

Elsewhere in this issue (pages 13/14-5 through 13/14-9) appears a "skeletonized" version of the SYM-FORTH source code, adapted from our disassembly of the FORTH described above, so that you can get some idea of its structure and method of implementation.

Note that only a relatively simple data management (essentially a stack handler) subsystem and the first few FORTH words are written in ML. The remaining words are written in FORTH itself, with only very infrequent references to ML. Only the ML portions need be rewritten for the 6809, for example, and hand assembled into the source code, in .BY $XX $YY $ZZ form, to have a TOTALLY COMPATIBLE FORTH for the MOD-69 SYM. This should not be too difficult, since less than 1K of object code is involved. We will then copy one of the published 6809 FORTH ASSEMBLER vocabularies and have a truly powerful 6809 System.

Note, also, that it will be easier to write other high level languages, e. g., Pascal, in FORTH than in ML, and that the FORTH version will be machine independent. Thus a high level language need be written and debugged only once, for all machines. We're not sure that we'd really want the Pascal if we had the FORTH, but we can see emulating at least the I/O syntax of other languages in FORTH.

```
                0010             ;   **** MODIFIED SUPERMON ****
                0020
                0030             ;         BY PAUL L. BEAUPRE
                0040
                0050             ; THIS PROGRAM GIVES THE SYM-1 ONE
                0060             ; THING IT LACKS, A DTR INPUT.
                0070             ; BY MODIFYING THE 'TTY IN' PORT TO
                0080             ; AN RS-232 LOOP, YOU CAN HOOK UP YOUR
                0090             ; KTM-2'S DTR LINE, PRINTER READY,
                0100             ; OR ANYTHING ELSE THAT NEEDS TO
                0110             ; HAVE THE SYM WAIT.
                0120
                0130             ; I BURNED THIS PROGRAM IN A
                0140             ; 2732, COPYING SUPERMON WITH A
                0150             ; BLOCK MOVE AND MODIFYING LOCATION
                0160             ; $8BA7 WHICH WAS FORMERLY THE TTY
                0170             ; LOG ON LOCATION. IF YOU DON'T USE
                0180             ; A TTY THEN USE THIS LOCATION.YOU
                0190             ; MUST ALSO CHANGE LOCATION $8C73 AND
                0200             ; $8C74 TO POINT OUTVEC TO THE NEW
                0210             ; ROUTINE. THIS WILL PROVIDE THE SYM
                0220             ; WITH THIS PROGRAM AUTOMATICALLY.
                0230
                0240 ACCESS      .DE $8B86
                0250 PBDA        .DE $A402
                0260 TOUT        .DE $8AA0
                0270
                0280             .BA $8BA7
                0290
```

```
8BA7- 20 86 8B  0300            JSR ACCESS
8BAA- 48        0310            PHA
8BAB- AD 02 A4  0320 WAIT       LDA PBDA
8BAE- 29 40     0330            AND #$40
8BB0- F0 F9     0340            BEQ WAIT
8BB2- 68        0350            PLA
8BB3- 4C A0 8A  0360            JMP TOUT
                0370
                0380            .EN
```

A RAE/BASIC LINKER - BY M. A. CUSITER

Take a good look at lines 120 - 330 in the "PROGRAMME LISTING" below. While the main program is in BASIC, these lines are written in RAE-1 format! The BASIC program actually calls on RAE to assemble object code for it (BASIC) to use.

While we have seen assemblers written in BASIC before, these were usually slow, and much too long to include within a BASIC application program. The USR call in line 20 is to the object code stored by the ASS / BAS LINKER EXTENSION ROUTINE which immediately follows the RUN and LISTing of the BASIC program. The USR call in line 350 is to the object code which "LINKER" prepared from the "source code" in lines 120 - 330 inclusive.

A study of "LINKER" will reveal many details of the inner workings of both RAE and BASIC. Be sure to reserve memory space for the USR calls!

```
MEMORY SIZE? 8192
WIDTH? 80

 7679 BYTES FREE

BASIC V1.1
COPYRIGHT 1978 SYNERTEK SYSTEMS CORP.

OK
.LOD LINK3

OK
RUN
```

DEMONSTRATION OF LINKER PROGRAM

The linker expects text to be assembled enclosed in [[....]
The linker as it stands is called prior to an assembly routine
but could simply be changed so that it need be called once only.
The linker evapourates after use and does not need any zpage
locations. It assembles RAE text at BASIC's variables + $100
to make room for zpage storage. The available space is
divided into 3/4 text, 1/4 labels. An overflow in text gives
an 'ASSEMBLER TEXT OVERFLOW' error.

The linker tolerates tokens in
the BASIC text by fixing +, -, and =, and converting
the others to letters. This could give rise to
the rare duplicate label, but this can be completely
avoided by using lower case.

PROGRAMME LISTING

```
 10 PRINTTAB(20)"DEMONSTRATION OF LINKER PROGRAM"
 20 X=USR(&"3000",0)
 30 PRINT"The linker expects text to be assembled enclosed in [[ ....]"
 40 PRINT"The linker as it stands is called prior to an assembly routine"
 50 PRINT"but could simply be changed so that it need be called once only."
 60 PRINT"The linker evapourates after use and does not need any zpage"
 70 PRINT"locations.  It assembles RAE text at BASIC's variables + $100"
 80 PRINT"to make room for zpage storage.  The available space is"
 90 PRINT"divided into 3/4 text, 1/4 labels.  An overflow in text gives"
 100 PRINT"an 'ASSEMBLER TEXT OVERFLOW' error."
 105 :
 110 REM - START OF ASSEMBLY CODE
 120 [[ .BA $2000 ;the linker inserts a space at the beginning of each
 130 ;line only.: ;this means that labels must always: ;be preceded
 140 ;by colons.: ;more than one statement per line
 150 ;new line, new statement
 160 .OS :TOUT .DE $8AA0 ;linker changes TOut to something else
 170 LDX #0
 180 :MESS LDA MESSAGE,X
 190 BEQ OUT
 200 JSR TOUT ;output character
 270 INX: BNE MESS :OUT JMP $D14C ;return to BASIC
 280 :MESSAGE .by $0a $0d 'The linker tolerates tokens in'
 290 .by $0a $0d 'the BASIC text by fixing +, -, and =, and converting'
 300 .by $0a $0d 'the others to letters.  This could give rise to'
 310 .by $0a $0d 'the rare duplicate label, but this can be completely'
 320 .by $0a $0d 'avoided by using lower case.' $0a $0d $0d $00
 330 .en]
 340 REM - END OF ASSEMBLY CODE
 345 :
 350 X=USR(&"2000",0):REM - CALL OUR ROUTINE!
 360 PRINT:PRINTTAB(20)"PROGRAMME LISTING":PRINT:LIST
OK
```

```
                0010
                0020            ;****************************************
                0030            ;* ASS / BAS LINKER EXTENSION ROUTINE   *
                0040            ;*      WRITTEN BY Dr. M. A. Cusiter    *
                0050            ;****************************************
                0060
                0070            .BA $3000
                0080            .OS
                0090
                0100            ;VARIOUS STORES FOR POINTERS, REGISTERS
                0110
0120 LINNUM     .DE $1C
0130 STST       .DE $83
0140 VEND       .DE $81
0150 RTXTPTR    .DE $AD
0160 BLOK.2     .DE $AF
0170 ESTOR      .DE $EF         ;ECHO STOR
0180 VESTOR     .DE $F0
0190 TXTPTR     .DE $D3
0200 RETAD      .DE $F4
0210 STSTOR     .DE $F5
0220
0230            ;PAGE ONE VECS
0240
0250 RTXST      .DE $100
0260 RTXEN      .DE $102
0270 RLST       .DE $104
0280 RLEN       .DE $106
0290
0300            ;MONITOR AND SYS RAM
0310
0320 ACCESS     .DE $8B86
0330 OUTVEC     .DE $A664
0340 TECHO      .DE $A653
0350 TOUT       .DE $8AA0
0360
```

```
                    0370            ;BASIC
                    0380
                    0390 CHRGET     .DE $CC
                    0400 CHRGOT     .DE $D2
                    0410
                    0420            ;REVECTOR BASOUT
                    0430
                    0440 !!!TV      .MD (OLD NEW) ;TRANSFER VEC
                    0450            LDA OLD
                    0460            STA NEW
                    0470            LDA OLD+1
                    0480            STA NEW+1
                    0490            .ME
                    0500
                    0510 !!!CV      .MD (DATA ADDRS) ;CHANGE VEC
                    0520            LDA #L,DATA
                    0530            STA ADDRS
                    0540            LDA #H,DATA
                    0550            STA ADDRS+1
                    0560            .ME
                    0570
                    0580            .ES
                    0590
3000- 20 86 8B      0600            JSR ACCESS
                    0610            TV (OUTVEC OLDVEC)

3003- AD 64 A6
3006- 8D 21 30
3009- AD 65 A6
300C- 8D 22 30

                    0620            CV (TRAPOUT OUTVEC)

300F- A9 1C
3011- 8D 64 A6
3014- A9 30
3016- 8D 65 A6

3019- 4C 4C D1      0630            JMP $D14C     ;BACK TO BASIC
                    0640
301C- C9 0D         0650 TRAPOUT    CMP #$0D
301E- F0 03         0660            BEQ ANALYSE
                    0670
3020- 4C            0680            .BY $4C
3021-               0690 OLDVEC     .DS 2
                    0700
3023- 20 D2 00      0710 ANALYSE    JSR CHRGOT
3026- C9 5B         0720            CMP #'[
3028- F0 04         0730            BEQ PROSTAK   ;ASSEMBLY COMING UP
302A- A9 0D         0740            LDA #$0D
302C- D0 F2         0750            BNE OLDVEC-1
302E- A2 0D         0760 PROSTAK    LDX #13
3030- 68            0770 PULL       PLA
3031- CA            0780            DEX
3032- 10 FC         0790            BPL PULL
                    0800
                    0810            ;BEGIN MAINLINE
                    0820
3034- AE 53 A6      0830 ASSBAS     LDX TECHO     ;WHATEVER IT IS
3037- 86 EF         0840            STX *ESTOR
3039- BA            0850            TSX
303A- 86 F5         0860            STX *STSTOR
303C- A5 81         0870            LDA *VEND
303E- 85 F0         0880            STA *VESTOR
3040- A5 82         0890            LDA *VEND+1
```



```
3042- 85 F1         0900            STA *VESTOR+1
3044- AD 00 01      0910            LDA RTXST     ;GET POINTER TO PAGE 1
3047- 85 AF         0920            STA *BLOK.2
3049- AD 01 01      0930            LDA RTXST+1
304C- 8D B0 00      0940            STA BLOK.2+1
304F- 68            0950            PLA
3050- 85 F4         0960            STA *RETAD
3052- 68            0970            PLA
3053- 85 F5         0980            STA *RETAD+1          ;STORE RETURN ADDRS
                    0990
                    1000            ;CALCULATE FILE BOUNDARIES FOR RAE
                    1010
3055- 18            1020            CLC
3056- A5 81         1030            LDA *VEND     ;LEAVE SPACE FOR ZPAGE
3058- 8D 00 01      1040            STA RTXST
305B- A5 82         1050            LDA *VEND+1
305D- 69 01         1060            ADC #1
305F- 8D 01 01      1070            STA RTXST+1
                    1080
                    1090            ;CALCULATE BYTES AVAILABLE
                    1100            ;FOR RAE FILES
                    1110
3062- 38            1120            SEC  ;CALCULATE RLEN ADDRS
3063- A5 83         1130            LDA *STST
3065- E9 04         1140            SBC #4
3067- 8D 06 01      1150            STA RLEN
306A- A5 84         1160            LDA *STST+1
306C- E9 00         1170            SBC #0
306E- 8D 07 01      1180            STA RLEN+1    ;4 BYTES OFF
3071- 38            1190            SEC
3072- AD 06 01      1200            LDA RLEN
3075- ED 00 01      1210            SBC RTXST
3078- 8D 04 01      1220            STA RLST      ;CONTAINS LEN FRE SPACE
307B- AD 07 01      1230            LDA RLEN+1
307E- ED 01 01      1240            SBC RTXST+1
3081- 8D 05 01      1250            STA RLST+1
3084- 18            1260            CLC  ;FIND 1/4 FRE SPACE
3085- 6E 05 01      1270            ROR RLST+1
3088- 6E 04 01      1280            ROR RLST
308B- 18            1290            CLC
308C- 6E 05 01      1300            ROR RLST+1
308F- 6E 04 01      1310            ROR RLST
3092- 38            1320            SEC  ;SUBTRACT 1/4FS FROM RLEN
3093- AD 06 01      1330            LDA RLEN
3096- ED 04 01      1340            SBC RLST
3099- 8D 04 01      1350            STA RLST
309C- AD 07 01      1360            LDA RLEN+1
309F- ED 05 01      1370            SBC RLST+1
30A2- 8D 05 01      1380            STA RLST+1
30A5- 38            1390            SEC  ;SET RTXEN
30A6- AD 04 01      1400            LDA RLST
30A9- E9 04         1410            SBC #4
30AB- 8D 02 01      1420            STA RTXEN
30AE- AD 05 01      1430            LDA RLST+1
30B1- E9 00         1440            SBC #0
30B3- 8D 03 01      1450            STA RTXEN+1  ;4 BYTES BELOW
                    1460
30B6- A0 00         1470            LDY #0
                    1480
                    1490            ;GENERATE RAE TXT
                    1500
30B8- 20 CC 00      1510 ASSTXT     JSR CHRGET    ;GET NXT CHR FROM BAS
30BB- D0 06         1520            BNE PASS.1    ;END OF LINE?
30BD- 20 03 32      1530            JSR NXTLINE   ;INCREMENT PAST 4 BYTES
30C0- 20 CC 00      1540            JSR CHRGET    ;GET NXT CHR
```

```
30C3- C9 5B      1550 PASS.1     CMP #'[
30C5- F0 03      1560            BEQ SETUP
30C7- 4C B6 CC   1570            JMP $CCB6     ;ERRORMSGE
30CA- A0 00      1580 SETUP      LDY #0
30CC- 84 1C      1590            STY $LINNUM
30CE- 84 1D      1600            STY $LINNUM+1         ;SET LINE NUMBER TO 0
30D0- AD 00 01   1610            LDA RTXST
30D3- 85 AD      1620            STA $RTXTPTR          ;GET START
30D5- AD 01 01   1630            LDA RTXST+1
30D8- 85 AE      1640            STA $RTXTPTR+1
30DA- 20 0C 32   1650            JSR INCLN     ;FIRST LINE = 1
                 1660
                 1670            ;ASSEMBLE RAE TXT
                 1680
30DD- 20 27 32   1690 TEXTIN     JSR COMPARE
30E0- 20 FA 31   1700            JSR BINCPTR   ;INC BAS PTR
30E3- C9 3A      1710            CMP #':       ;NEW COMMAND?
30E5- D0 0C      1720            BNE CHECKEND
30E7- 20 3F 32   1730            JSR SETEND    ;SET BIT 7
30EA- 20 38 32   1740            JSR RINCPTR   ;INC RAE PTR
30ED- 20 0C 32   1750            JSR INCLN
30F0- 4C DD 30   1760            JMP TEXTIN
30F3- C9 00      1770 CHECKEND   CMP #0        ;END LINE?
30F5- D0 16      1780            BNE ASSEND
30F7- 20 3F 32   1790            JSR SETEND
30FA- 20 38 32   1800            JSR RINCPTR
30FD- 20 0C 32   1810            JSR INCLN
3100- 20 38 32   1820            JSR RINCPTR   ;PUT IN SPACE
3103- A9 20      1830            LDA #$20      ;AFTER NEW LINE
3105- 91 AD      1840            STA (RTXTPTR),Y
3107- 20 03 32   1850            JSR NXTLINE   ;SKIP 4 BYTES BAS TXT
310A- 4C DD 30   1860            JMP TEXTIN
310D- C9 5D      1870 ASSEND     CMP #']       ;END OF ASSEMBLY?
310F- F0 26      1880            BEQ MARKEND
3111- C9 A4      1890            CMP #$A4      ;PLUS TOKEN
3113- D0 04      1900            BNE MINUS
3115- A9 2B      1910            LDA #$2B      ;FIX IT
3117- D0 0E      1920            BNE STORCHR
3119- C9 A5      1930 MINUS      CMP #$A5      ;MINUS TOKEN
311B- D0 04      1940            BNE EQUALS
311D- A9 2D      1950            LDA #$2D
311F- D0 06      1960            BNE STORCHR
3121- C9 AC      1970 EQUALS     CMP #$AC      ;EQUALS TOKEN
3123- D0 02      1980            BNE STORCHR
3125- A9 3D      1990            LDA #$3D
3127- C9 7F      2000 STORCHR    CMP #$7F      ;ANY MORE TOKENS?
3129- 90 04      2010            BCC STORCHAR
312B- 09 41      2020            ORA #$41      ;YES THERE ARE
312D- 29 7F      2030            AND #$7F      ;TRANSFORM IT
312F- 20 38 32   2040 STORCHAR   JSR RINCPTR   ;STORE CHAR IN TEXT
3132- 91 AD      2050            STA (RTXTPTR),Y
3134- 4C DD 30   2060            JMP TEXTIN
3137- 20 3F 32   2070 MARKEND    JSR SETEND    ;MARK OFF END
313A- 98         2080            TYA
313B- A2 03      2090            LDX #3
313D- 20 38 32   2100 ZEND       JSR RINCPTR   ;MARK OFF END
3140- 91 AD      2110            STA (RTXTPTR),Y       ;OF RAE TXT
3142- CA         2120            DEX  ;WITH 3 ZEROS
3143- D0 F8      2130            BNE ZEND
                 2140
                 2150            ;START SHIFTING Z-PAGE ,ETC
                 2160
3145- A2 00      2170            LDX #0
3147- B9 00 00   2180 SHIFTOUT   LDA 0,Y
314A- 91 F0      2190            STA (VESTOR),Y
```



```
314C- 8A         2200            TXA  ;FILL WITH 00
314D- 99 00 00   2210            STA 0,Y
3150- C8         2220            INY
3151- C0 F0      2230            CPY #$F0      ;ONLY UP TO $EF
3153- D0 F2      2240            BNE SHIFTOUT
                 2250
3155- A0 00      2260            LDY #0        ;NOW PAGE 1
3157- 99 08 01   2270 ZERO.1     STA $108,Y
315A- C8         2280            INY
315B- C0 2E      2290            CPY #$2E
315D- D0 F8      2300            BNE ZERO.1
                 2310
                 2320            ;FILL RAE BUFF WITH $20
                 2330
315F- A0 00      2340            LDY #0
3161- A9 20      2350            LDA #$20      ;SPACE
3163- 99 35 01   2360 FILLBUF    STA $135,Y
3166- C8         2370            INY
3167- C0 56      2380            CPY #$56
3169- D0 F8      2390            BNE FILLBUF
                 2400
                 2410            ;STORE OLD OUTVEC, PATCH NEW
                 2420
316B- 20 86 8B   2430            JSR ACCESS
316E- AD 64 A6   2440            LDA OUTVEC
                 2450            CV (ASSEM OUTVEC)

3171- A9 E2
3173- 8D 64 A6
3176- A9 31
3178- 8D 65 A6

                 2460
317B- 4C 03 B0   2470            JMP $B003     ;INITIALISE RAE
                 2480
317E- 29 7F      2490 ASSEMBLE   AND #$7F
3180- C9 3E      2500            CMP #'>
3182- F0 20      2510            BEQ RESTORE   ;PROMPT?
3184- C9 07      2520            CMP #7        ;BEL
3186- F0 01      2530            BEQ PRINTERR
3188- 60         2540            RTS  ;BACK
3189- 20 86 8B   2550 PRINTERR   JSR ACCESS
318C- 48         2560            PHA
                 2570            CV (ERROUT OUTVEC)

318D- A9 9B
318F- 8D 64 A6
3192- A9 31
3194- 8D 65 A6

3197- 68         2580            PLA
3198- 4C A0 8A   2590            JMP TOUT
319B- 29 7F      2600 ERROUT     AND #$7F
319D- C9 3E      2610            CMP #'>
319F- F0 03      2620            BEQ RESTORE
31A1- 4C A0 8A   2630            JMP TOUT
31A4- 20 86 8B   2640 RESTORE    JSR ACCESS
                 2650            TV (OLDVEC OUTVEC)

31A7- AD 21 30
31AA- 8D 64 A6
31AD- AD 22 30
31B0- 8D 65 A6

                 2660
```

```
31B3- A0 00      2670            LDY #0        ;SHIFT BACK VECS
31B5- B1 F0      2680 SHIFTIN    LDA (VESTOR),Y
31B7- 99 00 00   2690            STA 0,Y
31BA- C8         2700            INY
31BB- C0 F1      2710            CPY #$F1
31BD- D0 F6      2720            BNE SHIFTIN
31BF- A5 AF      2730            LDA *BLOK.2
31C1- 8D 00 01   2740            STA RTXST
31C4- A5 B0      2750            LDA *BLOK.2+1
31C6- 8D 01 01   2760            STA RTXST+1
31C9- A9 00      2770            LDA #0
31CB- 8D 02 01   2780            STA RTXST+2
31CE- 8D 03 01   2790            STA RTXST+3
                 2800
31D1- A6 EF      2810            LDX *ESTOR    ;RESTORE ECHO STATE
31D3- 8E 53 A6   2820            STX TECHO
31D6- A6 F5      2830            LDX *STSTOR
31D8- 9A         2840            TXS
31D9- A5 F5      2850            LDA *RETAD+1
31DB- 48         2860            PHA
31DC- A5 F4      2870            LDA *RETAD
31DE- 48         2880            PHA
31DF- 4C CC 00   2890            JMP CHRGET    ;BACK TO BASIC
                 2900
31E2- 20 86 8B   2910 ASSEM      JSR ACCESS
                 2920            CV (ASSEMBLE OUTVEC)

31E5- A9 7E
31E7- 8D 64 A6
31EA- A9 31
31EC- 8D 65 A6

31EF- 68         2930            PLA
31F0- 68         2940            PLA  ;REMOVE RET. ADDRS
31F1- A9 20      2950            LDA #$20      ;SET REGS
31F3- A0 00      2960            LDY #0
31F5- A2 00      2970            LDX #0
31F7- 4C FC B0   2980            JMP $B0FC     ;START ASSEMBLY
                 2990
                 3000            ;SUBROUTINES FOLLOW
                 3010
31FA- E6 D3      3020 BINCPTR    INC *TXTPTR
31FC- D0 02      3030            BNE INCP
31FE- E6 D4      3040            INC *TXTPTR+1
3200- B1 D3      3050 INCP       LDA (TXTPTR),Y
3202- 60         3060            RTS
                 3070
3203- A2 04      3080 NXTLINE    LDX #4
3205- 20 FA 31   3090 NXTBYT     JSR BINCPTR
3208- CA         3100            DEX
3209- D0 FA      3110            BNE NXTBYT
320B- 60         3120            RTS
                 3130
320C- F8         3140 INCLN      SED  ;RAE LINES IN DECIMAL
320D- 18         3150            CLC
320E- A5 1C      3160            LDA *LINNUM
3210- 69 01      3170            ADC #1
3212- 85 1C      3180            STA *LINNUM
3214- A5 1D      3190            LDA *LINNUM+1
3216- 69 00      3200            ADC #0
3218- 85 1D      3210            STA *LINNUM+1
321A- D8         3220            CLD
321B- A5 1C      3230            LDA *LINNUM
321D- 91 AD      3240            STA (RTXTPTR),Y      ;PUT IT IN TXT
321F- 20 38 32   3250            JSR RINCPTR
```



```
3222- A5 1D      3260            LDA *LINNUM+1
3224- 91 AD      3270            STA (RTXTPTR),Y
3226- 60         3280            RTS
                 3290
3227- 38         3300 COMPARE    SEC
3228- AD 02 01   3310            LDA RTXEN
322B- E5 AD      3320            SBC *RTXTPTR          ;CHECK TO SEE
322D- AD 03 01   3330            LDA RTXEN+1  ;IF ENOUGH SPACE
3230- E5 AE      3340            SBC *RTXTPTR+1
3232- B0 03      3350            BCS CLEAR
3234- 4C 46 32   3360            JMP ATOMESS
3237- 60         3370 CLEAR      RTS
                 3380
3238- E6 AD      3390 RINCPTR    INC *RTXTPTR
323A- D0 02      3400            BNE RINCP
323C- E6 AE      3410            INC *RTXTPTR+1
323E- 60         3420 RINCP      RTS
                 3430
                 3440 SETEND
323F- B1 AD      3450            LDA (RTXTPTR),Y
3241- 09 80      3460            ORA #$80
3243- 91 AD      3470            STA (RTXTPTR),Y
3245- 60         3480            RTS
                 3490
3246- A2 00      3500 ATOMESS    LDX #0
3248- BD 56 32   3510 MSG.1      LDA MESS.1,X
324B- F0 06      3520            BEQ FIN.1
324D- 20 A0 8A   3530            JSR TOUT
3250- E8         3540            INX
3251- D0 F5      3550            BNE MSG.1
3253- 4C 7E C2   3560 FIN.1      JMP $C27E     ;BAS WARM
                 3570
3256- 41 53 53   3580 MESS.1     .BY 'ASSEMBLER TEXT OVERFLOW' $0D $0A $00
3259- 45 4D 42
325C- 4C 45 52
325F- 20 54 45
3262- 58 54 20
3265- 4F 56 45
3268- 52 46 4C
326B- 4F 57 0D
326E- 0A 00
                 3590            .EN
```

A MODEM INTERFACE PROGRAM FOR SYM

The KTM-2/80 (or KTM-2) can be connected directly into any modem which will accept inverted TTL voltage levels (< 0.8 V = logic one, > 2.8 V = logic zero) as well as standard RS-232-C (EIA) levels (+/- 3 V approx). This includes all modems which use the 1488/1489 EIA transceiver chip pair. With some older modems it may be necessary to bring a -5 V supply voltage to the KTM-2 and change the appropriate jumpers. This terminal-modem combination will allow you to communicate with any of the time-share systems to which you arrange access.

Unfortunately, however, the data you receive in this way is evanescent. This problem is easily solved by getting SYM into the system to record the incoming data. How to do this is described in the following paragraphs:

First, the 20 mA current loops (CL interface), both input and output, must be converted to EIA (or inverted TTL) for interfacing SYM to the modem. While this can be done by modifications directly on the SYM board itself, by "rebuilding" the CL interface into a "twin" of the existing EIA interface (spare inverters are available on-board the SYM which may be used for this purpose, as pointed out in an earlier issue), we prefer an alternate approach, for two reasons.

First, we have several CL devices which we wish to continue using, e.g., a decwriter II, a KSR-35 TTY (really!), etc. Second, we have some ancient modems around, which we occasionally use; a couple of these put out voltages as high as +/- 25 V, and we don't like the idea of bringing such high voltage levels to the SYM.

We recommend converting CL to and from EIA with a pair of optoisolaters at the modem end of the CL line from the SYM. Placement at the modem end of the line is suggested because if the modem requires bipolar input signals, i.e., +/- 3 V or greater, +/-12 V will be available somewhere around the modem itself for this purpose. Unfortunately, you will need to bring an additional wire from the SYM to the (SYM) receive optoisolater, with +5 V. This is because the SYM is designed to be the "active" element in both the transmit and receive CLs.

The term "active" in CL systems is used to describe the unique element in any CL serial chain which incorporates the "battery", or current source for the entire loop. Two "batteries" are required for full duplex systems, one in each of the two required loops. The SYM provides both.

Anyway, once you have interfaced your SYM to the modem over the CL interface (and with the KTM-2 or other terminal through the EIA interface), the following program, by Jeff Lavin, will enable the SYM to store "incoming" files from the "remote" system. We have not yet actually used the program, but we know it works, since Jeff has provided us with a number of data files he has down-loaded from various "sources" (he also provided us with a long listing of such sources).

One such data file, "SUSAN", is reproduced (partially only, the original was much longer) on pages 13/14-21 through 13/14-27, and the program on page 13/14-28 to print that data file was abstracted from this MODEM COMMUNICATION PROGRAM. The communication protocols involved are easily deduced by studying the comments in the source code.

```
0010 ;  MODEM COMMUNICATION PROGRAM
0015 ;  JEFF LAVIN - 1982
0020
0030 ;  When uploading, it is very IMPORTANT
0040 ;  to be sure an EOT Char ($04 = ^D)
0050 ;  is the last character of the program.
0060
0070 MODFLG     .DE $FA
0080 LODFLG     .DE $FB
0090 WARM       .DE $8003
0100 SAVER      .DE $8188
0110 INCCMP     .DE $82B2
0120 CRLFSZ     .DE $8316
0130 OUTQM      .DE $8320
0140 SPACE      .DE $8342
0150 CRLF       .DE $834D
0160 INSTAT     .DE $8386
0170 OUTCHR     .DE $8A47
0180 INTCHR     .DE $8A58
0190 TIN        .DE $8A6A
0200 ACCESS     .DE $8B86
0210 PBDA       .DE $A402
0220 TECHO      .DE $A653
0230 TOUTFL     .DE $A654
0240 INVEC      .DE $A660
0250 LINK       .DE $F03A     ;PUT YOUR OWN PRINTER LINK HERE
0260
0270            .BA $9000
0280 ; .OS
0290
```



```
9000- 20 86 8B  0300 MODINIT    JSR ACCESS
9003- A9 20     0310            LDA #L,MODEM
9005- 8D 61 A6  0320            STA INVEC+1
9008- A9 90     0330            LDA #H,MODEM
900A- 8D 62 A6  0340            STA INVEC+2
900D- A9 80     0350            LDA #$80
900F- 8D 53 A6  0360            STA TECHO
9012- A2 D0     0370            LDX #$D0
9014- 8E 54 A6  0380            STX TOUTFL
9017- A9 40     0390            LDA #$40
9019- 85 FA     0400            STA *MODFLG
901B- A9 00     0410            LDA #$00
901D- 85 FB     0420            STA *LODFLG
901F- 60        0430            RTS
                0440
9020- 20 6A 91  0450 MODEM      JSR NEWCHR    Get a char & determine source
9023- 29 7F     0460            AND #$7F      Strip possible parity
9025- A2 D0     0470            LDX #$D0
9027- 8E 54 A6  0480            STX TOUTFL    Turn off TTY OUT
902A- 24 FA     0490            BIT *MODFLG   Check mode
902C- 30 5D     0500            BMI TOMON     To MON if bit 7 is set
902E- 24 FB     0510            BIT *LODFLG   Check mode
9030- 30 62     0520            BMI DOWNLD    To download prgrm if bit 7 set
9032- 70 78     0530            BVS UPLOAD    To upload prgrm if bit 6 set
                0540
9034- C9 1B     0550            CMP #$1B      Esc ?  (MONITOR select char)
9036- D0 E8     0560            BNE MODEM     Else, get next char
9038- 20 58 8A  0570            JSR INTCHR    Get next char, do not send
903B- 29 7F     0580            AND #$7F      Strip parity
903D- C9 4D     0590            CMP #'M       M ? (MONITOR select char)
903F- F0 35     0600            BEQ GOMON
9041- C9 51     0610            CMP #'Q       Q ? (Toggle Keybd echo)
9043- F0 28     0620            BEQ TOGGL
9045- C9 50     0630            CMP #'P       P ? (Print select char)
9047- F0 34     0640            BEQ GO.PRNT
9049- C9 44     0650            CMP #'D       D ? (Download select char)
904B- F0 0E     0660            BEQ GO.DNLD
904D- C9 55     0670            CMP #'U       U ? (Upload select char)
904F- D0 CF     0680            BNE MODEM
                0690
9051- A9 40     0700 GO.UPLD    LDA #$40
9053- 85 FB     0710            STA *LODFLG   Set bit 6
9055- 20 58 91  0720            JSR SETPTRS
9058- 4C AC 90  0730            JMP UPLOAD
                0740
905B- 38        0750 GO.DNLD    SEC           Set carry
905C- 66 FB     0760            ROR *LODFLG   Roll carry into bit 7
905E- 20 58 91  0770            JSR SETPTRS
9061- 20 16 83  0780            JSR CRLFSZ
9064- 20 4D 83  0790            JSR CRLF
9067- 20 4D 83  0800            JSR CRLF
906A- 4C 20 90  0810            JMP MODEM
                0820
906D- A9 40     0830 TOGGL      LDA #$40      Mask bit 6
906F- 45 FA     0840            EOR *MODFLG   Invert bit 6
9071- 85 FA     0850            STA *MODFLG
9073- 4C 20 90  0860            JMP MODEM
                0870
9076- A9 C0     0880 GOMON      LDA #$C0      Set echo & go to MON
9078- 85 FA     0890            STA *MODFLG
907A- 4C 03 80  0900            JMP WARM      Go to MON and print prompt.
                0910
907D- 20 3A F0  0920 GO.PRNT    JSR LINK      Link printer
9080- A0 00     0930            LDY #$00
9082- 20 61 91  0940            JSR DELAY
9085- 20 58 91  0950            JSR SETPTRS
```

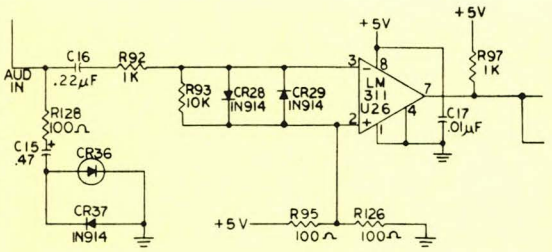
```
9088- 4C 1D 91  0960              JMP PRINT
                0970
908B- C9 04     0980 TOMON       CMP #$04      ^D ?  (EOT Char)
908D- D0 04     0990             BNE =+5       No, pass it on
908F- A9 40     1000             LDA #$40
9091- 85 FA     1010             STA *MODFLG   Yes, clear MON mode
9093- 60        1020             RTS
                1030
9094- C9 04     1040 DOWNLD      CMP #$04      ^D ?  (EOT Char)
9096- D0 0A     1050             BNE =+11      No, pass it on
9098- 48        1060             PHA
9099- 06 FB     1070             ASL *LODFLG   Clear download mode
909B- 20 16 83  1080             JSR CRLFSZ
909E- 20 4D 83  1090             JSR CRLF
90A1- 68        1100             PLA
90A2- A0 00     1110 LD>MEM      LDY #$00      Y is index
90A4- 91 FE     1120             STA ($FE),Y   Store ASCII in sequential
90A6- 20 B2 82  1130             JSR INCCMP    memory locations beginning
90A9- 4C 20 90  1140             JMP MODEM     at $200
                1150
90AC- A2 B0     1160 UPLOAD      LDX #$B0      Enable CRT & TTY out
90AE- 8E 54 A6  1170             STX TOUTFL    CRT only in
90B1- A0 00     1180             LDY #$00
90B3- 20 61 91  1190             JSR DELAY
90B6- 20 61 91  1200             JSR DELAY
90B9- A0 04     1210             LDY #$04
90BB- A9 00     1220             LDA #$00
90BD- 20 47 8A  1230 NULLS       JSR OUTCHR    Send null
90C0- 88        1240             DEY
90C1- D0 FA     1250             BNE NULLS
90C3- A0 00     1260 MEM>OUT     LDY #$00      Y is index
90C5- B1 FE     1270             LDA ($FE),Y   Get char at memory loc
90C7- C9 0D     1280             CMP #$0D      CR ? (End of line?)
90C9- D0 2B     1290             BNE ^D.CHECK
90CB- 48        1300             PHA           Save char
90CC- A9 20     1310             LDA #$20      Space
90CE- 20 47 8A  1320             JSR OUTCHR    Print it
90D1- 68        1330             PLA           Retrieve char
90D2- 20 47 8A  1340             JSR OUTCHR    Send & print char
90D5- 20 B2 82  1350             JSR INCCMP
90D8- B1 FE     1360             LDA ($FE),Y   Get next char
90DA- C9 0A     1370             CMP #$0A      LF ?
90DC- D0 0B     1380             BNE GETPRMPT
90DE- A2 90     1390             LDX #$90
90E0- 8E 54 A6  1400             STX TOUTFL    Only CRT in & out
90E3- 20 47 8A  1410             JSR OUTCHR    Print LF
90E6- 20 B2 82  1420             JSR INCCMP
90E9- A2 D0     1430 GETPRMPT    LDX #$D0
90EB- 8E 54 A6  1440             STX TOUTFL    Restore TTY in
90EE- A0 40     1450             LDY #$40
90F0- 20 61 91  1460             JSR DELAY
90F3- 4C 20 90  1470             JMP MODEM     Wait for response
90F6- C9 04     1480 ^D.CHECK    CMP #$04      ^D ?  (EOT Char)
90F8- F0 0B     1490             BEQ =+12
90FA- 20 47 8A  1500             JSR OUTCHR    Send & print char
90FD- 20 B2 82  1510             JSR INCCMP
9100- 20 86 83  1520             JSR INSTAT    Brk if key down
9103- 90 BE     1530             BCC MEM>OUT
9105- 20 42 83  1540             JSR SPACE
9108- 20 16 83  1550             JSR CRLFSZ
910B- 20 4D 83  1560             JSR CRLF
910E- 20 4D 83  1570             JSR CRLF
```



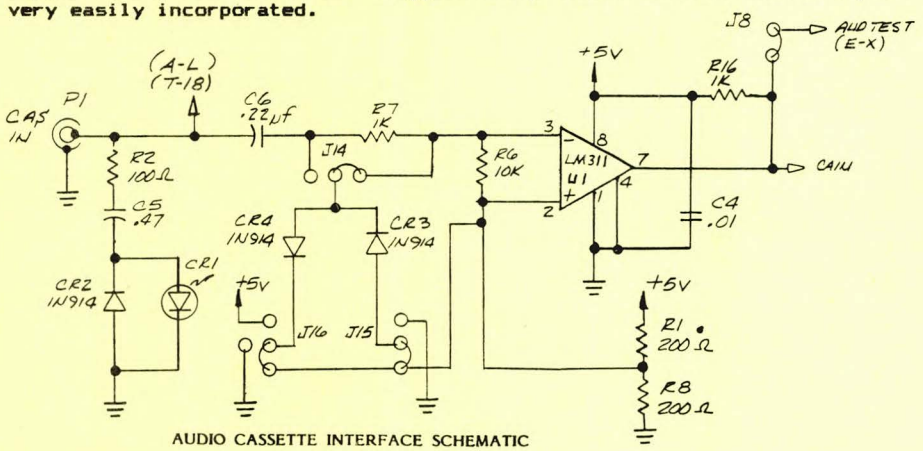
```
9111- A9 D0     1580             LDA #$D0      Restore previous setting
9113- 8D 54 A6  1590             STA TOUTFL
9116- A9 00     1600             LDA #$00
9118- 85 FB     1610             STA *LODFLG   Clear upload mode
911A- 4C 20 90  1620             JMP MODEM
                1630
911D- A2 B0     1640 PRINT       LDX #$B0
911F- 8E 54 A6  1650             STX TOUTFL
9122- A0 00     1660 MEM>PRNT    LDY #$00
9124- B1 FE     1670             LDA ($FE),Y
9126- C9 04     1680             CMP #$04
9128- F0 0B     1690             BEQ =+12
912A- 20 47 8A  1700             JSR OUTCHR
912D- 20 B2 82  1710             JSR INCCMP
9130- 20 86 83  1720             JSR INSTAT
9133- 90 ED     1730             BCC MEM>PRNT
9135- 20 4D 83  1740             JSR CRLF
9138- 20 20 83  1750             JSR OUTSM     Print "?"
913B- 20 42 83  1760             JSR SPACE
913E- 20 58 8A  1770             JSR INTCHR    Get a char
9141- 29 7F     1780             AND #$7F
9143- C9 59     1790             CMP #'Y       Is it Yes?
9145- D0 03     1800             BNE =+4
9147- 20 16 83  1810             JSR CRLFSZ    If so, print addr
914A- 20 4D 83  1820             JSR CRLF
914D- 20 4D 83  1822             JSR CRLF
9150- A9 D0     1830             LDA #$D0
9152- 8D 54 A6  1840             STA TOUTFL
9155- 4C 20 90  1850             JMP MODEM
                1860
9158- A9 02     1870 SETPTRS     LDA #$02
915A- 85 FF     1880             STA *$FF      ADH Store memory
915C- A9 00     1890             LDA #$00
915E- 85 FE     1900             STA *$FE      ADL Store memory
9160- 60        1910             RTS
                1920
9161- A2 00     1930 DELAY       LDX #$00
9163- CA        1940 DELOOP      DEX
9164- D0 FD     1950             BNE DELOOP
9166- 88        1960             DEY
9167- D0 FA     1970             BNE DELOOP
9169- 60        1980             RTS
                1990
                2000 ; THIS ROUTINE ONLY WORKS AT 300 BAUD
                2010
916A- 20 88 81  2020 NEWCHR      JSR SAVER     Copy part of INTCHR here
916D- A9 00     2030             LDA #$00
916F- 85 F9     2040             STA *$F9
9171- A9 C0     2050 LOOK        LDA #$C0      Mask all but bits 6 & 7
9173- 2C 02 A4  2060             BIT PBDA      Is there input & where?
9176- F0 F9     2070             BEQ LOOK      Loop if no input
9178- 70 0D     2080             BVS TTYIN     Branch if TTY is input source
917A- 24 FA     2090             BIT *MODFLG Input is from kybd; echo desired?
917C- 70 04     2100             BVS ECHO      Yes
917E- A0 E0     2110             LDY #$E0      No keybd echo to CRT
9180- D0 02     2120             BNE ECHO+2
9182- A0 F0     2130 ECHO        LDY #$F0      Echo keybd to modem
9184- 8C 54 A6  2140             STY TOUTFL
9187- 4C 6A 8A  2150 TTYIN       JMP TIN       Do the rest of INTCHR in MON
                2160
                2180             .EN
```

MORE ON THE CASSETTE INTERFACE

Readers of SYM-PHYSIS will, no doubt, remember the number of suggestions on improving the performance of the cassette interface which have been published during the past several years. To the right is a copy of the schematic of the SYM-1 cassette interface, from the Reference Manual.



Below, for information only, is a copy of the schematic for the SYM-2 cassette interface, which is essentially identical to that of the SYM-1. Note, however, the provision of rather extensive modifiable jumper capabilities, so that the suggestions published in SYM-PHYSIS may be very easily incorporated.



AUDIO CASSETTE INTERFACE SCHEMATIC

FDC-1 TECHNICAL NOTES - ISSUE 1

Because of the large number of SYM owners who have installed FDC-1 Disk Systems, FDC-1 Technical Notes will become a regular feature of the newsletter. Here is the first set of notes:

Number 1.1

About 10% of the FDC-1 boards seem to behave in a very erratic manner. A serendipitous fix was discovered by Jeff Lavin, who wired his board in such a way as to bring the +5V in from the SYM-1 instead of through the turret pin.

By adding the jumper shown in the figure below to the inoperative boards sent to him for trouble-shooting, Jeff got all of these boards to operate properly. The +5V can still be brought in at the turret pin as well as through pin 21.

We are not yet certain why the fix works and are discussing the problem and fix with a Synertek engineer who has been assigned to the problem.

Number 1.2

The timeout routines provided in the FDC-1 software do not set the timer

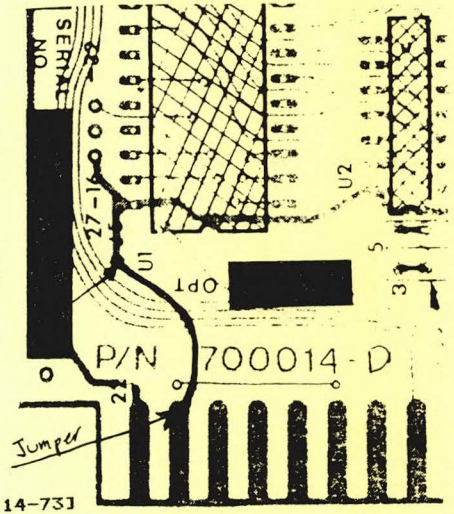
correctly, and must be rewritten. This will be done in the near future. Meanwhile, it does not matter anyway, inasmuch as the IRQ output of the 6532 has not even been connected on the SYM-1! You may wish to jumper pin 25 of the 6532 (U27) to pin 4 of the 6502 (U5) to enable the interrupt capability of the 6532.

Number 1.3

With some single-sided 5 1/4" drives, in particular those from BASF, the .L7 operation is unusually long because of the second-side search. While this may be fixed in the software, a quick hardware "fix" is as follows:

Bend up pins 9 and 10 of U14 so that they will not go into the socket. Tie them to pin 7 (GND), then replace the chip.

We can't remember who first gave us this fix, but we thank him for it.



Number 1.4

Correct the FDC-1 schematic as follows:

Pin 8 of RP1 is left unconnected.

[SEE ALSO "MORE ON FDC-1 FIX" ON PAGE 13/14-73]

```
0010 ;   Handshake through KTM 2/80
0020 ;   By V.I.Pancuska
0030 ;
0040 ;   In the following is a wiring diagram and output patch
0050 ;to enable handshake on SYM-1 when auxiliary port on
0060 ;KTM 2/80 is active
0070 ;
0080 ;   Pin 12 of U3 is enable auxiliary port
0090 ;   New IC 7400 is mounted similar way as Mr.Blalok's
0100 ;RAE 1/2
0110 ;
0120 ;
0130 ;       To pin 10 of Terminal                  From pin 4 of
0140 ;       conector on SYM - 1                    printer
0150 ;               !                                 !
0160 ;               !                                 >
0170 ;               !                                 > 3.3K
0180 ;               !                                 >
0190 ;               !                                 !
0200 ;               !     ,---------------------------!
0210 ;               !     !                 !         !
0220 ;          ,----------------------,     '-!<!-,   >
0230 ;  -----   !    !     !    -----   !          !   > 2K
0240 ;  !   !--'14  !     --------!1  !--'14       !   >
0250 ;  !   !       !   ! '----!27  !              ----!
0260 ;  ! 12!-------------      ,-!34  !                -
0270 ;  ! U !       !           !-!40  !                -
0280 ;  !   !       !           '-!50  !                -
0290 ;  ! 3 !       -------------!6   !
0300 ;,-!7  !                    ,-!7  !
0310 ;! -----                    ! -----
0320 ;'--------------------------!
0330 ;                           -
0340 ;                           -
0350 ;                           -
```

```
                0360 ;
                0370 ;   PIN 6 ON PBDA ($A402)   0=CONTINUE
                0380 ;                           1=BUSSY, WAIT
                0390 ;   TO ENABLE PATCH DO: SD F000.A664
                0400 ;
                0410 PBDA       .DE $A402
                0420 TOUT       .DE $8AA0
                0430            .BA $F000
                0440 ;
F000- AA        0450            TAX
F001- 2C 02 A4  0460 LOOP       BIT PBDA
F004- 50 FB     0470            BVC LOOP
F006- 8A        0480            TXA
F007- 4C A0 8A  0490            JMP TOUT
                0500            .EN
```

A 'LEARNING' GAME FOR SYM

The followins is an interestins little BASIC game for the SYM. It arrived too close to publication time for us to try out, but knowins Phillip Rinard as we do, we are pretty sure it should work as described.

Dear Lux: Nov. 1, 1982

I am submitting a program, NIM-WIT, for your consideration. If you would like to publish it in SYM-PHYSIS, go right ahead. Without having done anything in the field of artificial intelligence, I have found it fascinating and have been looking for a program that would be simple but show some amount of "intelligence".

The game of NIM has been around a long while and probably put into half the computers in the world. But this one has a twist: the computer is kept in the dark on one vital piece of information. The REM statements in the program review the rules and what the computer doesn't know, and therefore has to "learn". It's fun to watch it "catch on", especially when someone is playing it who doesn't know anything about computers beyond the keyboard.

Sincerely,

Phillip M. Rinard

Phillip M. Rinard

```
 10 REM         *************
 20 REM         *************
 30 REM         ** NIM-WIT **
 35 REM         *************
 40 REM         *************
 42 REM
 43 REM
 44 REM   A GAME OF "NIM" IN WHICH THE COMPUTER
 45 REM   IS AT A DISADVANTAGE UNTIL IT "LEARNS"
 46 REM   HOW BEST TO PLAY BY "WATCHING" ITS
 47 REM   HUMAN OPPONENT -- YOU!
 48 REM
 50 REM
 51 REM   ASSUMED EQUIPMENT:
 52 REM     SYM-1 WITH SUPERMON MONITOR
 53 REM     KTM-2 TERMINAL (40 OR 80 COLUMN)
 54 REM     SYNERTEK'S BASIC, OR EQUIVALENT
 60 REM
 65 REM
 70 REM       PHILLIP M. RINARD
 80 REM        1872 CAMINO UVA
 90 REM      LOS ALAMOS, NM 87544
100 REM
110 REM         OCTOBER, 1982
```



```
150 REM  RULES:
160 REM    1. HUMAN CHOOSES MAXIMUM NUMBER OF TOKENS
170 REM       THAT CAN BE REMOVED AT ONE TIME.
180 REM    2. HUMAN TELLS COMPUTER HOW MANY TOKENS
190 REM       EXIST AT THE BEGINNING.
200 REM    3. COMPUTER TAKES AWAY A NUMBER OF
210 REM       TOKENS (NO MORE THAN THE MAXIMUM).
220 REM    4. HUMAN TAKES AWAY A NUMBER OF
230 REM       TOKENS (NO MORE THAN THE MAXIMUM).
240 REM    5. COMPUTER AND HUMAN TAKE TURNS UNTIL
250 REM       NO TOKENS REMAIN.  WHOEVER TAKES
260 REM       THE LAST TOKEN IS THE WINNER.
262 REM    6. START ANOTHER GAME WITH STEP 2.
264 REM    7. TERMINATE GRACEFULLY WITH NULL ENTRY
265 REM       FOR INPUT.
270 REM
280 REM   THE COMPUTER MUST "LEARN" HOW MANY TOKENS
290 REM   CAN BE TAKEN AWAY AT ONE TIME.  AFTER
300 REM   IT DOES SO, ITS WINNING PERCENTAGE
310 REM   WILL INCREASE DRAMATICALLY.

400 PRINT CHR$(12) : REM--CLEAR THE SCREEN THROUGH KTM-2
410 PRINT : PRINT
420 PRINT"       WELCOME TO THE GAME OF "
430 PRINT : PRINT"                NIM-WIT"
440 PRINT : PRINT"<><><><><><><><><><><><><><><><><><><><><><><><><><><><><>"
450 PRINT : PRINT"I ALWAYS PLAY FIRST,"
460 PRINT"BUT ONLY YOU KNOW THE BEST STRATEGY!"
470 PRINT : PRINT
480 PRINT"CHOOSE THE MAXIMUM NUMBER OF TOKENS"
490 PRINT"THAT CAN BE TAKEN AWAY AT ONE TIME."
495 PRINT
500 PRINT"NEVER TELL ME WHAT THAT NUMBER IS!"
510 PRINT : PRINT"IT'S NOT FAIR TO CHANGE IT ON ME!"
520 PRINT : PRINT
530 REM
540 REM
550 MAX = 0 : REM--NO MAXIMUM ESTIMATED YET
560 REM
570 REM  <<< START A NEW GAME >>>
580 REM
600 PRINT
610 PRINT "WHAT NUMBER OF TOKENS DO WE START WITH?"
620 INPUT NUMBER
630 GOSUB 2000 : REM--DISPLAY TOKENS
640 REM
650 REM   <<< COMPUTER PLAYS >>>
660 REM
670 IF MAX > 0 THEN 700 : REM--FIRST TIME THROUGH?
680 MAX = 1 : TAKE = 1 : REM--FOR FIRST TIME ONLY
690 GOTO 760
695 REM
700 P=1 : REM--MULTIPLIER OF (MAX+1)
710 IF P*(MAX+1) > NUMBER THEN 730
720 P=P+1 : GOTO 710
725 REM
730 P=P-1 : REM--BEST P, BASED ON MAX
740 TAKE = NUMBER -P*(MAX+1)
750 IF TAKE = 0 THEN TAKE = INT((MAX+1)*RND(1))
760 A$="ARE" : IF NUMBER = 1 THEN A$="IS"
770 PRINT : PRINT"THERE ";A$;NUMBER;"LEFT."
780 NUMBER = NUMBER - TAKE
785 REM
790 GOSUB 5000 : REM--TIME DELAY
795 REM
```

```
800 PRINT "I'LL TAKE";TAKE;"."
805 REM
810 GOSUB 5000 : REM--TIME DELAYS
820 GOSUB 5000
825 REM
830 GOSUB 2000 : REM--DISPLAY TOKENS
840 REM
850 REM   <<< CHECK FOR COMPUTER VICTORY >>>
860 REM
870 PLAYER = 1 : REM--I.D. FOR COMPUTER
880 GOSUB 3000 : REM--VICTORY?
890 IF PLAYER = 0 THEN 600 : REM--ZERO FOR VICTORY
892 REM
893 REM   <<< HUMAN PLAYS >>>
894 REM
900 PRINT
910 PRINT"THERE ARE";NUMBER;"LEFT."
920 PRINT"HOW MANY DO YOU WANT?"
930 INPUT TAKE
940 NUMBER = NUMBER - TAKE
950 IF TAKE > MAX THEN MAX = TAKE : REM--NEW MAX ESTIMATE
960 REM
970 REM   <<< CHECK FOR HUMAN VICTORY >>>
980 REM
990 PLAYER = 2 : REM--I.D. FOR HUMAN
1000 GOSUB 3000 : REM--HUMAN VICTORY?
1010 IF PLAYER = 0 THEN 600 : REM--ZERO FOR VICTORY
1020 GOSUB 2000 : REM--DISPLAY TOKENS
1030 GOTO 700 : REM--COMPUTER'S TURN AGAIN
1040 REM
1060 REM        <<< END OF MAIN PROGRAM >>>
1070 REM
1970 REM
1980 REM  <<< SUBROUTINE TO DISPLAY TOKENS >>>
1990 REM
2000 IF NUMBER > 0 THEN 2030 : REM--THERE ARE SOME LEFT
2010 PRINT : PRINT"NONE LEFT."
2020 RETURN
2030 PRINT : PRINT : PRINT
2040 RESTORE
2045 REM
2050 DATA 6912, 18176, 6912, 20992
2060 REM   ESC   UC-G   ESC   UC-R
2070 REM--PUT KTM-2 INTO GRAPHICS MODE
2080 REM
2090 FOR I = 1 TO 4
2100 READ V : X=USR(&"8A47",V) : REM--SYM MONITOR'S OUTCHR
2110 NEXT I
2120 REM
2130 FOR I = 1 TO NUMBER
2140 PRINT "{"; : REM--THAT'S A SHIFT-ESC
2150 NEXT I
2160 REM
2170 DATA 6912, 26368, 6912, 29184
2180 REM   ESC   LC-G   ESC   LC-R
2190 REM--PUT KTM-2 INTO ALPHANUMERICS MODE
2200 REM
2210 FOR I = 1 TO 4
2220 READ V : X=USR(&"8A47",V)
2230 NEXT I
2250 PRINT
2260 RETURN
2270 REM
2280 REM   <<< SUBROUTINE TO CHECK FOR VICTORY >>>
2290 REM
```

NOTE BY LUX:

SHIFT ESC is +/- on the KTM-2, but in standard ASCII is "{" HEX $7B, DEC 123


```
3000 IF NUMBER > 0 THEN RETURN : REM--NO VICTORY
3010 PRINT : PRINT
3020 IF PLAYER = 1 THEN 3070 : REM--COMPUTER VICTORY
3030 PLAYER = 0 : REM--VICTORY
3040 PRINT"YOU DID IT...WISH I COULD!"
3050 RETURN
3060 REM
3070 PLAYER = 0
3080 PRINT"GOSH...I GOT LUCKY!"
3090 PRINT"  LET'S TRY IT AGAIN."
3100 RETURN
3110 REM
4970 REM
4980 REM   <<< SUBROUTINE FOR TIME DELAY >>>
4990 REM
5000 FOR I=1 TO 500 : NEXT I
5010 RETURN
5030 END
5040 REM
5050 REM          <<< END OF NIM-WIT >>>
```

RAM-BLINGS (~~continued~~ from page 13/14-1)

recognized by the Franchise Tax Board as a bona fide periodical, so we must ask California resident subscribers for an additional 6% sales tax!

Incidentally, we used a rather clumsy word above, "thriceannually", to indicate three times per annum. According to our dictionary, the prefix "tri-" could mean either thrice, i.e., three times per, or every third; rather ambiguous, to say the least! We once thought that triannually meant three times per year, and triennially every third year, but now we're not too sure. Just what is the correct word for three times per year, or, equivalently, every fourth month? Help!!!

There is a lot of work involved in publishing the newsletter, but, very, very, fortunately, it is definitely NOT a thankless job. The many phone calls and letters of commendation we keep getting do make it all seem worth the effort. How could we even consider quitting, when so many of you tell us, in effect, "Keep up the good work!" We appreciate such "carrots", and only twice in three years have we received what we considered to be unfair criticism. Thus, it's far more ego-gratifying to continue than to stop.

We wish to thank all of you who have sent in disks, cassettes, listings, Xeroxed reference materials, notes for publication, useful components, samples, etc. It is our firm intention, each time we sit down to open our mail, to send, immediately, a thank-you card or note, to inform the sender that the material did arrive safely, and was much appreciated. We get so entranced in going over the materials, transcribing the cassettes to diskettes, and in reading all the materials, either on-screen, or hard-copy, that the time zips by, and we're by then much too tired to do the polite thing. So please accept our apologies and thanks in this form, for now. Things will be different in the future!

Now that we are going thriceannually, we will be able to get better organized. We will ask Jean, who pre-screens our mail, and answers immediately whatever requests for help she can, to prepare a "pre-addressed" card on which we can express our thanks for the material received, immediately on opening the package.

And now for a personal note: Thanks to those who expressed their concern over the Intraocular Lens (IOL) Implant surgery, which was quite succesful. Although the operation is still considered "experimental" in the USA, and my surgeon has never before implanted two IOLs in the same

patient less than six months apart, he will implant another IOL in the other eye right after this issue goes to press (19 November).

The hospital stay is only about two days, and during our Thanksgiving holiday. Will be able to work with both eyes within a week, and with no more newsletters due out for nearly a whole semester, should be able to get caught up on correspondence and all of the interesting, long-deferred projects.

Vision in the "bionic" eye is now 20/20 (can't remember it ever being better than 20/40, corrected, before). Binocular vision is now near spectacular, and the sky looks ever so much bluer. The only problem with the new lens that I have found, so far, is at night, when the pupil is wide open. Then small bright lights tend to create diffraction patterns due, I think, to the two small "struts" which support the lens in place.

My natural lens was very foggy, and quite yellow. The new one is diamond-clear, and colorless. As a consequence, the new eye is about one "stop" brighter, and has a color temperature differential of 4800K (daylight blue photoflood) to 3200K (indoor photolamp). Made the measurements with a collection of neutral density and color balance filters left over from my photograph engineering days, using my two eyes as a comparison "flicker" photometer and colorimeter. Just can't help making these quantitative comparisons!

When I noticed that an old argon lamp I had lying around looked much more vividly blue and very much brighter with my new eye, I examined the spectrum of the lamp and found that the plastic lens had extended my color vision from its previous cutoff point in the blue-violet region way down into the near UV region. I like my new eye, and want a matched pair. When the state of the art of IOLs improves, I'd like one wide-angle and one telephoto lens, please!

INTRODUCING THE SYM-2

First the "pluses":

1) It is smaller than SYM-1, measuring 8.8" x 7.8".
2) It comes with a plug-in transformer which provides 10 VAC at 16 VA.
3) It has a row of 8 LEDs and 8 DIP switches for "experiments".
4) It has a pair of RCA phono jacks for easier cassette I/O connections.
5) Jumpers are provided for easier cassette interface modifications.

Now some "minuses":

1) Contains only 1 K RAM space; piggy-backing to 2 K worthless.
2) Valuable space taken up by filter caps and large voltage regulator.
3) Only one VIA (no AA connector).
4) No edge-fingers, instead holes for installing 44-contact sockets.
5) Only one 24-pin socket for ROM/EPROM expansion.
6) Priced slightly higher than SYM-1.

General comments:

The SYM-2 is a "different" kind of SYM-1, with much less on-board expansion capability, but is "ready-to-use", right out of the box, with no scrounging around or added cost for power supply, or LEDs or switches for I/O control experiments. This makes it particularly attractive for classroom use.

Synertek is planning an extensive advertising campaign, aimed at the

large educational market which is certainly out there, if properly exploited. The market is not computer users, but rather computer system design students. We have seen proofs of some of the ads, and think they are very well done. We are especially pleased to see that the advertising copy includes the phrase "A subsidiary of Honeywell", since the Honeywell name provides a larger degree of customer recognition.

We have had an evaluation SYM-2 for many months now, with the revised SUPERMON (SYM 2.0) in a 2532 EPROM. At that time we received only a preliminary copy of the manual, with no Reference Card, and no listing of 2.0. We hope to get the additional documentation very soon, so that we can help SYM-2 owners with their problems and/or questions, also.

If you look closely at the masthead of this issue, you will notice a slight name change; we dropped the -1 from our name, and are now the "SYM Users' Group". We intend to support the -2, as well as the -1, since we are obviously deeply into education, ourselves.

MORE ON "RADAR" AND A CQ FOR HAMS

Here are some extracts from a recent letter from Ian Dilworth about the program "RADAR", which appeared in Issue 12. We were curious about where the data came from, and how to get more data for ourselves. Ian provides several interesting suggestions in his letter, some of which we would like to try, particularly the Speak & Spell one. He is also doing more with his Visible Memory than anyone else we know of!

Dear Lux:

The data supplied with RADAR was just test stuff. We actually have radar inputing data via a VIA and A/D converter. One nice use would be to have an A/D converter on a VIA and connect this to the AGC line on a receiver or spectrum analyser and sweep the local oscillator frequency in synchronism with the Visible Memory axis -- you have then made a cheap spectrum analyser with 3-D display and storage screen. I could supply megabytes of data but I don't really think it would be worthwhile.

Changing the aspect ratio of RADAR and the hidden line is OK, but the 1 MHz 6502 is too slow to do it in real time (unfortunately). Try a microphone into a VIA to get a data bank and use RADAR.

Instead of a microphone, you could use T I's Speak & Spell and the phoneme (very good, by the way!) package, triggered off the S & S, and look at spectra vs time of utterances . Great for seeing and hearing phoneme effects. Also pulse rate monitor and storage screen on Vis Mem very easy to do even without an A/D converter. There are 64 bytes in one horizontal scan in "RADAR".

I was interested to read recently of an Apple II based light pen that can draw (in high definitions) in real time -- apparently the screen is scanned at 60 Hz!! I'd really like to know the algorithm for doing that -- with 255 x 255 pixels to select from!

I'm using a joystick to draw on the Vis Mem at present. Also, I have a VIA pin connected to the video modulator (via series R) to give me z-axis modulation, i.e., a grey scale of 8.

Can you put a request in the next issue please? I'd like to get in touch with any radio hams who use the SYM and particulary has anyone got a Morse code and RTTY program going? I'd rather not reinvent the wheel unless necessary. My call is G3WRT (W3 until December). Also how about slow-scan TV (SSTV) using the Vis Mem?

I am working at COMSAT for 3 months during my sabbatical. I may introduce a SYM or two here. But until I get home I probably won't do much with the system which I have actually brought with me.

Regards,

Ian Dilworth
COMSAT Labs
2230 Comsat Drive
Clarksburg, MD 20871

Ian's permanent address is: Dr. I. J. Dilworth, Department of Electrical Engineering Science, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, England.

We should note that it is not essential to have a Visible Memory installed to do the RADAR type printouts on a printer with point-graphics capability; the Vis Mem just saves you time and paper by showing you the image before printing.

All that is required is a 4K RAM block to store a block of 64 "Y-slices" for 64 "X-values" of an eight-bit variable "Z", which is computed (or sampled) as a function Z(X,Y). The points to be printed are then stored, temporarily, in a less than 8K RAM block (40 x 200 bytes), for the 320 x 200 pixel image.

This type of graphics data processing is a natural for FORTH, since the data can easily be handled in its 16 bit integer format and the fixed-point arithmetic is inherently much faster than software-implemented floating-point arithmetic (note that the Apple's high speed graphics are usually handled in the Integer, rather than the Applesoft BASIC).

Unfortunately, most "programmers" these days are not familiar with techniques for "scaling" away the decimal points to permit the use of fixed-point arithmetic trigonometric packages, for example. While floating-point packages can be added to FORTH, most FORTH programmers prefer to use the much faster double precision (32 bit integer) arithmetic instead. Jack Brown sent us some very dynamic FORTH programmed Vis Mem graphics which would have been much less impressive when run at BASIC speed.

Incidentally, the word to SAVE the 8-bit Z value computed for any pair of X,Y values into a block of RAM starting at CONSTANT ORIGIN would be, simply:

```
: SAVE   Z @ ORIGIN X @ 64 Y @ * + + C! ;
( @ meams fetch value of; C! means store only single-byte)
```

The rest of the program would involve writing a defining word, ZCOMPUTE, for the value of VARIABLE Z in terms of VARIABLE X and VARIABLE Y, and using a pair of nested 0 64 DO ..... LOOP structures to do the work of filling in the 4K array of DATA.

EPSON RIBBONS AND ROLLERS

We have been using WD-40 to "rejuvenate" our Epson printer ribbons, as reported earlier, to keep the costs down. It works very well; we use it two or three times on each ribbon. We found only one problem, and a fix for same. If you are using the FT model, the one with the friction drive, and don't allow the ribbon enough time to dry properly, the ink strikes through the paper and onto the rubber platen roller, causing it to "gum" up. The tackiness causes paper misalignment when using tractor feed paper. The cure? Clean the roller with alcohol, and apply talcum powder to its surface until any tendency to "grab" the paper is gone.

"DUALIZING" THE KTM-2

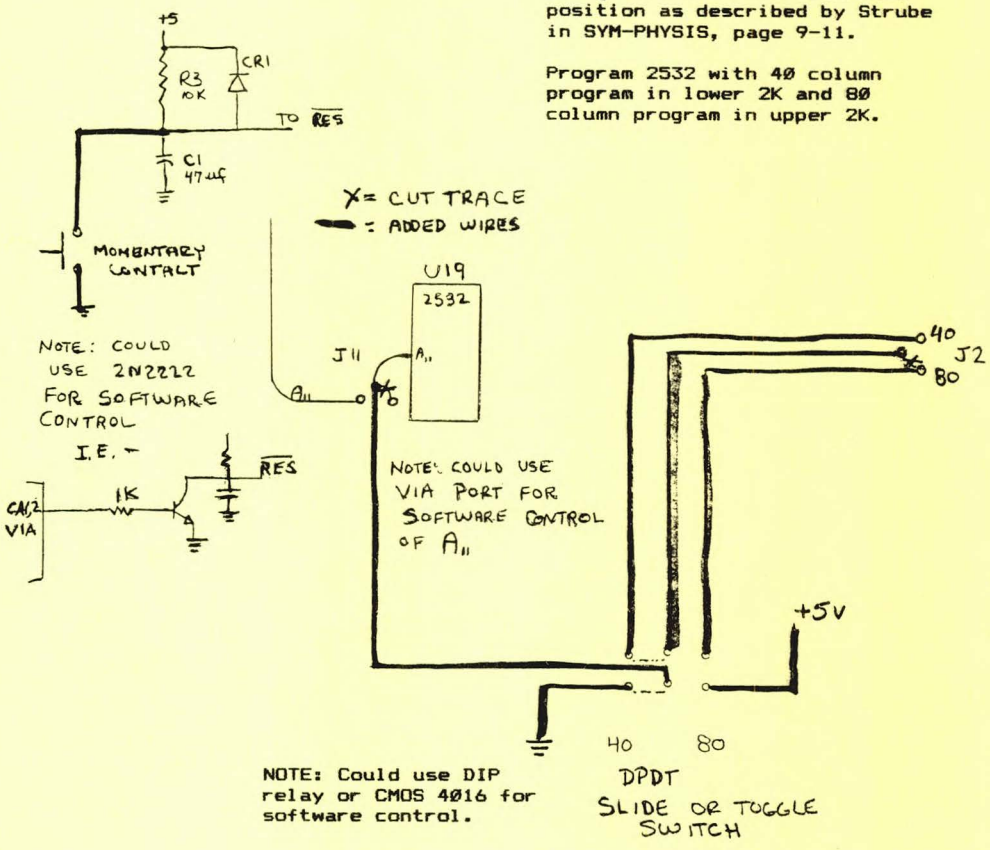
The KTM-2 (40 columns) can be used with an RF modulator and any TV set, and its character aspect ratio makes for nicer looking graphics than those provided by the KTM-2/80. The -2/80 is much better for word processing, however, and many purchasers of the -2 have upgraded to the -2/80, by replacing the ROM and adding the necessary support chips. We have done so with one of our -2 versions and reconvert by depowering, exchanging ROMs, and switching the jumper between -40 and -80 manually.

The sketch and the accompanying notes, below, by Steve Starre, Enfield, Connecticut, show how this may be done without depowering, and even under software control, if desired. Those owners of -80s who wish to follow Steve's example may order the KTM-2 ROM (02-0016B) from the Users's Group to copy into a 2532 as he describes (or just replace it manually, as we have been doing).

KTM-2 RESET needed to restart 6502
when ROM is switched (sometimes).

Change J18 & J19 to alternate position as described by Strube in SYM-PHYSIS, page 9-11.

Program 2532 with 40 column program in lower 2K and 80 column program in upper 2K.



NOTE: Could use DIP relay or CMOS 4016 for software control.

> > > NEW SWP-2.5 < < <

600 6TH AVENUE WEST
OWEN SOUND, ONTARIO
N4K 5E7 OCT 18 1982

SYM-1 USER'S GROUP
P.O. BOX 319
CHICO, CA 95927

Dear Jean & Lux:-

This is the latest, and, I hope, last edition of my rearrangement of SWP-2. I discovered some bugs in the last tape I sent you, and these have now been eliminated. Additionally, some extra goodies have been added. The new features of SWP-2.5 (from SWP-2) are as follows:

(1) Using command (period) I R will indent all lines in the paragraph AFTER the first line by R spaces. This could be done before by using the S command in the following lines, but then the first

line was not right justified. Anyway, this way is much easier.

(2) After setting the R parameter as in (1) above, the command I without parameters will continue doing the same thing.

(3) The command (period)X will indent ALL lines in the paragraph, including the first, by R spaces. This is used for additional paragraphs under the same heading.

(4) Page numbering is now justified over the right columns of text, with the number only being printed. I have tested it and it works up to page 9999, which should be enough for the average literary effort. Also, the page number for Page 1 is suppressed.

(5) If the final work will be in book form (or Xeroxed on both sides of the page) then use TB instead of T#. This will justify the odd page numbers over the right columns of text and the even numbers over the left columns, just like in a book.

(6) The FOOT command now works as before, except that if you use FOOT# with no other parameter, the page numbers will be centered at the bottom of the page, as, for example, in the CODOS manual.

(7) Formerly, if the command P appeared at the top of a new page, additional blank lines would be output. This is eliminated in this version for P. I and X.

The I R, I and X commands are used where P was used formerly. To make them work, the line immediately preceeding the FIRST I R must be (period)L 2. Then, the line immediately following any I R or I command must contain exactly R characters, of which at least the last MUST be an up arrow. Up arrows may also be used between characters as required.

For example, in the indented paragraph above, I set R at 6 and used (7) followed by three up arrows in the line following the I command.

This program has been changed in so many places (from SWP-2) that it is almost a new program. I have left in your FODS linkages, even though I don't have FODS, but this is easily changed for any system.

The cassette is recorded at double speed. There are two copies of the object code (L2 Ø1) which occupies from $200 to $ØACA, and two copies of the source code (GE F1) which occupies from $ØBØØ to $62AB. For convenience, there are two CT's, at lines 2229 and 4489.

As I said earlier, this now does everything I want, so I don't expect to alter it any further. I hope you try it - I think you'll like it.

Best wishes.

/s/ A. M. Mackay.

P.S. This letter is in RAE format on the tape, after the source code - GE F2.


```
>PR
ØØ1Ø  .M Ø 73 24 1
ØØ2Ø  .NOFILL
ØØ3Ø  .S 53
ØØ4Ø  .TB > > > NEW SWP-2.5 < < <
ØØ5Ø  .FOOT#
ØØ6Ø  6ØØ 6TH AVENUE WEST
ØØ7Ø  .S 53
ØØ8Ø  OWEN SOUND, ONTARIO
ØØ9Ø  .S 53
Ø1ØØ  N4K 5E7  OCT 18 1982
Ø11Ø  SYM-1 USER'S GROUP
Ø12Ø  P.O. BOX 319
Ø13Ø  CHICO, CA 95927
Ø14Ø  .L3
Ø15Ø  Dear Jean & Lux:-
Ø16Ø  .JU
Ø17Ø  .L2
Ø18Ø  .P
Ø19Ø  This is the latest, and, I hope, last edition of my
Ø2ØØ  rearrangement of SWP-2. I discovered some bugs in the
Ø21Ø  last tape I sent you, and these have now been eliminated.
Ø22Ø  Additionally, some extra goodies have been added.
Ø23Ø  The new features of SWP-2.5 (from SWP-2) are as follows:
Ø24Ø  .L2
Ø25Ø  .I 6
Ø26Ø  (1)^^^
Ø27Ø  Using command (period) I R will indent all lines in the
Ø28Ø  paragraph AFTER the first line by R spaces. This could be
Ø29Ø  done before by using the S command in the following lines,
Ø3ØØ  but then the first line was not right justified. Anyway,
Ø31Ø  this way is much easier.
Ø32Ø  .I
```

SWP 2.5
--- ---

The previous two pages contain an abridged copy of a letter received, on cassette, from Sandy Mackay, describing extensions he has added to SWP-2. We appended a portion of his text file to illustrate how the SWP editing commands are inserted into the manuscript, as and where required.

We have been using SWP-1 for a long time now, and have been slowly modifying it into a SWP-2. We have various modified versions on our master disk for special purposes, one of which is called %PUB (for PUBlish), for editing SYM-PHYSIS. We sent Sandy a copy of %PUB, calling it SWP-2, and he has added quite a few enhancements.

We read in his source code, changed the .BA to coreside with %PUB, reinserted the FODS linkage, and added SWP 2.5 to our master disk as %MAK (for MAcKay). We'll use it for the rest of this issue, since it is upward compatible with SWP-1.

During the next several months we'll give SWP-2.5 a good workout, and arrange to provide purchasers of SWP-1 an upgrade cassette to SWP-2.5 at a reasonable price.

HARDWARE RECOMMENDATION - REAL TIME (HARDWARE) CLOCK
-------- -------------- ---- ---- -------- -----

Jeff Lavin, of Alternative Energy Products, sent us the first prototype of his Real Time Clock card for testing. We tried it, returned it to him with one or two software suggestions, and placed an order with him for several of them. The clock card is designed to be mounted on the AEP-2 I/O board, which was described in an earlier issue.

[The AEP-2 I/O Board installs into the VIA #2 socket, and provides for up to four additional VIAs for the SYM-1. We plan to use our AEP-2s as follows: The Epson on the AA-connector, and the Hardware Clock, Speak & Spell, EPROM Burner, and ACIA Interface on flat 20 wire cables to the AEP-2. No more depowering and exchanging cables for us! (The EPROM burner and ACIA Interface are forthcoming AEP products which are in the development and early prototype stages at this writing, and will not be formally announced and available till early Spring 1983.)]

The Clock Card has provision for battery backup (NOTE: Batteries not supplied; must be user furnished and mounted to the board with tape, glue, double-sided sticky-stuff, Velcro, rubberbands. chewing gum, or whatever), and the clock may be removed and reinstalled without disturbing the set time. The Clock Card will be available through the Users' Group.

Here is a portion of the software provided to set and read the clock, to give you some idea of how it works. The software could be placed in EPROM, if desired, or could be downloaded from mass storage to RAM as needed.

```
0010  ; CLOCK/CALENDAR DRIVER PROGRAM
0020  ; for OKI MSM5832 MICROPROCESSOR
0030  ;    Real-Time Clock/Calendar
0040
0050  ;       Copyright  1982
0060  ; ALTERNATIVE ENERGY PRODUCTS
0070
0080  ; The Registers are:
0090
0100  ; REGISTERS:     1M 10M  1H 10H  W   D1 D10  M1 M10  Y1 Y10
0110
0120  ; EXAMPLE:      $6X $2X $9X $8X $2X $5X $0X $0X $1X $2X $8X
0130
```



```
0140  ; This example would print out as:
0150
0160  ;         "09:26:00 TUESDAY OCT 05 1982"
0170
0180  ; The program inserts the century "19"; may be changed
0190  ; when the 21st century arrives.
0200
0210  ; (Bit 7 of 10H is set (high) for 24 hour format; this bit
0220  ; is inserted by the program below)
0230
0240  ; Only the most significant nibbles above are meaningful,
0250  ; since this ia a four bit wide micro.  Note that a total
0260  ; of 16 registers can be accessed since the address bus is
0270  ; also four bits wide.  Only 11 registers are shown above.
0280  ; Two other registers, 1S and 10S are "read only", since
0290  ; they can only be written to as $0.  Two other "possible"
0300  ; registers are not implemented, and the final register,
0310  ; not used here, is also read only.  It generates a 1024
0320  ; Hz square wave on one of the data lines and pulses each
0330  ; second, minute, and hour on the other three data lines.
0340
0350  ; The clock/calendar is driven by a VIA
0360  ; The port assignments are as follows:
0370
0380  ;      PORT  A                PORT B
0390  ; 7 6 5 4 3 2 1 0      7 6 5 4 3 2 1 0
0400  ;-----------------------------------
0410  ; D D D D A A A A            T A R W H
0420  ; 3 2 1 0 3 2 1 0            E D E R O
0430  ;                            S J A I L
0440  ;                            T   D T D
0450  ;                                E
0460
0470
0480  ; N.B. The actual program has been omitted here.
0490  ;      Only the tables for "SET" prompting
0500  ;      and "READ" formatting are reproduced here.
0510
0520  SET.PRMPT   .BY 'Set time:' $D $A $A
0530  SET.MSG     .BY 'Y10' $A0 'Y01' $A0 'M10' $A0 'M01' $A0
0540              .BY 'D10' $A0 'D01' $A0 'DAY' $A0
0550              .BY '10H' $A0 '01H' $A0 '10M' $A0 '01M' $A0
0560              .BY $FF
0570
0580  DAY.TABL    .BY 'SUNDAY' $80 $80
0590              .BY 'MONDAY' $80 $80
0600              .BY 'TUESDAY' $80
0610              .BY 'WED' $27 'SDA' $D9
0620              .BY 'THURSDA' $D9
0630              .BY 'FRIDAY' $80 $80
0640              .BY 'SATURDA' $D9
0650
0660  MONTH.TABL  .BY 0 0 0 $80
0670              .BY 'JAN' $A0
0680              .BY 'FEB' $A0
0690              .BY 'MAR' $A0
0700              .BY 'APR' $A0
0710              .BY 'MAY' $A0
0720              .BY 'JUN' $A0
0730              .BY 'JUL' $A0
0740              .BY 'AUG' $A0
0750              .BY 'SEP' $A0
0760              .BY 'OCT' $A0
0770              .BY 'NOV' $A0
0780              .BY 'DEC' $A0
                  .EN
```

HARDWARE RECOMMENDATION - SYM/KTM ENCLOSURE

We have installed one of our SYM/KTM systems in a very elegant case made by KEN-WAY PRODUCTS, 831 Patton Road, New Brighton, Minnesota 55112.

To quote from the descriptive brochure: "The (aluminum) enclosure features a low profile design with durable textured baked charcoal finish that matches the KTM keys. Solid birch side panels are walnut stained. The SYM is mounted in the hinged top panel which also provides direct access to the SYM keypad."

Our system includes the SYM-1, a KTM-2, a 32K Beta DRAM Board, an FDC-1 Disk Controller, and an HDE FODS Disk Controller. There is still lots o space left over, into which we plan to build a compact 4 A power supply, using the case as the heat sink for the regulator (no fan for us!).

We power up on this system to the FODS DOS, then download the FDC-1 operating system into RAM at $9000. This is the system on which we will be evaluating and debugging any new DOSes developed for the FDC-1. We have two pairs of BASF 5 1/4 " drives on this system, one dual system for FODS, one dual system for FDC-1. We even have an extra cable coming off the FDC-1 controller card for a pair of 8" drives, for testing the software with 8" systems. We have only one pair of 8" drives around and these are installed on our MTU CODOS system, but can be switched over for testing.

We like the case very much and highly recommend it as a good value. Contact Ken Schaufler (KEN-WAY PRODUCTS), (612) 633-3035 for prices and any additional information.

Ken sent us a copy of one of Sylvia Porter's newspaper columns which pointed out that, this year only (1982), business equipment expenses up to $5000 may be written-off in full, rather than being depreciated! So, buy it this year, if you can manage it.

FORCED TAPE READ

S. G. Knox (we think that's who it was!) sent us this little program he got from Bob Peck to force a cassette read. Might be worth trying if you are having difficulty reading a cassette.

```
                0010 ; ******************************************
                0020 ; *                                        *
                0030 ; *    BOB PECK'S SYM MON 1.1 FORCED       *
                0040 ; *      CASSETTE TAPE READ ROUTINE        *
                0050 ; *              2 MAY 1982                *
                0060 ; *                                        *
                0070 ; ******************************************
                0080
                0090 ; .OS
                0100            .BA $0010     ;OR WHEREVER DESIRED
                0110
                0120 PLACE      .DE $0200     ;OR WHEREVER
                0130
                0140
0010- 20 86 8B  0150 START      JSR $8B86     ;ACCESS
0013- A9 02     0160            LDA #H,PLACE
0015- 8D 4D A6  0170            STA $A64D     ;P2H
0018- A9 00     0180            LDA #L,PLACE
001A- 8D 4C A6  0190            STA $A64C     ;P2L
001D- A0 80     0200            LDY #$80      ;MODE
001F- 20 A9 8D  0210            JSR $8DA9     ;START TAPE ROUTINE
0022- 20 52 8D  0220            JSR $8D52     ;READ HIGH SPEED BYTE - SYNC FIND
0025- 20 E5 8D  0230 LOOP       JSR $8DE5     ;READ A BYTE
0028- A0 00     0240            LDY #$00
```



```
002A- 91 FE     0250            STA ($FE),Y   ;PUT IT AWAY
002C- E6 FE     0260            INC **$FE     ;BUMP THE POINTER
002E- E6 FF     0270            INC **$FF
0030- 20 3C 8B  0280 INCDUN     JSR $8B3C     ;TSTAT - STOP IF KEY DOWN
0033- 90 F0     0290            BCC LOOP
0035- 00        0300            BRK
0036- EA        0310            NOP
                0320
                0330
                0340            .EN
```

C/O School of Optometry
University of NSW
P.O. Box 1
Kensington
NSW 2033
Australia

Dr. H. R. Luxenberg
SYM Users' Group
P.O. Box 319
Chico, CA 95927
USA

Dear Dr. Luxenberg:

I have been meaning to write to you for some time now, ever since first reading a copy of SYM-PHYSIS. I have not seen any copies dated later than 1980, and so I hope your excellent newsletter is still alive and well. I am writing for two reasons:

1. Is the Users' Group still active? If so I would like to join. Please send me all the details.

2. I thought you may like some details of my system. It was put together in a hurry (are'nt they all?) for data acquisition in the field of neurophysiology. The system supports a 16 input A/D and 2 output D/A, a BCD event counter and external switch register input. There is also an extra 4K RAM. All extras are built into standard 'Radio Shack' 44 pin proto boards. The SYM, memory, extras and KTM-2 all fit into an aluminium box a little bigger than an Apple. I also have an Apple 2+. This is not treason. The Apple communicates with the SYM via the VIAs (if you see what I mean). At present I am not using the full capacity of the SYM. However it is an indispensible part of a piece of apparatus providing timing signals to control stimuli for experiments in vision physiology.

This is not an original idea. The SYM program enclosed is based on a similar program written for a KIM by the Vision Research Labs at the NIH in Washington D.C. The program was whipped up in an afternoon (testimony to the quality of RAE-1). It works, but I'm sure you could find ways of improving the software.

NOTES ON SYM TIMER PROGRAM

The program enables four lines (Port A of 6522#1) to be used as outputs for pulses of precisely controlled length. The states of the lines and the times at which they change are determined by a table located at $30. When triggered, the program loads a 6522 timer with data for 1mS, and enables IRQ interrupts to IRQINT. At IRQINT, a 16 bit count location is decremented. If zeroed, then the count location is updated from the next two bytes of the table (ready for next time interval) and the third byte is output to Port A of 6522 #1. In this way the program makes its way down the table until either $FF or $FE are encountered. If $FE, then pulse train repeats indefinitely. If $FF then program disables IRQ and waits for another 'trig' pulse before

[ED NOTE: line(s) missing from manuscript here]

The enclosed version is not as general in application as was the original. It includes a facility where one line (bit 1) is only enabled during the first pulse train following a DELY pulse (that is what BGFLG is for).

Also, the annotation is a bit skimpy due to my having only 8K for the RAE - 1 files.

The important I/O bits are:

PORT A 6522#1
 Outputs
  Bit 0 - Trig output (e.g. to CRO)
  Bit 1 - Line 1 output (e.g. conditioning stimulus)
  Bit 2 - Line 2 output (e.g. test stimulus)
  Bit 3 - SYM BUSY output (handshake to Apple)
 Inputs
  Bit 6 - Change sixth byte in table to next byte in DLIST
  Bit 7 - Trig in (initialise a pulse sequence)

The function served by bit 6 is peculiar to the type of experiment in which I am presently engaged. This line, when pulsed low, causes the location DELY to be replaced with the next element in a list of delay values at DLIST.

You may guess that the equipment is controlled by an Apple program which calls up the various responses by pulsing bits 7 and 6 low when required.

I am enclosing a circuit diagram of the System as it now stands. I have not written any software for it, though I have written a waveform averaging progam for an Apple with the appropriate hardware. Members of the group are welcome to copies of this. I'm sure it could be adapted easily to any SYM system.

I hope this is of some use to you. If you want more information on my 6502 activities, please write. However I'm only an amateur, and I'm sure I would learn a lot more from the group than I could put into it.

Regards,

/s/ Philip J. Anderton

Here is Mr. Anderton's TIMER program, which we reprint, slightly edited, without having had the time to test it. Following the program we reproduce several of Mr. Anderton's sketches to show the very effective use he has made of the SYM's VIAs.

```
0010 ;PROGRAM TO USE SYM AS TIMER
0020
0030            .BA $1800 or wherever
0040 ; .OS
0050
0060 ;6522 ADDRESS DEFINITIONS
0070
0080 ;6522#1
0090
0100 PRTA1      .DE $A001
0110 DDRA1      .DE $A003
0120 T11LO      .DE $A004
0130 T11HI      .DE $A005
0140 ACR1       .DE $A00B
0150 IFR1       .DE $A00D
0160 IER1       .DE $A00E
```



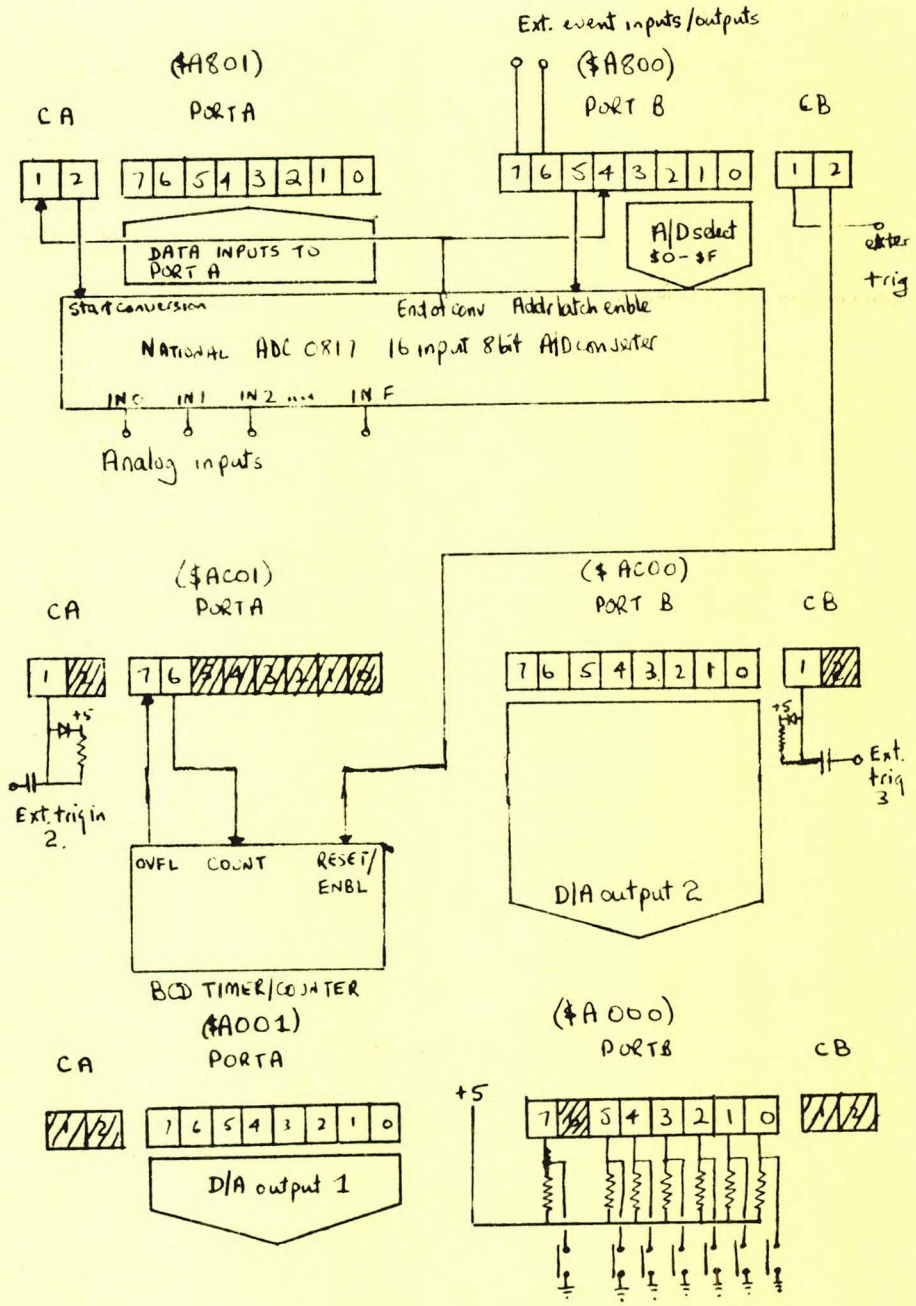
```
0170
0180 ;6522#2
0190
0200 PCR2       .DE $A80C
0210
0220 ;6522#3
0230
0240 PRTA3      .DE $AC01
0250 DDRA3      .DE $AC03
0260
0270 ;INITIALIZATION DATA
0280
0290 ENBT1      .DE $C0
0300 DIST1      .DE $40
0310 BUSY       .DE $08 bit 3
0320
0330 ;MONITOR ROUTINES
0340
0350 USRENT     .DE $8035
0360 TSTAT      .DE $8B3C
0370 OUTCHR     .DE $8A47
0380 UIRQVC     .DE $A678
0390 SAVER      .DE $8188
0400 RESALL     .DE $81C4
0410 GETKEY     .DE $88AF
0420 OUTBYT     .DE $82FA
0430 CRLF       .DE $834D
0440 INTCHR     .DE $8A58
0450 SCR0       .DE $A630
0460 ACCESS     .DE $8B86
0470 OBCRLF     .DE $834A
0480
0490 ;PROGRAM LOCATIONS
0500
0510 POINTR     .DE $2E 2 bytes
0520 TABLE      .DE $30 <256 bytes
0530 TCOUNT     .DE $2C 2 bytes
0540 DELY       .DE TABLE+6
0550 DELPTR     .DE $2A
0560 ENDFL      .DE $29
0570 BGFLG      .DE 28 mask to disable bit 1
0580
0590 ;PROGRAM STARTS HERE
0600
1800- 20 86 8B  0610 MAIN       JSR ACCESS
1803- 20 7F 18  0620            JSR INIT
1806- 20 0F 19  0630 MAIN1      JSR BUSYLO
1809- AD 01 A0  0640            LDA PRTA1 test for pulse inputs
180C- 09 0F     0650            ORA #$0F
180E- 2A        0660            ROL A
180F- 90 3B     0670            BCC TRIG tigger input low?
1811- 2A        0680            ROL A
1812- 90 07     0690            BCC NDLY new delay input low?
1814- 20 3C 8B  0700            JSR TSTAT key pressed?
1817- 90 ED     0710            BCC MAIN1
1819- 78        0720            SEI
181A- 60        0730            RTS
181B- 20 75 18  0740 NDLY       JSR WAIT
181E- 20 06 19  0750            JSR BUSYHI set bit 3 hi
1821- A9 FF     0760            LDA #$FF
1823- 85 1C     0770            STA *BGFLG enable one pulse only for bit 1
1825- A0 00     0780            LDY #$00 and update dely value
1827- E6 2A     0790            INC *DELPTR
1829- B1 2A     0800            LDA (DELPTR),Y
182B- C9 FF     0810            CMP #$FF
```

```
182D- D0 03     0820            BNE ND1
182F- 20 C3 18  0830            JSR INITPRD
1832- B1 2A     0840 ND1        LDA (DELPTR),Y
1834- 85 36     0850            STA *DELY
1836- 20 FA 82  0860            JSR OUTBYT
1839- E6 2A     0870            INC *DELPTR
183B- B1 2A     0880            LDA (DELPTR),Y
183D- 85 37     0890            STA *DELY+1
183F- 20 FA 82  0900            JSR OUTBYT print it for debugging
1842- 20 4D 83  0910            JSR CRLF
1845- 20 0F 19  0920            JSR BUSYLO
1848- D0 BC     0930            BNE MAIN1
184A- F0 BA     0940            BEQ MAIN1
184C- 20 75 18  0950 TRIG       JSR WAIT new pulse train
184F- A9 00     0960            LDA #$00
1851- 85 29     0970            STA *ENDFL
1853- 20 06 19  0980            JSR BUSYHI
1856- 20 FB 18  0990            JSR LDSTRT load and start clk
1859- A9 C0     1000            LDA #ENBT1
185B- 8D 0E A0  1010            STA IER1 enable
185E- 58        1020            CLI interrupts from clk
185F- A5 29     1030 TR1        LDA *ENDFL if either endflag or
1861- D0 05     1040            BNE TR2
1863- 20 3C 8B  1050            JSR TSTAT
1866- 90 F7     1060            BCC TR1 keypressed then stop
1868- A9 40     1070 TR2        LDA #DIST1
186A- 8D 0E A0  1080            STA IER1
186D- A9 00     1090            LDA #$00
186F- 8D 29 00  1100            STA ENDFL
1872- 78        1110            SEI
1873- F0 91     1120            BEQ MAIN1 ALWAYS
                1130
1875- AD 01 A0  1140 WAIT       LDA PRTA1 wait for all switched 2B cleared
1878- 09 0F     1150            ORA #$0F
187A- 49 FF     1160            EOR #$FF
187C- D0 F7     1170            BNE WAIT
187E- 60        1180            RTS
                1190
                1200 ;INIT ROUTINE
                1210
187F- 20 E5 18  1220 INIT       JSR LTBL initialise vectors and pointers
1882- 20 AF 18  1230            JSR INITPR
1885- A9 1A     1240            LDA #L,IRQINT
1887- 8D 78 A6  1250            STA UIRQVC
188A- A9 19     1260            LDA #H,IRQINT
188C- 8D 79 A6  1270            STA UIRQVC+1
188F- A9 0F     1280            LDA #$0F
1891- 8D 03 A0  1290            STA DDRA1
1894- A9 C0     1300            LDA #$C0
1896- 8D 0C A8  1310            STA PCR2 ENBL TIMER
1899- 8D 0B A0  1320            STA ACR1
189C- 8D 01 AC  1330            STA PRTA3
189F- 8D 03 AC  1340            STA DDRA3
18A2- A9 80     1350            LDA #$80
18A4- 8D 01 AC  1360            STA PRTA3
18A7- 20 CC 18  1370            JSR INITCNT
18AA- A9 FD     1380            LDA #$FD
18AC- 85 1C     1390            STA *BGFL6
18AE- 60        1400            RTS
                1410
18AF- 20 BA 18  1420 INITPR     JSR INITPRT initialise both pointers
18B2- 20 C3 18  1430            JSR INITPRD
18B5- A9 A6     1440            LDA #L,DLIST+9 pnt to 2nd last byte on initial
18B7- 85 2A     1450            STA *DELPTR
18B9- 60        1460            RTS
                1470
```



```
18BA- A9 30     1480 INITPRT    LDA #L,TABLE
18BC- 85 2E     1490            STA *POINTR
18BE- A9 00     1500            LDA #H,TABLE
18C0- 85 2F     1510            STA *POINTR+1
18C2- 60        1520            RTS
                1530
18C3- A9 9D     1540 INITPRD    LDA #L,DLIST
18C5- 85 2A     1550            STA *DELPTR
18C7- A9 19     1560            LDA #H,DLIST
18C9- 85 2B     1570            STA *DELPTR+1
18CB- 60        1580            RTS
                1590
18CC- A9 01     1600 INITCNT    LDA #$01 initialise 16 bit counter
18CE- 85 2C     1610            STA *TCOUNT
18D0- A9 00     1620            LDA #$00
18D2- 85 2D     1630            STA *TCOUNT+1
18D4- 60        1640            RTS
                1650
18D5- A9 80     1660 COUNT      LDA #$80 send a pulse to BCD counter
                1670 ;                  (port A 6522#3, bit 7)
18D7- 8D 01 AC  1680            STA PRTA3
18DA- A9 00     1690            LDA #$00
18DC- 8D 01 AC  1700            STA PRTA3
18DF- A9 80     1710            LDA #$80
18E1- 8D 01 AC  1720            STA PRTA3
18E4- 60        1730            RTS
                1740
18E5- 20 BA 18  1750 LTBL       JSR INITPRT load TABLE at zero page
18E8- A0 FF     1760            LDY #$FF
18EA- C8        1770 LT1        INY
18EB- B9 8A 19  1780            LDA DATA,Y
18EE- 91 2E     1790            STA (POINTR),Y
18F0- C9 FF     1800            CMP #$FF
18F2- F0 06     1810            BEQ LT2
18F4- C9 FE     1820            CMP #$FE
18F6- F0 02     1830            BEQ LT2
18F8- D0 F0     1840            BNE LT1
18FA- 60        1850 LT2        RTS
                1860
18FB- A9 E8     1870 LDSTRT     LDA #$E8 load and start clock
18FD- 8D 04 A0  1880            STA T1LO
1900- A9 03     1890            LDA #$03
1902- 8D 05 A0  1900            STA T1HI
1905- 60        1910            RTS
                1920
1906- AD 01 A0  1930 BUSYHI     LDA PRTA1 set bit 4 hi (anytime)
1909- 09 08     1940            ORA #BUSY
190B- 8D 01 A0  1950            STA PRTA1
190E- 60        1960            RTS
                1970
190F- A9 08     1980 BUSYLO     LDA #BUSY set bit 4 lo (anytime)
1911- 49 FF     1990            EOR #$FF (BUSYHI and BUSYLO can be corrupted
                2000 ;                  by interrupt routine)
1913- 2D 01 A0  2010            AND PRTA1
1916- 8D 01 A0  2020            STA PRTA1
1919- 60        2030            RTS
                2040
191A- 48        2050 IRQINT     PHA  ;interrupt routine does it all
191B- 98        2060            TYA
191C- 48        2070            PHA
191D- AD 0D A0  2080            LDA IFR1
1920- 8D 0D A0  2090            STA IFR1
1923- 8D 30 A6  2100            STA SCR0
1926- AD 0E A0  2110            LDA IER1
1929- 2D 30 A6  2120            AND SCR0
```

```
192C- D0 08     2130            BNE IQ1
192E- A9 00     2140            LDA #$00
1930- 8D 0B A0  2150            STA ACR1
1933- 20 35 80  2160            JSR USRENT if wrong source of int goes to mon
1936- F8        2170 IQ1        SED DECIMAL MODE
1937- 38        2180            SEC prepare to subtract
1938- A5 2C     2190            LDA *TCOUNT decrement 16 bit counter
193A- E9 01     2200            SBC #$01
193C- 85 2C     2210            STA *TCOUNT
193E- A5 2D     2220            LDA *TCOUNT+1
1940- E9 00     2230            SBC #$00
1942- 85 2D     2240            STA *TCOUNT+1
1944- A5 2C     2250            LDA *TCOUNT
1946- D0 3D     2260            BNE ENDINT end interrupt if not zero
1948- A5 2D     2270            LDA *TCOUNT+1
194A- D0 39     2280            BNE ENDINT "        "
194C- A0 00     2290            LDY #$00
194E- B1 2E     2300            LDA (POINTR),Y get next byte in table
1950- C9 FE     2310            CMP #$FE if $FE or $FF do appropriate things
1952- F0 12     2320            BEQ IQ2
1954- C9 FF     2330            CMP #$FF
1956- D0 1A     2340            BNE IQ3
1958- A9 01     2350            LDA #$01 if $FF then disable and end interrupts
195A- 8D 29 00  2360            STA ENDFL
195D- A9 40     2370            LDA #DIST1
195F- 8D 0E A0  2380            STA IER1
1962- A9 FD     2390            LDA #$FD
1964- 85 1C     2400            STA *BGFLG
1966- 20 D5 18  2410 IQ2        JSR COUNT if $FE count one and reinit pointers
                2420 ;                  without disable
1969- 20 BA 18  2430            JSR INITPRT
196C- 20 CC 18  2440            JSR INITCNT
196F- 4C 85 19  2450            JMP ENDINT
1972- 85 2D     2460 IQ3        STA *TCOUNT+1 if not $FF or $FE then must be
                2470 ;                          new time/port info
1974- E6 2E     2480            INC *POINTR load new timer
1976- B1 2E     2490            LDA (POINTR),Y and port data (masked by BGFLG)
1978- 85 2C     2500            STA *TCOUNT
197A- E6 2E     2510            INC *POINTR
197C- B1 2E     2520            LDA (POINTR),Y
197E- 25 1C     2530            AND *BGFLG
1980- 8D 01 A0  2540            STA PRTA1
1983- E6 2E     2550            INC *POINTR dont forget update TABLE pntr and
1985- D8        2560 ENDINT     CLD NORMAL exit interrupt
1986- 68        2570            PLA
1987- A8        2580            TAY
1988- 68        2590            PLA
1989- 40        2600            RTI
                2610
198A- 00 10 09  2620 DATA       .BY $00 $10 $09
198D- 01 00 08  2630 DELY1      .BY $01 $00 $08
1990- 01 00 0A  2640 DELY2      .BY $01 $00 $0A
1993- 01 00 0E  2650 TEST1      .BY $01 $00 $0E
1996- 01 00 0A  2660 TAIL1      .BY $01 $00 $0A
1999- 01 00 00  2670 TAIL2      .BY $01 $00 $00 $FF
199C- FF
199D- 01 00 02  2680 DLIST      .BY 01 00 02 00 03 00 04 00 05 00 $FF
19A0- 00 03 00
19A3- 04 00 05
19A6- 00 FF
19A8- FF FF FF  2690            .BY $FF $FF $FF $FF
19AB- FF
                2700
                2710            .EN
```

MR. PACMAN, MEET MR. SYMMAN!

We reproduce below extracts from a very recent letter sent us by Daniel Wuethrich (you may recognize his letterhead from a previous issue!). This is followed by an edited copy of the Instruction Manual he provided on FDC-1 format diskette and three Epson MX-80 printouts showing the appearance of the Visible Memory screen at various stages of the game. And, finally, some additional comments by us on SYMMAN . . . .

ibw

INGENIEURBÜRO WÜTHRICH BRUGG

Hardware Mikroprozessor-Software Prozesssteuerungen Prototyp-Entwicklungen Kleinserien

Dear Lux and Jean,

I like to send to You a computer game I made for my visible memory. It is similar to the well known arcade-game PACMAN. For copyright reason I call my game SYMMAN. If You like this game, I would be glad if You can sell it to other SYM-users. I let You select a resonable price. If You can sell the game please don't send any money to me, just keep it for my future orders.

Enclosed is a disk with the following files (in SYMDOS):
- T.SYMMAN = Manual for the game
- M.SYMMAN = Machine-Code of the game (Ready to start)
- S.SYMM.P1 = Source-Code Part 1
- S.SYMM.P2 = Source-Code Part 2

For the game You need a joystick with 4 switches for the 4 directions. The analog joystick (with variable resistors) You gave to me can not be used for this game. For Your information: I made the hard- and software for this analog joystick, but the precision was not good enough to draw nice figures on the screen.

I look forward to see You in Switzerland. Are You already planning Your Europe-trip ?

SYM-cerly Yours

Daniel

Daniel A. Wüthrich
Ing.büro Wüthrich

S Y M M A N M A N U A L

1. HARDWARE

You need a joystick with 4 switches for the directions UP, DOWN, LEFT and RIGHT plus an additional button for ACTION. You can buy such a joystick as a spare part from a computer game distributor (e.g., Commodore, Atari, etc.). With a little skill you can build your own.
Connect the joystick to any free 8-bit port as follows:

```
GROUND : all switches
bit 0  : UP-switch
bit 1  : DOWN-switch
bit 2  : ACTION-switch
bit 3  : -
bit 4  : -
bit 5  : -
bit 6  : LEFT-switch
bit 7  : RIGHT-switch
```

(text continued to page 13/14-67)

>RUN $4000

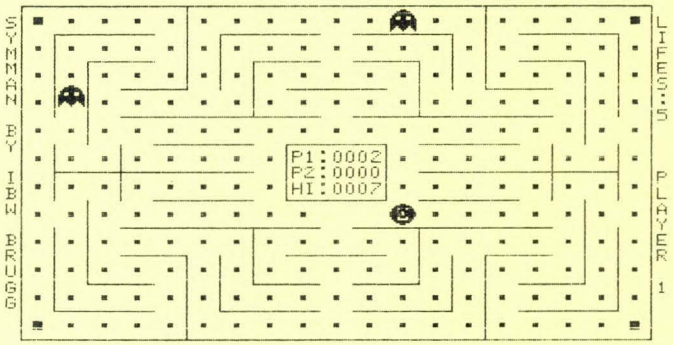


FIGURE 1: The game has just started, with Player 1. SYMman has 5 lives (forgive the mis-spelling) left. SYMman started at the center of the screen, just below the score display, and has gobbled up 2 of the dots, on his trip to the right. Since we had not added the joystick control, SYMman continued moving right, picking up 5 more dots, until he was trapped by the wall of the maze. The HI score of 7 was from previous "runs".

>RUN $4000

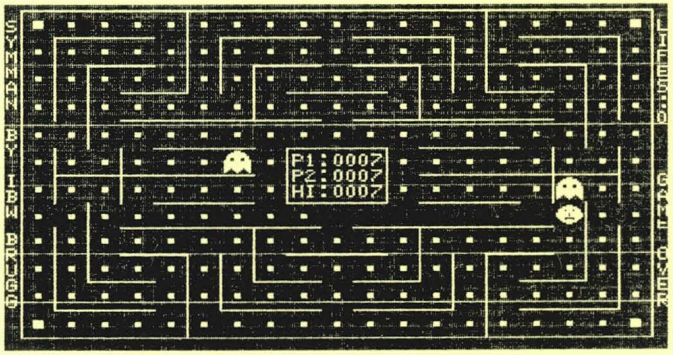
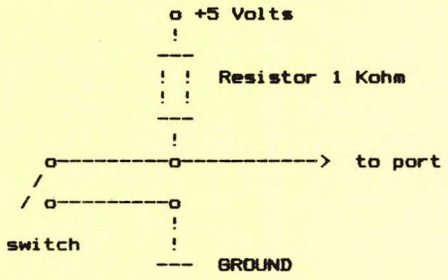


FIGURE 2: (Printed as a "negative" for variety) We watched, helplessly, as SYMman lost all five lives to the octopi, then watched the same sequence occur with Player 2. Here Player 2 has 0 lives left, and SYMman is just about to be devoured by the octopus just above him, ending the game.

(Yes, we will be adding a joystick, as soon as we can. We think we owe it to ourself, to play a game once in a while, and we do want to help SYMman rise above the measly 7 point score!!!)

TYPICAL OF ONE INPUT:



Memory needed: 8 K MTU Visible Memory
8 K to play the game
32 K to assemble

2. SOFTWARE

Load the programm M.SYMMAN (from $200-$1AFF). Set the following memory locations:

$203 : e.g., $20 Visible Memory origin (e.g., VM from $2000 to $3FFF)
$204 : e.g., $00 Low byte address of joystickport
$205 : e.g., $A8 High byte address of joystickport (e.g., $A800

Then type G 200 <CR>

3. THE GAME

Try to catch all dots in the maze by moving Your SYMMAN (smiling sun) with the joystick. Your enemies are the octopi (2 at the beginning up to 5). They follow you and try to catch you. Normally you have 5 lives, after that Player 2 can play. If you want to change the lives per game, then change memory location $208 (1...9). Catching one of the 4 large dots at the corners make the octopi black for a few seconds. When the octopi are black you can eat then and they are sent to the other end of the maze.

POINTS: small dot = 1 point
large dot = 5 points
black oct = 10 to 90 points depending on the number of octopi in the maze and on the number of octopi already eaten

After catching all dots the game starts automatically again with one octopus more (up to 5).

DISPLAY: In the middle of the maze you see the points of Players 1 and 2 and the high score. The high score is in memory

$206 and $207. If you want to save the high score after the game, then simply save the whole programm back to disk or cassette.
At the right border of the maze you see the number of lives and which Player has to play.
After Player 2 loses his last life GAME OVER is displayed.

RESTART: After GAME OVER press the ACTION button briefly to restart the game. If you press the ACTION button longer than 1 second a jump to SYM-MONITOR is executed.

>RUB $4000

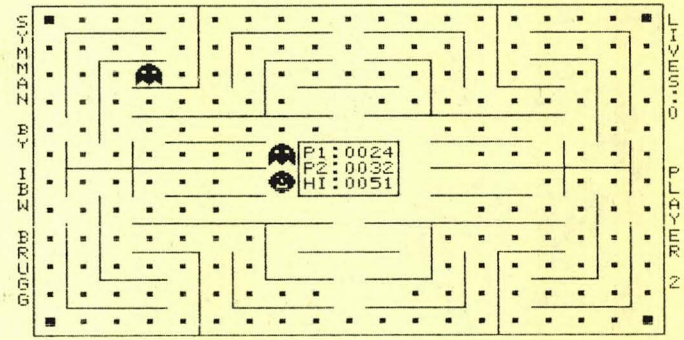


FIGURE 3: We gave SYMMAN a sporting chance by allotting him 9 lives and giving him the opportunity to take a "quasi-random walk", actually more nearly a drunken stagger. We did this by letting the program sample the free-running timer at $A804, rather than the nonexistent joystick at $A800. We inhibited the jump to SUPERMON after GAMEOVER so that the game would always restart, and left it running overnight. We stopped the game with RST just as Player 2 had lost his last life, and was about to be devoured, triggering GAMEOVER. We also corrected the spelling of "lives".

MORE ON SYMMAN

Dan Wuethrich was the second to send us material for review and publication on an FDC-1 diskette (Jeff Lavin was the first, but he also sent backup on cassette, since our FDC-1/FODS Dual-Dual Disk Drive System (F/F D-D DDS) was not yet ready). We read Dan's FDC-1 diskette and transcribed the material to a FODS diskette on our newly completed F/F D-D DDS.

We transfered the diskette to our main development system (the only one interfaced to the Epson for graphics printout; all others use the decwriter II on the 20 mA loop at 600 baud for printout, or, if the printer patch is not resident, we log-on with the decwriter as a TTY-type terminal at 110 baud whenever we need hard copy), so that we would, eventually, be able to make hard copy images of the Visible Memory display(s).

As of now, our only Visible Memory (which is an MTU product) is on another system at $2000-$3FFF, built into an MTU Card-File, with the MTU

CODOS (Channel Oriented Disk Operating System) resident at $4000-$7FFF, hence no FODS or FDC-1 available. Thus, we transfered material between the MTU/Vis Mem/CODOS system and our main development system over the cassette interface loop(s) so that we could see the dynamic interactive graphics on one, and print out the static "snapshot" type images on the other.

In the near future (see following article on SUPER-SYM) we hope to have another system going where we'll be able to interrupt a dynamic display during a transition period, so that the printout will show some "blur", thereby creating a feeling of motion in the image (there we go, thinking like a still photographer).

Incidentally, while SYMMAN is black/white only, and does not have the full audio capabilities of the arcade game which it resembles, the visual resolution is excellent, and Dan calls very effectively on JSR BEEP to provide very nice sound effects.

SUPER-SYM

Our SYMs are used rather heavily, by both ourselves and students, so we need to have a large number of them running on a multi-tasking basis. Each of our SYM's has its own personality and capabilities, sort of like siblings in a large family.

For many years we lived comfortably with SYM's built-in limitation of a maximum of 32 K contiguous RAM with a spare utility 4 K at $9000-$9FFF. We didn't like the fact that the 16 K RAM requirement of CODOS forced us to locate our 8 K Visible Memory at $2000-$3FFF, right in the middle of our SYM-FORTH, however, and we were looking for a way out of this dilemma.

It wasn't until we started disassembling and reassembling our newest FORTH (see elsewhere in this issue) that we really felt the 32 K limitation. We didn't mind the .CT assembly so much; what did bother us was trying to use KWOK's Cross Referencer to get the "oh, so elegant!" Label File Listings it provides. Now FORTH source code, by virtue of its threaded nature, is essentially a listing of label addresses, and many of the labels are called dozens, or even scores, of times. While each of the two source files could be cross-referenced individually, there was no way to get a complete cross reference label file for both source files into the contiguous 32 K!

We had bought for our own use the complete prototype run, some five or six boards, of Jeff Lavin's AEP-1 32 K RAM boards (the final production boards have additional jumper capabilities not present on the prototypes). We liked the capability of being able to interchange 2 K RAMs and 2 K EPROMs (2716s) so freely, anywhere on the board. We also had a spare Visible Memory, and of course a SYM. We ordered another MTU Card Cage, and shipped the whole collection of boards and stuff to Jeff Lavin, telling him, in a rather vague way, that we needed more contiguous RAM, and the Visible Memory as far up in the memory address space as he could get it.

What he came up with surpassed our wildest dreams. Here, in highly condensed form, are but a few of the details of the SUPER-SYM he built for us:

1) All I/O, etc., relocated and "compacted" as follows:

```
VIA #1 from $A000-$A3FF to $F800-$F87F
VIA #2 from $A800-$ABFF to $F880-$F8FF
VIA #3 from $AC00-$AFFF to $F900-$F97F
SYSI/O from $A400+echo  to $FF00-$FF7F
SYSRAM from $A600+echo  to $FF80-$FFFF
```


2) SUPERMON relocated to $E000-$EFFF in 2 2716s on the SYM-1

3) RAE-1 relocated to $C000-$DFFF in 2 2716s on an AEP-1 (HI-0)

4) BAS-1 still at $C000-$DFFF in 2 2716s on an AEP-1 (HI-1)

5) POR circuit modified, and all external addresses changed in the EPROMs for self-consistency

6) Write protect circuitry and .W command modified to provide for bank-switching between two AEP-1 boards at $0000-$7FFF, LO-0, LO-1 and between two additional AEP-1 boards at $8000-$FFFF, HI-0, HI-1

7) Visible Memory (8 K RAM) is located at $A000-$BFFF

Bank-switch default at POR is to LO-0, HI-0 (with RAE-1). In the command .W wxyz, where wxyz are the four bits corresponding to a single hex byte, "w" and "z" are not used (future expansion for "z"), "x" selects between LO-0 and LO-1, and "y" between HI-0 and HI-1.

The MTU Card Cage holds "cards" on five levels. The SYM is at the top level (Level 1), covered with "smokey" lucite, with cutouts for "rare" access to the keypad, and to an AEP-2 I/O board which provides for four additional VIAs in the space assigned to VIA #2 (the AEP-2 plugs into socket U 28 in place of VIA #2).

The Visible Memory is at Level 5. Level 2 holds LO-0 on the "right" and HI-0 (with RAE-1) on the "left" (the default AEP-1s), while Level 3 holds LO-1 and HI-1 (with BAS-1).

That leaves Level 4 . . . .

We haven't decided what goes on the left side (we're sure Jeff will come up with several suggestions), but an FDC-1, with custom EPROM and with a modified addressing PROM goes on the right (the Expansion Connector side). Note that we have well over 1 1/4 K available at $F980-$FEFF. We'll put a 2716 in there, using the PROM to give it all address space in the 2 K block $F800-$FFFF not otherwise spoken for in 1) above (allowing also for the five addresses needed by the FDC-1 I/O registers).

This 2716 will hold a BOOTstrap program to download into RAM whatever DOS we decide to use. Since BOOTs are almost trivially short, typically at most one page or so, we'll still have over 1 K for all sorts of utility "goodies", as well.

With a disk system available we'll remove the BASIC and RAE EPROMs from the AEP-1 HI boards and replace them with RAM, downloading BASIC and RAE (and FORTH, naturally) as needed. We will then have contiguous RAM from $0000-$BFFF! That is 56 K, friends, not counting the bank-switching!! And there is still an isolated 2 K of RAM at $F000-$F7FF. What about the DOS? The latest word from Steve Cole, of the UK SYMmers Group, is that Arthur Richards estimates that he is about 80% of the way towards completion, and that we should be getting a copy for testing right around the first of the year. It should be ready to announce with our next issue.

We saw the tremendous amount of enhancements Arthur added to FODS, while at the same time compacting the object code more than we would ever have believed possible. Knowing Arthur's work as we do, we believe that his new FDC-1 DOS will be among the very best we have seen. We estimate it will occupy perhaps 6 K or so. Since it will be able to use the 2 K of RAM at $F000-$F7FF for overlays and buffer space, it will not take up too much of the contiguous 56 K of RAM.

Incidentally, Steve has begun to devote most of his time to the BBC computer. We can't blame him; we tried one out when we were in Australia this spring (our spring, that is; their fall). The cost advantage of the single-board computer, e.g., the SYM, over the appliance-type computer, e.g., Atari 800, Commodore 64, BBC Acorn, Timex-Sinclair, etc., is long since gone. The only remaining two advantages of the single board computer are: 1) you are required to understand more of its "inner workings", and, 2) the single board computer is much more truly a personal sort of thing. The best example we can give for both of these points is the "dream system" we have been describing above.

Jeff did his usual great job on this custom system for us, and provided us with complete documentation for all modifications made to the software, and to the hardware for the SYM-1 itself, the Card Cage, and all other (memory) boards used. He gave us an annotated SYM-1 schematic showing all POR changes and a schematic of the added logic to do the more extensive I/O decoding required. Since this was a custom job, and since space is limited, we cannot reproduce the details here. Contact Jeff directly if you wish more information, especially if you would like similar services. He can provide customized and/or relocated EPROMS for MON, BAS-1, RAE-1, etc., on request. We are now referring all requests for customized and OEM systems to him, since he has a faster response time than we do, with Dick Albers available for support and backup as needed.

Note that, except for the relocations, all software developed on this system will be fully compatible with standard SYMs. We'll borrow an idea from Jack Brown, and use conditionals in our source code, i.e., there will be lines like the following:

```
MYSIM     .DE  1     ;OR Ø IF NOT MYSIM

          IFE MYSIM

;NORMAL CODE AND/OR DEFINITIONS FOR STANDARD SYM GO HERE

          ***

          IFN MYSIM

;SPECIAL CODE AND/OR DEFINITIONS FOR MYSIM GO HERE

          ***
```

We are doing this now with many of the programs we distribute for cassette based systems, in that we define FODS .DE Ø, and include lines with FODS .DE Ø, IFE FODS, and IFN FODS, so that the user with FODS can redefine FODS .DE 1 to get the FODS linkages inserted. We hope soon to be able to do the same for FDC1.

COMPUTER SPEECH

We have long been using the SP-1 SPEAK & SPELL (TM) INTERFACE, marketed as a kit by David P. Kemp, for voice output from our SYM. We have made our SYM into a talking clock, as a novelty demonstration, but have used the SP-1 for a more practical purpose, with the .V (Verify) command, to read back to us, for code checking purposes, a hex dump of long tables which we have entered "by hand".

We understand that the kit is no longer available, and we thought we understood why. At the time we bought the SP-1 it was just about the only way to add, inexpensively, at least, speech capabilities to the SYM (the excellent manual, Release 1.1, bears a copyright date way back in 1979, almost prehistoric, by now).

Since that time a number of alternate approaches have become available, and a casual examination of their specifications and prices led us to believe that some of these approaches might be cheaper and better than the SP-1 approach, especially since the price of the Speak & Spell seemed to be inflating. Dave must have felt that the SP-1 was "obsolete", and that the market for it was dead, and we would have agreed with him.

We have since reexamined the matter, and changed our opinion. First, as to cost: With TI's very active rebate policy (currently $15.00) the S & S is available for as little as $34.97. While the kit is no longer available, except for the PC board, and a special socket to fit the Expansion ROM connector on the S & S, all other parts are obtainable locally at a very nominal cost.

Here is the COMPLETE parts list:

1 ea	4.7 ufd Tantalum Capacitor
1 ea	10 K 1/4 W Resistor
1 ea	2N2907 Transistor
1 ea	74175 Quad D Flipflop
1 ea	74368 Hex Tri-state Inverter
2 ea	74395 Four Bit Tri-state Shift Registers

(plus sundry sockets and 16 wire flat cables)

These parts can't be too expensive, anywhere. If the PC board is no longer available, a prototype board of some type could be substituted. Also, we did not really like the makeshift socket provided for interfacing to the S & S PC board, which is much thinner than standard. We would just as soon solder flat cable wires directly to the edge connector traces of the TI board, ourselves.

So much for cost. We think that this approach has got to be the least expensive way to go. And now for the effectiveness: First of all, Kemp's manual provides an educational experience in itself, on the general theory of LPC, and the specifics of the S & S implementation thereof. He provides fully commented source code listings of all required software. Second, additional theory and software listings are available in several manuals written by John P. Cater, "6502 Experimenter Package", and "6502 Phonetic Generator Software".

While we do not fully agree with his specific selection and implementation of phonemes, enough information is provided to add additional phonemes, and to include as many allophones (variants of phonemes, with differing pitches, lengths, levels, and inflections) as desired, to produce very, very, natural sounding speech, even regional dialects, if you wish.

The information provided by Kemp and Cater make the S & S approach one of the most versatile computer speech systems we have seen, at the lowest cost we know of. The price of the manuals should not really be considered only as part of the hardware cost, but rather as supplementary reading, or "required texts", if you like. With this in mind, we now feel that the S & S/SP-1 approach is the most cost/effective way to get 6502 voice I/O, bar none (unless you get the chips, etc., as a gift, or donation, of course).

If any of you are interested in following this approach further, please let us know of your interest. We will then contact Cater and Kemp for resale or reprint rights to their manuals, and ask Kemp if he wishes to provide the PC boards (not complete kits, as in the past), or license us to have them made. We could also obtain the TI interface connectors, as well. We think Dave would certainly be agreeable, especially if we provide the customer support, rather than he having to do it. Customer support can be difficult if you have moved on to other projects, but if

you are still actively interested it is kind of fun.

We could also provide RAE source code on cassette, to save you many hours of keying time. Our source code includes conditionals for Kemp's various subprograms, so that any combination(s) of them can be co-resident (we have eliminated duplicated labels), and all sections, including the extensive packed tables (and for Cater's software, the phoneme tables) are fully relocatable and easily extensible.

To end on a humorous note, we have been listening to Victor Borge's, by now "classical", comedy albums, one of which contains a monolog called "Phonetic Punctuation", in which he maintains that spoken speech would be less error-prone if the punctuation marks of written speech were also "soundable". He then assigns various noises and funny sounds to ".", "!", "?", ",", ";", ":", etc., and then "reads" a short story, with the voiced punctuation. We'd like to emulate that with our SYM!

P.S. The Speak & Spell (t/m) is also endorsed by E.T.(c)! E.T. THE EXTRA-TERRESTRIAL

P.P.S. We'd sure like to find a way to use that very nice eight character full alphanumeric fluorescent display to supplement the SYM's six seven-segment LED displays. We do like the green color, too.

MORE ON FDC-1 FIX

To ensure greater stability (?) on the +5 V line, tie more of the +5 V points together. We tied the hole marked "+" between RP1 and pin 14 on U8 to the +5 V turret pin with #22 hookup wire. We don't think the problem is due to lack of decoupling capacitors; rather we feel that there is too much ohmic resistance in the traces, or plated-through holes. The suggested fixes work, although just why is still uncertain.

EDITING BASIC FILES WITH RAE

Here are some of the explanatory notes and comment lines from a RAE program sent us by Rudolf Karg, a Swiss SYMmer, for review and marketing, if we found it to be useful. We tried it, we liked it, and are pleased to offer it as a new product.

After the program is assembled and the object code is stored in high RAM, BASIC is entered as usual, with the usual memory reserve. After you have tested your BASIC program, if you wish to do major editing on any of the program lines, call on this program as instructed. You will then find yourself in RAE with appropriate >SEt limits, where you will be able to use all of RAE's editing features to do such things as finding and/or renaming variables, etc. RAE's >PRint command, if preceded by a CTRL Y will send the program (and you) back into BASIC. It is very interesting to watch the program at work.

Naturally we disobeyed M. Karg's injunction to use the cold entry only once; we wanted to see what would happen with multiple use. Well, each cold entry cut BASIC's memory limit in half till there was no more left to use. The program can be very helpful for "polishing" up your BASIC programs.

```
0010 ;*************************************************
0020 ;*                                               *
0030 ;*  LINK PROGRAM BASIC - RAE/TED FOR SYM-1       *
0040 ;*                                               *
0050 ;*  COPYRIGHT 1980      R. KARG                  *
0060 ;*                      WILENSTR.27              *
0070 ;*                      CH-9500 WIL              *
0080 ;*                      SWITZERLAND              *
```



```
0090 ;*                                               *
0100 ;*  TAPE FILE 01  8.JAN.81                       *
0110 ;*  TAPE 046 SIDE A                              *
0120 ;*                                               *
0130 ;*  ENTER BASIC AND ALLOCATE MEMORYSIZE OR        *
0140 ;*  TYPE RETURN IF THIS PROGRAM IS STORED        *
0150 ;*  IN EPROM.                                    *
0160 ;*  ADD THE FOLLOWING LINES TO YOUR BAS-FILE     *
0170 ;*                                               *
0180 ;*  9997 END                                     *
0190 ;*  9998 X=USR(&"XXXX",0):LIST                   *
0200 ;*  9999 X=USR(&"YYYY",0):LIST                   *
0210 ;*                                               *
0220 ;*        XXXX  =  COLD.ENTRY                    *
0230 ;*        YYYY  =  WARM.ENTRY                    *
0240 ;*                                               *
0250 ;*  START CONVERSION FROM BASIC TO RAE/TED       *
0260 ;*  WITH GOTO 9998 (COLD ENTRY POINT)            *
0270 ;*  RETURN TO BASIC WITH CTRL Y  PR  RETURN.     *
0280 ;*  START EACH FURTHER CONVERSION TO RAE/TED     *
0290 ;*  WITH GOTO 9999 (WARM ENTRY POINT).           *
0300 ;*                                               *
0310 ;*************************************************
0320            .BA $5000
0330            .OS
0340 ACCESS     .DE $8B86     ;UNWRITE PROT SYST RAM
0350 EXECUTE    .DE $8855     ;MON EXECUTE ROUTINE
0360 PARNR      .DE $A649     ;NUMBER OF PARMS
0370 PAR.3      .DE $A64A     ;POINTER FOR EXECUTE ROUTINE
0380 OUTVEC     .DE $A663     ;OUTPUT DRIVER VECTOR
0390 POINTER    .DE $70       ;ASCII TRANSFERBUFFER-POINTER
0400 LASTMEMORY .DE $FD       ;LAST STORED CHARACTER
0410 MEMORYSIZE .DE $87       ;BASIC MEMORY SIZE
0420 TERM.WIDTH .DE $1A       ;BASIC TERMINAL WIDTH
0430 BUFF.END   .DE $6E       ;TRANSFERBUFFER-END
0440 BUFF.START .DE $6C       ;TRANSFERBUFFER-START
0450 TOUT       .DE $8AA0     ;TERMINAL CHR OUT
0460 RAE.WARM   .DE $B003     ;RAE WARM ENTRY
0470 CTRL.Y     .DE $00       ;RAE CTRL.Y VECTOR
0480 SHIFT      .DE $76       ;SCRATCH PAD MEMORY
0490 ADDM       .DE $77       ;SCRATCH PAD MEMORY
0500 DECIMAL    .DE $F9       ;SCRATCH PAD MEMORY
0510 ;


0850 ;********** BASIC TO RAE TEXT EDITOR **********
0860 ;THIS PROGRAM DIRECTS THE ASCII OUTPUT STREAM, CAUSED BY
0870 ;THE BASIC "LIST" COMMAND, TO A TRANSFERBUFFER LOCATED
0880 ;ABOVE THE BASIC FILE.
0890 ;AS SOON AS THE BASIC "OK" MESSAGE IS DETECTED (END OF
0900 ;LISTING) A $00 IS PLACED AT THE END OF THE STREAM WORKING
0910 ;AS LIMITER FOR THE MONITOR EXECUTE COMMAND. THEN THE
0920 ;BASIC USER COMMAND FOR THE RAE-1 COLD ENTRY IS PLACED
0930 ;AT THE BEGIN OF THE TRANSFERBUFFER, FOLLOWED BY SOME
0940 ;RAE SET UP PARAMETERS. AFTERWARDS THE OUTVEC IS CHANGED
0950 ;BACK TO TOUT AND UNDER MON EXECUTE COMMAND (STARTING
0960 ;AT THE BEGIN OF THE TRANSFERBUFFER) THE RAE IS ENTERED
0970 ;AND THE RAE TEXTFILE IS FILLED UP UNTIL A $00 TERMINATES
0980 ;THE EXECUTE COMMAND, HANDING OVER CONTROL TO THE RAE TEXT
0990 ;EDITOR.
1000 ;IN CASE OF TRANSFERBUFFER OVERFLOW DURING TRANSFER BEFORE
1010 ;RECEIVING THE END OF LISTING MESSAGE, THE REMAINING BASIC
1020 ;LINES WILL BE OUTPUTTED AND CONTROL REMAINS UNDER BASIC.
1030 ;**********
```

```
2170 ;********** RAE TEXT EDITOR TO BASIC **********
2180 ;THIS PROGRAM DIRECTS THE ASCII OUTPUT STREAM, CAUSED BY
2190 ;RAE COMMAND CTRL Y FOLLOWED BY "PR" RETURN, TO TRANSFER-
2200 ;BUFFER LOCATED ABOVE THE RAE TEXT FILE.
2210 ;NOTE THAT THE "PR" COMMAND IS NOT ECHOED AND THAT THERE-
2220 ;FORE IS A DELAY UNTIL THE TRANSFER IS VISIBLE ON THE CRT.
2230 ;AS SOON AS THE END OF PRINT MESSAGE "//" IS DETECTED
2240 ;A $00 IS PLACED INSTEAD OF "//" AT THE END OF THE STREAM.
2250 ;THEN THE BASIC COLD ENTRY TEXT IS GENERATED AND PLACED AT
2260 ;THE BEGIN OF THE TRANSFERBUFFER. THE OUTVEC IS CHANGED
2270 ;TO TOUT AND UNDER MON EXECUTE COMMAND BASIC IS ENTERED
2280 ;AND THE BASIC TEXT FILE IS FILLED UP UNTIL A $00
2290 ;TERMINATES THE RETRANSFER.
2300 ;**********
2310 ;
```

SOME QUESTIONS AND SOME ANSWERS

We reprint below a letter with some interesting questions, plus some interesting suggestions. We'll also answer his questions, following the letter.

Dear Lux,

As per our conversation concerning the problem with the Basic "PRINT" statement when printing exponential numbers, I have enclosed a listing and tape of a program that demonstrates the problem. The tape is in H.S. format and uses standard default values. Jerry Larsen of SSC said the problem is at address $C92B in the BAS-1 chip. The stored value is $0E and would have to be changed to allow for a larger field. This change would require burning a new chip or putting BAS-1 in RAM. A "PRINT USING" statement as an enhancement to BAS-1 would remove this bug. Do you have one that could be patched to the BAS-1 command list?

Speaking of enhancements to BAS-1, has anyone written an enhancement package to allow disk operations (OPEN, CLOSE, GET, PUT, FIELD), for creating and using Record I/O Files. Virtual files would also be nice. But, while I believe in miracles, this is probably beyond the capability of the Sym-1 and FDC-1. The CHAIN command would allow us to use programs larger than will presently fit in the 32K contiguous memory available on the Sym.

The reason why this letter is so late (I spoke with you two weeks ago) is that I fried the power supply to my Sym. I piggy-backed 4K of static memory chips on the existing 4K of memory on board (see attached article "Beat the High Cost of H-88/89 Memory Expansion; Steve Howard; Microcomputing, August, 1982, pg. 80) and added 16K of static RAM using a board I designed and built. It was a great feeling to see 24K come up on the screen as I signed on to BAS-1, until I noticed smoke coming off the transformer on my power supply. I have since built a new supply using two LM323 chips as regulators - 6 amps. should be enough for a while.

I have added a 2Mhz crystal to my Sym. I can switch select either the 1 or 2Mhz crystal with the Sym under power. My printer is attached to my video terminal through a parallel bus and therefore both must run at the same speed with the printer speed (30 CPS) being the limiting factor. I must therefore run my terminal at 300 baud. This causes problems when I try to connect to my Sym with 2Mhz clock speed. Typing in a "Q" to set the baud rate does not work because doubling the clock on the Sym causes only the follwing baud rates to be recognized: 220, 600, 1200, 2400, 4800, 9600. I got around the problem as follows:

1) Set Sym clock at 1Mhz
2) Turn on system
3) Type in "Q"
4) Sign on to Basic
5) Type in the following:
   X=USR(-29818,0)␣:␣POKE 42580,144␣:␣POKE 42577,156
   return (the spaces (␣) are important) (taken from Sym Physics 7:4)
6) Set Sym clock at 2Mhz
   Double the H.S. tape format default values if you want to read tapes created at 1Mhz clock rate or read in the tape at 1Mhz before you perform step 5 above.

The 2Mhz crystal change was made necessary by a program I use to have the Sym create mazes for my 4 year old son to solve. A 24 block by 50 block maze takes about 20 minutes to compute and 1.5 minutes to print. Gregory got so good at solving them that the Sym could not create them fast enough. The 2Mhz clock rate solved the problem. Now the Sym creates a 24 by 100 maze in the same 20 minutes. I am still trying to solve one of those!

In designing the 16K memory board that I mentioned before, I found an error on page 8-3 of the Sym-1 Reference Manual. I have included a copy of the page with corrections noted.

I am working on a hardware method (no fancy programing) to transfer programs from the PDP-11/70 at work direct to my Sym over a telephone line. I sometimes have to do computation work at home and using the Sym will save money on the phone bills. Some of the programs are over 12K long and not worth the time to key in by hand. The Basic enhancements mentioned before would eliminate the need to rewrite sections of the program to make them compatable with BAS-1.

That's it for now. Waiting to hear from you concerning the enhancements.

Sincerely,

Dennis Kochansky

Dennis Kochansky
118 Hidden Trail
North Plainfield, N.J. 07060

Dennis' problem is that he is attempting to "PRINT" a nicely formatted tabular display of results, depending on the use of ";" in his PRINT statements to do the tabbing for him. Whenever the numbers are too small and have too many significant figures, e.g., 0.00987654321, which BASIC prints out as 9.87654321E-03, his numbers spill over the tab positions (the $0E gives a FIXED field of 13 positions, which doubles to 26 for long exponentials).

The simplest solution is to use "," between variables, not ";", and use TAB(N) to do the tabbing. Actually, you may not even need the ",", we think (we leave this as an exercise for the reader!). There is no problem in transfering BAS-1 into EPROMS, modifying it along the way, as desired, except for perhaps not having enough sockets.

Jack Brown's newest BASIC enhancements do have PRINT USING and CHAIN. And, please do continue to believe in miracles, since what you ask is NOT beyond the capabilities of the SYM-1/FDC-1 combo; just wait till next spring for SUPERDOS for FDC-1!

ANOTHER FDC-1 NOTE

We thought we had a real problem with our FDC-1 system, and spent several weeks thinking about a solution. It seems that 25-50% of the times that we powered on, the disk drives started spinning, often with heads loaded and LEDs on; resetting the SYM would not turn them off.

The problem is that the signals to the (read-only) Drive Control Register at $F1XX (or $AFXX on special order) are actually generated by the Hex D-Type Flip-Flop 74LS174 at U13 (MOTOR-ON, SIDE-SELECT, DS-1, and DS-2 go directly to the drives, HLT and DDEN* go to the SY1791-02). The power-on state of these flip-flops is, of course, indeterminate.

Writing $FF or, at least, $04, to $F1XX, will turn off the drives. You will get a "?" when you write, naturally, as the read-back of these addresses will always read $00. Entering $00 will give no "?", but will leave the drives running.

MISCELLANIA

DICK ALBERS and JEFF LAVIN of Alternative Energy Products advise that their next two products for the SYM-1 (and, incidentally, also for AIM 65), are now entering the final development stages, that we should have early prototype units for our evaluation before the end of the year, and that announcement of price and availability can be made in the next issue of SYM-PHYSIS.

These are: 1) an ACIA card permitting asynchronous communication at rates up to 9600 baud, and, 2) an EPROM Burner capable of handling ALL EPROMs from 2516 up, with simple header changes and software options. Jeff, with his hardware know-how, and Dick, with his software and analytic skills, form a truly SYM-biotic pair!

# # #

JACK BROWN, of Saturn Software, sent us five diskettes with review copies of new SYM software. Among the collection is a new DOS, called RAE-DOS, which adds a whole bunch of new commands to RAE, extending its capabilities tremendously. RAE-DOS is usable only with FODS systems.

One of the disks is used with the FODS system to reBOOT to RAE-DOS; the other four disks are then accessed with RAE-DOS. We briefly tested the system and examined many of the utilities and other goodies supplied; it would take many long hours to learn how to exploit all of the treasures there. Jack also supplied us with the RAE-DOS Manual, and the manual for Ralph Deane's MEAN14 (which adds Floating Point Arithmetic to RAE). That is one we will find useful for very fast scientific computations in machine language, such as FFT, etc.

While we are dealers for all of his earlier software, Jack wishes to have all orders for his newer software items to be placed directly through Saturn Software. Thus, we suggest that you contact him directly, and get on the Saturn Softnews mailing list for announcements of his new products.

On the other hand, Jack realizes that preparing and distributing Cassette, FODS, and CODOS versions take up so much of time and energy, that he is hesitant to also support FDC-1 versions. Since we have the ONLY Dual/Dual FODS/FDC-1 SYM that we know of, we have the equipment to do the conversion, testing, and distribution of FDC-1 versions of certain selected software items. We are currently discussing with Jack the possibility of becoming the distributor for FDC-1 SUPERDOS versions. Incidentally, there are already far more FDC-1 SYMs out there than FODS and CODOS combined.

# # #

SERGE MATOVICK, of Incon Electronics Inc., 782 Damien Way, Mississauga, Ontario, Canada L5C 3H2, (416) 273-4499, sent us photographs and spec sheets for three products he helped design. These are a Programmable Controller, a Programmer-Emulator, and a Simulator. These look OK, and the prices seem reasonable, but we have NOT tried them personally. If you wish additional information on these items, contact Serge directly.

# # #

Thanks to everyone whose contributed programs and/or articles were deferred to future issues. Space is at a premium, of course, so not every item submitted can be published. We "referee" the articles for "quality", of course, whatever that means, but our choice is based mainly on getting sufficient variety into each issue so that each reader, hopefully, will find at least one article per issue which justifies his subscription costs.

We try to "validate" each program by actual test, and each hardware suggestion by going over the theory involved. We now have enough voluntary reviewers to speed up the process as follows: We will transcribe all received cassettes to disk, or make copies of received diskettes, and Xerox all accompanying manuscript material. We will then send the original materials to the reviewers, notifying the authors of the status.

Several readers have been kind enough to have sent "Computerized Indexes" to partial volumes of SYM-PHYSIS, but these are not in a form which permits easy cumulative updating. "SANDY" MACKAY has sent us a copy of his DATA MANAGEMENT SYSTEM (DMS), which runs under Brown's Extended Disk BASIC (EDB-FODS version), but up to now we have been memory limited. When we get our FDC-1 SUPERDOS going on the SUPERSYM, with all that RAM available, we'll write our own DMS in FORTH. If all goes well, we'll mark up a complete set of back issues with appropriate KEYWORDs, and have the data entered in page number sequence, then sorted by KEYWORDs. This is a long-range project, and our plan is to include an Index to Issues 0 through 17 in Issue 17.

Here's a CONTEST ANNOUNCEMENT: We'll award a complimentary "Lifetime" Subscription to whoever submits the best new masthead to be used in Volume 4. We'd prefer something which uses the graphics capabilities of the Epson, but will accept camera-ready copy, otherwise. Entries due by 1 March 1983.

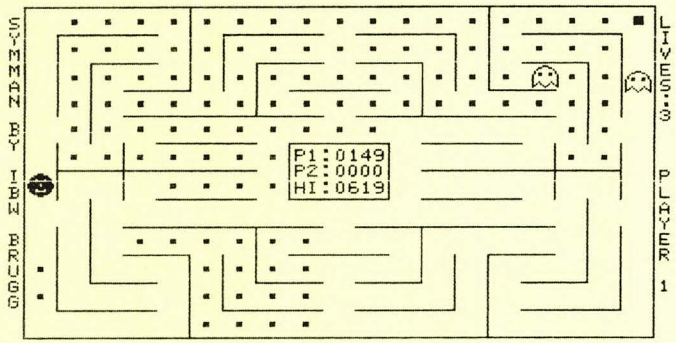
VIC-20 & COMMODORE 64

We had long been thinking that the VIC-20 would be a far better buy than the RCA VP3301 Data Terminal which we have been using as an output peripheral for our SYM, mainly for video titles. We were also thinking that the VIC-20, with its better keyboard than the Timex-Sinclair (plus color), was bringing closer the day when college students would be required to have their own computers, just as once they provided their own slide-rules (remember?), and we were "evaluating" the VIC-20 with that possibility in mind.

We still think the VIC-20 would provide a good beginners' introduction to computers, but the Commodore 64 is a much more value-packed item, one we want to learn more about. We therefore visited a local computer store to study the manuals on the Commodore 64. We skimmed through the User's Manual and plan to return when the Programmer's Manual is in stock to read that, too.

While there, we bought a Commodore 64 Joystick, and soon as we can get a DB-9 male connector, we'll be playing SYMMAN!

SYMMAN, REVISITED



We added the joystick (we just couldn't wait) and played a few (we'd rather not say how long we were at it!) games. Notice the HI score of 619, made by our oldest son, visiting, with his wife and our first grandchild, for Thanksgiving Day. He interfaced the joystick, so he got to play first.

We stopped the game after taking out a corner square so that we could make a printout for you. Note the change in "color" of the octopi; they are now vulnerable to Mr. SYMMAN. After cleaning out all of the dots additional octopi appear to make the game even more challenging.

LOGOUT

Issue 13/14 was fun to put together, with so many readers' contributions to chose from. The hardest part was not having room for all of them, and having to omit so many good items. Now we can turn our energies to personal studies in Voice I/O, and becoming thoroughly proficient in FORTH. We also will be "producing" several half-hour demo videotapes for classroom and lecture use. And, now that the pressure is off, for a while, at least, we'll try to answer the backlog of letters. Also, we'll try to acknowledge future contributions immediately on receipt.

This issue should reach you just before Christmas Day, so let us wish each of you the Season's Best, and a Very Happy New Year. The next three issues are scheduled for mailing at the end of March, July, and November of 1983. Look for us then.

PRODUCT ANNOUCEMENTS

HARDWARE

CLK-1

A new product from Alternative Energy Products (Jeff Lavin). Described in Issue No. 13/14. Ready to install and use, with software on cassette in RAE-1 source code. The card mounts directly on the AEP-2 I/O Board, or it may be "cabled" directly to either of the Application Edge Connectors. Price is $60.00 US/Canada, $63.00 elsewhere, postpaid first-class or airmail. Backup batteries not included!

CUSTOM PROMS FOR FDC-1

We have a few 256x4 N82S129 Bipolar PROMs with "pages" $F0 and $F1 relocated to $AE and $AF, respectively, available at $12.00, postpaid first-class or airmail anywhere.

SOFTWARE

KARG'S BASIC TO RAE EDITOR

Described in Issue No. 13/14. Cassette; complete RAE-1 source code, full instructions, $36.00 postpaid first-class or airmail anywhere.

WUETHRICH'S SYMMAN

Described in Issue No. 13/14. Cassette; complete RAE-1 source code, full instructions, $36.00 postpaid first-class or airmail anywhere.

CARL MOSER'S ASSM/TED (6800) FOR SYM-1

This is just a reminder to those of you using 6800 systems as well as SYMs that this outstanding 6502 to 6800 Cross Assembler (works just like RAE, except for the 6800 mnemonics) is still available at $75.00, first-class or airmail anywhere. Object code on cassettte, resident at $2000-$416A.

PUBLICATIONS

Elcomp's "MICROCOMPUTER HARDWARE HANDBOOK", an 846 page collection of off-prints of spec sheets of lots of TTL, FAST TTL, CMOS, Voltage Regulator, RAM, EPROM, EEPROM, ROM, CPU, Support Circuit, and Interfacing Circuit Chips is available at $17.00 US/Canada, and $18.00 overseas, surface mail only. While it does not cover the very newest, state-of-the-art chips, it is reasonably complete on the "classical" chips, and is handy to have around when you need it.

PRICE INCREASES

We regret that we must pass on publishers' price increases for the following two books:

Leventhal and Seville's "6502 ASSEMBLY LANGUAGE SUBROUTINES", now $15.50 US/Canada, book-rate, and $18.00 overseas, surface mail.

Zumchak's "MICROCOMPUTER DESIGN AND TROUBLESHOOTING", now $17.50 US/Canada, book-rate, and $18.50 overseas, surface mail.

The AEP-2 I/O Board is now $60.00 US/Canada, $63.00 elsewhere, postpaid first-class or airmail. The AEP-2 I/O Board plugs directly into the VIA #2 socket (U28). If your SYM has been built into an enclosure which does not include sufficient space for direct installation, a special model with an 8" extension cable is available for an additional $12.00.

ATTENTION - OVERSEAS SUBSCRIBERS

We can no longer accept checks from overseas customers which do not bear MICR coding because of the $5.00 to $10.00 surcharge required to cash them. Paying $10.00 to cash a $14.00 check is not a sound business practice. Please switch to a bank using MICR coding.

Here is a sample of MICR encoding: ⑈003439?⑈ ⑆063104972⑆

ATTENTION - ALL SUBSCRIBERS

CONTACT US FOR PRICES ON ANY SYNERTEK PRODUCT

WE WILL MEET ANY ADVERTISED PRICES

SPECIAL QUANTITY, ACADEMIC, AND STUDENT DISCOUNTS