SYM-PHYSIS

THE SYM USERS' GROUP NEWSLETTER

VOLUME IV, NUMBER 2 (ISSUE NO. 16) - SUMMER 1983 (MAY/JUN/JUL/AUG)

SYM-PHYSIS is a thrice yearly 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 IV, 1983, Issues 15 - 17)

USA/Canada - $10.50 for a volume of three 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 AVAILABLE AS FOLLOWS:

Issues 0 through 6 (Volume I, 1979/80), are available 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.

Issues 11 through 14 (Volume III, 1982), are available for $10.50, US/Canada, and $14.00, First Class/Airmail, elsewhere.

ANOTHER OUTSTANDING OFFER TO THE SYM COMMUNITY

On the page to the right is a map of the environs of the Caves of Nirdarf. This map is part of the 28 page manual for "SYM-VENTURE", sent to us, along with a (KIM-speed) cassette containing the object code for the game, by Matt Ganis. The object of the "adventure" is to find the "treasures" (gold and pearls), preferably visiting every location during the quest, and to return home (i. e., to the house), safely, with them.

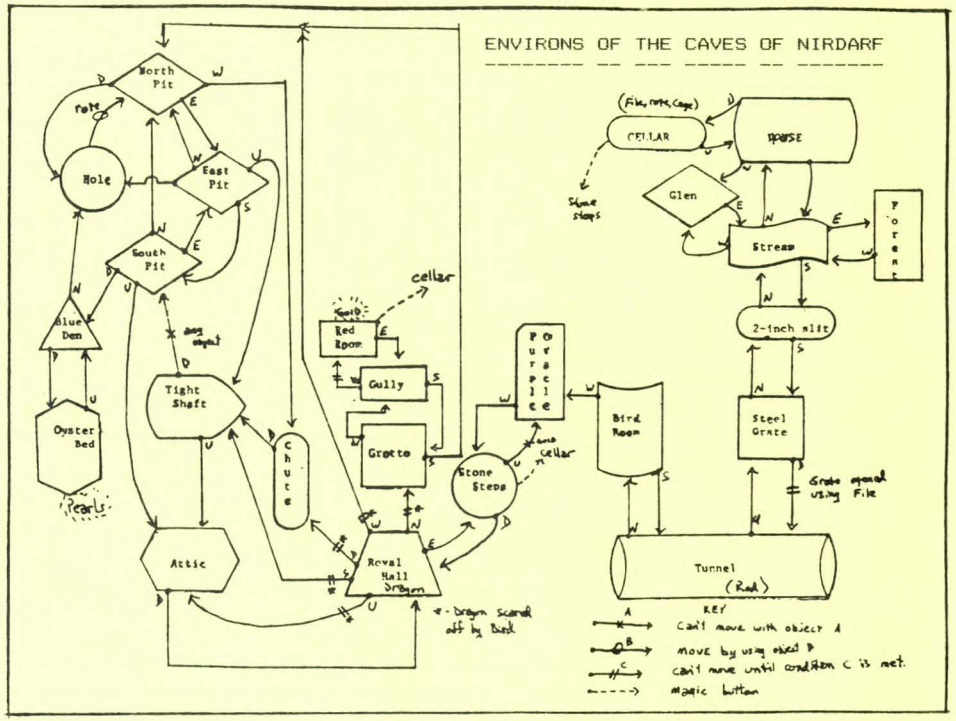
While we can't usually find the time for most computer games, we did make time for this one, for a number of very valid reasons:

1. It can be played on a 2K SYM-1 WITHOUT a terminal!
2. The manual contains fully commented source code, as well as very well written instructions and the map, which will make playing the game much more fun, and definitely much less frustating!
3. It is very inexpensive, almost at cost of media and shipping!
4. A study of the source code will not only reveal all the "secrets" of writing money-making Adventure-type games, but, because of the clever use of the 7-segment displays to display text messages, will also teach you much about how the SYM itself works!

SYM-VENTURE (c) is an adaptation (by permission) of Robert Leedom's original KIM-VENTURE (c), which we did not know existed, back in our KIM-1 days (incidentally, KIM-1 had only 1 1/8 K of RAM).

You may order either KIM-VENTURE or SYM-VENTURE (or both), cassette plus manual, at $15.00 each (overseas, please add $3.00 for Air Mail), directly from Matt Ganis, Sheridan Road, R. D. #3, Lebanon, NJ 08833. Copies may be duplicated by clubs, or users' groups for a $5.00 per copy royalty fee (very generous, and extremely reasonable, this!). Every SYMmer should send for a copy!

P. S. SYM-VENTURE, and "SWISS" CLOCK (see below), are ideal programs for demonstrating the potential of the SYMple (unexpanded) SYM-1!



MORE ON THE "SWISS" CLOCK

In the article "ADJUSTABLE REAL TIME (SWISS) CLOCK" (SYM-PHYSIS 13/14-9), we pointed out that the program was four bytes too long for an unexpanded SYM-1, or a SYM-2. The author himself, Mr. Schumacher, sent in a shortened version. We print, instead, the following postscript to a recent letter from Boris Goldowsky (author of the SYM-PHYSIS INDEX):

> P. S. To fit "SWISS CLOCK" into 1K, use one of my favorite tricks: On lines 3460-3470 & 3540-3550, replace LDA #$01 CLC with SEC. For astronomers, the clock can be made to show sidereal time by replacing 03B2 and 03C7 with C2, 03A5 with A8, and 03AD with CB.

MORE ON PRINTERS

Our original hardcopy device was a KSR-35 TTY, long since abandoned. Next came a decwriter LA-36, still useful on the 20 mA current loop, at 600 baud, and as an "emergency" full terminal at 110 baud on SYMs with no printer driver resident. Next came the Epson MX-80F/T, to which we added Graftrax; this is now our main workhorse. At the time we bought it we felt it was the most cost-effective printer available.

For our VIC=20 and COM-64 we purchased a VIC-1525 Graphics Printer, mainly because last December it was the ONLY choice. As of now it is still the only printer available which (easily) prints the CBM graphics characters. It does not work quite right with the Quick Brown Fox Word Processor, or with MAE, however, and we have not yet figured out the fix. [FIX: THE 1525 HAS A BUG AND NEEDS UPGRADING ROM TO BECOME 1525-E!]

We now have a Star Gemini 10X on our COM-64, and plan on getting several more for other systems, because we feel it now holds the "Lux Cost-Effectiveness Award". When we received the following material from "Sandy" Mackay, we added the "extra" lead to our interface (we had wisely left a few spare wires in the cable) and (temporarily) interchanged the Epson and the Gemini, so that we could reprint below his letter, his demonstration, and best of all, the complete source code for the SYM/RAE/BAS-Gemini Printer Patch. The patch also works with the Epson (with care to avoid conflicting control codes and escape sequences).

A. M. MACKAY P.ENG., O.L.S.
600 SIXTH AVENUE, WEST
OWEN SOUND, ONTARIO
(519) 376-8442

July 11, 1983

Dear Jean & Lux:-

As promised, here is the revised version of the GEMINI-10 printer driver. It is at least twice as fast as the one I sent earlier, and with fewer glitches. It runs at almost the full speed of the printer. Also, this driver will SWP your text AUTOMATICALLY. It's on the tape, RAE format, double speed, three times. GE F1.

You can print this with your Epson cable if you run the version I sent you previously; but better yet, why not add an extra line to your cable, connecting Gemini (or Epson) pin 10 to AA-E (CA-1), and you can use the same cable for both the Epson and the Gemini-10. Now that's versatility! You will probably have to disable your interrupts (see lines 780-800 this version) in your Epson driver.

Further to our telecon, I would prefer having this published in S.P. rather than marketing it, being a professed glory-hound with an insatiable ego, but I originally didn't think that you'd want to blow four or five pages on it. It is gratifying to realize that the distinguished publisher of our Bible knows quality when he sees it!

This letter and the Demo are also on the tape - GE F2 and GE F3 respectively.

Can't wait for the next issue of S.P. - I hope to convert one of my two SYM's to Jeff Lavin's Super SYM.

Keep the faith!

Sandy

GEMINI-10 PRINTER DEMO

COMPRESSED SUBSCRIPT nice and small
ELITE SUBSCRIPT a little larger
PICA SUBSCRIPT too many dots
REGULAR COMPRESSED for people with narrow paper
REGULAR ELITE nice size for personal letters
REGULAR PICA good for listings dump
DOUBLE WIDTH COMPRESSED effective block print
DOUBLE WIDTH ELITE not too bad
DOUBLE WIDTH PICA stretched out
ANY OF THESE CAN BE DOUBLE STRIKE
THIS IS EMPHASIZED PRINT and is nice for correspondence
THIS IS EMPHASIZED DOUBLE-STRIKE too dark - looks threatening
This is also emphasized double-strike
This is emphasized italics, underlined. Looks good!.
$A_1^2+B_1^2 = C_3^3$ subscript & superscript
I=10μA at 68°F or 20°C when R = 100KΩ
ΦθΣσωΞ±×÷○

NO TAB   TAB 10   TAB 25 TAB 50
Demo of tab setting (above), and margin setting (below):

LEFT MARGIN IN 20 SPACES
And all kinds of other goodies

[NOTE (by Lux): Today's dot matrix printers are much faster and much more versatile than shaped character printers, cost much less, and the printing can be just as attractive, including proportional spacing, if desired. The Gemini will accept a down-loaded character set into its RAM; this will enable it to print the CBM graphic sets, or your own design type fonts. Not demonstrated above are the Gemini's point graphics modes.]

```
;  by A. M. Mackay
;     600  6th Ave.  West                    ---------------
;     Owen Sound, Ontario                      CODOS  FILE
;     Canada        N4K 5E7                   GEMDRIVER.A
;     July 9, 1983  Rev.1                    ---------------



;      GEMINI-10 PRINTER DRIVER

;    GEMINI-10 PARALLEL PRINTER PATCH FOR SYM-1 AND KTM

;       (DISPLAYS CONTROL CODES AND ESCAPE SEQUENCES)


;  This program works if the Gemini-10 is connected to the
;  Sym-1 exactly as described in Sym-Physis 9:5 (Alternative
;  Wiring List), EXCEPT that "BUSY" is not used. Instead connect
;  Gemini pin 10 "ACK" to pin AA-E (CA1).

;  If X-RAY is being used, driver must be running for
;  KTM to accept Gemini-10 commands.

;  RU $5E00, then ↑E to take printer off line.
;  ↑W puts printer back on line. To print RAE source,
;  type FR↑W. Don't forget ↑E when finished.

;  Many CTRL sequences used by the Gemini-10 conflict with
```

```
0300  ;  those used by KTM, RAE and X-RAY. To get around this,
0310  ;  SUBSTITUTE ↑\ for ↑@, ↑V for ↑I, ↑W for ↑Q, ↑E for ↑S,
0320  ;  ↑F for ↑H, ↑J for ↑T,and ↑↑ for ↑O IN YOUR GEMINI-10 MANUAL.
0330  ;  ↑\ will be "displayed" as ↑@, and the others will be displayed
0340  ;  as substituted. ↑G and ↑M are not used in this driver.
0350
0360  ;  Most esc sequences also conflict, so we will SUBSTITUTE
0370  ;  SHIFT ESC (+/-) for ESC in ALL text to be printed.
0380
0390  ;  The Gemini-10 sequences are kept from the terminal, but
0400  ;  are "displayed" there with the prefixes "↑" for CTRL and
0410  ;  "+/-" for ESC.
0420
0430  ;  Where "n" is called for in the Gemini-10 manual, EACH "n"
0440  ;  MUST be PRECEDED WITH A SHIFT ALPHA , and "n" MUST be
0450  ;  entered as a TWO-DIGIT HEX number, eg. (shift alpha)3D or
0460  ;  (shift alpha)0A, with a MAXIMUM of (shift alpha)7F.
0470
0480  ;  To use SWP-2.5, change "JSR WRT." under "WRT.XY" in your SWP
0490  ;  source code to JSR $D036 (or wherever "PRINT" is in your driver).
0500  ;  Then instead of PR^W just hit ↑P and your text will be SWP'd
0510  ;  automatically. Don't forget ↑E when finished, to get off line.
0520
0530
0540  ;  ALL ESCAPE SEQUENCES MUST BE ENTERED AS SHIFT ESCAPE (+/-).
0550  ;  ----------------------------------------------------------
0560

                0570            .BA $5E00     ;or somewhere
                0580            .OS
                0590
                0600 SWP        .DE $C000     ;or wherever your SWP-2.5 is
                0610 TOUT       .DE $8AA0     ;Vector to terminal
                0620 ACCESS     .DE $8B86     ;Un-write protect sysram
                0630 OUTVEC     .DE $A663     ;Vector to printer
                0640 PAD        .DE $A801     ;Port A Data Register
                0650 PADD       .DE $A803     ;Port A I/O Direction Reg.
                0660 PCR        .DE $A80C     ;Perif. Ctrl Reg.
                0670 IFR        .DE $A80D
                0680 IER        .DE $A80E
                0690
                0700
5E00- 20 86 8B  0710 INIT       JSR ACCESS    ;unprotect Sysram
5E03- A9 FF     0720            LDA #$FF      ;initialize flags
5E05- 8D A2 5F  0730            STA BRFL      ;flags are $FF if down
5E08- 8D A3 5F  0740            STA ESCFL     ;  and $00 if raised
5E0B- 8D A4 5F  0750            STA NFL
5E0E- 8D A5 5F  0760            STA PRFL
5E11- 8D A6 5F  0770            STA STRIPFL
5E14- 8D A7 5F  0780            STA SWPFL
5E17- A9 36     0790            LDA #L,PRINT          ;Set printer vectors
5E19- 8D 64 A6  0800            STA OUTVEC+1
5E1C- A9 5E     0810            LDA #H,PRINT
5E1E- 8D 65 A6  0820            STA OUTVEC+2
5E21- A9 0B     0830            LDA #%00001011        ;CA2 = strobe out
5E23- 8D 0C A8  0840            STA PCR       ;CA1 = ACK (pos trans)
5E26- A9 7F     0850            LDA #%01111111
5E28- 8D 03 A8  0860            STA PADD      ;bits 0-6 data out
5E2B- 8D 0E A8  0870            STA IER       ;disable interrupts
5E2E- AD 01 A8  0880            LDA PAD
5E31- 8D 01 A8  0890            STA PAD
5E34- 18        0900            CLC
5E35- 60        0910            RTS
                0920
                0930
```



```
5E36- C9 7D     0940 PRINT      CMP #$7D      ;is it a shift alpha?
5E38- D0 07     0950            BNE ESC?
5E3A- EE A4 5F  0960            INC NFL       ;raise flag for "n"
5E3D- 20 A0 8A  0970            JSR TOUT
5E40- 60        0980            RTS
                0990
5E41- C9 7B     1000 ESC?       CMP #$7B      ;is it shift esc?
5E43- D0 0B     1010            BNE STRIP?
5E45- EE A3 5F  1020            INC ESCFL     ;raise flag for next char.
5E48- 20 A0 8A  1030            JSR TOUT      ;send +/- to CRT
5E4B- A9 1B     1040            LDA #$1B      ;and esc to printer
5E4D- 4C 70 5F  1050            JMP PR.OUT
                1060
5E50- 48        1070 STRIP?     PHA  ;         strip off ASCII "3"?
5E51- AD A6 5F  1080            LDA STRIPFL
5E54- 30 0D     1090            BMI PROC.ESC
5E56- CE A6 5F  1100            DEC STRIPFL
5E59- 68        1110            PLA
5E5A- 20 A0 8A  1120            JSR TOUT
5E5D- 38        1130            SEC
5E5E- E9 30     1140            SBC #$30
5E60- 4C 70 5F  1150            JMP PR.OUT
                1160
5E63- AD A4 5F  1170 PROC.ESC   LDA NFL       ;check for "n"
5E66- 10 28     1180            BPL H.NYB
5E68- AD A3 5F  1190            LDA ESCFL     ;esc sequence?
5E6B- 30 53     1200            BMI TEST^     ;br if no
5E6D- CE A3 5F  1210            DEC ESCFL     ;set up for
5E70- 68        1220            PLA  ;          esc sequence
5E71- C9 2D     1230            CMP #'-       ;if it's a "-"
5E73- F0 10     1240            BEQ SETSTRIP          ;  strip next char.
5E75- C9 42     1250            CMP #'B       ;    etc.
5E77- F0 0C     1260            BEQ SETSTRIP
5E79- C9 53     1270            CMP #'S
5E7B- F0 08     1280            BEQ SETSTRIP
5E7D- C9 55     1290            CMP #'U       ;less than "U"?
5E7F- 90 07     1300            BCC @TEST
5E81- C9 5A     1310            CMP #$5A      ;greater than "Y"?
5E83- B0 03     1320            BCS @TEST
5E85- EE A6 5F  1330 SETSTRIP   INC STRIPFL   ;flag to strip next char.
5E88- EE A5 5F  1340 @TEST      INC PRFL      ;don't trigger printer
5E8B- 48        1350            PHA
5E8C- 4C C0 5E  1360            JMP TEST^
5E8F- 60        1370            RTS
                1380
5E90- AD A2 5F  1390 H.NYB      LDA BRFL      ;pack next two chars
5E93- 10 10     1400            BPL L.NYB     ;into one byte
5E95- EE A2 5F  1410            INC BRFL      ;flag for 2nd char.
5E98- 68        1420            PLA
5E99- 20 A0 8A  1430            JSR TOUT
5E9C- 0A        1440            ASL A         ;move to high nybble
5E9D- 0A        1450            ASL A
5E9E- 0A        1460            ASL A
5E9F- 0A        1470            ASL A
5EA0- 18        1480            CLC
5EA1- 8D A8 5F  1490            STA NYBREG    ;store it and wait
5EA4- 60        1500            RTS  ;          for next char.
                1510
5EA5- CE A2 5F  1520 L.NYB      DEC BRFL      ;2nd char -
5EA8- CE A4 5F  1530            DEC NFL       ;lower flags.
5EAB- 68        1540            PLA
5EAC- 20 A0 8A  1550            JSR TOUT
5EAF- C9 3A     1560            CMP #$3A      ;digit or letter?
5EB1- B0 05     1570            BCS @1
5EB3- 38        1580            SEC  ;         it's a digit
```

```
5EB4- E9 30     1590          SBC #$30     ;strip off ASCII
5EB6- 10 02     1600          BPL @2
5EB8- E9 37     1610 @1       SBC #$37     ;letter - strip ASCII
5EBA- 0D A8 5F  1620 @2       ORA NYBREG   ;add to high nybble
5EBD- 4C 70 5F  1630          JMP PR.OUT   ;send it
                1640
5EC0- 68        1650 TEST^    PLA
5EC1- C9 5E     1660          CMP #'^      ;RAE flags ctrl codes with "^"
5EC3- F0 03     1670          BEQ @3       ;not a ctrl code
5EC5- 4C 6D 5F  1680          JMP NOT^
5EC8- A5 C9     1690 @3       LDA *$C9     ;BAS-1 stores a $4C here
5ECA- C9 4C     1700          CMP #$4C
5ECC- D0 03     1710          BNE @4
5ECE- 4C 6B 5F  1720          JMP PRINT^
5ED1- BA        1730 @4       TSX
5ED2- BD 03 01  1740          LDA $103,X   ;RAE "stacks" ctrl codes
5ED5- C9 20     1750          CMP #$20
5ED7- 90 03     1760          BCC @5
5ED9- 4C 6B 5F  1770          JMP PRINT^   ;not a ctrl code
5EDC- C9 0B     1780 @5       CMP #$0B
5EDE- F0 78     1790          BEQ GEM^
5EE0- C9 0C     1800          CMP #$0C
5EE2- F0 74     1810          BEQ GEM^
5EE4- 48        1820          PHA
5EE5- A9 40     1830          LDA #$40     ;RAE will convert to a null
5EE7- 9D 03 01  1840          STA $103,X
5EEA- A9 5E     1850          LDA #'^
5EEC- 20 A0 8A  1860          JSR TOUT     ;send "^" to crt
5EEF- 68        1870          PLA
5EF0- C9 10     1880 SWP?     CMP #$10     ;is it ^P?
5EF2- D0 13     1890          BNE E>S
5EF4- 20 62 5F  1900          JSR DISPL
5EF7- AD A7 5F  1910          LDA SWPFL    ;toggle SWP
5EFA- 49 FF     1920          EOR #$FF
5EFC- 8D A7 5F  1930          STA SWPFL
5EFF- A9 11     1940          LDA #$11     ;put printer on line
5F01- 20 70 5F  1950          JSR PR.OUT
5F04- 4C 00 C0  1960          JMP SWP
5F07- C9 05     1970 E>S      CMP #$05     ;is it a ^E?
5F09- D0 07     1980          BNE V>I      ;br if no
5F0B- 20 62 5F  1990          JSR DISPL
5F0E- A9 13     2000          LDA #$13     ;change to ^S
5F10- D0 3D     2010          BNE OK       ;br always
5F12- C9 16     2020 V>I      CMP #$16
5F14- D0 07     2030          BNE W>Q
5F16- 20 62 5F  2040          JSR DISPL
5F19- A9 09     2050          LDA #$09
5F1B- D0 32     2060          BNE OK       ;br always
5F1D- C9 17     2070 W>Q      CMP #$17
5F1F- D0 07     2080          BNE F>H
5F21- 20 62 5F  2090          JSR DISPL
5F24- A9 11     2100          LDA #$11
5F26- D0 27     2110          BNE OK       ;br always
5F28- C9 06     2120 F>H      CMP #$06
5F2A- D0 07     2130          BNE \>@
5F2C- 20 62 5F  2140          JSR DISPL
5F2F- A9 08     2150          LDA #$08
5F31- D0 1C     2160          BNE OK       ;br always
5F33- C9 1C     2170 \>@      CMP #$1C
5F35- D0 04     2180          BNE ]>T
5F37- A9 00     2190          LDA #$00     ;displays as ^@
5F39- D0 14     2200          BNE OK       ;br always
5F3B- C9 1D     2210 ]>T      CMP #$1D
5F3D- D0 07     2220          BNE ^>O
```



```
5F3F- 20 62 5F  2230          JSR DISPL
5F42- A9 14     2240          LDA #$14
5F44- D0 09     2250          BNE OK       ;br always
5F46- C9 1E     2260 ^>O      CMP #$1E
5F48- D0 08     2270          BNE NOTGEM
5F4A- 20 62 5F  2280          JSR DISPL
5F4D- A9 0F     2290          LDA #$0F
5F4F- 4C 70 5F  2300 OK       JMP PR.OUT   ;send to printer
                2310
                2320
5F52- 20 62 5F  2330 NOTGEM   JSR DISPL
5F55- 4C 70 5F  2340          JMP PR.OUT
                2350
5F58- 48        2360 GEM^     PHA
5F59- A9 5E     2370          LDA #'^
5F5B- 20 A0 8A  2380          JSR TOUT
5F5E- 68        2390          PLA
5F5F- 4C 70 5F  2400          JMP PR.OUT
                2410
5F62- 48        2420 DISPL    PHA  ;        display char
5F63- 18        2430          CLC  ;          on CRT
5F64- 69 40     2440          ADC #$40
5F66- 20 A0 8A  2450          JSR TOUT
5F69- 68        2460          PLA
5F6A- 60        2470          RTS
                2480
5F6B- A9 5E     2490 PRINT^   LDA #'^
5F6D- 20 A0 8A  2500 NOT^     JSR TOUT
                2510
5F70- 48        2520 PR.OUT   PHA
5F71- C9 1F     2530          CMP #$1F     ;don't trigger
5F73- 90 19     2540          BCC ACK      ;  if esc or
5F75- AD A7 5F  2550          LDA SWPFL    ;    flags are raised
5F78- 10 14     2560          BPL ACK
5F7A- AD A5 5F  2570          LDA PRFL
5F7D- 10 0F     2580          BPL ACK
5F7F- AD A3 5F  2590          LDA ESCFL
5F82- 10 0A     2600          BPL ACK
5F84- AD A4 5F  2610          LDA NFL
5F87- 10 05     2620          BPL ACK
5F89- A9 01     2630          LDA #$01
5F8B- 8D 01 A8  2640          STA PAD      ;trigger strobe
5F8E- AD 0D A8  2650 ACK      LDA IFR
5F91- 29 02     2660          AND #%00000010
5F93- F0 F9     2670          BEQ ACK      ;wait for ACK pulse
5F95- 68        2680          PLA
5F96- 8D 01 A8  2690          STA PAD
5F99- AD A5 5F  2700          LDA PRFL     ;make sure
5F9C- 30 03     2710          BMI EXIT     ;  prfl is
5F9E- CE A5 5F  2720          DEC PRFL     ;    down
5FA1- 60        2730 EXIT     RTS
                2740
                2750 ;   STORAGE LOCATIONS FOR FLAGS
                2760
5FA2-           2770 BRFL     .DS 1
5FA3-           2780 ESCFL    .DS 1
5FA4-           2790 NFL      .DS 1
5FA5-           2800 PRFL     .DS 1
5FA6-           2810 STRIPFL  .DS 1
5FA7-           2820 SWPFL    .DS 1
5FA8-           2830 NYBREG   .DS 1
                2840
                2850 ;END.PGM .EN      ;   END OF PROGRAM
//0000,5FA9,5FA9
```

ADDRESS DECODING, POR, and the SUPER SYM -- Part II

By Jeff Lavin - July 1983
P.O. Box 1019
Whittier, CA 90609

This month I will describe the operation of the "Super SYM" described in issue #13/14 of SYM-PHYSIS. This is not intended to be a constuction or "how-to" feature; little or no information will be given on how to modify the SYM PC board, or where to find all the hidden traces. Please DO NOT call with questions on these matters, or how to debug the completed disaster. This is not a job to be tackled by a rank beginner, nor can it be finished in an evening, or probably a weekend. On the other hand, it is not as difficult as building a 'scope kit. Read all relevant material BEFORE you start, and make an honest evaluation as to whether this project is within your abilities. Otherwise you may end up with a $239 paperweight. Synertek will not repair modified boards; if you really get stuck, call or write me at AEP for shop rates.

Begin by re-reading issue #15 to reaquaint yourself with the operation of the "stock" SYM. This is vital - if you don't understand where you've been, you won't know where you're going. The next step should be to read this article carefully until you fully understand everything involved. The SYM should be in the other room while this is going on.

The memory map of the Super SYM is modified in such a way as to make ECHO unnecessary. This is accomplished by actually having System Ram live at the top of address space: $F880-FFFF. The Monitor and other I/O have been moved up also. This was not strictly necessary for the sake of the vectors; it was done to give maximum contiguous ram, in this case 56K. All of the modifications described below were done to achieve these goals.

The first problem we run into is that the necessary decoding for the I/O at the top of memory is not present on the SYM. This is taken care of by a 74LS154 4:16 line decoder (labeled "NEW" on the schematic). This decoder gets its address range from the $\overline{F8}$ output (active low) of U11. The primary address range of $F800-FFFF is split into 16 parts of 1/2 page (128 bytes) each. The first three outputs (also active low) from the 74LS154 are used to select VIAs #1, 2 and 3 on their $\overline{CS2}$ active low chip selects. Since no further decoding is necessary, the active high chip selects are tied to +5V. Note that two chips could share the same select, as on the standard SYM, and address line A6 would be used to select either device.

Output #14 and 15 ($\overline{FF00}$ and $\overline{FF80}$) are used to select the 6532 RIOT. Both outputs are OR'd by 2 quarters of U4 (remember an AND gate acts as an OR gate for negative logic). U4 is used simply because it is available. The OR'd output (active low) selects the 6532 device in total. Meanwhile, $\overline{FF80}$ sneaks around and selects the $\overline{RS}$ (Ram Select) input. Referring to the 6532 select truth table from last issue, it is clear that when $\overline{RS}$ is low, the RAM portion of the RIOT is selected, and when $\overline{RS}$ is high, the I/OT portion is selected. Therefore, we select the RAM at $FF80 and the I/OT at $FF00. If you will refer to Listing 1, you will note that since System Ram is 128 bytes, this conveniently puts the machine vectors right where they'll do some good. We will get back to CS1 when we discuss the modified POR circuit.

The R/$\overline{W}$ line for the 6532 is write protected, but only when RAM is being addressed. This is accomplished by the following logic: Bit

0 of port A (VIA #3) is used to write protect System Ram ($\overline{WPM}$). When the SYM is reset, the VIA ports come up as inputs and float high. Resistor R83 makes certain this bit floats high. This signal is inverted and combined with an address line (normally A9) and R/$\overline{W}$ to form the R/W line for the RIOT. It is left as an exercise for the student to work out the logic. The reason for the address line is to prevent write protecting the I/O (TTY and CRT wouldn't work). The normal address of the I/O is $A400 or %1010 0100 0000 XXXX, and the normal address of the RAM is $A600 or %1010 0110 0XXX XXXX. Note that address line A9 is high for RAM and low for I/O. The new address for I/O is $FF00 or %1111 1111 0000 XXXX, and the new address for RAM is $FF80 or %1111 1111 1XXX XXXX. Note that now address line A7 is high for RAM and low for I/O. This is why we must change this input from A9 to A7. It is important that this be understood as it will be used again later.

System Ram is write protected by NACESS and un-write protected by ACCESS. ACCESS causes DDRA to be an output on bit 0 and Bit 0 to be high. When the SYM powers up System Ram is not write protected. This is why $\overline{WPM}$ can be defeated by cutting jumper 46-MM. NACESS simply makes bit 0 low.

In a similar way CA2 (VIA #1) controls the $\overline{POR}$ line. Last month we discussed how POR operates. We will not repeat the discussion here, except to add the concept of selective addressing. Normally when the SYM is reset, CA2 goes high. The processor addresses the reset vector at $FFFC, and it this combination; CA2=high AND address >= $F800 that causes $\overline{POR}$ to go low and select the ROM at the top of memory (and not select System Ram). The Reset vector points to $8B4A, and since this is below $F800, $\overline{POR}$ goes high and allows the ROM to be addressed at it's normal location. The central point here is that POR must be disabled BEFORE an address > $F800 is called. And it is; CA2 is set low just 8 instructions into the Reset routine AND VIA #1 resides below the critical address. As program execution gets underway, CA2 is set low.

Because VIA #1 now lives above $F800, this scheme will no longer serve. We have, instead, incorporated A10 into the POR circuit. This is because: The reset vector is %1111 1111 1111 110X. Note that A10 is high. The vector causes program execution (in the relocated monitor) to begin at $EB4A or $1110 1011 010X XXXX. Note that now A10 is low. Therefore, as long as POR is disabled before A10 goes high (address such as $ECXX or $1110 11XX XXXX XXXX), the ROM will stay where it is (at $E000) and System Ram will stay where it is (at $FF80).

Returning to the RIOT for a moment, it should be clear why the $\overline{POR}$ line is connected to CS1. On the standard SYM, the inclusion of $\overline{POR}$ in the address decoding chip (U11) causes the 6532 to be addressed at $AXXX, but not at $FXXX. Since we have done away with echo, we need another way to disable System Ram when the Monitor is supplying the machine vectors. In this case, POR simply chooses between ROM and RAM - we are bank switching!!! This is also the reason POR was removed from the decoding circuit. Otherwise we could never select the ROM above $C000 when POR is active.

The only other changes to the SYM are that the Monitor socket (U20) has been re-jumpered for a 2532 EPROM, and the socket is addressed from $\overline{E0}$ and $\overline{E8}$ (jumpers 15 and 16).

I would like to add a few cautions at this point: It is a good idea to remove all the 65XX "family" chips and memory from the SYM before beginning. Use only a low voltage (and current) ohmeter to "hunt" traces (hint: they ARE all accessable without unsoldering

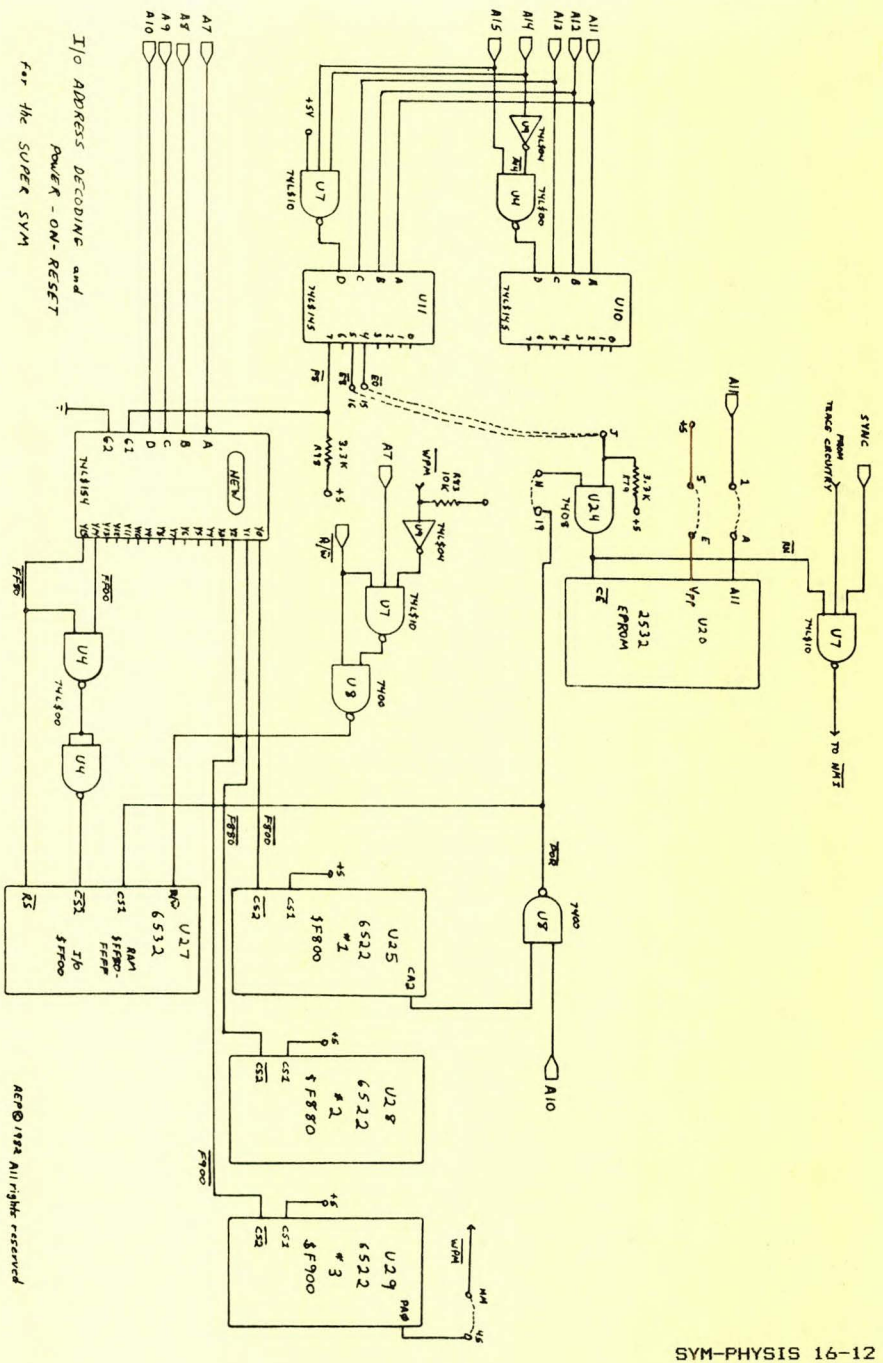
anything). Use a low wattage soldering pencil to keep from lifting traces.

Listing 1 is the new address definitions for the I/O and RAM. Since the monitor listing is available to all SYM owners (a BIG plus) all one need do is plug in the new definitions and assemble. If anyone out there hates typing, AEP will provide a free copy of the SUPERMON source with new definitions on FDC 51/4" DDEN disc ONLY! This is a limited time offer, and there will be a $10 (US) media & shipping charge.

I have asked for suggestions for future columns. I am asking again. SYM-PHYSIS is your newsletter, if you want it, you have to put in some energy. Don't assume the other guy did and I'm already swamped. I ain't! We have been working on some neat designs here and have talked to some of you about your neat designs. How about sharing them with the SYM community? We could use a good interrupt driven input and output buffer program. If you have something good, send it to Lux or myself. See you next time...

LISTING 1

```
0010  ;  >>> SOURCE CODE FOR SYM-1 MONITOR V1.1 <<<
0020
0030              .BA $E000     ;Relocated from $8000
0040
0050  ;   I/O LOCATIONS
0060
0070  TAPIN       .DE $F800     ;Formerly $A000
0080  DDRIN       .DE $F802
0090  VIAACR      .DE $F80B
0100  PCR1        .DE $F80C
0110  OR3A        .DE $F901     ;Formerly $AC01
0120  DDR3A       .DE $F903
0130  VIAPCR      .DE PCR1
0140  PADA        .DE $FF00     ;Formerly $A400
0150  DDRDIG      .DE $FF01
0160  PBDA        .DE $FF02
0170  TIMER       .DE $FF06
0180  TIM8        .DE $FF15
0190
0200  ; SYSTEM RAM LOCATIONS
0210
0220  SCPBUF      .DE $FF80     ;Formerly $A600
0230  JTABLE      .DE $FFA0
0240  TAPDEL      .DE $FFB0
0250  KMBDRY      .DE $FFB1
0260  HSBDRY      .DE $FFB2
0270  SCR3        .DE $FFB3
0280  SCR4        .DE $FFB4
0290  TAPET1      .DE $FFB5
0300  SCR6        .DE $FFB6
0310  SCRB        .DE $FFBB
0320  TAPET2      .DE $FFBC
0330  SCRD        .DE $FFBD
0340  SCRE        .DE $FFBE
0350  SCRF        .DE $FFBF
0360  DISBUF      .DE $FFC0
0370  RDIG        .DE $FFC5
0380
0390  ; PARAMETERS
0400
0410  PARNR       .DE $FFC9
0420  P3L         .DE $FFCA
0430  P3H         .DE $FFCB
0440  P2L         .DE $FFCC
0600
0610  ; USER REGISTERS
0620
0630  PCLR        .DE $FFD9
0640  PCHR        .DE $FFDA
0650  SR          .DE $FFDB
0660  FR          .DE $FFDC
0670  AR          .DE $FFDD
0680  XR          .DE $FFDE
0690  YR          .DE $FFDF
0700
0710  ; I/O VECTORS
0720
0730  INVEC       .DE $FFE0
```

```
0450  P2H       .DE $FFCD
0460  P1L       .DE $FFCE
0470  P1H       .DE $FFCF
0480
0490  ; FLAGS
0500
0510  PADBIT    .DE $FFD0
0520  SDBYT     .DE $FFD1
0530  ERCNT     .DE $FFD2
0540  TECHO     .DE $FFD3
0550  TOUTFL    .DE $FFD4
0560  KSHFL     .DE $FFD5
0570  TV        .DE $FFD6
0580  LSTCOM    .DE $FFD7
0590  MAXRC     .DE $FFD8
```

```
0740  OUTVEC    .DE $FFE3
0750  INSVEC    .DE $FFE6
0760  URSVEC    .DE $FFE9
0770  URCVEC    .DE $FFEC
0780  SCNVEC    .DE $FFEF
0790
0800  ; TRACE, INTERRUPT VECTORS
0810
0820  EXEVEC    .DE $FFF2
0830  TRCVEC    .DE $FFF4
0840  UBRKVC    .DE $FFF6
0850  UIRQVC    .DE $FFF8
0860  NMIVEC    .DE $FFFA
0870  RSTVEC    .DE $FFFC
0880  IRQVEC    .DE $FFFE
```

THE VIC-1541 DISK DRIVE

The VIC-1541 Single Drive Floppy Disk System, is available at around $300 US, complete with built-in power supply and controller (actually the entire DOS). The VIC-1541 is what might be called a "free-standing" system. Such systems are interfaced to their host computers via one of the "standard" communications links, e.g., RS-232-C, IEEE-488, or, in the case of the -1541, a simplified (serialized) version of IEEE-488.

All that is required to interface such systems to the SYM-1 is an understanding of the communications protocol, a two-way software driver, and a cable from one of the SYM's VIAs.

The -1541 is particularly attractive because of its very low cost, very modest memory requirements (simple driver and a data buffer area), and its widespread compatibility (SYM, KIM, AIM, as well as the CBM machines). While BASIC programs are not transportable, their ASCII printouts are readble by all systems, as are binary (hex) files. Most important of all, however, is the fact that RAE and MAE source files are 99% interchangeable.

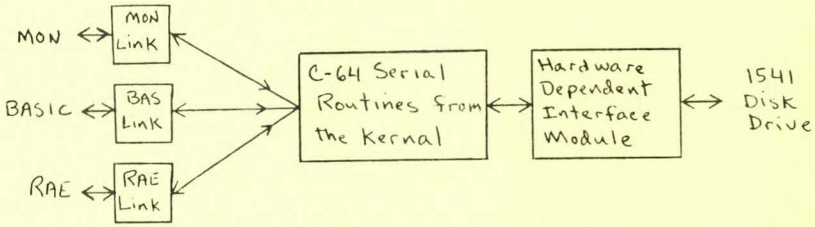
We hope to publish full information on the interfacing in Issue No. 17. Fortunately, as the letter below indicates, we are getting excellent help from others in reaching this objective. We sent Ron Jordan a listing of our VIC=20 Kernal disassembly, and he is continuing with the project, while we are working on the newsletter!

Dr. Ronald A. Jordan
2611 Madrono Drive
Ann Arbor, MI 48103
July 12, 1983

Dear Lux;

I would like to take a few minutes to describe a project which I am working on called the 6502 - 1541 Link. As you know the Commodore 1541 disk drive is relatively inexpensive and could provide a cheap disk system for the SYM, AIM, and other 6502 related computers. It might also enable the C-64 or Vic 20 to exchange information with the SYM. The drives are intelligent in that they contain the DOS. To communicate with the drive the C-64 sends information over a serial bus. From a hardware standpoint, the task of interfacing the 1541 to a VIA port of the SYM is very easy. However, the software required for communication is much more complex. To make the SYM work with the 1541 it is necessary first to know how the C-64 or the Vic 20 communicates with the drive. Presently, I have completely disassembled the C-64 operating system (Kernal) using Dessaintes' Disassembler, and have added labels and a few comments. This approach has made it much easier to track down the important serial routines for the disk drive.

To facilitate the completion of this project, Don Lewis and myself have decided to collaborate. His main interest is with an AIM-1541 Link where as I am interested in the SYM-1541 Link. We both agree on the overall design of the 6502 - 1541 software interface although many of the details must be worked out. A simple block diagram of the interface software might look as follows:



The plan is to utilize as much of the C-64 software as possible. Communication of the 6502 computer with the 1541 drive will be through a VIA port and will require hardware dependent interface software. The C-64 Kernal routines to be used are mainly for timing and formating the data that will be sent over the serial bus. To link MON, BASIC, and RAE to the C-64 routines, a software command protocol is required. It could be similar to the method used for the FDC-1 software, at least in theory. The final result will be several integrated modules which maybe easily modified, customized for a specific computer, or enhanced in the future. If the disk routines are basically the same as those used in the C-64, the disk file format will be the same. Therefore it will be possible to transfer Hex files between systems. Although BASIC files will not be interchangeable, it maybe possible to pass RAE files to MAE and vice versa. At present we are concentrating on the serial routines, the interface module, and MON link. When this phase is completed links to BASIC and RAE can be developed, possibly by others interested in getting involved in the project. Maybe there are other people as excited about this 6502 - 1541 link as I am.

Sincerely,

MORE ON THE VIC-1541 DRIVE INTERFACE

Here's more on the same subject from NICK VRTIS:

August 26, 1983
Nick Vrtis
5863 Pinetree S.E.
Kentwood, MI 49508
616-455-7594

Dear Lux,

I guess that it is about time I wrote. I have been trying to get to this letter since the last issue of SYM-PHYSIS. I wonder if I will ever catch up on the things I want to do (I hope never).

I have been busy lately. Most of the time I have been working on a version of Forth for the VIC (and the C64). It is being distributed by Abacus Software. I like my VIC, but if I had to learn all about a 6502 with that machine instead of my SYM, I would not know as much as I do now. It is just not conducive to writing and debuging code the way the SYM is. I never realized how handy the Debug Key was until I hung up the VIC for the first time.

I have also been working on a version of Dale Holts TECO for the VIC. I guess that it's about 3/4 done. It is tough to figure out how to map a 'standard' ASCII keyboard onto the VIC keyboard. I have chosen to use the Commodore key as the equivalent of the Control key (so a Commodore+H translates to an ASCII back-space). I still haven't figured out a good key to use for the Escape. I have gone through the Pound, the Shifted Pound, the Run/Stop and the Shifted Run/Stop. I may end up with the Function keys, but I have sort of been reserving those so I could set it up to 'program' those with a series of commands.

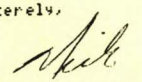
Finally, attached to the bottom of this letter are the comments from my version of a SYM interface to the Commodore 1541 disk drive. I have not had an opportunity to test it extensively, as I have a bad disk drive and am waiting for a replacement (due in the middle of September). So far, what I have tested works fine. I have tested all the functions, but have not exercised them very much. The basic SYM monitor extensions and disk I/O takes up ALL of 2k. The RAE stuff is still less than 1k. Note that the RAE stuff includes a .CD type of capability! The hardware interface which I needed to build was cheap. I think the most expensive part was the DIN connector. All you need is a 7416 and some 1k resistors. I have not really tested having both the VIC and the SYM plused into the Drive at the same time. I have gotten the SYM to act as a serial device to the VIC without the disk, but I haven't gotten the SYM to talk to the drive with the VIC attached. With the price reduction on the 1541 drives by Commodore, it isn't too bad a deal. I only wish that Synertek had Vectored the tape I/O. That would make it a lot easier to interface a disk to.

I learned a lot about RAE in working on the disk interface to it. There are some things which are not obvious about the GET and PUT vectors. The only thing really vectored is the I/O portion of the routines. Also not obvious is that they get called twice for each file, once for the header, and once for the data part. I have not checked it out, but is the write of the relocatable object vectored through the PUT also? If so, how do I tell that that's how I got there? I haven't used OUT since I got my 32k, but for completeness, it would be nice to know. It would also be nice to be able to write relocatable code out, so I could load it to the VIC.

Well, I guess I have run on enough, I still have to extract and edit the comments portion of this letter, so it will be a couple of days before it gets mailed at the rate things are getting done here. On a personal level, things are going pretty good. Everybody in the family is still healthy, and the house is still standing, so I guess I have a lot to be thankful for. One final question, how much is the 65C02? It sounds super neat, they have added some very useful instructions. The only trouble is that if you use them, the code is not very transportable. You can put it on the VIC and the SYM, but not on the C64. Are they going to come out with a 6510 version?

Oh well, on that note, I'll say goodbye. Have fun. Thanks again for SYM-PHYSIS.

Sincerely,



```
0010  ; SYM EXTENSIONS - BY NICK VRTIS 6/83
0020  ; 1) ADD SUPPORT FOR THE COMMODORE 1541 DISKETTE DRIVE
0030  ; 2) SET J 5 TO COLD START/SETUP RAE
0040  ; 3) SET BASE2 VECTOR
0050  ; 4) ADD MONITOR EXTENSIONS
0060  ;    LD - LOAD FROM DISKETTE
0070  ;         P1 - DISK DEVICE #
0080  ;         P2 - RELOCATE TO ADDRESS (OPTIONAL)
0090  ;    SD - SAVE TO DISKETTE
0100  ;         P1 - DISK DEVICE #
0110  ;         P2 - STARTING ADDRESS
0120  ;         P3 - ENDING ADDRESS
0130  ;    SC - SEND DISKETTE COMMAND
```



```
0140  ;         P1 - DISK DEVICE #
0150  ;    SR - CHECK & DISPLAY STATUS RETURN FROM DISKETTE
0160  ;         P1 - DISK DEVICE #
0170  ;    U0 - RELOCATE (P1=FROM P2=TO P3=START AT)
0180  ;    U2 - MINI LISTER (P1=START AT P2=GO UNTIL)
0190  ;    U4 - MEMORY SEARCH
0200  ;         0 PARMS - FROM 'CURAD+1' TO $8000
0210  ;         1 PARMS - FROM P1 TO $8000
0220  ;         2 PARMS - FROM P1 TO P2
0230  ;         3 PARMS - FOR P1 FROM P1 TO P2
0240  ;    U5 - DISPLAY ALPHA MEMORY
0250  ;         0 PARMS - 1 LINE FROM 'CURAD'
0260  ;         1 PARMS - 1 LINE FROM P1
0270  ;         2 PARMS - FROM P1 TO P2
0280  ;    L2 - RELOCATE LOAD STARTING AT P2
0290  ;    U6 - SETUP USER TRACE (TURNS OFF BASE2)
0300  ;         Y-X-A-FLAGS-STACK
0310  ;         A626 - INCLUSIVE STARTING ADR ($0000)
0320  ;         A62C - EXCLUSIVE ENDING ADR ($C800)
0330  ;
0340  ; NOTE: LD, SD, AND SC WILL PROMPT FOR ADDITIONAL INFORMATION
0350  ;       WITH A '>' AFTER THE COMMAND IS ENTERED. AN ESCAPE ($1B)
0360  ;       WILL ALLOW YOU TO RE-ENTER THE INFORMATION.
0370  ;
0380  ; DISK I/O ROUTINES TO INTERFACE A SYM-1 WITH COMMODORE 1541 DISKS
0390  ; AND WITH A VIC OR C64.
0400  ; THESE ROUTINES ARE ADAPTATIONS OF THE VIC-20 ROUTINES.
0410  ; NICK VRTIS - STARTED 5/83
0420  ;
0430  ; MAJOR CHANGES:
0440  ; 1) SOME P.Z. DATA MOVED TO SYM SYSTEM RAM
0450  ; 2) TIMING LOOPS CHANGED WHERE NECESSARY BECAUSE OF 1MEGG CLOCK
0460  ; 3) THE USER INSTALLED 6522 ON THE SYM IS USED FOR I/O
0470  ; 4) 6522 PIN USAGE CHANGED FROM THE VIC ASSIGNMENTS
0480  ; BIT 7 - CLOCK IN - AA-10 ORANGE
0490  ; BIT 6 - DATA  IN - AA-M  BLUE
0500  ; BIT 1 - ATN  OUT - AA-3  BROWN
0510  ; BIT 0 - ATN   IN - AA-D  GREED
0520  ; CB2   - DATA OUT - AA-5  RED
0530  ; CA2   - CLOCK OUT- AA-4  YELLOW
0540  ;
0550  ; STATUS BIT USAGE
0560  ; BIT 7 - $80 DEVICE NOT PRESENT (STD)
0570  ; BIT 6 - $40 EOI (STD)
0580  ; BIT 5 - $20 INVALID ATN (NEW)
0590  ; BIT 4 - $10 UNEXPECTED CTL CHR (NEW)
0600  ; BIT 3 - $08 UNEXPECTED SA (NEW)
0610  ; BIT 2 - $04 UNEXPECTED DEV (NEW)
0620  ; BIT 1 - $02 READ TIMEOUT (STD)
0630  ; BIT 0 - $01 SEND TIMEOUT (STD)
0640  ;
0650  ; NOTES:
0660  ; 1) ATN, DATA AND CLK ARE BI-DIRECTIONAL OPEN COLLECTOR LOGIC
0670  ;    THE CONTROLLER MUST BE HIGH TO ALLOW OTHERS TO GO LOW.
0680  ; 2) OUTPUTS GO THROUGH AN INVERTOR, SO A 0 COMES OUT A 1.
0690  ;
0700  VIC.DEV    .DE 15       THIS DEFINES THE DEVICE # OF THE VIC
0710  ;
0720  ; SRL.SAVE  P1-DEVICE NUMBER
0730  ;           P2-STARTING ADDR
0740  ;           P3-ENDING ADDR+1 (SAME AS CASSETTE)
0750  ;           NAME IS @ FNAME (MAX=30 BYTES)
0760  ;           NAME LENGTH @ FN.LNG
0770  ;
0780  ; SRL.LOAD  P1-DEVICE NUMBER
```

```
0790 ;           P2-RELOCATE TO ADR (IF NEEDED)
0800 ;           P3-RELOCATE FLAG (NON-0=RELOCATE)
0810 ;           NAME IS @ FNAME (MAX=30 BYTES)
0820 ;           NAME LENGTH @ FN.LNG
0830 ;
0840 ; RECEIVE A LINE INTO (CURAD)
0850 ; UNTIL 1) A C/R
0860 ;       2) EOI
0870 ;       3) FN.LNG CHRS REVEIVED
0880 ; FL.LNG RETURNED WITH # CHRS RECEIVED
0890 ;
```

```
0010 ; RAE COMMODORE 1541 DISK INTERFACE ROUTINES
0020 ; FILES ARE INTERFACED THROUGH THE GET/PUT ROUTINES.
0030 ; THE FORMAT IS
0040 ; GET.XXXXXXXXXX
0050 ; GET.XXXXXXXXXX A
0060 ; PUT.XXXXXXXXXX
0070 ; PUT.XXXXXXXXXX N1
0080 ; PUT.XXXXXXXXXX N1 N2
0090 ; .CT TO DISK IS IMPLEMENTED AS FOLLOWS:
0100 ; .CT XXXXXXXXXX
0110 ; THE .CT MUST BE ON THE LAST LINE OF EACH SOURCE FILE
0120 ; XXXXXXXXXX IS THE DISK NAME (NO SPACES).
0130 ;
0140 ; THE DC COMMAND IS USED FOR DISK COMMANDS AND CONTROL.
0150 ; DC #N -ESTABLISHES THE DISK DRIVE # AS N
0160 ; DC \  -GETS AND DISPLAYS THE DISK STATUS FROM THE LAST OPERATION
0170 ; DC ?  -DISPLAYS THE DIRECTORY OF THE DISK
0180 ; DC ,ZZZZZZZZZZ -TRANSMITS THE COMMAND ZZZZZZZZZZ TO THE DISK
0190 ; ZZZZZZZZZZ MAY BE UP TO 30 CHARACTERS LONG.
0200 ;
0210 ; NOTE THAT TAPE OPERATIONS STILL WORK PROPERLY.  TAPE IS ENABLED
0220 ; IF THE DISK DRIVE # IS ZERO (SET BY DC #0).  THE TAPE DRIVES ARE
0230 ; TURNED ON/OFF BY RAE BEFORE THE EXITS GET CALLED, SO 'PUT' HAS
0240 ; ABOUT A 4 SECOND DELAY BUILT IN.  THE EXITS TURN THE TAPE OFF FOR
0250 ; DISK OPERATIONS.
0260 ;
0270 ; NEW ERROR CODES:
0280 ; 31 - TEXT FILE OVERFLOW (BUT THE LOAD WAS DONE)
0290 ; 32 - INVALID DEVICE # (USUALLY 0)
0300 ; 33 - DISK I/O ERROR (ST IS OUTPUT BEFORE THIS)
0310 ; 34 - INVALID FILE NAME/COMMAND (USUALLY NONE GIVEN)
0320 ;
0330 ; RAE PARAMETER AREAS
0340 ;
```

THREE MORE JEFF LAVIN PROGRAMS

The first of Jeff Lavin's three programs below shows how OUTVEC may be temporarily changed by RAE to permit a subroutine called from RAE to "print" its output directly into RAE's input buffer. Jeff is using the output from his hardware clock calendar (CLK-1/S) to "date/time stamp" his RAE source/text files (see line 0000).

It would be instructive to compare this approach to date/timeing RAE files with the one presented by Dick Albers (with a software clock calendar) on pages 9:26 to 9:34 and pages 10:19 to 10:22.

Incidentally, Dick and Jeff are very close collaborators on both hardware and software developments (6809 as well as 6502), and in many cases, such as, in all probability, these programs, Dick has made significant contributions. (Dick and Jeff are a truly SYM-biotic pair, pun intended!)

The last two programs form a mutually supportive pair. As published, they permit very fast data transfer from SYM-1 to AIM-65 via direct VIA to VIA transfer. Obviously they can be modified to provide transfer in the reverse direction, or to accomodate data transfer to and from and between SYMs and VICs. [To adapt PART 2 from AIM to SYM as the "listener", just replace lines 800 through 860 with a simple RTS.]

To accomodate the COM-64, which uses a CIA (6526) instead of a VIA (6522), and hence "handshakes" differently (the 6526 does not have CA1, CA2, CB1, CB2 available), additional modifications are required.

Jeff's approach is far more versatile than the one we have been using to transfer data between SYMs by hard-wiring their cassette interfaces together. Also it is far simpler and much faster (parallel vs serial) than the approach we had been considering for VIC/VIC and VIC/SYM data transfer via an RS-232-C (actually inverted TTL) interface.

Actually VIC/VIC data transfer is most easily accomplished by physical transfer of floppy disks (if both have drives), since the data must be stored on a diskette if it is ever required for future use. We still use the "hard-wired cassette" method of transfering files between those of our SYMs which do not share a common disk system (one of our SYMs does have both FODS and FDC-1 installed!), but will examine Jeff's programs for possible SYM-VIC object code transfer. Of course, when the SYM to 1541 interface is available, we'll just use that instead, to get permanent data storage at the same time.

```
0000 ; 21:36:15 SUNDAY JUL 10 1983
0010
0020 ; This program will enter a line of text into
0030 ; RAEs input buffer, and from there into the
0040 ; Text File.  Each line must be < 72 chars &
0050 ; chars must not have bit 7 set.  Line must
0060 ; start with a number (0 to 9) or RAE will
0070 ; assume the returned input is a command.
0080 ; Commands may, of course, also be used, but
0090 ; if invalid will generate an error message.
0100 ; When used in this way, the operation is
0110 ; analogous to RIN (from the monitor).
0120
0130 ADDPAD     .DE $11A
0140 ; #of chars in RAEs input buffer,
0150 ; including line #, +1.
0160
0170 BUFFER     .DE $135
0180 ; RAEs input buffer.
0190
0200 TOUT       .DE $8AA0
0210 ACCESS     .DE $8B86
0220 TECHO      .DE $A653
0230 OUTVEC     .DE $A663
0240
0250 TXEN/DL    .DE $B264
0260 ; Calls CRTI which processes special chars
0270 ; (such as BS), and returns with # of chars
0280 ; in ADDPAD.  Then decides if line is text
0290 ; or a command.  If text, falls through to
0300 ; XX.. at $B275, which inserts line and
0310 ; returns to command level.
0320
0330 OUT.CRLF   .DE $E3CA
0340 ; Saves regs and prints CRLF.
0350
0360 TIME       .DE $F668
0370 ; AEP Real-Time Clock/Calendar program.
0380 ; Returns what you see here in line #0.
```

```
                        0390
                        0400            .BA $1000     ;or wherever
                        0410            .LS
                        0420
1000- 20 86 8B  0430 JTIME      JSR ACCESS
1003- A9 30     0440            LDA #'0       ;1 digit line #
1005- 8D 35 01  0450            STA BUFFER
1008- A9 3B     0460            LDA #';       ;Make a comment
100A- 8D 36 01  0470            STA BUFFER+1
100D- A9 20     0480            LDA #'        ;add space for clarity
100F- 8D 37 01  0490            STA BUFFER+2
1012- A9 03     0500            LDA #3        ;3 chars so far
1014- 8D 1A 01  0510            STA ADDPAD
1017- A9 3C     0520            LDA #L,TIME>BUF
1019- 8D 64 A6  0530            STA OUTVEC+1 :Point OUTVEC into buffer
101C- A9 10     0540            LDA #H,TIME>BUF
101E- 8D 65 A6  0550            STA OUTVEC+2
1021- 20 68 F6  0560            JSR TIME      ;Output TIME and DATE
1024- A9 A0     0570            LDA #L,TOUT
1026- 8D 64 A6  0580            STA OUTVEC+1
1029- A9 8A     0590            LDA #H,TOUT   ;Restore OUTVEC
102B- 8D 65 A6  0600            STA OUTVEC+2
102E- EE 1A 01  0610            INC ADDPAD    ;Add 1
1031- 20 CA E3  0620            JSR OUT.CRLF
1034- A9 00     0630            LDA #0
1036- 8D 53 A6  0640            STA TECHO
1039- 4C 67 B2  0650            JMP TXEN/DL+3 :Let RAE process
                        0660
103C- AC 1A 01  0670 TIME>BUF   LDY ADDPAD    ;Get current
103F- 29 7F     0680            AND #$7F      ;Strip bit 7
1041- C9 0A     0690            CMP #$0A      ;Ignore line feeds
1043- F0 0A     0700            BEQ NXTBUF
1045- C9 0D     0710            CMP #$0D      ; and carriage returns
1047- F0 06     0720            BEQ NXTBUF
1049- 99 35 01  0730            STA BUFFER,Y :Put in buffer
104C- EE 1A 01  0740            INC ADDPAD    ;Ready for next
104F- 60        0750 NXTBUF     RTS
                        0760
                        0770            .EN
```

```
                        0010 ; PARALLEL COMMUNICATION PROGRAM
                        0020 ;        PART 1 - For SYM
                        0030 ;  by Jeff Lavin  -  June 1983
                        0040
                        0050 ; VIAs connected as follows:
                        0060
                        0070 ;      PA0.S   >   PA0.R
                        0080 ;      PA1.S   >   PA1.R
                        0090 ;      PA2.S   >   PA2.R
                        0100 ;      PA3.S   >   PA3.R
                        0110 ;      PA4.S   >   PA4.R
                        0120 ;      PA5.S   >   PA5.R
                        0130 ;      PA6.S   >   PA6.R
                        0140 ;      PA7.S   >   PA7.R
                        0150 ;      CA1.S   >   CA2.R
                        0160 ;      CA2.S   >   CA1.R
                        0170 ;      GROUND  >  GROUND
                        0180
                        0190 ; Program is completely relocatable!
                        0200 ; To use, enter SAD of DATA to be sent
                        0210 ; to location FROM.  Then enter ESCAPE
                        0220 ; and $04 to end of data.  .G to INIT.
                        0230
                        0240            .BA $10
                        0250            .LS
```



```
                        0260
0010-                   0270 FROM       .DS 2         ;SAD of DATA to be sent
0012-                   0280 EOTFLG     .DS 1         ;80=Esc was last char
                        0290
                        0300 VIA        .DE $A880
                        0310 ORA        .DE VIA+1     ;TRANSMIT PORT
                        0320 DDRA       .DE VIA+3
                        0330 PCR        .DE VIA+12
                        0340 IFR        .DE VIA+13
                        0350
                        0360 ESC        .DE $1B       ;These 2 chars MUST be added
                        0370 EOT        .DE $04       ; to end of text
                        0380
                        0390            .BA $1000
                        0400
1000- A0 00     0410 INIT       LDY #0        ;Clear garbage
1002- 8C 81 A8  0420            STY ORA
1005- 84 12     0430            STY *EOTFLG :Clear flags
1007- 88        0440            DEY
1008- 8C 83 A8  0450            STY DDRA      ;All outputs
100B- A0 0A     0460            LDY #%00001010
100D- 8C 8C A8  0470            STY PCR       ;Pulse output
                0480 ;                         CA1 negative edge
                0490
1010- A0 00     0500 SEND       LDY #0
1012- A9 02     0510 SEND.LP    LDA #%00000010
1014- 2C 8D A8  0520 WAIT       BIT IFR
1017- F0 FB     0530            BEQ WAIT      ;Wait for AIM ready
1019- B1 10     0540            LDA (FROM),Y
101B- 8D 81 A8  0550            STA ORA
101E- E6 10     0560            INC *FROM
1020- D0 02     0570            BNE ESC.CK
1022- E6 11     0580            INC *FROM+1
1024- C9 1B     0590 ESC.CK     CMP #ESC
1026- D0 09     0600            BNE EOT.CK
1028- A5 12     0610 TOGL       LDA *EOTFLG
102A- 49 80     0620            EOR #$80
102C- 85 12     0630 FXFLG      STA *EOTFLG
102E- 38        0640            SEC
102F- B0 E1     0650            BCS SEND.LP   ;Always
1031- C9 04     0660 EOT.CK     CMP #EOT
1033- F0 04     0670            BEQ CHK.FLG
1035- A9 00     0680            LDA #0
1037- F0 F3     0690            BEQ FXFLG
1039- 24 12     0700 CHK.FLG    BIT *EOTFLG
103B- 10 D5     0710            BPL SEND.LP
103D- 60        0720            RTS
                0730
                0740            .EN
```

```
1000 A0 00 8C 81 A8 84 12 88 8C 83 A8 A0 0A 8C 8C A8,94
1010 A0 00 A9 02 2C 8D A8 F0 FB B1 10 8D 81 A8 E6 10,98
1020 D0 02 E6 11 C9 1B D0 09 A5 12 49 80 85 12 38 B0,1D
1030 E1 C9 04 F0 04 A9 00 F0 F3 24 12 10 D5 60,C6
 1CC6
```

```
                        0010 ; PARALLEL COMMUNICATION PROGRAM
                        0020 ;        PART 2 - For AIM
                        0030 ;  by Jeff Lavin  -  June 1983
                        0040
                        0050 ; VIAs connected as follows:
                        0060
```

```
                0070 ;      PA0.S   >   PA0.R
                0080 ;      PA1.S   >   PA1.R
                0090 ;      PA2.S   >   PA2.R
                0100 ;      PA3.S   >   PA3.R
                0110 ;      PA4.S   >   PA4.R
                0120 ;      PA5.S   >   PA5.R
                0130 ;      PA6.S   >   PA6.R
                0140 ;      PA7.S   >   PA7.R
                0150 ;      CA1.S   >   CA2.R
                0160 ;      CA2.S   >   CA1.R
                0170 ;      GROUND  >  GROUND
                0180
                0190 ; Program is completely relocatable!
                0200 ; To use, enter SAD of DATA to be stored
                0210 ; to location FROM.  Then enter *=INIT
                0220 ; and G.
                0230
                0240            .BA $10
                0250            .LS
                0260
0010-           0270 FROM       .DS 2       ;SAD of DATA to be sent
0012-           0280 EOTFLG     .DS 1       ;80=Esc was last char
                0290
                0300 START      .DE $E182
                0310 BLANK      .DE $E83E
                0320 INALL      .DE $E993
                0330 CRLOW      .DE $EA13
                0340 WRAX       .DE $EA42
                0350
                0360 VIA        .DE $A080
                0370 ORA        .DE VIA+1   ;TRANSMIT PORT
                0380 DDRA       .DE VIA+3
                0390 PCR        .DE VIA+12
                0400 IFR        .DE VIA+13
                0410
                0420 ESC        .DE $1B     ;These 2 chars MUST be added
                0430 EOT        .DE $04     ; to end of text
                0440
                0450            .BA $200
                0460
0200- A0 00     0470 INIT       LDY #0      ;Clear garbage
0202- 8C 8C A0  0480            STY PCR
0205- 8C 81 A0  0490            STY ORA
0208- 84 12     0500            STY *EOTFLG ;Clear flags
020A- 8C 83 A0  0510            STY DDRA    ;All inputs
020D- A0 0A     0520            LDY #%00001010
020F- 8C 8C A0  0530            STY PCR     ;Pulse output
                0540 ;                       CA1 negative edge
3212- AD 81 A0  0550            LDA ORA     ;Read to gen. DATA TAKEN
                0560
0215- A0 00     0570 RECV       LDY #0
0217- A9 02     0580 RECV.LP    LDA #%00000010
0219- 2C 8D A0  0590 WAIT       BIT IFR
021C- F0 FB     0600            BEQ WAIT    ;Wait for DATA
021E- AD 81 A0  0610            LDA ORA
0221- 91 10     0620            STA (FROM),Y
0223- E6 10     0630            INC *FROM
0225- D0 02     0640            BNE ESC.CK
0227- E6 11     0650            INC *FROM+1
0229- C9 1B     0660 ESC.CK     CMP #ESC
022B- D0 09     0670            BNE EOT.CK
022D- A5 12     0680 TOGL       LDA *EOTFLG
022F- 49 80     0690            EOR #$80
0231- 85 12     0700 FXFLG      STA *EOTFLG
0233- 38        0710            SEC
```



```
0234- B0 E1     0720            BCS RECV.LP  ;Always
0236- C9 04     0730 EOT.CK     CMP #EOT
0238- F0 04     0740            BEQ CHK.FLG
023A- A9 00     0750            LDA #0
023C- F0 F3     0760            BEQ FXFLG
023E- 24 12     0770 CHK.FLG    BIT *EOTFLG
0240- 10 D5     0780            BPL RECV.LP
                0790
0242- A5 11     0800            LDA *FROM+1
0244- A6 10     0810            LDX *FROM
0246- 20 42 EA  0820            JSR WRAX     ;Print ending addr
0249- 20 3E E8  0830            JSR BLANK    ;Print space
024C- 20 93 E9  0840            JSR INALL    ;Wait for input
024F- 20 13 EA  0850            JSR CRLOW    ;Scroll display
0252- 4C 82 E1  0860            JMP START    ;Go to Mon warm
                0870
                0880            .EN
```

```
0200 A0 00 8C 8C A0 8C 81 A0 84 12 8C 83 A0 A0 0A 8C,80
0210 8C A0 AD 81 A0 A0 00 A9 02 2C 8D A0 F0 FB AD 81,37
0220 A0 91 10 E6 10 D0 02 E6 11 C9 1B D0 09 A5 12 49,F4
0230 80 85 12 38 B0 E1 C9 04 F0 04 A9 00 F0 F3 24 12,57
0240 10 D5 A5 11 A6 10 20 42 EA 20 3E E8 20 93 E9 20,F6
0250 13 EA 4C 82 E1,A2
 27A2
```

ON SELECTING A (NEW) COMPUTER
-- --------- - ----- ---------

First come the "reasons" (which need not necessarily be rational!), then come the budgetary considerations. Our main purpose in buying our first microcomputer, back in 1977, was to learn how they worked. The choice, at that time, was simple: either one of the 8080/S-100 systems, one of the 6800 types, or the KIM-1. The Apple II, PET, and TRS-80 had not yet appeared on the scene.

It seemed to us that the KIM-1 would meet our requirements at the lowest possible price, so that was the route we chose, in spite of being "warned" that there was more software and hardware support available for the 8080/S-100 systems. The hardware argument made little sense, except for possible RAM expansion, since the most needed hardware add-ons for the S-100 systems, such as cassette interface, serial interface, parallel interface, etc., were already built-in on the KIM-1. All that was needed to add on was a power supply, cassette recorder, and (later) an old TTY.

When the time came to get a second system, sometime in 1979, the SYM-1 had become the "best buy", and it remained so for nearly four years, in spite of the ever increasing competition, especially as a "learning tool".

The situation has changed dramatically within the past year, and will, in all probability, continue to do so, from this point on. The Timex ZX-81 started the new trend, and the VIC-20 and COM-64 accelerated it. Today, either of these latter two is more cost effective than the SYM-1, and, with the addition of a Monitor ROM or Cassette to permit direct machine language entry, disassembly, etc., the inner workings of a very impressive internal operating system, the "KERNAL", are wide open to study and to learn much from.

While the source codes for the Microsoft BASIC and KERNAL ROMs are not published, the Reference Guides for the two systems provide enough information on the memory map and subroutine entry points to make analysis of a disassembly listing (the ML Monitor includes a simple disassembler) not too difficult. Thus the VIC-20 and COM-64 provide the

same learning potential as does the SYM-1, and both are as easily expandable, from the I/O standpoint. The VIC has a pair of 6522 VIAs and the "64" has a pair of 6526 CIAs (Complex Interface Adaptors).

By now you must have gathered that we are recommending the VIC=20 or the COM-64 as the "beginner's" entry-level system over the SYM-1. But what does this mean for old-time SYMmers? We have customized and personalized our SYM-1 systems to meet our own individualized needs (an ever on-going process), and we are as comfortable with them as with an old pair of shoes or a wife of long standing. We fully intend to keep our SYM-1s as our main systems, especially for word processing (we're so comfortable with SWP). We will be teaching a microprocessor course built around the -20/-64 systems so that we are studying their hardware configurations and operating systems for teaching purposes. While doing this we see how they (and their peripherals) can be used as peripherals for the SYM-1, and vice versa.

Examples: The COM-64 can be used as a 40 column (with color graphics as a "free" bonus) terminal for the SYM-1 in place of the KTM-2. The KTM-2/80 can be used as an 80 column terminal for the -20 and the -64, on their (inverted TTL-level) RS-232-C ports. Either the -20 or the -64 can be used as a color graphics (output) terminal for the SYM-1 (for our hobby of video recording). Most exciting of all would be the use of the VIC-1541 disk drives with the SYM-1.

Another point of compatibility between SYM-1 and the COM-64 is the availability of MAE (Macro Assembler Editor) and SWP (Simplified Word Processor) for the -64. These are first cousins to RAE and SWP, so the adaptation of the -1541 drives to the SYM-1 will permit either the SYM-1 or the -64 to be used as a development system for the other. We hope to be able to report progress along these lines in Issue No. 17.

We understand the prices of both machines in the PAL versions are still quite high overseas, but in the United States the -20 is below $90, and the -64 is below $200. Thus both are less expensive than the SYM-1, with far greater versatility. This would suggest that overseas users might consider buying the NTSC version and an NTSC monitor or TV, either color or B/W, and a voltage stepdown transformer (220 or 240 VAC to 117 VAC).

We know that the duty on colour TVs is high in some countries (when we were in Australia last year the newspapers were covering the sad story of a government official accused of bringing in, but not declaring, a colour TV set). Since an NTSC TV cannot be used as a TV receiver in PAL/SECAM countries, perhaps it would carry a lower duty???

A word of caution is in order here: Buying a COM-64 by mail-order may be troublesome, since the infant mortality rate (or incidence of failure to operate right-out-of-the-box) seems to be unusually high, based on our own experience, and talking with both other purchasers and several discount house dealers. It might be "safer" to have someone buy and "burn-in" the unit for you; then failure to work after shipment would most likely only be a connector or chip vibrated loose.

ON "OLD" COMPUTERS

Now that we have grown to love the VIC=20 and the COM-64, and recommend them as "entry-level" systems for beginners, provided only that the ML Monitor, on Cartridge, Cassette, or Diskette, be among the first "add-ons", the question naturally would arise: "But, what about SYM?".

When we switched from KIM to SYM, the reason was to have TWO systems, one for student use at school, the other for personal use at home. Obviously software/hardware/cassette interface compatibility made the SYM/KIM combination a "natural". The KIM (actually two of them) was

long ago retired to the shelf, mostly because it was not as versatile as the SYM, and would not directly take the KTM-2 terminal, requiring a Current Loop to EIA (RS-232-C) interface.

Our SYMs will not be retired. Each of our many systems was "customized" for a particular application, and will continue to be used as long as the need for those applications continues. Some of those systems will be given to students who can put them to use, and will give them good homes.

Our two major reasons for getting the VIC=20 and the COM-64 were, first, to learn how they worked, and how to use them, so that we could build a course around them, and second, to use them as peripherals for the SYM, e.g., as color graphic terminals (output only). During our learning process we found that the VIC-1541 Disk Drive would also be an excellent peripheral for the SYM.

Also, since so much software is becoming available for the Commodore machines, e.g., MAE, spread sheets, word processors, etc., that would be more effective with an 80 column terminal, we'd like to add a KTM-2/80 on the CBM RS-232-C (actually at TTL levels) port for this purpose, as output only. The Commodore would still be used for input, because of its inherent full-screen editing capability.

Thus, we're not abandoning SYM; it's just that as our needs expand into areas where little commercially available software exists for the SYM, it's more time and cost effective to get new hardware that will run the software we want, especially, when both the hardware and software are so inexpensive. There is so much "public domain" and published software available for the Commodore machines through Users's Groups that, when you "average" the cost of purchased software with the free software, the cost per program is ridiculously low.

Our Commodores are supplementing our SYMs, not replacing them!

DOSES FOR THE SYM

We have SYMs with three different DOSes (CODOS, FODS, and FDC-1), each with (naturally) its own syntax and "personality", and our own personality seems to "split" when we shift from one DOS to another on a rapidly rotating basis.

Actually, there are five DOSes available for the SYM, since two very elegant DOSes have been devloped for use with the HDE (FODS) Disk Controller. These are the UK-SYMmers DOS, and RAE.DOS, developed by Ralph Deane and Jack Brown. The latter works only with RAE-1 (but NOT with BASIC), and provides the most sophisticated microcomputer software development environment we have ever seen. RAE has been expanded to include a completely integrated DOS, as well as enhanced editing features (a la Dean and Brown's earlier X-RAY).

[In thinking about this matter today, the thought occcured to us that our situation had something in common with polygamy, in that each of the systems makes its own set of demands on us, and each gives us satisfaction and pleasure in its own way. The closest we can come to describing our situation is polySYMmetry, but this might imply that the other systems are more like concubines than wives.]

[We also have around a number of non-DOS systems. These include two KIM-1s and an AIM-65, which we put on the shelf to avoid further confusion, a SYM-69 (6809-based), a Sinclair ZX-81 (Z-80-based), an RCA COSMAC-VIP (1802-based) and an SDK-85 (8085-based). These latter we leave alone because we seem to have become 6502-based ourself.]

We now have a COM-64 and a VIC=20 (fully expanded to 40K RAM), both with CBM DOS V2.6 (1541), and soon, an Apple II-E,.with its own variety of DOSes. It should be noted that the CBM BASIC V2 (by Microsoft) in both the -20 and the -64 is rather clumsily interfaced to the DOS, compared to the Version 4 BASIC, and a variety of "WEDGES" to improve the interface is available, including one for the -64 called DOS 5.1; there is so much to learn!

We'll play with the Apple II-E for awhile, when we get it, to see how much we still remember about running the Apple II, and how the II-E differs from the II. Denny Hall, and the other Apple II owners we know, don't really care for the changes in the II-E; but we think we can compare the two Apples more impartially.

We very much like both the VIC=20 and the COM-64, especially on the basis of their low price, and think that these are the "wave of the future" (Commodore says that the VIC=20 is the most widely sold computer). We're spending many happy, frustrating, hours with the VIC and the -64, getting to know and love them, in order to help beginneers (schoolteachers, and young students, in particular) learn to do the same.

Mostly, though, we also like the low price of the VIC-1541 Disk Drive, and would very much like to be able to add, very inexpensively, CBM DOS V2.6 to the SYM-1 (see elsewhere in this issue for further info)! Concomitant with the low price of the 1541 Single Drive System, unfortunately, is its slowness, by a factor of at least eight, as compared with the more expensive Dual Drive Systems available for other CBM Computers. This is because the -1541 Drives use a serial hardware implementation of the (full parallel) IEEE-488 interface.

RAM VS ROM IN DISK-BASED SYSTEMS

We have long favored our FODS system over our CODOS system for program development work because the FODS required only 8K of RAM ($6000-$7FFF), while the CODOS required 16K ($4000-$7FFF). We were receiving a number of CODOS disks from Jack Brown, "Sandy" Mackay, Lee Longstreet, Jack Gieryic, and others, which we could not use, because they had reconfigured their systems. They reasoned, and correctly so, that with a disk system the only ROM requirement is for a BOOT program. [The BOOT program could be in a POWER ON RESET ROM so that not even SUPERMON need be resident in ROM, but this would lead to system non-compatibility.]

So, we followed their lead, and replaced the BAS-1 and RAE-1 ROMs from $B000-$EFFF with RAM (also added 6116 2K RAMs at $9000, $9800, and $F000). Since RAE-1 does not require the co-residency of BAS-1, and FORTH does not need either BASIC or RAE resident, we run these from a special version of CODOS (supplied on the original distribution disk) which resides in two segments, at $6000-$7FFF and $C000-$DFFF (vacated by BASIC!). Eventually we'll relocate BAS-1 to occupy $B000-$BFFF and $E000-$EFFF (vacated by RAE!) so that it, too, can operate with the High RAM version of CODOS.

Having pulled CODOS out of $4000-$5FFF, we relocated the visible memory from $2000-$3FFF, where it was "clobbered" by Jack Brown's CODOS BASIC and CODOS FORTH (the best FORTH we've seen), to the higher location, well above FORTH's working dictionary. We also gave FORTH 8 screen buffers in the 8K+ block from $B000-$D0A0.

Nearly two years ago Jack Brown sent us (as a Christmas gift), a cassette-based FORTH with a beautiful "POLYSHOW" graphics demonstration for the Visible Memory at $4000. Installing CODOS "bumped" the VM down to $2000, where it was essentially useless, since every useful program tracked garbage through the display. It was nice to be able to give FORTH back its fast, high resolution grahics capability, again, and with

CODOS this time. CODOS can SAVE or GET an 8K graphics screen within a second or so!

"Sandy" Mackay sent us copies of his CODX and WORDX packages for the CODOS/SYM. CODX is an enhanced RAEINTERFACE which includes X-RAY [NOTE: X-RAY is not available (as yet) in a CODOS version], and a "souped-up" version of the Gemini-10X printer driver published elswhere in this issue. WORDX is a superb word processing package which fully integrates CODOS/X-RAY/SWP2.5+/10XDRIVER. Both packages require RAM at $C000-$DFFF, however, which we now, fortunately, have.

Putting RAM at $B000-$EFFF is the wisest move one can make in any SYM/DOS system. Doing so in our CODOS system has made a wealth of great software available for the SYM. Since our CODOS system is our only SYM with a Visible Memory installed, and has our most powerful FORTH system also installed, we will be using it more and more as we get around (finally) to the image processing experiments we have long had in mind.

"UPGRADE" TO SWP 2.5

As mentioned above, "Sandy" Mackay, who wrote SWP-2.5, and who wrote the Gemini-10X Printer Driver, published elswhere in this issue, provided us with a copy of his WORDX for SYM/CODOS systems. WORDX contains several enhancements to the Gemini Printer Driver (too extensive to publish here) and one major enhancement to SWP 2.5. Our CODOS system is still using the 20 ma loop decwriter printer, since the Gemini is currently interfaced to the COM-64. Thus we have not tested the SWP 2.5 improvement. We reprint it below, however, for users of SWP:

```
3500  ;
3510  ;
3520  ;  ------------------------------------------------------------
3530  ;  Change SWP-2.5 source as follows to preserve right
3540  ;  justification when using {4italics{5 and {-1underlines{-0,
3550  ;  and to permit change in character size.
3560  ;  ------------------------------------------------------------
3570
3580
3590  ;  CHANGE ALL THREE SYMBOLIC SPACE (#'^) IN SWP CODE TO #$60
3600  ;  AND ENTER ALL SYMBOLIC SPACES AS SHIFT RETURN (SIGMA).
3610
3620  ;  Under "JUST.", after LDX #$FF, add "STX ESCFL".
3630  ;  At end, under STORAGE VARIABLES, add "ESCFL .DS 1".
3640
3650  ;  Change the code starting at INSWORD as follows:
3660
3670  INSWORD     LDA #$81      ;CODE FOR 1 SPACE
3680  LPINS       LDY CURPOS    ;GET POS. IN CRTEX
3690              BEQ INSPOS1   ;NO SPACE IN COL 1
3700              LDX CRTEX-1,Y         ;NO SPACE B4 ^
3710              CPX #$60      ;SYMBOLIC SPACE - NOTE SHIFT RETURN (SIGMA)
3720              BEQ INSPOS1
3730  LPINSX      LDY CURPOS
3740              STA CRTEX,Y   ;PUT CHAR.
3750              INC CURPOS    ;INC TO NEXT POSITION
3760  INSPOS1     LDY WDSTART   ;COMPLETED INSERT?
3770              CPY WDEND     ; *
3780              BCS EXITINS   ;BR. IF COMPLETED
3790              INC WDSTART   ;ELSE INC. TO NEXT CHAR
3800              LDA CRT,Y     ;GET NEXT CHAR.
3810              CMP #$7B      ;SHIFT ESCAPE?
3820              BNE ESC?
3830              INC ESCFL
3840              BPL NOINC     ; BR ALWAYS
```

```
3850  ESC?        LDX ESCFL
3860              BEQ NOINC
3870              CMP #'-       ;TOGGLE UNDERLINE?
3880              BNE B?
3890              BEQ TWOPARMS          ;BR ALWAYS
3900  B?          CMP #'B       ;CHANGE TYPESIZE?
3910              BNE S?
3920              BEQ TWOPARMS          ;BR ALWAYS
3930  S?          CMP #'S       ;SUB/SUPERSCRIPT?
3940              BNE W?
3950              BEQ TWOPARMS          ;BR ALWAYS
3960  W?          CMP #'W       ;DOUBLE WIDTH?
3970              BNE ONEPARM
3980  TWOPARMS    INC REMAINLEN
3990              INC PR.POS
4000  ONEPARM     INC REMAINLEN
4010              INC REMAINLEN
4020              INC PR.POS
4030              INC PR.POS
4040              DEC ESCFL
4050  NOINC       JMP LPINSX    ;LOOP
4060
4070
4080  EXITINS     LDY CURPOS    ;CKG. FOR END OF SENTENCE
4090
4100
4110  ;  (W1THAT'S IT, FOLKS!(W0
4120
4130  ; [NOTE BY LUX: The "gibberish" in lines 3540 and 4110 is
4140  ; because we printed material intended for the Gemini-10X
4150  ; printer with appropriate driver on an Epson MX-80FT with
4160  ; Graftrax Plus with its own printer driver, and the con-
4170  ; trol codes and sequences are obviously different.  Such
4180  ; are some of the problems of software "transportability"!]
```

THE END OF SYM?

Several of our readers have written to us, and also to MICRO, concerning MICRO's decision (September 1983) to abandon its support of single board systems, i.e., AIM, SYM, and KIM (the "AimSymKim", or "ASK", systems which formed its initial basis for publication.

We understand MICRO's reasons. While we are not privy to the exact figures, the number of VIC=20s out there must be approaching two million, while the number of SYMs must be way under 100,000. This is not surprising, since the price of a single BAS-1 ROM is more than the price of a complete VIC=20!

As many of you have heard, Synertek will be giving up SYM/KTM production by the end of the year, and until then, delivery schedules will be "uncertain", to say the least. We also understand Synertek's reasons. Their prices are no longer competitive.

We did not feel that the SYM-2 and KTM-3 were viable improvements over the SYM-1 and the KTM-2. Our suggestion to Synertek was to consider a package deal, with the SYM-1 with BAS-1 and RAE-1 installed (there is no reason why RAE-1/2 continued to exist; remember the two chip BAS-1 was replaced with a single chip version at the same price?) and a KTM-2/80. The selling price for the combination could have been in the $595 region, same as the original Commodore-64 price. If Commodore can provide their Microsoft BASIC at little more than the chip cost, couldn't Synertek do the same?

We even suggested that the major elements of the SYM (forget the no

longer needed hex pad and 7-segment displays) could be mounted in a corner of a redesigned KTM to make a really single board computer, much more powerful than the AIM-65, but were told that such a product would not fit into their planned product line.

At any rate, there will be no new SYMs or KTMs after the end of the year, unless someone elects to purchase the manufacturing rights (we'll keep you posted on this), as was done for the KIM-1, when Commodore decided to concentrate on the PET product line.

The Users' Group will be winding down its activities also. We will maintain a small stock of replacement (proprietary) ROMs, and fill orders for software and publications as long as there is a reasonable demand. Issue #17 will be our closing issue; our four years plus should qualify us for a Guinness record as the longest running Users's Group Newsletter.

ON UNANSWERED LETTERS

We are able to meet most of our commitments except for answering each of the letters you write personally. It frustrates us not to be able to do so, especially when there are important questions which must be left unanswered, but to do so would require finding about four hours of "free" time each and every day, an obvious impracticality.

The letters get "stacked", and we do try to get to them, sometimes FIFO, sometimes LIFO, sometimes by random selection, but the stack continues to grow. [Just today, for example, we are answering a very well thought out, but somewhat critical letter we received back in January of 1982!]

We pride ourselves in being "organized", and wish there was some way in which our many computers could help us be even more so, but the hard copy problem is one on which we are still working. We had been feeling very guilty about all this, blaming our own inadequacies, until we came across a brief essay by Andy Rooney, seen regularly on CBS's "60 Minutes" show, entitled "The Organized Man's Vacation", in "This World" (24 July 1983).

We now feel less lonely in our guilt, knowing that our situation is not unique, and that other, far more successful individuals share our problem. We reprint below a few paragraphs from Mr. Rooney's essay, by way of letting him do our explaining for us:

"Because it seemed like a good time for me to get caught up on all those odds and ends of papers I had around the office and in my bureau drawers at home, I packed up several boxes and a briefcase and brought them with me to the country on vacation.

"Naturally I didn't get at it the first day or two. Any time now though, I think I'll get started. I try to decide when to get at them every morning and several times during the day. The papers are still right there in those boxes, ready to be gone over. It'll be good to get them cleaned up so I can start fresh when I get back to work.

"Last year I brought several boxes of important papers with me to go over on vacation, too. I still have those around here somewhere. I don't know what happened last summer. I never did get at them. That's not going to happen this year, you can bet on that.

"I'm not exactly sure what's in the boxes. There are a lot of letters I ought to answer, I know that, and some bank statements, I think. . .

"What I think I'll do is save the boxes I have the papers in, and when I get at working on them I'll put them back in the boxes all carefully arranged. Then, when I go home from vacation, I can file them. Next winter I'll be able to put my finger on anthing I need."

MORE ON FORTH

We now have three excellent disk-linked FORTHs available for SYM, both fig-FORTH and 79-STANDARD versions (fig-FORTH is easily "upgradable" to the 79-STANDARD by a few simple new definitions and a few simple re-definitions). The CODOS and FODS versions are by Jack Brown, and the FDC-1 version is by Bill Wharrie. Both authors have provided a number of very useful application and utility "screens" (the FORTH source-code format) with their systems, and are developing others. Bill Wharrie sent us a number of screens which are too extensive to publish here.

We suggest that FORTH users work directly with the authors of their versions to form specialized users's groups for software interchange directly on the appropriate magnetic medium. We would go even further to suggest that the authors (Jack and Bill) continue to work together to interchange screens on cassette, since their disk formats are not compatible.

We have an excellent C-64 FORTH (Datatronic AB, Stockholm, Sweden), into which we plan to key-in many of Jack's and Bill's screens. The task is slightly complicated here since the "standard" screen format of 16 lines of 64 characters (1024 bytes) has been replaced by a screen format of 25 lines of 40 characters (1000 bytes) to better conform to the video display of the COM-64.

We never cease to be amazed at the high degree of transportability of FORTH programs. Except for a few "primitive" FORTH words, and the DOS and Terminal/Video interfacing, no other modifications are required. We particularly wish to get Michel Dessaintes DECOMPILER from our SYM-FORTH into the COM-64, to study the structure of that FORTH. [Dessaintes' DECOMPILER is to FORTH as his DISASSEMBLER is to 6502 Machine Language.] Then we will be able (someday!) to modify the Datatronic AB version to work directly with a KTM-2/80 on the RS-232-C interface with 16x64 screens. This will be particularly useful when SYM-FORTH is interfaced to the VIC-1541 Disk Drive.

Here is some information supplied by Bill Wharrie for FDC-1 FORTH users:

Lux: August 24/83

Bill Wharrie

Enclosed is some material for the next issue -- a Forth CASE statement, a 'video editor' and a correction for a bug in the U/ routine.

First the U/ bug. I have never, to my knowledge, been bitten by this bug -- it involves a carry error that only occurs under specific conditions. The correction is painless to add. I would like the correction to be published and added to all subsequent releases of FDC-Forth if possible. (i.e. included in the boot-up object code as outlined in screen 125).

The next set of screens present the CASE statement and the video editor as one package. Screens 171 - 173 provide a brief explanation of the code in screens 195-197 and 190-194.

Also included in this package is a set of screens that link in the BAS-1 floating point routines for use from Forth. This work is PRELIMINARY ONLY and suffers from two major shortcomings -- there is no provision for direct input of floating point numbers and any errors trapped by the BAS-1 routines (such as division by 0) will dump you into BAS-1 which will then crash (since page 0 is not set up). The second problem can be taken care of with error traps in the Forth words. The solution to the first problem involves

finding out how to use the INPUT routine in BAS. Execution addresses for the BAS f.p. routines published in SYM-PHYSIS were a great help in writing this code. If you here of anyone interested in persuing this, please feel free to photocopy these screens and send them off.

After working out the BAS links I felt I had created a monstrous kludge -- areas of page 0 have to be swapped out as well as the problems mentioned above -- so my thinking now is to develop a floating-point package that is integrated into Forth. I'd like to buy a copy of Huey II (and charge it against my royalties) and re-write it for use in Forth. I've included this request on a separate sheet headed ORDER.

One last VERY IMPORTANT thing. There is an ERROR in the FDC-Forth Installation Instructions. About 2/3 of the way down page 1 the list of changes for a 24K system is given. The address 08AF is wrong - it should be 08A3. Please publish this change if you think it is necessary and correct subsequent releases of the instructions.

That's about it for now -- I'll have more goodies for the next issue. I hope this isn't too late for the September issue.

By the way, FDC-Forth will work without modification with double-sided disks! Drive 0, side 1 contains screens 0 to 199, drive 1 side 1 has screens 200 to 399, drive 0 side 2 has screens 400 to 599 and drive 1 side 2 has screens 600 to 799.

Bill

NOTE TO EARLY PURCHASERS OF FDC-FORTH

The Installation Corrections for a 24K system are incorrectly given in the documentation provided. Here are the correct corrections:

0211:59 0897:4D 08A3:59 13C5:** 13D0:**

[** means subtract $20 from whatever is currently in these locations]

[EDITOR'S NOTE: Space limitations obviously prevent us from reprinting all of Bill's material. We suggest, therefore, that all interested FORTH users contact Bill directly. We are, however, reprinting his two screens on the "U/ 'BUG'" (of which we were unaware), because of their applicability to all versions of FORTH.]

```
SCR # 124
  0 ( CORRECTION FOR U/ BUG )   FORTH DEFINITIONS DECIMAL
  1
  2 CODE (U/)  0 # LDY,  N 1+ STY,
  3     BEGIN,  SEC 2+ ROL,  SEC 3 + ROL,  N 1+ ROL,
  4           SEC,  SEC 2+ LDA,  BOT SBC,  TAY,
  5           SEC 3 + LDA,  BOT 1+ SBC,  PHA,
  6           N 1+ LDA,  0 # SBC,  0 # LDA,  N 1+ STA,  PLA,
  7           CS IF,  SEC 2+ STY,  SEC 3 + STA,  THEN,
  8           SEC ROL,  SEC 1+ ROL,  N DEC,  0= UNTIL,
  9       POP  JMP,  END-CODE
 10 HEX
 11 ' U/ 16 +  DUP  4C SWAP C!  1+  ' (U/) SWAP !
 12 DECIMAL
 13 ;S
 14
 15
```

```
SCR # 125
  0                THE U/ BUG
  1 In Volume V no. 1 of Forth Dimensions a bug in the U/ code is
  2 identified and corrected. Screen 124 is a slight adaptation of
  3 the code presented. When screen 124 is loaded the new code is
  4 assembled and then the old code is patched to jump into the new.
  5 To make this correction a permanent part of your boot-up code,
  6 execute the following from the terminal after loading scr 124.
  7 HERE 28 +ORIGIN !      HERE 30 +ORIGIN !
  8 HERE FENCE !    LATEST 12 +ORIGIN !
  9 Then exit to the monitor with MON, change location A624 (the
 10 FIXBLK flag) to $FF and do a S3 FORTH 0,200,41FF. Then reset
 11 FIXBLK to zero and return to FORTH with G.
 12 The correction can be buried lower in the boot-up file but this
 13 should only be necessary if you are writing large applications
 14 where you would want to FORGET the editor and assembler before
 15 compiling the application.
```

FDC-FORTH AUGUST/83 BILL WHARRIE

SPEAKING OF FORTH!

Just today we received a CODOS format disk from Jack Brown, describing a very "intimate" RING (Users' Group), and bearing the source code for HYPER FORTH 79.

The easiest way to describe the contents of the disk, and to give at least a brief introduction to the power of HYPER FORTH, is to reprint (with Jack's permission) several brief extracts from the material thereon:

```
>DIR *.T

DEMO.T          :0 -  30-AUG-83  $0002DE
LABEL.T         :0 -  30-AUG-83  $000182
RING_LETTER.T   :0 -  30-AUG-83  $000A68

>TYPE DEMO.T

TYPE ONE OF THE JOB COMMANDS SHOWN BELOW.

DO   DEMO1.J        GENERATE HYPER FORTH IN PLACE.
                    ( ONLY ONE SET OF HEADERS )

DO   DEMO2.J        GENERATE HYPER FORTH OFFSET
                    FROM RUNTIME POSITION.
                    ( TWO SETS OF HEADERS GENERATED )

DO   DEMO3.J        GENERATE RUNTIME ONLY PACKAGE.
                    ( HEADERLESS CODE GENERATED     )
                    TOWERS OF HANOI IS USED FOR DEMO APPLICATION.

DO   FORTH.J        RUN PRE-COMPILED VERSION OF HYPER FORTH79.
                    WITH EDITORS ADDED.

REFERENCE:          METAFORTH  by  John J.  Cassady
                    available from Mountain View Press.

>TYPE LABEL.T

HYPER FORTH79 4.0 DISTRIBUTION FILES
COPYRIGHT 1983 SATURN SOFTWARE LIMITED
RING_LETTER.T     SHORT NOTE FROM JACK
VMEMS.F=  SCR# 101-272    FORTH  SOURCE
  SCR#  50- 99    HYPER COMPILER
  SCR# 273-278    VEDIT  ( LOOK!! )
  SCR# 010-017    LEDIT  ( LOOK!! )
```

NOTE


```
HYPER0200.C       H FORTH ORIGIN $0200
HYPERB100.C       H FORTH ORIGIN $B100
HYPER.C           WITH EDITORS ADDED

>TYPE RING_LETTER.T
```

SATURN RINGNEWS

June 28, 1983

(Modified August 30 for SYM-1 CODOS members)

Dear Fellows of the Ring:

This is going to be an informal short note. We are starting off small. Only six including myself. They are:

MTU-130 CODOS members
Donald Full - - - - main interest FORTH
Laughing Water - - - main interest FORTH
Bruce Carbrey - - - main interest NOT FORTH(??)
Jack Brown - - - main interest FORTH

SYM-1 with CODOS members
Jack Brown
Sandy MacKay - - - main interest ???
Marty MacieJewski - main interest ???

This issue will consist of the HYPER compiler which takes source for FORTH written in FORTH plus an application to generate a new FORTH system at a different runtime address or a runtime only application (headerless code). The documentation for this system is not complete. Study the DEMO Job files and the load screens 50 55 to see how to operate the system.

This system should not be further distributed without my written permission. You may, however use this system to generate runtime only (headerless code) for further distribution and much personal profit - we hope.

The next Issue of RING news will be available in September. In this issue we will present Laughing Water's Mailing, and General Ledger programs written in FORTH, Bruce Carbreys Floating Flash package (four significant digit floating point), and some new FORTH material from Donald Full. I plan to write a FORTH version of Bruce's Floating Flash.

For future issues we will be looking at 68000 FORTH for the datamover board.

I would also suggest that we exchange addresses and telephone numbers in the next issue. If you do not want yours released to the other ring members please let me know by next issue. Perhaps we could start exchanging files by phone. I have a Hayes smart modem and I think Bruce and Donald also have one.

Remember, to be a member of the ring, you must send a contribution on disk plus $3.00 to cover cost of mailing your disk back. At this time I am pleased with our small start and will be happy to see very slow growth in the membership of the ring.

Please note that I will not be available during July and August. The Brown family will be staying at a remote wilderness cabin on the shore of a lake 25 miles from the nearest roads, phones, power, and computers!

Best regards,

Jack Brown

PS SYM members will have the MTU related stuff replaced by SYM related STUFF and vice versa.

BOOK RECOMMENDATIONS

The following two books are worth noting, one for its extremely broad area of applicability, the other specifically for COM-64 users:

> "REAL TIME PROGRAMMING - Neglected Topics", Caxton C. Foster, Addison-Wesley Publishing Company, 1982. ($9.95 US)

Foster is the author of the earlier recommended "PROGRAMMING A MICRO-COMPUTER-6502". His work is at the level of a Leventhal or Zumchak, which is just about the highest recommendation we can give an author. Definitely MUST reading.

> "64 INTERN - Das grosse Buch zum Commodore 64", Angerhausen, et al, Data Becker GmbH, Duesseldorf, 1983. (69 Swiss Fr)

Half of this 310 page book is textual information on the COM-64 system, the remainder is fully commented (in German) source code for the PAL version of the BASIC and KERNAL ROMs. [Having had to pass German and French reading comprehension exams, ever so many years ago, as part of the Ph. D. preliminaries, we are finding this book, recommended and sent to us by Norbert Thuering, to be very helpful, indeed.]

A RELOCATED FODS

Paul Beaupre, of ECC Microwave Associates, has customized his FODS, relocating it, SUPERMON, and SYSRAM (and I/O, too, we believe) higher up in the Memory Map. Some of the details, and an offer to make copies available to interested FODS users are given in his letter, reproduced below. Following his letter is a Directory Listing for Paul's System Disk.

ECC Microware Associates
87 Francis Ave.
Newington, Ct. 06111

The disk enclosed is a modified FODS in which all the software has been relocated. Instead of residing at address $7300, it is now located at address $B300. This relocated version looks at an $E page moniter, systems RAM located at $FF80-$FFFF and the FODS controller located at $FE80.

All files which were at $6000 are now at $A000. This will now give you symmers 48K of user ram with FODS using 8K of it. %TED has been relocated for those of you who use it. %BLM has also been corrected.

Copies of the new operating system are available directly from ECC at $55 each. Please specify if it is for a single or a duel disk system. Also if you want a 35 or 40 track version. Copies are for a 5 1/4" system only!

Also available from ECC is a relocated RAE. This version is located at $C000-$DFFF. Just mail your original ROMS and a letter if you want different defaults or any other changes. The cost is $30, which includes the new EPROMS and return postage.


```
>DC DIR 2
Ø1 %FODS   B3ØØ BFFF Ø1 Ø1    Ø2 %DIR    ADØØ ADCØ Ø2 11
Ø3 %CPY    ADØØ ADE9 Ø2 13    Ø4 %DEL    ADØØ AD32 Ø2 15
Ø5 %FRE    ADØØ AEØE Ø2 16    Ø6 %*FM    ADØØ AFED Ø3 Ø3
Ø7 %REA    ADØØ ADA3 Ø3 Ø9    Ø8 %NAM    ADØØ AD7F Ø3 11
Ø9 %LDN    ADØØ AD73 Ø3 12    1Ø %FOD    ADØØ AD1E Ø3 13
11 %SOR    ADØØ AE74 Ø3 14    12 %PAK    ADØØ AEAD Ø4 Ø1
13 %BLM    ADØØ AFØ7 Ø4 Ø5    14 %NUM    ADØØ AF1C Ø4 1Ø
15 %PON    ADØØ AD49 Ø4 15    16 %POF    ADØØ AD49 Ø4 16
17 %VER    ADØØ AE14 Ø5 Ø1    18 %CYR    ADØØ ADA6 Ø5 Ø4
19 %PAC    ADØØ ADDC Ø5 Ø6    2Ø %BYE    ADØØ AD49 Ø5 Ø8
21 %RAE    ABØØ AC8Ø Ø5 Ø9    22 %RAY    ABØØ ACAØ Ø5 13
23 %ONN    AAØØ AA43 Ø6 Ø1    24 %OFF    AAØØ AA43 Ø6 Ø2
25 %BAS    AØØØ A62A Ø6 Ø3    26 %BAX    AØØØ A96F Ø6 16
27 %TED    AØØØ ABBA Ø8 Ø3    28 %LCP    AFEØ AFFF Ø9 11
29 %DIS    AØØØ A45A Ø9 12    3Ø %TOM    AØØØ A558 1Ø Ø5
31 %PUB    AØØØ A799 1Ø 16    32 %BOOT   5ØØØ 5ØD1 11 16
33 :BOOT   Ø2ØØ Ø93A 12 Ø2    34 :LINK   Ø2ØØ ØA15 13 Ø1

NEXT: T14 SØ2
```

APPLES AND SYMS

Many of our readers use their SYMs as "supplements" to other systems, and/or vice versa, down-loading data from one to the other via various ingenious interfacing methods. For some, the other computer might be one of the larger "main-frame" computers, or a mini of some type. For others it is an Apple; increasingly now, it might be the VIC=2Ø, or the COM-64.

Duncan Bailey sent us an excellent article, late last year, on his work with Apple/SYM systems. We didn't print it then, because as he indicates in his accompanying letter, which is reprinted below, his program listing was too light for satisfactory reproduction. We asked Denny Hall to get us a camera-ready listing on his Apple system, but, for some reason, he could not read Duncan's diskette.

In any event, here's Duncan's letter and we suggest that Apple/SYM owners contact him directly, perhaps sending along a blank diskette and enough to cover mailing costs, to get copies of his article (very well done, indeed), and the program listing in machine readable form.

[P.S. Our own copying machine wouldn't sufficiently "enhance" the original, very lightly printed, letter to get dark enough camera ready copy, so that we had to have our printing company use their super-"duper" (pun intended) copying equipment to do the job. Please use good fresh dark ribbons!]

Dear Lux: 609 Echo Glen, River Vale, NJ 07675 — November 24, 1982

I am enclosing a proposed article for Symphysis on my experience using an Apple II as a development system for the SYM. I am quite happy with the way it has worked, and am pleased to share my experiences with others.

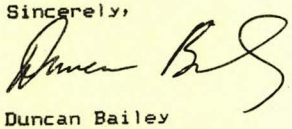
As noted in the article, I bought a SYM in order to build a home control system (I would never use a nasty old Z-80 as Steve Ciarcia does). In that regard, I would be delighted to correspond with others who are doing the same thing. Once it is all working (my wife says "if"), I plan to write up some of the interesting parts for you.

Since you complained about crummy ribbons, I have enclosed an Apple diskette which includes this letter and the article in Applewriter 1.0 format, and a copy of the

BASIC program, Appletosym.

I hope you can use this material.

Sincerely,



Duncan Bailey

DOS/65

One of the major problems affecting the 6502-based computer user community is the lack of a "universal" DOS, such as CP/M, for Z-80/80XX systems, or FLEX, for 68XX systems.

The Cedar Valley Computer Association (P.O. Box 671, Marion, Iowa, 52302), one of the largest AIM-65 Users' Groups of which we know, publishes an excellent newsletter, which we receive regularly on an exchange basis.

We reprint the following extract from the July/September 1983 issue of their newsletter, for its general interest:

> DOS/65 DISK OPERATING SYSTEM
>
> We are supporting two disk operating systems for the club Floppy Disk project. The simpler one, EL DOS, will be available at the cost of distribution, between $5 and $10. We are looking at a CP/M look-alike as a high-performance DOS. DOS/65 acts identical to Digital Research's CP/M. See information elsewhere in the newsletter. We have an OEM distribution agreement with the company to distribute under a group-purchasee clause at a price below the $125 minimum retail price.
>
> The club will actively discourage illegal copying of this copywrited disk operating system. All purchasers must sign an agreement not to distribute to others and not to copy except as to provide personal backup. The DOS committee recommends DOS/65 for anyone wishing more than a "super-fast cassette" capability from their disk. For $70 you get: A floppy disk with your own serial number; an editor; a macroassembler; Debug (similar to CP/M's Debug) monitor; BASIC-E/65; and 200 pages of documentation equivalent or better than CP/M's documentation. The disk format for DOS/65 is interchangable with CP/M. You can not execute CP/M software on your 6502 but you can edit ASCII files and perhaps run microsoft basic programs on your AIM-65 with minimum modification. We are hoping that DOS/65 becomes a 6502 standard simular to the 8080/Z80 standard CP/M.

We see no reason why DOS/65 would not be usable with the FDC-1 controller, but have not had time to pursue the matter. Perhaps one of our FDC-1 users might want to follow up on this? Being able to read at least ASCII text files on CPM formatted diskettes might be worth following up on. Since many Apple owners have added CP/M capability to their systems, the DOS/65 system could read certain Apple binary or source code files as well.

Here's another extract from the same issue:

> **INFO NEEDED!**
>
> **I am interested in trying to interface the Commodore 1541 or 1540 Disk Drives for the VIC 20 and C64 to the AIM. Since the 1541/1540 are "intelligent" drives it should be possible. If anyone has any ideas, comments, suggestions or other helpful words please write Don Lewis (or the Interchange.) Write Don Lewis at: 606 Hazel Ave., Folsom, Pa. 19033.**

We have been in correspondence with Don Lewis (who is working closely with Ron Jordan; see elsewhere in this issue). We are swapping COM-64 material and are providing him with SYM software in KIM cassette format (21 times as slow!) for conversion to the AIM.

We do not believe that there will ever be a "universal" 65XX DOS, but we do believe that the addition of the -1541 to SYMs and AIMs can open up a whole new world of PET/VIC/COM-64 software to users of these systems.

LOTS OF IDEAS FROM PHILIP KOHL

Philip Kohl does lots of experimental stuff with his SYM, his IBM PC, and his "company" computers. He keeps a very detailed log of his work. He also writes us long, rambling, letters filled with intriguing ideas, and various software tidbits.

We reprint below some of his most recent correspondence and software submissions. The first letter he sent on cassette, so that we were able to "SWP" it, and correct his abominable mis-spelling of SYM-PHYSIS, as part of our editing. The second came only as hard copy, so we publish it as-was.

We hope you get as much knowledge, and as many new ideas, out of his letters as we did, especially on the use of EEPROMs.

2465 N. W. 199th Street
Seattle, WA 98177
March 27, 1983

Dear Lux and Jean:

It's been just over 3 months since you were here. Very welcome visit. And a pleasant surprise!

What of my SYM?

Got a lot of ideas on reading the copious documentation that came with the IBM PC. Also got a lot of ideas while using the SYM in a day to day laboratory environment.

The need for restrapping U20-3 every time I wanted to try out a new idea led to a modification. Lifted all the 'Address Select' jumpers, cleaned out the holes, and replaced the outer set with wire wrap pins. The wire wrap pins are inserted from the bottom until the shoulder just touches the bottom. The holes are large enough to accept the square part of the pin, but too small to allow the rectangular shank to be forced into the board. No matter. The pins are simply soldered in place from the bottom. Then: 1) restrap with a wire-wrap tool, and 2) cut off the unneeded part of the pins protruding from the bottom of the board.

Added, June 13, 1983

Thought of using DIP sockets as described in SYM-PHYSIS #15. After some measurements, we found that the spacing wasn't quite right. Also double headers would be needed since jumpers cross the gap between the two sockets.

Original text

I started to experience random, inexplicable errors in the original SYM (intentional bad pun). Decided that the 6522 at U25 had failed, so replaced it. Turned out to be a temporary fix, for the problems started to recur about a month later. So, took the SYM that was included in the cost of the June 1981 SYMposium at Chico and installed it. Everything was suddenly normal with the usual reliability that I've come to expect from the SYM.

After some thought, I removed every low-profile socket from the original SYM, cleaned out the solder holes, and started installing new sockets of better quality. Just put the SYM back in service Thursday evening so it's too early to tell if the fix is permanent. One fear was unfounded. The SYM responded on first Power-On.

I copied RAE and BASIC to floppy. DRAM is now at locations $B000-$EFFF. Found that I was spending a lot of time at 'startup' just initializing RAM so I could do what I intended. I think you may recall how long it took just to load the DRAM from floppy. And keystroke errors happened more often than not.

How could that be fixed? The memory from $F000-$F7EF was still unused. ($F7F0-$F7FF is occupied by 4 6551's -- terminal at 19,200 baud, printer at 9,600 baud, modem at 1,200 baud, and data exchange with the IBM at a baud rate yet to be established.) So -- developed a program called STARTUP to do all the 6551 initializations, and allow the user to select the mode needed: BASIC (actually a modified EDB), MONITOR (which returns to MON1.1), TCP (communication via modem with a remote computer), and XRAY (which loads RAE and a modified version of XRAY). The software loads all the necessary files from disk for the option selected.

So, designed and built a board (KIM-4 bus) for a 2716 to be addressed between $F000 and $F7EF. Just as I finished wire wrapping the board, I realized that by the addition of an R/W signal at pin 21, I could also use a Synertek 2128 (2K by 8) in the same socket for software development and testing. So I did. Then I programmed and installed a 2716. Didn't work. Removed the R/W wire from pin 21.

Pin 21 to ground. All $00's, again. Finally found the Intel documentation: pin 21 at +5 volts. Working! Probably should have a 3.3K resistor to VCC to limit the current. Come to think of it, Intel recommends that the pin be tied to VCC.

One needs a SPDT switch on pin 21. One position provides R/W for a 2K by 8 SRAM. The other position provides +5 volts for an Intel 2716.

After using the STARTUP software for a while, I added code to tell me what file was being loaded. This is now installed on the 2716 to be discussed below.

We spoke of declining SYM interest. In the past three months, I've achieved a much broader scope with respect to personal computers. Which has led to some intensive software development effort. This after I said the SYM had no more potential for personal growth. Wrong. I've learned much these past few weeks by seeing what can be done. Maybe the Group is too small and it's too late. Simply put, my SYM is capable of a great deal more than I had ever imagined.

Have been working for the last 3 weeks on software to allow me to download software in Intel Hex format generated by a cross-assembler on a VAX, convert it to binary, and program an 8751. This is the version of the 8051 with 4K of EPROM. Expensive -- $150 each.

Last Thursday I looked at the EPROM programmer in BASIC purchased from the SUG. Decided that I just didn't have desire to go in and modify it one more time. So took the Brachman subroutines, threw away almost everything that Goodman had added, and wrote a whole new front end. All the features I liked about my BASIC version were included. Plus

1. Automatic board sensing and setting of parameters for the specific EPROM to be programmed.

2. Real-time display of programming and verification -- address

and value. Found that I had plenty of time while waiting for the 50 millisecond timer to run down to generate and transfer the data. Takes about 5 milliseconds for data transfer at at 19,200 baud.

As expected, the assembler version is FAST! I've just programmed the first 2716 and am working up courage to program the 8751.

Ruth and I wonder how you both are doing? Have not heard the results of Lux's most recent eye surgery. Nor have we heard of your exper- ience with the storms that we usually get. They appear to have been deflected to the north or south since the 1st of '83.

A long postscript: April 25, 1983

First, the new EPROM programmer now functions with Intel 2732's as well as 2716's. Verifying 2764 capability is in the very near future. I have about 16 2764's waiting for analysis. Went past the 8751 quite quickly. The SYM is a nifty base for an EPROM programmer.

Second, after a year of occasional, but very intensive, effort, I'm now coming close to understanding the NEC 765/Intel 8272 Floppy Disk Controller. And how to write software for it. Always said it took a year to 18 months to understand a VLSI chip. And it has.

Lost Drive 0 about 6 weeks ago. The heads would not load. (The drives are made by Magnetic Peripherals, Inc., a CDC subsidiary). So called MPI for documentation. Got a document number for ordering from St. Paul. MPI is in Oklahoma City. Document arrived. Good description of physical drive. Unfortunately, the data on the PCB's didn't match what I saw. PCB's? Yes. Two. Depending on whether one was dealing with hard sectored or soft sectored floppies. I was in a 'neither of the above' environment.

So gently did a lot of looking and an occasional touch with a probe from my DVOM. As one might expect, it takes a fairly healthy transistor to control a solenoid. I inadventently shorted something to something else. To be startled by a 'Thunk'. More cautious probing. Occasional 'Thunk'.

OK. The SYM is hot. Test from keyboard. Voila! Works. That was yesterday.

Had a conversation with CDC this morning. Accepted responsibility for providing insufficient information for ordering the first set of documentation. Ordered new set.

This evening I again tried Drive 0. Still works. One day is better than none. Damned if I know what happened, Lux.

April 29, 1983

Many of my friends are now aware of my tendency to write letters and forget to mail them. I received a letter from you that had been lost for a year. But one friend in the LA area lost one for seven years. Actually, the letter was typed, and in an addressed envelope. And we still lost it.

I wanted to report on changing sockets on the SYM. It has been an outstanding success. A couple of glitches the first couple of days; since then, rock solid. Don't underestimate the cost. Machined pin, gold plated, sockets. Four 24-pin, five 40-pin, eight 18-pin. It will be a couple of years before I know that it was really worth the cost and effort.

And yet another long postscript: June 11, 1983

Am considering replacing the 2716 at $F000 with an EEPROM -- the 2817. Intel has removed almost all the pain of interfacing; all that's needed now is a +21 volt supply. All of the external circuitry that is needed for the 2815/2816 in on-chip for the 2817. Programming requires an average of 10 miiilseconds per byte with a maximum of 75 milliseconds.

Small DC to DC converters are now inexpensive enough to be a good alternative to 9 volt batteries for EPROM programming. Example? The Elpac/TDK CE-0299 is a +5 volt to +21 volt DC to DC converter that fits in a 24 pin DIP socket and costs $16.56 in quantities of 1 to 9.

Was most intrigued by the CMOS version of the 6502 with 17 added instructions. So went to get the most recent Synertek catalog only to discover that you had published all of the available information in SYM-PHYSIS. I'm wondering if I learned enough from MEAN14 to be able to extend RAE?

Have spent the last six weeks writing software to interface an optical character reader with a VAX, or other computer. The OCR is made by Dest and has the capability of recognizing 8 of the more common IBM typewriter fonts. It cannot read proportional spacing. I think it can be made to read EPSON dot-matrix. Each font is in a pair of 2764's; have had good success with uploading from EPROM and analyzing the characters. So why interpose a SYM? To obtain 3 capabilities: 1) Alter the character translation table to corrrect consistent errors (example: change an '1' to a 'l') or take advantage of known characteristics of the data (all upper case or all ASCII hex), 2) delete a variable number of leading characters (the output from the Dest is not left justified to column 1, and 3) provide a preview capability before uploading the text to a computer.

I'm developing the software on my Televideo 950, and using an ADDS Regent 60 with the OCR. The conditional assembly capability of RAE has been most useful since the 'clear screen' and cursor addressing differs between the two terminals. Also, the location of the code is different. Tried the following and it works:

```
;  Conditional assembly
;  Set terminal = $00 for Televideo 950
;  Set terminal = $01 for ADDS Regent 60

          .ba $20
          set terminal = $00

          ife terminal
          .ba $7000
          ***
          ifn terminal
          .ba $9000
          .mc $7000
          ***
          .os
```

Obviously I have RAM at $7000. It's also the memory address that I use most often for programming EPROMs. (My DOS is at $9000.)

Have entertained myself with Dick Albers utilities from SYM-PHYSIS #15. Have not tried the 'wild card' memory search program yet. But did find what I think is an omission in the hex-dec dec-hex converter:

```
610           JSR INCHR     Get next character
615           CMP #$0D      Carriage return?
620           BNE D2H       Not CR; continue
```


The statement I numbered 615 is the one I think was omitted. I also let RAE figure out the decimal to hex conversions for me as follows:

```
1600TABL      .SI 10
1610          .SI 100
1620          .SI 1000
1630          .SI 10000
```

This reverses the bytes from what Dick had and hence requires three modifications to the table lookup statements. It's not that my approach is better; just wanted to see if it could be done.

Most cordially,

Philip H. Kohl

August 29, 1983

Dear Lux and Jean:

This is being written with a GTE Microsystems 65SC02 plugged in to the space that had been occupied with a Synertek 6502. Did that about two hours ago. The date code on the 65SC02 is 8331.

As a test I modified 'prompt' and installed it in my EEPROM. Which exercizes PHY and PLY. Code enclosed.

Have spent the idle moments of the past two weeks disassembling RAE-1. Idea? Extend to include the new op-codes. Where am I? Just finished with the code from EC4A to EFFF.

Went to a fair amount of trouble to obtain the 6 65SC02's that I now have. Called the number published in the last issue of SYM-PHASIS. That got me a 1-800 number in Tempe, AZ. That got me the names and phone numbers of a local rep and 2 local distributors. It also resulted in a 16-page brochure from GTE on the 65SCxx. And the information that there will be a 65SC21, 65SC22, 65SC32, and 65SC51.

Called one of the local distributors and found out I knew more than they did. Also checked with a Synertek distributor to learn that they would have them 4Q83 at the earliest. As a result of my interest, one GTE distributor decided to stock them. Which requires an investment, as I'm sure you well know. Boeing's cost? Just under $10 each in quantities of 6.

September 5, 1983

Called Jean last Wednesday for a copy of SWP-2.5. Arrived on Friday. Attempted to load the tape. Odd, low sound. The LED 'S' didn't even go out. Easy problem. Tape recorder was set to 15/16ths ips. So SWP is now on floppy, and listed. Waiting 'til I get around to installing it. Most of which is changing the DOS code so as to continue on disk. Although why I don't know since I don't write letters long enough to require more than the space between $1000 and $4FFC. Check enclosed.

The 65SC02 seems to be working nicely. The occasional errors I thought came from a 1 bit change in memory and/or dirty contacts seem to be gone. My system is now more 'solid' than I can remember. Thoughts of replacing the BETA dynamic RAM boards with RAM have been pushed well down in the stack.

The major effort of this Labor Day weekend is the continuing under-

standing of RAE. The disassembly is complete and I'm buried in paper. Looked for clues via RAE Notes, SYM-PHASIS, Saturn Softnews, and purchased software. Found some clues. But my admiration for Carl Moser grows. RAE is elegant -- including the traps for the unwary, not all of which I've found.

[EDITOR'S NOTE: This program is NOT for a 6502. It is for the 65SC02. Phil has "manually" added the OPCODEs for the two "new" operations, PHY and PLY. These, plus the other new OPCODEs in the 65SC02, could also have been added to RAE as MACROs. While this program, or variants of it, has been published widely and frequently, we reprint it here as our very first 65SC02 program!]

```
                        0005
                        0010 ; Vance's Prompt routine
                        0015
                        0020 ;This subroutine by H. T. Vance, Manchester
                        0025 ;  MA. manipulates the stack so that text can
                        0030 ;  be embedded directly in the program.  It
                        0035 ;  requires a 'JSR prompt' followed immediately
                        0040 ;  with a '.BY' containing the text to be
                        0045 ;  output. The text must be terminated with
                        0050 ;  '$00' as the last byte.
                        0055
                        0060 ;  Modifications:
                        0065 ;     Remove 256 byte limit: 07/24/83
                        0070 ;     Preserve r(y):         08/09/83
                        0075 ;     65SC02 version:        08/29/83
                        0080
                        0085 ptemp      .de $20
                        0090
                        0095 outchr     .de $8A47     ;Output character
                        0100
                        0105            .ba $E00
                        0110            .os
                        0115
0E00- 68                0120 prompt     PLA :         ;Program counter, low
0E01- 85 20             0125            STA *ptemp    ;PCL from stack
0E03- 68                0130            PLA :         ;Program counter, high
0E04- 85 21             0135            STA *ptemp+1  :PCH from stack
0E06- 5A                0140            .by $5A       ;PHY
0E07- A0 00             0145            ldy #$00
0E09- E6 20             0150            INC *ptemp    ;Next byte
0E0B- D0 02             0155            BNE display   ;Branch if same page
0E0D- E6 21             0160            INC *ptemp+1  :Next page
                        0165
0E0F- B1 20             0170 display    LDA (ptemp),y :Fetch character
0E11- F0 0B             0175            BEQ done      ;Branch if $00
0E13- 20 47 8A          0180            JSR outchr    ;Display character
0E16- E6 20             0185            INC *ptemp    ;Prepare for next character
0E18- D0 F5             0190            BNE display   ;Branch if same page
0E1A- E6 21             0195            INC *ptemp+1  ;Next page
0E1C- 80 F1             0200            .by $80 $F1   ;BRA display
                        0205
0E1E- 7A                0210 done       .by $7A       ;PLY
0E1F- A5 21             0215            LDA *ptemp+1  :Program counter, high
0E21- 48                0220            PHA :         ;PCH to stack
0E22- A5 20             0225            LDA *ptemp    ;Program counter, low
0E24- 48                0230            PHA :         ;PCL to stack
0E25- 60                0235            RTS
                        0240
                        0245            .EN
```


[And here is a program for those of you who are fortunate enough to be using EEPROMs. This is also for the 65SC02. For those using the 6502, replace line 0340 (BRA restore) with BEQ restore. BRA is the 65SC02 mnemonic for BRanch Always.]

```
                        0005
                        0010 ;  Interrupt handler for EEPROM programming
                        0015
                        0020 ;  Permits EEPROM programming using
                        0025 ;  MON1.1 commands
                        0030
                        0035 ;  Machine code resides in EEPROM at $F780 to
                        0040 ;  $F7C2.  Relocate to RAM before using with
                        0045 ;  the following command:
                        0050
                        0055 ;     .b <target address>,F780,F7C2
                        0060
                        0065 ;  The code is relocatable to any spare area in
                        0070 ;  RAM that has $42 unused bytes.
                        0075
                        0080 ;  Then issue the following command to set the
                        0085 ;  IRQ vector:
                        0090
                        0095 ;     .g <target address>
                        0100
                        0105 ;  Program the EEPROM using the mormal MON1.1
                        0110 ;  commands:  B, D, F, M.
                        0115
                        0120 ;  On completion, reset the SYM to restore the
                        0125 ;  IRQ vector.
                        0130
                        0135 ;  Original:  July 17, 1983
                        0140 ;  Sense return address:  July 23, 1983
                        0145 ;  65SC02 modification:  August 31, 1983
                        0150
                        0155 access     .de $8B86
                        0160
                        0165 irqvec     .de $A67E
                        0170
                        0175            .ba $F00
                        0180            .os
                        0185
                        0190 ;  Set IRQ vector
                        0195
0F00- 78                0200 intrupt    sei
0F01- 20 86 8B          0205            jsr access    ;Write enable system RAM
0F04- A9 10             0210            lda #1,wrwait
0F06- 8D 7E A6          0215            sta irqvec
0F09- A9 0F             0220            lda #h,wrwait
0F0B- 8D 7F A6          0225            sta irqvec+1
0F0E- 58                0230            cli
0F0F- 60                0235            rts
                        0240
                        0245 ;  Wait until EEPROM byte is programmed
                        0250
0F10- 78                0255 wrwait     sei
0F11- 85 20             0260            sta *$20      ;Accumulator scratch address
0F13- 68                0265            pla :         ;Processor status
0F14- 85 21             0270            sta *$21      ;Save processor status
0F16- 68                0275            pla :         ;Program counter, low
0F17- 85 22             0280            sta *$22      ;Save PCL
0F19- 68                0285            pla :         ;Program counter, high
0F1A- 85 23             0290            sta *$23      ;Save PCH
```

```
0F1C- C9 87     0295            cmp #$87     ;FILL, BLOCK return PCH?
0F1E- D0 0E     0300            bne notblock :Branch if not
0F20- A5 22     0305            lda *$22     ;Program counter, low
0F22- C9 BF     0310            cmp #$BF     ;BLOCK PCL return address?
0F24- D0 08     0315            bne notblock :Branch if not BLOCK PCL
                0320
0F26- A5 20     0325 block      lda *$20
0F28- D1 FC     0330            cmp ($FC),y
0F2A- D0 FA     0335            bne block    ;Loop until equal
0F2C- 80 06     0340            .by $80 $06  ;BRA restore
                0345
0F2E- A5 20     0350 notblock   lda *$20
0F30- D1 FE     0355            cmp ($FE),y
0F32- D0 FA     0360            bne notblock :Loop until equal
                0365
0F34- A5 23     0370 restore    lda *$23     ;PCH
0F36- 48        0375            pha :        ;To stack
0F37- A5 22     0380            lda *$22     ;PCL
0F39- 48        0385            pha :        ;To stack
0F3A- A5 21     0390            lda *$21     ;Processor status
0F3C- 09 02     0395            ora #$02     ;Set Z = 1
0F3E- 48        0400            pha :        ;To stack
0F3F- A5 20     0405            lda *$20     ;Restore accumulator
0F41- 58        0410            cli
0F42- 40        0415            rti
                0420
                0425            .en
```

SKF-FORTH FOR THE FDC-1
--- ----- --- --- --- -

Here is a brief note from Peter Ashby, c/o Victor Harbor H/S, Victor Harbor, SA 5211, Australia, giving the single screen necessary to add FDC I/O capability to Brown's SKF FORTH. We still continue to be amazed at FORTH's ever so easy extendability!

Conversion of SKF. (Brown's Forth) to run with FDC.
Note that I can only use 4 of the 5 sectors in each double density track. (For some reason my SHUGART drive locks up on the start of the 5th track).
Note also that I use no header information and store straight to tracks and sectors. This may be of some use to others.

Peter Ashby
30 May 1983

```
1 LIST
Screen 1      1 hex
  0 : WORD.IN DUP + [COMPILE] ' + ; IMMEDIATE
  1 : REPLACED.BY [COMPILE] ' 2 - SWAP ! ; IMMEDIATE
  2 CODE (MDISK.R)  8B86  JSR,
  3      XSAVE STX,  YSAVE STY,  04 IM, LDA,  9800 JSR,
  4      XSAVE LDX,  YSAVE LDY,  NEXT JMP,
  5 END-CODE
  6 CODE (MDISK.W)  8B86  JSR,
  7      XSAVE STX,  YSAVE STY,  05 IM, LDA,  9800 JSR,
  8      XSAVE LDX,  YSAVE LDY,  NEXT JMP,
  9 END-CODE  HEX
 10 : (SET1) >R       4  /MOD  A601 C!  1+ A602 C!
 11                 100  /MOD  SWAP     A603 C!  A604 C!
 12           R> IF (MDISK.R)  ELSE (MDISK.W)  THEN ; DECIMAL
 13 0 WORD.IN (SET) REPLACED.BY (SET1)   1 ' S/BLK !
 14 1 WORD.IN (SET) REPLACED.BY  EXIT
 15 0 WORD.IN DRW REPLACED.BY (SET)   1 WORD.IN DRW REPLACED.BY EXIT
OK
```


A BASIC APPLICATION PROGRAM
- ----- ----------- -------

As you must have noticed, most of the programs published in SYM-PHYSIS are "utilities", rather than applications, and utility programs are more efficient in ML than in BASIC (this "aside" is to explain away what may seem to be a double-bias against both applications and BASIC programs).

We received a BASIC application program on cassette from R. Dale Barber, and tried it out. Unfortunately, since we did not understand the application, we had no idea of how to answer the input requests, e.g., what were the units involved, and what would be "reasonable" (at least order-of-magnitude) inut values? Since our inputs were "garbage", so were our outputs; GIGO: Garbage In, Garbage Out. We pointed this out to Dale, and he sent along the necessary documentation.

Since the program (and the documentation) are quite lengthy, and the application is so highly specialized, our first thought was to "shelve" the material. On the other hand, it is important to show that the SYM can and does earn its keep. Additionally, many SYMs were originally bought by non-EEs, e.g., Mechanical and Civil Engineers (to list only the principal engineering fields, and to say nothing of the other, non-engineering, disciplines), primarily for self-training in the computer field.

This we know, and certainly do appreciate, since so many of them have called us for help during their first few weeks with the SYM! After that, we hear from many of them only much later, when they send along an example showing how much they have learned on their own, and how the SYM has more than repaid for itself.

Here then, is Barber's CHANNEL FLOW COMPUTATION program for the civil (or uncivil, to quote Dale!) engineers, among you:

```
 1 PRINT"WRITTEN BY R.D. BARBER, P.E. - AUGUST, 1982"
 10 PRINT"CHEZY-KUTTER CHANNEL FLOW COMPUTATION"
 20 PRINT"FOR TRAPEZOIDAL CHANNELS OF WIDTH W AND SIDE SLOPES SS:1"
 22 DIM DR(100,5):PRINT"NOTE: THIS PROGRAM WILL HANDLE UP TO 100 INCREME
NTS"
 23 PRINT"OF DEPTH. PLEASE TURN PRINTER ON FOR SOFTWARE CONTROL IF HARDC
OPY"
 24 PRINT"IS DESIRED."
 25 PRINT
 26 INPUT"ENTER YOUR JOB NAME ";JO$
 27 INPUT"ENTER THE DATE (MO/DAY/YR) ";DA$
 28 INPUT"TO SKIP TABLE OF N VALUES ENTER 1,OTHERWISE 2";J
 29 ON J GOTO 145
 30 PRINT"THE FOLLOWING TABLE GIVES THE RELATIVE ROUGHNESS FACTORS"
 40 PRINT"FOR VARIOUS CHANNEL CONDITIONS. SELECT THE FACTOR WHICH"
 50 PRINT"BEST FITS YOUR PROBLEM."
 60 PRINT:PRINT"FINISHED CEMENT MORTAR LINING - .011(BEST) TO .015(WORST
)"
 70 PRINT"WOOD PLANK LINED FLUMES - .010(BEST) TO .016(ROUGHEST)"
 80 PRINT"CEMENTED RUBBLE LINING - .017 TO .030"
 90 PRINT"SMOOTH STEEL PLATE - .011 TO .015"
 100 PRINT"CORRUGATED METAL LINING - .0225 TO .030"
 110 PRINT"STRAIGHT, UNIFORM EARTH CANALS - .017 TO .025"
 120 PRINT"ROUGH, WEEDY EARTH CANALS - .025 TO .040"
 130 PRINT"SMOOTH UNIFORM NATURAL CHANNELS - .025 TO .033"
 140 PRINT"WINDING, SLUGGISH, WEEDY CHANNELS - .040 TO .15 (VERY WORST)
 145 PRINT
 150 INPUT"ENTER THE VALUE SELECTED FOR N: ";N
 152 PRINT"ENTER THE MAXIMUM DEPTH OF INTEREST AND INCREMENTS IN FEET"
 154 INPUT D,I
 160 PRINT"NOW ENTER BOTTOM WIDTH IN FT., AND SIDE SLOPES (2 FOR 2:1)
 170 INPUT W,SS
 180 INPUT"NOW ENTER THE CHANNEL GRADE IN PERCENT ";G
```

```
211 DE=Ø
220 FOR A=1TOINT(D/I)
230 GOSUB 1ØØØ
240 NEXT A
245 POKE42576,216
250 PRINTCHR$(17):REM TURN ON PRINTER
260 PRINTCHR$(31):REM SET BOLD TYPE
270 PRINTSPC(6)"CHANNEL FLOW COMPUTATION"
275 PRINTCHR$(3Ø):REM SET NORMAL TYPE
280 PRINT:PRINT"FOR: ";JO$,"DATE: ";DA$
290 PRINT:PRINT"CHANNEL WIDTH=";W;"FT.","SIDE SLOPES=";SS;":1"
300 PRINT"GRADE=";G;"%","N FACTOR=";N
310 PRINT:PRINT"DEPTH","CFS","VEL.","AREA","HY RAD"
320 FOR A=1TOINT(D/I)
330 FOR B=1TO5
340 PRINTDR(A,B),
350 NEXTB
370 NEXTA
380 PRINTCHR$(12):REM FORMFEED
390 POKE42576,Ø1:REM RESET TERMINAL SPEED
395 FORJ=1TO5ØØØ:NEXTJ:REM DELAY TIL FF FINISHED
396 PRINTCHR$(19):REM PRINTER OFF
400 PRINT"TO RUN ANOTHER VARIATION UNDER THIS TITLE, ENTER 'Y'"
410 INPUT"OTHERWISE JUST HIT CARRIAGE RETURN TO ESCAPE";RU$
420 GOTO145
430 END
1000 REM CALCULATIONS
1010 DE=DE+I
1020 AR=DE*W+SS*DE*DE:REM AREA
1022 AR=INT(100*AR+.5)/100
1030 PE=W+2*SQR(SS*SS+1)*DE:REM PERIMETER
1040 R=AR/PE:R=INT(100*R+.5)/100
1041 F=.281/G+41.66
1042 C=(1.811/N+F)/(N/SQR(R)*F+1)
1044 V=C*SQR(R*G/100)
1045 V=INT(100*V+.5)/100
1046 Q=AR*V:Q=INT(100*Q+.5)/100
1060 DR(A,1)=DE
1070 DR(A,2)=Q
1080 DR(A,3)=V
1090 DR(A,4)=AR
1100 DR(A,5)=R
1120 RETURN
```


R. DALE BARBER
CONSULTING ENGINEER
SETON VILLAGE
SANTA FE, NEW MEXICO 87501

505-983-7021

7/22/83

Dear 'Lux' & Jean:

It is embarrassing to admit that I forget that the rest of the world doesn't talk 'ENGINEER'. The program I sent you was missing the sketch showing what a ditch looks like, and the related terminology. The equations used were developed empirically early in the century and are called the Chezy-Kutter formulas. They were fairly formidable to face with a slide rule, and published tables were most commonly used for practical work. Computers make duck soup out of the formulas, and this Channel Flow Computation program prints out a custom set of tables for virtually any ditch configuration.

The variables which determine how much water will flow in a ditch or channel are: 1. The cross-sectional area, computed from the depth of flow, width of the bottom, and geometry of the sides. 2. The roughness of the channel surface, selected from an empirical table of values listed in the program. 3. The ratio of the channel area to the wetted perimeter, called the hydraulic radius. 4. The slope, or grade of the channel bottom usually expressed in percent of grade (The vertical drop of the channel in 100 ft.of horizontal run).

Once the program has received the essential data, including the job name and date, it constructs a set of tables for increments of water depth specified in the input showing the flow in cubic feet per second, the water velocity in feet per second, the cross sectional area of flow in square feet, and the hydraulic radius, a dimensionless ratio.

If you like your answer in gallons per minute, multiply CFS by 448.8. For a channel with vertical walls the slope is zero. Enclosed with the sketch is a printout of a typical problem. The program is useful to determine how much improvement in flow can be attained by lining with a smoother material, such as concrete (changing the 'N' factor).

If there are any other civil (or uncivil) engineers out there interested in exchanging programs, I also have a Hardy Cross program (It reiteratively solves complex pipe network flow problems to you other folks), and a drainage runoff program based on the U.S. Soil Conservation publication 'Estimating Runoff for Small Drainage Areas' which will accommodate over 100 drainage areas and prints the estimated runoff for 10 and 100 year storms in tabular form.

The great utility programs published in SYM-PHYSIS have been the nuts and bolts to assemble a powerful machine for my work. Most of the commercially available programs for other computers are overpriced garbage, so it has been no disadvantage to write my own software on the SYM. It may be time for specialty user groups to form around the nucleus of SYM-PHYSIS and start pooling our combined efforts. There is certainly very little commercial software available for engineers outside of the electronic and computer field. You have already published some letters from people in the medical arts, so where are all the SYMgineers?

I am sending you the BASIC file for the Hardy-Cross pipe network (tape file 3) that you expressed an interest in because of it's iterative solution, but will warn you in advance, even setting up a problem for it is a formidable task if you aren't familiar with hydraulics engineering. The problem is similar to wiring up a large grid of interconnected resistors of different values to resemble a street map, connecting different constant current sources to ground at each intersection, to simulate the local usage, applying several voltages at different points on the grid to represent the water supply, and then computing the current and voltage drop across each resistor in the network to arrive at the pressure distribution around the whole system.

As a matter of fact, exactly such a machine was built and sold in the 1920's to solve pipe network problems, but it was very expensive and took weeks to set up a problem. The resistor values represented the solution of nonlinear hydraulic equations based on the diameter, roughness, flow velocity, and length of many different pipes. Voltages were read with a meter across each resistor to get the analog value of the pressure drop.

The mathematical solution to the problem is indeterminate, but the Hardy-Cross solution computes a discrete solution from assumed variables for each loop (Thevenin equivalent) and updates the assumed values for the next iteration. Depending on the complexity of the network, the program will converge to a reasonable set of values within five or six iterations, sometimes setting up interesting oscillations in portions of the net. The solution in Basic for fifty or sixty pipes can take fifteen to twenty minutes. A machine language program would be nice, but I don't have a year of spare time to write it.

Hugh Criswell's Basic Data Save routine (7:17) enables me to save the data from these runs on tape. Almost all my programs, including the double entry book-keeping system rely heavily on that neat subroutine.

I have rambled on enough and must exit. Am looking forward to the Forth to see how it will apply to control applications. Will send information on my new burglar alarm (the old one was wiped out by lightning) using Radio Shack optoisolators with TRIAC output as soon as it is up and working.

Sincerely Yours,
Dale Barber

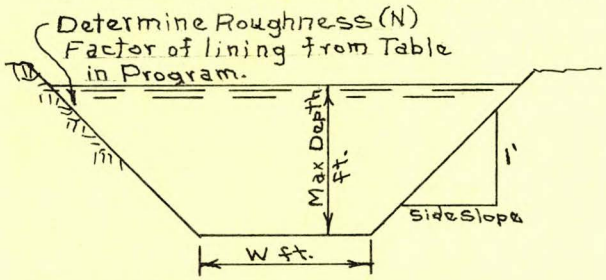
CHANNEL FLOW COMPUTATION

FOR: GOPHER GULLY DRAINAGE SYSTEM DATE: 7/22/83

CHANNEL WIDTH= 6 FT. SIDE SLOPES= 2 :1
GRADE= .56 % N FACTOR= .02

DEPTH	CFS	VEL.	AREA	HY RAD
.5	9.8	2.8	3.5	.42
1	35.2	4.4	8	.76
1.5	75.87	5.62	13.5	1.06
2	133	6.65	20	1.34
2.5	207.35	7.54	27.5	1.6
3	300.24	8.34	36	1.85
3.5	414.05	9.1	45.5	2.1
4	548.24	9.79	56	2.34

ILLUSTRATIVE CROSS SECTION OF A CHANNEL SHOWING DIMENSIONS CONTROLLING FLOW



Channel Grade % is the fall in ft. per 100 ft. along the flowline.

Side Slope is the horizontal distance for 1 ft. of rise. ie. 2:1 slope = 2 ft. hor. & 1 ft. vert.

A CRYPTOGRAPHIC PROGRAM

Dale Barber sent along several BASIC programs, as mentioned in his letter (above). One was a stock market simulation, called "BLACK FRIDAY", of which these are the opening lines:

```
100 REM******** ATARI BLACK FRIDAY *********
110 REM........ BY ROBERT BAKER .............
120 REM .ADAPTED FOR SYM-BAS BY R.D. BARBER FROM BYTE.
160 REM:************************************
```

We mention this only because Dale sent us an encrypted message, together with the required "KEY", plus the program printed below to do the decoding, and the clear-text message refers to this program.

First is our sample run, then the program itself. Note that the KEY was not printed out. [The garbling of the word "INTELLIGENT" in the decoded message is our fault, since our input was wrong. We absent-mindedly entered a "U" following the "Q" in the input text, and goofed-up on the correct next letters; GIGO!]

```
BAZERIES CYLINDER CRYPTOGRAPHIC SYSTEM
ADAPTED TO SYM-BASIC BY R.D.BARBER
FROM THE PROGRAM BY RINALDO F. PRISCO
PUBLISHED IN BYTE MAGAZINE JUNE 1983
COPYRIGHT-PRISCO-8/1/81


THE PROGRAM USES A 20-DISK BAZERIES CYLINDER
TO ENCODE MESSAGES. THE CYLINDERS ARE ROTATED
TO MATCH A MINIMUM 20 LETTER KEY SENTENCE.
SEE THE BYTE ARTICLE FOR FURTHER DETAILS.


ENTER KEY PHRASE AT LEAST 20 LETTERS LONG

LOADING CYLINDER . . . . .

ENTER TEXT

<E>NCODE OR <D>ECODE?
IIIFRGYHILMLIYIHSJCIWFIHKXZYIXMRTOGQUECLULA
THESTOCKMARKETGAMEISONEOFTHEMOREINTEHNIGENT

MORE TEXT? Y

ENTER TEXT

<E>NCODE OR <D>ECODE?
JMLSLYZXKCCRIONWOXLQUNHSEKMOJHNICJG
GAMESIHAVESEENFORCOMPUTERSSTOPTRYIT

MORE TEXT? N


OK
```

```
10 PRINT"BAZERIES CYLINDER CRYPTOGRAPHIC SYSTEM"
20 PRINT"ADAPTED TO SYM-BASIC BY R.D.BARBER"
30 PRINT"FROM THE PROGRAM BY RINALDO F. PRISCO"
40 PRINT"PUBLISHED IN BYTE MAGAZINE JUNE 1983"
50 PRINT"COPYRIGHT-PRISCO-8/1/81"
60 PRINT:PRINT
70 PRINT"THE PROGRAM USES A 20-DISK BAZERIES CYLINDER"
80 PRINT"TO ENCODE MESSAGES. THE CYLINDERS ARE ROTATED"
90 PRINT"TO MATCH A MINIMUM 20 LETTER KEY SENTENCE."
100 PRINT"SEE THE BYTE ARTICLE FOR FURTHER DETAILS."
110 PRINT:PRINT
1000 REM BAZERIES CRYPTOSYSTEM
1010 REM BYTE 6/83
1020 REM COPYRIGHT (C) 8/1/81
1030 REM RINALDO F. PRISCO
1040 REM DEPT. OF MATHEMATICS
1050 REM SUNY AT OSWEGO, N.Y. 13126
1110 DIMD$(20),P(20),K$(20)
1120 FORI=1TO20:READD$(I):NEXTI
1130 INPUT"ENTER KEY PHRASE AT LEAST 20 LETTERS LONG ";K$
1140 K=ASC(K$)
1150 REM ELIMINATE SPACES
1160 S$=K$:GOSUB1700:K$=S$:PRINT
```

```
1170 REM RESTRICT LENGTH TO 20
1180 IFLEN(K$)>20THENK$=LEFT$(K$,20)
1190 REM USE SORT ON KEY TO PERMUTE DISKS
1200 PRINT"LOADING CYLINDER . . . . ."
1210 FORJ=LEN(K$)-1TO2STEP-1
1220 F=0
1230 FORI=1TOJ
1240 L=I+1
1250 IFMID$(K$,I,1)<=MID$(K$,L,1)THEN1300
1260 T$=MID$(K$,I,1):U$=MID$(K$,L,1)
1262 K$=LEFT$(K$,I-1)+U$+T$+RIGHT$(K$,LEN(K$)-I-1)
1280 T$=D$(I):D$(I)=D$(L):D$(L)=T$
1290 F=1
1300 NEXTI
1310 IFF=0THEN1330
1320 NEXTJ
1330 REM CYLINDER LOADED
1340 PRINT
1350 INPUT"ENTER TEXT ";T$
1360 PRINT
1370 INPUT"<E>NCODE OR <D>ECODE? ";Y$:Y$=Y$+" "
1380 IFLEFT$(Y$,1)="D"THENF=1:GOTO1390
1385 F=0
1390 N=K-65
1400 REM ELIMINATE SPACES FROM T$
1410 S$=T$:GOSUB1700:T$=S$:PRINTT$
1420 REM LIMIT TO 20 CHARACTERS AT A TIME
1430 L=LEN(T$):IFL>20THENL=20
1440 REM ORIENT DISKS TO TEXT
1450 I=1
1460 FORJ=1TO26
1464 IFMID$(T$,I,1)=MID$(D$(I),J,1)THEN1468
1466 NEXTJ
1468 P(I)=J
1470 I=I+1:IFI<=LGOTO1460
1480 REM SET R TO PROPER ROW NUMBER
1490 N=N+1:R=N-20*INT(N/20)
1500 IFR=0THENR=1
1510 IFF=1THENR=26-R
1520 REM SET POINTERS TO ROW R
1530 FORI=1TOL
1540 P(I)=P(I)+R
1550 IFP(I)>26THENP(I)=P(I)-26
1560 NEXTI
1570 REM PRINT NEW TEXT
1580 FORI=1TOL
1590 PRINTMID$(D$(I),P(I),1);
1600 NEXTI
1610 REM ANY MORE TEXT?
1620 IFL=LEN(T$)THEN1640
1630 T$=RIGHT$(T$,LEN(T$)-L):GOTO1430
1640 PRINT:PRINT
1660 INPUT"MORE TEXT? ";Y$:Y$=Y$+" "
1670 PRINT
1680 IFLEFT$(Y$,1)="Y"THEN1350
1690 END
1700 REM REMOVE BLANKS FROM S$
1710 S$=" "+S$
1720 FORI=2TOLEN(S$)-2
1730 IFMID$(S$,I,1)<>" "THEN1750
1740 S$=LEFT$(S$,I-1)+RIGHT$(S$,LEN(S$)-I)
1750 NEXTI
1760 S$=RIGHT$(S$,LEN(S$)-1)
1770 RETURN
```



```
1780 DATAFNWADLZJKMQSCXHVPTGIBOEYRU
1790 DATAETXQPVJCBNRADSKHIYOGULMZFW
1800 DATALEVQXYGCDOZWTPJRHIBKAMSUNF
1810 DATAXYCVQWEITHNPLKSAOGRJBUZDFM
1820 DATAODTZCRFHENBYUMQXAWVGLJSIKP
1830 DATAVKUNYEWFMICOJLHGAPTZRXSBQD
1840 DATAJMPHVOXRIFKBECUQDZTALGNSWY
1850 DATARGJYZBNQHCFAMTILOWVEPUXSKD
1860 DATAZQJKOIBRMFHVTNWXEGSCUPYADL
1870 DATAUAXTORVWKHPZNLIMVQCJFGEYSD
1880 DATAMGHXLETYFKZSRABNOUPCQWDIJV
1890 DATAPGNBRTFVOWSCZXDLMIKUJAHYEQ
1900 DATAZEDIPGUOSMFBRXJCYWNVQKTALH
1910 DATAFDPSMLYKXZWJONCBUVEIRTHAQG
1920 DATAMIGHUOSLYCDJVQXBTREFKWNPAZ
1930 DATAGPZLTABUNEJSFVKRWMIHCXDQYO
1940 DATAXIDLETVZYHUBQNWAGMSKCROJPF
1950 DATAWHMFSGUZEYXRVICOLQKPBDANJT
1960 DATAIKLMATHNCZXWUOGSVYBQFPJDER
1970 DATAJVOHKYZCLUXESFWTRPQDBMAGNI
```

SALE SALE SALE SALE SALE SALE

The following, at rock bottom price, items are for sale on a first-come, first-served basis. After these items are gone from our inventory, we will no longer have them available.

RAE-1: list $159, now $120.

RAE-1/2: list $95, now $65.

BAS-1: list $95, now $65.

DAC MUSIC BOARD: 8-Bit Digital to Analog Converter, Audio Amplifier, Manual, Demo Digital Cassette, was $74, now $55.

EPROM ERASER KIT: was $20, now $15.

Note: If you now have our Eprom Eraser Kit you may want to buy a "spare" bulb, at our cost of $8, as they are almost impossible to find.

SALE SALE SALE SALE SALE SALE

ROM COPYRIGHTS

On 30 August 1983, a three-judge panel in the U.S. Court of Appeals (Philadelphia) overturned a lower (district) court ruling on the copyright protectability of firmware (object code in ROM), ruling in favor of Apple Computer, Inc., and against Franklin Computer Corp., one of the makers of Apple compatible systems.

Franklin intends to ask for a rehearing by the entire Court of Appeals, while Apple will return to the federal district court for a restraining order against the Franklin Ace 1000.

The appellate (no connection with Apple!) court ruled, in effect, that ROM chips (even if they are part of the operating system, as opposed to ROMs containing applications programs) are entitled to the same protection as computer programs written on paper or stored on tape.

THE FUTURE OF SYM

We have been watching with much interest Synertek's plans to find a successor to carry on the SYM/KTM product line. We ourselves were not available, for a number of reasons, chief of which was our intent to retire and travel.

As of this date negotiations for transfer of the product line are underway, and we will be meeting with the parties involved, to provide whatever assistance we can to the new vendor in establishing his customer support program.

We'll provide further information in Issue #17.

IMAGE PROCESSING

We ordered, and are eagerly awaiting delivery of, a MicronEye "Bullet" (COM-64 version), so that we could experiment with image processing. We'll try it out first on the Commodore, naturally, but would then like to switch it over to our MTU Visible Memory SYM/CODOS System, because of the faster operating DOS. We'll report on the MicronEye, which is a solid state video imaging system incorporating a device called the IS32 OpticRAM. Meanwhile, we'd like to hear from others working in image processing, or with the MicronEye.

MISCELLANEA

We had two SYMmers visit us from Switzerland this summer, Dr. ULRICH GUGGISBERG (from near Berne), and NORBERT THUERING (from near Zurich). Unfortunately, they arrived and departed within two days of each other and didn't get to meet.

Mr. Adel Madani, a Saudi Arabian student, working on his Ph.D. in Biomedical Engineering at Johns Hopkins (Baltimore, Maryland), spent three months studying with us on computer systems integration and application.

BILL CRAMER sent us a BASIC file handling program "BASIC ADDRESS AND PHONE BOOK", together with the RAE source code for an ML program to save and retrieve the data files. We ran out of space in this issue, but Bill offers to supply the programs on FDC-1 diskettes for a $10 media and handling charge. His address is 5609 N. Colony Blvd., The Colony, TX 75056.

RAM-BLINGS

As has become our custom, at least since Issue #15, we send the first 48 pages of SYM-PHYSIS to a "high-volume" printer who has a slow turn-around time and can work only in multiples of eight pages at a time, but folds and staples automatically during the press run, thus saving us many hours of hand work. This gives us an extra week to do the last four pages; these are then sent to an "Instant Printer". Both printers complete their work at the same time, and we can then stuff and mail both parts together.

The extra few days gained this way give us a little breathing spell to write these closing words, and to go over once again the vast amounts of material which we did not have the time or space to get into this issue. We always have guilt feelings at this stage for having had to omit so much valuable material, but are consoled by realizing how much material we already have on hand for the next issue.

We also spend some of this time worrying if we have provided the right "mix" of programs vs editorial opinions, simple vs advanced material, BASIC vs ML, hardware vs software, utilities vs applications, etcetera, etcetera, usw.

This time, with Synertek effectively "orphaning" the SYM/KTM systems by the end of 1983, and with our retiring from full-time teaching, also at the end of the year, we have been thinking of our own plans for 1984-??. Now that we have become a "Professor Emeritus", we are free to accept a Visiting Professorship anywhere in the world. "Have computer know-how, will travel." If any of our academic colleagues would be interested in the possibility of our joining their faculty on a temporary appointment basis, please let us know!

While we published (or gave a source for) two programs for the unexpanded SYM in this issue, most of the programs in recent issues required fair amounts of expansion, and most of the articles required a relatively high degree of sophistication on the part of the reader. If you have come along with us this far, you are ready to continue on your own the rest of the way with your SYM, in any direction you may chose.

During our twenty years in industry, followed by full-time teaching for thirteen years, the four years on SYM-PHYSIS has been our longest sustained effort on a single project, and it is time to move on. What we are saying, in effect, is that the Users' Group was a real necessity for beginners, but is far less urgent for experienced users, of whom there were very few, back in April of 1979, when the Users' Group was started. Since there will be very few new SYM beginners (for these there will still be back issues available), we feel that we can soon relax a little, and turn our attention to other interests.

Thus Issue #17 will be our final issue. We have gone from bimonthly to quarterly to three issues per volume. (We almost said per year, but Volume 4 will spread out over more than a year!) While the number of pages per volume (or, per year, if you like) has remained essentially constant, the work load has seemed to increase. This may be due to the increasing amount of correspondence, or to the sublimination of our own research interests, or just plain old "burn-out" after four years of doing the same thing.

We will be starting to put Issue #17 together during the Christmas recess, so that you should be receiving your copy early in 1984. There are more than enough good programs already on hand to fill a complete issue, so Issue #17 will have a much higher signal/noise ratio than this one!

Regards,