

SSSS S S S S SSSS S S S S S SSSS SSS SSS
 S
 SSSS SSS S S S S SSS SSSSS SSS SSS S S S S
 S
 SSSS S S S S S S S S S S S S S S S S S S

THE SYM-1 USERS' GROUP NEWSLETTER

ISSUE NUMBER 2 - MARCH/APRIL 1980

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

We welcome for publication all articles dealing with any aspect of the SYM-1, and its very close relatives. Authors retain all commercial copyrights. Portions of SYM-PHYSIS may be reproduced by clubs and educational institutions, and adaptations of programs for other computers may be freely published, with full credit given and complimentary copies provided to SYM-PHYSIS and the original author(s). Please include a self-addressed stamped envelope with all correspondence.

Editor/Publisher: H. R. "Lux" Luxenbergs
Business/Circulation: Jean Luxenbergs
Associate Editor Thomas Gettys

SUBSCRIPTION RATES:

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

FLOPPY DISK OPERATING SYSTEM FOR THE SYM

We are, as you should know by this time, very much impressed with the potential of SYMs but realize that not enough software exists (that is why we publish SYM-PHYSIS). It is fairly simple from the hardware standpoint to add a disk controller and a pair of floppies to the SYM, (it really was!), but good Disk Operating Systems (DOS) are not easy to come by. We watch the advertisements very carefully, and when we hear or see that a vendor announces the imminent availability of a SYM product, we write and/or phone to beg, buy, borrow or maybe even steal a pre-release or prototype version, offering to test, evaluate, shake-down or otherwise help to debug the product and its documentation, prior to official release. We did this with Bob Denison on the 2KSA, Synertek Systems Corporation on RAE-1, Carl Moser on several software packages we hope to begin to sell soon, etc. We have long been aware of Hudson Digital Electronics File Oriented Disk System (FODS) for KIM Systems, and had heard only good things about it. FODS may be used with either size floppy disk, so you can start out with the 5 1/4 inch disks and upgrade to the 8 inch system with no conversion problems. In fact, by using two disk controllers in the system you can retain both sets of disks on-line. How's that for upgrading without obsolescence?

Dick Grabowsky of HDE was kind enough to provide us with advance copies of the SYM Version of FODS (including source code!) and we have made the mods necessary to eliminate conflicts in page zero and elsewhere between MON, BAS, RAE, and FODS. Each language will be able to call any other in a manner natural to that language, and the calls will include

all patches to BAS and MON being published in SYM-PHYSIS. We also have suggested to Synertek Systems Corporation that they consider the use of FODS in their disk system planning because it exists and is very versatile. We are very pleased with FODS, and are using it, together with RAE to prepare this issue. The modifications are nearly complete, and the system works like a charm. We will keep you posted on the progress of the FODS MODS! Contact Hudson Digital Electronics, Box 120, Allamuchy, NJ 07820 for additional information (including price!). We plan to make available SYM software on FODS compatible Mini-Floppies in the future.

RAE NOTES (UPDATING SERVICE)

We have been using RAE-1 for many months now, since long before its official release in ROM, and have gotten to know it intimately. While we do not have access to the source code, we have disassembled it and studied its workings, and its author, Carl Moser, has answered all of our specific questions in detail, including the pages zero and one maps.

There are several instructions and features of RAE-1 which are not described in the Synertek Reference Manual, including printer vectors and disk vectors and commands. We are also adding patches to RAE, using its RUN command, to change baud rates and to switch between the RS-232 and the 20 mA ports as required, and to add text formatting capabilities. We had planned to describe these features in SYM-PHYSIS on a serial basis, but soon realized that these 'updates' would occupy too large a proportion of the newsletter if written in the detailed manner they deserved, and therefore had to be distributed as separate items. The updates will therefore be published separately as "RAE NOTES".

We will be mailing copies of RAE NOTES, as they are prepared, to all who purchase their RAE-1/2 from the Users' Group. RAE NOTES will be available to all others on a subscription basis. See back page.

RAE NOTES #1 will provide a handy "Reference Card", and describe how to find the last line number in the current file, how to adjust the label length to fit a 40-column CRT or printer, and how to set HA S to put the page numbers in the right place. RAE NOTES #2 will provide detailed pages 0 and 1 maps, including the disk and hard copy printing vectors. These issues are being prepared for printing now. RAE NOTES #3 will provide detailed information on RAE-to-Disk linkage. We hope that by RAE NOTES #4 some of our readers will be contributing results based on their own disassembly of RAE-1, to locate more of its user available subroutines, in order that RAE can be patched, in the same way that BAS-IC is patchable, to further enhance its capabilities.

PERIODICAL RECOMMENDATIONS

The 6502 USER NOTES are no more! Publication ended with Volume 3, Issue 5 (#17). We have not as yet received our final issue. All subscribers will receive a copy of Issue #2 (January/February, 1980) of COMPUTE, in place of #18. Eric Rehnke will write the section 'The Single-Board 6502', for COMPUTE. All back issues of the 6502 USER NOTES may be ordered from COMPUTE. COMPUTE is a new magazine covering the entire 6502 based computer community. The address is 900 Spring Garden St., Greensboro, NC 27403.

Issue #21 of Micro (February 1980) has an article by Robert A. Peck describing an inexpensive way to add an ASCII keyboard to SYM-1, and presents the required software.

Issue #2 of COMPUTE, (Jan/Feb. 1980) has an article by Jim Butterfield which gives BASIC page 0 maps for KIM, SYM, AIM, PET, and APPLE.

BOOK RECOMMENDATIONS

Add to the book recommendations printed in the Introductory Issue the following:

Leventhal, Lance: "6502 Assembly Language Programming", Osborne/McGraw-Hill Book Company, 1979.

We think so highly of this book that we feel every SYM owner should own a copy. It contains numerous useful programs which you can adapt immediately for your own purposes. See back page of this issue for ordering information.

Peck, Robert: "SYM-1 Hardware Theory of Operations Manual", 1979.

This is the second of three books he is writing for SYM-1. His third, "SYM-1 Monitor Theory of Operations Manual" is planned for release in March 1980. His first was the "Appendix to the First Book of KIM for the SYM", listed in the Introductory Issue. The price for the Hardware Theory Manual is \$6.00 U.S. delivered in the U.S. or Canada. Add \$1.25 U.S. for overseas delivery. The book is about 40 pages, includes a SYM-1 Schematic, and lists every signal available at each pin on the board. For a real understanding of how your SYM hardware is put together, this book will prove helpful. Order directly from Robert A. Peck, P. O. Box 2231, Sunnyvale, CA 94087.

SUPER BASIC

Several subscribers, including Sean McKenna, Gary Humphrey, and Jack Brown, working from their knowledge of Applesoft, KIM, and/or PET BASIC, have contributed articles on modifying SYM BASIC. Jack Brown, in particular, has submitted three extensively commented and well documented source codes, which are much too long to publish in their entirety. One of Jack's programs is a fantastically fast, two-pass, four parameter renumbering program, which takes advantage of the fact that each BASIC line is interpreted in a self-modifying section of code in page 0. BASIC line is interpreted in a self-modifying section of code in page 0, and patches in at that section. An uncommented version of that program is published here.

His second program is a terminal control patch which permits easy editing of BASIC lines, either current, or previously entered, by providing a set of cursor control commands. To quote Jack, and we agree with him, "With the editing capability of SUPER TCP and the versatility of the cursor extension command, SYM-1 now exceeds both Apple II and PET in editing and cursor control capability."

His third program provides for saving and loading both BASIC and 6502 files with names, and prints messages like SEARCHING FOR DATA, FOUND DATA, LOADING DATA, and SAVING DATA, where appropriate. This program is even more powerful than the PET equivalent. His second and third programs are patched to BASIC by trapping all carriage return outputs to see if they were prompted by detection of an error, and, if so, branching to the patch. This approach is much faster than the method used in the renumbering program, since it occurs only during output commands, when BASIC is being slowed down by the terminal output rate anyway.

Since these programs are so useful, too long to publish in SYM-PHYSIS, and require the full documentation and commenting for complete understanding, Jack has agreed to have us distribute them at a cost of \$.30 per page. This modest charge will pay for printing and mailing costs, and give Jack a nominal royalty for his efforts. See back page for ordering details. In the future, we will offer the documentation and complete source programs on cassette in RAE-1 format.

SYM-PHYSIS 2-3

ULTRA-RENUMBER FOR BAS-1

Here is an uncommented (to save space) source code version of Jack Brown's ULTRA-RENUMBER. So that it may be used in a 4K SYM it is located at 0C00-0FFF. Note that 0F00-0FFF is reserved as a working buffer for the line number tables. To use this program, enter BASIC with .J 0, answer MEMORY SIZE? with 3072, exit to MON with RST. Patch 4C 540E at 0ODD-0ODF, and reenter BASIC with .G 0. Whenever you wish to renumber a program use the command #LIST with up to four parameters (decimal) separated by commas, e.g.,

#LIST base, step, sline, hline (CR)

where base is the starting line number, step is the increment, and sline and hline are the (current) first and last line numbers of the segment of the BASIC program you wish to renumber. If the parameters are omitted, the default values are 100, 10, current lowest line number, current highest line number. ULTRA-RENUMBER will automatically take care of opening up (or closing up) spaces in case the new line number has more or fewer digits than the old line number! If, as sometimes happens, you requested a transfer to a non-existent line number, this will be flagged by the insertion of 65535. I think you will be so impressed with the speed and convenience of ULTRA-RENUMBER that you will want to relocate it together with the Tris Patch, and automate the patch at 0ODD in the same way as the patch for Tris Patch at 00C4 was automated in the TCP article in the Introductory Issue page 0-17.

```
OC00 20 D2 00 B0 4D 20 F5 C7,CB  
OC08 A5 1C 85 5A A5 1D 85 5B,0D  
OC10 20 D2 00 C9 2C D0 43 20,27  
OC18 CC 00 20 F5 C7 A5 1C 85,15  
OC20 A5 1C 85 5D 20 D2 00,07  
OC28 C9 2C D0 36 20 CC 00 20,0E  
OC30 F5 C7 A5 1C 85 5E A5 1D,30  
OC38 85 5F 20 D2 00 C9 2C D0,CB  
OC40 28 20 CC 00 20 F5 C7 A5,60  
OC48 1C 85 60 A5 1D 85 61 4C,55  
OC50 6F 0C A9 64 85 5A A9 00,65  
OC58 85 5B A9 0A 85 5C A9 00,82  
OC60 85 5D A2 01 86 5E CA 86,3B  
OC68 F9 FF 85 60 85 61 A9,B6  
OC70 FE 85 79 A9 0E 85 7A A9,11  
OC78 FC 85 58 A9 0E 85 59 A5,24  
OC80 7B 85 77 A5 7C 85 78 20,D9  
OC88 19 0E A0 03 B1 77 91 79,05  
OC90 88 B1 77 91 79 A0 03 B1,E3  
OC98 77 C5 5F F0 04 90 42 B0,F4  
OCA0 07 88 B1 77 C5 5E 90 39,97  
OCA8 A0 03 B1 77 C5 61 F0 04,7C  
OCB0 B0 1B 90 09 88 B1 77 C5,55  
OCB8 60 F0 02 B0 10 A0 03 B9,C3  
OCC0 AF 00 91 77 88 B9 AF 00,6A  
OCC8 91 77 20 22 0E A0 01 B1,14  
OCDD 77 F0 18 20 3B 0E AA 88,2E  
OCDB B1 77 85 77 86 78 4C 8A,26  
OCEO OC A5 79 85 58 A5 7A 85,D1  
OCE8 59 D0 E2 A9 FF C8 91 79,56  
OCFO C8 91 79 A5 7B 85 D3 A5,45  
OCFB 7C 85 D4 D0 03 20 45 0E,60  
7560  
OD00 20 45 0E D0 0A A0 C1 A9,57  
OD08 9E 20 54 C9 4C 23 C3 20,84  
OD10 45 0E 20 45 0E 20 45 0E,19  
OD18 AA F0 E2 A2 04 DD 4F 0E,19  
OD20 F0 05 CA D0 F8 F0 EE AS,23  
OD28 D3 48 A5 D4 48 20 CC 00,EB  
OD30 B0 E6 20 F5 C7 A5 1D C5,E4  
OD38 5F 90 4C F0 02 B0 06 A5,6C  
OD40 IC C5 5E 90 42 A5 1D C5,04  
OD48 61 F0 04 B0 3A 90 08 A5,80  
OD50 1C C5 60 F0 02 B0 30 20,B3  
OD58 92 0D 68 85 D4 68 85 D3,D3  
OD60 A0 00 A2 00 BD 01 01 F0,C4  
OD68 F0 48 20 CC 00 90 03 20,BA  
OD70 C9 0D 68 91 D3 E8 D0 EC,00  
OD78 20 CC 00 B0 0F 20 EB 0D,C3  
OD80 20 D2 00 90 F8 B0 05 68,5A  
OD88 68 20 D2 00 C9 2C F0 97,30  
OD90 D0 86 20 19 0E A5 58 85,4F  
OD98 79 A5 59 85 7A A0 05 B1,1B  
ODA0 79 C5 1D F0 15 C9 F0 D0,13  
ODA8 18 85 B2 85 B1 A5 B1 85,73  
ODB0 B3 A2 90 38 20 FF D9 4C,D4  
ODB8 9A DB 88 B1 79 C5 1C F0,CC  
ODC0 EC 20 22 0E 20 3B 0E D0,41  
ODC8 D4 20 08 0E A0 00 B1 77,13  
ODD0 C8 91 77 20 30 0E D0 08,19  
ODD8 E6 7D D0 02 E6 7E 88 60,9A  
ODE0 A4 77 D0 02 C6 78 C6 77,02  
ODE8 4C CC OD 20 0B 0E A0 01,FE  
ODF0 B1 79 88 91 79 20 30 0E,18  
ODF8 F0 05 20 3E 0E D0 EF A4,DC  
74DC
```

Hex Dump continues on bottom of Page 8- Souce Code Page 5

SYM-PHYSIS 2-4

ASSEMBLE LIST

```

0010 ;ULTRA-RENUMBER          0C32- A5 1C    0620      LDA *LINNUM        0CB2- 90 09    1250      BCC COPYR
0020 ;THE ULTIMATE SYM-1       0C34- 85 5E    0630      STA *SLINE         0CB4- 88     1260      CHKHLD DEY
0030 ;BASIC RENUMBER PROGRAM   0C36- A5 1D    0640      LDA *LINNUM+1     0CB5- B1 77    1270      LDA (INDEXA),Y
0040 ;                           0C38- 85 5F    0650      STA *SLINE+1      0CB7- C5 60    1280      CMP *HLINE
0050 ;COPYRIGHT 1979 BY        0C3A- 20 D2 00  0660      JSR CHRGET        0CB9- F0 02    1290      BEQ COPYR
0060 ;J. W. BROWN             0C3D- C9 2C    0670      CMP #'           0CBB- B0 10    1300      BCS COPYN
0070 ;ALL RIGHTS RESERVED     0C3F- D0 28    0680      BNE DHLINE       0CBD- A0 03    1310      COPYR
0080 ;                           0C41- 20 CC 00  0690      JSR CHRGET        0CBF- B9 AF 00  1320      LDY #$03
0090 ;                           .BA $0C00     0700      JSR LINGET       0CC2- 91 77    1330      LDA FACTO-3,Y
0100 ;                           .DS          0710      LDA *LINNUM        0CC4- 88     1340      STA (INDEXA),Y
0110 ;                           .MC $5000     0720      STA *HLINE         0CC5- B9 AF 00  1350      DEY
0120 NWSTRT      .DE $005B     0730      LDA *LINNUM+1     0CC8- 91 77    1360      LDA FACTO-3,Y
0130 BEGIN       .DE $005A     0740      STA *HLINE+1      0CCA- 20 22 0E  1370      STA (INDEXA),Y
0140 STEP        .DE $005C     0750      JMP COPY          0CCD- A0 01    1380      ADDSTP
0150 SLINE       .DE $005E     0760      DEFALT          0CCF- B1 77    1390      LDY #$01
0160 HLINE       .DE $0060     0770      LDA #$64          0CD1- F0 18    1400      LDA (INDEXA),Y
0170 VARTAB      .DE $007D     0780      STA *BEGIN        0CD3- 20 3B 0E  1410      BEQ COPYE
0180 FACTO      .DE $00B2     0790      LDA #$00          0CD6- AA     1420      JSR BUFBX
0190 LINNUM      .DE $001C     0800      STA *STEP          0CD7- 88     1430      TAX
0200 TXTTAB      .DE $007B     0810      DSTEP            0CD8- B1 77    1440      DEY
0210 TXTPTR      .DE $00D3     0820      LDA #$0A          0CDA- 85 77    1450      LDA (INDEXA),Y
0220 INDEXA      .DE $0077     0830      STA *STEP+1      0CDC- 86 78    1460      STA #INDEXA
0230 INDEXB      .DE $0079     0840      DSTEP            0CDE- 4C 8A OC  1470      STX *INDEXA+1
0240 WORKBF      .DI NUM+$0300 0850      LDX #$01          0CE1- A5 79    1480      COPYM
0250 ;(RELOCATE WORKBF WHEN 0860      STX *SLINE        0CE3- 85 58    1490      LDA #INDEXB
0260 ;RELOCATING PROGRAM)   0870      DEX              0CE5- A5 7A    1500      STA *NWSTRT+1
0270 ;                           .DE $00CC     0880      STX *SLINE+1     0CE7- 85 59    1510      BNE COPYN
0280 CHRGET      .DE $00D2     0890      LDA #$FF          0CE9- D0 E2    1520      LDA #$FF
0290 CHRGT       .DE $00D2     0900      STA *HLINE        0CEB- A9 FF    1530      COPYE
0300 FIXLNX      .DE $C323     0910      STA *HLINE+1     0CED- C8     1540      INY
0310 LINGET      .DE $C7F5     0920      LDA #WORKBF-2    0CEE- 91 79    1550      STA (INDEXB),Y
0320 FLDATC      .DE $D9FF     0930      STA *INDEXB      0CF0- C8     1560      INY
0330 FOUT        .DE $DB9A     0940      LDA #HWORKBF-2   0CF1- 91 79    1570      STA (INDEXB),Y
0340 MESSUB      .DE $C954     0950      STA *INDEXB+1    0CF3- A5 7B    1580      RENUM
0350 ;                           .DE $00DD     0960      LDA #WORKBF-4    0CF5- 85 D3    1590      LDA *TXTTAB
0360 BREAKIN     .DE $00E0     0970      STA *HWORKBF-4   0CF7- A5 7C    1600      STA *TXTPTR
0370 BACK        .DE $00E0     0980      STA *NWSTRT+1    0CF9- 85 D4    1610      LDA *TXTTAB+1
0380 ERMESS       .DE $C25A     0990      LDA *TXTTAB      0CFB- D0 03    1620      STA (INDEXB),Y
0390 GVARAB      .DE $CE63     1000      STA *INDEXA      0CFD- 20 45 0E  1630      RENA
0400 ;                           .DE $00E0     1010      LDA *TXTTAB+1    0D00- 20 45 0E  1640      GRAB
0410 NUM         JSR CHRGT       1020      STA *INDEXA+1    0D03- D0 0A    1650      JSR GRAB
0420 DEFALT      BCS DEFALT    1030      JSR SETFAC      0D05- A0 C1    1660      BNE RENB
0430 LINGET      JSR LINGET     1040      LDY #$03          0D07- A9 9E    1670      JSR MESSUB
0440 LINNUM      LDA *LINNUM     1050      LDA (INDEXA),Y  0D09- 20 54 C9  1680      JMP FIXLNX
0450 BEGIN       STA *BEGIN      1060      STA (INDEXB),Y  0DOC- 4C 23 C3  1690      JSR GRAB
0460 LINNUM      LDA *LINNUM+1  1070      DEY              0DOF- 20 45 0E  1700      RENC
0470 BEGIN       STA *BEGIN+1   1080      LDA (INDEXA),Y  0D12- 20 45 0E  1710      JSR GRAB
0480 CHRGT       JSR CHRGT      1090      STA (INDEXB),Y  0D15- 20 45 0E  1720      REND
0490 CMP #'       0C95- A0 03    1100      LDY #$03          0D18- AA     1730      TAX
0500 DSTEP       BNE DSTEP      1110      LDA (INDEXA),Y  0D19- F0 E2    1740      BEQ RENA
0510 CHRGET      JSR CHRGET     1120      CMP *SLINE+1    0D1B- A2 04    1750      LDX #$04
0520 LINGET      JSR LINGET     1130      BEQ CHKSLD      0D1D- DD 4F 0E  1760      CMP TOKEN-1,X
0530 LINNUM      LDA *LINNUM     1140      BCC COPYM      0D20- F0 05    1770      BEQ RENH
0540 STEP        STA *STEP      1150      BCS COPYC      0D22- CA     1780      DEX
0550 LINNUM      LDA *LINNUM+1  1160      DEY              0D23- D0 F8    1790      BNE RENG
0560 STEP        STA *STEP+1    1170      LDA (INDEXA),Y  0D25- F0 EE    1800      BEQ REND
0570 CHRGT       JSR CHRGT      1180      CMP *SLINE      0D27- A5 D3    1810      LDA *TXTPTR
0580 CMP #'       0CA4- C5 5E    1190      BCC COPYM      0D29- 48     1820      PHA
0590 DSLINE      BNE DSLINE     1200      LDY #$03          0D2A- A5 D4  1830      LDA *TXTPTR+1
0600 CHRGET      JSR CHRGET     1210      LDA (INDEXA),Y  0D2C- 48     1840      PHA
0610 LINGET      JSR LINGET     1220      CMP *HLINE+1    0D2D- 20 CC 00  1850      JSR CHRGET
                                1230      BEQ CHKHLD   0D30- B0 E6    1860      BCS RENE
                                1240      BCS COPYN   0D32- 20 F5 C7  1870      JSR LINGET

```

0D35- A5 10	1880	LDA *LINNUM+1	0DB4- 20 FF D9	2510	JSR FLOATC	0E32- C5 79	3140	CMP *INDEXB
0D37- C5 5F	1890	CMP *SLINE+1	0DB7- 4C 9A DB	2520	JMP FOUT	0E34- D0 04	3150	BNE CRTS
0D39- 90 4C	1900	BCC RENF	0DBA- B8	2530 FNC	DEY	0E36- A5 78	3160	LDA *INDEXA+1
0D3B- F0 02	1910	BEQ SLOCHK	0DBB- B1 79	2540	LDA (INDEXB),Y	0E38- C5 7A	3170	CMP *INDEXB+1
0D3D- B0 06	1920	BCS RENR	0DBD- C5 1C	2550	CMP *LINNUM	0E3A- 60	3180 CRTS	RTS
0D3F- A5 1C	1930 SLOCHK	LDA *LINNUM	0DBF- F0 EC	2560	BEQ FNB	0E3B- 20 3E OE	3190 BUPXB	JSR BUPXA
0D41- C5 5E	1940	CMP *SLINE	0DC1- 20 22 0E	2570 FND	JSR ADDSTP	0E3E- E6 79	3200 BUPXA	INC *INDEXB
0D43- 90 42	1950	RCC RENF	0DC4- 20 3B 0E	2580	JSR BUPXB	0E40- D0 02	3210	BNE BRTS
0D45- A5 1D	1960 RENR	LDA *LINNUM+1	0DC7- D0 D4	2590	BNE FNA	0E42- E6 7A	3220	INC *INDEXB+1
0D47- C5 61	1970	CMP *HLINE+1	0DC9- 20 08 0E	2600 MOVUP	JSR SETPTR	0E44- 60	3230 BRTS	RTS
0D49- F0 04	1980	BEQ HLOCHK	0DCC- A0 00	2610 MUA	LDY #\$00	0E45- A0 00	3240 GRAB	LDY \$00
0D4B- B0 3A	1990	BCS RENF	0DCE- B1 77	2620	LDA (INDEXA),Y	0E47- E6 D3	3250	INC *TXTPTR
0D4D- 90 08	2000	BCC RENS	0DD0- CB	2630	INY	0E49- D0 02	3260	BNE GRA
0D4F- A5 1C	2010 HLOCHK	LDA *LINNUM	0DD1- 91 77	2640	STA (INDEXA),Y	0E4B- E6 D4	3270	INC *TXTPTR+1
0D51- C5 60	2020	CMP *HLINE	0DD3- 20 30 0E	2650	JSR CMPX	0E4D- B1 D3	3280 GRA	LDA (TXTPTR),Y
0D53- F0 02	2030	BEQ RENS	0DD6- D0 08	2660	BNE MUC	0E4F- 60	3290	RTS
0D55- B0 30	2040	BCS RENF	0DD8- E6 7D	2670	INC *VARTAB	0E50- 88 89	3300 TOKEN	.BY \$88 \$89
0D57- 20 92 0D	2050 RENS	JSR FINUM	0DDA- D0 02	2680	BNE MUB	0E52- BC A1	3310	.BY \$8C \$A1
0D5A- 68	2060	PLA	0DDC- E6 7E	2690	INC *VARTAB+1	0E54- C9 23	3320 NUMPATCH	CMP #'
0D5B- 85 D4	2070	STA *TXTPTR+1	0DDE- 88	2700 MUR	DEY	0E56- F0 06	3330	BEQ CHECK
0D5D- 68	2080	PLA	0DDF- 60	2710	RTS	0E58- 38	3340 RESTR	SEC
0D5E- 85 D3	2090	STA *TXTPTR	0DE0- A4 77	2720 MUC	LDY *INDEXA	0E59- E9 30	3350	SBC #\$30
0D60- A0 00	2100	LDY #\$00	0DE2- D0 02	2730	BNE MUD	0E5B- 4C E0 00	3360	JMP BACK
0D62- A2 00	2110	LDX #\$00	0DE4- C6 78	2740	DEC *INDEXA+1	0E5E- 68	3370 CHECK	PLA
0D64- BD 01 01	2120 RENI	LDA \$0101,X	0DE6- C6 77	2750 MUD	DEC *INDEXA	0E5F- 48	3380	PHA
0D67- F0 0F	2130	BEQ RENK	0DE8- 4C CC OD	2760	JMP MUA	0E60- C9 63	3390	CMP #GVARAD
0D69- 48	2140	PHA	0DEB- 20 08 0E	2770 MOVDWN	JSR SETPTR	0E62- F0 04	3400	BEG CHECKA
0D6A- 20 CC 00	2150	JSR CHRGET	0DEE- A0 01	2780 MDA	LDY #\$01	0E64- A9 23	3410 RESTRA	LDA #'
0D6D- 90 03	2160	BCC RENJ	0DF0- B1 79	2790	LDA (INDEXB),Y	0E66- D0 F0	3420	BNE RESTR
0D6F- 20 C9 0D	2170	JSR MOVUP	0DF2- 88	2800	DEY	0E68- 68	3430 CHECKA	PLA
0D72- 68	2180 RENJ	PLA	0DF3- 91 79	2810	STA (INDEXB),Y	0E69- 68	3440	PLA
0D73- 91 D3	2190	STA (TXTPTR),Y	0DF5- 20 30 0E	2820	JSR CMPX	0E6A- 48	3450	PHA
0D75- E8	2200	INX	0DFB- F0 05	2830	BEQ MDC	0E6B- C9 CE	3460	CMP #H, GVARAD
0D76- D0 EC	2210	BNE RENI	0DFA- 20 3E 0E	2840 MDB	JSR BUPXA	0E6D- F0 05	3470	BEQ CHECKB
0D78- 20 CC 00	2220 RENK	JSR CHRGET	0DFD- D0 EF	2850	BNE MDA	0E6F- A9 63	3480	LDA #GVARAD
0D7B- B0 0F	2230	BCS RENZ	0DFF- A4 7D	2860 MDC	LDY *VARTAB	0E71- 48	3490	PHA
0D7D- 20 EB 0D	2240 RENM	JSR MOVDWN	0E01- D0 02	2870	BNE MDD	0E72- D0 F0	3500	BNE RESTRA
0D80- 20 D2 00	2250	JSR CHRGOT	0E03- C6 7E	2880	DEC *VARTAB+1	0E74- 68	3510 CHECKB	PLA
0D83- 90 FB	2260	BCC RENM	0E05- C6 7D	2890 MDD	DEC *VARTAB	0E75- 68	3520	PLA
0D85- B0 05	2270	BCS RENZ	0E07- 60	2900 MDE	RTS	0E76- 68	3530	PLA
0D87- 68	2280 RENF	PLA	0E08- A5 7D	2910 SETPTR	LDA *VARTAB	0E77- A9 0B	3540	LDA #H, NUM-1
0D88- 68	2290	PLA	0EOA- 85 77	2920	STA *INDEXA	0E79- 48	3550	PHA
0D89- 20 D2 00	2300	JSR CHRGOT	0EOC- A5 7E	2930	STA *VARTAB+1	0E7A- A9 FF	3560	LDA #NUM-1
0D8C- C9 2C	2310 RENZ	CMP #'	0EE0- 85 78	2940	STA *INDEXA+1	0E7C- 48	3570	PHA
0D8E- F0 97	2320	BEQ RENH	0E10- A5 D3	2950	LDA *TXTPTR	0E7D- 20 CC 00	3580	JSR CHRGET
0D90- D0 86	2330	BNE RENE	0E12- 85 79	2960	STA *INDEXB	0E80- C9 99	3590	CMP #99
0D92- 20 19 0E	2340 FINUM	JSR SETFAC	0E14- A5 D4	2970	LDA *TXTPTR+1	0E82- D0 03	3600	BNE ERROR
0D95- A5 58	2350	LDA *NWSTRT	0E16- 85 7A	2980	STA *INDEXB+1	0E84- 4C CC 00	3610	JMP CHRGET
0D97- 85 79	2360	STA *INDEXB	0E18- 60	2990	RTS	0E87- A2 02	3620 ERROR	LDX #\$02
0D99- A5 59	2370	LDA *NWSTRT+1	0E19- A5 5A	3000 SETFAC	LDA *BEGIN	0E88- 4C 5A C2	3630	JMP ERMESS
0D9B- 85 7A	2380	STA *INDEXB+1	0E1B- 85 B1	3010	STA *FACTO-1	3640	.EN	
0D9D- A0 05	2390 FNA	LDY #\$05	0E1D- A5 5B	3020	LDA *BEGIN+1			
0D9F- B1 79	2400	LDA (INDEXB),Y	0E1F- 85 B2	3030	STA *FACTO			
0DA1- C5 1D	2410	CMP *LINNUM+1	0E21- 60	3040	RTS	0E00 7D D0 02 C6 7E C6 7D 60,36	OE50 88 89 BC A1 C9 23 F0 06,D3	
0DA3- F0 15	2420	BEQ FNC	0E22- A5 B1	3050 ADDSTP	LDA *FACTO-1	0E08 A5 7D 05 77 A5 7E 85 78,74	OE58 38 E9 30 4C E0 00 68 4B,00	
0DA5- C9 FF	2430	CMP \$\$FFF	0E24- 18	3060	CLC	0E10 A5 D3 85 79 A5 D4 85 7A,62	OE60 C9 63 F0 04 A9 23 D0 F0,AC	
0DA7- D0 18	2440	BNE FND	0E25- 65 5C	3070	ADC *STEP	0E18 60 A5 5A 85 B1 A5 5B 85,7C	OE68 68 6B 4B C9 CE F0 05 A9,F9	
0DA9- 85 B2	2450	STA *FACTO	0E27- 85 B1	3080	STA *FACTO-1	0E20 B2 60 A5 B1 18 65 5C 85,42	OE70 63 48 D0 F0 68 68 6B,A9,45	
0DAB- 85 B1	2460	STA *FACTO-1	0E29- A5 B2	3090	LDA *FACTO	0E28 B1 A5 B2 65 5D 85 B2 60,A3	OE78 0B 48 A9 FF 48 20 CC 00,74	
0DAD- A5 B1	2470 FNB	LDA *FACTO-1	0E2B- 65 5D	3100	ADC *STEP+1	0E30 A5 77 C5 79 D0 04 A5 78,EE	OE80 C9 99 D0 03 4C CC 00 A2,63	
0DAF- 85 B3	2480	STA *FACTO+1	0E2D- 85 B2	3110	STA *FACTO	0E38 C5 7A 60 20 3E 0E E6 79,58	OE88 02 4C 5A C2,CD	
0DB1- A2 90	2490	LDX #\$90	0E2F- 60	3120 ARTS	RTS	0E40 D0 02 E6 7A 60 A0 00 E6,70	47CD	
0DB3- 38	2500	SEC	0E30- A5 77	3130 CMPX	LDA *INDEXA	0E48 D3 D0 02 E6 D4 B1 D3 60,B3		

DOODLING WITH THE KTM-2

Phillip M. Rinard
2019 Park Ave.
Emporia, KS 66801

The KTM-2 keyboard terminal module provides a convenient way to add a video interface to a SYM-1 as well as a keyboard. Unlike some other computers (e.g., PET, SORCERER), it is not possible to push a single keyboard button and place a graphic character anywhere on the video display. When the keyboard is communicating with the computer you must originate cursor movements and character generation with computer instructions. You can use the local mode if you alter some input-output lines on the KTM-2 edge connector, but there is a software approach that offers some advantages.

The program listed below allows you to "doodle" on the video display by controlling the cursor's position and enabling any of the keyboard characters to be placed on the screen. A display can be constructed and altered until a pleasing final form is achieved. It could then be made into a subsequent program or simply admired before destroyed. With this software approach, fewer button pushings are needed to make a drawing than would be needed in the local mode.

When the program begins, the screen is cleared and the cursor placed in the upper-left corner (the HOME position). The initialization also assumes you will want the "reversed graphics" mode, but you can change this at any time if you wish. The SYM waits for your commands. The SUPERMON subroutine INCHR would have been used to accept a character from the keyboard except for the fact that it converts lower-case letters into upper-case letters and thus prevents half the graphic characters from being accepted. Fortunately, this lower-to-upper case conversion is only a small part of the subroutine so the solution is easy. The subroutine is copied out of the monitor (using the "block move" command), and then NOPs placed in the offending registers which are \$307-\$310 in the listing here.

After receiving your keyboard input, the SYM searches for its significance. It first checks for a cursor control instruction. The keyboard numerals "1", "2", "3" and "4" each move the cursor one space from its present location; the motion is left, up, down and right, respectively. A "3" with the cursor on the bottom row will scroll the whole screen, not move the cursor.

For larger movements, "9", "0", "+" and "=" are available. The "9" key is a carriage return and "0" is a "HOME" instruction. The KTM-2 provides for "relative" and "absolute" addressing of the cursor, as explained in the manual, and the "+" and "=" keys begin those operations. To move the cursor five rows and ten columns from its present location with this program, type "+", "05" and "10". An "ESC" command must precede the "+", but with the SYM in control it is done automatically for you. The full byte "05" must be used, and base ten numbers must be entered; thus, the "10" is really a ten, not a sixteen. An example of absolute addressing with this program would be "=", "32", "28"; the cursor would jump to row 32 and column 28. A preceding "ESC" would again be supplied by the SYM.

As the cursor moves across an already-existing character on the display, it erases the character. Using the carriage return or home command leaves the screen unchanged and thus can get it out of a "tight" spot without erasing part of the doodle.

Should your keyboard entry not imply a cursor movement, the program next checks it against the graphic symbols available as part of the KTM-2. If you have typed in "n", for example, it will place the "club" suit symbol on the screen and the cursor moves one position to the right. If you type "N" you will get a thin horizontal line. It is assumed that you are still in the reversed graphics mode (called R,G in the manual).

Perhaps you would like an actual "N" and not its corresponding graphic character. You need to be in the r,g mode. This program allows you to form any combination of the G, R, g and r modes you wish by typing "5", "6", "7" and "8", respectively. To get the "N", therefore, type "7", "8" and then

"N". To return to the reversed graphics mode, type "5" and "6". In the g,R mode, you get an "N" with the black and white interchanged; with G,r you get the graphic symbol with black and white interchanged.

Certain keys (especially punctuation marks) are not assigned any graphics characters and are simply ignored in that respect. Pushing the space bar does not move the cursor one position to the right. The "SHIFT SPACE-BAR", however, is a valid graphic symbol and recognized as such.

The numerals can not be placed on the display since they are cursor control keys.

This program makes use of the KTM-2 features in a manner that is easier to use than in the local mode. The numerals control the cursor and the mode, while the alphabetical keys (and a few others) create the actual symbols. A sketch of the keyboard with the functions or symbols shown in place of the alphanumeric characters helps the user find the desired key.

After the program listing is a table of "data" for use with the program. It displays a 15x21 picture of a well-known U.S. president out of only six different characters. The top row of the table forms the top row of the picture, and so on. The "1" through "6" in the table correspond to these keyboard characters in the G,R mode: "\", "SHIFT LINE-FEED", "SHIFT ALPHA", "]", "SHIFT SPACE-BAR", and "SHIFT CARRIAGE-RETURN", respectively. If you don't recognize this man immediately, step back a few feet and squint. After finding a pleasing set of characters such as those suggested here, the display can be made into a program to be

put onto the screen automatically on future occasions.

0200 20 86 88 A9 18 20 4/ 8A±6	0280 8A A5 06 20 47 8A 4C 15,F3
0208 A9 45 20 47 8A A9 47 85,3A	0288 02 C9 40 30 2F C9 80 10,B6
0210 08 A9 52 85 09 A0 00 20,BB	0290 2B 85 07 A9 18 20 47 8A,22
0218 FF 02 85 07 20 EF 02 A5,CE	0298 A5 08 20 47 8A A9 1B 20,A4
0220 07 D9 CA 03 F0 OF C8 C0,02	02A0 47 8A A5 09 20 47 8A A5,B9
0228 05 D0 F6 D9 CA 03 D0 0E,51	02A8 07 20 47 8A A2 00 BD D6,E6
0230 A9 1B 20 47 8A B9 D0 03,92	02B0 03 20 47 8A EB E0 04 D0,76
0238 20 47 8A 4C 15 02 C9 2B,DA	02B8 F5 4C 15 02 A2 00 BD DA,07
0240 D0 04 A9 00 10 06 C9 3D,73	02C0 03 C5 07 F0 08 EB E0 04,9A
0248 D0 3F A9 80 85 07 20 D9,30	02C8 D0 F4 4C 15 02 BD DE 03,5F
0250 B1 20 E1 02 85 05 20 EF,4D	02D0 C9 52 F0 08 C9 72 F0 04,A1
0258 02 20 EF 02 20 D9 81 20,FA	02D8 85 08 F0 02 85 09 4C 15,0F
0260 E1 02 85 06 20 EF 02 20,99	02E0 02 A2 20 F8 3B E9 01 30,1D
0268 EF 02 A9 1B 20 47 8A 24,63	02E8 03 EB 10 F8 D8 8A 60 A9,7B
0270 07 30 04 A9 2B 10 02 A9,2D	02F0 08 20 47 BA A9 20 20 47,A4
0278 3D 20 47 BA A5 05 20 47,6C	02F8 8A A9 08 20 47 BA 60 20,50

0300 88 81 20 41 8A 29 7F EA,86	0378 8A 18 20 D4 8A A5 F9 49,DC
0308 EA EA EA EA EA EA EA,D6	0380 FF 4C B8 81 85 F9 20 88,86
0310 EA C9 0F D0 0B AD 53 A6,19	0388 81 20 E9 8A A9 30 8D 03,03
0318 49 40 8D 53 A6 18 90 E2,B2	0390 A4 A5 F9 A2 0B 49 FF 38,72
0320 C9 0D 4C B8 81 6C 61 A6,80	0398 20 D4 8A 20 E6 8A A0 06,26
0328 20 09 83 20 88 81 2C 53,D4	03A0 8B D0 FD EA 4A CA D0 F0,39
0330 A6 70 03 20 55 8A 4C C4,FC	03A8 A5 F9 C9 D0 F0 04 C9 0A,74
0338 81 6C 64 A6 20 88 81 A9,C5	03B0 D0 03 20 32 8B 4C C4 81,B5
0340 00 B5 F9 AD 02 A4 2D 54,17	03B8 4B AD 02 A4 29 0F 90 02,1A
0348 A6 38 E9 40 90 F5 20 E9,AC	03C0 09 30 2D 54 A6 8D 02 A4,AD
0350 8A AD 02 A4 2D 54 A6 38,EB	03C8 68 60 31 34 32 33 39 30,AB
0358 E9 40 2C 53 A6 10 06 20,6C	03D0 08 09 0B 0A 0D 48 1B 67,AS
0360 D4 8A 4C 87 8A A0 07 88,56	03D8 1B 72 35 36 37 38 47 52,AS
0368 D0 FD EA 66 F9 20 E9 BA,FF	03E0 67 72,7E
0370 48 B5 00 68 90 D8 20 E9,D5	677E

200 20 86 88	JSR ACCESS	Write-enable the system RAM.
203 A9 1B	LDAIM	Clear the screen
205 20 47 8A	JSR OUTCHR	with ESC-E.
208 A9 45	LDAIM	
20A 20 47 8A	JSR OUTCHR	
20D A9 47	LDAIM	Begin in reversed graphics
20F 85 08	STAZ	with G and R.
211 A9 52	LDAIM	
213 85 09	STAZ	
215 A0 00	START	LDYIM Initialize command table index.
217 20 FF 02	JSR INCH	Obtain a keyboard entry
21A 85 07	STAZ	and store it.
21C 20 EF 02	JSR BACKUP	Erase the entry from the screen.
21F A5 07	LDAZ	Compare entry with table of cursor commands.
221 D9 CA 03	NXTCOM	
224 F0 OF	CMP,AY	
226 C8	BEQ OUTCOM	
227 C0 05	INY	
229 D0 F6	CPYIM	
22B D9 CA 03	BNE NXTCOM	
22E D0 0E	CMP,AY	Is the command "HOME"?
230 A9 1B	BNE RELA	Not a simple cursor command at all.
232 20 47 8A	LDAIM	Use "ESC" before the "HOME" command.
235 B9 D0 03	OUTCOM	Send the "ESC" out.
238 20 47 8A	LDA,AY	Get the command's ASCII code.
23B 4C 15 02	JSR OUTCHR	Send the command out.
23E C9 2B	JMP START	Return for next keyboard entry.
240 D0 04	CMPIM	Is the entry "+" for relative addressing?
242 A9 00	BNE ABS	If not, try absolute addressing.
244 10 06	LDAIM	Temporary code for "+".
246 C9 3D	BPL DATA	Jump to a storage command.
248 D0 3F	CMPIM	Is the entry "=" for absolute addressing?
24A A9 80	BNE NXTSYM	If not, check the symbol table.
24C 85 07	DATA	Temporary code for "=".
24E 20 D9 81	STAZ	Store the temporary code.
251 20 E1 02	JSR INBYTE	Get the new row or the change in the row value.
254 85 05	JSR DECHEX	Change byte into a hex number for the KTM-2
256 20 EF 02	STAZ	and store it.
259 20 EF 02	JSR BACKUP	Erase the byte
25C 20 D9 81	JSR INBYTE	from the screen.
25F 20 E1 02	JSR DECHEX	Get the new column or change in column value.
262 85 06	STAZ	Change it into a hex number for the KTM-2
264 20 EF 02	JSR BACKUP	and store it.
267 20 EF 02	JSR BACKUP	Erase the byte
26A A9 1B	LDAIM	from the screen.
		Send out

26C	20 47 8A		JSR OUTCHR	ESC +		2EF	A9 08	BACKUP	LDAIM	The cursor is backspaced once,
26F	24 07		BITZ	or		2F1	20 47 8A		JSR OUTCHR	a space is placed on the screen
271	30 04		BMI EQUAL	ESC =		2F4	A9 20		LDAIM	to erase the latest entry,
273	A9 2B		LDAIM	and then		2F6	20 47 8A		JSR OUTCHR	and another backspace occurs,
275	10 02		BPL OUT	the row		2F9	A9 08		LDAIM	leaving the cursor in its original
277	A9 3D	EQUAL	LDAIM	and		2FB	20 47 8A		JSR OUTCHR	position.
279	20 47 8A	OUT	JSR OUTCHR	the column		2FE	60		RTS	
27C	A5 05		LDAZ	information.		2FF	20 88 81	INCH	JSR SAVER	This is the SYM-1 SUPERMON subroutine
27E	20 47 8A		JSR OUTCHR			302	20 41 8A		JSR INJINV	INCHR without the portion that converts
281	A5 06		LDAZ			305	29 7F		AND #\$7F	lower-case letters into upper-case letters.
283	20 47 8A		JSR OUTCHR			307	EA EA		NOP	
286	4C 15 02		JMP START	Get the next keyboard entry.		309	EA EA		NOP	
289	C9 40	NXTSYM	CMPIM	Check that the		30B	EA EA		NOP	
288	30 2F		BMI NIX	keyboard entry		30D	EA EA		NOP	
28D	C9 80		CMPIM	is a valid ASCII		30F	EA EA		NOP	
28F	10 2B		BPL NIX	symbol (40 to 7F).		311	C9 0F	INRT1	CMP #\$0F	
291	85 07		STAZ	Temporarily store the symbol.		313	DO 0B		BNE INRT2	
293	A9 1B		LDAIM	Get into graphics mode		315	AD 53 A6		LDA TECHO	
295	20 47 8A		JSR OUTCHR	with ESC G, ESC R		318	49 40		EOR #\$40	
298	A5 08		LDAZ	or leave graphics mode		31A	BD 53 A6		STA TECHO	
29A	20 47 8A		JSR OUTCHR	with ESC g, ESC r		31D	18		CLC	
29D	A9 1B		LDAIM	or use unreversed graphics		31E	90 E2		BCC INCHR+3	
29F	20 47 8A		JSR OUTCHR	with ESC G, ESC r.		320	C9 0D	INRT2	CMP #\$0D	
2A2	A5 09		LDAZ			322	4C 88 81		JMP RESXZF	
2A4	20 47 8A		JSR OUTCHR			325	6C 61 A6	INJINV	JMP (INVEC+1)	
2A7	A5 07		LDAZ	Retrieve the symbol		328	20 09 83		NBASOC	
2A9	20 47 8A		JSR OUTCHR	and send it to the screen.		32B	20 88 81		JSR NTBASC	37D A5 F9
2AC	A2 00		LDXIM	Always return to the normal		32B	20 88 81		JSR SAVER	EOR #\$FF
2AE	BD D6 03	NEXT1	LDA,AX	display mode (g,r).		32E	2C 53 A6			JMP RESXAF
2B1	20 47 8A		JSR OUTCHR			331	70 03		BIT TECHO	
2B4	E8		INX			333	20 55 8A		BVS *+5	
2B5	E0 04		CMPIM			336	4C C4 81		JSR INJOUV	
2B7	D0 F5		BNE NEXT1			339	6C 64 A6		JMP RECALL	
2B9	4C 15 02		JMP START	Get the next keyboard entry.		33C	20 88 81	INTCHR	JSR SAVER	
2BC	A2 00	NIX	LDXIM	Check that the keyboard entry		33F	A9 00		LDA #0	TOUT
2BE	BD DA 03	NEXT2	LDA,AX	was a G,R,g, or r		341	85 F9		STA \$F9	JSR DLYH
2C1	C5 07		CMPZ	encoded as a 5,6,7, or 8.		343	AD 02 A4	LOOK	INTCHR	
2C3	F0 08		BEQ GRAPH			346	2D 54 A6		LDA PBDA	
2C5	E8		INX			349	38		AND TOUTFL	
2C6	E0 04		CPX,IM			34A	E9 40		SEC	
2C8	D0 F4		BNE NEXT2			34C	90 F5		SBC #\$40	
2CA	4C 15 02		JMP START	If not, get another keyboard entry.		34E	20 E9 8A	TIN	BCC LOOK	
2CD	BD DE 03	GRAPH	LDXIM	Determine which command it is and		351	AD 02 A4		JSR DLYH	
2D0	C9 52		CMPIM	place it in the proper memory		354	2D 54 A6		LDA PBDA	
2D2	F0 08		BEQ RR	location to be used later.		357	38		AND TOUTFL	
2D4	C9 72		CMPIM			358	E9 40		SEC	
2D6	F0 04		BEQ			35A	2C 53 A6		SBC #\$40	
2D8	85 08		STAZ			35D	10 06		BIT TECHO	
2DA	F0 02		BEQ STRT			35F	20 D4 8A		BPL DMY1	
2DC	85 09	RR	STAZ			362	4C 87 8A		JSR OUT	
2DE	4C 15 02	STRT	JMP START	Get another keyboard entry.		365	AO 07		JMP SAVE	
2E1	A2 20	DECHEX	LDXIM	A decimal byte in A is converted to		367	88	DMY1	LDY #7	
2E3	F8		SED	a hex byte to complete the KTM-2		368	DO FD	TLP1	DEY	
2E4	38	NEXT	SEC	addressing command later.		36A	EA		BNE TLP1	
2E5	E9 01		SBCIM			36B	66 F9		NOP	
2E7	30 03		BMI QUIT			36D	20 E9 8A	SAVE	ROR \$F9	
2E9	E8		INX			370	48		JSR DLYH	
2EA	10 F8		BPL NEXT			371	B5 00		PHA	
2EC	D8	QUIT	CLD			373	68		LDA O,X	
2ED	8A		TXA			374	90 D8		PLA	
2EE	60		RTS			376	20 E9 8A		BCC TIN	
						379	18		JSR DLYH	
						37A	20 D4 8A		CLC	
									JSR OUT	

3CA 31	CURSOR	ASCII 1 Move cursor left. Input commands.
3CB 34		ASCII 4 Move cursor right.
3CC 32		ASCII 2 Move cursor up.
3CD 33		ASCII 3 Move cursor down.
3CE 39		ASCII 9 Carriage return.
3CF 30		ASCII 0 Home position.
3D0 08	COMMANDS	ASCII BACK SPACE Output commands.
3D1 09		ASCII HORIZONTAL TAB
3D2 0B		ASCII VERTICAL TAB
3D3 0A		ASCII LINE FEED
3D4 0D		ASCII CARRIAGE RETURN
3D5 48		ASCII H FOR HOME
3D6 1B	MODE	ASCII ESCAPE Data to leave graphics mode.
3D7 67		ASCII g
3D8 1B		ASCII ESCAPE
3D9 72		ASCII r
3DA 35	CHANGE	ASCII 5 Input commands to change graphics mode.
3DB 36		ASCII 6
3DC 37		ASCII 7
3DD 38		ASCII 8
3DE 47		ASCII G Output commands to change graphics mode.
3DF 52		ASCII R
3E0 67		ASCII g
3E1 72		ASCII r

TABLE OF "DATA" FOR A U.S. PRESIDENT

Top Row	6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6	6 6 2 3 1 4 2 2 2 2 2 3 2 6 6
	6 6 6 6 6 1 3 4 3 6 6 6 6 6	6 6 1 4 1 5 4 2 2 3 3 4 2 6 6
	6 6 6 6 3 5 5 5 5 4 6 6 6 6	6 6 1 2 5 4 4 2 2 1 4 3 6 6 6
	6 6 6 6 4 5 3 2 5 5 5 6 6 6	6 6 2 2 3 5 4 1 2 3 5 4 6 6 6
	6 6 6 3 5 4 1 1 1 2 5 5 1 6 6	6 2 2 2 3 4 5 4 4 3 4 3 6 6 6
	6 6 6 4 5 3 1 1 1 1 4 4 4 6 6	6 2 2 2 5 2 4 5 5 5 5 2 6 6 6
	6 6 6 5 5 4 2 1 1 1 2 2 5 6 6	6 2 3 4 5 4 1 2 4 3 2 2 6 6 6
	6 6 4 5 5 4 2 4 5 3 3 5 6 6	6 2 5 5 4 5 4 5 3 3 2 5 1 6 6
	6 6 4 2 4 4 2 3 4 3 4 5 2 6 6	6 4 5 5 5 5 5 5 4 4 5 5 5 4 6
Bottom Row	6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6	6 3 5 5 5 5 5 5 3 3 4 4 5 5 6
		6 4 5 5 5 5 5 5 5 1 1 1 1 5 6

SOFTWARE RECOMMENDATION

For those who wish to expand their SYM-1's beyond the single board configuration we recommend adding some kind of terminal capability. Once you have full alphanumeric I/O capabilities you will want some kind of software on which to exercise your system. For the assembly language programmers we recommend the 2KSA (see Issues #0, and #1). If you want to get into higher level languages inexpensively, consider Tom Pittman's TINY BASIC. It is comparable to the integer BASIC in the TRS-80 and the Apple II, and, being in RAM, is easily modified to meet your needs. It occupies only a little over 2K of RAM. The Users' Group has available an extended SYM-1 version of TINY BASIC on cassette. This version has such added features as SAVE, LOAD, CHARIN, CHAROUT, and string STORE and RECALL, as well as GOMON, BEEP, PEEK, and POKE. It links easily with machine language programs and is useful for control applications. The cassette is sold only with a copy of Pittman's TINY BASIC Manual. Also available is Pittman's Experimenter's Kit, needed if you expect to make additional modifications. The combination of TB and 2KSA make for a very powerful 'small system' at low cost. To order, see back page.

EDITOR'S NOTES

As of this date, over 12,000 SYMs have been sold, and we have over 500 subscribers, with more than a dozen new subscribers each week. We have received hundreds of pages of letters, manuscripts, and source listings, and it is becoming almost impossible to keep up with the correspondence. Where the writer had an immediate crisis, we tried to answer promptly; the rest of the writers may find their answers in this issue, or maybe in the next!

We have kept our promise on the graphics in this issue but will have to defer the music programs to the next issue. Until then we suggest that you read the article by Hal Chamberlin, "A Sampling of Techniques for Computer Performance of Music", in BYTE, Vol. 2, No. 9, Sept. 1977, also reprinted in "The BYTE Book of Computer Music". The D/A converter described in that article will form the basis for the computer music and oscilloscope graphics programs to appear in Issue #3.

FOR "SYM-PLE" SYMS

For those of you who are just beginning, here is a suggested way to go. After you have added your power supply (at least 1 A at 5 V regulated) and a cassette recorder, order a copy of "The First Book of KIM", and "The SYM/KIM Appendix to the FBOK", from Bob Peck. This should keep you busy for quite a while. We plan to publish in each issue at least one new program for these "SYM-ple" SYMs. You should also subscribe to one or more of the periodicals we have listed and set one or more of the books we have recommended. Readings and doing are essential if you wish to get the most out of your SYM.

Your first hardware expansion should be the addition of the extra 3K of RAM, available from some of the advertisers in the periodicals. If you decide to go any further you will need some kind of terminal, either an RS-232 device or a memory mapped display/keyboard system. These, too, are advertised in the magazines. If you intend to go into word processing or text editing you should get an 80 column terminal.

Once you have alphanumeric I/O, two good, inexpensive software additions are Bob Denison's 2K Symbolic Assembler, and Tom Pittman's 6502 TINY BASIC, both available from the Users' Group. After these, we recommend RAE-1 and BAS-1, in whichever order you prefer. You will then have 20K of ROM and 4K of RAM on-board. You can add 4K more of RAM with John Blalock's Memory Expansion Board (we understand that John is working on either a 16K or 32K dynamic memory expansion board which will mount directly above the ROM area on the SYM). By this time you will need a larger power supply, and you no longer are a "SYM-ple" SYMmer!

HARDWARE RECOMMENDATION

John Blalock, whose 4K Memory Expansion Board was described in Issue #0, now has available for sale ROM Socket Adapters, permitting 2-4K ROMs to be installed into a single socket on the SYM-1. If you order a pair of them, you can put both BAS-1 chips in one and both RAE-1/2 chips in the other, and still leave one socket free for an EPROM or another ROM. In order to use two of them you will have to move MON into socket U21, but that's easy to do while you are wiring all the other jumpers. The price is \$10.00. U.S. and Canada please add \$.50, Europe please add \$1.00, Asia and Pacific countries please add \$1.25 for First Class/Air Mail postage and handling. The 4K Ram expansion boards are still available, at \$5.00 plus a self-addressed stamped envelope (in the U.S.) or please add \$.25 (in Canada) or \$.75 (overseas). Order from him at 3054 West Evans Drive, Phoenix, AZ 85025.

CLOCK PROGRAM FOR SYM-1 WITH TERMINAL

Stephen E. Bach-AA4B
Rt 2, Box 50A-1
Scottsville, VA 24590

(Slightly modified, converted to RAE format and redocumented by editor)
This program adds a new command, .C with no parameters, to SYM. It
patches itself with a .G 0200. If you have set the time in hours, minutes
and seconds in 0000-0002, everytime you use the monitor command .C,
and hit return, the current time will appear in the upper right corner
of your KTM-2/80 screen. If you have any other type of terminal, change
the cursor control commands in the DISPLAY ROUTINES to match your terminal.
You can also use this program with a TTY by omitting the cursor
control commands; if you omit these commands the time will appear on the
screen of the CRT immediately following the request. If you wish to use
this program with other programs, (such as RAE-1) which use the BRK in-
struction, you will have to modify the interrupt service routine so that
it will differentiate between BRK and IRQ by examining the B flag. If
the B flag is set, so to MON with a JSR to USRENT at 8035; otherwise,
process the clock interrupt. If you don't add this modification, the
Control C exit from RAE will not work! You may also wish to change the
three page zero addresses to avoid conflicts with BASIC, which uses
these locations for its warm start reentry.

ASSEMBLE LIST

0010	.BA \$200
0020	.OS
0030 ;	24 HOUR CLOCK - ADAPTED FROM PROGRAM FOR AIM 65
0040 ;	BY MARVIN L. DEJONG
0050 ;	AS PUBLISHED IN MICRO, 3/79
0060 ;	ADAPTION BY STEPHEN BACH
0070 ACCESS	.DE \$8886
0080 URCVEC	.DE \$A66D
0090 IRQVEC	.DE \$A67E
0100 ACR	.DE \$A00B
0110 TILL	.DE \$A006
0120 T1CH	.DE \$A005
0130 IER	.DE \$A00E
0140 TILL2	.DE \$A004
0150 OUTCHR	.DE \$8A47
0160 OUTBYT	.DE \$82FA
0170 MONITR	.DE \$8000
0180 LSTCOM	.DE \$A657
0190 COUNT	.DE \$0003
0200 SECOND	.DE \$0002
0210 MINUTE	.DE \$0001
0220 HOUR	.DE \$0000
0230 ;	
0200- 20 86 88	0240 START JSR ACCESS
0203- 78	0250 SEI
0204- A9 75	0260 LDA \$L,RECOG *
0206- 8D 6D A6	0270 STA URCVEC
0209- A9 02	0280 LDA \$H,RECOG *
020B- 8D 6E A6	0290 STA URCVEC+1
020E- A9 88	0300 LDA \$L,INTRPT *
0210- 8D 7E A6	0310 STA IRQVEC
0213- A9 02	0320 LDA \$H,INTRPT *
0215- 8D 7F A6	0330 STA IRQVEC+1
0218- A9 C0	0340 LDA \$\$C0
021A- 8D 0E A0	0350 STA IER
021D- A9 40	0360 LDA \$\$40

021F- 8D 0B A0	0370 STA ACR
0222- A9 42	0380 LDA \$\$42
0224- 8D 06 A0	0390 STA T1LL
0227- A9 C3	0400 LDA \$\$C3
0229- 8D 05 A0	0410 STA T1CH
022C- A9 EC	0420 LDA \$\$EC
022E- 85 03	0430 STA *COUNT
0230- 58	0440 CLI
	0450 ;
	0460 ;
	0470 ;
0231- A9 1B	0480 DISPLAY LDA \$\$1B
0233- 20 47 8A	0490 JSR OUTCHR *
0236- A9 3D	0500 LDA \$\$3D *
0238- 20 47 8A	0510 JSR OUTCHR
023B- A9 21	0520 LDA \$\$21
023D- 20 47 8A	0530 JSR OUTCHR
0240- A9 66	0540 LDA \$\$66
0242- 20 47 8A	0550 JSR OUTCHR
0245- A5 00	0560 LDA *HOUR *
0247- 20 FA 82	0570 JSR OUTBYT *
024A- A9 20	0580 LDA \$\$20 *
024C- 20 47 8A	0590 JSR OUTCHR
024F- A5 01	0600 LDA *MINUTE *
0251- 20 FA 82	0610 JSR OUTBYT
0254- A9 20	0620 LDA \$\$20 *
0256- 20 47 8A	0630 JSR OUTCHR
0259- A5 02	0640 LDA *SECOND *
025B- 20 FA 82	0650 JSR OUTBYT
025E- A9 1B	0660 LDA \$\$1B
0260- 20 47 8A	0670 JSR OUTCHR
0263- A9 3D	0680 LDA \$\$3D *
0265- 20 47 8A	0690 JSR OUTCHR
0268- A9 37	0700 LDA \$\$37
026A- 20 47 8A	0710 JSR OUTCHR
026D- A9 20	0720 LDA \$\$20
026F- 20 47 8A	0730 JSR OUTCHR
0272- 4C 00 80	0740 JMP MONITR *
	0750 ;
	0760 ;
	0770 ;
0275- CD 57 A6	0780 RECOG CMP LSTCOM *
0278- F0 02	0790 BEQ OKAY *
027A- 38	0800 BAD SEC
027B- 60	0810 RTS
027C- C9 43	0820 OKAY CMP '\$C'
027E- D0 FA	0830 BNE BAD
0280- E0 00	0840 CPX \$\$00
0282- D0 F6	0850 BNE BAD
0284- 18	0860 CLC
0285- 4C 31 02	0870 JMP DISPLAY *
	0880 ;
	0890 ;
	0900 ;
0288- 48	0910 INTRPT PHA
0289- E6 03	0920 INC *COUNT
028B- D0 32	0930 BNE IDONE
028D- F8	0940 SED
028E- 18	0950 CLC
028F- A5 02	0960 LDA *SECOND *
0291- 69 01	0970 ADC \$\$01 *
0293- 85 02	0980 STA *SECOND *
0295- C9 60	0990 CMP \$\$60 *
0297- 90 22	1000 BCC NOTMIN *

```

0299- A9 00    1010      LDA #$00    *
029B- 85 02    1020      STA *SECOND  *
029D- 18       1030      CLC
029E- A5 01    1040      LDA *MINUTE *
02A0- 69 01    1050      ADC #$01    *
02A2- 85 01    1060      STA *MINUTE *
02A4- C9 60    1070      CMP #$60    *
02A6- 90 13    1080      BCC NOTMIN *
02A8- A9 00    1090      LDA #$00    *
02AA- 85 01    1100      STA *MINUTE *
02AC- 18       1110      CLC
02AD- A5 00    1120      LDA *HOUR  *
02AF- 69 01    1130      ADC #$01    *
02B1- 85 00    1140      STA *HOUR  *
02B3- C9 24    1150      CMP #$24    *
02B5- 90 04    1160      BCC NOTMIN *
02B7- A9 00    1170      LDA #$00    *
02B9- 85 00    1180      STA *HOUR  *
02BB- A9 EC    1190      NOTMIN    LDA $$EC    *
02BD- 85 03    1200      STA *COUNT  *
02BF- AD 04 A0  1210      IDONE    LDA T1LL2   *
02C2- D8       1220      CLD
02C3- 68       1230      PLA
02C4- 40       1240      RTI
1250      .EN

```

LABEL FILE: [/ = EXTERNAL]

```

/ACCESS=8B86      /URCVVEC=A66D      /IRQVEC=A67E
/ACR=A00B        /T1LL=A006        /T1CH=A005
/IER=A00E        /T1LL2=A004        /DUTCHR=8A47
/OUTBYT=82FA     /MONITR=8000        /LSTCOM=A657
/COUNT=0003      /SECOND=0002        /MINUTE=0001
/HOUR=0000      START=0200        DISPLAY=0231
RECOG=0275      BAD=027A        OKAY=027C
INTRPT=0288     NOTMIN=02BB        IDONE=02BF
RTI=02C4
//0000,02C5,02C5
>

```

```

0200 20 B6 BB 78 A9 75 BD 6B,C1
0208 A6 A9 02 BD 6E A6 A9 88,E4
0210 BD 7E A6 A9 02 BD 7F A6,F2
0218 A9 C0 BD 0E A0 A9 40 BD,0C
0220 0B A0 A9 42 BD 06 A0 A9,7E
0228 C3 BD 05 A0 A9 EC 85 03,90
0230 58 A9 1B 20 47 8A A9 3D,B3
0238 20 47 8A A9 21 20 47 8A,2F
0240 A9 66 20 47 8A A5 00 20,F4
0248 FA 82 A9 20 20 47 8A A5,CF
0250 01 20 FA 82 A9 20 20 47,9C
0258 8A A5 02 20 FA 82 A9 1B,2D
0260 20 47 8A A9 3D 20 47 8A,F5
                                508B
0268 A9 37 20 47 8A A9 20 20,AF
0270 47 8A 4C 00 80 CD 57 A6,16
0278 F0 02 38 60 C9 43 D0 FA,76
0280 E0 00 D0 F6 18 4C 31 02,B3
0288 48 E6 03 D0 32 F8 18 A5,9B
0290 02 69 01 85 02 C9 60 90,47
0298 22 A9 00 85 02 18 A5 01,57
02A0 69 01 85,01 C9 60 90 13,13
02A8 A9 00 85 01 18 A5 00 69,68
02B0 01 85 00 C9 24 90 04 A9,18
02B8 00 85 00 A9 EC 85 03 AD,67
02C0 04 A0 DB 68 40,BB

```

CLOCK PROGRAM FOR AN UNEXPANDED SYM-1

For you beginners, who are embarrassed when asked what your SYM can do, here is a simple demonstration program, which converts your SYM into a 24-hour clock. This gives your SYM somethings useful to do besides just sitting there! (Note: This is a simplified version of the clock program above.)

ASSEMBLE LIST

```

0010 ; 24-HOUR CLOCK FOR SYM-1
0020 ; H. R. LUXENBERG
0030 ;
0040 ;Enter Hours, Minutes, Seconds
0050 ;at $00, $01, $02
0060 ;
0070 ;Start Program at $0200
0080 ;
0090 .BA $200
0100 .OS
0110 ;
0120 ; DEFINITIONS
0130 ;
0140 ACCESS .DE $8B86
0150 IRQVEC .DE $A67E
0160 MONITR .DE $8000
0170 ACR .DE $A00B
0180 T1LL .DE $A006
0190 T1CH .DE $A005
0200 IER .DE $A00E
0210 T1LL2 .DE $A004
0220 OUTBYT .DE $82FA
0230 COUNT .DE $0003
0240 SECOND .DE $0002
0250 MINUTE .DE $0001
0260 HOUR .DE $0000
0270 ;
0280 ; INITIALIZATION
0290 ;
0300 START JSR ACCESS
0310 SEI
0320 LDA #L,INTRPT
0330 STA IRQVEC
0340 LDA #H,INTRPT
0350 STA IRQVEC+1
0360 LDA $$C0
0370 STA IER
0380 LDA $$40
0390 STA ACR
0400 LDA $$42
0410 STA T1LL
0420 LDA $$C3
0430 STA T1CH
0440 LDA $$EC
0450 STA *COUNT
0460 CLI
0470 JMP MONITR
0480 ;
0490 ; DISPLAY ROUTINE
0500 ;
0510 DISPLAY LDA *HOUR
0520 JSR OUTBYT
0530 LDA *MINUTE
0540 JSR OUTBYT
0550 LDA *SECOND
0560 JSR OUTBYT
0570 RTS
0580 ;
0590 ; INTERRUPT SERVICE ROUTINE
0600 ;
0610 INTRPT PHA continued on Page 22
0620 INC *COUNT

```

SYM-PLETON'S CORNER

Len Green
15 Yotam Street, Achuzah
Haifa, ISRAEL

(Editor's Note: This article is being printed in very nearly its original form. Mr. Green is far from being the 'sympleton' he calls himself. The program works beautifully, but BEEPER sounds like a crazy cricket!)

How about the above as a regular feature for beginners like me? For elementary questions and problems, and for our rudimentary programs, which will hopefully be corrected and streamlined by the veterans! Being amongst those mentioned for whom peripherals cost twice or more USA prices, I have also hooked up to a cheap 'scope for display and used the onboard keypad for input. Much better than the latter, for limited budgets, is a cannibalized calculator keyboard via one of the 6522's, which by strobing can give 70 or more characters for almost zero cost (other than some RAM).

Disassembler using ONLY onboard 7-segment display and beeper.

If anyone wants to use the disassembler (Symphysis, Introductory Issue) and has absolutely NO peripherals, the following routine will do it all ONBOARD. It is preferable to have at least 2K of onboard RAM, (although it could be maneuvered into the 1K supplied by the manufacturers).

In order to make it applicable to ANY amount of available RAM, I have deliberately located this in pages 2, 3 and 4, which leaves the following free for object code (hex dump) or other purposes: all of page 0 up to 00EF; all of page 1 (up to the stack); the first 30 or so bytes of page 2; everything above page 4. It can be relocated anywhere else, e.g. into the uppermost 3 pages of available RAM.

Program:

- (1) Make the following changes to the original disassembler program as published in SYM-PHYSIS, Introductory Issue:
 - a) Change page 20 to 03 in all 8 places.
 - b) Change page 21 to 04 in all 15 places.
 - c) Alter the following lines as follows:

2003- 4C 5E02 20DB- 20 1A04 2106- 20 8C02 210E- 20 B602
211A to 2121--4B/A900/20AF02/68/60

After amending, enter the whole original disassembler routine (or Block Move) between 0300 to 04FF (instead of 2000 to 21FF). Checksums (verify) from 0300 to 0421 should give 8270, and (the tables) from 0422 to 04FF should give 4742.

(2) Add the following 'new' routine:-

0240 20 86 88 A9 01 85 F2 A9,FB	0248 02 20 AF 02 68 AA 60 8D,85
0248 0B 20 7D 02 A9 1E BD B0,A9	02B0 BB 02 EE B0 02 60 20 FA,5C
0250 02 A2 21 A9 BF 9D 1E 02,93	02B8 B2 85 F9 8A 48 A5 F9 29,F5
0258 CA 10 FB 4C 00 03 A9 1E,7B	02C0 F0 4A 4A 4A 4A AA BD 29,9D
0260 BD 66 02 A2 05 BD BB 02,91	02C8 BC 20 AF 02 A5 F9 29 0F,00
0268 C9 BF 0D 9B 40 A6 CA,31	02D0 AA BD 29 BC 20 AF 02 68,25
0270 10 F3 A9 08 20 7D 02 EE,72	02D8 AA 60 20 23 28 29 2C 20,1C
0278 66 02 4C 63 02 BD 56 A6,14	02E0 00 49 39 0F 02 00 77 7C,A2
0280 20 5A 83 A2 04 20 72 89,D2	02E8 58 5E 79 71 6F 74 10 1E,53
0288 CA 10 FA 60 20 47 8A 85,7C	02F0 70 38 37 54 5C 73 67 50,0C
0290 F9 8A 48 A5 F9 A2 05 DD,69	02F8 6D 78 1C 62 3E 64 6E 5B,BA
0298 DA 02 F0 0A CA 10 FB AA,BB	4FDA
02A0 BD A5 02 4C A9 02 BD E0,B3	

Operation:

SHIFT/ S DBL/ 2C5/ -/ F0/CR...GO/ 240/ CR. . . and the disassembler will disassemble itself ONBOARD starting from (any say) location 02C5 etc.

Notes:

- (i) Enter starting addresses in 'everyday' form, viz:- SAH followed by SAL.
- (ii) BB at locations 0266 & 02B0 are 'blanks' and altered during execution.
- (iii) FF at locations 0254 & 0269 is an 'end sensor' for the line buffer (021E to 023F).
- (iv) Line pause & character timing can be adjusted at locations 0248 & 0273.
- (v) Locations 02DA to 02E5 are ASCII's for 6 symbols & their segment code equivalents.
- (vi) Locations 02E6 to 02FF are the 26 segment codes for the full alphabet (even though 5 letters are unused in this particular routine!!).

P.S. The above program adds another 192 bytes (plus a line buffer of 34 bytes) to the original routine which is only 504 bytes long. I am sure that clever programming could cut this to half or less, especially if duplication of ASCII & Hex tables in the original, and segment code tables in the above can be eliminated.

24-Hour Clock for Unexpanded SYM-1 Continued from Page 20

023D- D0 35	0630	BNE IDONE	
023F- 20 2A 02	0640	JSR DISPLAY	
0242- F8	0650	SED	
0243- 18	0660	CLC	
0244- A5 02	0670	LDA *SECOND	
0246- 69 01	0680	ADC #\$01	
0248- 85 02	0690	STA *SECOND	
024A- C9 60	0700	CMP #\$60	
024C- 90 22	0710	BCC NOTMIN	
024E- A9 00	0720	LDA #\$00	
0250- 85 02	0730	STA *SECOND	
0252- 18	0740	CLC	
0253- A5 01	0750	LDA *MINUTE	0200 20 86 8B 78 A9 3A 8D 7E,97
0255- 69 01	0760	ADC #\$01	0208 A6 A9 02 8D 7F A6 A9 C0,03
0257- 85 01	0770	STA *MINUTE	0210 8D 0E A0 A9 40 8D 0B A0,5F
0259- C9 60	0780	CMP #\$60	0218 A9 42 8D 06 A0 A9 C3 8D,76
025B- 90 13	0790	BCC NOTMIN	0220 05 A0 A9 EC 85 03 58 4C,DC
025D- A9 00	0800	LDA #\$00	0228 00 80 A5 00 20 FA B2 A5,42
025F- 85 01	0810	STA *MINUTE	0230 01 20 FA B2 A5 02 20 FA,A0
0261- 18	0820	CLC	0238 82 60 4B E6 03 D0 35 20,D8
0262- A5 00	0830	LDA *HOUR	0240 2A 02 F8 18 A5 02 69 01,25
0264- 69 01	0840	ADC #\$01	0248 85 02 C9 60 90 22 A9 00,30
0266- 85 00	0850	STA *HOUR	0250 85 02 18 A5 01 69 01 85,64
0268- C9 24	0860	CMP #\$24	0258 01 C9 60 90 13 A9 00 85,5F
026A- 90 04	0870	BCC NOTMIN	0260 01 18 A5 00 69 01 85 00,0C
026C- A9 00	0880	LDA #\$00	0268 C9 24 90 04 A9 00 85 00,BB
026E- 85 00	0890	STA *HOUR	0270 A9 EC 85 03 AD 04 A0 D8,01
0270- A9 EC	0900 NOTMIN	LDA #\$EC	0278 68 40,A9
0272- 85 03	0910	STA *COUNT	30A9
0274- AD 04 A0	0920 IDONE	LDA T1LL2	
0277- D8	0930	CLD	
0278- 68	0940	PLA	
0279- 40	0950 RTI	RTI	
	0960	.EN	

A BASIC/6502 GAME

Carl Moser, of Eastern Software House, was kind enough to send us a "preliminary" version of the following game, which we call Moser's Paddle Game, for want of a better name. It requires 8K of memory, and consists of two parts, one in machine language, the other in BASIC. First load the machine language portion at 1000-1089. Then load the BASIC portion, after answering the MEMORY SIZE? prompt with 4096. Before you even turn the power on, however, connect a jumper from Jumper point CC on the SYM-1 to connection point F on the AA connector. Point CC is the connection to PB7 @ \$A402, and AA-P is CA1 on VIA #3. PB7 is the CRT IN line and CA1 is used to generate an interrupt. We publish this game not only because it is fun in itself, but because it illustrates so many features of the SYM/KTM-2 combination (it can be used with other terminals by modifying the cursor control and graphics instructions in the BASIC portion of the program. It illustrates one of the many uses of the VIA, it illustrates memory mapping of the SYM to the KTM, and it shows how fast response can be provided in a BASIC program. Please note that the BASIC program is still in very rough form. Its response should be speeded up by replacing all constants such as 0, 1, -1, 61, 27, 32, 92, etc., with variables which have been assigned those values in the initialization portion of the program. This avoids the need for re-converting these constants to floating form each time they are used.

The program starts by asking "How many targets?" After a 20 second delay the requested number of targets appear randomly positioned on the screen and a moving square "ball" bounces back and forth across the screen, being "reflected" at the boundaries. By hitting the terminal keys marked "\\" and "/" at the right time, the "ball" is reflected at a 45-degree angle, and hopefully hits a target, which then is erased from the screen. The objective is to erase all of the targets with the fewest paddles. It should be obvious (but not necessarily immediately so!) how other games involving moving paddles may be generated. Those of you who are seriously interested in graphics games of this and related types should consider writing them entirely in assembly language, using the Graphics Drawing Compiler described in Issue #1. To abort the game, use Control C. This is a "preliminary" version only; Carl and I are both aware of a "bus" which causes the memory mappings to go astray after a certain number of misses. We leave the bus for you to find and exterminate.

Carl provided us with the object code only of the machine language portion of the program (on cassette); the version given here in source form is based on a disassembly and reassignment of symbolic labels, and relocation to low memory by myself; hence it is not in the elegant, well commented format typical of Carl's programs.

```
0010 !SOURCE CODE FOR MOSER'S PADDLE GAME
0020      .BA $1000
0030      .OS
0040 ACCESS    .DE $BB86
0050 NACCES   .DE $BB9C
0060 SAVER     .DE $B188
0070 TIN        .DE $BA6A
0080 TOUT       .DE $BAA0
0090 PCRVIA3   .DE $AC0C
0100 IERVIA3   .DE $AC0E
0110 PADVIA3   .DE $AC01
0120 PBD6532   .DE $A402
0130 TECHO     .DE $A653
0140 OUTVEC    .DE $A663
0150 IRQ        .DE $FFFE
0160 TEMP       .DE $00F9
0170 STORE      .DE $1100
```

1000- 78	0180	BEGIN	SEI
1001- 48	0190		PHA
1002- 20 86 BB	0200		JSR ACCESS
1005- A9 00	0210		LDA #\$00
1007- 8D 01 11	0220		STA STORE+1
100A- AD 0C AC	0230		LDA PCRVIA3
100D- 09 01	0240		DRA #\$01
100F- 8D 0C AC	0250		STA PCRVIA3
1012- A9 82	0260		LDA \$\$82
1014- 8D 0E AC	0270		STA IERVIA3
1017- A9 3C	0280		LDA \$L,INT
1019- 8D FE FF	0290		STA IRQ
101C- A9 10	0300		LDA \$H,INT
101E- 8D FF FF	0310		STA IRQ+1
1021- AD 53 A6	0320		LDA TECHO
1024- 29 7F	0330		AND #\$7F
1026- 8D 53 A6	0340		STA TECHO
1029- A9 84	0350		LDA \$L,OUT
102B- 8D 64 A6	0360		STA OUTVEC+1
102E- A9 10	0370		LDA \$H,OUT
1030- 8D 65 A6	0380		STA OUTVEC+2
1033- 20 9C 8B	0390		JSR NACCES
1036- AD 01 AC	0400		LDA PADVIA3
1039- 68	0410		PLA
103A- 58	0420		CLI
103B- 60	0430		RTS
103C- 48	0440	INT	PHA
103D- 20 51 10	0450		JSR ONE
1040- 29 7F	0460		AND #\$7F
1042- 8D 00 11	0470		STA STORE
1045- C9 03	0480		CMP #\$03
1047- F0 36	0490		BEQ TWO
1049- CE 01 11	0500		DEC STORE+1
104C- AD 01 AC	0510		LDA PADVIA3
104F- 68	0520		PLA
1050- 40	0530		RTI
1051- 20 88 81	0540	ONE	JSR SAVER
1054- A9 00	0550		LDA \$\$00
1056- 85 F9	0560		STA *TEMP
1058- AD 02 A4	0570		LDA PBD6532
105B- 4C 6A 8A	0580		JMP TIN
105E- 20 86 BB	0590	THREE	JSR ACCESS
1061- A9 02	0600		LDA \$\$02
1063- 8D 0E AC	0610		STA IERVIA3
1066- AD 01 AC	0620		LDA PADVIA3
1069- AD 53 A6	0630		LDA TECHO
106C- 09 80	0640		DRA \$\$80
106E- 8D 53 A6	0650		STA TECHO
1071- A9 A0	0660		LDA \$L,TOUT
1073- 8D 64 A6	0670		STA OUTVEC+1
1076- A9 8A	0680		LDA \$H,TOUT
1078- 8D 65 A6	0690		STA OUTVEC+2
107B- 20 9C 8B	0700		JSR NACCES
107E- 60	0710		RTS
107F- 20 5E 10	0720	TWO	JSR THREE
1082- 68	0730		PLA
1083- 40	0740		RTI
1084- 78	0750	OUT	SEI
1085- 20 A0 8A	0760		JSR TOUT
1088- 58	0770		CLI
1089- 60	0780		RTS
	0790		.EN

1000 78 48 20 86 BB A9 00 8D,27
1008 01 11 AD 0C AC 09 01 8D,35
1010 0C AC A9 82 8D 0E AC A9,08
1018 3C 8D FE FF A9 10 8D FF,13
1020 FF AD 53 A6 29 7F 8D 53,40
1028 A6 A9 84 8D 64 A6 A9 10,63
1030 8D 65 A6 20 9C 8B AD 01,F0
1038 AC 68 58 60 48 20 51 10,85
1040 29 7F 8D 00 11 C9 03 F0,87
1048 36 CE 01 11 AD 01 AC 68,5F
1050 40 20 88 81 A9 00 85 F9,EF
1058 AD 02 A4 4C 6A 8A 20 86,28
1060 BB A9 02 8D 0E AC AD 01,53
1068 AC AD 53 A6 09 80 8D 53,0E
1070 A6 A9 A0 8D 64 A6 A9 8A,C7
1078 8D 65 A6 20 9C 8B 60 20,26
1080 5E 10 68 40 78 20 A0 8A,FE
1088 58 60,B6
3BB6

LABEL FILE: [/ = EXTERNAL]

```
/ACCESS=8B86      /NACCES=8B9C      /SAVER=8188  
/TIN=8A6A        /TOUT=BAA0       /PCRVIA3=AC0C  
/IERVIA3=AC0E    /PADVIA3=AC01     /PB06532=A402  
/TECHO=A653     /OUTVEC=A663     /IRQ=FFFFE  
/TEMP=00F9       /STORE=1100     BEGIN=1000  
INT=103C         ONE=1051      THREE=105E  
INT=107F         OUT=1084  
//0000:108A,108A
```

```
1000 LN=24 :REM LINES/SCREEN  
1010 CH=40 :REM CHARACTERS/LINE -- 80 MAX.  
1020 GOTO 2080  
1030 NU=LN*CH  
1040 SP=32  
1050 TA=160  
1060 LR=47  
1070 LL=61 :REM --"=" SINCE "\\" IS UPPER CASE  
1080 ME=4354  
1090 T=0  
1100 PRINT CHR$(27);CHR$(69);
```

```
1110 SS=0  
1120 LE=08  
1130 RI=09  
1140 UP=11  
1150 DN=10  
1160 POKE ME+SS,SP  
1170 IF SS>NU THEN 1200  
1180 SS=SS+1  
1190 GOTO 1160
```

```
1200 K=0  
1210 FOR I=1 TO J  
1220 A=INT(NU*RND(1))  
1230 IF PEEK(ME+A)=TA THEN 1220  
1240 POKE ME+A,TA  
1250 GOSUB 19601 :REM SET UP CRT CORD  
1260 PRINT CHR$(27);CHR$(71);CHR$(64);CHR$(27);CHR$(103);  
1270 NEXT I  
1280 A=INT(NU*RND(1))  
1290 IF PEEK(ME+A)=TA THEN 1280  
1300 NE=A+ME  
1310 GOSUB 1960  
1320 DI=1:EC=RI:IF RND(1)>.5 THEN DI=-1:EC=LE  
1330 AA=USR(4096,AB)  
1340 REM --LOOP FOR NEXT POSITION  
1350 IF PEEK(NE)=TA THEN 1380
```

```
1360 IF PEEK(4353)<>0 THEN POKE 4353,0:AA=PEEK(4352):GOTO 1720  
1370 GOTO 1410  
1380 K=K+1:PRINT " ";CHR$(08);  
1390 POKE NE,SP :REM --CLEAR TARGET IN MAP  
1400 IF K=J THEN 2010  
1410 IF ABS(DI)=1 THEN 1530  
1420 IF DI>0 THEN 1480  
1430 REM -- THE FOLLOWING PROCESSES -40  
1440 IF PEEK(NE)=LL THEN DI=-1:EC=LE:GOTO 1800  
1450 IF PEEK(NE)=LR THEN DI=1:EC=RI:GOTO 1820  
1460 IF NE+DI<ME THEN DI=-DI:EC=DN:GOTO 1340  
1470 GOTO 1640  
1480 REM --THE FOLLOWING PROCESSES +40  
1490 IF PEEK(NE)=LL THEN DI=1:EC=RI:GOTO 1840  
1500 IF PEEK(NE)=LR THEN DI=-1:EC=LE:GOTO 1860  
1510 IF NE+DI>ME+NU THEN DI=-DI:EC=UP:GOTO 1340  
1520 GOTO 1640
```

```
1530 IF DI>0 THEN 1590  
1540 REM --THE FOLLOWING PROCESSES -1  
1550 IF PEEK(NE)=LL THEN DI=-CH:EC=UP:GOTO 1880  
1560 IF PEEK(NE)=LR THEN DI=CH:EC=DN:GOTO 1900  
1570 IF ((NE-ME)/CH)-INT((NE-ME)/CH)=0 THEN DI=-DI:EC=RI:GOTO 1340  
1580 GOTO 1640  
1590 REM  
1600 REM --THE FOLLOWING PROCESSES +1  
1610 IF PEEK(NE)=LL THEN DI=CH:EC=DN:GOTO 1920  
1620 IF PEEK(NE)=LR THEN DI=-CH:EC=UP:GOTO 1940  
1630 IF ((NE-ME+1)/CH)-INT((NE-ME+1)/CH)=0 THEN DI=-DI:EC=LE:GOTO 1340  
1640 REM --LAST RESORT DEFAULTS  
1650 IF (NE+DI) >= ME+NU THEN DI=-CH:EC=UP  
1660 IF (NE+DI) < ME THEN DI=CH:EC=DN  
1670 IF DI=1 THEN IF ((NE-ME+1)/CH)-INT((NE-ME+1)/CH)=0 THEN DI=-DI:EC=LE  
1680 IF DI=-1 THEN IF ((NE-ME)/CH)-INT((NE-ME)/CH)=0 THEN DI=-DI:EC=RI  
1690 PRINT CHR$(EC);  
1700 NE=NE+DI:REM --DO UPDATE  
1710 GOTO 1340  
1720 IF AA=LR THEN 1750  
1730 IF AA=LL THEN 1750  
1740 GOTO 1410  
1750 POKE NE,AA :REM --ENTER IN MAP  
1760 IF AA=LL THEN AA=92 :REM --MAKE "\\" PRINTABLE  
1770 PRINT CHR$(AA);CHR$(B);  
1780 T=T+1  
1790 GOTO 1410  
1800 IF (NE-ME)/CH-INT((NE-ME)/CH)=0 THEN DI=CH:EC=DN  
1810 GOTO 1640  
1820 IF (NE-ME+1)/CH-INT((NE-ME+1)/CH)=0 THEN DI=CH:EC=DN  
1830 GOTO 1640  
1840 IF (NE-ME+1)/CH-INT((NE-ME+1)/CH)=0 THEN DI=-CH:EC=UP  
1850 GOTO 1640  
1860 IF (NE-ME)/CH-INT((NE-ME)/CH)=0 THEN DI=-CH:EC=UP  
1870 GOTO 1640  
1880 IF NE+DI<ME THEN DI=1:EC=RI  
1890 GOTO 1640  
1900 IF NE+DI>ME+NU THEN DI=1:EC=RI  
1910 GOTO 1640  
1920 IF NE+DI>ME+NU THEN DI=-1:EC=LE  
1930 GOTO 1640  
1940 IF (NE+DI)<ME THEN DI=-1:EC=LE  
1950 GOTO 1640  
1960 REM --SET UP CRT COORDINATES  
1970 X=INT(A/CH):REM --FIND WHICH LINE  
1980 Y=A-X*CH :REM --FIND CHAR. POSITION  
1990 PRINT CHR$(27);CHR$(61);CHR$(X+32);CHR$(Y+32);  
2000 RETURN  
2010 PRINT CHR$(27);CHR$(69);:REM --HOME AND CLEAR  
2020 FOR I=1 TO 50:NEXT:FOR I=1 TO 10:PRINT:NEXT  
2030 PRINT "You eliminated"J"targets in" T" attempts  
2040 PRINT  
2050 PRINT "You averaged" T/J" attempts per target"  
2060 PRINT:PRINT  
2070 AA=USR(4190,AB)  
2080 INPUT "How many targets? " ;J  
2090 J=ABS(INT(J))  
2100 IF J<1 THEN 2120  
2110 K=0:T=0:GOTD1030  
2120 PRINT "Thank you. Goodbye for now.";END  
OK
```

SHOPPING LIST OF ITEMS AVAILABLE FROM SYM-1 USER'S GROUP
All prices given below are now obsolete. Please use prices
on the most recent issued 'Shopping List'.
EXTENDED TINY BASIC FOR SYM-1 (INCLUDES TOM FITTMAN'S TB MANUAL, TOM
PITTMAN'S TB EXPERIMENTER'S KIT, HEX DUMP WITH CHECKSUMS,
AND CASSETTE OBJECT DUMP)
\$20 IN US FUNDS - AIR MAIL POSTPAID ANYWHERE IN THE WORLD.

6502 ASSEMBLY LANGUAGE PROGRAMMING (BY LANCE A. LEVENTHAL,
PUBLISHED BY OSBORNE/MCGRAW-HILL)
LIST PRICE \$12.50, DISCOUNTED TO SUBSCRIBERS
\$11.25 IN US FUNDS - SURFACE MAIL ANYWHERE IN THE WORLD.
FOR AIR MAIL EUROPEAN COUNTRIES PLEASE ADD \$3.00
FOR AIR MAIL ASIA/PACIFIC PLEASE ADD \$4.00

THREE SYM-1 BASIC ENHANCEMENTS BY JACK BROWN (FULLY DOCUMENTED
AND COMMENTED SOURCE CODE LISTINGS)
ULTRA-RENUMBER ----- 22 PAGES, \$6.60
SUPER TCP WITH FULL CURSOR CONTROL ----- 14 PAGES, \$4.20
"NAMED" CASSETTE SAVE AND LOAD ----- 22 PAGES, \$6.60
PRICE AS LISTED, AIR MAIL POSTPAID ANYWHERE IN THE WORLD

RAE NOTES (UPDATING SERVICE, THREE MAILINGS PLANNED)
PRICE \$7.50 US FUNDS, AIR MAIL POSTPAID ANYWHERE IN THE WORLD

SEE ISSUE #1 FOR PRICES ON THE FOLLOWING:
2K SYMBOLIC ASSEMBLER (BY ROBERT DENISON)
SYNERTEK TECHNICAL NOTES
SUPERMON VERSION 2 (MON 1.1).
RAE-1/2.
SCHEMATIC DIAGRAM OF SYM-1

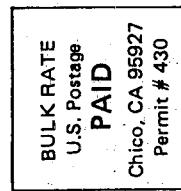
MISCELLANEOUS ITEMS

William Augur, 34A Mindi Ct., Erie, PA 16510, wants to contact SYMmers
in his area for information exchange. Thanks to Jeff Holtzman for cas-
settes and listings of MON extensions, including STRING FIND, and ASCII
VERIFY, which will appear in Issue #3. Raymond L. Reome, 1535 Starlings
Dr., Florissant, MO 63031, would like to hear from anyone who has in-
terfaced SYM to the Motorola MC6847 or the Atari Video Computer Game.
F. L. Winter, VK2BLF, in Australia, wants to hear from SYMmers on 20
Meters. Happy Tenth Birthday to Michael Blaszczak, our youngest sub-
scriber; hope you find the Technical Notes and Schematic Diagram your
parents save you helpful. We look forward to having you submit an ar-
ticle for publication. Thanks to Jan Skov, who submitted a KIM HYPER-
TAPE LOAD program, commented in Danish. Thanks to all others who sub-
mitted material which could not be published at this time; as SYMmers
write asking questions which your material could answer we will send a
copy on to them.

PAGE 0 AND 1 ASSIGNMENTS

Those of you who are using RAE, BAS, and MON together know there are
some potential Page 0 problems. BAS uses 0000-00E8; MON uses 00FB-00FF;
RAE uses 00B6-00F7 (also 0000-005 if you use USR or "Y"). Obviously it
is difficult for RAE and BAS to call each other except at cold starts,
unless the calling patches preserve the overlapping portion of page 0.
Although none of page 0 seems to be available for user programs, there
are many locations used only as 'scratchpad', or whose intended use is
replaceable. For examples: 00F8 is used only for cassette operations;
0000-0002 is always available if you use .G C27E as your warm start to
BASIC instead of .G 0. We'll list more such 'free' locations next time.
BAS and RAE also conflict in page 1 up through 010A, and RAE uses up
through 0184 (and 018F-01DE). Forget about page 1, if you use RAE!

SYM-PHYSIS 2-27



TIME VALUE PRINTED MATTER

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

Address Correction Requested