SYM-PHYSIS

THE SYM-1 USERS' GROUP NEWSLETTER

VOLUME II, NUMBER 3 (ISSUE NO. 9) - AUTUMN 1981 (JUL/AUG/SEP)

SYM-PHYSIS is a quarterly publication of the SYM-1 Users' Group, P. O. Box 315, Chico, CA 95927. SYM-PHYSIS and the SYM-1 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*
Associate Editors: *Dennis Hall, Jack Brown, Jack Gieryic*

SUBSCRIPTION RATES (1981)

USA/Canada - $10.00 for a volume of four issues. Elsewhere - $13.50. Make checks payable in US dollars to "SYM-1 Users' Group", P. O. Box 315, Chico, CA 95927, Telephone (916) 895-8751.

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

RAE'S SUG-GESTIONS

We have had little success in thinking of a clever title for the introductory editorial material which appears in this position in each issue. It dawned on us, as we were assembling and editing the material for this issue, that we were serving as the Resident Assembler Editor (RAE) for the SYM USER'S GROUP (SUG); hence the above caption. Surely some reader can provide a better heading (but certainly never a worse pun!)?

We seem to be following the, by now traditional, custom of user group newsletters in being late with each issue. We'll skip the apologies and excuses, to save space and time for all, and proceed to the facts:

The emphasis in this issue is on printers and RAE-1. As you must know by now, cost/effectiveness is one of the most important factors affecting our buy/no-buy decisions. In every-day words, we want to spend as little as possible on the necessities (but can easily be persuaded to pay much more for luxuries!). We first determine our minimal requirements, then decide whether additional capabilities justify their added cost.

Take printers, for example. Our minimal requirements are for a "good-looking", easily readable page (this implies lower case descenders to us). Also, we don't try to hide the fact that our correspondence is prepared on a computer; after all, computers are our way-of-life. Thus, if a dot matrix printer meets our minimal requirements, it would not be worth an additional $1000 to us to pay for so-called "typographic quality". We might pay $100, or $200, but $500 more would take a lot more soul-searching!

As of Summer '81, we consider the Epson MX-80 to be the "best buy" in

printers. We will be incorporating it in our OEM (SYM-based) word processing systems. IBM is following our lead in this instance, by also providing the MX-80 (under their own label) as part of their newly announced (8086-based) personal computer system! According to the trade journals, the MX-80 is now the "best-seller" among printers, selling like Apples!

Here are some printing "samples" from our MX-80:

FONTS

Compressed, Standard, Expanded-Compressed, Expanded
Compressed, Standard, Expanded-Compressed, Expanded

MODES

Normal, Double-Strike, Emphasized, **Emphasized-Double**

The emphasized-double-strike mode compares very favorably with a conventional typewriter with cloth ribbon, but the printing rate is reduced from the normal 80 cps to a mere 20 cps. All of the eight type fonts (the italics come only with GRAFTRAX-80) shown above can be printed in the normal and/or double-strike modes, but only the standard and expanded fonts can be printed in the two emphasized modes.

LOOK AT THIS!

The MX-80 also has underlining capability, based on its ability to generate any desired inter-line spacing. One of our planned near-future tasks is to prepare a version of SWP which can take full advantage of this, and other features of the MX-80. These include fully selectable horizontal and vertical tabbing. Text and graphics can be intermixed, as well. Look for demonstrations of this in future issues, as we grow in our understanding of this marvelous new tool!

RAE-1 is one of three features which make the SYM-1 an outstanding performer in any price class (the other two features are SUPERMON and the versatile I/O capabilities). In this issue are several support programs for RAE (including an Epson MX-80 interface) and the announcement of several add-on programs, including a Cross Reference Lister and a Structured Programming Enhancement Package. These two add-ons, plus SWP-1, and Hissink's Disassembler into RAE, make RAE-1 an even more versatile software package.

And now, on with the news.....

ON PRINTERS

Our first "printer" was a Teletype KSR-35 (the "heavy-duty" model of the KSR-33), working at a slow, noisy, oily-smelling, 110 baud, upper case only. This was replaced by a decwriter II (LA-36), when we were offered one at a price too low to refuse (still over $1000, though). We were thrilled by the lower case, and the 300 baud, which we almost immediately upped to 600 baud by a very simple modification. No "hand-shaking" required at this rate, either!

The LA-36 is a truly remarkable, thoroughly reliable, piece of equipment, and has needed no maintenance, preventive or otherwise, in over two years. Our only troubles were self-induced, when twice we used cheap ribbons, the lint from which clogged one of the print wires so that it would not retract. The problem was easily solved with a few squirts of TV Contact Cleaner.

We had only three objections to the LA-36. First was its lack of portability. It was a back-breaking task to get it loaded into the

trunk of our car (the lid wouldn't close either) for a demonstration. Second was the lack of descenders on the g, j, p, q, and y. The g was the worst appearing of the letters, and one had to look carefully to distinguish between the g and the s in such words as sag, gag, gas, gabs, etc. Third, like the 8" disk drives, its motor runs all of the time it is powered-on, and we find the noise objectionable.

As you can see, we now have a new printer. We chose the Epson MX-80, which answers all three objections, at much less cost, and with far greater versatility. We have the FT model, in case we ever wish to feed single sheets of letterhead, or envelopes, and have added the GRAFTRAX-80 EPROMS, 3 2716s, in place of the original 2332 ROM, to give us italics and high resolution bit graphics (to go with the MTU Visible Memory). We are now on the parallel interface, but will switch to the current loop, via the serial interface card, to free up the VIA for other purposes.

We are only just now learning to use some its features, most of which are invoked with ESC sequences and CONTROL codes. RAE-1 users are aware that RAE does not emit most of the CONTROL codes; instead, RAE prints them as "up-arrows" followed by the corresponding alpha (SWP-1 won't print up-arrows, incidentally). Thus RAE's output must be monitored for "up-arrows", and when one is encountered in the output stream, if RAE is emiting it because it has encountered a control code, the actual control code itself must be sent to the printer.

Also, the ESC sequences are not "visible" on the terminal and two very important Epson sequences, ESC E and ESC G affect the KTM-2 by clearing the screen and setting its graphics mode. The former is disconcerting, since the number of the line which contains it is unseeable, but the latter may be nullified with an ESC g. It is therefore desirable to inhibit any ESC sequence from reaching the terminal, and substituting in its place a "token", such as, for example, the "$". The printer patch published below "protects" the KTM-2 from all ESC sequences and CONTROL codes, but "displays" them with "$" and "up-arrow" prefixes for editing.

```
0010 ;EPSON PARALLEL PRINTER PATCH
0020 ;<DISPLAYS CONTROL CODES
0030 ; AND ESC SEQUENCES>
0040
0050            .BA $6A00     ;or wherever!
0070
0080 PBD        .DE $A800
0090 PBDD       .DE $A802
0100 PCR        .DE $A80C
0110
0120 ACCESS     .DE $8B86
0130 OUTVEC     .DE $A664
0140 TOUT       .DE $8AA0
0150
0160 DISKS      .DE 0         ;0 = no disks, 1 = disks
0170 ;
0180 ;NOTE 0: Bits 0 thru 6 of the "B" port are
0190 ;        the OUTPUTs to the 7 LSB's of the
0200 ;        Epson. Since bit 7 of the A register
0210 ;        is always zero on calls to OUTCHR
0220 ;        why "waste" PB7, when we can put it
0230 ;        to good use elsewhere?
0240 ;
0250 ;        THE MSB LINE OF THE EPSON MUST BE
0260 ;        TIED TO GROUND, SINCE IT IS NOT
0270 ;        DRIVEN BY THE SYM.
0280 ;
0290 ;        Bit 7 of the "B" port is the "BUSY"
0300 ;        signal INPUT.
0310 ;
0320 ;        CB2 is the "STROBE" signal OUTPUT.
0330 ;
```



```
6A00- 20 86 8B  0340 INIT       JSR ACCESS
6A03- A9 19     0350            LDA #L,PRINT
6A05- 8D 64 A6  0360            STA OUTVEC
6A08- A9 6A     0370            LDA #H,PRINT
6A0A- 8D 65 A6  0380            STA OUTVEC+1
6A0D- A9 A0     0390            LDA #%10100000
6A0F- 8D 0C A8  0400            STA PCR       ;Set for one-shot "hand-shake"
6A12- A9 7F     0410            LDA #$7F
6A14- 8D 02 A8  0420            STA PBDD
                0430
                0440 ;    For non-disk systems
                0450            IFE DISKS
6A17- 60        0460            RTS
6A18- EA        0470            NOP
                0480            ***
                0490
                0500 ;    For disk systems
                0510            IFE DISKS-1
                0520            CLC
                0530            BRK
                0540            ***
                0550 ;
                0560 ;NOTE 1: If you wish to suppress RAE's "//" and
                0570 ;        ">" at the end of your letters, "trap"
                0580 ;        them here.  Just send "nulls" to your
                0590 ;        printer and the proper codes to
                0600 ;        your terminal.  You may wish a
                0610 ;        variety of printer patches with
                0620 ;        differing features.
                0630
                0640 ;NOTE 2: ESC sequences and CONTROL codes
                0650 ;        are kept from the terminal,
                0660 ;        but are "displayed" there
                0670 ;        with "$" and "^" prefixes.
                0680 ;
6A19- C9 1B     0690 PRINT      CMP #$1B      ;ESC code
6A1B- D0 09     0700            BNE TEST^
6A1D- 20 4E 6A  0710            JSR WAIT
6A20- A9 24     0720            LDA #'$
6A22- 20 A0 8A  0730            JSR TOUT
6A25- 60        0740            RTS
                0750
6A26- C9 5E     0760 TEST^      CMP #'^       ;RAE "flags" CONT codes with "^"
6A28- D0 21     0770            BNE NOT^
6A2A- BA        0780            TSX
6A2B- BD 03 01  0790            LDA $103,X    ;RAE "stacks" control codes
6A2E- C9 20     0800            CMP #$20
6A30- B0 17     0810            BCS PRINT^    ;Not a control code
6A32- 48        0820            PHA
6A33- A9 40     0830            LDA #$40      ;RAE will convert to null
6A35- 9D 03 01  0840            STA $103,X
6A38- A9 5E     0850            LDA #'^
6A3A- 20 A0 8A  0860            JSR TOUT
6A3D- 68        0870            PLA
6A3E- 48        0880            PHA
6A3F- 18        0890            CLC
6A40- 69 40     0900            ADC #$40      ;Make it printable!
6A42- 20 A0 8A  0910            JSR TOUT
6A45- 68        0920            PLA
6A46- 4C 4E 6A  0930            JMP WAIT
6A49- A9 5E     0940 PRINT^     LDA #'^
6A4B- 20 A0 8A  0950 NOT^       JSR TOUT
6A4E- 2C 00 A8  0960 WAIT       BIT PBD
6A51- 30 FB     0970            BMI WAIT
6A53- 8D 00 A8  0980            STA PBD
6A56- 60        0990            RTS
                1000            .EN
```

EPSON-SYM WIRING LIST

EPSON NAME	EPSON PIN	SYM PIN	SYM NAME
DATA 1	2	AA- L	2PB0
DATA 2	3	AA- 9	2PB1
DATA 3	4	AA- K	2PB2
DATA 4	5	AA- 8	2PB3
DATA 5	6	AA- J	2PB4
DATA 6	7	AA- 7	2PB5
DATA 7	8	AA- H	2PB6
DATA 8	9	--	--
STROBE	1	AA- 5	2CB2
BUSY	11	AA- 6	2PB7
LOGIC GRND	16	AA- 1	GRND

Remarks: Only 10 wires go between SYM and Epson. Tie DATA 8 to ground at Epson connector. All return lines (19 thru 30) are tied to ground at the Epson.

ALTERNATIVE WIRING LIST

You may wish to use the A Port instead, reserving the B Port for applications requiring its Darlington drive capability, or the special purpose functions available through PB6 and PB7. We will be making this switch ourselves. We had planned to use the 20 mA current loop for the printer, but instead will be converting that to inverted TTL (RS-232C compatible) for use with a modem. Here is the PORT A Wiring List:

EPSON NAME	EPSON PIN	SYM PIN	SYM NAME
DATA 1	2	AA- D	2PA0
DATA 2	3	AA- 3	2PA1
DATA 3	4	AA- C	2PA2
DATA 4	5	AA-12	2PA3
DATA 5	6	AA- N	2PA4
DATA 6	7	AA-11	2PA5
DATA 7	8	AA- M	2PA6
DATA 8	9	--	--
STROBE	1	AA- 4	2CA2
BUSY	11	AA-10	2PA7
LOGIC GRND	16	AA- 1	GRND

A convenience feature you will wish to add (based on an idea of Jeff Holtzman's) is an "IGNORE BUSY" switch. This will permit you to put the printer OFF LINE while you are entering data at the terminal which you do not wish printed. The switch will then permit terminal entry while the printer is switched off-line. The simplest method of implementing such a function is to switch in a pull-down resistor from the BUSY input to GRND. A suitable value is around 1K. This will also permit a quick "preview" on the terminal before the final printing.

EPSON MX-80 / GRAFTRAX-80 ESCAPE AND CONTROL CODES

ESCAPE SEQUENCES

ESC 0 Set to 8 lines per inch
ESC 1 Set to 7/72 inch per line (approx 10 lpi)
ESC 2 Set to 6 lines per inch (*default*)
ESC 3 n Set to n/216 inch per line

ESC 4 Set to italics
ESC 5 Set to roman (*default*)

ESC 8 Disables paper out error (permits single-sheet loading)
ESC 9 Enables paper out error (*default*)

ESC : Set TRS-80 mode (not used with SYM)
ESC ; Clear TRS-80 mode (*default*)

ESC > Set MSB (for TRS-80 graphics from SYM)
ESC = Clear MSB (*default*)

ESC ? For redefining ESC codes
ESC @ Return to default settings, reset top-of-form
ESC < Return print head to home (left) position

ESC A n Set to n/72 inch per line
ESC B Set vertical tab positions (see manual for details)
ESC C n Set to n lines per page (*default* 66)
ESC D Set horizontal tab positions (see manual for details)

ESC E Set emphasized mode
ESC F Clear emphasized mode (*default*)

ESC G Set double-strike mode
ESC H Clear double-strike mode (*default*)

ESC J,K,L Used for bit graphics control

ESC P Set compressed mode (same as CONT O)
ESC Q Clear compressed mode (same as CONT R, *default*)

ESC S Set expanded mode (same as CONT N)
ESC T Clear expanded mode (same as CONT T, *default*)

"SPECIAL" CONTROL CODES

CONT N Set expanded mode (same as ESC S)
CONT T Clear expanded mode (same as ESC T, *default*)

CONT O Set compressed mode (same as ESC P)
CONT R Clear compressed mode (same as ESC Q, *default*)

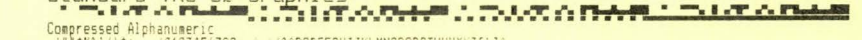
CONT Q Select printer (*default*)
CONT S Deselect printer

REMARKS

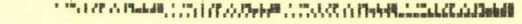
ESC G sets graphics mode on KTM-2; cancel with ESC g
ESC E clears screen on KTM-2; no way to cancel!
Except for CONT Q, CONT S, ESC sequences suffice for all purposes

The TRS-80 graphics set "replaces" the ASCII values from $A0 through $DF with the 64 "symbols" made up by blacking in all possible combinations of squares within a two (horizontal) by three (vertical) rectangle. When operating in the condensed character mode, the TRS-80 graphics set provides 264 "pixels" across an 8 inch paper. Following are examples of the TRS-80 graphics font:

Standard Alphanumeric
!"#$%&'()*+,-./0123456789:;<=>/@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
Standard TRS-80 Graphics



Compressed Alphanumeric
!"#$%&'()*+,-./0123456789:;<=>/@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
Compressed TRS-80 Graphics



Note that the "up arrow", "^", which is not printable with SWP-1, actually appears above, together with its corresponding TRS-80 graphics symbol. SWP-1 uses the "^" as a "symbolic space" to signal the justification subroutine not to "adjust" the spaces so indicated. The "^" is printed as a space ($20). We have frequently felt the need to print the "^", as in the example. We therefore have modified our SWP to substitute a character for the "symbolic space" for which we can see absolutely no use whatsoever! This is the character whose ASCII code is $60. This shows up on the KTM-2 screen as a "sigma", and on the printer as a "little hook". It is entered on the KTM-2 by holding the shift key down while hitting the "return" key; this is no more complicated than entering the "^".

INTER-SYM COMMUNICATIONS

The fastest, and perhaps simplest, method of interchanging files between two closely adjacent SYMs is via the cassette interface. First connect their grounds together, then connect the "left"-end of R90 on the "talking" SYM to the "right"-end of R92 on the "listening" SYM. Then enter L2 <cr>, and S2 ID,SA,EA <cr> on the proper SYMs.

We made up an eight foot long flexible three-wire cable to enable us to interconnect any two of our SYMs. The three wires were once part of a 25-wire flat ribbon cable, and micro-probe connectors are used to permit rapid changes.

To speed up the interchange by a factor of more than 6X, try the following (Only HSBDRY need be changed at the receiver, and only the others at the sender):

1) Change $A630 from $04 to $01 to shorten the synch signal time from approximately 4 sec to approximately 1 sec.

2) Change $A632 from $46 to $0C to shorten HSBDRY from 70 x 8 = 560 usec to 12 x 8 = 96 usec.

3) Change $A635 from $33 to $08 to shorten TAPET1 from 51 x 5 = 255 usec to 8 x 5 = 40 usec.

4) Change $A63C from $5A to $0F to shorten TAPET2 from 90 x 5 = 450 usec to 15 x 5 = 75 usec.

Since 255 + 450 = 705 usec and 40 + 75 = 140 usec, the baudrate is increased from 1418 to 8696. Since the data is sent byte-for-byte, rather than nibbled into two ASCII bytes/data byte, the "effective" baudrate is > 17 KHz. According to scope measurements, there appears to be around 1 1/2 bits of "idle" time between bytes. Rounding this up to 2 bits, we get a transfer rate of > 1700 bytes per second.

We could not work at a 10X rate, and did not try any rates between 6X and 10X.

SYM TO AND FROM KIM AND AIM 65

Since the SYM-1, KIM-1, and AIM-65 all have in common the KIM cassette format, interchange between these three over the cassette interface is possible, but takes 21 times as long as the standard SYM cassette dump. All three also support the "DEMON" paper tape format, which is not necessarily implemented at 110 baud over the TTY interface. It can also be used over the CRT interface at 4800 baud.

To try it between SYMs, ground the two SYMs together, and connect U38-pin 2 of the transmitter to U38-pin 5 of the receiver, with a switch in the line so that the receiver echo can be turned off when desired. Enter LP <cr> on the receiver and SP SA,EA <cr> on the transmitter. Close the switch, and enter <cr> on the transmitter. After the transmission is complete, open the switch, and enter ;00 at the receiver to exit the LP command. The switch is not required, but the echo from the receiver on the transmitter monitor may be confusing.

You should easily find the the corresponding connectioon points on the KIM and AIM systems. We understand that a number of 6800 based systems also support the "DEMON" format. Note that the AIM-65 uses the X OFF character, also known as (aka, or alias) DC3, CONT S, or ASCII $13) to end the transmission. We were not able to check out the ideas presented here with either the KIM or the AIM, but see no real problems in this area. We'd like to hear from any of you who succeed or fail in this intercommunication task. Incidentally, DC1 (CONT Q), DC2 (CONT R), and DC4 (CONT T), are aka X ON, TAPE ON, and TAPE OFF, respectively.

Thanks to Steve Waldman, of Screen Sound Inc., of Burbank, CA, for many of the ideas in this section in particular, and many other ideas, in general. We spent a pleasant late evening together in his lab, working out many fascinating little problems.

MORE ON RECORDERS

We purchased a Radio Shack Realistic CTR-80 Cassette Recorder (this is the recorder recommended for use with the TRS-80) nearly a year ago, but found it to be somewhat marginal in performance with the SYM-1. We found it necessary to approximately double the signal available at the Audio Out (LO) pin (A-M) by replacing the 470 ohm resistor at R88 with a 1 Kohm resistor. This improved the performance considerably, but a still better fix is described below!

We probably also should have brought out the Audio Out (HI) signal at pin A-P and tried it in the AUX input of the CTR-80, to have been completely thorough in our test. The AUX input might be better for recording. We will ask at Radio Shack which input is recommended for use with the TRS-80/CTR-80 combination.

Most small recorders, however, do not have an AUX input, which is a "high level" input, for cassette to cassette, or radio to cassette, transfers; thus, only the Audio Out (LO) should be used. Audio Out (HI) would require an external attenuator of some type.

We read somewhere that Radio Shack would upgrade the CTR-80 to the CTR-80A at NO CHARGE, to improve its performance as a computer I/O peripheral (with the TRS-80, of course!). Thus, when an internal mechanical part came "unglued", and we had to have it repaired we asked for the free upgrading. The repair invoice noted that a 10 uF capacitor and a "spike mod" had been installed. We don't know the details of the upgrading, but the "spike mod" is the essential item to improve performance, We now find the CTR 80"A" highly reliable, but the R88 modification may still be necessary.

A really nice feature of the CTR-80 is that the FAST-F and REWIND controls operate even when the remote control inhibits the PLAY and PLAY/RECORD actions. The Tape Counter is a useful feature for those who use longer than the C-10 (50 foot, 5 minutes per side) tapes we have been using, with only a multiple dump of very few files on one cassette. The CTR-80A is not listed in the Radio Shack Catalog, but is listed in the TRS-80 Computer Catalog as item 26-1206.

We recently purchased a Realistic Minisette-9 for a portable system we are building. It works very well (i.e., reliably) and is so elegant in its compactness. While it is relatively expensive, if you need or want the small size, the price differential is reasonable. The Minisette-9 is recommended by Radio Shack for use with the TRS-80 Pocket Computer.

One of our readers mentioned casually that his cassette recorder worked very reliably if it was not too near his video monitor. We had similar problems when using a SYM system on a card table with a metal rim. The metal "loop" antenna coupled the sweep frequency (approx 15.75 KHz) from the CRT yoke to the read head! The spikes showed up beautifully on a scope, together with the cassette signals we were monitoring, changing amplitude as we reoriented the monitor and/or recorder!

Although it probably does not directly affect cassette performance, it is interesting to note that we can actually "hear" the vertical scrolling of the terminal display through signals picked up by an unshielded cassette input lead, and passed through PB6 to our second DAC into a speaker. We must keep the volume control on DAC No. 2 turned down when not using the music system, partly for this reason, but mostly because otherwise we would be forced to listen to the busy buzz of tape loads. Some 60 Hz hum also enters the audio system through this route.

The moral here is that some cassette problems may be caused by the video monitor. This may be checked out by seeing if reliability is enhanced by turning off the monitor.

KTM-2 MODIFICATIONS
(AND RELATED TOPICS)

We have been using Jack Gieryic's JBP-4 KTM-2 Character Generator EPROM Burner program to replace the Character Generator 2316B ROMs in our own KTM-2 and KTM-2/80s with 2716 EPROMs. We particularly like the way the program makes full use of the KTM's cursor control capabilities, letting us use U, D, L, R, to "draw" the desired new character within an 8 by 8 rectangle indicated on the screen.

We never quite found the time to analyze either Jack's BASIC programs or the original ROMs to find out how the character generation is done by the KTM-2s. The following set of three short, but very elegant, notes by Dr. Gerhard Strube, not only describes the organization of the character generator ROMs, but describes in some detail the organization of the control ROM (in particular, for the KTM-2/80, but the KTM-2 ROM is essentially similar), and how to modify the keyboard encoding.

Two such mods we would like in our own KTM keyboard encoder are:

1) Interchange BREAK and TAB; why must we shift for BREAK?

2) Let SHIFT/SPACEBAR also enter a space; who needs the "pi"?

Synertek's 2316B is equivalent to Intel's 2316E. Both have three chip select pins, numbers 20, 18, and 21. The same numbered pins in the 2716 EPROM are OE (low), CE (low), and Vpp (high). The low and high in parentheses refer to the conditions for READ. There are eight possible configurations for the set of three chip selection pins on the 2316E/B, only one of which is compatible with that of the 2716. The control ROM of the KTM-2, (and incidentally, the Apple II ROMs) are, for some devious (?!) reason, not 2716-compatible, but this presents no obstacle to the determined researcher! A voltmeter or logic probe, on the hardware side, and a disassembler, on the software side, will reveal all secrets, and EPROMs are easily substituted.

The Epson MX-80 is delivered with its operating system and character generator in a single 4K ROM (NEC's D2332C); the microprocessor is a NEC 8049. Undoubtedly, users having access to an 8049 development system will disassemble the contents of the ROM and modify the character set to match their own personal requirements. The Graftrax 80, a set of three replacement 2716s, is sold only under the sub-license agreement that the contents are trade secrets of Epson, and that the purchaser "will not create or attempt to create, by reverse engineering, modification or otherwise, the source programs or any part thereof from the Graftrax 80 object program." Thus only the original 2332C ROM may be analyzed, and modified, and EPROMed, as Dr. Strube has done for the KTM-2/80, to meet one's needs. It certainly might be desirable to provide identical character sets, including the graphics symbols, in both the terminal and the printer.

And now, here are the three notes, photocopied from his original manuscript. Note that his manuscript is right-justified, printed on a shaped-character printer, with all of the DIN standard characters. We wonder what printer he uses?

C U S T O M I Z I N G T H E K T M / 2 - 8 0

Dr. Gerhard Strube, Sckellstr.5, D-8000 München 80, Germany

(I) Changing the character set

SYNERTEK`s KTM/2-80 terminal board features a 1920 character display, 24 lines by 80 characters each. The characters - both ASCII and graphics - are represented by a `pixel` of 8

bytes (= 8 * 8 bits), each bit corresponding to a dot on the screen. Since a total of 256 different characters constitutes the KTM`s character font, 256 * 8 bytes, i.e. 2 kbytes, are needed to store the pixel patterns. The KTM uses two identical 2 K-ROMs (type 2316-B) which are switched to give the high speed necessary for a CRT display.

How are the pixel patterns coded? Page 0 of the ROMs comprises the top line of all the 256 patterns, page 1 the next line, etc., page 7 the bottom line of each. Bit 7 corresponds to the leftmost dot, bit 0, accordingly, to the rightmost one. Therefore, the pattern code for any character is to be found at a fixed lo-byte address (determined by the ASCII code), while the hi-byte of its address varies from 0 to 7. The lo-byte address is exactly the ASCII code of the character (0 to $7F ASCII, $80 to $FF graphics), r e a d b a c k w a r d s ! Graphic characters fill the 8-by-8 matrix completely, while ASCII characters leave the left and right margins empty and usually the bottom row, too.

Suppose you want to change the appearance of the tilde ($ 7E) to `ß` (Greek beta, or German `sharp` s). You first invert the ASCII code, reading it from right to left, which gives (in that special case, only, the same code!) $ 7E. This is the lo-byte address. Next, you look up the codes for all pixel rows, top row first, at addresses $07E, $17E, ... $77E:

00 00 00 32 4C 00 00 00.

Rows 4 and 5 are non-empty, resulting in the following pattern:

```
row 4:      ..XX..X.
row 5:      .X..XX..      See the wave-form?
```

Let us now devise the `ß`:

```
row 1:      ..XXX...   = $ 38 at address $ 07E.
row 2:      .X...X..   = $ 44 at address $ 17E.
row 3:      .X.XX...   = $ 58 at address $ 27E.
row 4:      .X...X..   = $ 44 at address $ 37E.
row 5:      .X....X.   = $ 42 at address $ 47E.
row 6:      .XX..X..   = $ 64 at address $ 57E.
```

row 7: .X.XX... = $ 58 at address $ 67E.
row 8: .X..... = $ 40 at address $ 77E.

These values have to be stored to obtain a beta instead of a tilde for a $ 7E. Since the KTM character ROMS are pin-compatible with standard 2716 EPROMs, any user with access to an EPROM programming device can alter the screen appearance of all the KTM characters.

(II) Changing the keyboard encoder

Another 2316-B ROM (Synertek § 02-0050A) contains program codes and the keyboard encoding table. Caution: pins 20 and 21 (unlike 2716`s) are positive chip-select; you need have them at high level in order to read the ROM. Exchanging the ROM for a 2716 EPROM requires to change the jumper settings at J 18 and J 19.

The KTM uses two input lines to check the status of the SHIFT and CONTROL keys, and an 8-by-8 matrix for checking the other keys. This lay-out requires 2 * 64 = 128 bytes of storage for decoding the keys in lower-case and upper-case mode. Not all of the 64 cross-points of the matrix are used (the KTM has 51 keys apart from SHIFT and CTRL). These crosspoints are decoded by a null code. The KTM control functions ALPHA and BREAK are identified by a dedicated bit (bit 7 = 1, = 0 for all other characters). Bits 6 to 0 contain the ASCII code in r e v e r s e d order! For instance, the key `8` (ASCII $ 38) thus encodes to $ 0E.

The keyboard encoding table starts from address $ 723 up to $ 7A2. Standard KTM lay-out is (only lower-case is shown):

U2 (6522)	PA	0	1	2	3	4	5	6	7	ROM addr.
U2 (6522)	PB 0	/	8	7	6	5	4	3	2	1	$ 723-72A
"	PB 1	/			TAB		-	$7E	0	9	$ 72B-732
"	PB 2	/	u	y	t	r	e	w	q	ESC	$ 733-73A
"	PB 3	/			RET	LF	$7D	p	o	i	$ 73B-742
"	PB 4	/	j	h	g	f	d	s	a		$ 743-74A
"	PB 5	/			ALPHA	DEL	$7B	$7C	l	k	$ 74B-752
"	PB 6	/	m	n	b	v	c	x	z		$ 753-75A
U1 (6522)	PB 5	/				SPACE		/	.	,	$ 75B-762

note: some characters I use for control (e.g., $5F) are replaced in the table with their respective ASCII hex codes.

The information given should suffice to change the keyboard according to any desired lay-out. (E.g., my KTM now meets German standard DIN 2137 with Ä,Ö,Ü, ä,ö,ü, and ß at their proper position.)

(III) Attaching a numeric keyboard

Provided you do a little soldering, you may attach any kind of extra keyboard to the KTM by hooking it onto the keyboard matrix lines shown above. You are free to either parallel an existing KTM key with a new one (e.g., the cipher keys 0 to 9), or else use the hitherto free cross-points. I combined both these approaches when attaching a pocket calculator keyboard to ease the tedious keying-in numbers when inputting to SYM-Basic. The design uses the 8 PA-lines as well as PB 0 and 1, and §1 PB 5. Parallel keys are used for 0 to 9, for - and ., while unused cross-points now decode E (for entering floating-point numbers in scientific format), $ 5F (the SYM Basic cancel character), and Return on U1 / PB 5. (The keyboard cost half an hour of work and $ 2.- for a surplus calculator case.)

Let me conclude by expressing the hope that these suggestions will provide KTM users with a terminal both cheap and unbeatably versatile.

A RAE CROSS-REFERENCE LISTER

Kin-ping Kwok sent us three RAE-1 enhancements for review, with permission to publish two of them freely, and asking for comments on the third. The first two programs provide alphabetic and numeric sorts of the RAE-1 label file after assembly. While we ourselves have little need for the numeric sort, we publish it here for a specific reason, to be revealed below. Kwok's alphabetic sort is much faster, in general, than the earlier published alphabetic sort by Cyr, but the Cyr sort is faster if the labels are in close to the proper order.

We compared Cyr vs. Kwok on an extremely long label file. We used the label file for Brown's new Extended Disk Basic; we don't know how many labels it has, but it is some $1568 bytes long! Kwok's numerical sort was used first, to ensure a common starting sequence for the two alphabetic sorts. We also wanted to be able to "break up" any starting alphabetization to some extent; otherwise the "bubble sort" would have an unfair advantage. Kwok is considerably faster, but when applied twice in a row, both trials take the same length of time; the second pass of Cyr is nearly instantaneous, because of the already established sequence.

If you look at the label files following the assembly listings below, you will see why Mr. Kwok's third program really thrilled us. We must admit that RAE's normal label file looks pretty sloppy. Kwok's alphabetized cross-referenced label file adds a real touch of class to a RAE listing, and it will really be helpful in analyzing a DISARAEd listing. Note that macro names and labels (those with !!! and ... prefixes) do not appear in the listings, and that the macro "dummy" variables' "defining" line numbers do not include the # prefix (e. g., ADDRS, FROM, TO).

Mr. Kwok's preliminary version was somewhat awkward to use; it required a preliminary alphabetic sort, and some fancy manipulations if the source file was segmented, i. e., if it required one or more .CTs. The "final" version includes the initial sort, and has simplified the .CT problems. We have not fully tested the .CT workings on large source programs. To speed up the testing we will first modify the program to permit .CT (Disk Filename). One problem we forsee is the possible necessity of dumping partial label files to disk or cassette, with .CTs themselves, since the Cross-Reference files replace the source code files as they are generated, and running out of space is a real possibility.

As soon as this issue goes to press we will give this program highest priority, because of its obvious utility value. We will distribute a combined cassette/disk version using conditionals to permit user selection at assembly time. See shopping list for details. Now here are the two sort programs, with cross referenced label files:

```
0010            ;SORT LABELS IN ALPHANUMERICAL ORDER
0020            ;COPYRIGHT JULY,1981 BY Kin-ping Kwok
0030            ;       ALL RIGHTS RESERVED
0040            .BA $9000
0050            .OS
0060 STST       .DE $104
0070 LBLPTR     .DE $92
0080 CURPTR     .DE $94
0090 USERPTR    .DE $96
0100 TEMP       .DE $135
0110 !!!MOVE    .MD (FROM TO)
0120            IFP FROM-$100
0130            LDA FROM
0140            ***
0150            IFM FROM-$100
0160            LDA *FROM
0170            ***
0180            STA *TO
0190            IFP FROM-$FF
0200            LDA FROM+1
0210            ***
0220            IFM FROM-$FF
0230            LDA *FROM+1
0240            ***
0250            STA *TO+1
0260            .ME
0270 !!!NEXT    .MD (ADDRS)
0280 ...NEXT1   INY
0290            LDA (ADDRS),Y
0300            BPL ...NEXT1
0310            SEC
0320            TYA
0330            ADC *ADDRS
0340            STA *ADDRS
0350            BCC ...NEXT2
0360            INC *ADDRS+1
0370 ...NEXT2   .ME
```



```
                0380 SORT       MOVE (STST LBLPTR)
900A- A0 02     0390            LDY #2
900C- B1 92     0400            LDA (LBLPTR),Y
900E- D0 01     0410            BNE NEXTLBL
9010- 60        0420 SEND       RTS
9011- A0 01     0430 NEXTLBL    LDY #1
                0440            NEXT (LBLPTR)
9022- A0 02     0450 NEXTLBL1   LDY #2
9024- B1 92     0460            LDA (LBLPTR),Y
9026- F0 E8     0470            BEQ SEND
                0480            MOVE (STST CURPTR)
9032- A0 01     0490 COMPARE    LDY #1
9034- C8        0500 COMPARE1   INY
9035- B1 94     0510            LDA (CURPTR),Y
9037- 30 27     0520            BMI COMPARE2
9039- B1 92     0530            LDA (LBLPTR),Y
903B- 30 28     0540            BMI COMPARE3
903D- D1 94     0550            CMP (CURPTR),Y
903F- 90 2F     0560            BCC INSERT
9041- F0 F1     0570            BEQ COMPARE1
                0580 NEXTCUR    NEXT (CURPTR)
9052- A5 94     0590            LDA *CURPTR
9054- C5 92     0600            CMP *LBLPTR
9056- D0 DA     0610            BNE COMPARE
9058- A5 95     0620            LDA *CURPTR+1
905A- C5 93     0630            CMP *LBLPTR+1
905C- D0 D4     0640            BNE COMPARE
905E- F0 B1     0650            BEQ NEXTLBL
9060- B1 92     0660 COMPARE2   LDA (LBLPTR),Y
9062- 09 80     0670            ORA #$80
9064- 2C        0680            .BY $2C
9065- 29 7F     0690 COMPARE3   AND #$7F
9067- D1 94     0700            CMP (CURPTR),Y
9069- 90 05     0710            BCC INSERT
906B- D0 D7     0720            BNE NEXTCUR+1
906D- 0A        0730            ASL A
906E- B0 D4     0740            BCS NEXTCUR+1
9070- A0 FF     0750 INSERT     LDY #$FF
9072- C8        0760 SAVE       INY
9073- B1 92     0770            LDA (LBLPTR),Y
9075- 99 35 01  0780            STA TEMP,Y
9078- 10 F8     0790            BPL SAVE
907A- C0 02     0800            CPY #2
907C- 90 F4     0810            BCC SAVE
907E- A2 00     0820            LDX #0
                0830            MOVE (LBLPTR USERPTR)
9088- C8        0840            INY
9089- A5 96     0850 INS1       LDA *USERPTR
908B- D0 02     0860            BNE INS2
908D- C6 97     0870            DEC *USERPTR+1
908F- C6 96     0880 INS2       DEC *USERPTR
9091- A1 96     0890            LDA (USERPTR,X)
9093- 91 96     0900            STA (USERPTR),Y
9095- A5 96     0910            LDA *USERPTR
9097- C5 94     0920            CMP *CURPTR
9099- D0 EE     0930            BNE INS1
909B- A5 97     0940            LDA *USERPTR+1
909D- C5 95     0950            CMP *CURPTR+1
909F- D0 E8     0960            BNE INS1
90A1- 88        0970            DEY
90A2- 98        0980            TYA
90A3- AA        0990            TAX
90A4- B9 35 01  1000 INS3       LDA TEMP,Y
90A7- 91 96     1010            STA (USERPTR),Y
90A9- 88        1020            DEY
```

```
90AA- 10 F8     1030            BPL INS3
90AC- 8A        1040            TXA
90AD- 4C 1A 90  1050            JMP NEXTLBL+9
                1060            .EN


LABEL FILE:  [ / = EXTERNAL ]     # = LINE DEFINED

 SYMBOL     ;VALUE                CROSS-REFERENCES
 /CURPTR    ;$0094     #0080    0480    0510    0550    0580    0590
            ;           0620    0700    0920    0950
 /LBLPTR    ;$0092     #0070    0380    0400    0440    0460    0530
            ;           0600    0630    0660    0770    0830
 /STST      ;$0104     #0060    0380    0480
 /TEMP      ;$0135     #0100    0780    1000
 /USERPTR   ;$0096     #0090    0830    0850    0870    0880    0890
            ;           0900    0910    0940    1010
 ADDRS      ;$0094      0270    0290    0330    0340    0360
 COMPARE    ;$9032     #0490    0610    0640
 COMPARE1   ;$9034     #0500    0570
 COMPARE2   ;$9060     #0660    0520
 COMPARE3   ;$9065     #0690    0540
 FROM       ;$0092      0110    0120    0130    0150    0160    0190
            ;           0200    0220    0230
 INS1       ;$9089     #0850    0930    0960
 INS2       ;$908F     #0880    0860
 INS3       ;$90A4     #1000    1030
 INSERT     ;$9070     #0750    0560    0710
 NEXTCUR    ;$9043     #0580    0720    0740
 NEXTLBL    ;$9011     #0430    0410    0650    1050
 NEXTLBL1   ;$9022     #0450    ****
 SAVE       ;$9072     #0760    0790    0810
 SEND       ;$9010     #0420    0470
 SORT       ;$9000     #0380    ****
 TO         ;$0096      0110    0180    0250

                0010            ;SORT LABELS IN NUMERICAL ORDER
                0020            ;COPYRIGHT JULY,1981 BY Kin-ping Kwok
                0030            ;       ALL RIGHTS RESERVED
                0040            .BA $9000
                0050            .OS
                0060 STST       .DE $104
                0070 LBLPTR     .DE $92
                0080 CURPTR     .DE $94
                0090 USERPTR    .DE $96
                0100 !!!MOVE    .MD (FROM TO)
                0110            IFP FROM-$100
                0120            LDA FROM
                0130            ***
                0140            IFM FROM-$100
                0150            LDA *FROM
                0160            ***
                0170            STA *TO
                0180            IFP FROM-$FF
                0190            LDA FROM+1
                0200            ***
                0210            IFM FROM-$FF
                0220            LDA *FROM+1
                0230            ***
                0240            STA *TO+1
                0250            .ME
                0260 !!!NEXT    .MD (ADDRS)
                0270 ...NEXT1   INY
                0280            LDA (ADDRS),Y
                0290            BPL ...NEXT1
```



```
                0300            SEC
                0310            TYA
                0320            ADC *ADDRS
                0330            STA *ADDRS
                0340            BCC ...NEXT2
                0350            INC *ADDRS+1
                0360 ...NEXT2   .ME
                0370 SORT       MOVE (STST LBLPTR)
900A- A0 02     0380            LDY #2
900C- B1 92     0390            LDA (LBLPTR),Y
900E- D0 01     0400            BNE NEXTLBL
9010- 60        0410 SEND       RTS
9011- A0 01     0420 NEXTLBL    LDY #1
                0430            NEXT (LBLPTR)
9022- A0 02     0440 NEXTLBL1   LDY #2
9024- B1 92     0450            LDA (LBLPTR),Y
9026- F0 E8     0460            BEQ SEND
                0470            MOVE (STST CURPTR)
9032- A0 01     0480 COMPARE    LDY #1
9034- B1 92     0490            LDA (LBLPTR),Y
9036- D1 94     0500            CMP (CURPTR),Y
9038- 90 48     0510            BCC INSERT
903A- D0 19     0520            BNE NEXTCUR
903C- A2 00     0530            LDX #0
903E- A1 92     0540            LDA (LBLPTR,X)
9040- C1 94     0550            CMP (CURPTR,X)
9042- 90 3E     0560            BCC INSERT
9044- D0 0F     0570            BNE NEXTCUR
9046- C8        0580 COMPARE1   INY
9047- B1 94     0590            LDA (CURPTR),Y
9049- 30 27     0600            BMI COMPARE2
904B- B1 92     0610            LDA (LBLPTR),Y
904D- 30 28     0620            BMI COMPARE3
904F- D1 94     0630            CMP (CURPTR),Y
9051- 90 2F     0640            BCC INSERT
9053- F0 F1     0650            BEQ COMPARE1
                0660 NEXTCUR    NEXT (CURPTR)
9064- A5 94     0670            LDA *CURPTR
9066- C5 92     0680            CMP *LBLPTR
9068- D0 C8     0690            BNE COMPARE
906A- A5 95     0700            LDA *CURPTR+1
906C- C5 93     0710            CMP *LBLPTR+1
906E- D0 C2     0720            BNE COMPARE
9070- F0 9F     0730            BEQ NEXTLBL
9072- B1 92     0740 COMPARE2   LDA (LBLPTR),Y
9074- 09 80     0750            ORA #$80
9076- 2C        0760            .BY $2C
9077- 29 7F     0770 COMPARE3   AND #$7F
9079- D1 94     0780            CMP (CURPTR),Y
907B- 90 05     0790            BCC INSERT
907D- D0 D7     0800            BNE NEXTCUR+1
907F- 0A        0810            ASL A
9080- B0 D4     0820            BCS NEXTCUR+1
9082- A0 FF     0830 INSERT     LDY #$FF
9084- C8        0840 SAVE       INY
9085- B1 92     0850            LDA (LBLPTR),Y
9087- 99 C2 90  0860            STA TEMP,Y
908A- 10 F8     0870            BPL SAVE
908C- C0 02     0880            CPY #2
908E- 90 F4     0890            BCC SAVE
9090- A2 00     0900            LDX #0
                0910            MOVE (LBLPTR USERPTR)
909A- C8        0920            INY
909B- A5 96     0930 INS1       LDA *USERPTR
909D- D0 02     0940            BNE INS2
```

```
909F- C6 97     0950            DEC *USERPTR+1
90A1- C6 96     0960 INS2       DEC *USERPTR
90A3- A1 96     0970            LDA (USERPTR,X)
90A5- 91 96     0980            STA (USERPTR),Y
90A7- A5 96     0990            LDA *USERPTR
90A9- C5 94     1000            CMP *CURPTR
90AB- D0 EE     1010            BNE INS1
90AD- A5 97     1020            LDA *USERPTR+1
90AF- C5 95     1030            CMP *CURPTR+1
90B1- D0 E8     1040            BNE INS1
90B3- 88        1050            DEY
90B4- 98        1060            TYA
90B5- AA        1070            TAX
90B6- B9 C2 90  1080 INS3       LDA TEMP,Y
90B9- 91 96     1090            STA (USERPTR),Y
90BB- 88        1100            DEY
90BC- 10 F8     1110            BPL INS3
90BE- 8A        1120            TXA
90BF- 4C 1A 90  1130            JMP NEXTLBL+9
90C2-           1140 TEMP       .DS 20
                1150            .EN
```

LABEL FILE: [/ = EXTERNAL] # = LINE DEFINED

```
SYMBOL      ;VALUE              CROSS-REFERENCES
/CURPTR     ;$0094    #0080   0470   0500   0550   0590   0630
            ;          0660   0670   0700   0780   1000   1030
/LBLPTR     ;$0092    #0070   0370   0390   0430   0450   0490
            ;          0540   0610   0680   0710   0740   0850
            ;          0910
/STST       ;$0104    #0060   0370   0470
/USERPTR    ;$0096    #0090   0910   0930   0950   0960   0970
            ;          0980   0990   1020   1090
ADDRS       ;$0094     0260   0280   0320   0330   0350
COMPARE     ;$9032    #0480   0690   0720
COMPARE1    ;$9046    #0580   0650
COMPARE2    ;$9072    #0740   0600
COMPARE3    ;$9077    #0770   0620
FROM        ;$0092     0100   0110   0120   0140   0150   0180
            ;          0190   0210   0220
INS1        ;$909B    #0930   1010   1040
INS2        ;$90A1    #0960   0940
INS3        ;$90B6    #1080   1110
INSERT      ;$9082    #0830   0510   0560   0640   0790
NEXTCUR     ;$9055    #0660   0520   0570   0800   0820
NEXTLBL     ;$9011    #0420   0400   0730   1130
NEXTLBL1    ;$9022    #0440   ****
SAVE        ;$9084    #0840   0870   0890
SEND        ;$9010    #0410   0460
SORT        ;$9000    #0370   ****
TEMP        ;$90C2    #1140   0860   1080
TO          ;$0096     0100   0170   0240
```

How do you like the CROSS-REFERENCED Label Files?

Note that the listings above do not include the macro expansions, because of the paper waste in listings with .ES. We also did not provide hex dumps, which would, of course, have included the bytes missing from the source code listings, because every user of these programs must have RAE-1 to use them, and every user of RAE-1 will key in the source codes rather than the object codes, anyway.

Incidentally, the sorting algorithm used by Mr. Kwok in these two programs is known as the "insertion" sort. We have heard of it, certainly, but have never used it before this. It is definitely faster than the usual "bubble"-sort, unless the data to be sorted has (have?) only a very few out-of-place items.

NUMBERING RAE FILES

The use of the RAE NUmber command is not well covered in the Reference Manual. The following example shows how to set both the starting number and interval easily. Suppose you have a source code consisting of several sections, each ending with a .CT, except the last, and you wish the numbers to continue in sequence from one section to the next. This is particularly important in using CROSS REFERENCE, where a single Cross Reference Listing is prepared to cover all sections.

Assume that the starting number of the first section is **0005**, the interval is 5, and the ending number is 1755. You would like the second section to begin with 1760, with interval 5, etc. Use the NU command twice, as follows: NU 0 1755 <cr>, NU 1755 5 <cr>. Disregard any error message (!10, indicating line number overflow) after the first use of NU. The second section is then numbered as specified.

A NEAT RAE TRICK

The following letter from James Duckworth presents a very neat idea we wish we had thought of..... Our excuse for not having thought of it first is that we call all of our supporting programs, such as SWP, PON (printer on), etc., from RAE with the DC command, as "overlays" in the FODS utility program area. This means we have been wasting time waiting for the disk load. We shall be rewriting many of our RAE enhancement packages, including our new XREF, to use Jim's trick.

Jim did not send his letter on cassette, but rather on a thermal printout paper, and we reproduce it as a photocopy of a Xerox copy of his original letter. The printer, whose name we cannot recall at the moment, provides excellent hard copy by "burning" away a silvery appearing coating to reveal a black undercoating. We like the lower case descenders, and the paper Xeroxes reasonably well.

We seriously considered this printer, especially because of its low cost. We decided against it because our main "product" is the newsletter, and any "rough" handling of the printouts in the process of cutting and pasting the camera-ready copy would introduce black appearing scratch marks on the copy. If we had a plain paper copier this would have solved the problem, as we could then have cut and pasted copies. As of this writing, however, the Epson MX-80 is very little more expensive than the thermal printer, but far more versatile.

JAMES J. DUCKWORTH
5 OSAGE AVENUE, ROCKAWAY, N. J.
07866
(201) 625-4413
September 17, 1981

Dear Lux,

This is just a quick note to let other SWP-1 users in on a little trick I discovered, because I got tired of typing "RU $3800", to invoke the SWP program. The first method, which works pretty well, is to put at $0003 through $0006 the code "4C 00 38", and then call SWP with a simple "US", followed by a carriage return. This method is a quite obvious use of RAE's USER command, which transfers control through a vector in ZERO page memory, to a users program.

A not so obvious way to call SWP is to create a dummy label, called "SWP" of course. It is done in the following way. After calling RAE, but at sometime before SWP is needed, type in the following;

```
1SWP .DE $3800 cr
AS cr
```

After the error message, type in;

```
DE 1 cr
```

RAE has now forgotten all about line 1, but a label exists in the LABEL file named "SWP" with an address of $3800. Now a call to RUN SWP does exactly that. The label and its address remain there in memory until over-written or the power is shut off. Now this is what I call a neat trick. It is so much easier to remember "SWP" than some hexadecimal address.

Jim Duckworth

MULTI-PURPOSE RAE PATCH

The following program was intended to be a very simple patch to RAE to permit only one added feature: increasing the length of the SYNCH interval for PUt, for the benefit of those readers who were having troubles reading their RAE files because of sluggish recorder time-constants. Since this involved using PUt, we decided to include the .CT patch, which involves GEt. The program began growing in an uncontrolled fashion when we began adding explanations of how ENter and LOad influence PUt and GEt. As long as we had gone this far, we added some additional features, taking care of ALL the vectors and jumps. We could have added even more, but decided to leave the rest to you.

```
0010 ;      A MULTI-PURPOSE RAE-1 PATCH
0020 ;            18-AUG-81/LUX
0030
0040 ; RAE-1 "ordinarily" uses only from $B6 thru $EA
0050 ; and $EE and $EF on page zero. $B6 is initialized to
0060 ; 60 (RTS), and $EE and $EF are initialized to $00.
0070
0080 ; If >HA S is given as a command, all output is vec-
0090 ; tored through $B6.  Thus $B6 can be used to vector
0100 ; to your printer patch, if desired.  >HA C disables
0110 ; the vectoring.
0120
0130 ; GEt and PUt examine the "flags" at $EE and $EF; if
0140 ; these are zero the "normal" cassette I/O is sup-
0150 ; ported. If these flags are non-zero, GEt and PUt are
0160 ; vectored through $F6/$F7 and $F4/$F5 respectively.
0170
0180 ; Three "undocumented" commands, >LO (for LOad, or LOok-
0190 ; up), >EN (for ENter), and >DC (for Disk Command) are
0200 ; available for linking to disk operating systems (DOS),
0210 ; e.g., FODS or CODOS.  When not required for disk I/O
0220 ; these commands may be used for other purposes, as
0230 ; shown by the following example.
0240
0250 ; Not shown in the example, however is the method by
0260 ; which these (and all RAE commands, for that matter)
0270 ; pass their parameters.  At entry to the vectors the
0280 ; Y register contains the hex value of the position
0290 ; within RAE's CRT Buffer of the first non-space
0300 ; character following the command, e.g., ENTER.
0310
0320 ; If Y=$50, no parameters were passed.  Otherwise the
0330 ; parameters may be "picked-up" and interpreted be-
0340 ; gining at $0135,Y.  Hence, for example, you may
0350 ; add as many "new" commands as you wish to RAE by
0360 ; calling >DC PRINTERON, >DC PRINTEROFF, >DC SWP,
0370 ; >DC SORT, >DC CROSSREF, etc.
0380
```



```
0390 ; USR may also be used with parameter passing, but not
0400 ; ^Y.
0410
0420 ;     GETPATCH ENABLES .CT (CONTINUE ON TAPE), BUT
0430 ;     DISABLES THE ABILITY TO SPECIFY AN ID FOR "GET"
0440 ;     LINKED BY CALLING LOad X (X = ANY ALPHANUMERIC)
0450 ;     UNLINKED BY CALLING LOad, WITH NO PARAMETER
0460
0470 ;     PUTPATCH INCREASES SYNCH DURATION FOR "PUT"
0480 ;     LINKED BY CALLING ENter X (X = ANY ALPHANUMERIC)
0490 ;     UNLINKED BY CALLING ENter, WITH NO PARAMETER
0500
0510 ;     CALLING LOad AND/OR ENter WITH A PARAMETER SETS
0520 ;     INP.FLG AND/OR OUT.FLG TO $01.
0530 ;     THEREAFTER ALL GEts AND/OR PUts ARE VECTORED
0540 ;     THROUGH GET.VEC AND/OR PUT.VEC.
0550
0560 ;     WHEN CALLED WITHOUT A PARAMETER THEY SET
0570 ;     THE FLAGS TO $00.  THIS FEATURE IS
0580 ;     BUILT INTO LOad AND ENter TO OPEN AND
0590 ;     CLOSE NAMED DISK FILES AUTOMATICALLY.
0600 ;     GEts and PUts THEN MANIPULATE ONLY SUBFILES.
0610
0620 ;     THIS PATCH ALSO MODIFIES THE DEFAULT VALUES
0630 ;     TO BETTER SUIT A 16K TO 32K SYSTEM.  THE PATCH
0640 ;     OCCUPIES $06-$A6.  IN A LARGE SYSTEM IT COULD
0650 ;     BE PLACED IN HIGH MEMORY OR IN ROM.
0660
0670 RAEPATCH    .DE $0006
0680             .BA RAEPATCH
0690             .OS
0700
0710 ^Y.JMP      .DE $00
0720 USR.JMP     .DE $03
0730 PRNT.JMP    .DE $B6
0740 RELOCBUF    .DE $C8
0750
0760 INP.FLG     .DE $EE        ;RAE TAPE/DISK INPUT FLAG
0770 LOD.VEC     .DE $F2        ;RAE DISK LOD VECTOR
0780 GET.VEC     .DE $F6        ;RAE DISK GET VECTOR
0790
0800 OUT.FLG     .DE $EF        ;RAE TAPE/DISK OUTPUT FLAG
0810 ENT.VEC     .DE $F0        ;RAY DISK ENT VECTOR
0820 PUT.VEC     .DE $F4        ;RAE DISK PUT VECTOR
0830
0840 DC.VEC      .DE $EC        ;RAE DISK COMMAND (DC) VECTOR
0850
0860 ;     SUPERMON ADDRESSES
0870
0880 SAVER       .DE $8188
0890 BEEP        .DE $8972
0900
0910 ;     RAE ADDRESSES
0920
0930 TXST        .DE $0100
0940 FILE.NO     .DE $0110
0950 RAEWARM     .DE $B0AC
0960 RAEHOT      .DE $B05E
0970 CLEAR_ALL   .DE $E602
0980 U/LOAD1     .DE $EF68
0990 TRANSFER    .DE $EFC4
1000 TAP.NEW     .DE $EFA4
1010
1020 ;     SET ALL VECTORS AND JUMPS
1030
```

```
0006- A9 77     1040 SET.VECS   LDA #L,GETPATCH
0008- 85 F6     1050            STA *GET.VEC
000A- A9 00     1060            LDA #H,GETPATCH
000C- 85 F7     1070            STA *GET.VEC+1
                1080
000E- A9 7F     1090            LDA #L,PUTPATCH
0010- 85 F4     1100            STA *PUT.VEC
0012- A9 00     1110            LDA #H,PUTPATCH
0014- 85 F5     1120            STA *PUT.VEC+1
                1130
0016- A9 8C     1140            LDA #L,LODPATCH
0018- 85 F2     1150            STA *LOD.VEC
001A- A9 00     1160            LDA #H,LODPATCH
001C- 85 F3     1170            STA *LOD.VEC+1
                1180
001E- A9 8D     1190            LDA #L,ENTPATCH
0020- 85 F0     1200            STA *ENT.VEC
0022- A9 00     1210            LDA #H,ENTPATCH
0024- 85 F1     1220            STA *ENT.VEC+1
                1230
0026- A9 8E     1240            LDA #L,DC.PATCH
0028- 85 EC     1250            STA *DC.VEC
002A- A9 00     1260            LDA #H,DC.PATCH
002C- 85 ED     1270            STA *DC.VEC+1
                1280
002E- A9 4C     1290 SET.JMPS   LDA #$4C
0030- 85 00     1300            STA *^Y.JMP
0032- 85 03     1310            STA *USR.JMP
0034- 85 B6     1320            STA *PRNT.JMP
                1330
0036- A9 91     1340            LDA #L,^Y.PATCH
0038- 85 01     1350            STA *^Y.JMP+1
003A- A9 00     1360            LDA #H,^Y.PATCH
003C- 85 02     1370            STA *^Y.JMP+2
                1380
003E- A9 9A     1390            LDA #L,USR.PATCH
0040- 85 04     1400            STA *USR.JMP+1
0042- A9 00     1410            LDA #H,USR.PATCH
0044- 85 05     1420            STA *USR.JMP+2
                1430
0046- A9 97     1440            LDA #L,PRNT.PATCH
0048- 85 B7     1450            STA *PRNT.JMP+1
004A- A9 00     1460            LDA #H,PRNT.PATCH
004C- 85 B8     1470            STA *PRNT.JMP+2
                1480
                1490 ; THIS IS A GOOD TIME TO CHANGE THE DEFAULTS
                1500 ; TO VALUES BETTER SUITED TO A 16K SYSTEM
                1510
004E- A2 00     1520            LDX #$00
0050- BD 6D 00  1530 PARMS      LDA VALUES,X
0053- 9D 00 01  1540            STA TXST,X
0056- E8        1550            INX
0057- E0 08     1560            CPX #$08
0059- 90 F5     1570            BCC PARMS
005B- AD 75 00  1580            LDA VALUES+8
005E- 85 C8     1590            STA *RELOCBUF
0060- AD 76 00  1600            LDA VALUES+9
0063- 85 C9     1610            STA *RELOCBUF+1
0065- A2 00     1620            LDX #$00
0067- 20 02 E6  1630            JSR CLEAR_ALL
                1640
006A- 4C AC B0  1650            JMP RAEWARM
                1660
006D- 00 02 FD  1670 VALUES     .BY $00 $02 $FD $3F $00
0070- 3F 00
```



```
0072- 20 FD 3F  1680            .BY $20 $FD $3F $F0 $3F
0075- F0 3F
                1690
                1700 ;    HERE ARE THE PATCHES
                1710
0077- A9 00     1720 GETPATCH   LDA #$00
0079- 8D 10 01  1730            STA FILE.NO
007C- 4C 68 EF  1740            JMP U/LOAD1
                1750
007F- 20 88 81  1760 PUTPATCH   JSR SAVER
0082- A9 00     1770            LDA #$00
0084- 20 C4 EF  1780            JSR TRANSFER
0087- A9 04     1790            LDA #$04      ;REPLACE THE BUILT-IN $01
0089- 4C A6 EF  1800            JMP TAP.NEW+2
                1810
008C- 60        1820 LODPATCH   RTS
                1830
008D- 60        1840 ENTPATCH   RTS
                1850
                1860 ;    REPLACE THE CALLS TO "BEEP" BELOW WITH YOUR
                1870 ;    DESIRED SUBROUTINES.  NOTE HOW "JSR" AND "JMP"
                1880 ;    ARE USED BELOW TO PROVIDE CORRECT RETURNS TO RAE.
                1890
008E- 4C 72 89  1900 DC.PATCH   JMP BEEP      ;WHY NOT?
                1910
0091- 20 72 89  1920 ^Y.PATCH   JSR BEEP      ;WHY NOT?
0094- 4C 5E B0  1930            JMP RAEHOT    ;OR RAYWARM, IF YOU PREFER
                1940
0097- 4C 72 89  1950 PRNT.PATCH JMP BEEP      ;WHY NOT?
                1960
009A- 20 72 89  1970 USR.PATCH  JSR BEEP      ;WHY NOT?
009D- 4C 5E B0  1980            JMP RAEHOT    ;OR RAYWARM, IF YOU PREFER
                1990 ; TO USE RAE MULTIPATCH, ENTER RAE AT COLDSTART,
                2000 ; EXIT WITH CONT C, PATCH (AND RE-ENTER WITH .G 6 <cr>
                2010            .EN
```

LABEL FILE: / = EXTERNAL # = LINE DEFINED

SYMBOL	;VALUE		CROSS-REFERENCES			
/BEEP	;$8972	#0890	1900	1920	1950	1970
/CLEAR_ALL	;$E602	#0970	1630			
/DC.VEC	;$00EC	#0840	1250	1270		
/ENT.VEC	;$00F0	#0810	1200	1220		
/FILE.NO	;$0110	#0940	1730			
/GET.VEC	;$00F6	#0780	1050	1070		
/INP.FLG	;$00EE	#0760	****			
/LOD.VEC	;$00F2	#0770	1150	1170		
/OUT.FLG	;$00EF	#0800	****			
/PRNT.JMP	;$00B6	#0730	1320	1450	1470	
/PUT.VEC	;$00F4	#0820	1100	1120		
/RAEHOT	;$B05E	#0960	1930	1980		
/RAEPATCH	;$0006	#0670	0680			
/RAEWARM	;$B0AC	#0950	1650			
/RELOCBUF	;$00C8	#0740	1590	1610		
/SAVER	;$8188	#0880	1760			
/TAP.NEW	;$EFA4	#1000	1800			
/TRANSFER	;$EFC4	#0990	1780			
/TXST	;$0100	#0930	1540			
/U/LOAD1	;$EF68	#0980	1740			
/USR.JMP	;$0003	#0720	1310	1400	1420	
/^Y.JMP	;$0000	#0710	1300	1350	1370	
DC.PATCH	;$008E	#1900	1240	1260		
ENTPATCH	;$008D	#1840	1190	1210		
GETPATCH	;$0077	#1720	1040	1060		

```
LODPATCH    ;$008C    #1820    1140    1160
PARMS       ;$0050    #1530    1570
PRNT.PATCH  ;$0097    #1950    1440    1460
PUTPATCH    ;$007F    #1760    1090    1110
SET.JMPS    ;$002E    #1290    ****
SET.VECS    ;$0006    #1040    ****
USR.PATCH   ;$009A    #1970    1390    1410
VALUES      ;$006D    #1670    1530    1580    1600
^Y.PATCH    ;$0091    #1920    1340    1360
```

AN IMPROVED PRINTER PATCH

We moved our MX-80 printer from the B port to the A port to free PB6 and PB7 for counter/timer applications. We also modified our method of setting PCR to do so without affecting CB1 and CB2 when activating the printer patch, so that these lines could be used for other purposes.

NOTE THAT THE A AND B PORTS HANDLE HANDSHAKING DIFFERENTLY; FAILURE TO ALLOW FOR THIS WILL CAUSE SYSTEM HANGUP! CB2 pulses low only after a write operation to PBD, while CA2 pulses low after either a write to or a read from PAD. The BIT test for BUSY must therefore be made at the no-handshake PAD register at $XX0F.

The original patch was designed to work with RAE-1, and would not handle the "up-arrow" of BAS-1 properly. The byte at $C1 is a "JMP" in BAS-1, and the high byte of the start address of the relocating buffer for RAE-1. By checking this memory location when the "up-arrow" occurs, the correct handling for BAS-1 is provided. Here are all required program changes for the "improved" parallel patch:

```
0010  ;ADD THE FOLLOWING DEFINITIONS
0020
0030  PAD         .DE $A801
0040  PADHI       .DE $A80F
0050  PADD        .DE $A803
0060
0070  ;REPLACE LINES 390-420 WITH THE FOLLOWING
0080
0090              LDA PCR
0100              AND #%11110000
0110              ORA #%00001010
0120              STA PCR
0130              LDA #%01111111
0140              STA PADD
0150
0160  ;INSERT THE FOLLOWING BETWEEN LINES 770 AND 780
0170
0180              LDA #$C9
0190              CMP #$4C
0200              BEQ PRINT^
0210
0220  ;REPLACE LINES 960 980 WITH THE FOLLOWING
0230
0240  WAIT        BIT PADHI
0250              BMI WAIT
0260              STA PAD
```

STILL MORE ON PRINTERS

As you can see above, we have learned to underline from RAE and SWP on the Epson! And installed a "software switch" to turn the printer off and on, as well. In addition, we now can also use the horizontal tab features of the Epson.

As you know, RAE handles CTRL H, I, and Q in peculiar ways. Unfortunately, these peculiar ways conflict with the Epson way of doing things. Epson uses CTRL Q to enable the printer (and CTRL S to disable it), CTRL I for tab control (in a manner inconsistent with RAE) and CTRL H in a manner which, in effect, allows backspacing without erasure, thereby permitting underlining, and, if desired, overstriking).

Fortunately both RAE and the Epson treat the DELETE ($7F) in essentially the same way (except that RAE does echo a "\". As will be seen below, this feature makes for "clean" soft (on the CRT) copy.

The trick is to have RAE echo and output CTRL U, V, and W (or any other unused control characters) as if they were CTRL H, I, and Q. The first line of the listing from which this text was formatted prints on the terminal as ^Q, although CTRL W was actually keyed in and stored in RAE. This turns on the printer. The second line contains the SWP instruction .L, to indicate a fresh line start. The third line contains the data "STILL MORE ON PRINTERS" followed by 22 "^H"s followed by the correct sequence of underlines and spaces _____ ____ __ ________. The ^H's were entered as CTRL U's. The last line of the listing contains a "^S", entered as CTRL S.

To turn the printer on and off from RAE's command mode, enter CTRL W or CTRL S, followed by a DELETE ($7F, RAE echoes "\"). The DELETE prevents an error message from cluttering up the screen.

Here is the section of the printer patch which does the job:

```
>pr 800 1070
0800              BNE NOT^
0810              LDA #$C9      ;BAS-1 STORES A $4C HERE
0820              CMP #$4C
0830              BEQ PRINT^    ;IF IN BASIC, PRINT THE "^"
0840              TSX
0850              LDA $103,X    ;RAE "stacks" control codes
0860              CMP #$20
0870              BCS PRINT^    ;Not a control code
0880              PHA
0890              LDA #$40      ;RAE will convert to null
0900              STA $103,X
0910              LDA #'^
0920              JSR TOUT
0930              PLA
0940  U_H         CMP #$15
0950              BNE V_I
0960              LDA #$08
0970              BPL OK
0980  V_I         CMP #$16
0990              BNE W_Q
1000              LDA #$09
1010  W_Q         CMP #$17
1020              BNE OK
1030              LDA #$11
1040  OK          PHA
1050              CLC
1060              ADC #$40      ;Make it printable!
1070              JSR TOUT
```

Note that to permit "Printer Select" from the host computer, DIP Switch 8 must be set to the "Select Not Fixed" position, and the printer then powers-on as "non-READY". You may wish to include a LDA #$11, and a JSR OUTCHR in the Initialization Routine to make READY the "default" condition. $11 is the ASCII code for CTRL Q (DC1). An external hardware on-off switch can also be provided. Grounding pin 36 on the Amphenol connector will turn the printer on, i.e., "select" it. The hardware and "software" switches are wire-"or"ed together (active low). Dip Switch 8 essentially grounds pin 36; CTRL Q sets a flip-flop which does the same thing; CTRL S resets the flip-flop.

ON WORD PROCESSORS

Most of our work involves the use of RAE rather than BASIC, hence we find it convenient to use SWP (a very highly customized version) as our word processor. Actually we use RAE as the Text Editor, and SWP only as the Formatter.

Many of our readers prefer BASIC to RAE, and do not even have RAE installed. For them we are distributing KWOK'S BASIC WORD PROCESSOR (BWP-1). We have tried, and liked BWP-1, but have not yet built up any real skill in its use.

We have tried TECO, a very popular Text Editor developed by DEC for their PDP- systems, and very graciously placed in the public domain by them. Many months ago Frank Winter sent us a Formatter Supplement to the TEC65 (editor) program package distributed by the 6502 Program Exchange. Shortly thereafter, Dale Holt sent us a complete package for review, but the source code was not in RAE format. Dick Albers worked with Dale, converting source code to RAE format, and helping to debug and enhance Dale's original version.

The Holt/Albers version of SYM-TECO should be ready for distribution early next year. It is stand-alone, requiring neither BAS-1 or RAE-1 (although the latter would be very helpful in customizing it).

Aside from the obvious cost savings, the major advantage of TECO lies in its "universality", in that many time-share users learn TECO as their first word processor, and TECO-type word processors are available for most systems.

Denny Hall, who uses his Apple II for most applications, feels that the SYM-1 is a better way to go for word processing, and that the cost of a SYM-based word processor to supplement his APPLE is much less than the cost of the added "cards" necessary to convert the Apple to a word processor. Also, he has found, as we have, that two individual systems can, at times, be more useful than one dual-purpose system, since they can be used simultaneously on different tasks.

(Parenthetically speaking, one of our "one-of-these-days" ambitions is to develop an interface card (and the software!) to join the Apple and the SYM. A single SYM would, at much lower cost, replace many RS-232, Serial/Parallel I/O, Printer Interface, 80 Character/Lower Case, etc., add-on cards for the Apple. What a market for this!)

At any rate, Denny is putting the "finishing" touches (we are sure he will continue to provide continuing customer support) to a SYM-based Word Processor, with special control features, for a student with limited use of his hands. Denny plans to OEM SYM-based word processors, standard or customized with, for example, large character sizes on the CRT, verbal feedback, elimination of the need to strike two keys simultaneously as for CTRL characters, etc. He had planned to use RAE/SWP, and is doing so for his first systems, but is seriously considering SYM/TECO in EPROM instead. The student for whom he is building the first system will also be using the system as a terminal (from home) to the school's PDP-11, on which TECO is also available, and the compatibility will be a strong plus! Incidentally, we both consider the Novation D-CAT Modem (the "D" is for Direct Connection) to be the "best-buy" for computer networking.

MORE ON BUFFERING

We have mentioned that "over"-buffering of the SYM's Address and Data Busses can cause the kinds of problems buffering is supposed to solve. The following extract from a recent letter presents additional experience with "overkill" in buffering:

> Another point, and this is really the last one. You remember Lux that I wrote you about my memory problems some time ago when I bought FORTH. [Yes I certainly do remember, Marc, how discouraged and frustrated you were over the memory problems you were having at that time! - Lux]
>
> Here is my mistake; perhaps it can help others: I decoded the addresses for the RAMs with 2 74LS138s. The first for the 8K blocks, the other for the 1K blocks. Also on each card I buffered Address as well as Data lines even when they were already buffered on my mother-board. I eliminated the buffers on each card and replaced the 74LS138 with 74S138 and now everything works fine.
>
> Yours SYMcerely,
>
> Marc SYMons
> Woudlaan,50
> B-1970 Wezembeek-Oppem
> Belgium

A RAE CLOCK/TIMER

Here is a truly powerful utility for RAE-1! It provides the time and date information on request, and even inserts the information into your RAE file at Line 0000, so that you can mark a file with the latest revision date and time.

Here are a few precautions on its use: It will not interfere with Cassette I/O at the normal 1400 baud rate, nor with serial terminal I/O at less than 1200 baud. It also will not interfere with Disk I/O, since the DOS checks all reads and writes against check sums until it is satisfied that all has gone well (it will, of course, give up, and let you know about it, after a prespecified number of failures). The interrupt service time is short enough to cause no problems, except possibly at midnight, especially at the end of the month, and New Year's Eve, when you probably are not at the computer (at least leave the terminal for a few seconds).

Operation of the terminal at 4800 baud, however, could cause problems. If interrupt occurs during a character output time the character on the terminal will be garbled. This is not too serious. An interrupt during a character input time could be disasterous. What if the character was garbled into a CTRL B, which sends the SYM into BASIC, and destroys pages 0 and 1??????????? Dick Albers guards against this kind of near catastrophe by calling an input program which calls SEI during character input time (this feature should have been in SUPERMON!) to inhibit interrupts.

We would like to see the matter handled by having a keystroke also create an IRQ (or maybe an NMI) as in the Moser Paddle Game in an early issue, and giving keystrokes priority over timer interrupts. It would be fun to work out the details; meanwhile we will keep the clock running whenever we are in RAE, to get some idea of the frequency of catastrophes! We are putting our faith in being able to recover any "lost" RAE files by the

It just happened!!! Hit the <cr> and an up-arrow N appeared. Hit another key and the program went into limbo.....

As we were saying, we can recover "lost" files by the method described in the RAE-1 REFERENCE DATA CARD sent out with RAE NOTES #1, on how to recover after "accidental" CLear. If the problem occurs too frequently, we shall write the priority interrupt handling program. Incidentally, the use of an interrupt driven clock is one more justification for using a parallel interfaced printer rather than a serial driven one; clock interrupts can't botch up the hard copy. And now, here's the program:

(15:05:46 SUN 1 NOV 1981)

```
                        0000            ; 12:51:58 SUN  1 NOV 1981
                        0010            ; 21:45:21 SUN 20 SEP 1981
                        0020            ; RAE CLOCK
                        0030            ; By Richard R. Albers
                        0040
                        0050            ; Assemble program (requires $387 bytes)
                        0060            ; RUn LINK to link to Supermon and RAE
                        0070            ; USer S to Set time/date (prompts given)
                        0080            ; USer P to Print time on CRT
                        0090            ; USer I to Input time into RAE file (line #0)
                        0100
                        0110            .BA $9000
                        0120            .OS
                        0130 UIRQVC     .DE $A678    User IRQ vector
                        0140 T1LL2      .DE $A804 Read counter/clear IRQ
                        0150 T1CH       .DE $A805 Write hi latch/transfer
                        0160 T1LL       .DE $A806 Write low latch
                        0170 ACR        .DE $A80B Aux control reg
                        0180 IER        .DE $A80E Interrupt control reg
                        0190 ACCESS     .DE $8B86
                        0200 OUTBYT     .DE $82FA
                        0210 OUTCHR     .DE $8A47
                        0220 BLK3       .DE $8739
                        0230 RAEHOT     .DE $B0B1
                        0240 ERMSG      .DE $8171
                        0250 BUFF       .DE $135
                        0260 TXST       .DE $100
                        0270 TPRES      .DE $D3
                        0280 P3L        .DE $A64A
                        0290 P3H        .DE $A64B
                        0300 P2L        .DE $A64C
                        0310 P2H        .DE $A64D
                        0320 P1L        .DE $A64E
                        0330 P1H        .DE $A64F
                        0340 INBYTE     .DE $81D9
                        0350 CRLF       .DE $834D
                        0360
                        0370            ; LINKING SUBROUTINES
                        0380
9000- 20 86 8B          0390 LINK       JSR ACCESS   Allow writing SYSRAM
9003- A9 37             0400            LDA #L,ISERV
9005- 8D 78 A6          0410            STA UIRQVC   Link interrupt routine
9008- A9 90             0420            LDA #H,ISERV
900A- 8D 79 A6          0430            STA UIRQVC+1
900D- A9 19             0440            LDA #L,RAEUS Link to RAE-1 "USr" command
900F- 85 04             0450            STA *$04
9011- A9 91             0460            LDA #H,RAEUS
9013- 85 05             0470            STA *$05
9015- A9 4C             0480            LDA #$4C     "JMP"
9017- 85 03             0490            STA *$03         Replace the
9019- 60                0500            RTS                  RTS and NOP here with
901A- EA                0510            NOP                  CLC and BRK, respectively,
                        0520                                 to call from RAE via FODS.
                        0530            ; START THE CLOCK
                        0540
901B- 78                0550 START      SEI          Disable IRQ
901C- A9 C0             0560            LDA #$C0     Initialize interrupt reg
901E- 8D 0E A8          0570            STA IER
9021- A9 40             0580            LDA #$40     Free-run timer 1
9023- 8D 0B A8          0590            STA ACR      Clear all else
9026- A9 51             0600            LDA #$51     Adjust this to Xtal
9028- 8D 06 A8          0610            STA T1LL     Set timer latches
902B- A9 C3             0620            LDA #$C3      (low then high)
902D- 8D 05 A8          0630            STA T1CH     And trigger timer 1
9030- A9 00             0640            LDA #$00
```



```
9032- 8D BD 90          0650            STA TIMR     Clear fractions of secs
9035- 58                0660            CLI          Allow IRQs
9036- 60                0670            RTS
                        0680
9037- 48                0690 ISERV      PHA          Interrupt service routine
9038- 8A                0700            TXA          Save regs
9039- 48                0710            PHA
903A- F8                0720            SED          Work in decimal mode
903B- A2 00             0730            LDX #$00     Set up index
903D- 18                0740 TIMLP      CLC
903E- BD BD 90          0750            LDA TIMR,X   Get current value
9041- 69 01             0760            ADC #$01     Decimal increment
9043- B0 0E             0770            BCS NEW      Only on new century
9045- 9D BD 90          0780            STA TIMR,X   Store new value
9048- DD A8 90          0790            CMP TABL,X   Check limit
904B- 90 0C             0800            BCC CKDOM    May be done
904D- E0 05             0810            CPX #$05     Day of month reg?
904F- F0 13             0820            BEQ DOM      Yes, check days in month
9051- A9 00             0830            LDA #$00     Otherwise,
9053- 9D BD 90          0840 NEW        STA TIMR,X   Zero this register,
9056- E8                0850 MORE       INX          And index next reg.
9057- 10 E4             0860            BPL TIMLP    (Always)
                        0870
                        0880            ;  LOCK DAY OF MONTH INCREMENT TO
                        0890            ;  DAY OF WEEK (SUN-SAT) INCREMENT
                        0900
9059- E0 04             0910 CKDOM      CPX #$04     Just do day of week?
905B- F0 F9             0920            BEQ MORE     Inc day of month also
                        0930
905D- AD 04 A8          0940 RETURN     LDA T1LL2    Clear 6522 interrupt flag
9060- 68                0950            PLA          Restore registers
9061- AA                0960            TAX
9062- 68                0970            PLA
9063- 40                0980            RTI          Restore flags & return
                        0990
                        1000            ;  CHECK FOR END OF MONTH
                        1010
9064- AE C3 90          1020 DOM        LDX MONTHS   Get current month
9067- E0 01             1030            CPX #$01     February?
9069- F0 0E             1040            BEQ FEBR
906B- AD C2 90          1050            LDA DAY/MO   Get current day
906E- DD B1 90          1060            CMP DTAB,X   At limit for month?
9071- 90 EA             1070            BCC RETURN   No, done
9073- A2 05             1080 REMO       LDX #$05     Restore index to day/mo
9075- A9 01             1090            LDA #$01     Start new month at day 1
9077- D0 DA             1100            BNE NEW      (Always)
                        1110
                        1120            ;  DETERMINE LENGTH OF FEBRUARY
                        1130
9079- AD C4 90          1140 FEBR       LDA YEARS    Check for leap year
907C- F0 1D             1150            BEQ CKCENT   Even century
907E- 29 0F             1160            AND #$0F
9080- 8D C9 90          1170            STA SCRA     Save units
9083- AD C4 90          1180            LDA YEARS
9086- 29 10             1190 CFEB       AND #$10     Get tens odd/even bit
9088- 4A                1200            LSR A        Clears carry
9089- 4A                1210            LSR A
908A- 4A                1220            LSR A
908B- 6D C9 90          1230            ADC SCRA     Add shifted tens & units
908E- 29 03             1240            AND #$03     Get divide by 4 remainder
9090- D0 E1             1250            BNE REMO     Not leap year; Mar 1
9092- AD C2 90          1260            LDA DAY/MO   Check current day #
9095- C9 30             1270            CMP #$30     Leap year limit
9097- 90 C4             1280            BCC RETURN   Leave day = 29
9099- B0 D8             1290            BCS REMO     Start March
```

```
                        1300
                        1310            ;  DETERMINE LENGTH OF FEBRUARY IN
                        1320            ;  EVEN NUMBERED CENTURIES
                        1330            ;   (current calendar only)
                        1340
909B- AD C5 90  1350 CKCENT     LDA CENTURY  Get year divided by 400
909E- 29 0F     1360            AND #$0F
90A0- 8D C9 90  1370            STA SCRA
90A3- AD C5 90  1380            LDA CENTURY
90A6- D0 DE     1390            BNE CFEB     (Always)
                        1400
                        1410            ;  TABLE OF LIMITS: Fractions >> Centuries
                        1420
90A8- 20 60 60  1430 TABL       .BY $20 $60 $60 $24 $08 $29 $12 $9A $99
90AB- 24 08 29
90AE- 12 9A 99
                        1440            ; Year limit $9A allows year = 99
                        1450
                        1460            ;  DAYS PER MONTH LIMITS
                        1470
90B1- 31 28 31  1480 DTAB       .BY $31 $28 $31 $30 $31 $30
90B4- 30 31 30
90B7- 31 31 30  1490            .BY $31 $31 $30 $31 $30 $31
90BA- 31 30 31
90BD- 00        1500 TIMR       .BY $00       ;00  ;Fractions of seconds
90BE- 00        1510 SECONDS    .BY $00       ;01
90BF- 00        1520 MINUTES    .BY $00       ;02
90C0- 00        1530 HOURS      .BY $00       ;03
90C1- 00        1540 DAY/WK     .BY $00       ;04  ;Day of week, 0 = SUN
90C2- 00        1550 DAY/MO     .BY $00       ;05  ;Day of month, 1 - 31
90C3- 00        1560 MONTHS     .BY $00       ;06  ;Month, 0 - 11
90C4- 00        1570 YEARS      .BY $00       ;07  ;Low digits of year, 00 - 99
90C5- 00        1580 CENTURY    .BY $00       ;08  ;High digits of year
90C6- 20        1590            .BY $20       ;09  ;Space
90C7- 3A        1600            .BY $3A       ;0A  ;Colon
90C8- 3B        1610            .BY $3B       ;0B  ;Semicolon
90C9- 00        1620 SCRA       .BY $00       ;0C  ;Scratch area
90CA- 00        1630 SCRB       .BY $00
90CB- 00        1640 SCRC       .BY $00
                        1650
90CC- 20 53 55  1660 DWTAB      .BY ' SUN' ' MON' ' TUE' ' WED'
90CF- 4E 20 4D
90D2- 4F 4E 20
90D5- 54 55 45
90D8- 20 57 45
90DB- 44
90DC- 20 54 48  1670            .BY ' THU' ' FRI' ' SAT'
90DF- 55 20 46
90E2- 52 49 20
90E5- 53 41 54
90E8- 20 4A 41  1680 MOTAB      .BY ' JAN' ' FEB' ' MAR'
90EB- 4E 20 46
90EE- 45 42 20
90F1- 4D 41 52
90F4- 20 41 50  1690            .BY ' APR' ' MAY' ' JUN'
90F7- 52 20 4D
90FA- 41 59 20
90FD- 4A 55 4E
9100- 20 4A 55  1700            .BY ' JUL' ' AUG' ' SEP'
9103- 4C 20 41
9106- 55 47 20
9109- 53 45 50
910C- 20 4F 43  1710            .BY ' OCT' ' NOV' ' DEC '
910F- 54 20 4E
9112- 4F 56 20
9115- 44 45 43
```



```
9118- 20
                        1720
                        1730            ;  RAE'S USr COMMAND ENTRY
                        1740
9119- A8        1750 RAEUS      TAY           Save argument
911A- A9 B0     1760            LDA #H,RAEHOT
911C- 48        1770            PHA           Push return address
911D- A9 B0     1780            LDA #L,RAEHOT-1
911F- 48        1790            PHA
9120- 98        1800            TYA
9121- C9 41     1810            CMP #'A       Test for alpha
9123- 90 10     1820            BCC NOGOOD
9125- 29 5F     1830            AND #$5F      Be sure it's u/c
9127- 48        1840 PENT       PHA           Save argument
9128- C9 50     1850            CMP #'P       Print time?
912A- F0 12     1860            BEQ COMPT
912C- C9 49     1870            CMP #'I       Input time to RAE?
912E- F0 0E     1880            BEQ COMPT
9130- C9 53     1890            CMP #'S       Set time?
9132- F0 06     1900            BEQ TSET
9134- 68        1910            PLA           Clean stack
9135- 38        1920 NOGOOD     SEC
9136- 98        1930            TYA           Get original input
9137- 4C 71 81  1940            JMP ERMSG     Invalid argument
913A- 68        1950 TSET       PLA           Discard "S"
913B- 4C 4F 92  1960            JMP TIMSET
                        1970
                        1980            ; PUT TIME INTO BUFFER, FORMATTED
                        1990
913E- 78        2000 COMPT      SEI           Prevent time changes
913F- A2 02     2010            LDX #$02
9141- 8E C9 90  2020            STX SCRA      Set BUFF index
9144- A2 00     2030            LDX #$00      Clear DRTAB index and
9146- 8E 35 01  2040            STX BUFF      Load BCD line #
9149- 8E 36 01  2050            STX BUFF+1
914C- BD 79 93  2060 NEXTT      LDA DRTAB,X   Get index byte
914F- F0 50     2070            BEQ CDONE     Terminate time transfer
9151- 30 0A     2080            BMI SPECL     Store ASCII char(s)
9153- A8        2090            TAY           Index into TIMR
9154- B9 BD 90  2100            LDA TIMR,Y    Get time byte
9157- 20 26 92  2110            JSR STOBCD    Store it as ASCII
915A- E8        2120 LRET       INX
915B- D0 EF     2130            BNE NEXTT     (Always)
                        2140
915D- 8D CB 90  2150 SPECL      STA SCRC      Save for test
9160- A9 01     2160            LDA #$01
9162- 2C CB 90  2170            BIT SCRC      Test bit 6
9165- 70 02     2180            BVS SHORT     Store 1 char
                        2190
9167- A9 04     2200 LONG       LDA #$04      4 chars to store
9169- 8D CB 90  2210 SHORT      STA SCRC
916C- BD 79 93  2220            LDA DRTAB,X   Get index again
916F- 29 3F     2230            AND #$3F      Mask off flag bits
9171- A8        2240            TAY
9172- B9 BD 90  2250            LDA TIMR,Y    Get char to store
9175- C0 09     2260            CPY #$09      Day, month, or other?
9177- B0 1D     2270            BCS SP;:      Other
                        2280
9179- C9 10     2290            CMP #$10
917B- 90 08     2300            BCC X4
917D- F0 04     2310            BEQ NOV
917F- A9 0B     2320            LDA #$0B      Must be December
9181- D0 02     2330            BNE X4        Always
9183- A9 0A     2340 NOV        LDA #$0A
9185- 0A        2350 X4         ASL A         X4=Index for day
9186- 0A        2360            ASL A
```

```
9187- C0 06     2370            CPY #$06      Month?
9189- D0 07     2380            BNE DAY
918B- 18        2390 MON        CLC
918C- D8        2400            CLD
918D- 69 1C     2410            ADC #$1C Add 28 to index month
918F- EE CB 90  2420            INC SCRC      ;Need five bytes (two spaces)
                2430
9192- A8        2440 DAY        TAY
9193- B9 CC 90  2450 NXTD       LDA DWTAB,Y   Store day of week
9196- 20 3E 92  2460 SP;:       JSR STOASC
9199- CE CB 90  2470            DEC SCRC      Count chars
919C- F0 BC     2480            BEQ LRET
919E- C8        2490            INY
919F- D0 F2     2500            BNE NXTD      (Always)
                2510
91A1- 58        2520 CDONE      CLI           Allow time changes
91A2- AE C9 90  2530            LDX SCRA      Get BUFF index
91A5- BD 34 01  2540            LDA BUFF-1,X
91A8- 09 80     2550            ORA #$80      Mark line end
91AA- 9D 34 01  2560            STA BUFF-1,X
91AD- AD 47 01  2570            LDA BUFF+$12
91B0- C9 30     2580            CMP #$30      Test for leading zero
91B2- D0 05     2590            BNE NLZ
91B4- A9 20     2600            LDA #$20      Replace with a space
91B6- 8D 47 01  2610            STA BUFF+$12
91B9- 68        2620 NLZ        PLA           Check argument
91BA- C9 50     2630            CMP #'P       Print time only?
91BC- F0 5C     2640            BEQ TIMOUT
                2650
                2660            ; MOVE TIME INTO RAE TEXT FILE
                2670
91BE- 20 86 8B  2680 MOVFIL     JSR ACCESS    Move existing text
91C1- AD 01 01  2690            LDA TXST+1
91C4- 8D 4D A6  2700            STA P2H
91C7- AD 00 01  2710            LDA TXST
91CA- 8D 4C A6  2720            STA P2L
91CD- 18        2730            CLC
91CE- 69 1D     2740            ADC #$1D Add 29 bytes
91D0- 8D 4E A6  2750            STA P1L
91D3- AD 01 01  2760            LDA TXST+1
91D6- 69 00     2770            ADC #$00      Pick up any carry
91D8- 8D 4F A6  2780            STA P1H
91DB- A5 D3     2790            LDA *TPRES
91DD- 69 02     2800            ADC #$02      Include end-of-file
91DF- 8D 4A A6  2810            STA P3L
91E2- A5 D4     2820            LDA *TPRES+1
91E4- 69 00     2830            ADC #$00      Pick up any carry
91E6- 8D 4B A6  2840            STA P3H
91E9- A5 D3     2850            LDA *TPRES    Compute new end
91EB- 69 1D     2860            ADC #$1D
91ED- 85 D3     2870            STA *TPRES
91EF- A5 D4     2880            LDA *TPRES+1
91F1- 69 00     2890            ADC #$00      Pick up any carry
91F3- 85 D4     2900            STA *TPRES+1
91F5- 20 40 87  2910            JSR BLK3+7    Let Supermon move it
91F8- AD 00 01  2920            LDA TXST
91FB- 85 CA     2930            STA *$CA      Set up indirect addr
91FD- AD 01 01  2940            LDA TXST+1
9200- 85 CB     2950            STA *$CB
9202- A0 00     2960            LDY #$00
9204- B9 35 01  2970 MOVIT      LDA BUFF,Y    Move from BUFF to text
9207- 91 CA     2980            STA ($CA),Y
9209- 30 03     2990            BMI PRT
920B- C8        3000            INY
920C- 10 F6     3010            BPL MOVIT     (Always)
```



```
                3020
920E- A9 00     3030 PRT        LDA #$00      Print line # as "0000"
9210- 20 FA 82  3040            JSR OUTBYT
9213- 20 FA 82  3050            JSR OUTBYT
9216- A2 01     3060            LDX #$01      Print the rest of line
9218- D0 02     3070            BNE OUT       (Always)
                3080
921A- A2 03     3090 TIMOUT     LDX #$03      Print the time
921C- E8        3100 OUT        INX
921D- BD 35 01  3110            LDA BUFF,X
9220- 20 47 8A  3120            JSR OUTCHR
9223- 10 F7     3130            BPL OUT
9225- 60        3140            RTS
                3150
9226- 8E CA 90  3160 STOBCD     STX SCRB      Save DRTAB index
9229- AE C9 90  3170            LDX SCRA      Get BUFF index
922C- 48        3180            PHA           Save byte to store
922D- 4A        3190            LSR A         Store high nibble
922E- 4A        3200            LSR A
922F- 4A        3210            LSR A
9230- 4A        3220            LSR A
9231- 09 30     3230            ORA #$30      Convert to ASCII
9233- 9D 35 01  3240            STA BUFF,X
9236- E8        3250            INX
9237- 68        3260            PLA
9238- 29 0F     3270            AND #$0F      Store low nibble
923A- 09 30     3280            ORA #$30
923C- D0 06     3290            BNE STOB      (Always)
                3300
923E- 8E CA 90  3310 STOASC     STX SCRB      Save DRTAB index
9241- AE C9 90  3320            LDX SCRA      Get BUFF index
9244- 9D 35 01  3330 STOB       STA BUFF,X
9247- E8        3340            INX
9248- 8E C9 90  3350            STX SCRA      Save BUFF index
924B- AE CA 90  3360            LDX SCRB      Get DRTAB index
924E- 60        3370            RTS
                3380
                3390            ; SET TIME
                3400
924F- 78        3410 TIMSET     SEI           Inhibit time changes
9250- A2 08     3420            LDX #$08      Set index regs
9252- A0 01     3430            LDY #$01
9254- B9 CD 92  3440 PMOR       LDA MSTAB,Y   Print <reg name>=
9257- F0 06     3450            BEQ NXTR
9259- 20 47 8A  3460            JSR OUTCHR
925C- C8        3470            INY
925D- D0 F5     3480            BNE PMOR      (Always)
                3490
925F- 20 D9 81  3500 NXTR       JSR INBYTE    Get value for reg
9262- 48        3510            PHA
9263- B0 2F     3520            BCS BAD       Only decimal wanted
9265- 29 0F     3530            AND #$0F      Test low nibble
9267- C9 0A     3540            CMP #$0A
9269- B0 29     3550            BCS BAD
926B- 68        3560            PLA           Test high at STOT
926C- E0 06     3570            CPX #$06
926E- F0 04     3580            BEQ ADJUST
9270- E0 04     3590            CPX #$04
9272- D0 06     3600            BNE STOM
9274- F8        3610 ADJUST     SED
9275- 38        3620            SEC
9276- E9 01     3630            SBC #$01 Adjust to JAN=00,SUN=00
9278- 90 1A     3640            BCC BAD
927A- E0 05     3650 STOM       CPX #$05
927C- D0 06     3660            BNE STOT
```

```
927E- C9 32     3670           CMP #$32     Allow day of month = 31
9280- B0 12     3680           BCS BAD
9282- 90 05     3690           BCC STO
9284- DD A8 90  3700 STOT      CMP TABL,X   Check range
9287- B0 0B     3710           BCS BAD
9289- 9D BD 90  3720 STO       STA TIMR,X
928C- CA        3730           DEX
928D- E0 07     3740           CPX #$07     Year needs 4 digits
928F- F0 03     3750           BEQ BAD      Skip CRLF if year
9291- 20 4D 83  3760           JSR CRLF
9294- BD 70 93  3770 BAD       LDA TSTAB,X
9297- A8        3780           TAY
9298- D0 BA     3790           BNE PMOR
                3800
929A- A0 00     3810 PRTS      LDY #$00     Prepare to start clock
929C- 20 C1 92  3820           JSR SETMES   Print "AT EXACTLY "
929F- A9 50     3830           LDA #'P
92A1- 20 27 91  3840           JSR PENT     Print the TIME
92A4- A0 0C     3850           LDY #$0C
92A6- 20 C1 92  3860           JSR SETMES   Print " TYPE RETURN"
92A9- 20 D9 81  3870           JSR INBYTE   Get it
92AC- 20 1B 90  3880           JSR START    Start clock
92AF- A9 00     3890           LDA #$00
92B1- 8D BD 90  3900           STA TIMR     Clear fractions
92B4- 20 4D 83  3910           JSR CRLF
92B7- 20 4D 83  3920           JSR CRLF     Make room to print
92BA- 20 1A 92  3930           JSR TIMOUT   Print the time again
92BD- 20 4D 83  3940           JSR CRLF
92C0- 60        3950           RTS
                3960
92C1- B9 57 93  3970 SETMES    LDA SETAB,Y  Print time-set message
92C4- F0 06     3980           BEQ SDONE
92C6- 20 47 8A  3990           JSR OUTCHR
92C9- C8        4000           INY
92CA- D0 F5     4010           BNE SETMES   (Always)
92CC- 60        4020 SDONE     RTS
                4030
92CD- 20 34 20  4040 MSTAB     .BY ' 4 DIGIT YEAR=' $00
92D0- 44 49 47
92D3- 49 54 20
92D6- 59 45 41
92D9- 52 3D 00
92DC- 32 20 44  4050           .BY '2 DIGIT MONTH (01-12)=' $00
92DF- 49 47 49
92E2- 54 20 4D
92E5- 4F 4E 54
92E8- 48 20 28
92EB- 30 31 2D
92EE- 31 32 29
92F1- 3D 00
92F3- 32 20 44  4060           .BY '2 DIGIT DAY (01-31)=' $00
92F6- 49 47 49
92F9- 54 20 44
92FC- 41 59 20
92FF- 28 30 31
9302- 2D 33 31
9305- 29 3D 00
9308- 32 20 44  4070           .BY '2 DIGIT WEEKDAY (SUN=01,SAT=07)=' $00
930B- 49 47 49
930E- 54 20 57
9311- 45 45 4B
9314- 44 41 59
9317- 20 28 53
931A- 55 4E 3D
931D- 30 31 2C
```



```
9320- 53 41 54
9323- 3D 30 37
9326- 29 3D 00
9329- 32 20 44  4080           .BY '2 DIGIT HOUR=' $00
932C- 49 47 49
932F- 54 20 48
9332- 4F 55 52
9335- 3D 00
9337- 32 20 44  4090           .BY '2 DIGIT MINUTE=' $00
933A- 49 47 49
933D- 54 20 4D
9340- 49 4E 55
9343- 54 45 3D
9346- 00
9347- 32 20 44  4100           .BY '2 DIGIT SECOND=' $00
934A- 49 47 49
934D- 54 20 53
9350- 45 43 4F
9353- 4E 44 3D
9356- 00
9357- 41 54 20  4110 SETAB     .BY 'AT EXACTLY ' $00
935A- 45 58 41
935D- 43 54 4C
9360- 59 20 00
9363- 20 54 59  4120           .BY ' TYPE RETURN' $00
9366- 50 45 20
9369- 52 45 54
936C- 55 52 4E
936F- 00
9370- 00 7A 6A  4130 TSTAB     .BY $00 $7A $6A $5C $3B $26 $0F $0E $01
9373- 5C 3B 26
9376- 0F 0E 01
9379- C9 CB C9  4140 DRTAB     .BY $C9 $CB $C9 $03 $CA $02 $CA $01
937C- 03 CA 02
937F- CA 01
9381- 84 C9 05  4150           .BY $84 $C9 $05 $86 $08 $07 $00
9384- 86 08 07
9387- 00
                4160
                4170           .EN
```

EDITOR'S NOTE:

WE PUBLISH THIS OVER DICK ALBER'S PROTESTS THAT IT IS NOT YET READY FOR PUBLICATION! HE DID APPROVE PUBLICATION IF WE CLEARLY MARKED IT AS A "PRELIMINARY DRAFT", AND DID POINT OUT THE THREE PROBLEM AREAS, FOR ALL OF WHICH HE HAS SOLUTIONS. HE HAS PROVIDED THESE SOLUTIONS, WHICH, AFTER DEBUGGING AND "POLISHING", WE SHALL PUBLISH IN THE NEXT ISSUE OF SYMPHYSIS.

THE FIRST TWO PROBLEM AREAS, AS MENTIONED IN THE INTRODUCTORY TEXT, REQUIRE POINTING INVEC AND OUTVEC TO SUBROUTINES WHICH INHIBIT INTERRUPTS DURING THE ACTUAL I/O OF THE SERIAL BITS.

THE THIRD PROBLEM AREA IS DUE TO THE OFTEN OVERLOOKED FACT THAT THE BREAK INSTRUCTION SETS THE I FLAG AS WELL AS THE B FLAG! THUS, ANY EXITS TO MON FROM RAE (DONE VIA THE BRK INSTRUCTION) STOP THE CLOCK. THE SOLUTION TO THIS IS TO POINT UBRKVEC TO A ROUTINE WHICH INCLUDES THE NECESSARY CLI INSTRUCTION.

A BASIC TIMER

Now you will see why we moved our printer to free up the B port for more valuable uses. The following BASIC program, by Joe Hobart. provides a very informative demonstration of the use of the Timer/Counter in the VIA. We have modified Joe's original program to use Jack Gieryic's BASIC 'GET' Function, which should be of much interest to BASIC users.

AN INTERVAL TIMER FOR SYM BASIC

BY JOE HOBART, 3465 ANDES DRIVE, FLAGSTAFF, AZ 86001

PROGRAMS LIKE "OREGON TRAIL" USE A TIMER TO MEASURE THE INTERVAL BETWEEN THESE PROGRAMS' STIMULUS AND A PERSON'S RESPONSE TO THAT STIMULUS. THE 6522 LOCATED AT U-28 PROVIDES A SIMPLE WAY TO IMPLEMENT SUCH A TIMER. THE FOLLOWING PROGRAM ILLUSTRATES THE USE OF THE 6522 AS A VERY ACCURATE TIMER THAT CAN BE USED IN A BASIC PROGRAM TO GIVE INTERVAL MEASUREMENTS FROM 0 TO 600.00 SECONDS. THE ADDRESSES SPECIFIED REQUIRE THAT U-28 HAS BEEN INSTALLED AND THAT PB7 (AA-6) BE CONNECTED TO PB6 (AA-H). I USED AN EDGE CONNECTOR WITH PIN 6 WIRED TO PIN H SO THAT IT COULD BE EASILY REMOVED TO USE OTHER ACCESSORIES AT THE AA PORT.

THE PROGRAM CAN BE MODIFIED TO GIVE OTHER RANGES BY CHANGING THE NUMBER STORED IN T1 REGISTERS. 5000 GIVES A .01 SECOND WAVEFORM AT PB7. 50000 WILL GIVE A 0.1 SECOND WAVEFORM AND A TIMER THAT WILL COUNT BY TENTHS TO SIX THOUSAND SECONDS. THE DIVISOR IN LINE 320 WILL HAVE TO BE CHANGED TO 10 TO GIVE THE CORRECT TIME, HOWEVER. NOTE THAT READING THE T2 REGISTERS DOES NOT STOP THE COUNT DOWN. THE ONLY WAY TO RESET THE TIME IS TO RELOAD THE T2 REGISTERS BY POKE STATEMENTS. ONCE STARTED, THE TIMER T1 WILL PROVIDE A STEADY OUTPUT WITHOUT FURTHER ATTENTION. T2 WILL THEN DECREMENT ONCE FOR EACH PULSE FROM T1. IF LEFT UNATTENDED, T2 WILL PASS ZERO AND CONTINUE COUNTING DOWN FROM 65536.

ONCE ALL THE REMARK STATEMENTS HAVE BEEN REMOVED, THE PROGRAM IS QUITE SHORT. T1 ONLY HAS TO BE INITIALIZED ONCE. T2 CAN BE LOADED OR READ AS DESIRED DURING THE PROGRAM. THE TIMER IS COMPLETELY INDEPENDENT OF ANY OTHER PROCESSING ON THE SYM.

(Added notes by Lux - We have nade up a number of patch cords with micro-clips on both ends, and used such a cord to jumper AA-6 to AA-H on the 44 pin connector wired to the Epson. We used the square wave generated at PB-7 to check out the time base calibration of our scope. We have ordered a really nice frequency meter made by Albia, and we look forward to its arrival so that we can use this program to generate test signals for it.)

```
1 REM Program by Joe Hobart
2 REM Modified by Lux to include
3 REM "A Simple BASIC GET Function"
4 REM by Jack Gieryic
5 REM
100 REM
110 REM  CONNECT SYM AA-6 TO AA-H.  U-28 MUST BE INSTALLED.
120 POKE 42579,128
130 REM  INITIALIZE U-28 T1 AND INHIBIT IRQ
140 REM
150 POKE 43022,0 : REM INHIBIT IRQ ($A80E)
160 POKE 43019,224 : REM SET ACR ($A80B)
170 REM
180 REM  LOAD 5000 INTO T1 TO GIVE A .01 SECOND SQUARE WAVE AT PB7
190 REM
200 POKE 43012,136 : REM PUT 136 IN T1L-L ($A804)
210 POKE 43013,19 : REM PUT 19 IN T1L-H ($A805)
220 REM
230 REM  LOAD 60000 INTO T2 TO COUNT DOWN FROM 600.00 SECONDS
240 REM
250 PRINT"Hit any key to start timer, then any key to stop it."
251 PRINT"The time interval between keystrokes will be printed."
252 PRINT"Exit with ESC key."
253 PRINT
254 GOSUB1000
260 POKE 43016,96 : REM PUT 96 IN T2C-L ($A808)
270 POKE 43017,234 : REM PUT 234 IN T2C-H ($A809)
280 REM
290 REM  READ T2 COUNTERS AND CALCULATE TIME INTERVAL
```



```
300 PRINT"The timer is running.....":PRINT
310 GOSUB1000
320 T=INT(60000-PEEK(43016)-256*PEEK(43017))/100
330 PRINT:PRINT"Elapsed time was "T"seconds."
331 PRINT
340 GOTO100
1000 POKE42579,0:GOSUB1001:POKE42579,128:RETURN
1001 Q=USR(-30120,-11957,0):CH=128-(Q/(-256))
1002 IF CH=27 THEN POKE 42579,128:END
1003 RETURN
1004 REM Lines 1000 - 1003 are based on a very ingenious BASIC "GET"
1005 REM program submitted by Jack Gieryic many months ago but not yet
1006 REM published due to lack of space. We present it here for your use
1007 REM without explanation of how it works, except to say that line
1008 REM 1000 controls (i.e., suppresses echo of the "GOTTEN" character
1009 REM and may be omitted if desired) TECHO.
OK

Hit any key to start timer, then any key to stop it.
The time interval between keystrokes will be printed.
Exit with ESC key.

The timer is running.....


Elapsed time was  2.71 seconds.

Hit any key to start timer, then any key to stop it.
The time interval between keystrokes will be printed.
Exit with ESC key.
OK
```

ON PRINTER RIBBONS

One very minor problem we have found with the Epson is a relatively short ribbon life. By this we do not mean that the ribbon becomes unuseable, but that the impression seems to "weaken" very rapidly. The ribbon is not rolled tightly on a spool where it has a chance to "re-ink" itself from contact with adjacent layers. Instead it is literally pushed into one end of an enclosure and pulled out at the other end.

For most applications the gradual lightening of the impression would present no real problem; just discard the ribbon whenever the printing appears too light for your taste. This does present a serious problem in preparing camera-ready copy for publication, however, in that material printed several days apart cannot easily be matched in appearance. Pasting in a corrected word or phrase in the middle of a previously printed paragraph looks bad if the new material appears much lighter or darker. We find that we must reprint the material again, with the corrections made in the computer.

We tried to "revive" a ribbon by giving it a half-twist, and rewinding by hand to use the other edge. This made no difference. A friend passed on to us the following tip he had heard from an Epson user: Pry open the top cover of the ribbon cartridge and saturate the ribbon with WD-40 spray lubricant (a non-silicone containing product made in San Diego, CA; don't know if it is available on a national basis). Give the material an hour or two to penetrate and/or evaporate, wipe off the cartridge, and replace it in the printer. The ribbon will then have a new lease on life.

Our friend did not know how many re-leases were possible. We tried it, mostly out of curiosity, and it does seem to work. With new ribbons selling at around $15.00, this could be a real money saver! We now feel challenged to see just how many WD-40 recycles a ribbon can accept in its lifetime!

"STRUCTURED" ASSEMBLY PROGRAMMING IN RAE!

Norbert Thuering, address below, sent us a review copy of his MAC65 (why do we keep thinking of it as "Big MAC"?) enhancements to RAE-1. He used RAE'S built-in conditional and macro capabilities to incorporate a very powerful set of new pseudo opcodes into its structure, using the prefix "@" as the designator. These new pseudo-ops provide the experienced RAE user with a fully structured programming tool.

The easiest way to explain both MAC65 and the entire concept of "structured" programming is to reproduce below the simple, but elegant, two page instruction manual, which includes an informative comparision-by-example with Pascal, and portions of a sample run in which the "proprietary" macro definitions were "blanked" with the .LC (List Clear) pseudo opcode. We will be distributing MAC65 for Mr. Thuering (details elsewhere).

USE OF MAC65

When you wish to use MAC65, enter RAE at its cold start ($B000) and modify the default set parameters to meet your needs. Next get MAC65 from mass storage, and then begin entering your program at the end of MAC65 (just after .LS). Then ASsemble your program and check @SP and @NEST in the label file to be 0000 (no nesting errors). Now the program is ready for use.

Syntax of MAC65:

Here is a short macro-syntax description:
- all macro key-words begin with '@'

```
    [     ]  optional, may be omitted
...[     ]  block may be repeated
   <     >  syntactic constructs

<block> ::=  macros and mnemonics
<cond>  ::=  LE HI LO LS GE GT LT EQ NE HS
<expr>  ::=  RAE-expression, variable, constant


    @IF    ( <expr> <cond> <expr> )
...[@AND   ( <expr> <cond> <expr> )]
...[@OR    ( <expr> <cond> <expr> )]
    @THEN
     <block>
    [@ELSE  ]
    [<block>]
     @ENDIF


    @WHILE ( <expr> <cond> <expr> )
...[@AND   ( <expr> <cond> <expr> )]
...[@OR    ( <expr> <cond> <expr> )]
    @DO
    <block>
    @ENDWHI


    @FOR   ( <var> <start> <end> <step> )
    <block>
    @NEXT


    @CASE  ( <var> )
    @OF    ( <expr> )
...[@..    ( <expr> )
    <block>
...[@OF    ( <var> )]
...[@..    ( <var> )]
...[<block>        ]
    @ELSE
    [<block>]
     @ENDCAS


    @WRITE ( <label> )      Text must be declared as follows:
                            TEXT1  .BY 'This is a text' Ø .
```

--> IMPORTANT! Variables used in the program have to be declared before use, otherwise the macros of MAC65 won't work correctly (refer to example of HI-LO game).

The macros are intended for use with variables and constants restricted to the range of Ø, . . . ,255, or, $ØØ, . . . ,$FF.

Do not use variables located at page zero.

EXAMPLE: HI-LO GAME

The following example is divided in two parts: on the right is a PASCAL-like notation of the example program, on the left the equivalent MAC65 notation.

```
TIMER  .DE $A008
INBYT  .DE $81D9
       .BA $0200
       .OS
TRY    .DS 1                  -  BYTE: TRY,NUMBER,GUESS;
NUMBER .DS 1
GUESS  .DS 1
TEXT1  .BY 13 10 'Your guess? ' Ø
TEXT2  .BY 13 10 'Correct !' Ø
TEXT3  .BY 13 10 'Too high . . .' Ø
TEXT4  .BY 13 10 'Too low.  . . .' Ø

ENTRY

LDA #Ø                        -  TRY:=Ø;
STA TRY
LDA TIMER                     -  NUMBER:=RND(Ø);
STA NUMBER
@WHILE (TRY EQ Ø)             -  WHILE TRY=Ø DO
@DO
@WRITE (TEXT1)                -    WRITE("Your guess? ");
JSR INBYT                     -    READ(GUESS);
STA GUESS
@IF (NUMBER EQ GUESS)         -    IF NUMBER=GUESS THEN
@THEN
@WRITE (TEXT2)                -      WRITE("Correct !");
LDA #1                        -      TRY:=1;
STA TRY
@ELSE                         -    ELSE
@IF (NUMBER LO GUESS)         -      IF NUMBER < GUESS  THEN
@THEN
@WRITE (TEXT3)                -        WRITE("Too high . . .");
@ELSE                         -      ELSE
@WRITE (TEXT4)                -        WRITE("Too low.  . . .");
@ENDIF                        -      END;
@ENDIF                        -    END;
@ENDWHI                       -  END.
RTS
.EN
```

```
>ASSEMBLE LIST
                0010                .LS
                0020 ;*******************************
                0030 ;*                             *
                0040 ;* MAC65 V1.Ø  DATE 23.06.81 *
                0050 ;*                             *
                0060 ;*******************************
                0070 ;*                             *
                0080 ;* Norbert C. Thuering         *
                0090 ;* Rainstrasse 15              *
                0100 ;* 8103 Unterengstringen       *
                0110 ;* SWITZERLAND                 *
                0120 ;*                             *
                0130 ;*******************************
                0140
                0420                .LS
                0430
                0440 ; CONDITION CODES FOR MACROS
                0450
                0460 EQ             .DE Ø equal
                0470 NE             .DE 1 not equal
                0480 HS             .DE 2 higher or same (binary)
                0490 HI             .DE 3 higher (binary)
                0500 LO             .DE 4 lower (binary)
                0510 LS             .DE 5 lower or same (binary)
                0520 LT             .DE 6 less than (2's complement)
                0530 LE             .DE 7 less or equal (2's complement)
                0540 GT             .DE 8 greater than (2's complement)
                0550 GE             .DE 9 greater or equal (2's complement)
                0560 IN             .DE Ø
                0570
                4280                .LS
                4290
                4300 ; SAMPLE PROGRAM BEGINS HERE
                4310
                4320 TIMER          .DE $A008
                4330 INBYT          .DE $81D9
                4340                .BA $0200
```

```
                        4350              .OS
                        4360
0200-                   4370 TRY          .DS 1
0201-                   4380 NUMBER       .DS 1
0202-                   4390 GUESS        .DS 1
0203- 0D 0A 47          4400 TEXT1        .BY 13 10 'Guess? ' 0
0206- 75 65 73
0209- 73 3F 20
020C- 00
020D- 0D 0A 43          4410 TEXT2        .BY 13 10 'Correct!' 0
0210- 6F 72 72
0213- 65 63 74
0216- 21 00
0218- 0D 0A 54          4420 TEXT3        .BY 13 10 'Too high . . .' 0
021B- 6F 6F 20
021E- 68 69 67
0221- 68 20 2E
0224- 20 2E 20
0227- 2E 00
0229- 0D 0A 54          4430 TEXT4        .BY 13 10 'Too low. . . .' 0
022C- 6F 6F 20
022F- 6C 6F 77
0232- 2E 20 2E            NOTE: The hex addresses and dashes "sprinkled" about
0235- 20 2E 20                  in the source code below were NOT entered by
0238- 2E 00                     the programmer!  They are artifacts of RAE-1
                                whenever .EC (macro Expand Clear, default set-
                        4440    ting) is used. .ES (macro Expand Set) wastes
                        4450    much, too much, paper to use very often!
023A- A9 00             4460 ENTRY        LDA #0
023C- 8D 00 02          4470              STA TRY
023F- AD 08 A0          4480              LDA TIMER
0242- 8D 01 02          4490              STA NUMBER
                        4500              @WHILE (TRY EQ 0)
                        4510              @DO024C-
                        4520              @WRITE (TEXT1)
025D- 20 D9 81          4530              JSR INBYT
0260- 8D 02 02          4540              STA GUESS
                        4550              @IF (NUMBER EQ GUESS)
                        4560              @THEN026B-
                        4570              @WRITE (TEXT2)
027C- A9 01             4580              LDA #1
027E- 8D 00 02          4590              STA TRY
                        4600              @ELSE0281-0281-
                        4610              @IF (NUMBER LO GUESS)
                        4620              @THEN028E-
                        4630              @WRITE (TEXT3)
                        4640              @ELSE029F-029F-
                        4650              @WRITE (TEXT4)
                        4660              @ENDIF
                        4670              @ENDIF
                        4680              @ENDWHI
02B3- 60                4690 END          RTS
                        4700
//0000,02B4,02B4
```

```
.
0200 01 BE BE 0D 0A 47 75 65,B5
0208 73 73 3F 20 00 0D 0A 43,54
0210 6F 72 72 65 63 74 21 00,04
0218 0D 0A 54 6F 6F 20 68 69,3E
0220 67 68 20 2E 20 2E 20 2E,F7
0228 00 0D 0A 54 6F 6F 20 6C,CC
0230 6F 77 2E 20 2E 20 2E 20,9C
0238 2E 00 A9 00 8D 00 02 AD,AF
0240 08 A0 8D 01 02 AD 00 02,96
0248 C9 00 F0 03 4C B3 02 A2,F5
0250 00 BD 03 02 F0 07 20 47,15
0258 8A E8 4C 51 02 20 D9 81,A0
0260 8D 02 02 AD 01 02 CD 02,B0
0268 02 F0 03 4C 84 02 A2 00,19
0270 BD 0D 02 F0 07 20 47 8A,CD
0278 E8 4C 70 02 A9 01 8D 00,AA
0280 02 4C B0 02 AD 01 02 CD,27
0288 02 02 F0 02 90 03 4C A2,9E
0290 02 A2 00 BD 18 02 F0 07,10
0298 20 47 8A E8 4C 93 02 4C,16
02A0 B0 02 A2 00 BD 29 02 F0,42
02A8 07 20 47 8A E8 4C A4 02,14
02B0 4C 45 02 60,07
 3707
```

NOTES:
1) The hex code may be entered from MON, but the game MUST be played from RAE by calling RUN $023A. The clocks, all of them, behave "strangely" when called from MON as this program uses them!
2) The random number is in HEX, so enter two hex digits.

RAM-BLINGS

Being fond of puns, we changed the name of this closing section of the newsletter from MISCELLANEA to RAM-BLINGS, since we randomly access our memory for items to fill the remaining space.

You will notice that parts of this issue are printed 8 lines to the inch, instead of the usual six. If there are no complaints, we will change over to the eight lines per inch, permitting 33 1/3 % more material in the same space. We'll stay with the pica size (10 pitch, or CPI), and 70% photoreduction and avoid the 16.5 CPI of which the Epson is capable.

Jack Brown has asked us to announce, in a preliminary way, that he is nearly ready to formally announce the existence of his SYM-Pascal. Both cassette-based and disk-based (FODS) versions will be available. Detailed information on availability, specifications, and prices will appear in issue number three of his newsletter, *Saturn Softnews*.

We have seen the first two issues, which provide very strong support to users of his FORTH and BASIC enhancements. Those interested are urged to contact Jack, directly, for a subscription. Address: P. O. Box 397, New Westminster, B. C., V3L 4Y7, Canada. Price $10.00 (US) or $12.00 (Canadian) plus $4.00 (US) for overseas airmail.

It wasn't the teaching, traveling/lecturing, hardware and software developing, etc., that delayed this issue. It was a very, very, slowly moving kidney stone that lowered our energy level by over 50% for some six weeks, and reduced our work output. Now that we are no longer "stoned", you can expect Issue #10 to arrive within six weeks after this one, around the first of the year.

We had hoped to announce, with this issue, prices and availability for a number of new software products, but these will have to wait until Issue #10, which will include a completely updated Shopping List. The new packages include Kwok's Cross Reference Lister, Kwoks's BASIC Word Processor, Thuering's MAC65, Holt's TECO Word Processor, a Music Package for limited RAM systems, and several others.

We are OEMing customized SYM-based systems, and have been purchasing in quantity, to get good price breaks, such items as Epsons, Cassette Recorders, Power Supplies, Disk Drives, etc. Contact us about special prices for any peripherals for your SYM!

Jack Gieryic has developed the required CODOS/SYM-BASIC link, based on Jack Brown's Extended SYM-BASIC. The pair of Jacks are working out marketing and distribution agreements. This was the "missing link" as far as the Users' Group being able to provide software support to SYM/CODOS systems.

We now have two, soon three, SYM/FODS Dual-5" systems, and one SYM/CODOS Single-8" system. It is not easy to prepare software for distribution on a single drive disk system!!!!! By year's end we will have upgraded our SYM/CODOS system to Dual-8", and installed a SYM/FODS Dual-8" as well. We will attempt to make SYM software available on any medium used by a significant number of users. By next summer, if all goes according to schedule, your modem can talk to ours!

We will be on sabbatical leave from California State University, Chico, from January through August, 1982. We plan to spend much of this time "researching" and writing, but will be making two 4-6 week trips, one to Australia, New Zealand, and the Orient, the second to Europe and the Middle East. The southwest Pacific trip will be during April/May, the European trip has not yet been scheduled. If you are associated with an overseas university or educational institution at any level, please write. We would like to visit, and perhaps arrange a guest lecture or seminar session, at no charge, except for local expenses.

Steve Perry, of Perry Peripherals, P.O. Box 924, Miller Place, NY 11764, asked us to mention that he is now set up to provide a quick turn-around time on SYM-1 repairs.

As usual, we had more material, and really quality material, submitted then could be published. Even if we had more space, there would not have been enough time to go over the material, anyway. We will not publish any software, or technical articles, unless we have thoroughly verified the validity (or is it validated the verity?). We are quite proud that there have been so few typos, and only one program bug (and we pointed out the existence of that bug when we published the program). Much of the submitted but as yet unpublished material will appear in a special volume next summer; so keep the material coming!

We thank those of our readers who are helping us to review submitted material, especially Dick Albers, who has also provided us with a 6809 Tiny BASIC for the SYM-69. His version was based on an article in a recent issue of Dr. Dobbs' Journal.

Issue #10 will feature BASIC (as this issue featured RAE). We will also review, based on hands-on testing, the SYM-2 and the recently announced Synertek Disk Controller Card.

Happy SYMmering . . . . . Jean-Luc