SYM-1 USERS' GROUP
P. O. BOX 315
CHICO, CA 95927
(916) 895-8751

RAE NOTES - ISSUE NO. 3

PAGES ZERO AND ONE MEMORY MAP

At long last, we bring you the explicit pages zero and one memory assignments for RAE-1, printed below with Carl Moser's permission:

```
>PRINT
 0010  ;            RAE-1 PAGES ZERO AND ONE ASSIGNMENTS
 0020  ;
 0030  ;EDITED AND REPRINTED, WITH PERMISSION, BY SYM-1 USERS GROUP
 0040  ;
 0050  ;        ***** COPYRIGHT 1978 BY CARL MOSER *****
 0060  ;
 0070  ;                    ALL RIGHTS RESERVED
 0080  ;
 0090  ;
 0100                .BA $B000
 0110  TECHO         .DE $A653    ;SYM ECHO CONTROL
 0120  ;
 0130  ;       THE FOLLOWING MUST BE CONTIGUOUS
 0140  ;
 0150  PRINT.VEC     .DE $B6      ;THREE BYTES - PRINT CALL
 0160  SAVEM         .DE $B9      ;SAVE FOR MSYT (2 BYTES)
 0170  ASSM.LNKNO    .DE $BA      ;FILE # LINK VIA >PR, .CT
 0180  MAC.SUPPRS    .DE $BB      ;=1 WHEN INPUTTING MACRO BODY
 0190  MAC.EXPAND    .DE $BC      ;=1 WHEN EXPANDING MACRO
 0200  MAC.SRCH      .DE $BD      ;=1 THEN INFORMS GSYT TO SEARCH MACRO
 0210  CALL.EXP      .DE $BE      ;=1 THEN CURRENT CALL FOR EXPANSION
 0220  ;
 0230  COND.SUP      .DE $BF      ;=1 THEN SUPPRESS ASSEMBLE
 0240  SYMBOLIC      .DE $C0      ;=1 THEN SYMBOLIC LABEL
 0250  LST.EXP.OB    .DE $C1      ;=1 THEN PRINT EXP. OBJ.
 0260  MAC.SEQ       .DE $C2      ;(2 BYTES) SEQ# FOR ...LABEL
 0270  FILE,SEQ      .DE $C4      ;(2 BYTES) SEQ# FOR !!!LABEL
 0280  ;
 0290  SAVEXX        .DE $C6      ;TEMPORARY
 0300  SAVEYY        .DE $C7      ;TEMPORARY
 0310  ;
 0320  PURECL        .DE $C8      ;RELOCATING BUFFER ADDRESS LO
 0330  PURECH        .DE $C9      ;RELOCATING BUFFER ADDRESS HI
 0340  DELIM         .DE $CD      ;>ED COMMAND STRING DELIMITER
 0350  SAVE          .DE $CE      ;TEMPORARY STORAGE
 0360  DC            .DE $CF      ;>ED COMMAND DON'T CARE CHARACTER
 0370  PNTR          .DE $D0
 0380  PROC_ADDRS    .DE $D1      ;CONTAINS COMPOSITE ADDRS FROM LABEL
 0390  TPRES         .DE $D3      ;PRESSENT END OF TEXT FILE
 0400  SPRES         .DE $D5      ;PRESENT END OF LABEL FILE
 0410  PC            .DE $D7      ;PROGRAM COUNTER
 0420  REL           .DE $D9      ;RELOCATE CODE
 0430  ERRORS        .DE $DB      ;# OF ERRORS
 0440  SCRATO        .DE $DD
```

```
0450  SCRAT1       .DE $DF
0460  SCRAT2       .DE $E0
0470  SCRAT3       .DE $E1
0480  SCRAT4       .DE $E2
0490  SUPPRESS.O   .DE $E3      ;=1 THEN SUPPRESS OUTPUT
0500  SAVEY        .DE $E4
0510  SAVEX        .DE $E5
0520  ;
0530  LN.2B.OUT    .DE $E6      ;=1 THEN REC. TO BE OUT.
0540  LN.INDEX     .DE $E7      ;POSITION OF LINE ON SCREEN
0550  LIB.YM       .DE $E       ;=1 THEN CURRENT SYMBL=LIBRARY
0560  LINE.LEN     .DE $E9      ;LEN. OF SOURCE PART OF OUTPUT
0570  AUTO.TAB     .DE $EA      ;TAB FOR FORMAT SET
0580  ;  (NOTE: $EB IS NOT USED BY RAE-1)
0590  ;
0600  ;THE FOLLOWING ARE DISC VARIABLES
0610  ;
0620  DISCC.VEC    .DE $EC      ;DISC COMMAND VECTOR
0630  DISCI        .DE $EE      ;=1 THEN INPUTFROM DISC ELSE TAPE
0640  DISCO        .DE $EF      ;=1 THEN UTPUT TO DISC ELSE TAPE
0650  DISC1        .DE $F0      ;DISC OUT SETUP VECTOR
0660  DISC2        .DE $F2      ;DISC IN SETUP VECTOR
0670  DISCO.VEC    .DE $F4      ;DISC OUT DATA VECTOR
0680  DISCI.VEC    .DE $F6      ;DISC IN DATA VECTOR
0690  ;
0700  CRT.END      .DE $FF50    ;80 BYTE CRT BUFFER
0710  ;
0720  ;THE FOLLOWING MUST BE CONTIGUOUS
0730  ;
0740  NONO_BLK     .DE $100     ;BEGINNING OF ABSOLUTE BLOCK
0750  TXST         .DE $100     ;START OF TEXT FILE
0760  TXEN         .DE $102     ;END OF TEXT FILE
0770  STST         .DE $104     ;START OF LABEL FILE
0780  STEN         .DE $106     ;END OF TEXT FILE
0790  FIRST        .DE $108     ;FIRST LINE #
0800  LAST         .DE $10A     ;LAST LINE #
0810  INCBY        .DE $10C     ;INCREMENT FOR AUTO LINE #-ING
0820  MANU         .DE $10E     ;MANUSCRIPT OPTION SWITCH
0830  FORMAT       .DE $10F     ;FORMAT OPTION SWITCH
0840  FILE.NO      .DE $110     ;CONTAINS CURRENT FILE NUMBER
0850  HEX/DEC      .DE $111     ;0=HEX;  1=DEC
0860  TERM         .DE $112     ;TERMINATE ON ERRORS
0870  PASS         .DE $113     ;INDICATES WHICH ASSEMBLY PASS
0880  CON_TAPE     .DE $114     ;CONTINUE ON TAPE
0890  AU           .DE $115     ;AUTO LINE #-ING SWITCH
0900  OSTORE       .DE $116     ;STORE OBJECT CODE
0910  TLIST        .DE $117     ;ASSEMBLY LIST OPTION
0920  ADDRS        .DE $118
0930  ADDPAD       .DE $11A
0940  CUR.SAV      .DE $11C     ;USED TO LOCATE LAST LINE
0950  EXT          .DE $11E     ;INDICATES INTERNAL OR EXTERNAL
0960  CKG_SUM      .DE ERRORS
0970  PRINT/CTL    .DE $11F     ;PRINTER CONTROL SWITCH
0980  CRTEX        .DE $18F     ;CRT EXTENSION BUFFER
0990  LINE_CNTR    .DE $120     ;>HA COMMAND LINE COUNTER
1000  PAGE_NUMB    .DE $121     ;>HA COMMAND PAGE # (2 BYTES)
1010  REC_POINT    .DE $122     ;POINTER IN RELOCATING BUFFER RECORD
1020  LOAD/NO      .DE $123     ;INDICATES IF TO LOAD IN MEMORY
1030  TSTART       .DE $124     ;TAPE STARTING ADDRESS
```

```
1040  TEND        .DE $126     ;TAPE ENDING ADDRESS
1050  HFILE/NO    .DE $128     ;HEADER FILE NUMBER
1060  HSTART      .DE $129     ;HEADER START INFO.
1070  HEND        .DE $12B     ;HEADER END INFO.
1080  DELAY1      .DE $12D     ;DELAY USED FOR TAPE MOTOR
1090  DELAY2      .DE $12E     ;DELAY USED FOR TAPE MOTOR
1100  AS_LN_SV    .DE $12F     ;ASSM LINE # SAVE
1110  EDIT/FINDC  .DE $131     ;0 THEN >ED; ELSE >FI
1120  SWEET16     .DE $132     ;0 THEN 6502; ELSE SWEET 16
1130  TABSN       .DE $133     ;=0 THEN FUNCTIONAL TABS; ELSE ^I
1140  CRT         .DE $135     ;CRT BUFFER
```

The "SWEET 16" referred to in line 1120 is a machine language program available for the Apple II, which permits the Apple to emulate a sixteen bit, register-oriented, machine (in some ways, very similar to the RCA 1802). See BYTE, November 1977, for a full listing of the Sweet 16 Interpreter. Sweet 16 programs may be written in a symbolic form, with (pseudo) opcodes and labels, and then machine assembled into hexadecimal format for the interpreter. Note that the Sweet 16 assembler is NOT included in RAE-1.

HYPER.SORT FOR RAE-1

Here is a very beautiful program submitted by J. Cyr. His letter, also printed below, explains ALMOST everything. We'll explain the one missing point! Notice the '.ct' before the '.en'? That causes the assembly to halt and inform you that it is ready for the second pass. At this time you can call HYPER.SORT by RUN $address, where $address is the hex value where you have (previously) stored the object code for HYPER.SORT . Then, at the end of PASS 2, the label file will be printed in a neat, alphabetically sorted, form. We have HYPER.SORT on disk, and call it with DC 'name'. We named HYPER.SORT on our system disk in honor of its creator. That's why the DC CYR appears in the listing. The DC stands for Disc Command.

Note that, on PASS 2, the .CT (or .ct) is ignored, because it is not the last pseudo op in the file. If you are really continuing on tape, be sure to CLear, before reloading the first file and going on to PASS 2. If you don't, you may not be successful in completing your assembly.

Dear Lux:

For those, such as yourself, who have noticed that the RAE LABEL FILE SORT published in SYM-PHYSIS gets exponentially slower as the number of labels increases, I have included a copy of HYPER.SORT . You once waited 20 to 30 minutes for a sort of BROWN'S BASIC ENHANCEMENTS to end; HYPER.SORT will complete the same task in less than 1 minute. A table of timing results follows to support my claim:

#LABELS	SORT	HYPER.SORT
25	00.00.8	00.00.6
50	00.03.6	00.01.1
100	00.26.0	00.02.9
200	03.29.5	00.12.8
400	27.13.0	00.48.0

Randomly ordered label files were used. Times are in minutes.seconds.tenths .

I have included a cassette containing three copies of this letter in SWP format, followed by three copies of HYPER.SORT in RAE format. I hope you are able to publish this program, so that your readers who have not yet corrected the algorithmic deficiencies of my first sort can convert to this slightly bigger, but much more powerful, sort.

Sincerely,

J. Cyr
28 Greenboro Crescent
Ottawa, Ontario
Canada, K1T-1W5

```
>LOAD CYR2

>ASSEMBLE LIST
READY FOR PASS 2

>DC CYR

>PASS2
```

```
0005 ;=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=;
0010 ; HYPER.SORT                            ;
0015 ; MARCH 8, 1981                         ;
0020 ; BY: J.CYR, 28 GREENBORO CRESC.        ;
0025 ;      OTTAWA, ONTARIO                  ;
0030 ;      CANADA, K1T-1W5                  ;
0035 ; FOR: SYM-1 USERS' GROUP               ;
0040 ;=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=;
0045             .os
0050             .ba $5f00
0055 ;
0060 ;=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=;
0065 ; PAGE ZERO/ONE STORAGE                 ;
0070 ;=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=;
0075 ; various permanent and work pointers
0080 buffer.ptr .de $c8
0085 curnt.ptr  .de $fc
0090 nxt.ptr    .de $fe
0095 label.ptr  .de $104
0100 ;
0105 ;=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=;
0110 ; MACRO DEFINITION                      ;
0115 ;=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=;
0120 ; copy label
0125 !!!cl       .md (from to)
0130             ldy #0
0135             lda (from),y
0140             sta (to),y
0145             iny
0150             lda (from),y
0155             sta (to),y
0160 ...cl1      iny
0165             lda (from),y
0170             sta (to),y
0175             bpl ...cl1
0180             .me
```

```
                    0185 ;
                    0190 ;=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=;
                    0195 ; SORT LABELS                              ;
                    0200 ;=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=;
                    0205 ; start at beginning of label file,
                    0210 ;  and clear exchange flag.
5F00- A2 00         0215 hyper.sort ldx #0
5F02- AD 04 01      0220            lda label.ptr
5F05- 85 FE         0225            sta *nxt.ptr
5F07- AD 05 01      0230            lda label.ptr+1
5F0A- 85 FF         0235            sta *nxt.ptr+1
                    0240 ;
                    0245 ; check for empty label file,
                    0250 ;  and exit immediately if such.
5F0C- A0 02         0255            ldy #2
5F0E- B1 FE         0260            lda (nxt.ptr),y
5F10- F0 1B         0265            beq sort.exit
                    0270 ;
                    0275 ; make next label the current label.
5F12- A5 FE         0280 nxt.label  lda *nxt.ptr
5F14- 85 FC         0285            sta *curnt.ptr
5F16- A5 FF         0290            lda *nxt.ptr+1
5F18- 85 FD         0295            sta *curnt.ptr+1
                    0300 ;
                    0305 ; find the next label.
5F1A- A0 01         0310            ldy #1
5F1C- C8            0315 find.nxt   iny
5F1D- B1 FC         0320            lda (curnt.ptr),y
5F1F- 10 FB         0325            bpl find.nxt
                    0330 ;
                    0335 ; check for end of label file.
5F21- 20 95 5F      0340            jsr calc.nxt
5F24- A0 02         0345            ldy #2
5F26- B1 FE         0350            lda (nxt.ptr),y
5F28- D0 05         0355            bne compare+1
                    0360 ;
                    0365 ; check if we need another pass.
5F2A- 8A            0370            txa
5F2B- D0 D3         0375            bne hyper.sort
                    0380 ;
                    0385 ; return to RAE.
5F2D- 60            0390 sort.exit  rts
                    0395 ;
                    0400 ; compare current label with next label
                    0405 ;  and exchange if out of sequence.
5F2E- C8            0410 compare    iny
5F2F- B1 FC         0415            lda (curnt.ptr),y
5F31- 51 FE         0420            eor (nxt.ptr),y
5F33- 30 0A         0425            bmi end.label
5F35- B1 FE         0430            lda (nxt.ptr),y
5F37- D1 FC         0435            cmp (curnt.ptr),y
5F39- 90 1A         0440            bcc exchange
5F3B- D0 D5         0445            bne nxt.label
5F3D- F0 EF         0450            beq compare
                    0455 ;
                    0460 ; end of label reached.
5F3F- B1 FE         0465 end.label  lda (nxt.ptr),y
5F41- 10 0A         0470            bpl end.curnt
                    0475 ;
```

```
                 ; end of next label case.
5F43- 29 7F                     and #$7f
5F45- D1 FC                     cmp (curnt.ptr),y
5F47- F0 0C                     beq exchange
5F49- 90 0A                     bcc exchange
5F4B- B0 C5                     bcs nxt.label
                 ;
                 ; end of current label case.
5F4D- 09 80      end.curnt      ora #$80
5F4F- D1 FC                     cmp (curnt.ptr),y
5F51- F0 BF                     beq nxt.label
5F53- B0 BD                     bcs nxt.label
                 ;
                 ; exchange current label with next
                 ;  label.
                 exchange       cl (curnt.ptr buffer.ptr)
                                cl (nxt.ptr curnt.ptr)
5F79- 20 95 5F                  jsr calc.nxt
                                cl (buffer.ptr nxt.ptr)
                 ;
                 ; set exchange flag.
5F8E- E8                        inx
5F8F- D0 81                     bne nxt.label
5F91- E8                        inx
5F92- 4C 12 5F                  jmp nxt.label
                 ;
                 ; subroutine to calculate value of next
                 ;  pointer from value of current pointer
                 ;   and content of y.
5F95- 98         calc.nxt       tya
5F96- 38                        sec
5F97- 65 FC                     adc *curnt.ptr
5F99- 85 FE                     sta *nxt.ptr
5F9B- A5 FD                     lda *curnt.ptr+1
5F9D- 69 00                     adc #0
5F9F- 85 FF                     sta *nxt.ptr+1
5FA1- 60                        rts
                 ;
                 ;=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=;
                 ; THE END                               ;
                 ;=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=;
                                .ct
                                .en
```

```
LABEL FILE:  [ / = EXTERNAL ]


/buffer.ptr=00C8        /curnt.ptr=00FC         /label.ptr=0104
/nxt.ptr=00FE           calc.nxt=5F95           compare=5F2E
end.curnt=5F4D          end.label=5F3F          exchange=5F55
find.nxt=5F1C           from=00C8               hyper.sort=5F00
nxt.label=5F12          sort.exit=5F2D          to=00FE

//0000,5FA2,5FA2
```

See how the LABEL file has been alphabetized, external labels first!

ENHANCEMENTS FOR SWP-1

Here are two enhancements to SWP-1, created by Tom Gettys. The first, called by .N number, lets you select the page number. The second, called by .FILE number, lets you continue your manuscript on tape, with a specified file. We have not tested this ourselves, because our disk system has spoiled us; Tom assures us that it works properly. To keep the modifications "in context", we reprint portions of the original SWP-1 source code, so that you can see where and how the new commands are entered into the COMMAND TABLE.

The next enhancement we really need is a .TAB column!

```
                  0610 ;SUBROUTINE LINKAGE FOR RAE1.0
                  0620 ;----------------------------
                  0630 MNEXT       .DE $B4FF     ;MOVE TO NEXT FIELD
                  0640 MPSPACE     .DE $B502     ;MOVE PAST SPACES IN CRT
                  0650 SCRA0>TXST  .DE $B4BF     ;SCRATO TO TXST
                  0660 FORMATE     .DE $B3B2     ;FORMAT TEXT FILE TO BUFFER
                  0670 GET_NX_LN   .DE $B3A4     ;GET NEXT LINE IN TEXT FILE
                  0680 BREAK_RE    .DE $B05E     ;REENTER RAE AT WARM START
                  0690 WRT.        .DE $E3A4     ;WRITE ASCII TO CRT
                  0700 MESS_OUT    .DE $B51E     ;MESSAGE OUTPUT ROUTINE
                  0710 GSYT_PRO_S  .DE $E24A     ;GET FROM SYMBOL TABLE
                  0720 MTSPACE     .DE $B510     ;MOVE PAST SPACES
                  0730 GET_UP.C    .DE $B6A0     ;GET CHAR; MAKE UPPER CASE
                  0740 ERROR_02    .DE $B44B     ;OUTPUT !02 AT....
                  0750 ERROR_12    .DE $B43B     ;OUTPUT !12 AT......
                  0760 PG.INDX     .DE $41       ;INDEX FOR "PAGE" MESSAGE
```

```
                  3150 ;COMMAND TABLE (ALL ENTRIES S/B U.C.)
63C8- 43 00       3160 CMDTBL      .BY 'C' 0
63CA- F3 64       3170             .SI CMD.C-1
63CC- 50 00       3180             .BY 'P' 0
63CE- CF 65       3190             .SI CMD.P-1
63D0- 4C 00       3200             .BY 'L' 0
63D2- 34 65       3210             .SI CMD.L-1
63D4- 54 00       3220             .BY 'T' 0
63D6- A0 65       3230             .SI CMD.T-1
63D8- 4D 00       3240             .BY 'M' 0
63DA- 53 65       3250             .SI CMD.M-1
63DC- 52 52 00    3260             .BY 'RR' 0
63DF- 1E 65       3270             .SI CMD.RR-1
63E1- 52 4C 00    3280             .BY 'RL' 0
63E4- 2A 65       3290             .SI CMD.RL-1
63E6- 4A 55 00    3300             .BY 'JU' 0
63E9- EC 64       3310             .SI CMD.JU-1
63EB- 46 46 00    3320             .BY 'FF' 0
63EE- DA 64       3330             .SI CMD.FF-1
63F0- 3B 00       3340             .BY ';' 0
63F2- D9 64       3350             .SI CMNT-1
63F4- 53 48 41    3360             .BY 'SHAPE' 0
63F7- 50 45 00
63FA- 23 67       3370             .SI SHAPE-1
63FC- 53 57 41    3380             .BY 'SWAP' 0
63FF- 50 00
6401- E3 64       3390             .SI CMD.SWAP-1
```

```
6403- 53 00                     .BY 'S' 0
6405- 16 66                     .SI CMD.S-1
6407- 4E 4F 46                  .BY 'NOFILL' 0
640A- 49 4C 4C
640D- 00
640E- 82 67                     .SI CMD.NOFILL-1
6410- 46 4F 4F                  .BY 'FOOT' 0
6413- 54 00
6415- 53 67                     .SI CMD.FOOT-1
6417- 56 53 50                  .BY 'VSPACE' 0
641A- 41 43 45
641D- 00
641E- 42 67                     .SI CMD.VSPC-1
                 ; HERE ARE THE TWO NEW COMMANDS BEING PATCHED IN
6420- 4E 00                     .BY 'N' 0
6422- 2B 64                     .SI CMD.NUM-1
6424- 46 49 4C                  .BY 'FILE' 0
6427- 45 00
6429- 43 64                     .SI CMD.FIL-1
                 ;
                 ;
642B- 00                        .BY 0          ;END OF TABLE (EOT)
                 ;

642C- C8         CMD.NUM        INY
642D- 20 02 B5                  JSR MPSPACE    !MOVE TO PAGE NUMBER
6430- 88                        DEY
6431- A9 24                     LDA #'$        !FLAG NUMBER AS HEX
6433- 99 35 01                  STA CRT,Y
6436- 20 4A E2                  JSR GSYT_PRO_S
6439- A5 D1                     LDA *PROC_ADDRS
643B- 8D EE 67                  STA PAGECTR
643E- A5 D2                     LDA *PROC_ADDRS+1
6440- 8D EF 67                  STA PAGECTR+1
6443- 60                        RTS

6444- 68         CMD.FIL        PLA            !PULL OFF RETURN ADDRESS
6445- 68                        PLA            !TO PROCMCODE
6446- 68                        PLA            !PULL OFF THE OLD
6447- 68                        PLA            !CONTENTS OF SCRATO
6448- C8                        INY
6449- 20 FF B4                  JSR MNEXT      !MOVE TO FILE NUMBER
644C- 20 4A E2                  JSR GSYT_PRO_S
644F- A5 D1                     LDA *PROC_ADDRS
6451- 8D 10 01                  STA $110       !# OF FILE TO GET
6454- 20 68 64                  JSR LOAD.FIL
6457- AD 28 01   L.LOOP         LDA $128       !HEADER FILE NUMBER
645A- CD 10 01                  CMP $110       !IS THIS THE ONE?
645D- F0 06                     BEQ GOT.IT     !IF SO, EXIT LOAD LOOP
645F- 20 76 64                  JSR LOD.NXT.F
6462- 4C 57 64                  JMP L.LOOP

6465- 4C A0 B0   GOT.IT         JMP $B0A0      !SET EOF MARK IN TEXT FILE

6468- 20 96 B0   LOAD.FIL       JSR $B096      !CLEAR TEXT FILE
646B- A5 D3                     LDA *$D3       !PRESENT END OF TEXT
646D- 85 DD                     STA *SCRATO
646F- A5 D4                     LDA *$D4
6471- 85 DE                     STA *SCRATO+1
```

```
6473- 20 88 E3  3910               JSR $E388     !TAPE 1 ON
6476- 20 11 E5  3920 LOD.NXT.F     JSR $E511     !HEADER PARMS TO TAPE PARMS
6479- 8D 23 01  3930               STA $123      !MEMORY LOAD FLAG
647C- 20 5D EF  3940               JSR $EF5D     !READ IN HEADER FILE
647F- D0 49     3950               BNE ERROR     !BRANCH ON CHKSUM ERROR
                3951
                3952 ;       DETERMINE NUMBER OF BYTES TO BE LOADED
                3953
6481- A5 DD     3960               LDA *SCRAT0
6483- 8D 24 01  3970               STA $124      !TAPE STARTING ADDRESS
6486- A5 DE     3980               LDA *SCRAT0+1
6488- 8D 25 01  3990               STA $125
648B- 38        4000               SEC
648C- AD 2B 01  4010               LDA $12B      !HEADER END
648F- ED 29 01  4020               SBC $129      !HEADER START
6492- 48        4030               PHA
6493- AD 2C 01  4040               LDA $12C      !HEADER END+1
6496- ED 2A 01  4050               SBC $12A      !HEAD START+1
6499- AA        4060               TAX
                4061
                4062 ;       SET UP END ADDRESS FOR LOADING
                4063
649A- 68        4070               PLA
649B- 85 D1     4080               STA *PROC_ADDRS
649D- 18        4090               CLC
649E- 65 DD     4100               ADC *SCRAT0
64A0- 8D 26 01  4110               STA $126      !TAPE END
64A3- 8A        4120               TXA
64A4- 85 D2     4130               STA *PROC_ADDRS+1
64A6- 65 DE     4140               ADC *SCRAT0+1
64A8- 8D 27 01  4150               STA $127      !TAPE END+1
64AB- A9 00     4160               LDA #0        !FLAG NO LOAD
64AD- 8D 23 01  4170               STA $123
64B0- AD 10 01  4180               LDA $110
64B3- F0 05     4190               BEQ F.OK      !LOAD IF NO FILE SPECIFIED
64B5- CD 28 01  4200               CMP $128      !LOAD IF THIS IS THE ONE
64B8- D0 03     4210               BNE RANGE.OK
64BA- EE 23 01  4220 F.OK          INC $123      !FLAG LOAD TO MEMORY
64BD- 20 5D EF  4230 RANGE.OK      JSR $EF5D     !READ IN FILE
64C0- D0 08     4240               BNE ERROR     !BRANCH ON CHKSUM ERROR
64C2- A2 00     4250               LDX #0
64C4- 20 97 E5  4260               JSR $E597     !ADJUST END OF FILE PTR
64C7- 4C 96 E3  4270               JMP $E396     !TAPE 1 OFF
                4275
64CA- A2 00     4280 ERROR         LDX #0
64CC- AD 23 01  4290               LDA $123
64CF- F0 03     4300               BEQ ERR.17
64D1- 8D 12 01  4310               STA $112      !HALT ON ERRORS
64D4- 20 A0 B0  4320 ERR.17        JSR $B0A0
64D7- 4C 36 B4  4330               JMP $B436     !CHECKSUM ERROR
```

USEFUL SUBROUTINES
------ -----------

RAE-1 contains several subroutines which duplicate those in SUPERMON, since it (RAE) was originally written, under another name, for another 6502 based computer system, with a much less elegant monitor. These we shall not consider further.

A second group of subroutines includes those necessary to link RAE to pre- and post- processors, such as, for example, SWP-1. It is necessary to know how RAE files and terminal I/O are handled. These are best learned, short of having available the original source code, by studying the source code for SWP-1. Carl Moser has been very helpful in this regard, by answering all of our questions, by letting us examine the RAE source code, and by permitting the publishing of the SWP source.

The third group of subroutines includes those involved in disk and cassette I/O operations. A good insight into disk I/O can be gained by examining Tom Gettys' RAE/FODS Linkage program in RAE Notes - Issue No. 2.

Concerning the cassette interface, however, one of the most frequently asked questions about RAE is "Why doesn't my 'READ' cassette turn on with 'LOAD' in BASIC and/or with 'L2' in MON?" (this, of course, after having installed the second cassette remote control circuit, which is supported ONLY by RAE). We can only answer this one by pointing out that RAE has its own 'GET' subroutine, which, in addition to reading the separate 'header' file which carries the file id number for the following 'data' file, turns on the 'READ' cassette motor but NOT the 'WRITE' cassette motor.

To implement automatic control of the 'second' cassette from BASIC, you wil need a similar subroutine, patched at $00C9, and, from MON, you will need to write your own 'L3' ! The following information is provided to help you with these tasks.

MISCELLANEOUS INFORMATION ON RAE-1 CASSETTE I/O SUBROUTINES
------------- ----------- -- ----- -------- --- -----------

```
TAPE_OF_0       .DI $E30F       TURN OFF WRITE RECORDER
TAPE_OF_1       .DI $E318
TAPE_ON_0       .DI $E321
TAPE_ON_1       .DI $E32A
4SEC_DELAY      .DI $E36A
2SEC_DELAY      .DI $E36D
1SEC_DELAY      .DI $E370
TAPE_ON_1D      .DI $E388       TURN ON READ RECORDER, WITH DELAY
TAPE_ON_0D      .DI $E38F
TAPE_OF_1D      .DI $E396
TAPE_OF_0D      .DI $E39D

U/LOAD          .DI $EF5D       START OF MODIFIED CASSETTE I/O ROUTINE

LOAD_C          .DI $B0BC       GEt COMMAND IS IMPLEMENTED STARTING HERE
REC/COMMON      .DI $E524       PUt COMAND IS IMPLEMENTED STARTING HERE
```

(INCIDENTALLY, THE RAE COMMAND VECTOR TABLE RUNS FROM
$B741 TO $B7B0, FIRST THE TWO BYTE COMMAND, THEN THE ADDRESS)

SUGGEST THE FOLLOWING APPROACH: YOUR BASIC "LOAD" OR MON "L3" COULD
BEGIN WITH JSR START, JSR TAPE_OF_0, JSR TAPE_ON_1, AND SO ON.
IT SHOULD END WITH JSR TAPE_OF_1, RTS.
AGAIN INCIDENTALLY, BROWN'S EXTENDED SYM BASIC (ESB-1) INCLUDES FULL
DUAL CASSETTE CONTROL, AS WITH RAE, BUT DOES NOT CALL ON RAE ROUTINES.