Apple Assembly Line

Volume 1 -- Issue 2 November, 1980

Our second issue is 33% larger than the first! And not only so, but also there is useful information on the back page! I found a source for 6x9 white envelopes, so your address can be external to the newsletter, and so your copy will arrive in better condition. In less than a month since the newsletter was first announced, we already have over 45 paid subscribers. They are sprinkled all over the map, including one in Japan!

In This Issue...

A Bug in S-C ASSEMBLER II Disk Version 4.0

One real bug has turned up, and a few of you have had the bad luck to discover it the hard way. The assembler is free-format, in that opcodes and directives may start in any column after the blank which terminates the label field. However, the ".IN" directive will malfunction unless there are at least six spaces. If you tab over before typing ".IN" there will be no problem. However, if you type your line like "1230 .IN FILE1", with only two spaces between the line number and the period, you are in for a long wait. The processor goes into a loop printing D's. If you have the MON C mode on, you will see "LOADDDDDDDDDDDDDDDD....." with D's forever appear on your screen. Remember to TAB OVER, and it will not malfunction.

One fancied bug has been reported, and I would like to explain it. A user pointed out that you cannot shorten the SAVE command to three letters if you wish to save the source program on a disk file. Why? Because "SAVE" or "SAV" with no file name is not a DOS command. It is an assembler command to save the source program on cassette tape! On the other hand, SAVE with a filename is not an assembler command. It is a DOS command, and the assembler never sees it. The same goes for "LOAD", "LOA", and LOAD with a filename.

Variable Cross Reference for Applesoft Programs

Besides illustrating a lot of programming techniques, the VCR program is a very useful tool when you are writing large Applesoft programs. As listed here, it requires a 48K Apple, and assumes that HIMEM is set to at least $8AA7. You BRUN it, and it sets up the &-vector. When you are ready to print a cross reference, you merely type "&" and a carriage return, and out it comes. It is very fast: About 15 times faster than the VCR program included in Apple's DOS Tool Kit. It also takes less memory than Apple's version, both for the program itself and for the tables it constructs during execution.

The main body of the program is in lines 1400 thru 1460. After calling INITIALIZATION, the subroutine PROCESS.LINE is called until there are no more lines. Then PRINT.REPORT is called, and finally INITIALIZATION is called again to restore Applesoft's tables to their original form.

INITIALIZATION sets up PNTR to point to the beginning of the program, and EOT to point to the end of the table area. It also clears out a set of 26 2-byte pointers in HSHTBL (hash table). PROCESS.LINE scans a single line looking for variables by calling SCAN.FOR.VARIABLES, until the end of the program is reached. PRINT.REPORT merely prints a nice orderly report from the data which has been stored in the table by SCAN.FOR.VARIABLES.

The symbol table routines used in VCR are very similar to the ones used inside S-C ASSEMBLER II Version 4.0. There are 26 pointers starting at HSHTBL ($280), each one representing one letter of the alphabet. The first letter of a variable name selects one of these pointers. The pointer points at the first entry in a chain of variable names. When a new variable name is found, it is inserted in the appropriate chain at the place where it will be in alphabetical order. A sub-chain is kept for each variable name of all the line numbers from which it is referenced. The line number chain is maintained in numerical order. Thus there is no sorting necessary when it comes time to print the report.

Since no routines from the Applesoft ROMs are used, VCR will work with no changes with the RAM version of Applesoft. Since it loads below $9000, it will not conflict with Neil Konzen's PLE (Program Line Editor). Since it is just straight-forward code, with no address tables or embedded data, you can easily relocate it to a different running address; only the 3-byte instructions with the third byte equal to $88, $89, or $8A need to be changed. Or, you can type it in, and use a different origin (line 1040).

If you like to modify programs, this one needs one improvement. (Only one?) I forgot to take note of the FN token, so any FN definitions or uses will look like references to an array variable. Another kind of modification, called "major" perhaps, will turn the VCR into LNCR (Line Number Cross Reference).

```
:ASM
                 1000 *---------------------------------
                 1010 *      VARIABLE CROSS REFERENCE
                 1020 *       FOR APPLESOFT PROGRAMS
                 1030 *---------------------------------
8800-            1040 ZZ.BEG .EQ $8800
                 1050        .OR ZZ.BEG
                 1060        .TF B.VCR
                 1070 *---------------------------------
8800- A9 4C      1080        LDA #$4C      AMPERSAND VECTOR
8802- 8D F5 03   1090        STA $3F5
8805- A9 10      1100        LDA #VCR
8807- 8D F6 03   1110        STA $3F6
880A- A9 88      1120        LDA /VCR
880C- 8D F7 03   1130        STA $3F7
880F- 60         1140        RTS
                 1150 *---------------------------------
0018-            1160 PNTR   .EQ $18,19    POINTER INTO PROGRAM
001A-            1170 DATA   .EQ $1A THRU $1D
001A-            1180 LZFLAG .EQ $1A       LEADING ZERO FLAG
001A-            1190 NEXTLN .EQ $1A,1B    ADDRESS OF NEXT LINE
001C-            1200 LINNUM .EQ $1C,1D    CURRENT LINE NUMBER
001E-            1210 STPNTR .EQ $1E,1F    POINTER INTO VARIABLE TABLE
009B-            1220 TPTR   .EQ $9B,9C    TEMP POINTER
009D-            1230 SYMBOL .EQ $9D THRU $A4  8 BYTES
009E-            1240 VARNAM .EQ SYMBOL+1
0280-            1250 HSHTBL .EQ $280
00A5-            1260 ENTRY.SIZE .EQ $A5,A6
                 1270 *---------------------------------
0067-            1280 PRGBOT .EQ $67,68    BEGINNING OF PROGRAM
0069-            1290 LOMEM  .EQ $69,6A    BEGINNING OF VARIABLE SPACE
006B-            1300 EOT    .EQ $6B,6C    END OF VARIABLE TABLE
                 1310 *---------------------------------
00B2-            1320 TKN.REM     .EQ 178
0083-            1330 TKN.DATA    .EQ 131
                 1340 *---------------------------------
0024-            1350 MON.CH      .EQ $24
F94A-            1360 MON.PRBL2   .EQ $F94A
FDED-            1370 MON.COUT    .EQ $FDED
FD8E-            1380 MON.CROUT   .EQ $FD8E
                 1390 *---------------------------------
                 1400 VCR
8810- 20 1F 88   1410        JSR INITIALIZATION
8813- 20 3A 88   1420 .1     JSR PROCESS.LINE
8816- D0 FB      1430        BNE .1        UNTIL END OF PROGRAM
8818- 20 A9 89   1440        JSR PRINT.REPORT
881B- 20 1F 88   1450        JSR INITIALIZATION  ERASE VARIABLE TABLE
881E- 60         1460        RTS
                 1470 *---------------------------------
                 1480 INITIALIZATION
881F- A5 69      1490        LDA LOMEM
8821- 85 6B      1500        STA EOT
8823- A5 6A      1510        LDA LOMEM+1
8825- 85 6C      1520        STA EOT+1
8827- A2 34      1530        LDX #52       # OF BYTES FOR HASH POINTERS
8829- A9 00      1540        LDA #0
882B- 9D 7F 02   1550 .1     STA HSHTBL-1,X
882E- CA         1560        DEX
882F- D0 FA      1570        BNE .1
8831- A5 67      1580        LDA PRGBOT
8833- 85 18      1590        STA PNTR
8835- A5 68      1600        LDA PRGBOT+1
8837- 85 19      1610        STA PNTR+1
8839- 60         1620        RTS
                 1630 *---------------------------------
                 1640 PROCESS.LINE
883A- A0 03      1650 .3     LDY #3        CAPTURE POINTER AND LINE #
883C- B1 18      1660 .1     LDA (PNTR),Y
883E- 99 1A 00   1670        STA DATA,Y
8841- 88         1680        DEY
8842- 10 F8      1690        BPL .1
8844- 18         1700        CLC           SKIP OVER DATA
8845- A5 18      1710        LDA PNTR
8847- 69 04      1720        ADC #4
8849- 85 18      1730        STA PNTR
884B- 90 02      1740        BCC .2
884D- E6 19      1750        INC PNTR+1
884F- 20 5D 88   1760 .2     JSR SCAN.FOR.VARIABLES
```

```
8852- A5 1A     1770        LDA DATA
8854- 85 18     1780        STA PNTR
8856- A5 1B     1790        LDA DATA+1
8858- 85 19     1800        STA PNTR+1
885A- D0 DE     1810        BNE .3
885C- 60        1820        RTS
                1830 *--------------------------------
                1840 SCAN.FOR.VARIABLES
885D- 20 8E 88  1850 .1     JSR GET.NEXT.VARIABLE
8860- F0 2B     1860        BEQ .3        END OF LINE
8862- 20 CE 88  1870        JSR PACK.VARIABLE.NAME
8865- 20 06 89  1880        JSR SEARCH.VARIABLE.TABLE
8868- 90 16     1890        BCC .2        FOUND SAME VARIABLE
886A- A9 00     1900        LDA #0
886C- 85 A1     1910        STA SYMBOL+4  START OF LINE NUMBER CHAIN
886E- 85 A2     1920        STA SYMBOL+5
8870- A5 1D     1930        LDA LINNUM+1  MSB FIRST
8872- 85 A3     1940        STA SYMBOL+6
8874- A5 1C     1950        LDA LINNUM
8876- 85 A4     1960        STA SYMBOL+7
8878- A9 08     1970        LDA #8        ADD 8 BYTE ENTRY
887A- 20 47 89  1980        JSR ADD.NEW.ENTRY
887D- 4C 5D 88  1990        JMP .1
8880- 20 89 89  2000 .2     JSR SEARCH.LINE.CHAIN
8883- 90 D8     2010        BCC .1        FOUND SAME LINE NUMBER
8885- A9 04     2020        LDA #4        ADD 4 BYTE ENTRY
8887- 20 47 89  2030        JSR ADD.NEW.ENTRY
888A- 4C 5D 88  2040        JMP .1
888D- 60        2050 .3     RTS
                2060 *--------------------------------
                2070 GET.NEXT.VARIABLE
888E- 20 AC 88  2080 .1     JSR NEXT.CHAR.NOT.QUOTE
8891- F0 0D     2090        BEQ .2        END OF LINE
8893- C9 83     2100        CMP #TKN.DATA
8895- F0 0A     2110        BEQ .3
8897- C9 B2     2120        CMP #TKN.REM
8899- F0 05     2130        BEQ .2        SKIP TO NEXT LINE
889B- 20 D2 89  2140        JSR LETTER    LETTER?
889E- 90 EE     2150        BCC .1        NO, KEEP LOOKING
88A0- 60        2160 .2     RTS
                2170 *   DATA, SO SKIP TO NEXT STATEMENT
88A1- 20 AC 88  2180 .3     JSR NEXT.CHAR.NOT.QUOTE
88A4- F0 FA     2190        BEQ .2        EOL, RETURN
88A6- C9 3A     2200        CMP #':       COLON?
88A8- D0 F7     2210        BNE .3        NOT END YET
88AA- F0 E2     2220        BEQ .1        ...ALWAYS
                2230 *--------------------------------
                2240 NEXT.CHAR.NOT.QUOTE
88AC- 20 C1 88  2250 .1     JSR NEXT.CHAR
88AF- F0 04     2260        BEQ .2        EOL, RETURN
88B1- C9 22     2270        CMP #'"       QUOTE?
88B3- F0 01     2280        BEQ .3        YES, SCAN OVER QUOTATION
88B5- 60        2290 .2     RTS           RETURN
88B6- 20 C1 88  2300 .3     JSR NEXT.CHAR
88B9- F0 FA     2310        BEQ .2        EOL, RETURN
88BB- C9 22     2320        CMP #'"       TERMINAL QUOTE?
88BD- D0 F7     2330        BNE .3        NOT YET
88BF- F0 EB     2340        BEQ .1        ...ALWAYS
                2350 *--------------------------------
                2360 *      NEXT CHARACTER FROM LINE
                2370 *        CALL:  JSR NEXT.CHAR
                2380 *      RETURN:  (A)=CHAR FROM LINE
                2390 *               IF CHAR .NE. EOL,
                2400 *                  INCREMENT PNTR AND
                2410 *                  STATUS Z=0
                2420 *               IF CHAR .EQ. EOL,
                2430 *                  STATUS Z=1
                2440 *--------------------------------
                2450 NEXT.CHAR
88C1- A0 00     2460        LDY #0
88C3- B1 18     2470        LDA (PNTR),Y
88C5- F0 06     2480        BEQ .1        EOL
88C7- E6 18     2490        INC PNTR      BUMP POINTER
88C9- D0 02     2500        BNE .1
88CB- E6 19     2510        INC PNTR+1
88CD- 60        2520 .1     RTS
                2530 *--------------------------------
                2540 PACK.VARIABLE.NAME
88CE- 85 9E     2550        STA VARNAM    FIRST CHAR OF NAME
88D0- A9 20     2560        LDA #'        BLANKS FOR OTHER TWO CHARS
```

```
88D2- 85 9F              STA VARNAM+1
88D4- 85 A0              STA VARNAM+2
88D6- 20 C1 88           JSR NEXT.CHAR
88D9- F0 2A              BEQ .5        END OF LINE
88DB- 20 CA 89           JSR LTRDIG
88DE- 90 0C              BCC .2        NOT LETTER OR DIGIT
88E0- 85 9F              STA VARNAM+1
88E2- 20 C1 88     .1    JSR NEXT.CHAR IGNORE EXCESS NAME
88E5- F0 1E              BEQ .5        END OF LINE
88E7- 20 CA 89           JSR LTRDIG
88EA- B0 F6              BCS .1        LETTER OR DIGIT
88EC- C9 24        .2    CMP #'$       DOLLAR SIGN?
88EE- F0 04              BEQ .3        YES
88F0- C9 25              CMP #'%       PER CENT?
88F2- D0 07              BNE .4        NO
88F4- 85 A0        .3    STA VARNAM+2
88F6- 20 C1 88           JSR NEXT.CHAR
88F9- F0 0A              BEQ .5        END OF LINE
88FB- C9 28        .4    CMP #'(       LEFT PAREN?
88FD- D0 06              BNE .5
88FF- A5 A0              LDA VARNAM+2 SET HIGH BIT
8901- 09 80              ORA #$80      TO FLAG ARRAY
8903- 85 A0              STA VARNAM+2 REFERENCE
8905- 60           .5    RTS
                   *--------------------------------
                   SEARCH.VARIABLE.TABLE
8906- 38                 SEC           CONVERT 1ST CHAR TO
8907- A5 9E              LDA VARNAM    HASH TABLE INDEX
8909- E9 41              SBC #'A
890B- 0A                 ASL
890C- 69 80              ADC #HSHTBL
890E- 85 1E              STA STPNTR
8910- A9 02              LDA /HSHTBL
8912- 69 00              ADC #0
8914- 85 1F              STA STPNTR+1
                   *---  FALL INTO CHAIN SEARCH ROUTINE
                   *--------------------------------
                   CHAIN.SEARCH
8916- A0 00        .1    LDY #0        POINT AT POINTER IN ENTRY
8918- B1 1E              LDA (STPNTR),Y
891A- 85 9B              STA TPTR
891C- C8                 INY
891D- B1 1E              LDA (STPNTR),Y
891F- F0 1A              BEQ .4        END OF CHAIN, NOT IN TABLE
8921- 85 9C              STA TPTR+1
8923- A2 02              LDX #2        2 MORE CHARS IN SYMBOL
8925- A0 02              LDY #2        POINT AT NAME IN ENTRY
8927- B1 9B        .2    LDA (TPTR),Y COMPARE NAMES
8929- D9 9D 00           CMP SYMBOL,Y
892C- 90 08              BCC .3        NOT THIS ONE, BUT KEEP LOOKING
892E- D0 0B              BNE .4        NOT IN THIS CHAIN
8930- CA                 DEX
8931- F0 0A              BEQ .5        NAME IS THE SAME
8933- C8                 INY           NEXT BYTE PAIR
8934- D0 F1              BNE .2        ...ALWAYS
                   *--------------------------------
8936- 20 3D 89     .3    JSR .5        UPDATE POINTER, CLEAR CARRY
8939- 90 DB              BCC .1        ...ALWAYS
                   *--------------------------------
893B- 38           .4    SEC           DID NOT FIND
893C- 60                 RTS
                   *--------------------------------
893D- A5 9B        .5    LDA TPTR
893F- 85 1E              STA STPNTR
8941- A5 9C              LDA TPTR+1
8943- 85 1F              STA STPNTR+1
8945- 18                 CLC
8946- 60                 RTS
                   *--------------------------------
                   ADD.NEW.ENTRY
8947- 85 A5              STA ENTRY.SIZE
8949- 18                 CLC           SEE IF ROOM
894A- A2 01              LDX #1
894C- A0 00              LDY #0
894E- 84 A6              STY ENTRY.SIZE+1
8950- B1 1E        .1    LDA (STPNTR),Y  GET CURRENT POINTER
8952- 99 9D 00           STA SYMBOL,Y
8955- B9 6B 00           LDA EOT,Y
8958- 91 1E              STA (STPNTR),Y
895A- 99 9B 00           STA TPTR,Y
```

```
895D- 79 A5 00           ADC ENTRY.SIZE,Y
8960- 99 6B 00           STA EOT,Y
8963- C8                 INY
8964- CA                 DEX
8965- 10 E9              BPL .1
               *---      SEE IF GOING TO BE ENOUGH ROOM
8967- A5 6B              LDA EOT
8969- C9 00              CMP #ZZ.BEG
896B- A5 6C              LDA EOT+1
896D- E9 88              SBC /ZZ.BEG
896F- B0 14              BCS .3         MEM FULL ERR
               *---      MOVE ENTRY INTO VARIABLE TABLE
8971- A4 A5              LDY ENTRY.SIZE
8973- 88                 DEY
8974- B9 9D 00 .2        LDA SYMBOL,Y
8977- 91 9B              STA (TPTR),Y
8979- 88                 DEY
897A- 10 F8              BPL .2
897C- A5 9B              LDA TPTR
897E- 85 1E              STA STPNTR
8980- A5 9C              LDA TPTR+1
8982- 85 1F              STA STPNTR+1
8984- 60                 RTS
8985- 4C 88 89 .3        JMP MEM.FULL.ERR
               MEM.FULL.ERR
8988- 00                 BRK
               *--------------------------------
               SEARCH.LINE.CHAIN
8989- 18                 CLC            ADJUST POINTER TO START
898A- A5 1E              LDA STPNTR     OF LINE # CHAIN
898C- 69 04              ADC #4
898E- 85 9D              STA SYMBOL
8990- A5 1F              LDA STPNTR+1
8992- 69 00              ADC #0
8994- 85 9E              STA SYMBOL+1
8996- A9 9D              LDA #SYMBOL
8998- 85 1E              STA STPNTR
899A- A9 00              LDA /SYMBOL
899C- 85 1F              STA STPNTR+1
899E- A5 1C              LDA LINNUM     PUT LINE NUMBER INTO SYMBOL
89A0- 85 A0              STA SYMBOL+3
89A2- A5 1D              LDA LINNUM+1
89A4- 85 9F              STA SYMBOL+2
89A6- 4C 16 89           JMP CHAIN.SEARCH
               *--------------------------------
               PRINT.REPORT
89A9- A9 41              LDA #'A        START WITH A'S
89AB- 85 9E    .1        STA VARNAM
89AD- 38                 SEC
89AE- E9 41              SBC #'A        CONVERT TO HSHTBL INDEX
89B0- 0A                 ASL
89B1- A8                 TAY
89B2- B9 81 02           LDA HSHTBL+1,Y
89B5- F0 0A              BEQ .2         NO ENTRY FOR THIS LETTER
89B7- 85 19              STA PNTR+1
89B9- B9 80 02           LDA HSHTBL,Y
89BC- 85 18              STA PNTR
89BE- 20 DE 89           JSR PRINT.LETTER.CHAIN
89C1- E6 9E    .2        INC VARNAM     NEXT LETTER
89C3- A5 9E              LDA VARNAM
89C5- C9 5B              CMP #'Z+1
89C7- 90 E2              BCC .1         STILL MORE LETTERS
89C9- 60                 RTS            FINISHED
               *--------------------------------
               LTRDIG
89CA- C9 30              CMP #'0        DIGIT?
89CC- 90 0D              BCC LD1        NO
89CE- C9 3A              CMP #'9+1
89D0- 90 0A              BCC LD2        YES
               LETTER
89D2- C9 41              CMP #'A        LETTER?
89D4- 90 05              BCC LD1        NO
89D6- C9 5B              CMP #'Z+1
89D8- 90 02              BCC LD2        YES
89DA- 18                 CLC            NO
89DB- 60       LD1       RTS
89DC- 38       LD2       SEC
89DD- 60                 RTS
               *--------------------------------
               PRINT.LETTER.CHAIN
```

```
89DE- A5 9E     4170 .1      LDA VARNAM    FIRST LETTER
89E0- 20 A1 8A  4180         JSR PRINT.CHAR
89E3- A0 01     4190         LDY #1
89E5- C8        4200 .2      INY
89E6- B1 18     4210         LDA (PNTR),Y REST OF NAME
89E8- 29 7F     4220         AND #$7F
89EA- C9 20     4230         CMP #'        BLANK?
89EC- F0 03     4240         BEQ .3
89EE- 20 A1 8A  4250         JSR PRINT.CHAR
89F1- C0 03     4260 .3      CPY #3
89F3- 90 F0     4270         BCC .2
89F5- B1 18     4280         LDA (PNTR),Y CHECK IF ARRAY
89F7- 10 05     4290         BPL .4
89F9- A9 28     4300         LDA #'(
89FB- 20 A1 8A  4310         JSR PRINT.CHAR
89FE- 18        4320 .4      CLC           POINT AT LINE # CHAIN
89FF- A5 18     4330         LDA PNTR
8A01- 69 04     4340         ADC #4
8A03- 85 9B     4350         STA TPTR
8A05- A5 19     4360         LDA PNTR+1
8A07- 69 00     4370         ADC #0
8A09- 85 9C     4380         STA TPTR+1
8A0B- 20 23 8A  4390         JSR PRINT.LINNUM.CHAIN
8A0E- 20 8E FD  4400         JSR MON.CROUT
8A11- A0 01     4410         LDY #1
8A13- B1 18     4420         LDA (PNTR),Y POINTER TO NEXT VARIABLE
8A15- F0 0B     4430         BEQ .5        NO MORE
8A17- 48        4440         PHA
8A18- 88        4450         DEY
8A19- B1 18     4460         LDA (PNTR),Y
8A1B- 85 18     4470         STA PNTR
8A1D- 68        4480         PLA
8A1E- 85 19     4490         STA PNTR+1
8A20- D0 BC     4500         BNE .1        ...ALWAYS
8A22- 60        4510 .5      RTS
                4520 *-------------------------------
                4530 PRINT.LINNUM.CHAIN
8A23- 20 49 8A  4540 .1      JSR TAB.NEXT.COLUMN
8A26- A0 02     4550         LDY #2        POINT AT LINE #
8A28- B1 9B     4560         LDA (TPTR),Y
8A2A- 85 1D     4570         STA LINNUM+1
8A2C- C8        4580         INY
8A2D- B1 9B     4590         LDA (TPTR),Y
8A2F- 85 1C     4600         STA LINNUM
8A31- 20 60 8A  4610         JSR PRINT.LINE.NUMBER
8A34- A0 01     4620         LDY #1        SET UP NEXT POINTER
8A36- B1 9B     4630         LDA (TPTR),Y
8A38- F0 0B     4640         BEQ .2
8A3A- 48        4650         PHA
8A3B- 88        4660         DEY
8A3C- B1 9B     4670         LDA (TPTR),Y
8A3E- 85 9B     4680         STA TPTR
8A40- 68        4690         PLA
8A41- 85 9C     4700         STA TPTR+1
8A43- D0 DE     4710         BNE .1        ...ALWAYS
8A45- 60        4720 .2      RTS
                4730 *-------------------------------
                4740 TAB.NEW.LINE
8A46- 20 8E FD  4750         JSR MON.CROUT
                4760 TAB.NEXT.COLUMN
8A49- A9 07     4770 .1      LDA #7        FIRST TAB STOP
8A4B- C5 24     4780 .2      CMP MON.CH    CURSOR POSITION
8A4D- B0 08     4790         BCS .3        PERFORM TAB
8A4F- 69 06     4800         ADC #6        NEXT TAB STOP
8A51- C9 21     4810         CMP #33       END OF LINE?
8A53- 90 F6     4820         BCC .2
8A55- B0 EF     4830         BCS TAB.NEW.LINE   ...ALWAYS
8A57- F0 06     4840 .3      BEQ .4        ALREADY THERE
8A59- E5 24     4850         SBC MON.CH    CALCULATE # OF BLANKS
8A5B- AA        4860         TAX
8A5C- 20 4A F9  4870         JSR MON.PRBL2
8A5F- 60        4880 .4      RTS
                4890 *-------------------------------
                4900 PRINT.LINE.NUMBER
8A60- A2 04     4910         LDX #4        PRINT 5 DIGITS
8A62- 86 1A     4920         STX LZFLAG    TURN ON LEADING ZERO FLAG
8A64- A9 30     4930 .1      LDA #'0       DIGIT=0
8A66- 48        4940 .2      PHA
8A67- 38        4950         SEC
8A68- A5 1C     4960         LDA LINNUM
```

```
8A6A- FD 97 8A 4970            SBC PLNTBL,X
8A6D- 48       4980            PHA
8A6E- A5 1D    4990            LDA LINNUM+1
8A70- FD 9C 8A 5000            SBC PLNTBH,X
8A73- 90 0A    5010            BCC .3         LESS THAN DIVISOR
8A75- 85 1D    5020            STA LINNUM+1
8A77- 68       5030            PLA
8A78- 85 1C    5040            STA LINNUM
8A7A- 68       5050            PLA
8A7B- 69 00    5060            ADC #0         INCREMENT DIGIT
8A7D- D0 E7    5070            BNE .2         ...ALWAYS
8A7F- 68       5080 .3         PLA
8A80- 68       5090            PLA
8A81- C9 30    5100            CMP #'0
8A83- F0 0A    5110            BEQ .5         ZERO, MIGHT BE LEADING
8A85- 38       5120            SEC            TURN OFF LZFLAG
8A86- 66 1A    5130            ROR LZFLAG
8A88- 20 A1 8A 5140 .4         JSR PRINT.CHAR
8A8B- CA       5150            DEX
8A8C- 10 D6    5160            BPL .1
8A8E- 60       5170            RTS
8A8F- 24 1A    5180 .5         BIT LZFLAG     LEADING ZERO FLAG
8A91- 30 F5    5190            BMI .4         NO
8A93- A9 20    5200            LDA #'         BLANK
8A95- D0 F1    5210            BNE .4         ...ALWAYS
8A97- 01       5220 PLNTBL     .DA #1
8A98- 0A       5230            .DA #10
8A99- 64       5240            .DA #100
8A9A- E8       5250            .DA #1000
8A9B- 10       5260            .DA #10000
8A9C- 00       5270 PLNTBH     .DA /1
8A9D- 00       5280            .DA /10
8A9E- 00       5290            .DA /100
8A9F- 03       5300            .DA /1000
8AA0- 27       5310            .DA /10000
               5320 *--------------------------------
               5330 PRINT.CHAR
8AA1- 09 80    5340            ORA #$80
8AA3- 20 ED FD 5350            JSR MON.COUT
8AA6- 60       5360            RTS
               5370 *--------------------------------
8AA7-          5380 ZZ.END     .EQ *
02A7-          5390 ZZ.SIZ     .EQ ZZ.END-ZZ.BEG
```

Bags, boxes, et cetera

Since I sell software in stores, I buy a lot of zip-lock bags, cardboard mailing boxes, diskettes, and so on. I thought that maybe you need some of these, and haven't been able to find a source at good prices in small quantities. I will sell you some of mine, at the following prices:

6"x9" zip-lock bags	$8.50/100
9"x12" zip lock bags	$12/100
Verbatim diskettes	
without hub rings	$30 for box of ten, $265 for 100
with hub rings	$32 for box of ten, $285 for 100

Anything else you need? Let me know, maybe I have it or can get it for you or tell you where you can get it at a good price.

Assembly Source on Text Files

Version 4.0 of the S-C ASSEMBLER II allows you to EXEC a source program, if it is on a DOS text file. This is handy if you have created it with a different editor, or perhaps with a compiler. But what if you want to go the other way? What if you want to save a source program on a text file, so that it can be used in another editor, or by another assembler?

There is no built-in command to allow it, so I have now written a separate program to do it. The program loads at $0800 thru $093C, and does not borrow any code from the assembler. It does use some routines in the Monitor ROMs, and the DOS I/O rehook routine. If you BRUN the program, it will assume the pointers at $CA,CB and $4C,4D are bracketing a valid assembly source program, and try to list it on a text file.

The main body of the program is in lines 1190 thru 1630. Lines 1200 and 1210 serve to un-hook the S-C ASSEMBLER II from the output. They will also turn off your printer, if you had it on. Lines 1220 and 1230 tell DOS that it should recognize commands printed after a control-D. Lines 1240 and 1250 change the prompt symbol to a blank, so that the monitor input subroutine will not print a colon or some other character as the prompt when reading the file name.

Lines 1290 thru 1360 request you to enter a file name, read it into the monitor buffer starting at $0200, and move it to a safe place at $0280. It has to be moved, because when we print DOS commands later the area starting at $0200 will be written on by DOS.

Once the file name you have typed is safely stored at $0280 and following, lines 1410 thru 1490 will set up the file for writing. This is done in five steps. First, close all files. Second, issue an OPEN-DELETE-OPEN sequence, with the file name (of course); this will make sure that we are writing on a fresh empty file. Then the WRITE command is sent, and we are ready to roll.

Line 1530 calls a subroutine which lists your source program. since the file is OPEN and in WRITE mode, the listing goes into your text file. If you have MON O mode set, you will also see the listing on your screen. Note that it is not really necessary for me to use a subroutine at this point. ASM.LIST is only called once, and it is not very long. But I did it anyway, to keep the main body short enough to fit on a page, easy to understand, modular, structured, etc.

After the listing is completed, Line 1570 will close the text file. Lines 1610 and 1620 turn of the DOS run flag, so that DOS will not look for control-D commands. And finally, line 1630 re-enters the S-C ASSEMBLER II through its soft entry point.

For example, the source line

```
     1000 ABC    LDA SAM
```

```
is stored as:  OF  (total of 15 bytes in line image)
               E8 03   (line number 1000)
               41 42 43 84  ("ABC" and 4 blanks)
               4C 44 41 81  ("LDA" and 1 blank)
               53 41 4D     ("SAM")
               00           (end of line indicator)
```

The subroutine ASM.LIST.LINE, at lines 2490 thru 2610, prints one source line. A subroutine named GNB ("get next byte") is called to skip over the length byte, and to pick up the line number. PRINT.LINNUM is called to convert the line number to decimal and print it, with leading zeroes if necessary, as a four digit number. The loop at lines 2570 thru 2600 is seeded with a blank (because the blank between the line number and the label field is not actually stored in the source program), and the text of the line is printed. The loop prints a character, and then calls NEXT.TOKEN to get the next one. When the token returned equals $00, the line is finished.

GNB, lines 2630 thru 2690, clears the queued blank count, picks up the character pointed at by SRCP, and increments SRCP.

NEXT.TOKEN, lines 2710 thru 2820, tests the blank count. If it is non-zero, the count is decremented and a blank ($20) character is returned. If the count was zero, the next character is picked up from the line. If this character is not a blank count token, it is returned and the pointer in SRCP is incremented. If the character is a blank count token, it is saved, the SRCP pointer is incremented past the token, and then the count is decremented and a blank returned.

The PRINT.LINNUM routine, lines 2860 thru 3170, is a revision of a routine used in the Integer BASIC ROMs. I think it is commented well enough for you to follow. The general idea is to divide by 1000 and print the quotient; divide by 100 and print the quotient; then by 10; and finally print the remainder.

Since severaly of you have asked me to provide the capability to list programs onto text files, you should be pleased with this program. If you do not need it, then maybe it has shed some light on the internal structure of part of the assembler, or served as a tutorial in programming.

Lines 1670 thru 1780 are text strings, printed by the subroutine named PRINT.QUOTE. Each string is written with the sign bit of every byte zero except for the last byte. The sign bit of the last byte is 1, telling PRINT.QUOTE that it is finished. For example, the first message is the word "CLOSE" and a carriage return. The carriage return is entered in hex with the sign bit+1 as $8D. The second message is the word "OPEN", and the letter "N" is preceded by a minus sign in the .AS directive to indicate that the sign bit should be 1.

The PRINT.QUOTE subroutine is at lines 2140 thru 2200. It expects the Y-register to contain the offset of the desired message from the beginning of all the messages at QTS. It calls on PRINT.CHAR to actualy send each character.

PRINT.CHAR, at lines 2020 thru 2100, calls on the monitor print character routine at $FDED. This branches through DOS, and DOS writes the character on the text file. PRINT.CHAR saves and restores the Y-register and A-register contents. It also sets the sign bit on each character before printing it. Upon exit, the status will reflect the value of the character printed.

Lines 1820 thru 1980 issue a DOS command. The Y-register points at one of the message strings in QTS. Control-D is printed, followed by the command key word, a space, and the file name you previously typed. Since DOS does not allow slot and drive specifications on the WRITE command, and since it is sufficient to specify them only once, the subroutine chops them off after printing them once. The logic for this is in lines 1910 thru 1940: after printing a comma, it is replaced with a carriage return. The next time the name is printed, the carriage return will be the end.

The subroutine which really controls the listing is in lines 2330 thru 2450. The first four instructions set up a zero-page pointer SRCP to point at the beginning of your source program. Lines 2380 thru 2420 compare the pointer with HIMEM to see if the listing is completed. If you really had no source program, we would already be finished at this point. If there is another line (or more), the subroutine named ASM.LIST.LINE is called to list the next line. The process is repeated until the last line has been printed onto your text file.

At this point it might be helpful to explain how source lines are stored in memory. Each line begins with a single byte which contains the byte-count of the line. Next are a byte-pair containing the line number of the line, in the usual backwards 6502 format. The text of the line follows, and a final byte containing $00 ends the line. No carriage return is stored. Blanks are treated specially. A single blank is stored as $81. Two blanks in a row are replaced by one byte of value $82. Any string of blanks up to 63 blanks is thus replaced by a single token of value $80 plus the blank count. Longer strings of blanks will take more than one token.

```
:ASM

                  1000 *---------------------------------
                  1010 *      WRITE ASSEMBLY SOURCE ON A TEXT FILE
                  1020 *---------------------------------
                  1030        .OR $800
0033-             1040 MON.PROMPT .EQ $33
00CA-             1050 PP      .EQ $CA,CB
004C-             1060 HIMEM   .EQ $4C,4D
00D9-             1070 DOS.RUNFLAG .EQ $D9
0200-             1080 MON.BUFFER .EQ $200
0280-             1090 DOS.BUFFER .EQ $280
FD6A-             1100 MON.GETLN  .EQ $FD6A
FD8E-             1110 MON.CROUT  .EQ $FD8E
FDED-             1120 MON.COUT   .EQ $FDED
FE93-             1130 MON.SETVID .EQ $FE93
03EA-             1140 DOS.REHOOK .EQ $3EA
0000-             1150 BLANK.COUNT .EQ $00
0001-             1160 SRCP    .EQ $01,02
0003-             1170 LINNUM .EQ $03,04
                  1180 *---------------------------------
                  1190 TEXT.LIST
0800- 20 93 FE    1200        JSR MON.SETVID
0803- 20 EA 03    1210        JSR DOS.REHOOK
0806- A9 FF       1220        LDA #$FF
0808- 85 D9       1230        STA DOS.RUNFLAG
080A- A9 A0       1240        LDA #' '+$80   SET PROMPT CHAR = BLANK
080C- 85 33       1250        STA MON.PROMPT
                  1260 *---------------------------------
                  1270 *      GET FILE NAME
                  1280 *---------------------------------
080E- A0 15       1290        LDY #QFILNAM-QTS
0810- 20 9E 08    1300        JSR PRINT.QUOTE
0813- 20 6A FD    1310        JSR MON.GETLN
0816- A0 7F       1320        LDY #$7F      MOVE FILE NAME TO SEPARATE
0818- B9 00 02    1330 .1     LDA MON.BUFFER,Y
081B- 99 80 02    1340        STA DOS.BUFFER,Y
081E- 88          1350        DEY
081F- 10 F7       1360        BPL .1
                  1370 *---------------------------------
                  1380 *      SET UP THE TEXT FILE
                  1390 *      (CLOSE, OPEN, DELETE, OPEN, WRITE)
                  1400 *---------------------------------
0821- 20 A7 08    1410        JSR CLOSE.FILE               BUFFER
0824- A0 06       1420        LDY #QOPEN-QTS
0826- 20 6B 08    1430        JSR ISSUE.DOS.COMMAND
0829- A0 0A       1440        LDY #QDELETE-QTS
082B- 20 6B 08    1450        JSR ISSUE.DOS.COMMAND
082E- A0 06       1460        LDY #QOPEN-QTS
0830- 20 6B 08    1470        JSR ISSUE.DOS.COMMAND
0833- A0 10       1480        LDY #QWRITE-QTS
0835- 20 6B 08    1490        JSR ISSUE.DOS.COMMAND
                  1500 *---------------------------------
                  1510 *      LIST THE SOURCE PROGRAM
                  1520 *---------------------------------
0838- 20 B4 08    1530        JSR ASM.LIST
                  1540 *---------------------------------
                  1550 *      CLOSE THE FILE
                  1560 *---------------------------------
083B- 20 A7 08    1570        JSR CLOSE.FILE
                  1580 *---------------------------------
                  1590 *      RETURN TO CALLER
                  1600 *---------------------------------
083E- A9 00       1610        LDA #0
0840- 85 D9       1620        STA DOS.RUNFLAG
0842- 4C 03 10    1630        JMP $1003
                  1640 *---------------------------------
                  1650 *      MESSAGE TEXT
                  1660 *---------------------------------
0845-             1670 QTS     .EQ *
0845- 43 4C 4F
0848- 53 45       1680 QCLOSE  .AS /CLOSE/
084A- 8D          1690         .HS 8D
084B- 4F 50 45    1700 QOPEN   .AS /OPE/
084E- CE          1710         .AS -/N/
084F- 44 45 4C
0852- 45 54       1720 QDELETE .AS /DELET/
0854- C5          1730         .AS -/E/
0855- 57 52 49
```

```
0858- 54          QWRITE .AS /WRIT/
0859- C5                 .AS -/E/
085A- 0D          QFILNAM .HS 0D
085B- 54 45 58
085E- 54 20 46
0861- 49 4C 45
0864- 20 4E 41
0867- 4D 45 3A           .AS /TEXT FILE NAME:/
086A- A0                 .AS -/ /
                  *-----------------------------------
                  *      ISSUE DOS COMMAND
                  *-----------------------------------
                  ISSUE.DOS.COMMAND
086B- A9 84              LDA #$84     CONTROL-D
086D- 20 8E 08           JSR PRINT.CHAR
0870- 20 9E 08           JSR PRINT.QUOTE
0873- A0 00              LDY #0
0875- A9 20              LDA #'       PRINT A SPACE
0877- 20 8E 08    .5     JSR PRINT.CHAR
087A- C9 8D              CMP #$8D
087C- F0 0F              BEQ .7
087E- C9 2C              CMP #',      COMMA?
0880- D0 05              BNE .6
0882- A9 8D              LDA #$8D
0884- 99 80 02           STA DOS.BUFFER,Y
0887- B9 80 02    .6     LDA DOS.BUFFER,Y
088A- C8                 INY
088B- D0 EA              BNE .5       ...ALWAYS
088D- 60          .7     RTS
                  *-----------------------------------
                  *      PRINT CHARACTER
                  *-----------------------------------
                  PRINT.CHAR
088E- 48                 PHA
088F- 8C 9C 08           STY PC.SAVEY
0892- 09 80              ORA #$80
0894- 20 ED FD           JSR MON.COUT
0897- AC 9C 08           LDY PC.SAVEY
089A- 68                 PLA
089B- 60                 RTS
089C-             PC.SAVEY .BS 1
                  *-----------------------------------
                  *      PRINT A QUOTATION
                  *-----------------------------------
                  PRINT.QUOTE.NEXT
089D- C8                 INY
                  PRINT.QUOTE
089E- B9 45 08           LDA QTS,Y
08A1- 20 8E 08           JSR PRINT.CHAR
08A4- 10 F7              BPL PRINT.QUOTE.NEXT
08A6- 60                 RTS
                  *-----------------------------------
                  *      CLOSE ALL FILES
                  *-----------------------------------
                  CLOSE.FILE
08A7- 20 8E FD           JSR MON.CROUT
08AA- A9 84              LDA #$84
08AC- 20 8E 08           JSR PRINT.CHAR CONTROL-D
08AF- A0 00              LDY #QCLOSE-QTS
08B1- 4C 9E 08           JMP PRINT.QUOTE
                  *-----------------------------------
                  *      LIST SOURCE PROGRAM
                  *-----------------------------------
                  ASM.LIST
08B4- A5 CA              LDA PP
08B6- 85 01              STA SRCP
08B8- A5 CB              LDA PP+1
08BA- 85 02              STA SRCP+1
08BC- A5 01       .1     LDA SRCP
08BE- C5 4C              CMP HIMEM
08C0- A5 02              LDA SRCP+1
08C2- E5 4D              SBC HIMEM+1
08C4- B0 06              BCS .2       FINISHED
08C6- 20 CD 08           JSR ASM.LIST.LINE
08C9- 4C BC 08           JMP .1
08CC- 60          .2     RTS
                  *-----------------------------------
                  *      LIST ONE SOURCE LINE
                  *-----------------------------------
                  ASM.LIST.LINE
```

```
08CD- 20 EC 08  2500          JSR GNB          SKIP OVER BYTE COUNT
08D0- 20 EC 08  2510          JSR GNB          GET LINE NUMBER
08D3- 85 03     2520          STA LINNUM
08D5- 20 EC 08  2530          JSR GNB
08D8- 85 04     2540          STA LINNUM+1
08DA- 20 0F 09  2550          JSR PRINT.LINNUM
08DD- A9 20     2560          LDA #'       BLANK
08DF- 20 8E 08  2570 .1       JSR PRINT.CHAR
08E2- 20 F9 08  2580          JSR NEXT.TOKEN
08E5- C9 00     2590          CMP #0
08E7- D0 F6     2600          BNE .1
08E9- 4C 8E FD  2610          JMP MON.CROUT
                2620 *-------------------------------
08EC- A0 00     2630 GNB      LDY #0
08EE- 84 00     2640          STY BLANK.COUNT
08F0- B1 01     2650          LDA (SRCP),Y
08F2- E6 01     2660 GNBI     INC SRCP
08F4- D0 02     2670          BNE .1
08F6- E6 02     2680          INC SRCP+1
08F8- 60        2690 .1       RTS
                2700 *-------------------------------
                2710 NEXT.TOKEN
08F9- A0 00     2720          LDY #0
08FB- A5 00     2730          LDA BLANK.COUNT
08FD- D0 0B     2740          BNE .1
08FF- B1 01     2750          LDA (SRCP),Y
0901- 10 EF     2760          BPL GNBI
0903- 29 7F     2770          AND #$7F
0905- 85 00     2780          STA BLANK.COUNT
0907- 20 F2 08  2790          JSR GNBI
090A- C6 00     2800 .1       DEC BLANK.COUNT
090C- A9 20     2810          LDA #'       BLANK
090E- 60        2820          RTS
                2830 *-------------------------------
                2840 *        PRINT LINE NUMBER
                2850 *-------------------------------
                2860 PRINT.LINNUM
090F- A2 03     2870          LDX #3           PRINT 4 DIGITS
0911- A9 30     2880 .3       LDA #'0          SET DIGIT TO ASCII ZERO
0913- 48        2890 .1       PHA              PUSH DIGIT ON STACK
0914- 38        2900          SEC              SUBTRACT CURRENT DIVISOR
0915- A5 03     2910          LDA LINNUM
0917- FD 35 09  2920          SBC PLNTBL,X
091A- 48        2930          PHA              SAVE BYTE ON STACK
091B- A5 04     2940          LDA LINNUM+1
091D- FD 39 09  2950          SBC PLNTBH,X
0920- 90 0A     2960          BCC .2           LESS THAN DIVISOR
0922- 85 04     2970          STA LINNUM+1
0924- 68        2980          PLA              GET LOW BYTE OFF STACK
0925- 85 03     2990          STA LINNUM
0927- 68        3000          PLA              GET DIGIT FROM STACK
0928- 69 00     3010          ADC #0           INCREMENT DIGIT
092A- D0 E7     3020          BNE .1           ...ALWAYS
092C- 68        3030 .2       PLA              DISCARD BYTE FROM STACK
092D- 68        3040          PLA              GET DIGIT FROM STACK
092E- 20 8E 08  3050          JSR PRINT.CHAR
0931- CA        3060          DEX              NEXT DIGIT
0932- 10 DB     3070          BPL .3
0934- 60        3080          RTS              RETURN
                3090 *-------------------------------
0935- 01        3100 PLNTBL   .DA #1
0936- 0A        3110          .DA #10
0937- 64        3120          .DA #100
0938- E8        3130          .DA #1000
0939- 00        3140 PLNTBH   .DA /1
093A- 00        3150          .DA /10
093B- 00        3160          .DA /100
093C- 03        3170          .DA /1000
```

A Use for the USR Command

The S-C ASSEMBLER II Version 4.0 has one user-programmable command, called "USR". (The Quick Reference Card spells it erroneously "USEr".) One good use for it is to re-print the current symbol table.

After an assembly, if the listing was not printed, it is often desirable to be able to see what the spelling or value of a symbol or group of symbols is. If the VAL command is not enough for you, then the following steps will set up the USR command to re-list the symbol table on the screen. And, if your printer is selected, it will also print there.

Get into the assembler, by using BRUN ASMDISK 4.0 from either Applesoft or Integer BASIC. Type "$1E4EL" after the prompt. The first two lines listed should be "LDY #$02" and "STY $E1". If they are not, you have a different version. (It is still version 4.0, but slightly different.) The "LDY #$02" line is the first instruction of the symbol table printing subroutine.

Patch the USR vector by typing "$1007:4E 1E", and then BSAVE the result like this:

```
:BSAVE ASMDISK 4.0 (WITH USR),A$1000,L$14FB
```

This new version, whenever you type "USR", will print out the current symbol table. It will look exactly the same as the symbol table printed out at the end of an assembly.

A Simulated Numeric Key-Pad

This little program will turn part of your Apple's keyboard into a simulated numeric key-pad. A lot cheaper than buying a real one! It is set up to run in page 3, and assumes you are using DOS. If not, just change line 1120 to an RTS.

If you BRUN it or CALL it at 768, the input vector is patched to input all characters through the NKP program. Typing a control-S will toggle the numeric key-pad translator on and off. When the translator is off, all keyboard action is normal, except that another control-S will turn it back on again. When the translator is on, all keys which are not part of the simulated key-pad will input normally.

The keys translated by the simulator are listed in line 1390. The slash key duplicates RETURN, because it is easier to hit when you are entering a lot of numbers. For the same reason, the L-key duplicates "-", in case you are in a hurry to enter negative numbers too. The space bar is used for "0". I set it up to use "NM," for "123", "HJK" for "456", and "YUI" for "789". You should be able to easily change tese translations to any other combination, by changing lines 1390 thru 1420.

The heart of the translator is the search loop in lines 1240 thru 1280. If the input character is not found in CHRTBL, the search loop drops out and the character is not changed. If the character is found, line 1310 picks up the alias for the key, and returns. That's all there is to it!

```
1000 *--------------------------------
1010 *     NUMERIC KEY PAD FOR APPLE
1020 *--------------------------------
1030        .OR $300
1040        .TF B.NKP
1050 *--------------------------------
1060        LDA #1
1070        STA TOGGLE
1080        LDA #NKP
1090        STA $38
1100        LDA /NKP
1110        STA $39
1120        JMP $3EA
1130 *--------------------------------
1140 TOGGLE .BS 1
1150 SAVEY  .BS 1
1160 *--------------------------------
1170 NKP
1180        JSR $FD1B
1190        CMP #$93     CONTROL-S
1200        BEQ .4
1210        BIT TOGGLE
1220        BMI .2       NOT IN NUMERIC MODE
1230        STY SAVEY
1240        LDY #TBLSIZ-1
1250 .1     CMP CHRTBL,Y
1260        BEQ .3       FOUND IN TABLE
1270        DEY
1280        BPL .1
1290        LDY SAVEY
1300 .2     RTS
1310 .3     LDA ALIAS,Y
1320        LDY SAVEY
1330        RTS
1340 .4     LDA TOGGLE
1350        EOR #$80
1360        STA TOGGLE
1370        JMP $FD0C
1380 *--------------------------------
1390 CHRTBL .AS -"/L NM,HJKYUI"
1400 TBLSIZ .EQ *-CHRTBL
1410 ALIAS  .HS 8D
1420        .AS -"-0123456789"
1430 *--------------------------------
```

Apple Assembly Line is published monthly by S-C SOFTWARE, P. O. Box 5537, Richardson, TX 75080. Subscription rate is $12/year, in the U.S.A., Canada, and Mexico. Other countries add $6/year for extra postage. All material herein is copyrighted by S-C SOFTWARE, all rights reserved. Unless otherwise indicated, all material herein is authored by Bob Sander-Cederlof. (Apple is a registered trademark of Apple Computer, Inc.)