

!A

LLOAD RANA.L,A\$4000

*** End of Pass 1

LLOAD RANA1.L,A\$4000

LLOAD RANA2.L,A\$4000

LLOAD RANA3.L,A\$4000

LLOAD RANA4.L,A\$4000

LLOAD RANA5.L,A\$4000

LLOAD RANA.L,A\$4000

*** End of Pass 2

```
0800      1      ttl "Rana ROM Code, RANA.L"
0800      2      src "RANA.L"
0800      3      ;
0800      4      ;
0800      5      ; RANA.L
0800      6      ;
0800      7      ;
0800      8      ; Rana ROM Code
0800      9      ;
0800     10      ; 2024 February 14
0800     11      ;
0800     12      ;
0800     13      ; DOS 4.5, Build 06
0800     14      ;
0800     15      ; 2024 February 14
0800     16      ;
0800     17      ;
0800     18      ; Start of Source Code: 0x4000
0800     19      ; Start of Symbol List: 0x7800
0800     20      ;
0800     21      ;
0800     22      ; Copyright (c) 2024 February 14 by
0800     23      ; Walland Philip Vrbancic Jr
0800     24      ;
0800     25      ; 6223 East Peabody Street
0800     26      ; Long Beach, California 90808
0800     27      ; Unitied States of America
0800     28      ;
0800     29      ; All Rights Reserved
0800     30      ;
0800     31      ; This software is the confidential and
0800     32      ; proprietary intellectual property of
0800     33      ; Walland Philip Vrbancic Jr
0800     34      ;
0800     35      ;
0000     36      LOC0      epz $00
0800     37      ;
0026     38      BUFRADRZ  epz $26
0026     39      TEMPZ     epz $26
0027     40      TEMP2Z    epz $27
002A     41      CURTRKZ   epz $2A
002B     42      SLOT16Z  epz $2B
002C     43      DRVFLAG   epz $2C
002C     44      ADRDATMK  epz $2C
002C     45      ADRFIELD  epz $2C
002D     46      SECFNDZ   epz $2D
002E     47      TRKFNDZ   epz $2E
002F     48      VOLFNDZ   epz $2F
0800     49      ;
0034     50      PHASE     epz $34
0035     51      SYNCNT    epz $35
003C     52      ROMTEMPZ  epz $3C
003C     53      MOTORTIM  epz $3C
003D     54      ROMSECTR  epz $3D
003E     55      BUFADR2Z  epz $3E
003E     56      ODDBITSZ  epz $3E
003F     57      SECTORZ   epz $3F
0800     58      ;
0040     59      ROMDATA    epz $40
0040     60      TRACKZ     epz $40
```

0041	61	ROMTRACK	epz	\$41
0041	62	VOLUMEZ	epz	\$41
004A	63	IOBADR	epz	\$4A
0800	64	;		
0800	65		enz	
0800	66	;		
0000	67	DEBUG	equ	0
0800	68	;		
0000	69	ZERO	equ	\$00
00FF	70	NEGONE	equ	\$FF
0800	71	;		
0006	72	RANAVRSN	equ	\$06
0006	73	RANABLD	equ	\$06
0800	74	;		
0045	75	DOS4VRSN	equ	\$45
0006	76	DOS4BLD	equ	\$06
0800	77	;		
0007	78	SLOTMASK	equ	\$07
000F	79	SECMASK	equ	\$0F
003F	80	TRKMASK	equ	\$3F
0800	81	;		
0003	82	MAXDRIVE	equ	3
0004	83	DFLTPHAS	equ	4
0010	84	PHASMAX	equ	16
0030	85	MAXTRACK	equ	48
0800	86	;		
0006	87	HDRSYNC	equ	6
0008	88	MINSYNC	equ	8
0020	89	MAXSYNC	equ	32
0020	90	MAXRETRY	equ	32
0500	91	SYNCBITS	equ	MAXSYNC*40
0800	92	;		
0010	93	HIGHSECS	equ	\$10
0020	94	MAXSEC	equ	\$20
0800	95	;		
0010	96	ENDTRK45	equ	\$10
001C	97	ENDSEC45	equ	\$1C
0800	98	;		
D8EF	99	MOTONTIM	equ	!-10000-1
0800	100	;		
0000	101	TBLTYPE	equ	\$00
0001	102	SNUM16	equ	\$01
0002	103	DNUM	equ	\$02
0003	104	VOLEXPT	equ	\$03
0004	105	TNUM	equ	\$04
0005	106	SNUM	equ	\$05
0008	107	USRBUF	equ	\$08
000A	108	IOCBPHAS	equ	\$0A
000B	109	BYTCNT	equ	\$0B
000C	110	CMDCODE	equ	\$0C
000D	111	ERRCODE	equ	\$0D
000E	112	VOLFND	equ	\$0E
000F	113	SLOTFND	equ	\$0F
0010	114	DRVFND	equ	\$10
0800	115	;		
0000	116	RWTSSEEK	equ	\$00
0001	117	RWTSREAD	equ	\$01
0002	118	RWTSWRIT	equ	\$02
0004	119	RWTSFRMT	equ	\$04
0800	120	;		
0000	121	RWNOERR	equ	\$00

```

0008      122  RWINITER equ $08
0010      123  RWPROTER equ $10
0020      124  RWVOLERR equ $20
0030      125  RWSYNERR equ $30
0040      126  RWDRVERR equ $40
0080      127  RWREADER equ $80
0800      128  ;
0056      129  NBUF2SIZ equ $56
00AA      130  ODDBITS  equ $AA
0800      131  ;
00D5      132  ADRMARK1 equ $D5
00AA      133  ADRMARK2 equ $AA
0096      134  ADRMARK3 equ $96
0800      135  ;
00D5      136  DATMARK1 equ $D5
00AA      137  DATMARK2 equ $AA
00AD      138  DATMARK3 equ $AD
0800      139  ;
00DE      140  SLPMARK1 equ $DE
00AA      141  SLPMARK2 equ $AA
00EB      142  SLPMARK3 equ $EB
0800      143  ;
00FF      144  SYNCMARK equ $FF
0800      145  ;
0100      146  PAGESIZE equ $100
0100      147  STACK    equ $100
0800      148  ;
0112      149  SAVYREG  equ $112
0113      150  SECTOR   equ $113
0114      151  NBUFLAG  equ $114
0115      152  DNUM0    equ $115
0800      153  ;
0116      154  ENDTRK    equ $116      ; from DOS 4.X INITVALS
0117      155  ENDSEC    equ $117      ; from DOS 4.X INITVALS
0118      156  STRTSEC   equ $118
0119      157  STOPSEC   equ $119
0800      158  ;
011A      159  HOOKCODE  equ $11A      ; 6 bytes
0120      160  SECMAP    equ $120      ; 32 bytes
0800      161  ;
0300      162  NBUF2BT   equ $300
0800      163  ;
03D0      164  DOSWARM   equ $3D0
03EA      165  HOOKDOS    equ $3EA
0800      166  ;
0478      167  FINDTRK   equ $478
04F8      168  RECALCNT  equ $4F8
0800      169  ;
04FB      170  XMODE     equ $4FB
0800      171  ;
0578      172  SEEKCNT   equ $578
05F8      173  RETRYCNT  equ $5F8
0800      174  ;
0678      175  NEXTON    equ $678
06F8      176  NEXTOFF   equ $6F8
0800      177  ;
0778      178  SLOT16    equ $778
07F8      179  MSLOT     equ $7F8      ; set MSB if use 0xC800 space
0800      180  ;
0800      181  ;
0800      182  ; DRV.TRK, DRV.PHAS variables are indexed by slot number.

```

```

0800      183      ;
0478      184      DRV0TRK      equ      $478      ; drive 1 track
04F8      185      DRV1TRK      equ      $4F8      ; drive 2 track
0578      186      DRV2TRK      equ      $578      ; drive 3 track
05F8      187      DRV3TRK      equ      $5F8      ; drive 4 track
0800      188      ;
0678      189      DRV0PHAS      equ      $678      ; drive 1 phase
06F8      190      DRV1PHAS      equ      $6F8      ; drive 2 phase
0778      191      DRV2PHAS      equ      $778      ; drive 3 phase
07F8      192      DRV3PHAS      equ      $7F8      ; drive 4 phase
0800      193      ;
0800      194      ;
08FE      195      BOOTADR      equ      $8FE
08FF      196      BOOTPGS      equ      $8FF
0800      197      ;
0800      198      PAGE08      equ      $0800
1000      199      PAGE10      equ      $1000
2000      200      PAGE20      equ      $2000
0800      201      ;
9D00      202      NBUF1L      equ      $9D00
9E00      203      NBUF2L      equ      $9E00
0800      204      ;
DE00      205      NBUF1H      equ      $DE00
DF00      206      NBUF2H      equ      $DF00
0800      207      ;
BFF0      208      BLDVRSN      equ      $BFF0
BFF1      209      BLDNMBR      equ      $BFF1
0800      210      ;
BFF2      211      MNGDISK      equ      $BFF2
BFFA      212      INITVAL      equ      $BFFA
BFFD      213      NBUF1PG      equ      $BFFD
0800      214      ;
C082      215      ROM2WP      equ      $C082
C08B      216      RAM1WE      equ      $C08B
0800      217      ;
C080      218      PHASEOFF      equ      $C080
C081      219      PHASEON      equ      $C081
C088      220      MOTOROFF      equ      $C088
C089      221      MOTORON      equ      $C089
C08A      222      DRV0EN      equ      $C08A
C08B      223      DRV1EN      equ      $C08B
C08C      224      STROBE      equ      $C08C
C08D      225      LATCH      equ      $C08D
C08E      226      DATAIN      equ      $C08E
C08F      227      DATAOUT      equ      $C08F
0800      228      ;
C000      229      SLOTROM0      equ      $C000
C800      230      SLOTROM8      equ      $C800
0800      231      ;
CFFF      232      CLRROM      equ      $CFFF
0800      233      ;
D003      234      DISKADRS      equ      $D003
0800      235      ;
FCA8      236      WAIT      equ      $FCA8
FF58      237      IORTS      equ      $FF58
0800      238      ;
0800      239      ;
0800      240      icl      "RANA1.L"

```

```

LLOAD RANA1.L,A$4000

```

```

0800          1          ttl "Rana ROM Code, RANA1.L"
0800          2          ;
0800          3          ;
0800          4          ; RANA1.L
0800          5          ;
0800          6          ;
0800          7          .if DEBUG
0800          8          org PAGE20
0800          9          obj PAGE20
0800         10          .el
C800         11          org SLOTROM8
C800         12          obj PAGE10
C800         13          .fi
C800         14          ;
C800         15          usr
C800         16          ;
C800         17          ;
C800         18          .if DEBUG
C800         19          ;
C800         20          ; Install Rana driver into DOS 4.5.
C800         21          ;
C800         22          bit RAM1WE
C800         23          bit RAM1WE
C800         24          ;
C800         25          lda #RANARWTS
C800         26          sta DISKADRS+2*6-1
C800         27          ;
C800         28          lda /RANARWTS
C800         29          sta DISKADRS+1+2*6-1
C800         30          ;
C800         31          bit ROM2WP
C800         32          ;
C800         33          jmp DOSWARM
C800         34          ;
C800         35          ;
C800         36          dfs PAGE SIZE-*&NEGONE,ZERO
C800         37          ;
C800         38          ;
C800         39          .fi
C800         40          ;
C800         41          ;
C800         42          ; Rana hardware selects drives 1 and 2 when ROMCODE is
C800         43          ; written or drives 3 and 4 when ROMCODE+1 is written.
C800         44          ;
C800 00         45  ROMCODE  hex 00
C801 00         46          hex 00
C802         47          ;
C802         48          ;
C802         49          ; Extract the slot page from the return address on the
C802         50          ; stack. Fall into SELCHEAD for track 0. Protect Y-reg.
C802         51          ;
C802 BA        52  GETSLOT  tsx
C803         53          ;
C803         54          .if DEBUG
C803         55          lda #$C6
C803         56          nop
C803         57          .el
C803 BD 02 01   58          lda STACK+2,X
C806         59          .fi
C806         60          ;

```

```

C806 8D F8 07      61      sta MSLOT
C809              62      ;
C809 29 07        63      and #SLOTMASK
C80B              64      ;
C80B 0A          65      asl
C80C 0A          66      asl
C80D 0A          67      asl
C80E 0A          68      asl
C80F              69      ;
C80F 85 2B        70      sta SLOT16Z
C811              71      ;
C811              72      ;
C811              73      ; Always select the lower head.
C811              74      ;
C811 A6 2B        75      SELCHEAD ldx SLOT16Z
C813              76      ;
C813 BD 81 C0     77      lda PHASEON,X
C816 BD 85 C0     78      lda PHASEON+4,X
C819              79      ;
C819 20 C0 C8     80      jsr WAIT52
C81C              81      ;
C81C BD 80 C0     82      lda PHASEOFF,X
C81F BD 84 C0     83      lda PHASEOFF+4,X
C822              84      ;
C822 18           85      clc
C823              86      ;
C823 60           87      rts
C824              88      ;
C824              89      ;
C824              90      ; Routine to save the half-phase value of a drive in a
C824              91      ; slot dependant location.
C824              92      ;
C824              93      ; Do not modify the TRK, DRV, or PHAS table addresses.
C824              94      ;
C824              95      ; First calculate the half-phase value based on track
C824              96      ; and VALSPHAS. Add PHASE/2 if SECTOR > 0x0F.
C824              97      ;
C824 85 26        98      SAVETRK sta TEMPZ
C826              99      ;
C826 A9 00        100     lda #ZERO
C828              101     ;
C828 AC 15 01     102     ldy DNUM0
C82B C0 03        103     cpy #MAXDRIVE
C82D 90 0B        104     bcc >1
C82F              105     ;
C82F AC 13 01     106     ldy SECTOR
C832 C0 10        107     cpy #HIGHSECS
C834 90 04        108     bcc >1
C836              109     ;
C836 A5 34        110     lda PHASE
C838 4A           111     lsr
C839              112     ;
C839 18           113     clc
C83A              114     ;
C83A A4 34        115     ^1 ldy PHASE
C83C              116     ;
C83C 65 26        117     ^2 adc TEMPZ
C83E              118     ;
C83E 88           119     dey
C83F D0 FB        120     bne <2
C841              121     ;

```

```

C841 8D 78 04    122 SAVETRK2 sta FINDTRK
C844            123 ;
C844            124 ;
C844            125 ; Access the address based on slot and drive number.
C844            126 ;
C844 A2 04      127         ldx /DRV0TRK
C846 20 17 CC   128         jsr DRVINDEX
C849            129 ;
C849 B1 26      130         lda (TEMPZ),Y
C84B 85 2A      131         sta CURTRKZ
C84D            132 ;
C84D AD 78 04   133         lda FINDTRK
C850 91 26      134         sta (TEMPZ),Y
C852            135 ;
C852 60         136         rts
C853            137 ;
C853            138 ;
C853            139         dfs NBUF2SIZ-*)&NEGONE,ZERO
C856            140 ;
C856            141 ;
C856            142 ; Write translate table.
C856            143 ;
C856 96 97 9A   144 WRNIBL    hex 96979A9B9D9E9FA6
C859 9B 9D 9E
C85C 9F A6
C85E A7 AB AC   145         hex A7ABACADAEAFB2B3
C861 AD AE AF
C864 B2 B3
C866 B4 B5 B6   146         hex B4B5B6B7B9BABBBC
C869 B7 B9 BA
C86C BB BC
C86E BD BE BF   147         hex BDBEBFCBCDCECFD3
C871 CB CD CE
C874 CF D3
C876 D6 D7 D9   148         hex D6D7D9DADBDCDDDE
C879 DA DB DC
C87C DD DE
C87E DF E5 E6   149         hex DFE5E6E7E9EAEBEC
C881 E7 E9 EA
C884 EB EC
C886 ED EE EF   150         hex EDEEEFF2F3F4F5F6
C889 F2 F3 F4
C88C F5 F6
C88E F7 F9 FA   151         hex F7F9FAFBFCFDFF
C891 FB FC FD
C894 FE FF
C896            152 ;
C896            153 ;
C896            154 ; Read translate table.
C896            155 ;
C896            156 ; Allocate 9 bytes from this table for WAIT routines.
C896            157 ;
C896 00 01      158 RDNIBL    hex 0001
C898 00 00 02   159         hex 0000020300040506
C89B 03 00 04
C89E 05 06
C8A0 00 00 00   160         hex 0000000000000708
C8A3 00 00 00
C8A6 07 08
C8A8 00 00 00   161         hex 000000090A0B0C0D
C8AB 09 0A 0B

```

```

C8AE 0C 0D
C8B0 00 00 0E    162          hex 00000E0F10111213
C8B3 0F 10 11
C8B6 12 13
C8B8 00 14 15    163          hex 001415161718191A
C8BB 16 17 18
C8BE 19 1A
C8C0          164      ;
C8C0          165      ;          hex 0000000000000000
C8C0          166      ;          hex 0000001B001C1D1E
C8C0          167      ;
C8C0 EA        168 WAIT52      nop
C8C1 EA        169          nop
C8C2 20 C5 C8   170 WAIT48      jsr WAIT24
C8C5 20 C8 C8   171 WAIT24      jsr WAIT12
C8C8 60         172 WAIT12      rts
C8C9          173      ;
C8C9 00 00 1B   174          hex 00001B001C1D1E
C8CC 00 1C 1D
C8CF 1E
C8D0          175      ;
C8D0 00 00 00   176          hex 0000001F00002021
C8D3 1F 00 00
C8D6 20 21
C8D8 00 22 23   177          hex 0022232425262728
C8DB 24 25 26
C8DE 27 28
C8E0 00 00 00   178          hex 0000000000292A2B
C8E3 00 00 29
C8E6 2A 2B
C8E8 00 2C 2D   179          hex 002C2D2E2F303132
C8EB 2E 2F 30
C8EE 31 32
C8F0 00 00 33   180          hex 0000333435363738
C8F3 34 35 36
C8F6 37 38
C8F8 00 39 3A   181          hex 00393A3B3C3D3E3F
C8FB 3B 3C 3D
C8FE 3E 3F
C900          182      ;
C900          183      ;
C900          184          icl "RANA2.L"

```

```

LLOAD RANA2.L,A$4000

```

```

C900          1          ttl "Rana ROM Source Code, RANA2.L"
C900          2          ;
C900          3          ;
C900          4          ; RANA2.L
C900          5          ;
C900          6          ;
C900          7          ; These routines are all time critical.  Be carefull with
C900          8          ; page boundries and do not change any instructions.
C900          9          ;
C900         10          ; Prepare TEMPZ for later and write HDRSYNC sync bytes and
C900         11          ; data marks 0xD5, 0xAA, and 0xAD to disk.
C900         12          ;
C900 AC 00 DF  13 WRITBUFH ldy NBUF2H
C903 20 04 CA  14          jsr WRITSYN0
C906          15          ;
C906          16          ;
C906          17          ; Checksum is cleared by writing the last byte of NBUF2.
C906          18          ; Apple originally published this routine using a 36 usec
C906          19          ; entrance.  I have changed this logic.
C906          20          ;
C906          21          ; Use NBUF1H and NBUF2H buffers.
C906          22          ;
C906 AE 55 DF  23          ldx NBUF2H+NBUF2SIZ-1
C909          24          ;
C909 A0 55     25          ldy #NBUF2SIZ-1
C90B D0 07     26          bne >2          ; always taken
C90D          27          ;
C90D          28          ;
C90D          29          ; Get prior 6-bit nibble and XOR with current nibble to
C90D          30          ; form index into the write translate table.
C90D          31          ;
C90D B9 01 DF  32 ^1      lda NBUF2H+1,Y
C910          33          ;
C910 59 00 DF  34          eor NBUF2H,Y
C913 AA       35          tax
C914          36          ;
C914 BD 56 C8  37 ^2      lda WRNIBL,X
C917          38          ;
C917 AE 78 07  39          ldx SLOT16
C91A          40          ;
C91A 9D 8D C0  41          sta LATCH,X
C91D BD 8C C0  42          lda STROBE,X
C920          43          ;
C920 88       44          dey
C921 10 EA     45          bpl <1
C923          46          ;
C923          47          ;
C923          48          ; Write NBUF1H to disk using the same logic.
C923          49          ;
C923 C8       50          iny
C924          51          ;
C924 A5 26     52          lda TEMPZ
C926          53          ;
C926 59 00 DE  54 ^3      eor NBUF1H,Y
C929 AA       55          tax
C92A          56          ;
C92A BD 56 C8  57          lda WRNIBL,X
C92D          58          ;
C92D AE 78 07  59          ldx SLOT16
C930          60          ;

```

```

C930 9D 8D C0      61          sta LATCH,X
C933 BD 8C C0      62          lda STROBE,X
C936              63          ;
C936 B9 00 DE      64          lda NBUF1H,Y
C939              65          ;
C939 C8            66          iny
C93A D0 EA        67          bne <3
C93C              68          ;
C93C F0 4E        69          beq WRITRTN          ; always taken
C93E              70          ;
C93E              71          ;
C93E              72          ; Write a sector data field routine. A sector data field
C93E              73          ; consists of pre-nibbled data contained in NBUF1H and
C93E              74          ; NBUF2H. Check write protect sense from disk controller.
C93E              75          ;
C93E 38           76          WRITSCTR sec
C93F              77          ;
C93F BD 8D C0      78          lda LATCH,X
C942 BD 8E C0      79          lda DATAIN,X
C945              80          ;
C945 30 68         81          bmi SETREAD
C947              82          ;
C947              83          ;
C947              84          ; Check NBUFLAG for routine and buffers for WRITSCTR.
C947              85          ;
C947 A9 AD         86          lda #DATMARK3
C949              87          ;
C949 2C 14 01      88          bit NBUFLAG
C94C 30 B2         89          bmi WRITBUFH
C94E              90          ;
C94E              91          ;
C94E              92          ; Prepare TEMPZ for later and write HDRSYNC sync bytes and
C94E              93          ; data marks 0xD5, 0xAA, and 0xAD to disk.
C94E              94          ;
C94E AC 00 9E      95          ldy NBUF2L
C951 20 04 CA      96          jsr WRITSYN0
C954              97          ;
C954              98          ;
C954              99          ; Checksum is cleared by writing the last byte of NBUF2.
C954            100          ; Apple originally published this routine using a 36 usec
C954            101          ; entrance. I have changed this logic.
C954            102          ;
C954            103          ; Use NBUF1L and NBUF2L buffers.
C954            104          ;
C954 AE 55 9E      105          ldx NBUF2L+NBUF2SIZ-1
C957              106          ;
C957 A0 55         107          ldy #NBUF2SIZ-1
C959 D0 07         108          bne >2          ; always taken
C95B              109          ;
C95B              110          ;
C95B            111          ; Get prior 6-bit nibble and XOR with current nibble to
C95B            112          ; form index into the write translate table.
C95B            113          ;
C95B B9 01 9E      114          ^1      lda NBUF2L+1,Y
C95E              115          ;
C95E 59 00 9E      116          eor NBUF2L,Y
C961 AA           117          tax
C962              118          ;
C962 BD 56 C8      119          ^2      lda WRNIBL,X
C965              120          ;
C965 AE 78 07      121          ldx SLOT16

```

```

C968      122 ;
C968 9D 8D C0 123      sta LATCH,X
C96B BD 8C C0 124      lda STROBE,X
C96E      125 ;
C96E 88      126      dey
C96F 10 EA    127      bpl <1
C971      128 ;
C971      129 ;
C971      130 ; Write NBUF1L to disk using the same logic.
C971      131 ;
C971 C8      132      iny
C972      133 ;
C972 A5 26    134      lda TEMPZ
C974      135 ;
C974 59 00 9D 136 ^3    eor NBUF1L,Y
C977 AA      137      tax
C978      138 ;
C978 BD 56 C8 139      lda WRNIBL,X
C97B      140 ;
C97B AE 78 07 141      ldx SLOT16
C97E      142 ;
C97E 9D 8D C0 143      sta LATCH,X
C981 BD 8C C0 144      lda STROBE,X
C984      145 ;
C984 B9 00 9D 146      lda NBUF1L,Y
C987      147 ;
C987 C8      148      iny
C988 D0 EA    149      bne <3
C98A      150 ;
C98A 24 26    151      bit TEMPZ
C98C      152 ;
C98C      153 ;
C98C      154 ; Write the checksum to disk.
C98C      155 ;
C98C A8      156 WRITRTN tay
C98D      157 ;
C98D B9 56 C8 158      lda WRNIBL,Y
C990      159 ;
C990 20 42 CA 160      jsr WNIBL
C993      161 ;
C993 24 26    162      bit TEMPZ
C995      163 ;
C995      164 ;
C995      165 ; Write slip marks 0xDE, 0xAA, and 0xEB to disk.
C995      166 ;
C995      167 ;
C995 EA      168 WRITEEXIT nop
C996 EA      169      nop
C997      170 ;
C997 A9 DE    171      lda #SLPMARK1
C999 20 41 CA 172      jsr WNIBL2
C99C      173 ;
C99C A9 AA    174      lda #SLPMARK2
C99E 20 3F CA 175      jsr WNIBL9
C9A1      176 ;
C9A1 A9 EB    177      lda #SLPMARK3
C9A3 20 3F CA 178      jsr WNIBL9
C9A6      179 ;
C9A6      180 ;
C9A6      181 ; Terminate the address/data footer.
C9A6      182 ;

```

```

C9A6 A9 FF      183      lda #SYNCMARK
C9A8 20 3F CA    184      jsr WNIBL9
C9AB              185      ;
C9AB 20 C5 C8    186      jsr WAIT24          ; allow all bits to be written
C9AE EA          187      nop
C9AF              188      ;
C9AF              189      ;
C9AF              190      ; Return to read mode.
C9AF              191      ;
C9AF BD 8E C0    192      SETREAD  lda DATAIN,X
C9B2 BD 8C C0    193      lda STROBE,X
C9B5              194      ;
C9B5 60          195      rts
C9B6              196      ;
C9B6              197      ;
C9B6              198      ; Write an address data field routine.  An address data
C9B6              199      ; field consists of the volume, track, sector, and checksum
C9B6              200      ; values.  Check write protect sense from the Disk ][.
C9B6              201      ;
C9B6 38          202      WRITADR  sec
C9B7              203      ;
C9B7 A6 2B       204      ldx SLOT16Z
C9B9              205      ;
C9B9 BD 8D C0    206      lda LATCH,X
C9BC BD 8E C0    207      lda DATAIN,X
C9BF              208      ;
C9BF 30 EE       209      bmi SETREAD
C9C1              210      ;
C9C1              211      ;
C9C1              212      ; Prepare TEMPZ for later with the address header checksum
C9C1              213      ; and write Y-reg number of sync bytes.  Then write data
C9C1              214      ; marks 0xD5, 0xAA, and 0x96 to disk.
C9C1              215      ;
C9C1 A5 41       216      lda VOLUMEZ
C9C3 45 40       217      eor TRACKZ
C9C5 45 3F       218      eor SECTORZ
C9C7 85 26       219      sta TEMPZ
C9C9              220      ;
C9C9 A9 96       221      lda #ADRMARK3
C9CB 20 08 CA    222      jsr WRITSYNC
C9CE              223      ;
C9CE              224      ;
C9CE              225      ; Write volume, track, sector, and checksum to disk.
C9CE              226      ;
C9CE A5 41       227      lda VOLUMEZ
C9D0 20 30 CA    228      jsr WBYTE
C9D3              229      ;
C9D3 A5 40       230      lda TRACKZ
C9D5 20 30 CA    231      jsr WBYTE
C9D8              232      ;
C9D8 A5 3F       233      lda SECTORZ
C9DA 20 30 CA    234      jsr WBYTE
C9DD              235      ;
C9DD A5 26       236      lda TEMPZ
C9DF 20 30 CA    237      jsr WBYTE
C9E2              238      ;
C9E2 90 B1       239      bcc WRITEXIT          ; always taken
C9E4              240      ;
C9E4              241      ;
C9E4              242      ; Sector interleave remapping table to support 32 sectors
C9E4              243      ; per track.

```

```

C9E4          244 ;
C9E4 00 0D 0B 245 INTRLEAV hex 000D0B0907050301
C9E7 09 07 05
C9EA 03 01
C9EC 0E 0C 0A 246          hex 0E0C0A080604020F
C9EF 08 06 04
C9F2 02 0F
C9F4 10 1D 1B 247          hex 101D1B1917151311
C9F7 19 17 15
C9FA 13 11
C9FC 1E 1C 1A 248          hex 1E1C1A181614121F
C9FF 18 16 14
CA02 12 1F
CA04          249 ;
CA04          250 ;
CA04          251 ; Y-reg contains the number of sync bytes to write.
CA04          252 ;
CA04          253 ; Configure firmware to generate sync bytes. Sync bytes
CA04          254 ; are 40 usecs in length and are written as %1111111100.
CA04          255 ; The first sync byte must be written exactly 40 usecs
CA04          256 ; later and continually written in a 40 usec loop.
CA04          257 ;
CA04 84 26      258 WRITSYN0 sty TEMPZ
CA06          259 ;
CA06 A0 06      260          ldy #HDRSYNC
CA08          261 ;
CA08 85 2C      262 WRITSYNC sta ADRDATMK
CA0A          263 ;
CA0A A9 FF      264          lda #SYNCMARK
CA0C          265 ;
CA0C 9D 8F C0   266          sta DATAOUT,X
CA0F 1D 8C C0   267          ora STROBE,X
CA12          268 ;
CA12 48          269          pha
CA13 68          270          pla
CA14          271 ;
CA14 20 C5 C8   272 ^1          jsr WAIT24
CA17          273 ;
CA17 9D 8D C0   274          sta LATCH,X
CA1A 1D 8C C0   275          ora STROBE,X
CA1D          276 ;
CA1D EA          277          nop
CA1E          278 ;
CA1E 88          279          dey
CA1F D0 F3      280          bne <1
CA21          281 ;
CA21          282 ;
CA21          283 ; Write address/data marks 0xD5, 0xAA, and TEMP2Z to disk.
CA21          284 ;
CA21 A9 D5      285          lda #DATMARK1          ; same as ADRMARK1
CA23 20 3F CA   286          jsr WNIBL9
CA26          287 ;
CA26 A9 AA      288          lda #DATMARK2          ; same as ADRMARK2
CA28 20 3F CA   289          jsr WNIBL9
CA2B          290 ;
CA2B A5 2C      291          lda ADRDATMK          ; recall 3rd address/data mark
CA2D          292 ;
CA2D 18          293          clc
CA2E 90 0F      294          bcc WNIBL9          ; always taken
CA30          295 ;
CA30          296 ;

```

```

CA30          297 ; Write a byte as two four-bit nibbles to the disk
CA30          298 ; starting with the odd bits.
CA30          299 ;
CA30 48       300 WBYTE      pha
CA31          301 ;
CA31 4A       302          lsr
CA32 05 3E    303          ora ODDBITSZ
CA34          304 ;
CA34 9D 8D C0 305          sta LATCH,X
CA37 BD 8C C0 306          lda STROBE,X
CA3A          307 ;
CA3A          308 ;
CA3A          309 ; Now write the even bits.
CA3A          310 ;
CA3A 68       311          pla
CA3B 05 3E    312          ora ODDBITSZ
CA3D          313 ;
CA3D 48       314          pha
CA3E 68       315          pla
CA3F          316 ;
CA3F          317 ;
CA3F          318 ; Wait 9 clock cycles, then write to disk.
CA3F          319 ;
CA3F 48       320 WNIBL9     pha
CA40 68       321          pla
CA41          322 ;
CA41          323 ;
CA41          324 ; Wait 2 clock cycles, then write to disk.
CA41          325 ;
CA41 18       326 WNIBL2     clc
CA42          327 ;
CA42          328 ;
CA42          329 ; Write nibble to disk.
CA42          330 ;
CA42 9D 8D C0 331 WNIBL      sta LATCH,X
CA45 BD 8C C0 332          lda STROBE,X
CA48          333 ;
CA48 60       334          rts
CA49          335 ;
CA49          336 ;
CA49          337 ; READ routine reads nibblized data from the disk and
CA49          338 ; stores the data in NBUF1L/NBUF2L or NBUF1H/NBUF2H. Fail
CA49          339 ; after N read attempts where N corresponds to the
CA49          340 ; possibility that there may be MAXSYNC sync bytes before
CA49          341 ; the data header.
CA49          342 ;
CA49 A9 56    343 READSCTR  lda #NBUF2SIZ
CA4B 85 26    344          sta TEMPZ
CA4D          345 ;
CA4D A2 AD    346          ldx #DATMARK3          ; for ADRDATMK
CA4F          347 ;
CA4F A0 28    348          ldy #SYNCBITS/32
CA51 A9 00    349          lda #ZERO          ; for TEMP2Z
CA53          350 ;
CA53 20 01 CB 351          jsr READMRKS
CA56 30 55    352          bmi READERR
CA58          353 ;
CA58 2C 14 01 354          bit NBUFLAG
CA5B 30 24    355          bmi READBUFH
CA5D          356 ;
CA5D          357 ;

```

```

CA5D      358 ; Checksum has been initialized to zero.  Read nibbles into
CA5D      359 ; NBUF2L.
CA5D      360 ;
CA5D C6 26 361 ^6      dec TEMPZ
CA5F      362 ;
CA5F BC 8C C0 363 ^7      ldy STROBE,X
CA62 10 FB 364          bpl <7
CA64      365 ;
CA64 59 00 C8 366          eor RDNIBL-$96,Y
CA67      367 ;
CA67 A4 26 368          ldy TEMPZ
CA69      369 ;
CA69 99 00 9E 370          sta NBUF2L,Y
CA6C      371 ;
CA6C D0 EF 372          bne <6
CA6E      373 ;
CA6E      374 ;
CA6E      375 ; Now read nibbles into NBUF1L.
CA6E      376 ;
CA6E BC 8C C0 377 ^8      ldy STROBE,X
CA71 10 FB 378          bpl <8
CA73      379 ;
CA73 59 00 C8 380          eor RDNIBL-$96,Y
CA76      381 ;
CA76 A4 26 382          ldy TEMPZ
CA78      383 ;
CA78 99 00 9D 384          sta NBUF1L,Y
CA7B      385 ;
CA7B E6 26 386          inc TEMPZ
CA7D D0 EF 387          bne <8
CA7F      388 ;
CA7F F0 22 389          beq >9 ; always taken
CA81      390 ;
CA81      391 ;
CA81      392 READBUFH:
CA81      393 ;
CA81      394 ; Checksum has been initialized to zero.  Read nibbles into
CA81      395 ; NBUF2H.
CA81      396 ;
CA81 C6 26 397 ^6      dec TEMPZ
CA83      398 ;
CA83 BC 8C C0 399 ^7      ldy STROBE,X
CA86 10 FB 400          bpl <7
CA88      401 ;
CA88 59 00 C8 402          eor RDNIBL-$96,Y
CA8B      403 ;
CA8B A4 26 404          ldy TEMPZ
CA8D      405 ;
CA8D 99 00 DF 406          sta NBUF2H,Y
CA90      407 ;
CA90 D0 EF 408          bne <6
CA92      409 ;
CA92      410 ;
CA92      411 ; Now read nibbles into NBUF1H.
CA92      412 ;
CA92 BC 8C C0 413 ^8      ldy STROBE,X
CA95 10 FB 414          bpl <8
CA97      415 ;
CA97 59 00 C8 416          eor RDNIBL-$96,Y
CA9A      417 ;
CA9A A4 26 418          ldy TEMPZ

```

```

CA9C          419 ;
CA9C 99 00 DE 420          sta NBUF1H,Y
CA9F          421 ;
CA9F E6 26    422          inc TEMPZ
CAA1 D0 EF    423          bne <8
CAA3          424 ;
CAA3          425 ;
CAA3          426 ; Verify the checksum byte.
CAA3          427 ;
CAA3 BC 8C C0 428 ^9      ldy STROBE,X
CAA6 10 FB    429          bpl <9
CAA8          430 ;
CAA8 D9 00 C8 431          cmp RDNIBL-$96,Y
CAAB F0 28    432          beq READEXIT
CAAD          433 ;
CAAD 38       434 READERR sec
CAAE          435 ;
CAAE 60       436          rts
CAAF          437 ;
CAAF          438 ;
CAAF          439 ; Read address field. Address field nibbles are odd/even
CAAF          440 ; encoded. Read over 0x400 disk nibbles before giving up.
CAAF          441 ;
CAAF A2 96    442 READADR ldx #ADRMARK3          ; for ADRDATMK
CAB1          443 ;
CAB1 A0 04    444          ldy #4
CAB3 98       445          tya          ; for TEMP2Z
CAB4          446 ;
CAB4 20 01 CB 447          jsr READMRKS
CAB7 30 F4    448          bmi READERR
CAB9          449 ;
CAB9          450 ;
CAB9          451 ; Read the four-byte address data field nibbles. First
CAB9          452 ; read the 'odd' bit nibble. The C-flag is set previously
CAB9          453 ; from the ADRDATMK comparison in READMRKS.
CAB9          454 ;
CAB9 85 27    455 ^6      sta TEMP2Z
CABB          456 ;
CABB BD 8C C0 457 ^7      lda STROBE,X
CABE 10 FB    458          bpl <7
CAC0          459 ;
CAC0 2A       460          rol
CAC1 85 26    461          sta TEMPZ
CAC3          462 ;
CAC3          463 ;
CAC3          464 ; Now read the 'even' bit nibble and merge the two nibbles.
CAC3          465 ; Store the data byte, then update the checksum and repeat
CAC3          466 ; until the entire address field is read.
CAC3          467 ;
CAC3 BD 8C C0 468 ^8      lda STROBE,X
CAC6 10 FB    469          bpl <8
CAC8          470 ;
CAC8 25 26    471          and TEMPZ
CACA          472 ;
CACA 99 2C 00 473          sta ADRFIELD,Y
CACD 45 27    474          eor TEMP2Z
CACF          475 ;
CACF 88       476          dey
CAD0 10 E7    477          bpl <6
CAD2          478 ;
CAD2          479 ;

```

```
CAD2          480 ; Checksum in A-reg must be zero for no error.
CAD2          481 ;
CAD2 A8       482          tay
CAD3 D0 D8    483          bne READERR
CAD5          484 ;
CAD5          485 ;
CAD5          486 READEXIT:
CAD5          487 ;
CAD5          488 ; Check for slip mark 1.
CAD5          489 ;
CAD5 BD 8C C0 490 ^1      lda STROBE,X
CAD8 10 FB    491          bpl <1
CADA          492 ;
CADA C9 DE    493          cmp #SLPMARK1
CADC D0 CF    494          bne READERR
CADE          495 ;
CADE EA       496          nop
CADF          497 ;
CADF          498 ;
CADF          499 ; Check for skip mark 2. Slip mark 3 is not checked.
CADF          500 ;
CADF BD 8C C0 501 ^2      lda STROBE,X
CAE2 10 FB    502          bpl <2
CAE4          503 ;
CAE4 C9 AA    504          cmp #SLPMARK2
CAE6 D0 C5    505          bne READERR
CAE8          506 ;
CAE8 18       507          clc
CAE9          508 ;
CAE9 60       509          rts
CAEA          510 ;
CAEA          511 ;
CAEA          512          icl "RANA3.L"
```

LLOAD RANA3.L,A\$4000

```

CAEA          1          ttl "Rana ROM Source Code, RANA3.L"
CAEA          2          ;
CAEA          3          ;
CAEA          4          ; RANA3.L
CAEA          5          ;
CAEA          6          ;
CAEA          7          ; Connect or disconnect the Rana to DOS based on the value
CAEA          8          ; in the Y-reg.  If Y-reg is zero, then connect to DOS.
CAEA          9          ;
CAEA 20 54 CB 10  HOOKRANA jsr DISKMNG
CAED          11         ;
CAED A2 05     12         ldx #EXITLEN-1
CAEF          13         ;
CAEF BD FB CA 14 ^1      lda EXITCODE,X
CAF2 9D 1A 01 15         sta HOOKCODE,X
CAF5          16         ;
CAF5 CA       17         dex
CAF6 10 F7     18         bpl <1
CAF8          19         ;
CAF8 4C 1A 01 20         jmp HOOKCODE
CAFB          21         ;
CAFB          22         ;
CAFB 2C FF CF 23  EXITCODE bit CLRROM
CAFE          24         ;
CAFE 4C EA 03 25         jmp HOOKDOS
CB01          26         ;
0006          27  EXITLEN equ *-EXITCODE
CB01          28         ;
CB01          29         ;
CB01 86 2C     30  READMRKS stx ADDRATMK          ; save address/data mark 3
CB03 85 27     31         sta TEMP2Z              ; save MSB disk nibble counter
CB05          32         ;
CB05 A6 2B     33         ldx SLOT16Z              ; recall SLOT*16 value
CB07          34         ;
CB07 88        35 ^1      dey                      ; LSB disk nibble counter
CB08 D0 04     36         bne >2
CB0A          37         ;
CB0A C6 27     38         dec TEMP2Z                ; MSB disk nibble counter
CB0C 30 20     39         bmi >6
CB0E          40         ;
CB0E          41         ;
CB0E          42         ; Check for address/data mark 1.
CB0E          43         ;
CB0E BD 8C C0 44 ^2      lda STROBE,X              ; read sequencer latch
CB11 10 FB     45         bpl <2
CB13          46         ;
CB13 C9 D5     47 ^3      cmp #ADRMARK1            ; same as DATMARK1
CB15 D0 F0     48         bne <1
CB17          49         ;
CB17 EA       50         nop                      ; waste 2 cycles
CB18          51         ;
CB18          52         ;
CB18          53         ; Check for address/data mark 2.
CB18          54         ;
CB18 BD 8C C0 55 ^4      lda STROBE,X              ; read sequencer latch
CB1B 10 FB     56         bpl <4
CB1D          57         ;
CB1D C9 AA     58         cmp #ADRMARK2            ; same as DATMARK2
CB1F D0 F2     59         bne <3
CB21          60         ;

```

```

CB21 A0 03      61          ldy #3                ; needed for READADR
CB23           62          ;
CB23           63          ;
CB23           64          ; Check for address/data mark 3 saved in ADRDATMK.
CB23           65          ;
CB23 BD 8C C0    66          ^5          lda STROBE,X          ; read sequencer latch
CB26 10 FB      67          bpl <5
CB28           68          ;
CB28 C5 2C      69          cmp ADRDATMK
CB2A D0 E7      70          bne <3
CB2C           71          ;
CB2C           72          ;
CB2C           73          ; Initialize the checksum to zero.
CB2C           74          ;
CB2C A9 00      75          lda #ZERO
CB2E           76          ;
CB2E 60         77          ^6          rts
CB2F           78          ;
CB2F           79          ;
CB2F           80          ; Check progress of Boot Stage 1. If DOS 4.X RWTS is in
CB2F           81          ; memory, fall into DISKMNG with Y-reg set to zero. Enter
CB2F           82          ; with Y-reg set to zero.
CB2F           83          ;
CB2F AD FF 08   84          CHKDOS4X lda BOOTPGS
CB32 10 58      85          bpl MOVRTN
CB34           86          ;
CB34           87          ;
CB34           88          ; Set all the DRVnTRK and DRVnPHAS locations to zero.
CB34           89          ;
CB34 A5 2B      90          lda SLOT16Z
CB36           91          ;
CB36 4A         92          lsr
CB37 4A         93          lsr
CB38 4A         94          lsr
CB39 4A         95          lsr
CB3A           96          ;
CB3A AA         97          tax
CB3B           98          ;
CB3B 98         99          tya
CB3C          100         ;
CB3C 9D 78 04   101         sta DRV0TRK,X
CB3F 9D F8 04   102         sta DRV1TRK,X
CB42 9D 78 05   103         sta DRV2TRK,X
CB45 9D F8 05   104         sta DRV3TRK,X
CB48          105         ;
CB48 9D 78 06   106         sta DRV0PHAS,X
CB4B 9D F8 06   107         sta DRV1PHAS,X
CB4E 9D 78 07   108         sta DRV2PHAS,X
CB51 9D F8 07   109         sta DRV3PHAS,X
CB54          110         ;
CB54          111         ;
CB54          112         ; Check for DOS 4.X initialization values.
CB54          113         ;
CB54 AD F0 BF   114         DISKMNG lda BLDVRSN
CB57 C9 45      115         cmp #DOS4VRSN
CB59 D0 31      116         bne MOVRTN
CB5B          117         ;
CB5B AD F1 BF   118         lda BLDNMBR
CB5E C9 06      119         cmp #DOS4BLD
CB60 D0 2A      120         bne MOVRTN
CB62          121         ;

```

```

CB62      122 ;
CB62      123 ; Recall Y-reg: zero to connect and not zero to disconnect
CB62      124 ; Rana from the DOS 4.X Disk Address Table. C-flag is set.
CB62      125 ;
CB62 98    126 tya
CB63 F0 01 127 beq >1
CB65      128 ;
CB65 18    129 clc
CB66      130 ;
CB66 A6 2B 131 ^1 ldx SLOT16Z
CB68      132 ;
CB68 A0 20 133 ldy #RANARWTS
CB6A      134 ;
CB6A      135 .if DEBUG
CB6A      136 lda /RANARWTS
CB6A      137 nop
CB6A      138 .el
CB6A AD F8 07 139 lda MSLOT
CB6D      140 .fi
CB6D      141 ;
CB6D 6C F2 BF 142 jmp (MNGDISK)
CB70      143 ;
CB70      144 ;
CB70      145 ; Routines to move the Disk ][ head to the track in the
CB70      146 ; A-reg on the current drive. Returns with zero in A-reg.
CB70      147 ;
CB70 A9 00 148 MOVHEAD0 lda #ZERO
CB72 AA    149 tax
CB73 F0 0B 150 beq MOVHEAD ; always taken
CB75      151 ;
CB75 A0 05 152 MOVHEADN ldy #SNUM-TBLTYPE
CB77      153 ;
CB77 B1 4A 154 lda (IOBADR),Y
CB79 AA    155 tax
CB7A      156 ;
CB7A A0 04 157 ldy #TNUM-TBLTYPE
CB7C      158 ;
CB7C B1 4A 159 lda (IOBADR),Y
CB7E 29 3F 160 and #TRKMASK
CB80      161 ;
CB80 8E 13 01 162 MOVHEAD stx SECTOR ; for SAVETRK
CB83      163 ;
CB83 48    164 pha
CB84      165 ;
CB84 20 8D CB 166 jsr MOVEHD
CB87      167 ;
CB87 68    168 pla
CB88 85 2A 169 sta CURTRKZ
CB8A      170 ;
CB8A A9 00 171 lda #ZERO
CB8C      172 ;
CB8C 60    173 MOVRTN rts
CB8D      174 ;
CB8D      175 ;
CB8D      176 ; Move the disk head to the requested track in half-phase
CB8D      177 ; steps. Set C-flag for drive #4.
CB8D      178 ;
CB8D 20 24 C8 179 MOVEHD jsr SAVETRK
CB90      180 ;
CB90 C5 2A 181 cmp CURTRKZ
CB92 F0 F8 182 beq MOVRTN

```

```

CB94      183 ;
CB94      184 ;
CB94      185 ; Determine the offsets to the next motor phase based on
CB94      186 ; the direction the head needs to move.
CB94      187 ;
CB94 A2 FF 188      ldx #NEGONE
CB96 8A    189      txa
CB97      190 ;
CB97 B0 03 191      bcs >0
CB99      192 ;
CB99 E8    193      inx
CB9A A9 01 194      lda #1
CB9C      195 ;
CB9C 8E 78 06 196 ^0      stx NEXTON
CB9F 8D F8 06 197      sta NEXTOFF
CBA2      198 ;
CBA2      199 ;
CBA2      200 ; Determine the direction to move and increment/decrement
CBA2      201 ; the current position.
CBA2      202 ;
CBA2 A0 00 203      ldy #ZERO
CBA4 84 26 204      sty TEMPZ
CBA6      205 ;
CBA6 20 EB CB 206 ^1      jsr CHKPOS
CBA9      207 ;
CBA9 38    208      sec
CBAA      209 ;
CBAA A5 2A 210      lda CURTRKZ
CBAC ED 78 04 211      sbc FINDTRK
CBAF F0 1B 212      beq >6
CBB1      213 ;
CBB1 B0 06 214      bcs >2
CBB3      215 ;
CBB3 49 FF 216      eor #NEGONE
CBB5      217 ;
CBB5 E6 2A 218      inc CURTRKZ
CBB7 90 04 219      bcc >3 ; always taken
CBB9      220 ;
CBB9 69 FE 221 ^2      adc #!-2
CBBB      222 ;
CBBB C6 2A 223      dec CURTRKZ
CBBD      224 ;
CBBD C5 26 225 ^3      cmp TEMPZ
CBBF 90 02 226      bcc >4
CBC1      227 ;
CBC1 A5 26 228      lda TEMPZ
CBC3      229 ;
CBC3 C9 0C 230 ^4      cmp #OTBLLEN
CBC5 B0 01 231      bcs >5
CBC7      232 ;
CBC7 A8    233      tay
CBC8      234 ;
CBC8 E6 26 235 ^5      inc TEMPZ
CBCA D0 DA 236      bne <1 ; always taken
CBCC      237 ;
CBCC      238 ;
CBCC      239 ; At final destination. Lock in half-phase or phase
CBCC      240 ; position.
CBCC      241 ;
CBCC 20 06 CC 242 ^6      jsr MSWAIT
CBCF      243 ;

```

```

CBCF A5 2A      244      lda CURTRKZ
CBD1           245      ;
CBD1 29 06      246      and #6
CBD3 05 2B      247      ora SLOT16Z
CBD5           248      ;
CBD5 AA        249      tax
CBD6           250      ;
CBD6 A5 2A      251      lda CURTRKZ
CBD8 4A         252      lsr
CBD9           253      ;
CBD9 69 00      254      adc #ZERO
CBDB 29 03      255      and #3
CBDD           256      ;
CBDD 0A         257      asl
CBDE 05 2B      258      ora SLOT16Z
CBE0           259      ;
CBE0 A8         260      tay
CBE1           261      ;
CBE1 BD 80 C0   262      lda PHASEOFF,X
CBE4 B9 80 C0   263      lda PHASEOFF,Y
CBE7           264      ;
CBE7 A9 80      265      lda #$80
CBE9 D0 1B      266      bne MSWAIT          ; always taken
CBEB           267      ;
CBEB           268      ;
CBEB           269      ; Select the next motor phase to turn off or turn on.
CBEB           270      ; Fall into MSWAIT.
CBEB           271      ;
CBEB A5 2A      272      CHKPOS      lda CURTRKZ
CBED           273      ;
CBED 4A         274      lsr
CBEE 90 06      275      bcc >7
CBF0           276      ;
CBF0 ED 78 06   277      sbc NEXTON
CBF3           278      ;
CBF3 38         279      sec
CBF4 B0 04      280      bcs >8
CBF6           281      ;
CBF6 6D F8 06   282      ^7      adc NEXTOFF
CBF9           283      ;
CBF9 18         284      clc
CBFA           285      ;
CBFA 2A         286      ^8      rol
CBFB           287      ;
CBFB 29 07      288      and #7
CBFD 05 2B      289      ora SLOT16Z
CBFF           290      ;
CBFF AA        291      tax
CC00           292      ;
CC00 BD 80 C0   293      lda PHASEOFF,X
CC03           294      ;
CC03 B9 2B CC   295      lda ONOFFTBL,Y
CC06           296      ;
CC06           297      ;
CC06           298      ; Routine to delay A-reg * 99 + 13 usecs. This routine
CC06           299      ; must reside on the same page.
CC06           300      ;
CC06 38         301      MSWAIT      sec
CC07           302      ;
CC07 A2 11      303      ^1      ldx #17
CC09           304      ;

```

```

CC09 CA          305 ^2      dex
CC0A D0 FD       306      bne <2
CC0C            307      ;
CC0C E6 3C       308      inc MOTORTIM
CC0E D0 02       309      bne >3
CC10            310      ;
CC10 E6 3D       311      inc MOTORTIM+1
CC12            312      ;
CC12 E9 01       313 ^3      sbc #1
CC14 D0 F1       314      bne <1
CC16            315      ;
CC16 60          316      rts
CC17            317      ;
CC17            318      ;
CC17            319      ; Calculate the slot-indexed address for the current track
CC17            320      ; or phase value based on slot and drive number. Enter
CC17            321      ; with X-reg containing the /DRV0TRK or /DRV0PHAS value.
CC17            322      ;
CC17 A9 78       323 DRVINDEX lda #DRV0TRK          ; same for #DRV0PHAS
CC19 85 26       324      sta TEMPZ
CC1B            325      ;
CC1B A5 2B       326      lda SLOT16Z
CC1D            327      ;
CC1D 4A          328      lsr
CC1E 4A          329      lsr
CC1F            330      ;
CC1F 0D 15 01    331      ora DNUM0
CC22            332      ;
CC22 4A          333      lsr
CC23 6A          334      ror
CC24            335      ;
CC24 A8          336      tay
CC25            337      ;
CC25 90 01       338      bcc >1
CC27            339      ;
CC27 E8          340      inx
CC28            341      ;
CC28 86 27       342 ^1      stx TEMPZ+1
CC2A            343      ;
CC2A 60          344      rts
CC2B            345      ;
CC2B            346      ;
CC2B            347      ; PHASEON/PHASEOFF table. Time delay uses MSWAIT.
CC2B            348      ;
CC2B 40 32 2A    349 ONOFFTBL hex 40322A242120
CC2E 24 21 20
CC31 1F 1E 1E    350      hex 1F1E1E1D1D1C
CC34 1D 1D 1C
CC37            351      ;
000C            352 OTBLLEN equ *-ONOFFTBL
CC37            353      ;
CC37            354      ;
CC37            355      icl "RANA4.L"

```

LLOAD RANA4.L,A\$4000

```

CC37          1          ttl "Rana ROM Code, RANA4.L"
CC37          2          ;
CC37          3          ;
CC37          4          ; RANA4.L
CC37          5          ;
CC37          6          ;
CC37          7          ; RWTs handler routine for Disk ][ or Rana.
CC37          8          ;
CC37          9          ; This Rana firmware reads sectors 0x00-0x0F every PHASE
CC37         10          ; and sectors 0x10-0x1F the next PHASE/2 of that track.
CC37         11          ;
CC37         12          ; Get slot number and confirm NBUF1PG value.
CC37         13          ;
CC37 8E 78 07   14  RWTSENT stx SLOT16
CC3A          15          ;
CC3A 20 AF C9   16          jsr SETREAD          ; ensure read mode is enabled
CC3D          17          ;
CC3D A9 EF      18          lda #MOTONTIM
CC3F 85 3C      19          sta MOTORTIM
CC41          20          ;
CC41 A9 D8      21          lda /MOTONTIM
CC43 85 3D      22          sta MOTORTIM+1
CC45          23          ;
CC45 A0 00      24          ldy #ZERO
CC47 84 2C      25          sty DRVFLAG
CC49          26          ;
CC49 AD FD BF   27          lda NBUF1PG
CC4C C9 9D      28          cmp /NBUF1L
CC4E F0 05      29          beq >0
CC50          30          ;
CC50 C9 DE      31          cmp /NBUF1H
CC52 D0 26      32          bne SYNERR
CC54          33          ;
CC54 88         34          dey
CC55          35          ;
CC55 8C 14 01   36          ^0          sty NBUFLAG
CC58          37          ;
CC58          38          ;
CC58          39          ; Check whether data is changing on this controller card
CC58          40          ; even though the drive motor is currently off but still
CC58          41          ; possibly spinning.
CC58          42          ;
CC58 A0 08      43          ldy #8
CC5A          44          ;
CC5A BD 8C C0   45          ^1          lda STROBE,X
CC5D          46          ;
CC5D 20 C5 C8   47          jsr WAIT24
CC60          48          ;
CC60 DD 8C C0   49          cmp STROBE,X
CC63 D0 05      50          bne >2
CC65          51          ;
CC65 88         52          dey
CC66 D0 F2      53          bne <1
CC68          54          ;
CC68 E6 2C      55          inc DRVFLAG          ; data not changing
CC6A          56          ;
CC6A          57          ;
CC6A          58          ; Start the motor, then adjust the drive to 0:3 and select
CC6A          59          ; the requested drive.
CC6A          60          ;

```

```

CC6A BD 89 C0      61 ^2      lda MOTORON,X
CC6D              62 ;
CC6D A0 02        63      ldy #DNUM-TBLTYPE
CC6F              64 ;
CC6F B1 4A        65      lda (IOBADR),Y
CC71 48           66      pha
CC72              67 ;
CC72 38           68      sec
CC73              69 ;
CC73 E9 01        70      sbc #1          ; adjust to 0:3
CC75              71 ;
CC75 C9 04        72      cmp #4
CC77 90 06        73      bcc >0
CC79              74 ;
CC79 68           75      pla
CC7A              76 ;
CC7A A9 30        77 SYNERR  lda #RWSYNERR
CC7C              78 ;
CC7C 4C E3 CD     79      jmp ERREXIT
CC7F              80 ;
CC7F              81 ;
CC7F              82 ; Select drives 1:2 or 3:4.  Save adjusted drive number.
CC7F              83 ;
CC7F 8D 15 01     84 ^0      sta DNUM0
CC82              85 ;
CC82 4A           86      lsr
CC83              87 ;
CC83 A8           88      tay
CC84              89 ;
CC84 99 00 C8     90      sta ROMCODE,Y
CC87              91 ;
CC87 90 01        92      bcc >3
CC89              93 ;
CC89 E8           94      inx
CC8A              95 ;
CC8A BD 8A C0     96 ^3      lda DRV0EN,X
CC8D              97 ;
CC8D              98 ;
CC8D              99 ; If the drive number has changed save the new drive
CC8D             100 ; number in the IOB and wait 180 msec for the old drive
CC8D             101 ; to come to rest.
CC8D             102 ;
CC8D A0 10        103      ldy #DRVFND-TBLTYPE
CC8F              104 ;
CC8F 68           105      pla
CC90 D1 4A        106      cmp (IOBADR),Y
CC92 F0 0C        107      beq >5
CC94              108 ;
CC94 91 4A        109      sta (IOBADR),Y
CC96              110 ;
CC96 A0 08        111      ldy #8
CC98              112 ;
CC98 20 06 CC     113 ^4      jsr MSWAIT
CC9B              114 ;
CC9B 88           115      dey
CC9C D0 FA        116      bne <4
CC9E              117 ;
CC9E E6 2C        118      inc DRVFLAG          ; drive changed
CCA0              119 ;
CCA0              120 ;
CCA0              121 ; If there is no data changing on this controller card or

```

```

CCA0      122 ; the drive number has changed, wait for this drive motor
CCA0      123 ; to come up to speed.
CCA0      124 ;
CCA0 A5 2C 125 ^5      lda DRVFLAG
CCA2 F0 07 126          beq >7
CCA4      127 ;
CCA4 20 06 CC 128 ^6      jsr MSWAIT
CCA7      129 ;
CCA7 24 3D 130          bit MOTORTIM+1
CCA9 30 F9 131          bmi <6
CCAB      132 ;
CCAB      133 ;
CCAB      134 ; Always select the lower disk head.
CCAB      135 ;
CCAB 20 11 C8 136 ^7      jsr SELCHEAD
CCAE      137 ;
CCAE      138 ;
CCAE      139 ; Initialize PHASE from the selected saved PHASE location
CCAE      140 ; if it is not zero. If it is zero, initialize the saved
CCAE      141 ; PHASE location with #DFLTPHAS.
CCAE      142 ;
CCAE A2 06 143          ldx /DRVOPHAS
CCB0 20 17 CC 144          jsr DRVINDEX
CCB3      145 ;
CCB3 8C 12 01 146          sty SAVYREG
CCB6      147 ;
CCB6 B1 26 148          lda (TEMPZ),Y
CCB8 D0 04 149          bne >8
CCBA      150 ;
CCBA A9 04 151          lda #DFLTPHAS
CCBC 91 26 152          sta (TEMPZ),Y
CCBE      153 ;
CCBE 85 34 154 ^8      sta PHASE
CCC0      155 ;
CCC0      156 ;
CCC0      157 ; Obtain the requested IOCB PHASE value. If it is zero,
CCC0      158 ; use PHASE. Range check the value in A-reg.
CCC0      159 ;
CCC0 A0 0A 160          ldy #IOCBPHAS-TBLTYPE
CCC2      161 ;
CCC2 B1 4A 162          lda (IOBADR),Y
CCC4 D0 02 163          bne >9
CCC6      164 ;
CCC6 A5 34 165          lda PHASE
CCC8      166 ;
CCC8 C9 11 167 ^9      cmp #PHASMAX+1
CCCA B0 AE 168          bcs SYNERR
CCCC      169 ;
CCCC      170 ;
CCCC      171 ; Compare the requested PHASE value to the saved PHASE
CCCC      172 ; value in PHASE. If they differ move the disk head to
CCCC      173 ; track 0 using the value already in PHASE. Then set PHASE
CCCC      174 ; to the requested PHASE value.
CCCC      175 ;
CCCC 91 4A 176          sta (IOBADR),Y
CCCE      177 ;
CCCE C5 34 178          cmp PHASE
CCD0 F0 0C 179          beq >1
CCD2      180 ;
CCD2 AC 12 01 181          ldy SAVYREG
CCD5      182 ;

```

```

CCD5 91 26      183      sta (TEMPZ),Y
CCD7 48          184      pha
CCD8            185      ;
CCD8 20 70 CB   186      jsr MOVHEAD0
CCDB            187      ;
CCDB 68          188      pla
CCDC 85 34       189      sta PHASE
CCDE            190      ;
CCDE            191      ;
CCDE            192      ; Position the disk head over the requested track using
CCDE            193      ; the PHASE value, verify the requested command, and
CCDE            194      ; process it.
CCDE            195      ;
CCDE 20 75 CB   196      ^1      jsr MOVHEADN
CCE1            197      ;
CCE1 A0 0C       198      ldy #CMDCODE-TBLTYPE
CCE3            199      ;
CCE3 B1 4A       200      lda (IOBADR),Y
CCE5 F0 0F       201      beq >2          ; RWTSSEEK
CCE7            202      ;
CCE7 C9 01       203      cmp #RWTSREAD
CCE9 F0 59       204      beq >7
CCEB            205      ;
CCEB C9 02       206      cmp #RWTSWRIT
CCED F0 0A       207      beq >3
CCEF            208      ;
CCEF C9 04       209      cmp #RWTSFRMT
CCF1 D0 03       210      bne >2
CCF3            211      ;
CCF3 4C ED CD    212      jmp DISKFMT
CCF6            213      ;
CCF6 4C E0 CD    214      ^2      jmp RWTSEXIT
CCF9            215      ;
CCF9            216      ;
CCF9            217      ; A write command, so prenibblize the data first.
CCF9            218      ;
CCF9            219      ; The prenibblize routine converts 256 bytes pointed at by
CCF9            220      ; BUFADR2Z to 342 6-bit nibbles of the form 00XXXXXX.
CCF9            221      ;
CCF9            222      ; First clear NBUF2.
CCF9            223      ;
CCF9 A2 55       224      ^3      ldx #NBUF2SIZ-1
CCFB            225      ;
CCFB A9 00       226      lda #ZERO
CCFD            227      ;
CCFD 2C 14 01    228      bit NBUFLAG
CD00 30 22       229      bmi PRENIBLH
CD02            230      ;
CD02            231      ;
CD02            232      ; Clear NBUF2L.
CD02            233      ;
CD02 9D 00 9E    234      ^4      sta NBUF2L,X
CD05            235      ;
CD05 CA          236      dex
CD06 10 FA       237      bpl <4
CD08            238      ;
CD08            239      ;
CD08            240      ; Process BUFADR2Z into NBUF1 and NBUF2.
CD08            241      ;
CD08 A0 02       242      ldy #2
CD0A            243      ;

```

```

CD0A A2 00      244 ^5      ldx #ZERO
CD0C           245 ;
CD0C 88         246 ^6      dey
CD0D           247 ;
CD0D B1 3E      248      lda (BUFADR2Z),Y
CD0F           249 ;
CD0F           250 ;
CD0F           251 ; Shift low order two bits into NBUF2L.
CD0F           252 ;
CD0F 4A         253      lsr
CD10 3E 00 9E   254      rol NBUF2L,X
CD13           255 ;
CD13 4A         256      lsr
CD14 3E 00 9E   257      rol NBUF2L,X
CD17           258 ;
CD17           259 ;
CD17           260 ; Put low order six bits into NBUF1L.
CD17           261 ;
CD17 99 00 9D   262      sta NBUF1L,Y
CD1A           263 ;
CD1A E8         264      inx
CD1B           265 ;
CD1B E0 56      266      cpx #NBUF2SIZ
CD1D D0 ED      267      bne <6
CD1F           268 ;
CD1F 98         269      tya
CD20 D0 E8      270      bne <5
CD22           271 ;
CD22 F0 20      272      beq >7          ; always taken
CD24           273 ;
CD24           274 ;
CD24           275 ; Clear NBUF2H.
CD24           276 ;
CD24           277 PRENIBLH:
CD24 9D 00 DF   278 ^4      sta NBUF2H,X
CD27           279 ;
CD27 CA         280      dex
CD28 10 FA      281      bpl <4
CD2A           282 ;
CD2A           283 ;
CD2A           284 ; Process BUFADR2Z into NBUF1 and NBUF2.
CD2A           285 ;
CD2A A0 02      286      ldY #2
CD2C           287 ;
CD2C A2 00      288 ^5      ldX #ZERO
CD2E           289 ;
CD2E 88         290 ^6      dey
CD2F           291 ;
CD2F B1 3E      292      lda (BUFADR2Z),Y
CD31           293 ;
CD31           294 ;
CD31           295 ; Shift low order two bits into NBUF2H.
CD31           296 ;
CD31 4A         297      lsr
CD32 3E 00 DF   298      rol NBUF2H,X
CD35           299 ;
CD35 4A         300      lsr
CD36 3E 00 DF   301      rol NBUF2H,X
CD39           302 ;
CD39           303 ;
CD39           304 ; Put low order six bits into NBUF1H.

```

```

CD39          305 ;
CD39 99 00 DE 306      sta NBUF1H,Y
CD3C          307 ;
CD3C E8       308      inx
CD3D          309 ;
CD3D E0 56    310      cpx #NBUF2SIZ
CD3F D0 ED    311      bne <6
CD41          312 ;
CD41 98       313      tya
CD42 D0 E8    314      bne <5
CD44          315 ;
CD44          316 ;
CD44          317 ; Set up for one disk head recalibrate, 2 track seeks,
CD44          318 ; and 32 sector retries.
CD44          319 ;
CD44 A9 02    320 ^7      lda #2
CD46 8D F8 04 321      sta RECALCNT
CD49          322 ;
CD49 A9 02    323 ^1      lda #2
CD4B 8D 78 05 324      sta SEEKCNT
CD4E          325 ;
CD4E A9 20    326      lda #32
CD50 8D F8 05 327      sta RETRYCNT
CD53          328 ;
CD53          329 ;
CD53          330 ; The time it takes WRITADR to write SLPMARK3 and a
CD53          331 ; SYNCMARK and call WRITSCTR is 80 usecs. The time it
CD53          332 ; takes READADR to return from reading SLPMARK2 and
CD53          333 ; check CMDCODE for RWTSREAD before calling WRITSCTR is
CD53          334 ; 72 usecs. Both READSCTR and WRITSCTR have enough time.
CD53          335 ;
CD53 20 AF CA 336 ^2      jsr READADR
CD56 90 19    337      bcc >5
CD58          338 ;
CD58 CE F8 05 339 ^3      dec RETRYCNT
CD5B D0 F6    340      bne <2
CD5D          341 ;
CD5D          342 ;
CD5D          343 ; Recalibrate the disk head. Set the disk head track as
CD5D          344 ; if it was on track 48. Move the disk head to track 0,
CD5D          345 ; then to requested track.
CD5D          346 ;
CD5D A9 40    347 ^4      lda #RWDRVERR      ; get bad drive error
CD5F          348 ;
CD5F CE F8 04 349      dec RECALCNT
CD62 F0 41    350      beq RWTSERR
CD64          351 ;
CD64 A9 C0    352      lda #DFLTPHAS*MAXTRACK ; set track to 48
CD66 20 41 C8 353      jsr SAVETRK2
CD69          354 ;
CD69 20 70 CB 355      jsr MOVHEAD0
CD6C          356 ;
CD6C 20 75 CB 357      jsr MOVHEADN
CD6F F0 D8    358      beq <1      ; always taken
CD71          359 ;
CD71          360 ;
CD71          361 ; Save the found volume number in the IOB. Check address
CD71          362 ; field for the requested track.
CD71          363 ;
CD71 A0 0E    364 ^5      ldy #VOLFND-TBLTYPE
CD73          365 ;

```

```

CD73 A5 2F      366      lda VOLFNDZ
CD75 91 4A      367      sta (IOBADR),Y
CD77           368      ;
CD77 A5 2E      369      lda TRKFNDZ
CD79 C5 2A      370      cmp CURTRKZ
CD7B F0 0D      371      beq >6
CD7D           372      ;
CD7D CE 78 05   373      dec SEEKCNT
CD80 F0 DB      374      beq <4
CD82           375      ;
CD82           376      ;
CD82           377      ; Set the disk head to the track found, then to requested
CD82           378      ; track.
CD82           379      ;
CD82 20 24 C8   380      jsr SAVETRK
CD85           381      ;
CD85 20 75 CB   382      jsr MOVHEADN
CD88 F0 C9      383      beq <2                ; always taken
CD8A           384      ;
CD8A           385      ;
CD8A           386      ; Check for correct sector from the interleave table.
CD8A           387      ;
CD8A A0 05      388      ^6      ldy #SNUM-TBLTYPE
CD8C           389      ;
CD8C B1 4A      390      lda (IOBADR),Y
CD8E A8         391      tay
CD8F           392      ;
CD8F B9 E4 C9   393      lda INTRLEAV,Y
CD92 C5 2D      394      cmp SECFNDZ
CD94 D0 BD      395      bne <2
CD96           396      ;
CD96           397      ;
CD96           398      ; Now at requested sector for read or write operation.
CD96           399      ;
CD96 A0 0C      400      ldy #CMDCODE-TBLTYPE
CD98           401      ;
CD98 B1 4A      402      lda (IOBADR),Y
CD9A C9 01      403      cmp #RWTSREAD
CD9C F0 0A      404      beq >1
CD9E           405      ;
CD9E           406      ;
CD9E           407      ; Write the nibblized data to the requested sector.
CD9E           408      ; If C-flag returns set then write protect sense.
CD9E           409      ;
CD9E 20 3E C9   410      jsr WRITSCTR
CDA1 90 3D      411      bcc RWTSEXIT
CDA3           412      ;
CDA3 A9 10      413      RWPERR  lda #RWPROTER        ; write protect error
CDA5           414      ;
CDA5 38         415      RWTSEERR sec
CDA6 B0 3B      416      bcs ERREXIT                ; always taken
CDA8           417      ;
CDA8           418      ;
CDA8           419      ; Read the requested sector and convert the disk nibbles to
CDA8           420      ; bytes. The post-nibblize routine converts 342 nibbles of
CDA8           421      ; the form 00XXXXXX to eight bit data bytes. The nibbles
CDA8           422      ; are stored in NBUF1 and NBUF2, and the 8-bit bytes are
CDA8           423      ; stored at BUFADR2Z.
CDA8           424      ;
CDA8 20 49 CA   425      ^1      jsr READSCTR
CDAB B0 AB      426      bcs <3

```

```

CDAD          427 ;
CDAD A0 00    428      ldy #ZERO
CDAF          429 ;
CDAF 2C 14 01 430      bit NBUFLAG
CDB2 30 17    431      bmi POSTNIBH
CDB4          432 ;
CDB4          433 ;
CDB4          434 ; Convert the NBUF1L and NBUF2L nibbles to bytes.
CDB4          435 ;
CDB4 A2 56    436 ^2      ldx #NBUF2SIZ
CDB6          437 ;
CDB6 CA      438 ^3      dex
CDB7 30 FB    439      bmi <2
CDB9          440 ;
CDB9          441 ;
CDB9          442 ; Get byte and shift in low order two bits from NBUF2L.
CDB9          443 ;
CDB9 B9 00 9D 444      lda NBUF1L,Y
CDBC          445 ;
CDBC 5E 00 9E 446      lsr NBUF2L,X
CDBF 2A      447      rol
CDC0          448 ;
CDC0 5E 00 9E 449      lsr NBUF2L,X
CDC3 2A      450      rol
CDC4          451 ;
CDC4          452 ;
CDC4          453 ; Store 8-bit data byte in BUFADR2Z buffer and continue.
CDC4          454 ;
CDC4 91 3E    455      sta (BUFADR2Z),Y
CDC6          456 ;
CDC6 C8      457      iny
CDC7 D0 ED    458      bne <3
CDC9          459 ;
CDC9 F0 15    460      beq RWTSEXIT          ; always taken
CDCB          461 ;
CDCB          462 ;
CDCB          463 ; Convert the NBUF1H and NBUF2H nibbles to bytes.
CDCB          464 ;
CDCB          465 POSTNIBH:
CDCB A2 56    466 ^2      ldx #NBUF2SIZ
CDCD          467 ;
CDCD CA      468 ^3      dex
CDCE 30 FB    469      bmi <2
CDD0          470 ;
CDD0          471 ;
CDD0          472 ; Get byte and shift in low order two bits from NBUF2H.
CDD0          473 ;
CDD0 B9 00 DE 474      lda NBUF1H,Y
CDD3          475 ;
CDD3 5E 00 DF 476      lsr NBUF2H,X
CDD6 2A      477      rol
CDD7          478 ;
CDD7 5E 00 DF 479      lsr NBUF2H,X
CDDA 2A      480      rol
Cddb          481 ;
Cddb          482 ;
Cddb          483 ; Store 8-bit data byte in BUFADR2Z buffer and continue.
Cddb          484 ;
Cddb 91 3E    485      sta (BUFADR2Z),Y
CDDD          486 ;
CDDD C8      487      iny

```

```

CDDE D0 ED      488          bne <3
CDE0            489      ;
CDE0            490      ;
CDE0            491      ; RWTS exit with no error.  Clear C-flag.
CDE0            492      ;
CDE0 18          493  RWTSEXIT clc
CDE1            494      ;
CDE1 A9 00       495          lda #RWNOERR
CDE3            496      ;
CDE3            497      ;
CDE3            498      ; RWTS exit with error in A-reg.
CDE3            499      ;
CDE3 A0 0D       500  ERREXIT  ldy #ERRCODE-TBLTYPE
CDE5            501      ;
CDE5 91 4A       502          sta (IOBADR),Y
CDE7            503      ;
CDE7 A6 2B       504          ldx SLOT16Z
CDE9            505      ;
CDE9 BC 88 C0    506          ldy MOTOROFF,X
CDEC            507      ;
CDEC 60          508          rts
CDED            509      ;
CDED            510      ;
CDED            511      ; Routine to format a diskette in the requested drive.
CDED            512      ; Initialize VOLUMEZ from VOLEXPT, ODDBITSZ for WRITADR,
CDED            513      ; and RECALCNT for one recalibration.
CDED            514      ;
CDED A0 03       515  DISKFMT  ldy #VOLEXPT-TBLTYPE
CDEF            516      ;
CDEF B1 4A       517          lda (IOBADR),Y
CDF1 85 41       518          sta VOLUMEZ
CDF3            519      ;
CDF3 A9 AA       520          lda #ODDBITS
CDF5 85 3E       521          sta ODDBITSZ
CDF7            522      ;
CDF7 A9 02       523          lda #2
CDF9 8D F8 04    524          sta RECALCNT
CDFC            525      ;
CDFC            526      ;
CDFC            527      ; Get the ENDTRK and ENDSEC values for DOS 4.X.
CDFC            528      ;
CDFC AD FA BF    529          lda INITVAL
CDFF 85 26       530          sta TEMPZ
CE01            531      ;
CE01 AD FB BF    532          lda INITVAL+1
CE04 85 27       533          sta TEMPZ+1
CE06            534      ;
CE06 A0 10       535          ldy #ENDTRK45
CE08            536      ;
CE08 B1 26       537          lda (TEMPZ),Y
CE0A 8D 16 01    538          sta ENDTRK
CE0D            539      ;
CE0D A0 1C       540          ldy #ENDSEC45
CE0F            541      ;
CE0F B1 26       542          lda (TEMPZ),Y
CE11 8D 17 01    543          sta ENDSEC
CE14            544      ;
CE14            545      ;
CE14            546      ; Begin with MAXSYNC sync bytes and set the disk head
CE14            547      ; location to track 48.  Move the disk head to track 0
CE14            548      ; and then to track 2 to ensure disk head stability.

```

```

CE14          549 ;
CE14 A9 20    550 ^1      lda #MAXSYNC
CE16 85 35    551          sta SYNCNT
CE18          552 ;
CE18 A9 C0    553          lda #DFLTPHAS*MAXTRACK ; set track to 48
CE1A 20 41 C8 554          jsr SAVETRK2
CE1D          555 ;
CE1D 20 70 CB 556          jsr MOVHEAD0          ; returns with zero in A-reg
CE20          557 ;
CE20 AA      558          tax
CE21 A9 02    559          lda #2
CE23          560 ;
CE23 20 80 CB 561          jsr MOVHEAD          ; returns with zero in A-reg
CE26          562 ;
CE26          563 ;
CE26          564 ; Initialize the NBUF1 and NBUF2 nibble buffers. NBUF2
CE26          565 ; must follow NBUF1. The time to clear these buffers will
CE26          566 ; help to stabilize the disk head on track 2.
CE26          567 ;
CE26          568 ;      lda #ZERO
CE26 A8      569          tay
CE27          570 ;
CE27 2C 14 01 571          bit NBUFLAG
CE2A 30 0B    572          bmi CLRBUFRH
CE2C          573 ;
CE2C          574 ;
CE2C          575 ; Clear the NBUF1L and NBUF2L nibble buffers.
CE2C          576 ;
CE2C 99 00 9D 577 ^2      sta NBUF1L,Y
CE2F 99 56 9D 578          sta NBUF1L+NBUF2SIZ,Y
CE32          579 ;
CE32 C8      580          iny
CE33 D0 F7    581          bne <2
CE35          582 ;
CE35 F0 09    583          beq >3          ; always taken
CE37          584 ;
CE37          585 ;
CE37          586 ; Clear the NBUF1H and NBUF2H nibble buffers.
CE37          587 ;
CE37          588 CLRBUFRH:
CE37 99 00 DE 589 ^2      sta NBUF1H,Y
CE3A 99 56 DE 590          sta NBUF1H+NBUF2SIZ,Y
CE3D          591 ;
CE3D C8      592          iny
CE3E D0 F7    593          bne <2
CE40          594 ;
CE40          595 ;
CE40          596 ; Initialize TRACKZ and format only track 0 to establish a
CE40          597 ; working value for SYNCNT. TRACKFMT returns A-reg = 0.
CE40          598 ; Give SYNCNT a final adjustment and begin the disk format.
CE40          599 ;
CE40 85 40    600 ^3      sta TRACKZ
CE42          601 ;
CE42 AA      602          tax          ; for STRTSEC
CE43 A0 10    603          ldy #HIGHSECS      ; for STOPSEC
CE45          604 ;
CE45 20 8A CE 605          jsr TRACKFMT          ; returns with zero in A-reg
CE48 B0 31    606          bcs >7
CE4A          607 ;
CE4A C6 35    608          dec SYNCNT
CE4C C6 35    609          dec SYNCNT

```

```

CE4E          610 ;
CE4E          611 ;
CE4E          612 ; Format the selected track and increment track number
CE4E          613 ; until ENDTRK.
CE4E          614 ;
CE4E A2 00    615 ^4      ldx #ZERO          ; for STRTSEC
CE50          616 ;
CE50 AC 15 01 617          ldy DNUM0
CE53 C0 03    618          cpy #MAXDRIVE
CE55 D0 10    619          bne >5
CE57          620 ;
CE57          621 ;
CE57          622 ; Drive #4 special processing. If ENDSEC is 0x10, then
CE57          623 ; do normal format, otherwise do special format.
CE57          624 ;
CE57 A0 10    625          ldy #HIGHSECS      ; for STOPSEC
CE59 CC 17 01 626          cpy ENDSEC
CE5C F0 0C    627          beq >6
CE5E          628 ;
CE5E 20 8A CE 629          jsr TRACKFMT
CE61 B0 18    630          bcs >7
CE63          631 ;
CE63 A5 40    632          lda TRACKZ          ; recall track number
CE65 A2 10    633          ldx #HIGHSECS      ; for STRTSEC
CE67          634 ;
CE67 AC 17 01 635 ^5      ldy ENDSEC          ; for STOPSEC
CE6A          636 ;
CE6A 20 8A CE 637 ^6      jsr TRACKFMT
CE6D B0 0C    638          bcs >7
CE6F          639 ;
CE6F E6 40    640          inc TRACKZ
CE71          641 ;
CE71 A5 40    642          lda TRACKZ
CE73 CD 16 01 643          cmp ENDTRK
CE76 D0 D6    644          bne <4
CE78          645 ;
CE78 4C E0 CD 646          jmp RWTSEXIT
CE7B          647 ;
CE7B          648 ;
CE7B          649 ; Error from TRACKFMT. Try a recalibration, then exit.
CE7B          650 ;
CE7B CE F8 04 651 ^7      dec RECALCNT
CE7E D0 94    652          bne <1
CE80          653 ;
CE80 A9 08    654          lda #RWINITER      ; track init error
CE82          655 ;
CE82 4C E3 CD 656          jmp ERREXIT
CE85          657 ;
CE85          658 ;
CE85          659 ; Handle write protect sense error.
CE85          660 ;
CE85 68      661 DORWPERR pla
CE86 68      662          pla
CE87          663 ;
CE87 4C A3 CD 664          jmp RWPERR
CE8A          665 ;
CE8A          666 ;
CE8A          667 ; Routine to format the selected track. Move the disk head
CE8A          668 ; to the selected track and allow for 32 track retries.
CE8A          669 ; Mark all sectors as unformatted and initialize SECTORZ to
CE8A          670 ; zero.

```

```

CE8A      671 ;
CE8A 8E 18 01 672 TRACKFMT stx STRTSEC
CE8D 8C 19 01 673 sty STOPSEC
CE90      674 ;
CE90 20 80 CB 675 jsr MOVHEAD
CE93      676 ;
CE93 A9 20 677 lda #MAXRETRY
CE95 8D F8 05 678 sta RETRYCNT
CE98      679 ;
CE98 A9 00 680 ^1 lda #ZERO
CE9A      681 ;
CE9A AC 19 01 682 ldy STOPSEC
CE9D      683 ;
CE9D 99 1F 01 684 ^2 sta SECMAP-1,Y
CEA0      685 ;
CEA0 88 686 dey
CEA1      687 ;
CEA1 CC 18 01 688 cpy STRTSEC
CEA4 D0 F7 689 bne <2
CEA6      690 ;
CEA6 84 3F 691 sty SECTORZ
CEA8      692 ;
CEA8      693 ;
CEA8      694 ; Begin with 128 sync bytes before sector 0 address field,
CEA8      695 ; then use SYNCNT sync bytes before all other sector
CEA8      696 ; address fields.
CEA8      697 ;
CEA8 A0 80 698 ldy #128
CEAA      699 ;
CEAA 2C 00 00 700 bit *-*
CEAD      701 dfs !-2
CEAB      702 ;
CEAB      703 ;
CEAB      704 ; Write an address field and exit if disk is write
CEAB      705 ; protected. Then write the data field. Increment sector
CEAB      706 ; number until STOPSEC.
CEAB      707 ;
CEAB A4 35 708 ^3 ldy SYNCNT
CEAD      709 ;
CEAD 20 B6 C9 710 jsr WRITADR
CEB0 B0 D3 711 bcs DORWPERR
CEB2      712 ;
CEB2 20 3E C9 713 jsr WRITSCTR
CEB5      714 ;
CEB5 E6 3F 715 inc SECTORZ
CEB7      716 ;
CEB7 A5 3F 717 lda SECTORZ
CEB9 CD 19 01 718 cmp STOPSEC
CEBC D0 ED 719 bne <3
CEBE      720 ;
CEBE      721 ;
CEBE      722 ; Ensure delay between last and first sector is a least
CEBE      723 ; ( SYNCNT / 2 ) * 100 usecs.
CEBE      724 ;
CEBE A5 35 725 lda SYNCNT
CEC0 4A 726 lsr
CEC1      727 ;
CEC1 20 06 CC 728 jsr MSWAIT
CEC4      729 ;
CEC4      730 ;
CEC4      731 ; Read the first address field found. If STRTSEC is not

```

```

CEC4      732 ; found, reduce SYNCNT and try again.
CEC4      733 ;
CEC4 20 AF CA 734      jsr READADR
CEC7 B0 0F    735      bcs >4
CEC9      736 ;
CEC9 A5 2D    737      lda SECFNDZ
CECB CD 18 01 738      cmp STRTSEC
CECE F0 14    739      beq >7
CED0      740 ;
CED0 C6 35    741      dec SYNCNT
CED2      742 ;
CED2 A5 35    743      lda SYNCNT
CED4 C9 08    744      cmp #MINSYNC
CED6 90 05    745      bcc >5
CED8      746 ;
CED8      747 ;
CED8      748 ; Any error found reading an address or data field, or
CED8      749 ; finding a duplicate address field or chnage in SYNCNT
CED8      750 ; will cause the entire track to be reformatted.
CED8      751 ;
CED8 CE F8 05 752 ^4      dec RETRYCNT
CEDB D0 BB    753      bne <1
CEDD      754 ;
CEDD 38      755 ^5      sec
CEDE      756 ;
CEDE 60      757      rts
CEDF      758 ;
CEDF      759 ;
CEDF      760 ; Read the next address field and data field.
CEDF      761 ;
CEDF 20 AF CA 762 ^6      jsr READADR
CEE2 B0 F4    763      bcs <4
CEE4      764 ;
CEE4 20 49 CA 765 ^7      jsr READSCTR
CEE7 B0 EF    766      bcs <4
CEE9      767 ;
CEE9      768 ;
CEE9      769 ; Mark the map with the sector found and verify this
CEE9      770 ; is not a duplicate sector found.
CEE9      771 ;
CEE9 A6 2D    772      ldx SECFNDZ
CEEB      773 ;
CEEB BD 20 01 774      lda SECMAPI,X
EEEE 30 E8    775      bmi <4
CEF0      776 ;
CEF0 DE 20 01 777      dec SECMAPI,X
CEF3      778 ;
CEF3 C6 3F    779      dec SECTORZ
CEF5      780 ;
CEF5 A4 3F    781      ldy SECTORZ
CEF7 CC 18 01 782      cpy STRTSEC
CEFA D0 E3    783      bne <6
CEFC      784 ;
CEFC 18      785      clc
CEFD      786 ;
CEFD 60      787      rts
CEFE      788 ;
CEFE      789 ;
CEFE 06      790      byt RANAVERSN
CEFF 06      791      byt RANABLD
CF00      792 ;

```

```
CF00          793  ;  
CF00          794      icl "RANA5.L"
```

```
LLOAD RANA5.L,A$4000
```

```

CF00          1          ttl "Rana ROM Source Code, RANA5.L"
CF00          2          ;
CF00          3          ;
CF00          4          ; RANA5.L
CF00          5          ;
CF00          6          ;
CF00          7          ; This is the firmware on the Rana Disk Controller card no
CF00          8          ; matter which slot it resides in.
CF00          9          ;
CF00         10          ;
CF00         11          .if DEBUG
CF00         12          .el
CF00         13          phs SLOTR0M0
C000         14          .fi
C000         15          ;
C000         16          ;
C000         17          ; Signature bytes for Rana Controller Card.
C000         18          ;
C000 09 20     19          ora #$20
C002 A0 00     20          ldy #$00
C004 A2 03     21          ldx #$03
C006 86 3C     22          stx ROMTEMPZ
C008          23          ;
C008 2C FF CF  24          bit CLRROM
C00B          25          ;
C00B 20 02 C8  26          jsr GETSLOT          ; falls into SELCHEAD
C00E 90 1C     27          bcc BOOT            ; always taken
C010          28          ;
C010          29          ;
C010          30          ; Connect Rana RWTS to DOS.
C010          31          ;
C010 2C FF CF  32  ROMHOOK bit CLRROM
C013          33          ;
C013 EA       34          nop
C014          35          ;
C014 A0 00     36          ldy #ZERO
C016 F0 0E     37          beq >1              ; always taken
C018          38          ;
C018          39          ;
C018          40          ; Disconnect Rana RWTS from DOS.
C018          41          ;
C018 2C FF CF  42  ROMUHOOK bit CLRROM
C01B          43          ;
C01B EA       44          nop
C01C          45          ;
C01C A0 FF     46          ldy #NEGONE
C01E 30 06     47          bmi >1              ; always taken
C020          48          ;
C020          49          ;
C020          50          ; Entry for CALLRWTS for both Disk ][ and Rana drives.
C020          51          ;
C020 2C FF CF  52  RANARWTS bit CLRROM
C023          53          ;
C023 4C 37 CC  54          jmp RWTSENT
C026          55          ;
C026          56          ;
C026 20 02 C8  57  ^1      jsr GETSLOT          ; falls into SELCHEAD
C029          58          ;
C029 4C EA CA  59          jmp HOOKRANA
C02C          60          ;

```

```

C02C      61 ;
C02C      62 ; Establish bank 1, read mode, drive 1, and enable motor.
C02C      63 ;
C02C 8D 00 C8 64 BOOT      sta ROMCODE
C02F      65 ;
C02F BD 8E C0 66          lda DATAIN,X
C032 BD 8C C0 67          lda STROBE,X
C035      68 ;
C035 BD 8A C0 69          lda DRV0EN,X
C038 BD 89 C0 70          lda MOTORON,X
C03B      71 ;
C03B      72 ;
C03B      73 ; Recalibrate the disk head. Assume the disk head is at
C03B      74 ; track 48. Y-reg must be less than 0x80.
C03B      75 ;
C03B      76 ; The disk drive uses a four phase motor to move the track
C03B      77 ; I/O sensor. Each phase must be energized for a period of
C03B      78 ; time, then de-energized. To cause the motor to rotate,
C03B      79 ; the next phase windings are selected sequentially.
C03B      80 ;
C03B A0 60      81          ldy #MAXTRACK*DFLTPHAS/2
C03D      82 ;
C03D BD 80 C0 83 ^2      lda PHASEOFF,X
C040      84 ;
C040 98          85          tya
C041      86 ;
C041 29 03      87          and #$03
C043 0A          88          asl
C044 05 2B      89          ora SLOT16Z
C046      90 ;
C046 AA          91          tax
C047      92 ;
C047 BD 81 C0 93          lda PHASEON,X
C04A      94 ;
C04A A9 56      95          lda #$56
C04C 20 A8 FC 96          jsr WAIT
C04F      97 ;
C04F 88          98          dey
C050 10 EB      99          bpl <2
C052      100 ;
C052      101 ;
C052      102 ; Initialize variables with A-reg = 0 (returned from WAIT).
C052      103 ;
C052 85 26      104          sta BUFRADRZ
C054 85 3D      105          sta ROMSECTR
C056 85 41      106          sta ROMTRACK
C058      107 ;
C058 A9 08      108          lda /PAGE08
C05A 85 27      109          sta BUFRADRZ+1
C05C      110 ;
C05C      111 ;
C05C      112 ; Boot firmware entry. If C-flag = 0, read address header.
C05C      113 ; If C-flag = 1, then read data header.
C05C      114 ;
C05C 18          115 BOOTFW   clc
C05D      116 ;
C05D 08          117 BOOTFW2  php
C05E      118 ;
C05E      119 ;
C05E      120 ; Look for first address/data mark (0xD5).
C05E      121 ;

```

```

C05E BD 8C C0    122  ^1      lda STROBE,X
C061 10 FB      123          bpl <1
C063            124          ;
C063 49 D5      125  ^2      eor #ADRMARK1
C065 D0 F7      126          bne <1
C067            127          ;
C067            128          ;
C067            129          ; Look for second address/data mark (0xAA).
C067            130          ;
C067 BD 8C C0    131  ^3      lda STROBE,X
C06A 10 FB      132          bpl <3
C06C            133          ;
C06C C9 AA      134          cmp #ADRMARK2
C06E D0 F3      135          bne <2
C070            136          ;
C070 EA         137          nop
C071            138          ;
C071            139          ;
C071            140          ; Look for third address or data mark (0x96 or 0xAD).
C071            141          ;
C071 BD 8C C0    142  ^4      lda STROBE,X
C074 10 FB      143          bpl <4
C076            144          ;
C076 C9 96      145          cmp #ADRMARK3
C078 F0 09      146          beq FNDADDR
C07A            147          ;
C07A 28         148          plp
C07B 90 E0      149          bcc BOOTFW2
C07D            150          ;
C07D 49 AD      151          eor #DATMARK3
C07F F0 25      152          beq FNDDATA
C081            153          ;
C081 D0 D9      154          bne BOOTFW          ; always taken
C083            155          ;
C083            156          ;
C083            157          ; An Address header has been found. Read in volume, track,
C083            158          ; and sector; checksum is ignored. Only looking for sector
C083            159          ; number in A-reg. ROMDATA will contain the track number.
C083            160          ;
C083 A0 03      161  FNDADDR  ldy #3
C085            162          ;
C085 85 40      163  ^1      sta ROMDATA
C087            164          ;
C087 BD 8C C0    165  ^2      lda STROBE,X
C08A 10 FB      166          bpl <2
C08C            167          ;
C08C 2A         168          rol
C08D 85 3C      169          sta ROMTEMPZ
C08F            170          ;
C08F BD 8C C0    171  ^3      lda STROBE,X
C092 10 FB      172          bpl <3
C094            173          ;
C094 25 3C      174          and ROMTEMPZ
C096            175          ;
C096 88         176          dey
C097 D0 EC      177          bne <1
C099            178          ;
C099            179          ;
C099            180          ; Fix stack pointer from BOOTFW and check for desired
C099            181          ; sector number.
C099            182          ;

```

```

C099 28          183      plp
C09A          184      ;
C09A C5 3D      185      cmp ROMSECTR
C09C D0 BE      186      bne BOOTFW
C09E          187      ;
C09E          188      ;
C09E          189      ; At correct sector. Check for desired track number.
C09E          190      ;
C09E A5 40      191      lda ROMDATA
C0A0 C5 41      192      cmp ROMTRACK
C0A2 D0 B8      193      bne BOOTFW
C0A4          194      ;
C0A4          195      ;
C0A4          196      ; At correct track and sector. Read the following data
C0A4          197      ; header and the data. C-flag is now set.
C0A4          198      ;
C0A4 F0 B7      199      beq BOOTFW2          ; always taken
C0A6          200      ;
C0A6          201      ;
C0A6          202      ; A data header has been found. Read in the first 0x56
C0A6          203      ; nibbles as the index into the RDNIBL table found at
C0A6          204      ; 0xC896, and save the data in NBUF2BT at 0x0300.
C0A6          205      ;
C0A6 A0 56      206      FNDDATA ldy #NBUF2SIZ
C0A8          207      ;
C0A8 84 3C      208      ^1      sty ROMTEMPZ
C0AA          209      ;
C0AA BC 8C C0   210      ^2      ldy STROBE,X
C0AD 10 FB      211      bpl <2
C0AF          212      ;
C0AF 59 00 C8   213      eor RDNIBL-$96,Y
C0B2          214      ;
C0B2 A4 3C      215      ldy ROMTEMPZ
C0B4          216      ;
C0B4 88         217      dey
C0B5          218      ;
C0B5 99 00 03   219      sta NBUF2BT,Y
C0B8          220      ;
C0B8 D0 EE      221      bne <1
C0BA          222      ;
C0BA          223      ;
C0BA          224      ; Read in the next 0x100 nibbles as the index into the
C0BA          225      ; RDNIBL table, and save the data at the requested memory
C0BA          226      ; location, thus NBUF1 is not necessary since data is
C0BA          227      ; already on a page boundary.
C0BA          228      ;
C0BA 84 3C      229      ^3      sty ROMTEMPZ
C0BC          230      ;
C0BC BC 8C C0   231      ^4      ldy STROBE,X
C0BF 10 FB      232      bpl <4
C0C1          233      ;
C0C1 59 00 C8   234      eor RDNIBL-$96,Y
C0C4          235      ;
C0C4 A4 3C      236      ldy ROMTEMPZ
C0C6          237      ;
C0C6 91 26      238      sta (BUFRADRZ),Y
C0C8          239      ;
C0C8 C8         240      iny
C0C9 D0 EF      241      bne <3
C0CB          242      ;
C0CB          243      ;

```

```

C0CB          244 ; Finally read in checksum and test for 0.
C0CB          245 ;
C0CB BC 8C C0 246 ^5      ldy STROBE,X
C0CE 10 FB     247      bpl <5
C0D0          248 ;
C0D0 59 00 C8 249      eor RDNIBL-$96,Y
C0D3          250 ;
C0D3 D0 87     251 ^6      bne BOOTFW
C0D5          252 ;
C0D5          253 ;
C0D5          254 ; Post nibblize the data buffer with NBUF2BT.
C0D5          255 ;
C0D5 A0 00     256      ldy #ZERO
C0D7          257 ;
C0D7 A2 56     258 ^7      ldx #NBUF2SIZ
C0D9          259 ;
C0D9 CA        260 ^8      dex
C0DA 30 FB     261      bmi <7
C0DC          262 ;
C0DC B1 26     263      lda (BUFRADRZ),Y
C0DE          264 ;
C0DE 5E 00 03 265      lsr NBUF2BT,X
C0E1 2A        266      rol
C0E2          267 ;
C0E2 5E 00 03 268      lsr NBUF2BT,X
C0E5 2A        269      rol
C0E6          270 ;
C0E6 91 26     271      sta (BUFRADRZ),Y
C0E8          272 ;
C0E8 C8        273      iny
C0E9 D0 EE     274      bne <8
C0EB          275 ;
C0EB          276 ;
C0EB          277 ; Check if DOS 4.X RWTS is now in memory. Y-reg is zero.
C0EB          278 ;
C0EB 20 2F CB 279      jsr CHKDOS4X
C0EE          280 ;
C0EE          281 ;
C0EE          282 ; Increment data buffer address and sector number.
C0EE          283 ;
C0EE E6 27     284      inc BUFRADRZ+1
C0F0 E6 3D     285      inc ROMSECTR
C0F2          286 ;
C0F2          287 ;
C0F2          288 ; If sector number is less than 1 found at location 0x0800,
C0F2          289 ; start looking for the requested track and sector. For
C0F2          290 ; sector numbers greater than 0x00, enter the Boot Stage 1
C0F2          291 ; code at 0x0801.
C0F2          292 ;
C0F2 A5 3D     293      lda ROMSECTR
C0F4 CD 00 08 294      cmp PAGE08
C0F7          295 ;
C0F7 A6 2B     296      ldx SLOT16Z
C0F9          297 ;
C0F9 90 D8     298      bcc <6 ; need non-zero value set also
C0FB          299 ;
C0FB 4C 01 08 300      jmp PAGE08+1 ; process boot stage 0
C0FE          301 ;
C0FE          302 ;
C0FE 06        303      byt RANAVRSN
C0FF 06        304      byt RANABLD

```

```
C100          305  ;  
C100          306  ;
```

```
BSAVE RANA,A$1000,B,L$0800
```

```
C100          307          usr RANA  
C100          308  ;  
C100          309  ;  
C100          310          stt "Rana ROM Symbol Table"  
C100          311  ;  
C100          312  ;  
C100          313          end 111
```

```
*** End of Assembly
```

Symbol List starts at 0x7800, ends at 0x807A, used 0x087A, remaining 0x358E

Symbols unsorted:

LOC0	0000	BUFRADRZ	0026	TEMPZ	0026	TEMP2Z	0027	CURTRKZ	002A
SLOT16Z	002B	DRVFLAG	002C	ADRDATMK	002C	ADRFIELD	002C	SECFNDZ	002D
TRKFNDZ	002E	VOLFNDZ	002F	PHASE	0034	SYNCNT	0035	ROMTEMPZ	003C
MOTORTIM	003C	ROMSECTR	003D	BUFADR2Z	003E	ODDBITSZ	003E	SECTORZ	003F
ROMDATA	0040	TRACKZ	0040	ROMTRACK	0041	VOLUMEZ	0041	IOBADR	004A
DEBUG	0000	ZERO	0000	NEGONE	00FF	RANAVRSN	0006	RANABLD	0006
DOS4VRSN	0045	DOS4BLD	0006	SLOTMASK	0007	SECMASK	000F	TRKMASK	003F
MAXDRIVE	0003	DFLTPHAS	0004	PHASMAX	0010	MAXTRACK	0030	HDRSYNC	0006
MINSYNC	0008	MAXSYNC	0020	MAXRETRY	0020	SYNCBITS	0500	HIGHSECS	0010
MAXSEC	0020	ENDTRK45	0010	ENDSEC45	001C	MOTONTIM	D8EF	TBLTYPE	0000
SNUM16	0001	DNUM	0002	VOLEXPT	0003	TNUM	0004	SNUM	0005
USRBUF	0008	IOCBPHAS	000A	BYTCNT	000B	CMDCODE	000C	ERRCODE	000D
VOLFND	000E	SLOTFND	000F	DRVFND	0010	RWTSSEEK	0000	RWTSREAD	0001
RWTSWRIT	0002	RWTSFRMT	0004	RWNOERR	0000	RWINITER	0008	RWPROTER	0010
RWVOLERR	0020	RWSYNERR	0030	RWDRVERR	0040	RWREADER	0080	NBUF2SIZ	0056
ODDBITS	00AA	ADRMARK1	00D5	ADRMARK2	00AA	ADRMARK3	0096	DATMARK1	00D5
DATMARK2	00AA	DATMARK3	00AD	SLPMARK1	00DE	SLPMARK2	00AA	SLPMARK3	00EB
SYNCKMARK	00FF	PAGESIZE	0100	STACK	0100	SAVYREG	0112	SECTOR	0113
NBUFLAG	0114	DNUM0	0115	ENDTRK	0116	ENDSEC	0117	STRTSEC	0118
STOPSEC	0119	HOOKCODE	011A	SECMAP	0120	NBUF2BT	0300	DOSWARM	03D0
HOOKDOS	03EA	FINDTRK	0478	RECALCNT	04F8	XMODE	04FB	SEEKCNT	0578
RETRYCNT	05F8	NEXTON	0678	NEXTOFF	06F8	SLOT16	0778	MSLOT	07F8
DRV0TRK	0478	DRV1TRK	04F8	DRV2TRK	0578	DRV3TRK	05F8	DRV0PHAS	0678
DRV1PHAS	06F8	DRV2PHAS	0778	DRV3PHAS	07F8	BOOTADR	08FE	BOOTPGS	08FF
PAGE08	0800	PAGE10	1000	PAGE20	2000	NBUF1L	9D00	NBUF2L	9E00
NBUF1H	DE00	NBUF2H	DF00	BLDVRSN	BFF0	BLDNMBR	BFF1	MNGDISK	BFF2
INITVAL	BFFA	NBUF1PG	BFFD	ROM2WP	C082	RAM1WE	C08B	PHASEOFF	C080
PHASEON	C081	MOTOROFF	C088	MOTORON	C089	DRV0EN	C08A	DRV1EN	C08B
STROBE	C08C	LATCH	C08D	DATAIN	C08E	DATAOUT	C08F	SLOTROM0	C000
SLOTROM8	C800	CLRROM	CFFF	DISKADRS	D003	WAIT	FCA8	IORTS	FF58
ROMCODE	C800	GETSLOT	C802	SELCEHAD	C811	SAVETRK	C824	SAVETRK2	C841
WRNIBL	C856	RDNIBL	C896	WAIT52	C8C0	WAIT48	C8C2	WAIT24	C8C5
WAIT12	C8C8	WRITBUFH	C900	WRITSCTR	C93E	WRITRTN	C98C	WRITEEXIT	C995
SETREAD	C9AF	WRITADR	C9B6	INTRLEAV	C9E4	WRITSYN0	CA04	WRITSYNC	CA08
WBYTE	CA30	WNIBL9	CA3F	WNIBL2	CA41	WNIBL	CA42	READSCTR	CA49
READBUFH	CA81	READERR	CAAD	READADR	CAAF	READEXIT	CAD5	HOOKRANA	CAEA
EXITCODE	CAFB	EXITLEN	0006	READMRKS	CB01	CHKDOS4X	CB2F	DISKMNG	CB54
MOVHEAD0	CB70	MOVHEADN	CB75	MOVHEAD	CB80	MOVRTN	CB8C	MOVEHD	CB8D
CHKPOS	CBEB	MSWAIT	CC06	DRVNDX	CC17	ONOFFTBL	CC2B	OTBLLEN	000C
RWTSENT	CC37	SYNERR	CC7A	PRENIBLH	CD24	RWPERR	CDA3	RWTSEERR	CDA5
POSTNIBH	CDCB	RWTSEXIT	CDE0	ERREXIT	CDE3	DISKFMT	CDED	CLRBURFH	CE37
DORWPERR	CE85	TRACKFMT	CE8A	ROMHOOK	C010	ROMUHOOK	C018	RANARWTS	C020
BOOT	C02C	BOOTFW	C05C	BOOTFW2	C05D	FNDADDR	C083	FNDDATA	C0A6

Symbols alphabetically sorted:

ADRDATMK	002C	ADRFIELD	002C	ADRMARK1	00D5	ADRMARK2	00AA	ADRMARK3	0096
BLDNMBR	BFF1	BLDVRSN	BFF0	BOOT	C02C	BOOTADR	08FE	BOOTFW	C05C
BOOTFW2	C05D	BOOTPGS	08FF	BUFADR2Z	003E	BUFRADRZ	0026	BYTCNT	000B
CHKDOS4X	CB2F	CHKPOS	CBEB	CLRBURFH	CE37	CLRROM	CFFF	CMDCODE	000C
CURTRKZ	002A	DATAIN	C08E	DATAOUT	C08F	DATMARK1	00D5	DATMARK2	00AA
DATMARK3	00AD	DEBUG	0000	DFLTPHAS	0004	DISKADRS	D003	DISKFMT	CDED
DISKMNG	CB54	DNUM	0002	DNUM0	0115	DORWPERR	CE85	DOS4BLD	0006
DOS4VRSN	0045	DOSWARM	03D0	DRV0EN	C08A	DRV0PHAS	0678	DRV0TRK	0478

DRV1EN	C08B	DRV1PHAS	06F8	DRV1TRK	04F8	DRV2PHAS	0778	DRV2TRK	0578
DRV3PHAS	07F8	DRV3TRK	05F8	DRVFLAG	002C	DRVFND	0010	DRVNDX	CC17
ENDSEC	0117	ENDSEC45	001C	ENDTRK	0116	ENDTRK45	0010	ERRCODE	000D
ERREXIT	CDE3	EXITCODE	CAFB	EXITLEN	0006	FINDTRK	0478	FNDADDR	C083
FNDATA	C0A6	GETSLOT	C802	HDRSYNC	0006	HIGHSECS	0010	HOOKCODE	011A
HOOKDOS	03EA	HOOKRANA	CAEA	INITVAL	BFFA	INTRLEAV	C9E4	IOBADR	004A
IOCBPHAS	000A	IORTS	FF58	LATCH	C08D	LOC0	0000	MAXDRIVE	0003
MAXRETRY	0020	MAXSEC	0020	MAXSYNC	0020	MAXTRACK	0030	MINSYNC	0008
MNGDISK	BFF2	MOTONTIM	D8EF	MOTOROFF	C088	MOTORON	C089	MOTORTIM	003C
MOVEHD	CB8D	MOVHEAD	CB80	MOVHEAD0	CB70	MOVHEADN	CB75	MOVRTN	CB8C
MSLOT	07F8	MSWAIT	CC06	NBUF1H	DE00	NBUF1L	9D00	NBUF1PG	BFFD
NBUF2BT	0300	NBUF2H	DF00	NBUF2L	9E00	NBUF2SIZ	0056	NBUFLAG	0114
NEGONE	00FF	NEXTOFF	06F8	NEXTON	0678	ODDBITS	00AA	ODDBITSZ	003E
ONOFFTBL	CC2B	OTBLLEN	000C	PAGE08	0800	PAGE10	1000	PAGE20	2000
PAGESIZE	0100	PHASE	0034	PHASEOFF	C080	PHASEON	C081	PHASMAX	0010
POSTNIBH	CDCB	PRENIBLH	CD24	RAM1WE	C08B	RANABLD	0006	RANARWTS	C020
RANAVRSN	0006	RDNIBL	C896	READADR	CAAF	READBUFH	CA81	READERR	CAAD
REDEXIT	CAD5	READMRKS	CB01	READSCTR	CA49	RECALCNT	04F8	RETRYCNT	05F8
ROM2WP	C082	ROMCODE	C800	ROMDATA	0040	ROMHOOK	C010	ROMSECTR	003D
ROMTEMPZ	003C	ROMTRACK	0041	ROMUHOOK	C018	RWDRVERR	0040	RWINITER	0008
RWNOERR	0000	RWPERR	CDA3	RWPROTER	0010	RWREADER	0080	RWSYNERR	0030
RWTSENT	CC37	RWTSERR	CDA5	RWTSEXIT	CDE0	RWTSFRMT	0004	RWTSREAD	0001
RWTSSEEK	0000	RWTSWRIT	0002	RWVOLERR	0020	SAVETRК	C824	SAVETRК2	C841
SAVYREG	0112	SECFNDZ	002D	SECMAP	0120	SECMASK	000F	SECTOR	0113
SECTORZ	003F	SEEKNT	0578	SELCEAD	C811	SETREAD	C9AF	SLOT16	0778
SLOT16Z	002B	SLOTFND	000F	SLOTMASK	0007	SLOTROM0	C000	SLOTROM8	C800
SLPMARK1	00DE	SLPMARK2	00AA	SLPMARK3	00EB	SNUM	0005	SNUM16	0001
STACK	0100	STOPSEC	0119	STROBE	C08C	STRTSEC	0118	SYNCBITS	0500
SYNCMARK	00FF	SYNCNT	0035	SYNERR	CC7A	TBLTYPE	0000	TEMP2Z	0027
TEMPZ	0026	TNUM	0004	TRACKFMT	CE8A	TRACKZ	0040	TRKFNDZ	002E
TRKMASK	003F	USRBUF	0008	VOLEXPT	0003	VOLFND	000E	VOLFNDZ	002F
VOLUMEZ	0041	WAIT	FCA8	WAIT12	C8C8	WAIT24	C8C5	WAIT48	C8C2
WAIT52	C8C0	WBYTE	CA30	WNIBL	CA42	WNIBL2	CA41	WNIBL9	CA3F
WRITADR	C9B6	WRITBUFH	C900	WRITEXIT	C995	WRITRTN	C98C	WRITSCTR	C93E
WRITSYN0	CA04	WRITSYNC	CA08	WRNIBL	C856	XMODE	04FB	ZERO	0000

Symbols numerically sorted:

ZERO	0000	TBLTYPE	0000	RWTSSEEK	0000	RWNOERR	0000	LOC0	0000
DEBUG	0000	SNUM16	0001	RWTSREAD	0001	RWTSWRIT	0002	DNUM	0002
VOLEXPT	0003	MAXDRIVE	0003	TNUM	0004	RWTSFRMT	0004	DFLTPHAS	0004
SNUM	0005	RANAVRSN	0006	RANABLD	0006	HDRSYNC	0006	EXITLEN	0006
DOS4BLD	0006	SLOTMASK	0007	USRBUF	0008	RWINITER	0008	MINSYNC	0008
IOCBPHAS	000A	BYTCNT	000B	OTBLLEN	000C	CMDCODE	000C	ERRCODE	000D
VOLFND	000E	SLOTFND	000F	SECMASK	000F	RWPROTER	0010	PHASMAX	0010
HIGHSECS	0010	ENDTRK45	0010	DRVFND	0010	ENDSEC45	001C	RWVOLERR	0020
MAXSYNC	0020	MAXSEC	0020	MAXRETRY	0020	TEMPZ	0026	BUFRADRZ	0026
TEMP2Z	0027	CURTRKZ	002A	SLOT16Z	002B	DRVFLAG	002C	ADRFIELD	002C
ADRDATMK	002C	SECFNDZ	002D	TRKFNDZ	002E	VOLFNDZ	002F	RWSYNERR	0030
MAXTRACK	0030	PHASE	0034	SYNCNT	0035	ROMTEMPZ	003C	MOTORTIM	003C
ROMSECTR	003D	ODDBITSZ	003E	BUFADR2Z	003E	TRKMASK	003F	SECTORZ	003F
TRACKZ	0040	RWDRVERR	0040	ROMDATA	0040	VOLUMEZ	0041	ROMTRACK	0041
DOS4VRSN	0045	IOBADR	004A	NBUF2SIZ	0056	RWREADER	0080	ADRMARK3	0096
SLPMARK2	00AA	ODDBITS	00AA	DATMARK2	00AA	ADRMARK2	00AA	DATMARK3	00AD
DATMARK1	00D5	ADRMARK1	00D5	SLPMARK1	00DE	SLPMARK3	00EB	SYNCMARK	00FF
NEGONE	00FF	STACK	0100	PAGESIZE	0100	SAVYREG	0112	SECTOR	0113
NBUFLAG	0114	DNUM0	0115	ENDTRK	0116	ENDSEC	0117	STRTSEC	0118
STOPSEC	0119	HOOKCODE	011A	SECMAP	0120	NBUF2BT	0300	DOSWARM	03D0
HOOKDOS	03EA	FINDTRK	0478	DRV0TRK	0478	RECALCNT	04F8	DRV1TRK	04F8

XMODE	04FB	SYNCBITS	0500	SEEKCNT	0578	DRV2TRK	0578	RETRYCNT	05F8
DRV3TRK	05F8	NEXTON	0678	DRV0PHAS	0678	NEXTOFF	06F8	DRV1PHAS	06F8
SLOT16	0778	DRV2PHAS	0778	MSLOT	07F8	DRV3PHAS	07F8	PAGE08	0800
BOOTADR	08FE	BOOTPGS	08FF	PAGE10	1000	PAGE20	2000	NBUF1L	9D00
NBUF2L	9E00	BLDVRSN	BFF0	BLDNMBR	BFF1	MNGDISK	BFF2	INITVAL	BFFA
NBUF1PG	BFFD	SLOTROM0	C000	ROMHOOK	C010	ROMUHOOK	C018	RANARWTS	C020
BOOT	C02C	BOOTFW	C05C	BOOTFW2	C05D	PHASEOFF	C080	PHASEON	C081
ROM2WP	C082	FNDADDR	C083	MOTOROFF	C088	MOTORON	C089	DRV0EN	C08A
RAM1WE	C08B	DRV1EN	C08B	STROBE	C08C	LATCH	C08D	DATAIN	C08E
DATAOUT	C08F	FNDDATA	C0A6	SLOTROM8	C800	ROMCODE	C800	GETSLOT	C802
SELCHEAD	C811	SAVETRK	C824	SAVETRK2	C841	WRNIBL	C856	RDNIBL	C896
WAIT52	C8C0	WAIT48	C8C2	WAIT24	C8C5	WAIT12	C8C8	WRITBUFH	C900
WRITSCTR	C93E	WRITRTN	C98C	WRITEEXIT	C995	SETREAD	C9AF	WRITADR	C9B6
INTRLEAV	C9E4	WRITSYN0	CA04	WRITSYNC	CA08	WBYTE	CA30	WNIBL9	CA3F
WNIBL2	CA41	WNIBL	CA42	READSCTR	CA49	READBUFH	CA81	READERR	CAAD
READADR	CAAF	READEXIT	CAD5	HOOKRANA	CAEA	EXITCODE	CAFB	READMRKS	CB01
CHKDOS4X	CB2F	DISKMNG	CB54	MOVHEAD0	CB70	MOVHEADN	CB75	MOVHEAD	CB80
MOVRTN	CB8C	MOVEHD	CB8D	CHKPOS	CBEB	MSWAIT	CC06	DRVINDEX	CC17
ONOFFTBL	CC2B	RWTSENT	CC37	SYNERR	CC7A	PRENIBLH	CD24	RWPERR	CDA3
RWTSERR	CDA5	POSTNIBH	CDCB	RWTSEXIT	CDE0	ERREXIT	CDE3	DISKFMT	CDED
CLRBURFH	CE37	DORWPERR	CE85	TRACKFMT	CE8A	CLRROM	CFFF	DISKADRS	D003
MOTONTIM	D8EF	NBUF1H	DE00	NBUF2H	DF00	WAIT	FCA8	IORTS	FF58