```
PAGE -   0
Current memory available:  125436
00001                      .ABSOLUTE
      2 blocks for procedure code    124262 words left
PAGE -   1  LISA      FILE:


00001                      .PROC LISA
Current memory available:  124909
0000|                      .LIST
0000|                      .TITLE  "Sony DRIVER FOR LISA"
0000|                      .INCLUDE VAR
PAGE -   2  LISA      FILE: VAR.TEXT  Sony DRIVER FOR LISA


0000|                      .PAGE
0000|
0000|                      ;--
0000|                      ;                          ZERO PAGE MAPING
0000|                      ;
0000|                      ;       The following equates are for dividing the RAM into logical areas;
0000|                      ;       however, as the code grew this convention was not strictly adhered
0000|                      ;       to and therefore one will find both "LOCAL" and "GLOBAL" variables
0000|                      ;       scattered throughout the RAM area.
0000|                      ;**
0000|
0000| 0000                 IOB       .EQU  00              ; INPUT OUTPUT BLOCK
0000| 0010                 SHARERAM  .EQU  10              ; READ/WRITE SHARED RAM, INITIALIZED BY 6504 ON BOOT
0000| 0020                 STATUS    .EQU  20              ; READ ONLY STATUS FROM THE 6504
0000| 0030                 IIOB      .EQU  30              ; INTERNAL IOB
0000| 0040                 GLOBALS   .EQU  40              ; 6504 INTERNAL GLOBALS
0000| 0068                 LOCALS    .EQU  68              ; LOCAL VARIABLES
0000|
PAGE -   3  LISA      FILE: VAR.TEXT  Sony DRIVER FOR LISA


0000|                      .Page
0000|
0000|                      ;--
0000|                      ;
0000|                      ;                                  IOB
0000|                      ;
0000|                      ; THE IOB IS ALWAYS COPIED INTO THE INTERNAL IOB (IIOB) AREA BEFORE USAGE
0000|                      ; SO THAT THE 68K CAN START TO BUILD A NEW COMMAND INTO THE IOB AS SOON AS
0000|                      ; POSSIBLE. ALL REFERENCES TO THE IOB OR IIOB ARE DONE IN THE FOLLOWING
0000|                      ; FORMAT:
0000|                      ;                 IOB             IIOB
0000|                      ;                 --------        --------
0000|                      ; GOBYTE          IOB+GOBYTE      IIOB+GOBYTE
0000|                      ; DRIVE           IOB+DRIVE       IIOB+DRIVE
0000|                      ;
0000|                      ;**
0000|
0000| 0007                 IOBSIZE   .EQU    07.             ; SIZE OF IOB BLOCK USED FOR PARAMETER PASSING
0000| 0000                 GOBYTE    .EQU    00              ; COMMAND BYTE FROM 68K
0000|
0000|                      ;          00       COMMAND ACCEPTED BY THE 6504, 68K MAY ISSUE A NEW COMMAND
0000|                      ;          80-89    COMMAND FROM 68K TO 6504
0000|                      ;          80       NULL, TESTS HANDSHAKE
0000|                      ;          81       RWTS COMMAND, COMMAND CODE IN 'COMMAND'
0000|                      ;          83       SEEK
0000|                      ;          84       CALL ADDRESS IN 6504
0000|                      ;          85       CLEAR STATUS
0000|                      ;          86       SET MASK
0000|                      ;          87       CLEAR MASK
0000|                      ;          88       WAIT IN ROM
0000|                      ;          89       Go jump to self forever
0000|                      ;
0000|                      ;          01-7F,82,90-FF      *** RESERVED ***
0000|
0000| 0001                 COMMAND   .EQU  01              ;       RWTS COMMAND CODE
0000|
0000|                      ;          00       READ                Read the Data @ Drive/Side/Track/Sector
0000|                      ;          01       WRITE               Write the Data @ Drive/Side/Track/Sector
0000|                      ;          02       UNCLAMP             Unclamp the disk in Drive
0000|                      ;          03       FORMAT              Format the disk in Drive
0000|                      ;          04       VERIFY              Verify the disk in Drive
0000|                      ;          05       FORMAT TRACK        Format single Track on the disk in Drive
0000|                      ;          06       VERIFY TRACK        Verify single Track on the disk in Drive
0000|                      ;          07       READBF              Read w/o checksum verification
0000|                      ;          08       WRITEBF             Write w/o checksum creation
0000|                      ;          09       CLAMP               Clamp the Disk in Drive
0000|                      ;          0A-FF                          *** RESERVED ***
0000|
0000| 0001                 MASK      .EQU  COMMAND         ; MASK FOR SETTING AND RESETTING IMSK & IST
0000|
0000|                      ;          08       SET OR CLEAR INTERRUPT MASK FOR UPPER DRIVE
0000|                      ;          80       SET OR CLEAR INTERRUPT MASK FOR LOWER DRIVE
0000|                      ;          01       CLEAR DISK INSERTED INTERRUPT FOR UPPER DRIVE
0000|                      ;          10       CLEAR DISK INSERTED INTERRUPT FOR LOWER DRIVE
PAGE -   4  LISA      FILE: VAR.TEXT  Sony DRIVER FOR LISA


0000|                      ;          02       CLEAR BUTTON PRESSED INTERRUPT FOR UPPER DRIVE
0000|                      ;          20       CLEAR BUTTON PRESSED INTERRUPT FOR LOWER DRIVE
0000|                      ;          04       CLEAR R/W COMMAND COMPLETED INTERRUPT FOR UPPER DRIVE
```

Rom. 88 2/10

FCC001

FCC003

```
0000|                           ;          40      CLEAR R/W COMMAND COMPLETED INTERRUPT FOR LOWER DRIVE
0000|
0000|  0001                     ADRL       .EQU  COMMAND          ; LOW BYTE OF ADDRESS FOR 6504 CALL
0000|  0002                     ADRH       .EQU  ADRL+1           ; HIGH BYTE OF ADDRESS FOR 6504 CALL
0000|                                                             ; A call to "1FFB" will reset the 6504.
0000|                           ;
0000|
0000|  0002                     DRIVE      .EQU  02               ; DRIVE NUMBER
0000|
0000|                           ;          00                     DRIVE 0 UPPER DRIVE
0000|                           ;          80                     DRIVE 80 LOWER DRIVE
0000|
0000|  0003                     SIDE       .EQU  03               ; SIDE NUMBER
0000|
0000|                           ;          00                     SIDE 0 UPPER SIDE OF MEDIA
0000|                           ;          01                     SIDE 1 LOWER SIDE OF MEDIA
0000|
0000|  0004                     SECTOR     .EQU  04               ; SECTOR NUMBER
0000|
0000|                           ;          00-15                  MAXIMUM NUMBER DEPENDS ON THE TRACK NUMBER
0000|
0000|  0005                     TRACK      .EQU  05               ; TRACK NUMBER
0000|
0000|                           ;          00-2D                  46 TRACKS TOTAL
0000|
0000|  0006                     SPEED      .EQU  06               ; SPEED OVERRIDE
0000|
0000|                           ;          00                     NO OVERRIDE, SPEED IS DEPENDING ON TRACK NUMBER
0000|                           ;          01-FF                  Modifier value added to nominal speed
0000|
0000|  0007                     FMTCNFM    .EQU  07               ; Format confirmation byte
0000|                                                             ; Used to ensure format is not executed by mistake
0000|                           ;          FF                     FMTCNFM must be = FF for format/format track to work
0000|
0000|  0008                     ERRSTAT    .EQU  08               ; ERROR STATUS, RETURNED AFTER R/W COMMANDS
0000|                                                             ; See constants for current error code values.
0000|
0000|  0009                     DISKID     .EQU  09               ; Current id of disk last accessed
0000|
0000|                           ;          00    UniFile/DuoFile disk
0000|                           ;          01    Lisa disk
0000|                           ;          02    Macintosh disk
0000|
0000|  000A                     NoSides    .EQU  0A               ; Number of sides of disk drive
0000|
0000|  000B                     DrvError   .EQU  0B               ; Hard errors get returned through this byte
0000|
0000|  000C                     HostSeek   .EQU  0C               ; When moving the head this location = 'FF'
0000|
0000|  000D                     SeekErr    .equ  0D               ; When seek does not handshake then = 'OF'
0000|
PAGE -   5  LISA      FILE: VAR.TEXT  Sony DRIVER FOR LISA


0000|                           .Page
0000|
0000|                           ;--
0000|                           ;                                 SHARED RAM
0000|                           ;
0000|                           ;
0000|                           ; SHARED RAM COMES IN TWO FLAVORS: 'READ/WRITE' AND 'READ ONLY'. THE 68K CAN,
0000|                           ; OF COURSE, READ AND WRITE TO ANY BYTE IN THE RAM AT ANY TIME BUT THIS IS NOT
0000|                           ; VERY WISE, (ONE MIGHT SAY VERY FOOLISH... ), SO THAT BY 'READ ONLY' WE MEAN
0000|                           ; MEMORY THAT NEVER SHOULD BE WRITTEN TO BY THE 68K BUT IS VALID TO READ AT
0000|                           ; ANY TIME. TYPICAL 'READ ONLY' VARIABLES ARE THE STATUS FLAGS CLMPED0 AND
0000|                           ; CLMPED80 THAT TELLS THE 68K THAT A DISK IS CLAMPED IN DRIVE 0 OR 80.
0000|                           ;
0000|                           ; 'READ/WRITE' SHARED MEMORY IS, FOR EXAMPLE, THE IOB BUT IN THIS CASE IT IS
0000|                           ; A FAMILY OF 'CONSTANTS' SET UP ON COLD START BY THE 6504 TO THEIR DEFAULT
0000|                           ; VALUES, BUT THEY CAN BE CHANGED AT ANY TIME BY THE 68K TO ANY VALUE. THERE
0000|                           ; IS NO CHECKING OF THE RANGE OF THESE VALUES SO THE NEW ONE BETTER MAKE SENSE
0000|                           ; OR THE 6504 MIGHT GO TO NEVER, NEVER LAND...
0000|                           ;
0000|                           ;                 SHARED: READ/WRITE
0000|                           ;--
0000|
0000|  0010                     MSpdTbl    .equ  ShareRam
0000|  0015                     SCDLY      .EQU  SHARERAM+5.      ; Speed change delay in 5 ms intervals
0000|  0016                     HEADELAY   .EQU  SHARERAM+6.      ; Head settling time in 5 ms intervals
0000|  0017                     MAXDDLY    .EQU  SHARERAM+7.      ; Timer value in 2/3 second before motor off
0000|  0018                     ROMIDNUM   .EQU  SHARERAM+8       ; ROM identification number ( 0018/FCC031 )
0000|  0019                     MAXRETRY   .EQU  SHARERAM+9       ; Maximum number of retries during a read/write
0000|  001A                     MAXRECAL   .EQU  SHARERAM+10.     ; Maximum number of recalibrations during a r/w
0000|  001B                     StpDly     .EQU  SHARERAM+11.     ; Step delay time in 100 usec intervals
0000|  001C                     MONDLY     .EQU  SHARERAM+12.     ; Motor on delay time in 5 ms intervals
0000|
PAGE -   6  LISA      FILE: VAR.TEXT  Sony DRIVER FOR LISA


0000|                           .Page
0000|
0000|                           ;--
0000|                           ;                 SHARED: READ ONLY
0000|                           ;
0000|                           ;--
0000|
```

```
0000| 0020                  Clamped   .EQU  Status          ; Disk in Place (=0 EMPTY, =FF CLAMPED)
0000| 0021  - - - -         MtrOn     .EQU  Clamped+1       ; DRIVE MOTOR SELECT (0=OFF, FF=ON)
0000|
0000| 0022                  CurTrack  .EQU  Status+2        ; Value of current track
0000| 0023                  CurClass  .equ  CurTrack+1      ; Current track class (0:4)
0000|
0000| 0024                  DrvConn   .equ  Status+4        ; Will be 'FF' if a drive is physically there
0000| 0025                  FmtType   .equ  DrvConn+1       ; '2' for single, '22' for double sided
0000|
0000| 0026                  RetryCnt  .EQU  Status+6        ; RETRY COUNT
0000| 0027                  RecalCnt  .EQU  RetryCnt+1      ; RECALLIBRATION COUNT
0000|
0000| 0028                  ImAlive   .EQU  Status+8        ; THIS VARIABLE SPINS AS LONG AS THE MAIN LOOP IS EXECUTING
0000|
0000| 0029                  Counter   .equ  Status+9.       ; GENERAL COUNTER
0000| 002A                  HoldInx   .equ  Counter+1       ; Holds command index temporarily
0000| 002B                  FmtGap    .equ  Counter+2       ; Amt * 5 of 20 usec 'FF's to write as selfsync
0000|
0000| 002C                  Imsk      .EQU  STATUS+12.      ; Bits 7 & 3 are mask; if set then drive enabled
0000| 002D                  DipIntr   .EQU  Imsk+1          ; Flag reflects DIP interrupt
0000| 002E                  OkToGo    .EQU  Imsk+2          ; REFLECTS FDIR (=0, FDIRL; <>0, FDIRH)
0000| 002F                  IST       .EQU  Imsk+3          ; INTERUPT STATUS
0000|
0000|                       ;              7   6   5   4   3   2   1   0
0000|                       ;            +---+---+---+---+---+---+---+---+
0000|                       ;   IST:     |   |   |   |   |   |   |   |   |
0000|                       ;            +---+---+---+---+---+---+---+---+
0000|                       ;              |   |   |   |   |   |   |   |
0000|                       ;              |   |   |   |   |   |   |   +---  DRIVE 0 DISK INSERTED
0000|                       ;              |   |   |   |   |   |   +-------  DRIVE 0 BUTTON PRESSED
0000|                       ;              |   |   |   |   |   +-----------  DRIVE 0 R/W COMPLETED
0000|                       ;              |   |   |   |   +---------------  LOGICAL OR OF BITS 0,1 & 2
0000|                       ;              |   |   |   +-------------------  DRIVE 80 DISK INSERTED
0000|                       ;              |   |   +-----------------------  DRIVE 80 BUTTON PRESSED
0000|                       ;              |   +---------------------------  DRIVE 80 R/W COMPLETED
0000|                       ;              +-------------------------------  LOGICAL OR OF BITS 4,5 & 6
0000|
0000|
0000| 0038                  AdrMk1    .equ  Iiob+8          ; 5 values that indicate start & end of Address field
0000| 0039                  AdrMk2    .equ  AdrMk1+1
0000| 003A                  AdrMk3    .equ  AdrMk1+2
0000| 003B                  AdrMk4    .equ  AdrMk1+3
0000| 003C                  AdrMk5    .equ  AdrMk1+4
0000|
PAGE -   7  LISA       FILE: VAR.TEXT  Sony DRIVER FOR LISA


0000|                       .page                           ; Following 3 byte counter controls both testing
0000|                                                       ; for DIP and shutting off the motors.  When
0000| 0040                  WtLow     .EQU  Globals         ; the low 2 bytes = 0, then test for DIP.  When
0000| 0041                  WtMid     .EQU  WtLow+1         ; the third byte becomes 0 the heads are parked
0000| 0042                  WtHih     .EQU  WtLow+2         ; and the motors are turned off.
0000|                                                       ; For timing and space purposes during writing
0000|                                                       ; of data, an indexed by "y" through the zero
0000| 0043                  INXPTRL   .EQU  GLOBALS+3       ; page instruction is used.  These two bytes
0000| 0044                  INXPTRH   .EQU  INXPTRL+1       ; hold the base address for the index.
0000|
0000|                                                       ; Some constants for timing purposes.
0000| 0045                  K000      .EQU  GLOBALS+5       ; A constant '00'
0000| 0046                  KOFF      .EQU  K000+1          ; A constant 'FF'
0000|
0000|                                                       ; Following 8 locations hold error counters for
0000|                                                       ; various read errors.  The first 3 are for
0000|                                                       ; errors w/ reading data and the last five are
0000|                                                       ; for errors associated w/ the header.
0000| 0048                  STSLP     .EQU  GLOBALS+8.      ; Read Data Starting Bitslip
0000| 0049                  BSCNT     .EQU  StSlp+1         ; Read Data Ending Bitslip
0000| 004A                  CSERROR   .EQU  StSlp+2         ; Read Data Checksum error
0000| 004B                  RASTRT    .EQU  StSlp+3         ; Read Address Starting Bitslip
0000| 004C                  RAEND     .EQU  StSlp+4         ; Read Address Ending Bitslip
0000| 004D                  RASCTR    .EQU  StSlp+5         ; Read Address wrong sector
0000| 004E                  RATRK     .EQU  StSlp+6         ; Read Address wrong track
0000| 004F                  RACSUM    .EQU  StSlp+7         ; Read Address checksum error
0000| 0007                  ERRLEN    .EQU  7               ; 8 bytes, zero based
0000|
0000| 0050                  CSMFND    .EQU  Globals+16.     ; CHECKSUM read from disk
0000| 0051                  VOLFND    .EQU  CsmFnd+1        ; 0 = Unifile/DuoFile, 1 = Lisa, 2 = Mac
0000| 0052                  SDFND     .EQU  CsmFnd+2        ; SIDE FOUND
0000| 0053                  SECFND    .EQU  CsmFnd+3        ; SECTOR FOUND
0000| 0054                  TRKFND    .EQU  CsmFnd+4        ; TRACK FOUND
0000| 0055                  CSUM      .EQU  CsmFnd+5        ; Checksum calculated on read of ADDRESS data
0000| 0004                  ADRSLEN   .EQU  4               ; LENGTH OF THE ADDRESS HEADER - 1
0000|
0000| 0056                  TrkFlg    .equ  Globals+22.
0000| 0057                  MtrFlg    .equ  TrkFlg+1
0000| 0058                  StpAmt    .equ  TrkFlg+2
0000| 0059                  Direct    .equ  TrkFlg+3
0000|
0000| 005A                  IndexL    .equ  Globals+26.
0000| 005B                  IndexH    .equ  IndexL+1
0000| 005C                  RangeL    .equ  IndexL+2
0000| 005D                  RangeH    .equ  IndexL+3
0000|
0000| 005E                  CPBY01    .EQU  Globals+30.     ; Composite byte formed from BUFFER[2FF:301]
0000| 005F                  CPBY02    .EQU  CPBY01+1        ;  [3FE:3FF]
0000|
```

```
0000| 0060                    CPCKSUM   .EQU  Globals+32.      ; Composite byte formed from 3 checksum bytes
0000| 0061                    CKSUM1    .EQU  CPCKSUM+1        ; First checksum byte
0000| 0062                    CKSUM2    .EQU  CPCKSUM+2        ; Second checksum byte
0000| 0063                    CKSUM3    .EQU  CPCKSUM+3        ; Third checksum byte
0000|
PAGE -   8  LISA      FILE: VAR.TEXT  Sony DRIVER FOR LISA


0000| 0064                    TCKSM1    .EQU  Globals+36.      ; During a read of data, the checksum is read
0000| 0065                    TCKSM2    .EQU  TCKSM1+1         ; into "CKSUM1..3".  A new checksum is created
0000| 0066                    TCKSM3    .EQU  TCKSM1+2         ; and stored in these 3 bytes to verify matters.
0000|
0000| 0067                    TEMPSEC   .EQU  Globals+39.      ; TEMPORARY SECTOR COUNTER USED BY FORMAT
0000|
PAGE -   9  LISA      FILE: VAR.TEXT  Sony DRIVER FOR LISA


0000|                         .Page
0000|
0000|                         ;--
0000|                         ;               LOCAL VARIABLES USED IN ONE OR SEVERAL ROUTINES
0000|                         ;**
0000|
0000| 0068                    RWCSMFLG  .EQU  LOCALS          ; Flag for usage of host supplied checksum
0000| 0069                    DELAY     .EQU  LOCALS+1         ; COMPUTED DELAY FOR TOTAL SEEK
0000|
0000| 006A                    Sv1       .EQU  LOCALS+2.        ; storage during Write16
0000| 006B                    Sv2       .EQU  Sv1+1
0000| 006C                    Sv3       .EQU  Sv1+2
0000| 006D                    Sv4       .EQU  Sv1+3
0000|
0000| 006E                    TEMP1     .EQU  Locals+6.        ; 2 Locations for temporary usage by many routines
0000| 006F                    TEMP2     .EQU  Temp1+1
0000|
0000| 0070                    DatMk1    .equ  Locals+8         ; 5 values that indicate start & end of Data field
0000| 0071                    DatMk2    .equ  DatMk1+1
0000| 0072                    DatMk3    .equ  DatMk1+2
0000| 0073                    DatMk4    .equ  DatMk1+3
0000| 0074                    DatMk5    .equ  DatMk1+4
0000|
0000| 0075                    LOWCNT    .EQU  LOCALS+13.       ; Holds value for physical interleave count
0000| 0076                    HIHCNT    .EQU  LOWCNT+1.        ; Same but opposite/complementary value
0000| 0077                    CNTPTR    .EQU  LOWCNT+2.        ; Pointer to which ent to use ( high or low )
0000| 0078                    TOTCNT    .EQU  LOWCNT+3.        ; Total count of sectors written
0000|
0000| 0079                    TEMP3     .EQU  Locals+17.
0000| 007A                    TEMP4     .EQU  Temp3+1
0000|
0000| 007B                    RtyFlg    .equ  Locals+19.       ; flag for use in BadAddr error handling
0000| 007C                    Uu6       .equ  Locals+20.       ; 2 unused locations
0000|
0000| 007D                    Cmdx      .equ  Locals+21.
0000| 007E                    SaveL     .equ  Locals+22.
0000| 007F                    SaveH     .equ  Locals+23.
0000| 003F                    CmdLeng   .equ  3F               ; 64 byte ring buffer of 8 byte IOB's
0000| 0080                    SavIndex  .equ  80
0000|
0000|                         ; *** NOTE --  Ram from 'C0' to 'FF' is used by 68K as parameter memory ***
0000|
0000| 00BF                    LSTUSED   .EQU  0BF              ; Last used location in the ZERO PAGE RAM
0000|
PAGE -  10  LISA      FILE: VAR.TEXT  Sony DRIVER FOR LISA


0000|                         .Page
0000|                         ;--
0000|                         ;
0000|                         ;                               CONSTANTS
0000|                         ;
0000|                         ;**
0000|
0000| 000B                    BUFR12SZ  .EQU  0B               ; LENGTH OF 12 BYTE BLOCK HEADER -1
0000| 0020                    NIBLRETR  .EQU  20               ; THE NUMBER OF NIBBLES READ SEARCHING FOR THE
0000|                                                          ;  FIRST ADDRESS MARK DURING A READ
0000| 004F                    MAXTRACK  .EQU  4F               ; MAXIMUM TRACK NUMBER: 79
0000| 0000                    MINTRACK  .EQU  0                ; MINIMUM TRACK NUMBER: 0
0000| 0004                    MAXCLASS  .EQU  04               ; Maximum track class value -- range from 0..4
0000| 0008                    MINSECNT  .EQU  08               ; Minimum sector count
0000| 000C                    MAXSECNT  .EQU  0C               ; Maximum sector count
0000| 00D4                    MINSPEED  .EQU  0D4              ; Minimum speed value
0000| 0038                    MAXSPEED  .EQU  038              ; Maximum speed value -- Low # = high speed
0000| 001C                    OkDly     .equ  28.
0000|
0000| 00FF                    CNFMVAL   .EQU  0FF              ; Format confirmation check byte
0000| 003F                    LOW6      .EQU  3F               ; MASK FOR LOW 6 BIT
0000|
0000| 0009                    MAXCMD    .EQU  09               ; 10 commands return FDIRH ( - 1 )
0000| 0007                    CMDNUMB   .EQU  07               ; Seven commands not accessed through '81'
0000| 0080                    NULLCMD   .EQU  080              ; Null/Handshake command
0000| 0081                    RWTSCMD   .EQU  081              ; Read/Write Track/Sector command value
0000| 0083                    LWCMDNO   .EQU  083              ; Lowest command number ( not including '81' )
0000| 0085                    CLSTSCMD  .EQU  085              ; Command to clear interrupt status
0000| 0001                    WRTCMD    .EQU  01               ; Value of command to write data to disk
0000| 0008                    WRTBFCMD  .EQU  08               ; Write data to disk, brute force method
0000| 0003                    FRMTDSK   .EQU  03               ; Value of command from host to format disk
0000| 0004                    VRFYDSK   .EQU  04               ; Value of command from host to verify disk
```

```
0000| 0005                  FRMTTRK   .EQU  05              ; Value of command from host to format a track
0000| 0006                  VRFYTRK   .EQU  06              ; Value of command from host to verify a track
0000|
0000| 00D5                  ADM1      .EQU  0D5             ; ADDRESS MARK 1
0000| 00AA                  ADM2      .EQU  0AA             ; ADDRESS MARK 2
0000| 0096                  ADM3      .EQU  096             ; ADDRESS MARK 3
0000| 00AD                  DDM3      .EQU  0AD             ; DATA MARK 3
0000|
0000| 00DE                  BITSLP1   .EQU  0DE             ; BIT SLIP MARK 1
0000| 00AA                  BITSLP2   .EQU  0AA             ; BIT SLIP MARK 2
0000|
0000| 0004                  RclStep   .equ  4.              ; # of steps to take away from Trk00 during recal
0000| 00C8                  OneSec    .equ  200.            ; constant for a one second wait
0000| 0064                  TmOutRcl  .EQU  100.            ; Timeout for recal abort
0000| 0008                  RdAdrTmt  .equ  08.             ; Timeout for looking for Address header
0000| 001F                  IWMMode   .equ  01F             ; constant to setup IWM modes
0000| 0008                  TurnRnd   .equ  08.             ; 5*8=40 msec turn around time for changing directions
0000|
0000| 0005                  Lrge      .equ  05.
0000| 0001                  Smal      .equ  01.
0000| 0009                  TblJmp    .equ  09.
0000| 0011                  WHih      .equ  17.             ; '11' hex
PAGE -  11  LISA      FILE: VAR.TEXT  Sony DRIVER FOR LISA


0000| 0000                  WLow      .equ  00.
0000| 0014                  TLow      .equ  20.
0000|
0000|                       ; ERROR NUMBERS
0000|
0000| 0001                  GERRCMD   .EQU  01              ; GOBYTE ERROR: INVALID COMMAND
0000| 0002                  GERRDRV   .EQU  02              ; GOBYTE ERROR: INVALID DRIVE NUMBER
0000| 0003                  GERRSID   .EQU  03              ; GOBYTE ERROR: INVALID SIDE NUMBER
0000| 0004                  GERRSEC   .EQU  04              ; GOBYTE ERROR: INVALID SECTOR NUMBER
0000| 0005                  GERRTRK   .EQU  05              ; GOBYTE ERROR: INVALID TRACK NUMBER
0000| 0006                  GERRMSK   .EQU  06              ; GOBYTE ERROR: INVALID MASK
0000| 0007                  GERRCLM   .EQU  07              ; GOBYTE ERROR: NO CLAMPED DISK IN DRIVE
0000| 0008                  GERRENA   .EQU  08              ; GOBYTE ERROR: DRIVE NOT ENABLED
0000| 0009                  GERRINTR  .EQU  09              ; GOBYTE ERROR: Pending interrupts not cleared
0000| 000A                  GERRFMPR  .EQU  10.             ; GOBYTE ERROR: Invalid format parameter
0000|
0000| 000B                  PERRROM   .EQU  11.             ; PROGRAM ERROR: ROM TEST FAILED
0000| 000C                  PERRINT   .EQU  12.             ; PROGRAM ERROR: RANDOM IRQ, NMI OR BREAK
0000|
0000| 000D                  DErrCal   .EQU  13.             ; Drive error: time out while looking for track 0
0000| 000E                  IWMError  .EQU  14.             ; fatal error -- IWM doesn't respond to commands
0000| 000F                  StepErr   .EQU  15.             ; Handshake did not occur when stepping
0000| 0010                  DErrTk0   .EQU  16.             ; Drive error: unable to leave track 0 location
0000|
0000| 0014                  SERRPROT  .EQU  20.             ;  ERRSTAT ERROR: WRITE PROTECT ERROR
0000| 0015                  SERRFRMT  .EQU  21.             ;  ERRSTAT ERROR: CAN'T VERIFY DISK
0000| 0016                  SERRCLMP  .EQU  22.             ;  ERRSTAT ERROR: Unable to clamp diskette
0000| 0017                  SERRRD    .EQU  23.             ;  ERRSTAT ERROR: READ ERROR
0000| 0018                  SERRWR    .EQU  24.             ;  ERRSTAT ERROR: WRITE ERROR
0000| 0019                  SERRUCLMP .EQU  25.             ;  ERRSTAT ERROR: Unable to unclamp diskette
0000| 001A                  SERRNOA9  .EQU  26.             ;  ERRSTAT ERROR: Cannot find A9's during chkspd
0000| 001B                  SERRTMT   .EQU  27.             ;  ERRSTAT ERROR: Unable to adjust speed w/in timeout
0000| 001C                  SERRM1TK  .EQU  28.             ;  ERRSTAT ERROR: Cannot write speed track
0000|
0000| 001E                  ErrHdr    .equ  30.             ; UnderRun while writing header
0000| 001F                  ErrWrt    .equ  31.             ; UnderRun while writing data fields
0000|
0000|
PAGE -  12  LISA      FILE: VAR.TEXT  Sony DRIVER FOR LISA


0000|                       .Page
0000|                       ;++
0000|                       ;
0000|                       ;       Data Buffer equates  and Bad Block equates
0000|                       ;
0000|                       ;--
0000|
0000| 00CF                  StackSt   .equ  0CF             ; init stack to "01CF" -- push down stack
0000| 01D0                  SCTRCNT   .EQU  01D0            ; During VERIFY, will no. of bad sectors
0000| 01D1                  TRKNUMB   .EQU  SCTRCNT+1       ; Track number where bad sectors occured
0000| 01D2                  SIDNUMB   .EQU  SCTRCNT+2       ; Side where bad sectors occured
0000| 01D3                  SCTRSAV   .EQU  SCTRCNT+3       ; Start of buffer where sector numbers saved
0000|
0000| 0100                  PAGE01    .EQU  100             ; last 12 bytes of data are for read/write
0000| 01F4                  Bufr12    .EQU  1F4             ; last 12 bytes of data for read/write
0000| 0200                  PAGE02    .EQU  200             ; 256 bytes of data for read/write
0000| 0300                  PAGE03    .EQU  300             ;  "
0000|
0000| 00FF                  PG2LEN    .EQU  0FF             ; # of bytes to read during WRBF02 loop
0000| 00FE                  PG3LEN    .EQU  0FE             ; # of bytes to read during WRBF03 loop
0000|
PAGE -  13  LISA      FILE: VAR.TEXT  Sony DRIVER FOR LISA


0000|                       .Page
0000|
0000|                       ;--
0000|                       ;                               I/O SPACE
0000|                       ;++
0000|
```

```
0000| 0800                  IOSPACE   .EQU  800            ; NAME FOR BEGINNING OF I/O SPACE OFFSETS
0000| 0000                  OFF       .EQU  00             ; OFFSET TO SWITCH A PHASE OFF
0000| 0001                  ON        .EQU  01             ; OFFSET TO SWITCH A PHASE ON
0000| 0000                  Zero      .equ  00             ; offset for drive zero
0000| 0001                  Eighty    .equ  01             ; drive eighty
0000| 0000                  Low       .equ  00             ; PwmEna + Low enables output of PWMReg
0000| 0001                  High      .equ  01
0000| 0000                  InWord    .equ  00             ; direction offsets
0000| 0001                  OutWord   .equ  01
0000|
0000| 0800                  CA0       .EQU  IOSPACE        ; Control signal 0 for MCI PAL on Sony drive
0000| 0802                  CA1       .EQU  IOSPACE+2      ; signal 1
0000| 0804                  CA2       .EQU  IOSPACE+4      ; signal 2
0000| 0806                  LStrb     .EQU  IOSPACE+6      ; Load Strobe -- 0 to 1 to 0 will strobe PAL
0000|
0000| 0808                  MtEna     .EQU  IOSPACE+8      ; Enables output of DrvEna
0000|
0000| 080A                  DrEna     .EQU  IOSPACE+10.    ; = 0 ==> drive 0, = 1 ==> drive 80
0000|
0000| 080C                  Q6L       .EQU  IOSPACE+12.    ; Low = Read or Write
0000| 080D                  Q6H       .EQU  Q6L+1          ; High = Sense or Write Load
0000| 080E                  Q7L       .EQU  IOSPACE+14.    ; Low disables writing to disk
0000| 080F                  Q7H       .EQU  Q7L+1          ; Enables /WrReq output of IWM
0000|
0000| 0810                  CntEna    .EQU  IOSPACE+16.    ; low enables PWM counter/comparator
0000|
0000| 0816                  PwmEna    .EQU  IOSPACE+22.    ; High enables pulses to Sony, else always low
0000|
0000| 0818                  DISL      .EQU  IOSPACE+24.    ; MEMORY ENABLE FOR THE 68K
0000| 0819                  DISH      .EQU  DISL+1         ; MEMORY DISABLE FOR THE 68K
0000|
0000| 081A                  SIDE0SEL  .EQU  IOSPACE+26.    ; SELECTS SIDE 0
0000| 081B                  SIDE1SEL  .EQU  SIDE0SEL+1     ; SIDE 1
0000|
0000| 081C                  BOOTL     .EQU  IOSPACE+28.    ; Disk Diag line; when HIGH then I'm listening.
0000| 081D                  BOOTH     .EQU  BOOTL+1
0000|
0000| 081E                  FDIRL     .EQU  IOSPACE+30.    ; DESELECTS THE INTERUPT TO THE 68K
0000| 081F                  FDIRH     .EQU  FDIRL+1        ; SELECTS THE INTERUPT TO THE 68K
0000|
0000| 0820                  PWMReg    .EQU  IOSPACE+32.    ; Selects PWM register for writing
0000|
0000|                       ;
0000|
0000|                       .ORG 1000
1000|                       .INCLUDE TABLES
PAGE -  14  LISA      FILE: TABLES.TEXT  Sony DRIVER FOR LISA


1000|                        .PAGE
1000|                       ;--
1000|                       ;
1000|                       ;                              NIBBLE TABLE
1000|                       ;
1000|                       ;
1000|                       ; 6-BIT TO 7-BIT NIBL CONVERSION TABLE. CODES WITH MORE THAN ONE PAIR OF
1000|                       ; ADJACENT ZEROES OR WITH NO ADJACENT ONES (EXCEPT B7) ARE EXCLUDED.
1000|                       ; Table is now 256 bytes long for use w/ any byte.
1000|                       ;--
1000|
1000| 1000                  NIBL      .EQU  *
1000|
1000| 96 97 9A 9B 9D 9E 9F            .Byte  096,097,09A,09B,09D,09E,09F,0A6,0A7,0AB,0AC,0AD
1007| A6 A7 AB AC AD
100C| AE AF B2 B3 B4 B5 B6            .Byte  0AE,0AF,0B2,0B3,0B4,0B5,0B6,0B7,0B9,0BA,0BB,0BC
1013| B7 B9 BA BB BC
1018| BD BE BF CB CD CE CF            .Byte  0BD,0BE,0BF,0CB,0CD,0CE,0CF,0D3,0D6,0D7,0D9,0DA
101F| D3 D6 D7 D9 DA
1024| DB DC DD DE DF E5 E6            .Byte  0DB,0DC,0DD,0DE,0DF,0E5,0E6,0E7,0E9,0EA,0EB,0EC
102B| E7 E9 EA EB EC
1030| ED EE EF F2 F3 F4 F5            .Byte  0ED,0EE,0EF,0F2,0F3,0F4,0F5,0F6,0F7,0F9,0FA,0FB
1037| F6 F7 F9 FA FB
103C| FC FD FE FF                    .Byte  0FC,0FD,0FE,0FF
1040| 96 97 9A 9B 9D 9E 9F            .Byte  096,097,09A,09B,09D,09E,09F,0A6,0A7,0AB,0AC,0AD
1047| A6 A7 AB AC AD
104C| AE AF B2 B3 B4 B5 B6            .Byte  0AE,0AF,0B2,0B3,0B4,0B5,0B6,0B7,0B9,0BA,0BB,0BC
1053| B7 B9 BA BB BC
1058| BD BE BF CB CD CE CF            .Byte  0BD,0BE,0BF,0CB,0CD,0CE,0CF,0D3,0D6,0D7,0D9,0DA
105F| D3 D6 D7 D9 DA
1064| DB DC DD DE DF E5 E6            .Byte  0DB,0DC,0DD,0DE,0DF,0E5,0E6,0E7,0E9,0EA,0EB,0EC
106B| E7 E9 EA EB EC
1070| ED EE EF F2 F3 F4 F5            .Byte  0ED,0EE,0EF,0F2,0F3,0F4,0F5,0F6,0F7,0F9,0FA,0FB
1077| F6 F7 F9 FA FB
107C| FC FD FE FF                    .Byte  0FC,0FD,0FE,0FF
1080| 96 97 9A 9B 9D 9E 9F            .Byte  096,097,09A,09B,09D,09E,09F,0A6,0A7,0AB,0AC,0AD
1087| A6 A7 AB AC AD
108C| AE AF B2 B3 B4 B5 B6            .Byte  0AE,0AF,0B2,0B3,0B4,0B5,0B6,0B7,0B9,0BA,0BB,0BC
1093| B7 B9 BA BB BC
1098| BD BE BF CB CD CE CF            .Byte  0BD,0BE,0BF,0CB,0CD,0CE,0CF,0D3,0D6,0D7,0D9,0DA
109F| D3 D6 D7 D9 DA
10A4| DB DC DD DE DF E5 E6            .Byte  0DB,0DC,0DD,0DE,0DF,0E5,0E6,0E7,0E9,0EA,0EB,0EC
10AB| E7 E9 EA EB EC
10B0| ED EE EF F2 F3 F4 F5            .Byte  0ED,0EE,0EF,0F2,0F3,0F4,0F5,0F6,0F7,0F9,0FA,0FB
10B7| F6 F7 F9 FA FB
10BC| FC FD FE FF                    .Byte  0FC,0FD,0FE,0FF
10C0| 96 97 9A 9B 9D 9E 9F            .Byte  096,097,09A,09B,09D,09E,09F,0A6,0A7,0AB,0AC,0AD
```

```
10C7| A6 A7 AB AC AD
10CC| AE AF B2 B3 B4 B5 B6          .Byte  0AE,0AF,0B2,0B3,0B4,0B5,0B6,0B7,0B9,0BA,0BB,0BC
10D3| B7 B9 BA BB BC
10D8| BD BE BF CB CD CE CF          .Byte  0BD,0BE,0BF,0CB,0CD,0CE,0CF,0D3,0D6,0D7,0D9,0DA
10DF| D3 D6 D7 D9 DA
10E4| DB DC DD DE DF E5 E6          .Byte  0DB,0DC,0DD,0DE,0DF,0E5,0E6,0E7,0E9,0EA,0EB,0EC
10EB| E7 E9 EA EB EC
PAGE -  15  LISA      FILE: TABLES.TEXT  Sony DRIVER FOR LISA


10F0| ED EE EF F2 F3 F4 F5          .Byte  0ED,0EE,0EF,0F2,0F3,0F4,0F5,0F6,0F7,0F9,0FA,0FB
10F7| F6 F7 F9 FA FB
10FC| FC FD FE FF                   .Byte  0FC,0FD,0FE,0FF
1100|
1100| 1100                 WidcLow   .equ  *                 ; empirically determined speed tables
1100| 03 23                         .Byte 003, 023
1102| 02 E2                         .Byte 002, 0E2
1104| 02 9C                         .Byte 002, 09C
1106| 02 57                         .Byte 002, 057
1108| 02 15                         .Byte 002, 015
110A| 110A                 WidcHih   .equ  *
110A| 03 43                         .Byte 003, 043
110C| 02 FE                         .Byte 002, 0FE
110E| 02 B7                         .Byte 002, 0B7
1110| 02 70                         .Byte 002, 070
1112| 02 2B                         .Byte 002, 02B
1114| 1114                 ThinLow   .equ  *
1114| 03 30                         .Byte 003, 030
1116| 02 ED                         .Byte 002, 0ED
1118| 02 A7                         .Byte 002, 0A7
111A| 02 62                         .Byte 002, 062
111C| 02 1E                         .Byte 002, 01E
111E| 111E                 ThinHih   .equ  *
111E| 03 36                         .Byte 003, 036
1120| 02 F3                         .Byte 002, 0F3
1122| 02 AD                         .Byte 002, 0AD
1124| 02 66                         .Byte 002, 066
1126| 02 22                         .Byte 002, 022
1128|
PAGE -  16  LISA      FILE: TABLES.TEXT  Sony DRIVER FOR LISA


1128|                               .Page
1128|
1128|                               ;++
1128|                               ;
1128|                               ;         COMMAND PARSING AND TESTING TABLES
1128|                               ;
1128|                               ;--
1128|
1128|                               ; COMMAND PROPERTIES AND PARAMETERS USED, ONE BYTE PER COMMAND:
1128|                               ;
1128|                               ;                   7   6   5   4   3   2   1   0
1128|                               ;                 +---+---+---+---+---+---+---+---+
1128|                               ;                 |   |   |   |   |   |   |   |   |
1128|                               ;                 +---+---+---+---+---+---+---+---+
1128|                               ;                   |   |   |   |   |   |   |   |
1128|                               ;                   |   |   |   |   |   |   |   +--- Drive # & a clamped disk
1128|                               ;                   |   |   |   |   |   |   +------- Side #
1128|                               ;                   |   |   |   |   |   +----------- Sector #
1128|                               ;                   |   |   |   |   +--------------- Track #
1128|                               ;                   |   |   |   +------------------- Mask
1128|                               ;                   |   |   +----------------------- Confirmation byte
1128|                               ;                   |   +--------------------------- Write protection
1128|                               ;                   +------------------------------- Format/Verify Parameters
1128|                               ;++
1128|
1128| 0001                 TTCDRIVE  .EQU 001              ; TEST TABLE CONSTANT: CHECK VALIDITY OF DRIVE PARAMETER
1128| 0002                 TTCSIDE   .EQU 002              ; TEST TABLE CONSTANT: CHECK VALIDITY OF SIDE PARAMETER
1128| 0004                 TTCSECTR  .EQU 004              ; TEST TABLE CONSTANT: CHECK VALIDITY OF SECTOR PARAMETER
1128| 0008                 TTCTRACK  .EQU 008              ; TEST TABLE CONSTANT: CHECK VALIDITY OF TRACK PARAMETER
1128| 0010                 TTCMASK   .EQU 010              ; TEST TABLE CONSTANT: CHECK VALIDITY OF MASK PARAMETER
1128| 0020                 TTCFCFM   .EQU 020              ; TEST TABLE CONSTANT: Confirmation Byte
1128| 0040                 TTCWPRT   .EQU 040              ; TEST TABLE CONSTANT: Write Protection
1128| 0080                 TTCFVPM   .EQU 080              ; TEST TABLE CONSTANT: Format/verify
1128|
1128| 0F                   TESTTBL   .Byte  TTCDRIVE+TTCSIDE+TTCSECTR+TTCTRACK                          ; READ
1129| 4F                             .Byte  TTCDRIVE+TTCSIDE+TTCSECTR+TTCTRACK+TTCWPRT               ; WRITE
112A| 01                             .Byte  TTCDRIVE                                                 ; UNCLAMP
112B| EB                             .Byte  TTCDRIVE+TTCSIDE+TTCTRACK+TTCFCFM+TTCWPRT+TTCFVPM ; FORMAT
112C| 8B                             .Byte  TTCDRIVE+TTCSIDE+TTCTRACK+TTCFVPM                        ; VERIFY
112D| 6B                             .Byte  TTCDRIVE+TTCSIDE+TTCTRACK+TTCFCFM+TTCWPRT                ; FORMAT TRACK
112E| 0B                             .Byte  TTCDRIVE+TTCSIDE+TTCTRACK                                ; VERIFY TRACK
112F| 0F                             .Byte  TTCDRIVE+TTCSIDE+TTCSECTR+TTCTRACK                       ; READ W/O CKSM
1130| 4F                             .Byte  TTCDRIVE+TTCSIDE+TTCSECTR+TTCTRACK+TTCWPRT               ; WRITE W/O CKSM
1131| 00                             .Byte  000                                                      ; CLAMP
1132|
1132| 0B                   TESTGOB   .Byte  TTCDRIVE+TTCSIDE+TTCTRACK                                ; SEEK
1133| 00                             .Byte  000                                                      ; CALL
1134| 00                             .Byte  000                                                      ; CLRIST
1135| 10                             .Byte  TTCMASK                                                  ; SETIMSK
1136| 10                             .Byte  TTCMASK                                                  ; CLRIMSK
1137| 00                             .Byte  000                                                      ; WAITROM
1138| 00                             .Byte  000                                                      ; Drop Dead
1139|
PAGE -  17  LISA      FILE: TABLES.TEXT  Sony DRIVER FOR LISA
```

```
1139|                       .Page
1139|                       ;**
1139|                       ;                          SHARED RAM DEFAULT TABLES
1139|                       ;--
1139|
1139| 1139                  ShareRam .equ *
1139| D5                            .Byte  0D5              ;  0:15    Default speed codes for 5 classes
113A| C0                            .Byte  0C0              ; 16:31
113B| A7                            .Byte  0A7              ; 32:47
113C| 89                            .Byte  089              ; 48:63
113D| 64                            .Byte  064              ; 64:79
113E| 1E                            .Byte  01E              ; ScDly    Speed change Delay: 150 ms
113F| 04                            .Byte  004              ; HeaDelay Head settling time: 30 ms - 10 for SpdChk
1140| 09                            .Byte  009              ; MAXDDLY  DIP SAMPLE DELAY -- USED FOR MOTOR OFF
1141| 88                  RamID     .Byte  088              ; ROM identification number ( $0018/FCC031 )
1142| 64                            .Byte  064              ; MAXRETRY MAXIMUM RETRIES COUNT: 100
1143| 02                            .Byte  002              ; MAXRECAL MAXIMUM RECALLIBRATION COUNT: 1
1144| 82                            .Byte  130.             ; StpDly   * 100usec for step delay
1145| 4F                            .Byte  04F              ; MOnDly   Motor On delay: 400 ms - 10 for SpdChk
1146| 000D                ShareSz .EQU *-ShareRam           ; Size of shared variable area
1146|
1146|                       ;**
1146|                       ;               SECPRTRK ==  NUMBER OF SECTORS PER TRACK
1146|                       ;--
1146|
1146| 1146                  SECPRTRK  .EQU  *
1146| 0C                            .BYTE  12.              ;  0:15
1147| 0B                            .BYTE  11.              ; 16:31
1148| 0A                            .BYTE  10.              ; 32:47
1149| 09                            .BYTE  9.               ; 48:63
114A| 08                            .BYTE  8.               ; 64:79
114B|
114B|                       ;**
114B|                       ;               Two tables -- Address marks & Data marks
114B|                       ;               allows modification by high level software for copy protection
114B|                       ;--
114B|
114B| 114B                  SavAdr    .equ  *
114B| D5                            .byte 0D5               ; First 3 bytes indicate start of address field
114C| AA                            .byte 0AA
114D| 96                            .byte 096
114E| DE                            .byte 0DE               ; last 2 bytes finish the address field
114F| AA                            .byte 0AA
1150|
1150| 1150                  SavDat    .equ  *
1150| D5                            .byte 0D5               ; First 3 bytes indicate start of data field
1151| AA                            .byte 0AA
1152| AD                            .byte 0AD
1153| DE                            .byte 0DE               ; last 2 bytes finish the data field
1154| AA                            .byte 0AA
1155|
1155|
1155|                       .INCLUDE TRKCLASS
PAGE -  18  LISA      FILE: TRKCLASS.TEXT  Sony DRIVER FOR LISA


1155|                       .Page
1155|                       ;**
1155|                       ;               TRACKCLASS will return the class (from 0 to 8 ) of
1155|                       ;       the track in IIOB+TRACK in the "Y" register.
1155|                       ;
1155|                       ;       RETURNS:        "Y" reg = class of track
1155|                       ;                       "A" reg = IIOB+TRACK
1155|                       ;                       "X" reg = unchanged
1155|                       ;--
1155|
1155| 1155                  TRKCLSS   .EQU  *               ; Entry point
1155| A5 35                         LDA   IIOB+TRACK        ; Fetch current track value
1157| A0 04                         LDY   #MAXCLASS         ; Maximum class number
1159| D9 ****               $21       CMP   CLSSTBL,Y       ; Track >= Table entry?
115C| B0**                          BCS   $42               ; Yep
115E| 88                            DEY
115F| D0F8                          BNE   $21               ; Try next track class
115C* 03
1161| 60                  $42       RTS                     ; "Y" reg = track class value
1162|
1162|
1162|
115A* 6211
1162| 1162                  CLSSTBL   .EQU  *               ; Start of track class table
1162| 00                            .BYTE  00               ; Track 0:15
1163| 10                            .BYTE  10               ; Track 16:31
1164| 20                            .BYTE  20               ; Track 32:47
1165| 30                            .BYTE  30               ; Track 48:63
1166| 40                            .BYTE  40               ; Track 64:79
1167|
1167|                       .INCLUDE FR3TO1
PAGE -  19  LISA      FILE: FR3TO1.TEXT  Sony DRIVER FOR LISA


1167|                       .PAGE
1167|                       ;**
1167|                       ;       FR3TO1 will take 3 bytes and create 1 composite byte from them.
1167|                       ;
```

```
1167|                          ;       Input register usage:
1167|                          ;                       A :=    >CCCCCCCC<
1167|                          ;                       X :=    >BBBBBBBB<
1167|                          ;                       Y :=    >AAAAAAAA<
1167|                          ;
1167|                          ;       Output register usage:
1167|                          ;                       A :=    >00AABBCC< == Nibblized
1167|                          ;-->
1167|
1167| 1167                     FR3TO1    .EQU  *                       ; From 3 to 1 routine
1167| 4A                                 LSR   A                       ; Ignore low 6 bits of "A" as they
1168| 4A                                 LSR   A                       ;  will get shifted out and lost
1169| 85 6E                              STA   TEMP1                   ; >00CCxxxx<
116B| 8A                                 TXA
116C| 29 C0                              AND   #0C0                    ; Clear low bits
116E| 05 6E                              ORA   TEMP1                   ; >BBCCxxxx<
1170| 4A                                 LSR   A
1171| 4A                                 LSR   A
1172| 85 6E                              STA   TEMP1                   ; >00BBCCxx<
1174| 98                                 TYA
1175| 29 C0                              AND   #0C0                    ; Clear low bits
1177| 05 6E                              ORA   TEMP1                   ; >AABBCCxx<
1179| 4A                                 LSR   A
117A| 4A                                 LSR   A                       ; >00AABBCC<
117B| AA                                 TAX
117C| BD 0010                           LDA   NIBL,X                  ; Nibblize the new composite byte
117F| 60                                 RTS
1180|
1180|
1180|
1180|                          .INCLUDE DENIBBLE
PAGE -  20  LISA       FILE: DENIBBLE.TEXT  Sony DRIVER FOR LISA


1180|                          .PAGE
1180|                          ;--
1180|                          ;
1180|                          ;                               DE-NIBBLE TABLE
1180|                          ;
1180|                          ; 7-BIT TO 6-BIT  'DENIBLIZE' TABLE (16-SECTOR FORMAT).
1180|                          ; VALID CODES 096 TO 0FF ONLY.
1180|                          ; CODES WITH MORE THAN ONE PAIR OF ADJACENT ZEROES OR WITH NO ADJACENT ONES
1180|                          ; (EXCEPT BIT 7) ARE EXCLUDED.
1180|                          ;
1180|                          ;**
1180|
1180| 00 00 00 00 00 00 00     .ORG  1196
1196| 1100                     DNIBL     .EQU  *-0096
1196| 00 01 98                           .BYTE 000,001,098
1199| 99 02 03                           .BYTE 099,002,003
119C| 9C 04 05                           .BYTE 09C,004,005
119F| 06 A0 A1                           .BYTE 006,0A0,0A1
11A2| A2 A3 A4                           .BYTE 0A2,0A3,0A4
11A5| A5 07 08                           .BYTE 0A5,007,008
11A8| A8 A9 AA                           .BYTE 0A8,0A9,0AA
11AB| 09 0A 0B                           .BYTE 009,00A,00B
11AE| 0C 0D B0                           .BYTE 00C,00D,0B0
11B1| B1 0E 0F                           .BYTE 0B1,00E,00F
11B4| 10 11 12                           .BYTE 010,011,012
11B7| 13 B8 14                           .BYTE 013,0B8,014
11BA| 15 16 17                           .BYTE 015,016,017
11BD| 18 19 1A                           .BYTE 018,019,01A
11C0| C0 C1 C2                           .BYTE 0C0,0C1,0C2
11C3| C3 C4 C5                           .BYTE 0C3,0C4,0C5
11C6| C6 C7 C8                           .BYTE 0C6,0C7,0C8
11C9| C9 CA 1B                           .BYTE 0C9,0CA,01B
11CC| CC 1C 1D                           .BYTE 0CC,01C,01D
11CF| 1E D0 D1                           .BYTE 01E,0D0,0D1
11D2| D2 1F D4                           .BYTE 0D2,01F,0D4
11D5| D5 20 21                           .BYTE 0D5,020,021
11D8| D8 22 23                           .BYTE 0D8,022,023
11DB| 24 25 26                           .BYTE 024,025,026
11DE| 27 28 E0                           .BYTE 027,028,0E0
11E1| E1 E2 E3                           .BYTE 0E1,0E2,0E3
11E4| E4 29 2A                           .BYTE 0E4,029,02A
11E7| 2B E8 2C                           .BYTE 02B,0E8,02C
11EA| 2D 2E 2F                           .BYTE 02D,02E,02F
11ED| 30 31 32                           .BYTE 030,031,032
11F0| F0 F1 33                           .BYTE 0F0,0F1,033
11F3| 34 35 36                           .BYTE 034,035,036
11F6| 37 38 F8                           .BYTE 037,038,0F8
11F9| 39 3A 3B                           .BYTE 039,03A,03B
11FC| 3C 3D 3E                           .BYTE 03C,03D,03E
11FF| 3F                               .BYTE 03F
1200|
1200|
1200|                          .INCLUDE NEWRWADDR
PAGE -  21  LISA       FILE: NEWRWADDR.TEXT  Sony DRIVER FOR LISA


1200|                          .PAGE
1200|                          ;**
1200|                          ;
1200|                          ;                                       WRITE ADR FIELD SUBROUTINE
1200|                          ;
1200|                          ; (16-SECTOR FORMAT) WRITES 27, 40-USEC (10-BIT) SELF-SYNC NIBLS, ADR FIELDS
```

```
1200|                    ; 16-SECTOR START MARKS (0D5,0AA,096), BODY (TRACK, SECTOR, SIDE, VOLUME,
1200|                    ; CHECKSUM), END FIELD MARKS, AND THE WRITE TURN-OFF NIBL. It then jumps to
1200|                    ; the "WRSYNC" code which will write the starting sync for the data field,
1200|                    ; write the data fields and then return to the caller.
1200|                    ;--
1200|
1200|                    .ORG     1200                ; Ensure assembler error if moved.
1200|
1200| 1200               WRSYNTRK .EQU  *             ; Write a sync track before first track
1200| A0 00              ldy    #00                   ; 256 sync bytes * 6 to write (about 2 sectors)
1202| F0**               beq    WRADR01
1204|
1204| 1204               WADR16   .EQU  *
1204| A4 2B              LDY    FmtGap                ;              default to 6*5 20usec 'FF's
1202* 02
1206| 24 35              WRADR01  bit   IIob+Track    ;              look for bit 6
1208| 50**               bvc    $23
120A| A9 01              lda    #01
120C| 05 33              ora    IIob+Side
120E| 85 33              sta    IIob+Side             ;              set bit 0 in Side
1208* 06
1210| A9 01              $23      lda   #1            ;              flag for usage of AdrMk 1 & 2
1212| 20 ****            JSR    SYNC20                ; 6     10/10  Write sync fields
1215| A5 3A              lda    AdrMk3                ; 3     19     Last byte in starting bitslip
1217| 20 ****            jsr    WrNibl                ; 6
121A|
121A| A0 02              ldy    #02                   ; 2
121C| B6 33              $37      ldx   IIob+Side,y   ; 4            Write Track, Sector, then Side
121E| 20 ****            jsr    WrByteX               ; 6
1221| 88                 dey
1222| 10F8               bpl    $37
1224|
1224| A6 25              ldx    FmtType               ; 3     9      format type
1226| 20 ****            JSR    WrByteX               ; 6     13/6
1229| A5 35              LDA    IIOB+TRACK            ; 3     9      Create ADDRESS checksum.
122B| 45 34              EOR    IIOB+SECTOR           ; 3     12
122D| 45 33              EOR    IIOB+SIDE             ; 3     15
122F| 45 25              EOR    FmtType               ; 3     18
1231| 20 ****            JSR    WrByte                ; 6     24/6   write the address checksum
1234|
1234| A5 3B              lda    AdrMk4                ; 3     9
1236| 20 ****            jsr    WrNibl                ; 6
1239| A5 3C              lda    AdrMk5                ; 3     9      Last byte in address field
123B| 20 ****            jsr    WrNibl                ; 6
123E|
123E| 20 ****            JSR    SHTOFF                ; 6     16           Return to SENSE mode
1241| A9 1E              lda    #ErrHdr               ; 2                  UnderRun during header
1243| B0**               bcs    WstTm                 ; 2,3                abort upon error
PAGE -  22  LISA      FILE: NEWRWADDR.TEXT  Sony DRIVER FOR LISA


1245| 4C ****            JMP    Write16               ; 3                  Go & write data to disk
1248|
1243* 03
1248| 60                 WSTTM    RTS                 ; 6     10
1249|
PAGE -  23  LISA      FILE: NEWRWADDR.TEXT  Sony DRIVER FOR LISA


1249|                    .PAGE
1249|                    ;**
1249|                    ;                            RDADR
1249|                    ;--
1249|
1249| 1249               RDADR    .EQU  *
1249| A9 00              lda    #0
124B| 85 5C              sta    RangeL
124D| A9 08              lda    #RdAdrTmt
124F| 85 5D              sta    RangeH
1251| 20 ****            jsr    SetRMode              ; setup PAL on Sony to read mode
1254| AD 0C08            LDA    Q6L                   ; Switch from SENSE to READ
1257|
1257| E6 5C              RDASYN   inc   RangeL        ; 5
1259| D0**               bne    RDAD1                 ; 2,3
125B| C6 5D              dec    RangeH                ; 5
125D| F0**               beq    RAERR1                ; 2,3
125F|
1259* 04
125F| A2 00              RDAD1    ldx   #00
1261| AD 0E08            $24      lda   Q7L           ; 4..
1264| 30**               bmi    RdAsn1                ; 2,3          Valid if high bit = 1
1266| CA                 dex                          ; 2
1267| D0F8               bne    $24                   ; 2,3          loop 255 times == 85 bytes
1269| F0**               beq    RaErr1                ; 3
1268|
1264* 05
1268| C5 38              RDASN1   CMP   AdrMk1        ; 2..          ADDRESS MARK 1?
126D| D0E8               BNE    RDASYN                ; 2,3          BRANCH IF NOT
126F|
126F| AD 0E08            RDAD2    LDA   Q7L           ; 4..
1272| 10FB               BPL    RDAD2                 ; 2,3
1274| C5 39              CMP    AdrMk2                ; 2..          ADDRESS MARK 2?
1276| D0F3               BNE    RDASN1                ; 2,3
1278|
1278| AD 0E08            RDAD3    LDA   Q7L           ; 4..
127B| 10FB               BPL    RDAD3                 ; 2,3
```

```
127D| C5 3A                        CMP   AdrMk3          ; 2..          Address Mark 3?
127F| D0EA                        BNE   RDASN1          ; 2,3
1281|
1281|                    ; MARKS READ NOW READ ADDRESS
1281|                    ; CARRY IS SET
1281|
1281| A2 04                        LDX   #ADRSLEN        ; 2
1283| A9 00                        LDA   #000            ; 2 CLEAR CSUM
1285| 85 55              RFLD      STA   CSUM            ; 3     27
1287| AC 0E08            RDAD4     LDY   Q7L             ; 4
128A| 10FB                        BPL   RDAD4           ; 2      6      Do again if no valid data.
128C| B9 0011                     LDA   DNIBL,Y         ; 4     12      Unpack the data
128F| 95 50                       STA   CsmFnd,X        ; 4     16      Store in FOUND table
1291| 45 55                       EOR   CSUM            ; 3     19      Update checksum
1293| CA                          DEX                   ; 2     21      Next field
1294| 10EF                        BPL   RFLD            ; 2,3   24      Loop until "X" becomes negative.
1296| AA                          TAX                   ; 2             If "CSUM" = 0 then AOK.
PAGE -  24  LISA      FILE: NEWRWADDR.TEXT  Sony DRIVER FOR LISA


1297| D0**                        BNE   RAERR5          ; 2,3           Not OK, signal an error.
1299|
1299|                    ; Now compare against 2 final bytes and make sure at right track and sector.
1299|
1299| AD 0E08            RASLP1    LDA   Q7L             ; 4..
129C| 10FB                        BPL   RASLP1          ; 2,3
129E| C5 3B                       CMP   AdrMk4          ; 2..
12A0| D0**                        BNE   RAERR2          ; 2,3
12A2| A9 01                       lda   #01             ; 2
12A4| 24 52                       bit   SdFnd           ; 3
12A6| F0**                       beq   RaSlp2          ; 2,3
12A8| A9 40                       lda   #40
12AA| 05 54                       ora   TrkFnd
12AC| 85 54                       sta   TrkFnd
12A6* 06
12AE| AD 0E08            RASLP2    LDA   Q7L             ; 4..
12B1| 10FB                        BPL   RASLP2          ; 2,3
12B3| C5 3C                       CMP   AdrMk5          ; 2..
12B5| D0**                        BNE   RAERR2          ; 2,3
12B7| A5 35                       LDA   IIOB+TRACK      ; 3
12B9| C5 54                       CMP   TRKFND          ; 2
12BB| D0**                        BNE   RAERR4          ; 2
12BD| A5 34                       LDA   IIOB+SECTOR     ; 3
12BF| C5 53                       CMP   SECFND          ; 2
12C1| D0**                        BNE   RAERR3          ; 2
12C3| 18                          CLC                   ; 2.. NO ERROR
12C4| A5 51                       LDA   VOLFND          ; Load the Disk ID value just read.
12C6| 85 09                       STA   IOB+DISKID      ; Tell host about what type of disk it is
12C8| B8                 RAEXIT    CLV                   ; Clear overflow bit (PREVIOUSLY USED FOR FATAL)
12C9| AD 0D08            RAEXIT1   LDA   Q6H             ; Switch back from READ to SENSE
12CC| 60                          RTS
12CD|                    ; WE ABORT UPON FIRST ERROR-NO MATTER WHAT IT IS. THESE ALL GET RESET
12CD|                    ; ON SEEKING (EVEN MICROSTEPPING).
12CD|
1269* 62
125D* 6E
12CD| E6 4B              RAERR1    INC   RASTRT          ; Start bitslip Error -- Fatal error
12CF| 38                          SEC
12D0| B0F6                        BCS   RAEXIT
12D2|
12B5* 1B
12A0* 30
12D2| E6 4C              RAERR2    INC   RAEND
12D4| 38                          SEC                   ; Ending bitslip Error
12D5| B0F1                        BCS   RAEXIT
12D7|
12C1* 14
12D7| E6 4D              RAERR3    INC   RASCTR
12D9| 38                          SEC                   ; Sector error
12DA| B0EC                        BCS   RAEXIT
12DC|
1297* 43
12DC| E6 4F              RAERR5    INC   RACSUM
12DE| 38                          SEC                   ; Check sum error
12DF| B0E7                        BCS   RAEXIT
PAGE -  25  LISA      FILE: NEWRWADDR.TEXT  Sony DRIVER FOR LISA


12E1|
12BB* 24
12E1| E6 4E              RAERR4    INC   RATRK           ; Track error
12E3| 38                RAERR6    SEC
12E4| B0E2                        BCS   RAEXIT
12E6|
12E6|
12E6|                    .INCLUDE Interface
PAGE -  26  LISA      FILE: INTERFACE.TEXT  Sony DRIVER FOR LISA


12E6|                    .page
12E6|                    ;**
12E6|                    ;       Interface routines
12E6|                    ;--
12E6|
12E6| 12E6               SELSIDE   .EQU  *
12E6| A2 00                        ldx   #0
```

```
12E8| A9 20                        lda   #20
12EA| 24 33                        bit   IIOB+Side
12EC| F0**                        beq   SetScl
12EE| E8                          inx
12EF|
12EC* 01
12EF| 12EF                 SetScl .equ  *
12EF| AD 0108                      lda   Ca0+High
12F2| AD 0308                      lda   Ca1+High
12F5| BD 1A08                      lda   SideOScl,x
12F8| 8A                          txa
12F9| 60                          rts
12FA|
12FA| 12FA                 SetLow .equ  *                ; set Side Select line low
12FA| A2 00                        ldx   #Low
12FC| F0F1                        beq   SetScl
12FE|
12FE| 12FE                 SetHigh .equ  *               ; set line high
12FE| A2 01                        ldx   #High
1300| D0ED                        bne   SetScl
1302|
1302| 1302                 ReadHp .equ  *
1302| 20 FE12                      jsr   SetHigh
1305| AD 0208              RdS2   lda   Ca1+Low
1308| AD 0408              RdS1   lda   Ca2+Low
130B| 130B                 ReadIt .equ  *                ; read sense bit & rotate into ccC
130B| AD 0D08                      lda   Q6H             ; Ensure state machine is in proper state
130E| AD 0E08                      lda   Q7L             ; READ SENSE DATA INTO Bit07
1311| 0A                          asl   a               ; MOVE Bit07 TO CARRY
1312| 60                          rts
1313|
1313| 1313                 ReadIndx .EQU *
1313| 18                          clc
1314| 1314                 Rid    .equ  *
1314| 20 FE12                      jsr   SetHigh
1317| AD 0008                      lda   Ca0+Low
131A| 90EC                        bcc   RdS1            ; set Ca2 = 0
131C| B0E7                        bcs   RdS2            ; set Ca1 & Ca2 = 0
131E|
131E| 131E                 ReadDip .EQU *
131E| 38                          sec
131F| B0F3                        bcs  Rid
1321|
1321| 1321                 DoStrb .equ  *
1321| 8D 0708                      sta   LStrb+High
1324| EA                          nop
1325| EA                          nop                   ; leave LStrobe low for 4 usec per G. Crow
PAGE -  27  LISA      FILE: INTERFACE.TEXT  Sony DRIVER FOR LISA


1326| 8D 0608                      sta   LStrb+Low
1329| 60                          rts
132A|
132A| 132A                 TrMtOff .equ  *               ; Will turn motors off
132A| 8D 0508                      sta   Ca2+High
132D| A9 00                        lda   #00
132F| F0**                        beq   TurnMtr
1331|
1331| 1331                 TrnMtrOn .equ  *              ; Will turn the Sony motor on
1331| 8D 0408                      sta   Ca2+low
1334| A9 FF                        lda   #Off
1336|
132F* 05
1336| 1336                 TurnMtr .equ  *
1336| 85 21                        sta   MtrOn           ; set state register in Ram
1338| 20 FA12                      jsr   SetLow          ; set Side select line low
133B| 8D 0008                      sta   Ca0+Low
133E| 4C 2113                      jmp   DoStrb
1341|
1341| 1341                 TMOn   .equ  *              ; turn on motor & setup for tach reading
1341| 20 3113                      jsr   TrnMtrOn        ; will leave Ca2 = 0
1344| A9 50                        lda   #80.            ; 400(5*80) msec constant
1346| 20 ****                      jsr   Wait
1349| 1349                 SetTach .equ  *
1349| 20 FE12                      jsr   SetHigh         ; set Side select = 1 for Tach input
134C| AD 0408                      lda   Ca2+Low         ; '1011' for Tach mode
134F| 60                          rts
1350|
1350| 1350                 SetDrctX .equ  *              ; use value in Xreg for direction
1350| 86 59                        stx   Direct
1352| 1352                 SetDrct .equ  *
1352| 20 FA12                      jsr   SetLow          ; set Side select line low
1355| 8D 0008                      sta   Ca0+Low
1358| 8D 0208                      sta   Ca1+Low
135B| A6 59                        ldx   Direct          ; 0 or 1 for offset of direction
135D| 9D 0408                      sta   Ca2,x
1360| 4C 2113                      jmp   DoStrb
1363|
1363| 1363                 DoStep .equ  *
1363| A9 01                        lda   #1
1365| 85 6E                        sta   Temp1             ; try twice
1367| 20 FA12                      jsr   SetLow            ; set Side select line low
136A| 8D 0208                      sta   Ca1+Low
136D| 8D 0408                      sta   Ca2+Low
1370| 20 2113                      jsr   DoStrb
1373| 20 ****              $10     jsr   WaitAlt           ; wait 12 ms for /Step to go high
```

```
1376| 20 0B13                    jsr   ReadIt              ; ccC = /Step --  0 = low & 1 = high
1379| B0**                       bcs   DoStDone            ; /Step is high so all is cool
137B| A9 0F                      lda   #StepErr
137D| 85 0D                      sta   Iob+SeekErr         ; for external debugging aid ( FCC01B )
137F| C6 6E                      dec   Temp1
1381| 10F0                      bpl   $10                 ; wait another 12ms
1379* 08
1383| 1383              DoStDone .equ  *                   ; fall thru & exit w/ no error
1383| 60                        rts                       ; no one there to listen to error anyway
PAGE -  28  LISA      FILE: INTERFACE.TEXT  Sony DRIVER FOR LISA


1384|
1384| 1384              DoSeek   .equ  *
1384| 20 5213                    jsr   SetDrct             ; set the direction
1387| 20 6313          $23       jsr   DoStep
138A| C6 58                      dec   StpAmt              ; count down the number of steps to take
138C| D0F9                      bne   $23                 ; exit upon reaching zero
138E| 60                        rts
138F|
138F| 138F              SetSpeed .equ  *
138F| 20 5511                    jsr   TrkClss         ; return w/ 'Y' = class of IIob+Track
1392| 1392              SetSpdy  .equ  *                ; already has 'y' = track class
1392| 84 23                      sty   CurClass
1394| B9 10 00                   lda   MSpdTbl,y
1397| 8D 2008                    sta   PwmReg
139A| 60                        rts
139B|
1252* 9B13
139B| 139B              SetRMode .equ  *
139B| 20 E612                    jsr   SelSide         ; use value in IIOB+Side to set proper side
139E| 8D 0008                    sta   Ca0+Low
13A1| 8D 0208                    sta   Ca1+Low
13A4| 8D 0508                    sta   Ca2+High        ; 'x100' is input to MCI PAL on Sony
13A7| 60                        rts
13A8|
13A8| 13A8              UnClamp  .equ  *
13A8| 20 FA12                    jsr   SetLow
13AB| 8D 0508                    sta   Ca2+High          ; '0111'
13AE| AD 0708                    lda   LStrb+High
13B1| A9 96                      lda   #150.           ; 3/4 sec (200*5usec) delay for ejection
13B3| 20 ****                    jsr   Wait            ; Will wait & return w/ carry cleared
13B6| AD 0608                    lda   LStrb+Low
13B9| A9 00                      lda   #0
13BB| 85 20                      sta   Clamped         ; tell me disk is no longer clamped
13BD| 18                        clc
13BE| 60                        rts
13BF|
13BF| 13BF              HardInit .equ  *                 ; Initialize the hardware -- PWM & IWM
13BF| A2 0E                      ldx   #0E
13C1| 9D 0008          $7        sta   IOSpace,x         ; reset all IWM internal latches to zero state
13C4| CA                        dex
13C5| CA                        dex
13C6| 10F9                      bpl   $7                ; loop 8 times
13C8| A9 CA                      lda   #202.
13CA| 20 ****                    jsr   Wait              ; wait for 1.01 sec for timer to expire
13CD|
13CD| A2 00                      ldx   #0                ; $10000 times before timeout
13CF| 86 6E                      stx   Temp1
13D1|
13D1| CA                  $15     dex
13D2| D0**                       bne   $30
13D4| C6 6E                      dec   Temp1
13D6| F0**                       beq   HardAbort         ; if timeout then IWM is out to lunch
13D8|
13D2* 04
13D8| AD 0D08              $30     lda   Q6H               ; put IWM state machine into SENSE mode
PAGE -  29  LISA      FILE: INTERFACE.TEXT  Sony DRIVER FOR LISA


13DB| AD 0E08                    lda   Q7L               ; read status data
13DE| A8                        tay                     ; save IWM's status information
13DF| 29 20                      and   #20               ; test bit #5
13E1| D0EE                      bne   $15               ; wait for bit #5 to be clear ( 0 )
13E3|
13E3| 98                        tya                     ; restore status info
13E4| 29 1F                      and   #IWMMode          ; mask off unneeded bits
13E6| C9 1F                      cmp   #IWMMode          ; look for proper bits to be set
13E8| F0**                       beq   $45               ; if same then all is OK
13EA|
13EA| A9 1F                      lda   #IWMMode          ; Asynch, 8 MHz, latch, 2 us cells, no timer
13EC| 8D 0F08                    sta   Q7H               ; setup to proper Sony mode
13EF| AD 0E08                    lda   Q7L               ; back to Sense mode
13F2| 4C D113                    jmp   $15               ; walk through & make sure it works
13F5|
13E8* 0B
13F5| 8D 0B08              $45     sta   DrEno+High        ; select drive 1
13F8| 8D 0908                    sta   MtEno+On          ; now enable that drive forever & ever...
13FB| A6 10                      ldx   MSpdTbl           ; default speed code for track 0
13FD| 8E 1008                    stx   CntEno+Low        ; enable counter/comparator
1400| 8E 2008                    stx   PWMReg
1403| 8E 1708                    stx   PwmEno+High       ; enable output of Pwm pulse
1406| 60                        rts
1407|
13D6* 2F
1407| 1407              HardAbort .equ  *
```

```
140(                                    lda   #IWMError          ; fatal error w/ IWM -- does not respond
140(                                    sta   Iob+ErrStat
140B| 4C ****                          jmp   FatalErr           ; abort and never return
140E|
140E| 140E                  ChkDrv      .equ  *                  ; check for drive connected & number of sides
140E| A9 01                             lda   #1
1410| 85 0A                             sta   IOB+NoSides        ; assume single sided drive (1 = single sided)
1412| 20 FA12                           jsr   SetLow
1415| 8D 0508                           sta   Ca2+High           ; 'O111'
1418| 38                                sec                      ; assume no drive exists
1419| AD 0E08                           lda   Q7L                ; high bit = status
141C| 30**                             bmi   $30                ; if bit7 = 1 then no drive is connected
141E| C6 24                             dec   DrvConn            ; set drive connected flag
1420| 18                                clc
1421| 8D 0008                           sta   Ca0+Low            ; now look at actual number of sides
1424| A9 02                             lda   #2                 ; assume only single sided
1426| AE 0E08                           ldx   Q7L
1429| 10**                             bpl   $23
142B| E6 0A                             inc   IOB+NoSides        ; 2 = double sided drive
142D| 09 20                             ora   #20                ; set double side flag for format field
1429* 04
142F| 85 25                  $23        sta   FmtType            ; for use in formatting
141C* 13
1431| 60                    $30        rts                      ; ccC = 0 ==> drive connected
1432|
1432|
1432|
1432|
PAGE -  30  LISA       FILE: LOOP.TEXT  Sony DRIVER FOR LISA


1432|                                   .INCLUDE LOOP
PAGE -  31  LISA       FILE: LOOP.TEXT  Sony DRIVER FOR LISA


1432|                                   .PAGE
1432|                                   ;**
1432|                                   ;                                   RESTART
1432|                                   ;
1432|                                   ; RESTART IS THE ENTRY POINT AFTER A RESET TO THE 6504. THIS IS ALSO REFERRED
1432|                                   ; TO AS A COLD START.  It will first wait for memory to be valid, then it will
1432|                                   ; setup the stack, reset the drives, clear all of the used RAM, transfer all
1432|                                   ; "soft" parameters from the ROM to the RAM, initialize all timing constants,
1432|                                   ; check for Disk In Place and clamp any disks that are there, raise the DISK
1432|                                   ; DIAG line, and then finally look for a command.
1432|                                   ;
1432|                                   ;--
1432|                                   ;
1432|                                   ; REGISTERS
1432|                                   ;  IN
1432|                                   ;       RANDOM VALUES ASSUMED
1432|                                   ;  OUT
1432|                                   ;       ALL  DESTROYED
1432|                                   ;**
1432|
PAGE -  32  LISA       FILE: LOOP.TEXT  Sony DRIVER FOR LISA


1432|                                   .PAGE
1432|
1432|                                   ;**
1432|
1432| 1432                    RESTART   .EQU  *                  ;MAIN ENTRY POINT FOR THE 6504 FROM RESET
1432|
1432|                                   ;   CLEAR DECIMAL MODE;
1432|                                   ;   DISABLE INTERUPTS TO THE 6504;
1432|                                   ;   SWITCH OFF INTERUPT TO 68K;
1432|                                   ;   SET STACK POINTER TO 0B0;
1432|                                   ;   ENABLE MEMPOY FOR 68K
1432|
1432| D8                                      CLD                ; CLEAR DECIMAL MODE
1433| 78                                      SEI                ; MASK OFF INTERUPTS
1434|                                         ;                  ; following 3 steps needed only for warm restart
1434| 8D 1C08                 DNWHWT    STA   BOOTL              ; Tell host I'm not ready yet
1437| 8D 1E08                           STA   FDIRL              ; SET FDIR LOW, NO INTERUPT YET TO THE 68K
143A| 8D 1808                           STA   DISL               ; ENABLE 68K TO READ MEMORY
143D| A2 CF                            LDX   #StackSt
143F| 9A                              TXS                      ; SET STACK TO GROW DOWN FROM 01CF
1440| 20 BF13                          jsr   HardInit           ; initialize hardware - PWM, IWM, Latches
1443| 20 ****                          jsr   RomTest            ; will never return if an error is found
1446| 20 ****                          jsr   RomTest            ; currently an RTS
1449| A9 00                            lda   #00.
144B| 20 ****                          jsr   Wait               ; wait for Sony to power-up -- about 1.25 second
144E|
144E|                                   ;   FOR X:=#LstUsed DOWNTO 000 DO              RESET VARIABLE MEMORY FOR 6504
144E|                                   ;     MEMORY,X := 000;                         THIS RESETS ALL COUNTERS TO 000
144E|
144E| A2 BF                                   LDX   #LSTUSED     ;LENGTH OF VARIABLE AREA
1450| A9 00                                   LDA   #00          ;SOMETHING TO CLEAR WITH
1452| 95 00                    RESTART1  STA   GoByte,X         ; Init to 0
1454| CA                                    DEX
1455| D0FB                                   BNE   Restart1
1457| 85 00                                  STA   GoByte       ; And do the last/first byte
1459|
1459|                                   ;   FOR X:=SHARESZ DOWN TO 0 DO                INITIALIZE SHARED VARIABLES
```

```
1459|                          ;      SHARERAM,X := SHAREROM,X;
1459|
1459| A2 0D                          LDX   #SHARESZ          ;LENGTH OF SHARED VARIABLE SPACE
145B| BD 3911                RESTART2  LDA   SHAREROM,X      ;MOVE BYTES FROM ROM
145E| 95 10                          STA   SHARERAM,X        ;AND INTO RAM
1460| CA                            DEX                     ;NEXT BYTE
1461| 10F8                          BPL   RESTART2          ;DONE YET?
1463|
1463| A2 04                          ldx   #4                ; move 5 bytes, zero based
1465| BD 4B11                Xfer1     lda   SavAdr,x
1468| 95 38                          sta   AdrMk1,x          ; initialize Address mark table
146A| BD 5011                        lda   SavDat,x
146D| 95 70                          sta   DatMk1,x          ; init Data mark table
146F| CA                            dex
1470| 10F3                          bpl   Xfer1
1472|
1472| C6 46                          dec   KOFF
PAGE -  33  LISA      FILE: LOOP.TEXT  Sony DRIVER FOR LISA


1474| A2 3F                          ldx   #CmdLeng
1476| 86 7D                          stx   Cmdx              ; index for command saving
1478| A2 80                          ldx   #SavIndex
147A| 86 7E                          stx   SaveL
147C|
147C| A5 17                          LDA   MAXDDLY
147E| 85 42                          STA   WTHIH
1480| D0**                           bne   Loop1             ; Must see if a disk is inserted
1482|
1482|                          ;       *** NOTE ***  LOOP MUST BE THE VERY NEXT ROUTINE
PAGE -  34  LISA      FILE: LOOP.TEXT  Sony DRIVER FOR LISA


1482|                          .PAGE
1482|
1482|                          ;**
1482|                          ;
1482|                          ;                                       LOOP
1482|                          ;
1482|                          ; THIS IS THE MAIN 'IDLE' LOOP WHERE THE 6504 SPENDS MOST OF ITS TIME WHEN NOT
1482|                          ; DOING ANYTHING USEFUL. THE MAIN FUNCTIONS OF THE 'LOOP' IS TO SHUT OFF THE
1482|                          ; MOTORS AFTER THEY HAVE BEEN USED, SAMPLE THE THE DIP AND BUTTON WHEN NEEDED,
1482|                          ; AND LOOK FOR COMMANDS FROM THE 68K.;
1482|                          ;
1482|                          ; REGISTERS
1482|                          ;   IN
1482|                          ;       A =       ANY VALUE
1482|                          ;       X =       ANY VALUE
1482|                          ;       Y =       ANY VALUE
1482|                          ;   OUT
1482|                          ;       LOOP NEVER EXITS, ONLY CALLS OTHER PROCEDURES
1482|                          ;
1482|                          ; CALLS
1482|                          ;       GETDIP    GET THE DIP STATUS
1482|                          ;       TRNPRK    Park the heads and turn off the motors.
1482|                          ;       CMD       INTERPRETS, SYNTAX CHECKS, AND DISPATCHES ALL COMMANDS
1482|                          ;
1482|                          ;**
PAGE -  35  LISA      FILE: LOOP.TEXT  Sony DRIVER FOR LISA


1482|                          .PAGE
1482|
1482|                          ;**
1482|                          ;       Decrement 3 byte counter.  When all 3 bytes = 0 then turn motors off
1482|                          ;       ( even if they are already off ),  & look for Disk In Place
1482|                          ;--
1482|
1482| 1482                     LOOP      .EQU  *               ; START OF MAIN LOOP
1482| E6 28                          INC   IMALIVE
1484| C6 40                          DEC   WTLOW             ; DECREMENT LOW BYTE OF MOTOR OFF WAIT
1486| D0**                           BNE   LOOP2
1488| C6 41                          DEC   WTMID             ; DECREMENT Middle byte
148A| D0**                           BNE   LOOP2
1480* 0A
148C| 148C                     Loop1     .EQU  *               ; entry point for immediate check of DIP
148C| 20 ****                        JSR   GETDIP            ; Time to make some noise and get some status
148F| C6 42                          DEC   WTHIH             ; DECREMENT HIGH BYTE
1491| D0**                           BNE   LOOP2
1493| 20 2A13                        JSR   TrMtOff           ; turn off motor and reset motor on flag to zero
1496| A5 17                          LDA   MAXDDLY
1498| 85 42                          STA   WTHIH
149A|
1491* 07
148A* 0E
1486* 12
149A| A5 00                     LOOP2     LDA   IOB+GOBYTE
149C| 10E4                          BPL   LOOP
149E| 20 ****                        JSR   CMD               ; TRY TO EXECUTE THE COMMAND
14A1| 4C 8214                        JMP   Loop              ;   GO BACK TO LOOP AND SPIN SOME MORE...
14A4|
PAGE -  36  LISA      FILE: LOOP.TEXT  Sony DRIVER FOR LISA


14A4|                           .PAGE
14A4|                          ;**
```

```
14A4|                          ;                                ROMTEST
14A4|                          ;--
14A4|
1444* A414
14A4| 14A4                    RomTest   .equ   *               ;Entry point to test checksum of EPROM
14A4| 60                             rts
14A5| A9 00                          lda  #0
14A7| A8                             tay
14A8| 85 5C                          sta  RangeL
14AA| 85 5D                          sta  RangeH
14AC| 85 43                          sta  InxPtrL
14AE| A9 10                          lda  #10
14B0| 85 44                          sta  InxPtrH
14B2| AA                             tax
14B3| 18                    $21       clc
14B4| C8                             iny                      ; y = +1
14B5| A5 5C                          lda  RangeL
14B7| 71 43                          adc  @InxPtrL,y
14B9| 85 5C                          sta  RangeL
14BB| 88                             dey                      ; y = +0
14BC| A5 5D                          lda  RangeH
14BE| 71 43                          adc  @InxPtrL,y
14C0| 85 5D                          sta  RangeH
14C2| 18                             clc
14C3| 10**                          bpl  $43
14C5| 38                             sec
14C3* 01
14C6| 26 5C                $43       rol  RangeL
14C8| 26 5D                          rol  RangeH
14CA| C8                             iny
14CB| C8                             iny                      ; y = +2
14CC| D0E5                          bne  $21
14CE| E6 44                          inc  InxPtrH
14D0| CA                             dex
14D1| D0E0                          bne  $21
14D3| A5 5C                          lda  RangeL
14D5| D0**                          bne  RomErr
14D7| A5 5D                          lda  RangeH
14D9| F0**                          beq  RomTest
14DB|
14D5* 04
14DB| A9 0B                RomErr    lda  #PErrRom
14DC* DD14
14DD| 85 0B                FatalErr  sta   Iob+DrvError
14DF| 8D 1808                        sta   DisL               ; enable 68K to read memory
14E2| 8D 1D08                        sta   BootH              ; and reset flag line also
14E5| 4C ****                        jmp   ProgErr1           ; Bad EPROM -- tell host & loop forever
14E8|
14D9* 0D
1447* E814
14E8| 14E8                    RomTest   .equ   *               ;Test for bad Rom
14E8| 60                             RTS
PAGE -  37  LISA      FILE: LOOP.TEXT  Sony DRIVER FOR LISA


14E9|
14E9|                          ;**
14E9|                          ;
14E9|                          ;                GETDIP
14E9|                          ;
14E9|                          ; REGISTERS
14E9|                          ;
14E9|                          ;       All are destroyed
14E9|                          ;
14E9|                          ; CALLS
14E9|                          ;       UPDINT    UPDATES INTERUPT STATUS AND RAISES FDIR IF NEEDED
14E9|                          ;       READDIP   READS DIP STATUS
14E9|                          ;       ENBLTEST  TEST IMSK AGINST X-REG IF DRIVE IS ENABLED
14E9|                          ; Note:
14E9|                          ;       Should code ever get modified for a two drive system, this should
14E9|                          ; be re-written to allow commands to be recieved during the one second wait.
14E9|                          ;**
PAGE -  38  LISA      FILE: LOOP.TEXT  Sony DRIVER FOR LISA


14E9|                          .PAGE
14E9|
148D* E914
14E9| 14E9                    GETDIP    .EQU  *                ; ENTRY POINT FOR GETDIP
14E9| A5 24                          lda   DrvConn
14EB| D0**                          bne   $40                ; if <> 0 then drive is connected
14ED| 20 0E14                       jsr   ChkDrv             ; check for drive connected & number of sides
14F0| B0**                          bcs   GetDip5            ; exit if still no drive connected
14F2|
14EB* 05
14F2| 8D 1C08                $40       STA   BOOTL              ; Set flag to tell host I'm busy
14F5| 20 1E13                       JSR   READDIP            ; Read the status of the DIP into Carry
14F8| B0**                          BCS   GETDIP4            ; ecC = 1 ==> no DIP -- clear Clamped Disk Flag
14FA| A5 20                       LDA   Clamped            ; Already a clamped disk?
14FC| D0**                          BNE   GETDIP5            ; No need to clamp if already clamped
14FE|
14FE| A9 C8                       lda   #OneSec            ; one second wait constant
1500| 20 ****                       jsr   Wait               ; wait for sony drive to return to it's senses
1503| 20 ****                       jsr   ReCalMtr           ; Turn on motor & make sure head is at Track 0
1506| C6 20                       dec   Clamped            ; set clamped flag to 'FF'
1508|
```

```
1508| 20 ****                    JSR   ENBLTEST        ; If drive enabled then tell host disk inserted
150B| 90**                      BCC   GETDIP5         ; If carry set then disk enabled
150D|
150D| A9 10                     LDA   #010            ; use drive 80
150F| 05 2F              GETDIP2 ORA   IST             ; Combine with current IST
1511| 85 2F                     STA   IST
1513| D0**                      BNE   GETDIP5
1515|
1515| 1515              GETDIP3 .EQU  *
14F8* 1B
1515| A9 00              GETDIP4 LDA   #00
1517| 85 20                     STA   Clamped         ; Tell me no disk is clamped
1519|
1513* 04
150B* 0C
14FC* 1B
14F0* 27
1519| 1519              GETDIP5 .EQU  *               ; Fall thru and test for button pressed
1519| 8D 1D08                   STA   BOOTH           ; Clear busy flag to host
151C|                                                 ; Fall thru & update interrupt to host ( maybe )
151C|
PAGE -  39  LISA      FILE: LOOP.TEXT  Sony DRIVER FOR LISA


151C|                       .PAGE
151C|
151C|                       ;**
151C|                       ;
151C|                       ;                                UPDINT
151C|                       ;
151C|                       ; UPDINT IS RESPONSIBLE FOR CREATING AND EVALUATING THE IST (INTERUPT STATUS)
151C|                       ; AND RAISING (OR CLEARING) INTERUPTS TO THE 68K. IST CONTAINS THE FOLLOWING
151C|                       ; INFORMATION:
151C|                       ;
151C|                       ;          7   6   5   4   3   2   1   0
151C|                       ;        +---+---+---+---+---+---+---+---+
151C|                       ;        |   |   |   |   |   |   |   |   |
151C|                       ;        +---+---+---+---+---+---+---+---+
151C|                       ;          |   |   |   |   |   |   |   |
151C|                       ;          |   |   |   |   |   |   |   +---  DRIVE 0 DISK INSERTED
151C|                       ;          |   |   |   |   |   |   +-------  DRIVE 0 BUTTON PRESSED
151C|                       ;          |   |   |   |   |   +-----------  DRIVE 0 R/W COMPLETED
151C|                       ;          |   |   |   |   +---------------  LOGICAL OR OF BITS 0,1 & 2
151C|                       ;          |   |   |   +-------------------  DRIVE 80 DISK INSERTED
151C|                       ;          |   |   +-----------------------  DRIVE 80 BUTTON PRESSED
151C|                       ;          |   +---------------------------  DRIVE 80 R/W COMPLETED
151C|                       ;          +-------------------------------  LOGICAL OR OF BITS 4,5 & 6
151C|                       ;
151C|                       ; THEN IST IS MASKED WITH IMSK WHITCH CONTAINS THE ENABLE STATUS FOR BOTH
151C|                       ; DRIVES IN BIT 7 (=DRIVE80) AND BIT 3 (=DRIVE0). IF THE RESULT IS <>0 THEN
151C|                       ; RAISE FDIR TO SIGNAL AN INTERUPT TO THE 68K
151C|                       ;
151C|                       ;
151C|                       ; REGISTERS
151C|                       ;  IN
151C|                       ;       A =       ANY VALUE;
151C|                       ;       X =       ANY VALUE;
151C|                       ;       Y =       ANY VALUE;
151C|                       ;  OUT
151C|                       ;       A =       DESTROYED;
151C|                       ;       X =       UNCHANGED;
151C|                       ;       Y =       UNCHANGED;
151C|                       ;
151C|                       ;**
PAGE -  40  LISA      FILE: LOOP.TEXT  Sony DRIVER FOR LISA


151C|                       .PAGE
151C|
151C| 151C                  UPDINT    .EQU  *             ; ENTRY POINT FOR UPDINT
151C|
151C|                       ; IST.7 := IST.6 OR IST.5 OR IST.4;
151C|
151C| A5 2F                           LDA   IST
151E| 29 70                           AND   #070
1520| F0**                           BEQ   UPDINT2       ; SKIP IF NONE IS ON
1522| 09 80                           ORA   #080          ; SET BIT 7 IF ANY bit IS ON
1524| 85 2F                           STA   IST           ; AND PUT BACK INTO IST
1526|
1526|                       ;   IF (IST.7 AND IMSK.7) THEN
1526|                       ;     FDIR := HIGH
1526|                       ;   ELSE
1526|                       ;     FDIR := LOW
1526|                       ;   RETURN FROM UPDINT
1526|
1520* 04
1526| 25 2C                 UPDINT2   AND   IMSK          ; NOW TEST AGAINST THE MASK
1528| 85 2E                           STA   OkToGo        ; REMEMBER IF AN INTERUPT IS PENDING
152A| F0**                           BEQ   UPDINT3       ; SKIP IF NO INTERUPT PENDING
152C| 8D 1F08                        STA   FDIRH         ; RAISE INTERUPT
152F| 60                             RTS
152A* 04
1530| 8D 1E08              UPDINT3   STA   FDIRL         ; CLEAR INTERUPT
1533| 60                             RTS
1534|
1534|
```

```
1534|                       .INCLUDE CLRTRK
PAGE -  41  LISA      FILE: CLRTRK.TEXT  Sony DRIVER FOR LISA


1534|                       .PAGE
1534|                       ;++
1534|                       ;       SYNC20  will clear a track partially by writing 20 usec 'FF's to
1534|                       ;       the disk. SYNC20 will use the values in the "Y" reg as the number
1534|                       ;       of sets of sync nibbles to write to the disk.
1534|                       ;
1534|                       ;       INPUT:
1534|                       ;               Y = number to write
1534|                       ;               A = Addr/Data Mark flag --> 0 == Data & 1 == Addr
1534|                       ;
1534|                       ;       OUTPUT:
1534|                       ;               X = Destroyed
1534|                       ;               Y = 'FF'
1534|                       ;               A = Unknown
1534|                       ;       "COUNTER" = 'FF'
1534|                       ;--
1534|
1534| 1534                  BsTbl     .equ  *               ;  5 20usec bytes coded in following 6 bytes
1534| FF FC F3 CF 3F FF               .byte OFF, OFC, OF3, OCF, 03F, OFF
153A|
1213* 3A15
153A| 153A                  SYNC20    .EQU  *               ; Entry point
153A| 84 29                          sty   Counter
153C| 85 6E                          sta   Temp1
153E| A9 FF                          lda   #OFF            ;
1540| 8D 0F08                        sta   Q7H             ;               Change from SENSE to WRITE LOAD
1543| A0 05                $05       ldy   #05             ; 2             Transfer 6 bytes, zero based
1545| B9 3415               $07       lda   BsTbl,y         ; 4
1548| AE 0C08               $10       ldx   Q6L             ; 4             Sense line encoded in Bit 7
154B| 10FB                          bpl   $10             ; 2,3           Wait for Bit 7 to become a '1'
154D| 8D 0D08                        sta   Q6H             ; 4
1550|
1550| 88                            dey                   ; 2
1551| 10F2                          bpl   $07             ; 2,3
1553| C6 29                         dec   Counter         ; 5
1555| D0EC                          bne   $05             ; 2,3
1557|
1557| AE 0C08               $20       ldx   Q6L             ; 4
155A| 10FB                          bpl   $20             ; 2,3
155C| 8D 0D08                        sta   Q6H             ; 4
155F| A5 6E                          lda   Temp1           ; 3
1561| D0**                          bne   $50             ; 2,3           if != 0 then use address mark values
1563| A5 70                         lda   DatMk1          ; 3     12      first data field mark
1565| 20 ****                       jsr   WrNibl          ; 6     14/6
1568| A5 71                         lda   DatMk2          ; 3     9       second data field mark
156A| 4C ****                       jmp   WrNibl          ; 3     12      RTS to caller from WrNibl
156D|
1561* 0A
156D| A5 38                $50       lda   AdrMk1          ; 3     12      first address field mark
156F| 20 ****                       jsr   WrNibl          ; 6     14/6
1572| A5 39                         lda   AdrMk2          ; 3     9       second address field mark
1574| 4C ****                       jmp   WrNibl          ; 3     12      RTS to caller from WrNibl
1577|
1577|
PAGE -  42  LISA      FILE: PRENIB.TEXT  Sony DRIVER FOR LISA


1577|                       .INCLUDE PRENIB
PAGE -  43  LISA      FILE: PRENIB.TEXT  Sony DRIVER FOR LISA


1577|                       .PAGE
1577|                       ;++
1577|                       ;               New PRENIBBLE routine
1577|                       ;       This routine will create 5 bytes ( 4 from the 524 byte i/o buffer and
1577|                       ;       one from the checksum ) and will save them in the zero page for use
1577|                       ;       during the writing operation.  It is a brute force operation, reading
1577|                       ;       each set of three bytes from absolute locations ( via equates ) and
1577|                       ;       will call a routine that will combine the top 2 bits of each into
1577|                       ;       one byte that is returned within the 'A' register.
1577|                       ;
1577|                       ;       BUFFER[1] == >AAAAAAAA<
1577|                       ;       BUFFER[2] == >BBBBBBBB<
1577|                       ;       BUFFER[3] == >CCCCCCCC<
1577|                       ;       RESULT :=    NIBL[00AABBCC]
1577|                       ;--
1577|                       ;
1577|                       ;   REGISTERS:
1577|                       ;            ALL ==  DESTROYED
1577|                       ;
1577|                       ;       Local Equates
1577|                       ;--
1577|
1577| 02FF                  C1BT01    .EQU  02FF
1577| 0300                  C1BT02    .EQU  0300
1577| 0301                  C1BT03    .EQU  0301
1577|
1577| 03FE                  C2BT01    .EQU  03FE
1577| 03FF                  C2BT02    .EQU  03FF
1577|
1577|                       ;++
1577|                       ;       Global Equates
```

```
1577|                    ;       CKSUM1
1577|                    ;       CKSUM2
1577|                    ;       CKSUM3
1577|                    ;       CPCKSUM
1577|                    ;       CPBYO1
1577|                    ;       CPBYO2
1577|                    ;--
PAGE -  44  LISA      FILE: PRENIB.TEXT  Sony DRIVER FOR LISA


1577|                    .page
1577| 1577              PRENIB    .EQU  *                    ; Entry point for all callers
1577| AC FF02                     LDY   C1BTO1
157A| AE 0003                     LDX   C1BTO2
157D| AD 0103                     LDA   C1BTO3
1580| 20 6711                     JSR   FR3TO1
1583| 85 5E                       STA   CPBYO1               ; Save new composite byte
1585|
1585| AC FE03                     LDY   C2BTO1
1588| AE FF03                     LDX   C2BTO2
158B| A9 00                       LDA   #00                  ; Only two bytes in this one
158D| 20 6711                     JSR   FR3TO1
1590| 85 5F                       STA   CPBYO2               ; Save new composite byte
1592|
1592| A4 61                       LDY   CKSUM1
1594| A6 62                       LDX   CKSUM2
1596| A5 63                       LDA   CKSUM3
1598| 20 6711                     JSR   FR3TO1
159B| 85 60                       STA   CPCKSUM              ; Save new composite byte
159D| 60                          RTS                        ; Back to caller
159E|
159E|
159E|
159E|                    .INCLUDE CMD
PAGE -  45  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


159E|                    .PAGE
159E|                    ;**
159E|                    ;
159E|                    ;                                   CMD
159E|                    ;
159E|                    ; CMD IS THE COMMAND INTERPRETER; IT LOOKS AT THE IOB GOBYTE AND EVALUATES THE
159E|                    ; COMMAND TO BE DONE. If an interrupt is pending, an "81" command will be
159E|                    ; aborted.  All commands are checked for the validity of their parameters by
159E|                    ; the "VALIDATE" routine.
159E|                    ;
159E|                    ;
159E|                    ; REGISTERS
159E|                    ;       All ==    DESTROYED
159E|                    ;
159E|                    ; CALLS
159E|                    ;       VALIDATE        VALIDATES THE PARAMETERS OF THE IOB
159E|                    ;       RWTS            HANDLES ALL COMMANDS RELATED TO THE DISK
159E|                    ;       SEEK            Moves head to Drive/Side/Track
159E|                    ;       CALL            Will call routine
159E|                    ;       CLRIST          Clears interrupt status byte (IST)
159E|                    ;       SETIMSK         Enables drives for interrupts using mask byte
159E|                    ;       CLRIMSK         Disables drives for interrupts using mask byte
159E|                    ;       WAITROM         Waits for particular GoByte sequence before Cold Restart
159E|                    ;       ESAD            Will only access ROM; loops forever or until reset
159E|                    ;
159E|                    ;**
PAGE -  46  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


159E|                    .PAGE
159E|
159E|                    ;**
159E|                    ;   FOR X:=IOBSIZE DOWNTO 0 DO                  TRANSFER IOB INTO INTERNAL COPY
159E|                    ;     IIOB,X := IOB,X;
159E|                    ;   "A" reg will be left containing the GOBYTE
159E|                    ;--
159E|
149F* 9E15
159E| 159E              CMD       .EQU  *                    ; ENTRY POINT FOR CMD
159E| A4 7D                       ldy   Cmdx
15A0| A2 07                       LDX   #IOBSIZE             ; Move only the 8 bytes that are passed
15A2| B5 00              CMD1     LDA   IOB,X                ; FETCH
15A4| 95 30                       STA   IIOB,X               ; AND DEPOSIT
15A6| 91 7E                       STA   @SaveL,y             ; for posterity
15A8| 88                          dey
15A9| CA                          DEX                        ; COUNT DOWN
15AA| 10F6                       BPL   CMD1                 ; AND LOOP
15AC|
15AC| 84 7D                       sty   Cmdx
15AE| C8                          iny
15AF| D0**                       bne   $32
15B1| A0 3F                       ldy   #CmdLeng
15B3| 84 7D                       sty   Cmdx
15B5|
15B5|                    ;   IF IIOB.GOBYTE = 81 THEN
15B5|                    ;     GOTO RWTS
15B5|
15AF* 04
15B5| C9 81              $32       CMP   #RWTSCMD             ; RWTS COMMAND?
```

```
1577|                          ;       CKSUM1
1577|                          ;       CKSUM2
1577|                          ;       CKSUM3
1577|                          ;       CPCKSUM
1577|                          ;       CPBYO1
1577|                          ;       CPBYO2
1577|                          ;--
PAGE -  44  LISA       FILE: PRENIB.TEXT  Sony DRIVER FOR LISA


1577|                          .page
1577| 1577                     PRENIB    .EQU  *                    ; Entry point for all callers
1577| AC FF02                            LDY   C1BTO1
157A| AE 0003                            LDX   C1BTO2
157D| AD 0103                            LDA   C1BTO3
1580| 20 6711                            JSR   FR3TO1
1583| 85 5E                              STA   CPBYO1               ; Save new composite byte
1585|
1585| AC FE03                            LDY   C2BTO1
1588| AE FF03                            LDX   C2BTO2
158B| A9 00                              LDA   #00                  ; Only two bytes in this one
158D| 20 6711                            JSR   FR3TO1
1590| 85 5F                              STA   CPBYO2               ; Save new composite byte
1592|
1592| A4 61                              LDY   CKSUM1
1594| A6 62                              LDX   CKSUM2
1596| A5 63                              LDA   CKSUM3
1598| 20 6711                            JSR   FR3TO1
159B| 85 60                              STA   CPCKSUM              ; Save new composite byte
159D| 60                                 RTS                        ; Back to caller
159E|
159E|
159E|
159E|                          .INCLUDE CMD
PAGE -  45  LISA       FILE: CMD.TEXT  Sony DRIVER FOR LISA


159E|                          .PAGE
159E|                          ;++
159E|                          ;
159E|                          ;                                  CMD
159E|                          ;
159E|                          ; CMD IS THE COMMAND INTERPRETER; IT LOOKS AT THE IOB GOBYTE AND EVALUATES THE
159E|                          ; COMMAND TO BE DONE. If an interrupt is pending, an "81" command will be
159E|                          ; aborted.  All commands are checked for the validity of their parameters by
159E|                          ; the "VALIDATE" routine.
159E|                          ;
159E|                          ;
159E|                          ; REGISTERS
159E|                          ;       All ==    DESTROYED
159E|                          ;
159E|                          ; CALLS
159E|                          ;       VALIDATE        VALIDATES THE PARAMETERS OF THE IOB
159E|                          ;       RWTS            HANDLES ALL COMMANDS RELATED TO THE DISK
159E|                          ;       SEEK            Moves head to Drive/Side/Track
159E|                          ;       CALL            Will call routine
159E|                          ;       CLRIST          Clears interrupt status byte (IST)
159E|                          ;       SETIMSK         Enables drives for interrupts using mask byte
159E|                          ;       CLRIMSK         Disables drives for interrupts using mask byte
159E|                          ;       WAITROM         Waits for particular GoByte sequence before Cold Restart
159E|                          ;       ESAD            Will only access ROM; loops forever or until reset
159E|                          ;
159E|                          ;++
PAGE -  46  LISA       FILE: CMD.TEXT  Sony DRIVER FOR LISA


159E|                          .PAGE
159E|
159E|                          ;++
159E|                          ;   FOR X:=IOBSIZE DOWNTO 0 DO                  TRANSFER IOB INTO INTERNAL COPY
159E|                          ;     IIOB,X := IOB,X;
159E|                          ;   "A" reg will be left containing the GOBYTE
159E|                          ;--
159E|
149F* 9E15
159E| 159E                     CMD       .EQU  *                  ; ENTRY POINT FOR CMD
159E| A4 7D                              ldy   Cmdx
15A0| A2 07                              LDX   #IOBSIZE           ; Move only the 8 bytes that are passed
15A2| B5 00                     CMD1      LDA   IOB,X              ; FETCH
15A4| 95 30                              STA   IIOB,X             ; AND DEPOSIT
15A6| 91 7E                              STA   @SaveL,y           ; for posterity
15A8| 88                                 dey
15A9| CA                                 DEX                      ; COUNT DOWN
15AA| 10F6                              BPL   CMD1               ; AND LOOP
15AC|
15AC| 84 7D                            sty   Cmdx
15AE| C8                               iny
15AF| D0**                            bne   $32
15B1| A0 3F                            ldy   #CmdLeng
15B3| 84 7D                            sty   Cmdx
15B5|
15B5|                          ;   IF IIOB.GOBYTE = 81 THEN
15B5|                          ;     GOTO RWTS
15B5|
15AF* 04
15B5| C9 81                     $32       CMP   #RWTSCMD           ; RWTS COMMAND?
```

```
15B7| F0**                          BEQ   RWTS              ; Yes & let RTS from RWTS return to LOOP
15B9|
15B9|                        ;   ELSE
15B9|                        ;     BEGIN
15B9|                        ;       IF NOT (IIOB.GOBYTE IN [VALID IIOB.GOBYTE]) THEN
15B9|                        ;         BEGIN
15B9|                        ;           IOB.GOBYTE := 001;                INVALLID COMAND
15B9|                        ;           CARRY := CLEAR;                   COMMAND COMPLETED
15B9|                        ;           RETURN
15B9|                        ;         END;
15B9|                        ;       VALIDATE;                             Validate parameters
15B9|
15B9| C9 80                          CMP   #NULLCMD          ; If null command then exit
15BB| F0**                          beq   CmdNoErr
15BD|
15BD| 38                           SEC
15BE| E9 83                        SBC   #LWCMDNO          ; SUBTRACT LOWEST COMMAND NUMBER
15C0| 90**                        BCC   CMD3              ; OOOPS!, NOT A VALID GOBYTE COMMAND
15C2| C9 08                        CMP   #CMDNUMB+1        ; NUMBER OF COMMANDS IN RANGE
15C4| 90**                        BCC   CMD5              ; SKIP IF VALID GOBYTE COMMAND
15C0* 04
15C6| A9 01                  CMD3  LDA   #GERRCMD          ; ELSE SIGNAL INVALID GOBYTE
15C8| D0**                        BNE   CMDCLNUP          ; Cleanup & exit
15CA|
15C4* 04
PAGE -  47  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


15CA| AA                     CMD5  TAX                             ; INDEX TO COMMAND ALREADY IN A-REG!
15CB| 85 2A                        STA   HOLDINX                   ; SAVE INDEX FOR THE MOMENT
15CD| BD 3211                      LDA   TESTGOB,X                 ; FETCH PARSING FOR GOBYTE COMMANDS
15D0| 20 ****                     JSR   VALIDATE                  ; AND VALIDATE IT
15D3| B0**                        BCS   CMDCLNUP                  ; NOT A VALID COMMAND
15D5| A5 2A                        LDA   HOLDINX                   ; GET BACK THE INDEX TO THE COMMAND
15D7|
15D7|                        ;       IF VALID THEN
15D7|                        ;         CASE GOBYTE OF
15D7|                        ;           83: SEEK            Seek to track
15D7|                        ;           84: CALL;           CALL 6504 ADDRESS
15D7|                        ;           85: CLRIST;         CLEAR IST (INTERUPT STATUS) BITS
15D7|                        ;           86: SETIMSK;        SET (ENABLE) BITS IN IMSK (INTERUPT MASK)
15D7|                        ;           87: CLRIMSK;        CLEAR (DISABLE) BITS IN IMSK (INTERUPT MASK)
15D7|                        ;           88: WAITROM         WAIT IN ROM (USED BY 68K RAM TEST)
15D7|                        ;           89: ESAD            Jump to self in ROM and never return
15D7|                        ;         END;(CASE)
15D7|                        ;     END;
15D7|
15D7| 0A                           ASL   A                 ; to index into word table
15D8| AA                           TAX                     ; Use command number as index
15D9| 20 ****                     JSR   CMD6
15DC| A5 2A                        LDA   HOLDINX           ; Fetch the index ( 0..6 )
15DE| F0**                        BEQ   CMDRTS            ; Skip if command was a "SEEK" command
15E0| C9 01                        CMP   #01               ; Also skip if it was a "CALL" command
15E2| F0**                        BEQ   CMDRTS
15B8* 27
15E4| 15E4                  CmdNoErr .equ  *
15E4| A9 00                        LDA   #0                ; Signal no error
15E6|
15D3* 11
15C8* 1C
15E6| 85 08                  CMDCLNUP STA  IOB+ERRSTAT     ; Common error reporting code for CMD & RWTS
15E8| A9 00                        LDA   #00
15EA| 85 00                        STA   IOB+GOBYTE        ; Allow host to process error ( if any )
15E2* 08
15DE* 0C
15EC| 60                     CMDRTS RTS                     ; AND EXIT
15ED|
15ED|                        ;--
15ED|                        ;
15ED|                        ;       CMD JUMP TABLE DISPATCHER
15ED|
15DA* ED15
15ED| BD ****                 CMD6  LDA   CMDJMP+1,X        ; FETCH THE HIGH BYTE
15F0| 48                           PHA                     ; PUT IT ONTO THE STACK
15F1| BD ****                      LDA   CMDJMP,X          ; THEN THE LOW BYTE
15F4| 48                           PHA
15F5| 60                           RTS                     ;  AND JUMP INDIRECT TOP OF STACK
15F6|                        ;--
15F6|                        ;
15F6|                        ;               TABLES FOR CMD DISPATCHER
15F6|
15F2* F615
15EE* F715
PAGE -  48  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


15F6| **** **** **** ****   CMDJMP .Word Seek-1, CALL-1, CLRIST-1, SETIMSK-1, CLRIMSK-1
15FE| ****
1600| **** ****                   .Word WaitRom-1, ESAD-1
1604|
PAGE -  49  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


1604|                        .PAGE
1604|                        ;**
1604|                        ;
```

```
1604                            ;                                       RWTS
1604                            ;
1604                            ; RWTS (READ WRITE TRACK SECTOR) IS THE ROUTINE THAT PARSES AND DISPATCHES
1604                            ; ALL COMMANDS THAT AFFECT THE DISK ( except SEEK ).  If no interrupts are
1604                            ; pending and all parameters are valid, then the memory will be disabled to
1604                            ; access by the 68K and the proper routine will be called.
1604                            ;
1604                            ; REGISTERS
1604                            ;  IN
1604                            ;       A =         IIOB+COBYTE
1604                            ;
1604                            ;  OUT
1604                            ;     ALL ==        DESTROYED
1604                            ;
1604                            ; CALLS
1604                            ;       VALIDATE        VALIDATES PARAMETERS FOR THE CMD, DRIVEN BY TESTTBL
1604                            ;       INIT            SET UP GLOBAL CONDITIONS USED BY MOST COMMANDS
1604                            ;       READ            Reads Drive/Side/Track/Sector
1604                            ;       WRITE           Writes Drive/Side/Track/Sector
1604                            ;       UNCLAMP         Unclamps disk in Drive
1604                            ;       FORMAT          Formats disk in Drive starting at Side/Track
1604                            ;       VERIFY          Reads disk in Drive starting at Side/Track
1604                            ;       FORMTRAK        Formats single Track on disk in Drive
1604                            ;       VERTRACK        Reads single Track on disk in Drive
1604                            ;       READBF          Reads Drive/Side/Track/Sector w/o checksum verify
1604                            ;       WRITEBF         Writes Drive/Side/Track/Sector w/o checksum creation
1604                            ;       CLPENTY         Will clamp disk in Drive
1604                            ;
1604                            ;**
PAGE -  50  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


1604                            .PAGE
1604
15B7* 4B
1604  1604                      RWTS      .EQU  *               ; ENTRY POINT FOR RWTS
1604
1604                            ;   VALIDATE IIOB.COMMAND;
1604                            ;   IF NOT VALID THEN
1604                            ;     BEGIN
1604                            ;       IOB.COBYTE := COMMAND ERROR
1604                            ;       CARRY := CLEAR
1604                            ;       RETURN FROM RWTS
1604                            ;     END
1604                            ;       VALIDATE;
1604                            ;       IF NOT VALID THEN RETURN FROM RWTS
1604                            ;       ELSE
1604                            ;         BEGIN
1604                            ;           LOCK MEMORY;
1604                            ;           INIT;
1604                            ;           Call command
1604                            ;           IST := R/W COMMAND DONE;
1604                            ;           UPDINT
1604                            ;         END
1604                            ;     END;
1604                            ;   RETURN FROM RWTS
1604                            ;**
1604
1604  A5 2E                             LDA   OKTOGO          ; If >< 0 then interrupt is pending
1606  F0**                              BEQ   AOK
1608  A9 09                             LDA   #GERRINTR       ; Pending interrupt error
160A  20 E615                   NOTAOK    JSR   CMDCLNUP      ; Report error
160D  F0**                              BEQ   RW400           ; Send an interrupt back to host
160F
1606* 07
160F  A6 31                     AOK       LDX   IIOB+COMMAND  ; FETCH THE COMMAND NUMBER
1611  E0 0A                             CPX   #MAXCMD+1       ; COMPARE AGAINST MAXIMUM COMMAND
1613  90**                              BCC   RWTS4           ; SKIP IF VALID
1615  A9 01                             LDA   #GERRCMD        ; SIGNAL A COMMAND ERROR
1617  D0F1                              BNE   NOTAOK          ; Local error exit handler
1619
1613* 04
1619  BD 2811                   RWTS4     LDA   TESTTBL,X     ; Fetch bit mask of parameters to check
161C  20 ****                           JSR   VALIDATE        ; VALIDATE RETURNS CARRY CLEAR IF OK
161F  20 E615                           JSR   CMDCLNUP        ; Will clear GOBYTE and return error code
1622  B0**                              BCS   RW400           ; If error then return FDIRH
1624
1624  20 ****                   RWTS40    JSR   INIT          ; SET UP GLOBAL CONDITIONS USED BY MOST COMMANDS
1627  A5 31                             LDA   IIOB+COMMAND    ; Load with command byte
1629  0A                                ASL   A               ; Mult by two to address word array
162A  AA                                TAX                   ; Put into index register
162B  20 ****                           JSR   RWTS7           ; PUSH RETURN ADRESS ON STACK AND EXECUTE CMD
162E
162E  8D 1808                           STA   DISL            ; UNLOCK MEMORY
1631  8D 1D08                           sta   BootH           ; tell 68K that memory is renabled
1634  90**                              BCC   RWTS5           ; If carry clear then no error
PAGE -  51  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


1636  85 08                             STA   IOB+ERRSTAT     ; If error then must report this to host
1638
1622* 14
160D* 29
1638  1638                      rw400     .equ  *
1638  A4 7D                             ldy   CmdX            ; and save for posterity
163A  C0 3F                             cpy   #CmdLeng
```

```
163C| D0**                          bne   $10
163E| A0 06                         ldy   #6
1640| D0**                          bne   $20
163C* 04
1642| 98                    $10     tya
1643| 18                            clc
1644| 69 07                         adc   #7              ; point at 'speed' byte
1646| A8                            tay
1640* 05
1647| A5 08                $20     lda   IOB+ErrStat
1649| 91 7E                         sta   @SaveL,y
164B|
1634* 15
164B| A9 40                RWTS5   LDA   #040            ; use Drive 80
164D| 05 2F                        ORA   IST             ; SET BIT IN INTERUPT STATUS
164F| 85 2F                        STA   IST
1651| 4C 1C15                      JMP   UPDINT
1654|
1654|                      ;--
1654|                      ;
1654|                      ;       RWTS COMMAND DISPATCHER
1654|
162C* 5416
1654| BD ****              RWTS7   LDA   RWTSJMP+1,X     ; FETCH THE HIGH BYTE OF THE ROUTINE
1657| 48                            PHA                   ; PUSH IT
1658| BD ****                       LDA   RWTSJMP,X       ; FETCH THE LOW BYTE TO THE ROUTINE
165B| 48                            PHA                   ; PUSH IT TO THE STACK
165C| 8D 1C08                       STA   BootL           ; tell 68K that memory is disabled
165F| 8D 1908                       STA   DISH            ; Disable access at last possible moment
1662| 60                            RTS                   ; POP ADRESS AND JUMP TO IT
1663|
1663|                      ;--
1663|                      ;
1663|                      ; RWTS DISPATCHER JUMP TABLE
1663|
1659* 6316
1655* 6416
1663| **** **** A713       RWTSJMP .Word READ-1, WRITE-1, UNCLAMP-1
1669| **** **** **** ****          .Word FORMAT-1, VERIFY-1, FORMTRAK-1, VERTRACK-1
1671| **** **** ****               .Word READBF-1, WRITEBF-1, CLPENTY-1
1677|
PAGE -  52  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


1677|                      .PAGE
1677|                      ;**
1677|                      ;
1677|                      ;                                VALIDATE
1677|                      ;
1677|                      ; VALIDATES THE IOB PARAMETERS PASSED FROM THE 68K DEPENDING ON THE
1677|                      ; BIT PATTERN IN THE A-REG.
1677|                      ;
1677|                      ;
1677|                      ; REGISTERS
1677|                      ;  IN
1677|                      ;       A =     PARAMETERS IN THE IOB TO BE TESTED:
1677|                      ;
1677|                      ;                 7   6   5   4   3   2   1   0
1677|                      ;               +---+---+---+---+---+---+---+---+
1677|                      ;               |   |   |   |   |   |   |   |   |
1677|                      ;               +---+---+---+---+---+---+---+---+
1677|                      ;                 |   |   |   |   |   |   |   |
1677|                      ;                 |   |   |   |   |   |   |   +--- Drive # & a clamped disk
1677|                      ;                 |   |   |   |   |   |   +------- Side #
1677|                      ;                 |   |   |   |   |   +----------- Sector #
1677|                      ;                 |   |   |   |   +--------------- Track #
1677|                      ;                 |   |   |   +------------------- Mask
1677|                      ;                 |   |   +----------------------- Confirmation byte
1677|                      ;                 |   +--------------------------- Write Protection
1677|                      ;                 +------------------------------- Format/Verify parameters
1677|                      ;       X =     ANY VALUE
1677|                      ;       Y =     ANY VALUE
1677|                      ;  OUT
1677|                      ;       A =     DESTROYED
1677|                      ;       X =     DESTROYED
1677|                      ;       Y =     DESTROYED
1677|                      ;
1677|                      ;  CALLS
1677|                      ;       ENBLTEST        Test for drive being enabled for interrupts
1677|                      ;       TRKCLSS         Returns w/ "Y" = class of Track
1677|                      ;       READWP          Tests for a write protected disk
1677|                      ;--
1677|
PAGE -  53  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


1677|                      .PAGE
1677|
161D* 7716
15D1* 7716
1677| 1677                 VALIDATE .EQU  *              ; ENTRY POINT FOR VALIDATE
1677| 85 6E                        STA   TEMP1           ; SAVE "A" HERE DURING VALIDATION
1679| F0**                         BEQ   VLDNOTST        ; If zero then no testing required
167B| A2 0E                        LDX   #0E             ; TEST BITS 7-0, use 14. to index word array
167D| 46 6E              VALID1   LSR   TEMP1           ; SHIFT AND TEST THE CARRY BIT
167F| 90**                         BCC   VALID20         ; Bit not set; skip this test
```

```
1681| 86 6F                    STX   TEMP2           ; Save "X" counter value
1683| BD ****                  LDA   VALJMP+1,X      ; PICK UP THE HIGH BYTE
1686| 48                       PHA                   ; PUT IT ON THE STACK
1687| BD ****                  LDA   VALJMP,X        ; PICK UP THE LOW BYTE
168A| 48                       PHA                   ; PUT IT TOO ON THE STACK
168B| 60                       RTS                   ; JMP TO THE TEST
168C|
168C|                  ; THIS IS THE RETURN POINT AFTER A SUCCESSFUL TEST
168C|
168C| A6 6F            VALID2  LDX   TEMP2           ; Restore "X" counter value
167F* 0D
168E| CA                VALID20 DEX                   ; Loop count - 2 = next test address index
168F| CA                       DEX
1690| 10EB                     BPL   VALID1          ; Will loop until index is negative
1679* 17
1692| A9 00            VLDNOTST LDA  #0
1694| 18                       CLC                   ; Signal no errors
1695| 60                       RTS
1696|
1696|                  ; THIS IS THE RETURN POINT AFTER A FAILED TEST
1696|
1696| 38                VALID3  SEC                   ; SOMETHING WENT WRONG
1697| 60                       RTS
1698|
1698|                  ;--
1698|                  ;
1698|                  ;       TEST DRIVE NUMBER, IF THE DRIVE IS ENABLED AND FOR A CLAMPED DISK
1698|
1698| A5 32            VALIDDR  lda  IIob+Drive
169A| C9 80                     cmp   #80             ; must use 'lower' drive value
169C| D0**                     bne   $28
169E| A5 20                    LDA   Clamped
16A0| D0**                     BNE   $57             ; YES, THERE IS A CLAMPED DISK
169C* 04
16A2| A9 07            $28     LDA   #GERRCLM        ; No disk in drive error
16A4| D0F0                     BNE   VALID3          ; ALWAYS TAKEN
16A6|
16A0* 04
16A6| 20 ****            $57     JSR   ENBLTEST        ; TEST IF THE DRIVE IS ENABLED
16A9| B0E1                     BCS   VALID2          ; YES, THE DRIVE IS ENABLED
16AB| A9 08                    LDA   #GERRENA        ; DRIVE NOT ENABLED
16AD| D0E7                     BNE   VALID3          ; ALWAYS TAKEN
16AF|
16AF|                  ;--
PAGE -  54  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


16AF|                  ;
16AF|                  ;       TEST SIDE NUMBER
16AF|
16AF| A9 03            VALIDSI  LDA   #GERRSID        ; assume error
16B1| A4 33                     ldy   IIOB+SIDE
16B3| F0D7                     beq   Valid2          ; side 0 is always good
16B5| C0 01                    cpy   #01             ; now see if side 1
16B7| D0DD                     bne   Valid3          ; not side 1 either
16B9| A0 02                    ldy   #02
16BB| C4 0A                    cpy   IOB+NoSides     ; will range from 0 to 2 -- must be 2 for dual sided
16BD| D0D7                     bne   Valid3          ; not a dual sided drive
16BF| F0CB                     beq   Valid2
16C1|
16C1|                  ;--
16C1|                  ;
16C1|                  ;       TEST SECTOR NUMBER
16C1|
16C1| 20 5511           VALIDSE  JSR   TRKCLSS         ; Find track class
16C4| A5 34                     LDA   IIOB+SECTOR     ; New sector number
16C6| D9 4611                   CMP   SECPRTRK,Y      ; Compare against max sector per track class
16C9| 90C1                     BCC   VALID2          ; OK, WITHIN LIMITS
16CB| A9 04                    LDA   #GERRSEC        ; OOPS, TOO MANY
16CD| D0C7                     BNE   VALID3          ; ALWAYS TAKEN
16CF|
16CF|                  ;--
16CF|                  ;
16CF|                  ;       TEST TRACK NUMBER
16CF|
16CF| A5 35            VALIDTR  LDA   IIOB+TRACK      ; FETCH THE TRACK
16D1| C9 50                     CMP   #MAXTRACK+1     ; AND TEST AGAINST THE MAXIMUM TRACK NUMBER
16D3| 90B7                     BCC   VALID2          ; OK, NOTHING WRONG SO FAR
16D5| A9 05                    LDA   #GERRTRK        ; TRACK NUMBER IS TO HIGH, SIGNAL ERROR
16D7| D0BD                     BNE   VALID3          ; ALWAYS TAKEN
16D9|
16D9|                  ;--
16D9|                  ;
16D9|                  ;       TEST MASK
16D9|
16D9| A5 31            VALIDMA  LDA   IIOB+MASK       ; WE ONLY ALLOW SET/CLEAR OF BITS 7 AND 3
16DB| 29 77                     AND   #077            ; SO TEST THE REST (0,1,2,4,5 & 6) FOR 0
16DD| F0AD                     BEQ   VALID2          ; OK, NO FUNNY BITS
16DF| A9 06                    LDA   #GERRMSK        ; ELSE WE HAVE AN ERROR
16E1| D0B3                     BNE   VALID3          ; ALWAYS TAKEN
16E3|
16E3|                  ;--
16E3|                  ;
16E3|                  ;       TEST CONFIRMATION BYTE
16E3|
16E3| E6 07            VALIDCF  INC   IOB+FMTCNFM     ; If correct byte, this will inc to zero
```

```
16E5| F0A5                          BEQ   VALID2          ; Ok, byte was = FF
16E7| A9 0A                         LDA   #GERRFMPR       ; Wrong byte
16E9| D0AB                          BNE   VALID3          ; ALWAYS TAKEN
16EB|
16EB|                       ;--
16EB|                       ;
PAGE -  55  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


16EB|                       ;       TEST FOR WRITE PROTECTION
16EB|
16EB|
16EB| 20 0213             VALIDWP   JSR   READWP          ; Test for a write-protected disk
16EE| B09C                          BCS   VALID2          ; Carry = 1 ==> not protected
16F0| A9 14                         LDA   #SERRPROT       ; Write protection error constant
16F2| D0A2                          BNE   VALID3          ; ALWAYS TAKEN
16F4|
16F4|                       ;--
16F4|                       ;
16F4|                       ;       TEST FOR format & verify parameters
16F4|
16F4|
16F4| A5 35               VALIDFV   LDA   IIOB+TRACK
16F6| 05 33                         ORA   IIOB+SIDE       ; MUST HAVE TRK/SIDE = 0
16F8| F092                          BEQ   Valid2          ; PARMS ARE OK-CONTINUE
16FA| A9 0A                         LDA   #GERRFMPR       ; TRACK/SIDE<>0-ERROR
16FC| D098                          BNE   VALID3          ; ALWAYS TAKEN
16FE|
16FE|                       ;--
16FE|                       ;
16FE|                       ; VALIDATION JUMP TABLE
16FE|                       ;       Table must be in this order -- Drive must be validated and selected
16FE|                       ;       prior to checking for write protection.
16FE|
1688* FE16
1684* FF16
16FE| F316 EA16 E216 D816   VALJMP    .Word VALIDFV-1, VALIDWP-1, VALIDCF-1, VALIDMA-1, VALIDTR-1
1706| CE16
1708| C016 AE16 9716                 .Word VALIDSE-1, VALIDSI-1, VALIDDR-1
170E|
170E|
PAGE -  56  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


170E|                       .PAGE
170E|                       ;**
170E|                       ;
170E|                       ;                                   ENBLTEST
170E|                       ;
170E|                       ; TEST IMSK AGAINST X-REG TO FIND OUT IF THE DRIVE INDEXED BY X IS ENABLED.
170E|                       ;
170E|                       ;--
170E|                       ;
170E|                       ; REGISTERS
170E|                       ;  IN
170E|                       ;       A =       ANY VALUE
170E|                       ;       X =       =0, DRIVE 0; =2, DRIVE 80
170E|                       ;       Y =       ANY VALUE
170E|                       ;  OUT
170E|                       ;       A =       DESTROYED
170E|                       ;       X =       UNCHANGED
170E|                       ;       Y =       UNCHANGED
170E|                       ;       CARRY =   =CLEAR, DRIVE NOT ENABLED; =SET DRIVE ENABLED
170E|                       ;
170E|                       ;--
170E|
16A7* 0E17
1509* 0E17
170E| 170E                 ENBLTEST  .EQU  *               ; ENTRY POINT FOR ENBLTEST
170E|
170E| A9 80                          LDA   #080            ; DRIVE 80
1710| 25 2C                 ENBLTST1  AND   IMSK            ; TEST AGAINST INTERUPT MASK
1712| 69 F8                          ADC   #0F8            ; CAUSE OVERFLOW TO CARRY IF 008 OR LARGER
1714| 60                             RTS
1715|
PAGE -  57  LISA      FILE: CMD.TEXT  Sony DRIVER FOR LISA


1715|                       .PAGE
1715|                       ;**
1715|                       ;
1715|                       ;                                   INIT
1715|                       ;
1715|                       ; SET UP GLOBAL CONDITIONS USED BY MOST RWTS COMMANDS
1715|                       ;
1715|                       ; REGISTERS
1715|                       ;  IN
1715|                       ;       A =       ANY VALUE
1715|                       ;       X =       ANY VALUE
1715|                       ;       Y =       ANY VALUE
1715|                       ;  OUT
1715|                       ;       A =       DESTROYED
1715|                       ;       X =       DESTROYED
1715|                       ;       Y =       UNCHANGED
1715|                       ;
1715|                       ;**
```

```
1715|                          ;
1715|                          ;   RETRYCNT := MAXRETRY;
1715|                          ;   RECALCNT := MAXRECAL;
1715|                          ;   ERROR COUNTERS := 0;
1715|
1625* 1517
1715| 1715                     INIT      .EQU  *                 ; ENTRY POINT FOR INIT
1715| A5 19                              LDA   MAXRETRY
1717| 85 26                              STA   RETRYCNT
1719| A5 1A                              LDA   MAXRECAL
171B| 85 27                              STA   RECALCNT
171D| A5 17                              LDA   MAXDDLY           ; Motor off delay time
171F| 85 42                              STA   WTHIH
1721| A5 33                              lda   IIOB+Sidc
1723| F0**                               beq   $09
1725| A9 20                              lda   #20               ; if sidc 1 then set only bit 5
1727| 85 33                              sta   IIOB+Sidc
1723* 04
1729| A9 00                    $09       LDA   #00
172B| 85 40                              STA   WTLOH             ; reset 3 byte counter
172D| 85 41                              STA   WTMID
172F| 85 7B                              STA   RtyFlg            ; when = 2 then abort read or write operation
1731| A2 07                              LDX   #ERRLEN           ; Number of error counters
1733| 95 48                    $12       STA   STSLP,X           ; Zero the counter
1735| CA                                 DEX
1736| 10FB                               BPL   $12               ; Zero based index
1738| 60                                 RTS
1739|
PAGE -  58  LISA       FILE: CMD.TEXT  Sony DRIVER FOR LISA


1739|                          .PAGE
1739|                          ;**
1739|                          ;
1739|                          ;                                  CALL
1739|                          ;
1739|                          ; CALLS A ROUTINE IN THE ROM OR RAM AS SPECIFIED BY THE ADDRESS IN ADRL,ADRH
1739|                          ; THIS IS MOSTLY USED BY DIAGNOSTICS. YOU CAN ALWAYS RETURN FROM THE CALL BY
1739|                          ; EXECUTING A RTS.
1739|                          ;
1739|                          ;
1739|                          ; REGISTERS
1739|                          ;  IN
1739|                          ;       A =       ANY VALUE
1739|                          ;       X =       ANY VALUE
1739|                          ;       Y =       ANY VALUE
1739|                          ;  OUT
1739|                          ;       A =       DESTROYED
1739|                          ;       X =       DESTROYED
1739|                          ;       Y =       DESTROYED
1739|                          ;
1739|                          ;**
1739|
1739|
1739|                          ;  FIRST MAKE SURE THAT THE MOTORS ARE OFF
1739|                          ;   THEN JUMP OFF INTO THE BLUE...
1739|
15F8* 3817
1739| 1739                     CALL      .EQU  *                   ; ENTRY POINT FOR CALL
1739| 20 ****                            JSR   PRKCLR0             ; Park heads, turn off motors & clear GOBYTE
173C| 6C 3100                           JMP   @IIOB+ADRL          ; JUMP INDIRECT ON THE ADDRESS IN THE IIOB
173F|
PAGE -  59  LISA       FILE: CMD.TEXT  Sony DRIVER FOR LISA


173F|                          .PAGE
173F|
173F|                          ;**
173F|                          ;                                  CLRIST
173F|                          ;
173F|                          ; CLEARS SELECTED BITS IN THE IST(INTERUPT STATUS) AND THEN CHECKS IF FDIR
173F|                          ; SHOULD STILL BE HIGH. MASK SHOULD CONTAIN A 1 FOR THE BITS IN IST TO
173F|                          ; BE SWITCHED OFF.
173F|                          ;
173F|                          ; REGISTERS
173F|                          ;  IN
173F|                          ;       A =       ANY VALUE
173F|                          ;       X =       ANY VALUE
173F|                          ;       Y =       ANY VALUE
173F|                          ;  OUT
173F|                          ;       A =       DESTROYED
173F|                          ;       X =       UNCHANGED
173F|                          ;       Y =       UNCHANGED
173F|                          ;
173F|                          ; CALLS
173F|                          ;       UPDINT    UPDATES FDIR DEPENDING ON IST
173F|                          ;
173F|                          ;**
173F|
15FA* 3E17
173F| 173F                     CLRIST    .EQU  *                 ; ENTRY POINT FOR CLRIST
173F| A5 31                              LDA   IIOB+MASK         ; FETCH THE MASK
1741| 49 FF                              EOR   #OFF              ; ONES COMPLEMENT
1743| 25 2F                              AND   IST               ; AND WITH IST TO SWITCH OFF SELECTED BITS
1745| 85 2F                              STA   IST               ; SAVE NEW IST
1747| 4C 1C15                  CLREXIT   JMP   UPDINT            ; CALCULATE THE NEW STATUS OF FDIR
```

```
174A|                          .PAGE
174A|
174A|                          ;**
174A|                          ;
174A|                          ;                                   SETIMSK
174A|                          ;
174A|                          ; SETS (ENABLES) THE INTERUPT MASK FOR DRIVE 0 AND DRIVE 80 AND COMPUTES THE
174A|                          ; NEW STATUS OF FDIR.
174A|                          ;
174A|                          ; REGISTERS
174A|                          ;  IN
174A|                          ;       A =       ANY VALUE
174A|                          ;       X =       ANY VALUE
174A|                          ;       Y =       ANY VALUE
174A|                          ;  OUT
174A|                          ;       A =       DESTROYED
174A|                          ;       X =       UNCHANGED
174A|                          ;       Y =       UNCHANGED
174A|                          ;
174A|                          ; CALLS
174A|                          ;       UPDINT    UPDATE FDIR STATUS
174A|                          ;
174A|                          ;    IMSK := $88
174A|                          ;    CALL UPDINT -- WILL RETURN FROM UPDINT TO CALLER
174A|                          ;**
174A|
15FC* 4917
174A| 174A                     SETIMSK .EQU    *                   ; ENTRY POINT FOR SETIMSK
174A| A5 31                              LDA   IIob+Mask
174C| 05 2C                              ora   Imsk
174E| 85 2C                              STA   IMSK                ; SAVE NEW IMSK VALUE
1750| 4C 1C15                           JMP   UPDINT              ; CALCULATE THE NEW STATUS OF FDIR
1753|
```



```
1753|                          .PAGE
1753|                          ;**
1753|                          ;
1753|                          ;                                   CLRIMSK
1753|                          ;
1753|                          ;
1753|                          ; CLEARS (DISABLES) THE IMSK (INTERUPT MASK) SELECTIVLY. COMPUTES NEW STATUS
1753|                          ; OF THE FDIR.
1753|                          ;
1753|                          ; REGISTERS
1753|                          ;  IN
1753|                          ;       A =       ANY VALUE
1753|                          ;       X =       ANY VALUE
1753|                          ;       Y =       ANY VALUE
1753|                          ;  OUT
1753|                          ;       A =       DESTROYED
1753|                          ;       X =       UNCHANGED
1753|                          ;       Y =       UNCHANGED
1753|                          ;
1753|                          ; CALLS
1753|                          ;       UPDINT    UPDATE FDIR
1753|                          ;
1753|                          ;    CALL UPDINT -- WILL RETURN FROM UPDINT TO CALLER
1753|                          ;**
1753|
15FE* 5217
1753| 1753                     CLRIMSK   .EQU  *                   ; ENTRY POINT FOR CLRIMSK
1753| A5 31                              LDA   IIob+Mask
1755| 49 FF                              eor   #0FF
1757| 25 2C                              and   Imsk
1759| 85 2C                              STA   IMSK                ; SAVE NEW IMSK VALUE
175B| 4C 1C15                           JMP   UPDINT              ; CALCULATE THE NEW STATUS OF FDIR
175E|
175E|
175E|                          .INCLUDE SEEK
```



```
175E|                          .PAGE
175E|                          ;**
175E|                          ;                                   SEEK
175E|                          ;--
175E|                          ;
175E|                          ; REGISTERS
175E|                          ;  OUT
175E|                          ;       ALL =     DESTROYED
175E|                          ;
175E|                          ; CALLS
175E|                          ;
175E|                          ;**
```



```
175E|                          .PAGE
175E|
175E|
15F6* 5D17
```

```
175E| 175E                  Seek      .equ  *
175E| 20 1517                         jsr   Init            ; Global setup routine & return w/ 'a' = 0
1761| 20 E615                         jsr   CmdClnUp        ; Clear GoByte and ErrStat
1764|
1764| 1764                  SeekAlt   .equ  *             ; Alternate entry w/o 'INIT' of variables
1764| A2 00                         ldx   #00
1766| 86 56                         stx   TrkFlg          ; assume no head movement & motor already on
1768| 86 57                         stx   MtrFlg
176A| 86 59                         stx   Direct          ; assume '0' (toward spindle) direction
176C| CA                           dex
176D| 86 0C                         stx   HostSeek        ; tell 68K that I am seeking
176F|
176F| A5 21                         lda   MtrOn           ; if = 'ff' then motor is already on
1771| D0**                         bne   $20
1773| C6 57                         dec   MtrFlg          ; tell me to wait full 400 msec
1775| 20 3113                       jsr   TrnMtrOn        ; turn the motor on
1778|
1771* 05
1778| A5 35                  $20       lda   IIob+Track
177A| C5 22                         Cmp   CurTrack
177C| D0**                         bne   Seek1           ; wrong track
177E| A5 57                         lda   MtrFlg
1780| F0**                         beq   SeekEnd         ; same track & motor already on so select side
1782| D0**                         bne   JstMtr          ; just wait for the motor to settle
1784|
177C* 06
1784| 1784                  Seek1     .EQU  *             ; 'A' has IIob+Track already
1784| C6 56                         dec   TrkFlg          ; indicate that head needs positioning
1786| 38                           sec
1787| E5 22                         sbc   CurTrack        ; abs(destination - current) = amt to move
1789| B0**                         bcs   pstv            ; if ccC = 1 then positive result
178B| 49 FF                         eor   #0ff
178D| 69 01                         adc   #1              ; take 2's complement
178F| E6 59                         inc   Direct          ; set outward direction (away from spindle)
1789* 06
1791| 1791                  pstv      .equ  *
1791| 85 58                         sta   StpAmt
1793| A5 16                         lda   HeaDelay        ; assume only wait for head settling
1795| 85 69                         sta   Delay
1797|
1797| 1797                  Seek2     .equ  *
1797| 20 5511                       jsr   TrkClss         ; return w/ 'Y' = class of IIob+Track
179A| C4 23                         cpy   CurClass
179C| F0**                         beq   $60
179E| 20 9213                       jsr   SetSpdy         ; set the speed w/ 'Y" = trk class index
17A1| A5 58                         lda   StpAmt
17A3| 20 ****                       jsr   ClcScDly        ; calc speed change delay time, return in 'A'
17A6| 85 69                         sta   Delay
179C* 0A
17A8| 20 8413                  $60       jsr   DoSeek          ; finally do the actual seek
PAGE -  64  LISA      FILE: SEEK.TEXT  Sony DRIVER FOR LISA

17AB|
1782* 27
17AB| A5 1C                  JstMtr    lda   MOnDly
17AD| A6 57                         ldx   MtrFlg
17AF| D0**                         bne   $80
17B1| A5 69                         lda   Delay
17AF* 02
17B3| 20 ****                  $80       jsr   Wait            ; wait for Motor, speed change & head settling
17B6| A5 35                         lda   IIob+Track
17B8| 85 22                         sta   CurTrack
17BA| 20 ****                       jsr   SpdChk          ; check the speed & adjust if necessary
1780* 3B
17BD| 17BD                  SeekEnd   .equ  *
17BD| 20 3113                       jsr   TrnMtrOn        ; for floky PAL problems
17C0| A2 00                         ldx   #0
17C2| 86 0C                         stx   HostSeek        ; No more seeking
17C4| 4C E612                       jmp   SelSide         ; select proper side
17C7|
17C7|                        ;**
17C7|                        ;
17C7|                        ;                                   PROGERR
17C7|                        ;
17C7|                        ; This code will do many things.  It will either deselect the drives, turn off
17C7|                        ; the motors, park the heads first, clear the Gobyte and ErrStat, and maybe
17C7|                        ; jump to itself forever.
17C7|                        ;
17C7|                        ;
17C7|                        ; REGISTERS
17C7|                        ;  IN
17C7|                        ;       A =       ANY VALUE
17C7|                        ;       X =       ANY VALUE
17C7|                        ;       Y =       ANY VALUE
17C7|                        ;  OUT
17C7|                        ;     ALL =       DESTROYED
17C7|                        ;
17C7|                        ;--
17C7|
1602* C617
17C7| 17C7                  ESAD      .EQU  *             ; Used by drop dead command
17C7| 20 ****                       JSR   PRKCLR0         ; turn off motor and clear the GOBYTE
14E6* CA17
17CA| 4C CA17                PROGERR1  JMP   PROGERR1        ; LOOP FOREVER!
17CD|
```

```
17C8* CD17
173A* CD17
17CD  17CD                PRKCLR0   .EQU  *              ; Park the heads and clear the GOBYTE
17CD  20 2A13                       JSR   TrMtOff
17D0  A9 00                         lda   #00
17D2  4C E615                       JMP   CMDCLNUP
17D5
17D5
17D5                      .INCLUDE WAITROM
PAGE -  65  LISA      FILE: WAITROM.TEXT  Sony DRIVER FOR LISA


17D5                      .PAGE
17D5                      ;**
17D5                      ;
17D5                      ;                                   WAITROM
17D5                      ;
17D5                      ; WAITROM SWITCHES OFF BOTH MOTORS AND THEN WAITS UNTIL THE 68000 DIAGNOSTICS
17D5                      ; SETS LOCATION 000 (68K =  000C00) FIRST TO #069 AND THEN TO #096. THE 6504
17D5                      ; WILL THEN DO A COLD START OF THE SYSTEM.
17D5                      ;
17D5                      ;--
17D5                      ;
17D5                      ; REGISTERS
17D5                      ;  IN
17D5                      ;       A =       ANY VALUE
17D5                      ;       X =       ANY VALUE
17D5                      ;       Y =       ANY VALUE
17D5                      ;  OUT
17D5                      ;       RETURNS VIA A COLD START.
17D5                      ;
17D5                      ;**
17D5
1600* D417
17D5  17D5                WAITROM   .EQU  *              ; ENTRY POINT FOR WAITROM
17D5  20 CD17                       JSR   PRKCLR0        ; Park heads, turn motors off & clear GOBYTE
17D8
17D8                      ;   LOOP until 69/96 sequence occurs
17D8
17D8  A5 00               WAITROM1  LDA   000             ; WAIT FOR #069
17DA  C9 69                         CMP   #069
17DC  D0FA                          BNE   WAITROM1
17DE
17DE  C5 00               WAITROM2  CMP   000             ; WAIT FOR #096
17E0  F0FC                          BEQ   WAITROM2
17E2
17E2  38                            SEC
17E3  65 00                         ADC   000             ; WILL BE 0 IF = 96
17E5  D0F1                          BNE   WAITROM1
17E7  4C 3414                       JMP   DNWHWT          ; Reset the world w/o memory wait
17EA
17EA
17EA
17EA                      .INCLUDE NREAD16
PAGE -  66  LISA      FILE: NREAD16.TEXT  Sony DRIVER FOR LISA


17EA                      .PAGE
17EA                      ;**
17EA                      ;
17EA                      ;                                   NREAD16
17EA                      ;
17EA                      ;           NREAD16 assumes that the disk is spinning at the proper speed
17EA                      ;       and the head is located at the proper Side/Track and that the header
17EA                      ;       for the sector desired has already been found.  First it will read in
17EA                      ;       the beginning data marks, then the data, and finally the trailing data
17EA                      ;       bytes.  It will abort if the proper starting or ending sequence of
17EA                      ;       data bytes is not found.
17EA                      ;
17EA                      ;  REGISTERS:
17EA                      ;       IN:
17EA                      ;           ALL  ==  Any value
17EA                      ;
17EA                      ;      OUT:
17EA                      ;           ALL  ==  DESTROYED
17EA                      ;
17EA                      ;--
17EA
17EA  00 00 00 00 00 00 00  .ORG      1800
PAGE -  67  LISA      FILE: NREAD16.TEXT  Sony DRIVER FOR LISA


1800                      .PAGE
1800  1800                READ16    .EQU  *
1800  20 9B13                       jsr   SetRMode        ; setup PAL on Sony to read mode
1803  20 ****                       JSR   RDSYNTOP        ; Read the sync nibbles
1806  90**                         BCC   $03             ; If bitslip marks found then carry = 0
1808  60                            RTS
1806* 01
1809  A0 F4               $03       LDY   #0F4            ; 2     15      First time only 12 bytes
180B
180B  180B                RDBF01    .EQU  *
180B  E6 44                         INC   INXPTRH         ; 5     28      Point to next page
180D  180D                RDBF02    .EQU  *
180D  AE 0E08                       LDX   Q7L             ; 4
1810  10FB                         BPL   RDBF02          ; 2,3
```

```
1812| BD 0011                    LDA   DNIBL,X          ; 4     10      Map from 8 bits ==> 6 bits
1815| 0A                         ASL   A                ; 2     12
1816| 0A                         ASL   A                ; 2     14
1817| AA                         TAX                    ; 2     16      Save temporarily
1818| 0A                         ASL   A                ; 2     18
1819| 0A                         ASL   A                ; 2     20
181A| 85 6E                      STA   TEMP1            ; 3     23      >BBCC0000<
181C| 8A                         TXA                    ; 2     25
181D| 29 C0                      AND   #0C0             ; 2     27      >AA000000<
181F|
181F| AE 0E08              $10   LDX   Q7L              ; 4             Read low bits of byte A
1822| 10FB                      BPL   $10              ; 2,3
1824| 1D 0011                   ORA   DNIBL,X          ; 4     10      Combine low & high bits
1827| 91 43                      STA   @INXPTRL,Y       ; 6     16
1829| A5 6E                      LDA   TEMP1            ; 3     19      >BBCC0000<
182B| C8                         INY                    ; 2     21
182C| D0**                      BNE   $20              ; 2/3   23/24
182E| E6 44                      INC   INXPTRH          ; 5     28      Point to next page
1830|
182C* 02
1830| AE 0E08              $20   LDX   Q7L              ; 4             Read low bits of byte B
1833| 10FB                      BPL   $20              ; 2,3
1835| 29 C0                      AND   #0C0             ; 2      8      >BB000000<
1837| 1D 0011                   ORA   DNIBL,X          ; 4     12      Add low & high bits
183A| 91 43                      STA   @INXPTRL,Y       ; 6     18
183C| C8                         INY                    ; 2     20
183D| F0**                      BEQ   RDCSM01          ; 2/3   22/23
183F| A5 6E                      LDA   TEMP1            ; 3     24      >BBCC0000<
1841| 0A                         ASL   A                ; 2     26
1842| 0A                         ASL   A                ; 2     28
1843|
1843| AE 0E08              $30   LDX   Q7L              ; 4             Read low bits of byte C
1846| 10FB                      BPL   $30              ; 2,3
1848| 29 C0                      AND   #0C0             ; 2      8      >CC000000<
184A| 1D 0011                   ORA   DNIBL,X          ; 4     12      Add low & high bits
184D| 91 43                      STA   @INXPTRL,Y       ; 6     18
184F| C8                         INY                    ; 2     20
1850| F0B9                      BEQ   RDBF01           ; 2/3   22/23   Inc to next page if = 0
1852| D0B9                      BNE   RDBF02           ; 3     25      Branch always taken
1854|
PAGE -  68  LISA      FILE: NREAD16.TEXT  Sony DRIVER FOR LISA


1854|                         .PAGE
183D* 15
1854| 1854              RDCSM01  .EQU  *                ;               Read 4 bytes of checksum
1854| AE 0E08                    LDX   Q7L              ; 4
1857| 10FB                      BPL   RDCSM01          ; 2,3
1859| BD 0011                   LDA   DNIBL,X          ; 4     10      Map from 8 bits ==> 6 bits
185C| 0A                         ASL   A                ; 2     12
185D| 0A                         ASL   A                ; 2     14
185E| A8                         TAY                    ; 2     16      >AABBCC00<
185F| 29 C0                      AND   #0C0             ; 2     18      >AA000000<
1861|
1861| AE 0E08              $10   LDX   Q7L              ; 4             Read low bits of byte A
1864| 10FB                      BPL   $10              ; 2,3
1866| 1D 0011                   ORA   DNIBL,X          ; 4     10      Add low & high bits
1869| 85 61                      STA   CKSUM1           ; 3     13
186B| 98                         TYA                    ; 2     15
186C| 0A                         ASL   A                ; 2     17
186D| 0A                         ASL   A                ; 2     19
186E| A8                         TAY                    ; 2     21      >BBCC0000<
186F| 29 C0                      AND   #0C0             ; 2     23      >BB000000<
1871|
1871| AE 0E08              $20   LDX   Q7L              ; 4             Read low bits of byte B
1874| 10FB                      BPL   $20              ; 2,3
1876| 1D 0011                   ORA   DNIBL,X          ; 4     10      Add low & high bits
1879| 85 62                      STA   CKSUM2           ; 3     13
187B| 98                         TYA                    ; 2     15      >BBCC0000<
187C| 0A                         ASL   A                ; 2     17
187D| 0A                         ASL   A                ; 2     19      >CC000000<
187E|
187E| AE 0E08              $30   LDX   Q7L              ; 4             Read low bits of byte C
1881| 10FB                      BPL   $30              ; 2,3
1883| 1D 0011                   ORA   DNIBL,X          ; 4     10      Combine low & high bits
1886| 85 63                      STA   CKSUM3           ; 3     13
1888|
1888| AD 0E08              RD9   LDA   Q7L              ; 4             Check bitslip mark 1
188B| 10FB                      BPL   RD9              ; 2,3
188D| C5 73                      CMP   DatMk4           ; 3
188F| D0**                      BNE   BSERR            ; 2,3
1891|
1891| AD 0E08              RD10  LDA   Q7L              ; 4             Check bitslip mark 2
1894| 10FB                      BPL   RD10             ; 2,3
1896| C5 74                      CMP   DatMk5           ; 3
1898| D0**                      BNE   BSERR            ; 2,3
189A| 18                         CLC
189B| AD 0D08            RDEXIT  LDA   Q6H              ;               SENSE mode
189E| 60                  RDWASTE RTS
189F|
PAGE -  69  LISA      FILE: NREAD16.TEXT  Sony DRIVER FOR LISA


189F|                         .PAGE
1898* 05
188F* 0E
```

```
189F| E6 49                BSERR    INC   BSCNT          ; Trailing bitslip mark error counter
18A1| 38                   SEC
18A2| B0F7                 BCS   RDEXIT          ; Return to SENSE mode & exit
18A4|
18A4| E6 4A                CSERR    INC   CSERROR        ; Checksum error counter
18A6| 38                   RDERR    SEC
18A7| B0F2                 BCS   RDEXIT
18A9|
18A9| E6 48                RDERR2   INC   STSLP          ; Starting bitslip error count
18AB| 38                   SEC
18AC| B0ED                 BCS   RDEXIT
18AE|
1804* AE18
18AE| 18AE                RDSYNTOP .EQU  *              ; Subroutine to read starting bitslip marks
18AE| AD 0C08              LDA   Q6L             ; Switch from SENSE to READ MODE
18B1| A2 00                LDX   #000            ; For init of index base values
18B3| A0 20                LDY   #NIBLRETR       ; No. of retries allowed for bitslip read
18B5| 88                   RSYNC    DEY
18B6| F0F1                 BEQ   RDERR2
18B8|
18B8| AD 0E08              RS1      LDA   Q7L            ; 4
18BB| 10FB                BPL   RS1             ; 2,3    6
18BD| C5 70                RSYNC1   CMP   DatMk1         ; 3      9       Data mark 1
18BF| D0F4                BNE   RSYNC           ; 2,3    11,12
18C1|
18C1| 86 43                STX   INXPTRL         ; 3      14
18C3| AD 0E08              RS2      LDA   Q7L            ; 4
18C6| 10FB                BPL   RS2             ; 2,3    6
18C8| C5 71                CMP   DatMk2          ; 9      9       Data Mark 2
18CA| D0F1                BNE   RSYNC1          ; 2,3    11,12
18CC|
18CC| 86 44                STX   INXPTRH         ; 3      14
18CE| AD 0E08              RS3      LDA   Q7L            ; 4
18D1| 10FB                BPL   RS3             ; 2,3    6
18D3| C5 72                CMP   DatMk3          ; 3      9       Data Mark 3
18D5| D0E6                BNE   RSYNC1          ; 2,3    11,12
18D7| 18                   CLC                   ; 2      13
18D8|
18D8| AD 0E08              RS4      LDA   Q7L            ; 4             Read SECTOR number ( unchecked )
18DB| 10FB                BPL   RS4             ; 2,3    6
18DD| 60                   RTS                   ;                Back to caller
18DE|
18DE|
18DE|                     .INCLUDE READ
PAGE -  70  LISA      FILE: READ.TEXT  Sony DRIVER FOR LISA


18DE|                     .PAGE
18DE|                     ;**
18DE|                     ;                                READ
18DE|                     ;
18DE|                     ;       READ will read a sector from Drive/Side/Track/Sector, doing offtrack
18DE|                     ;       stepping if necessary to find the data.  It will also verify the
18DE|                     ;       checksum found if commanded to do so.
18DE|                     ;
18DE|                     ;  REGISTERS:
18DE|                     ;       IN:
18DE|                     ;           ALL  == Any value
18DE|                     ;
18DE|                     ;      OUT:
18DE|                     ;           ALL  ==  DESTROYED
18DE|                     ;
18DE|                     ;    CALLS:
18DE|                     ;       SEEK            Seeks to Drive/Side/Track
18DE|                     ;       RDADR           Reads address header field
18DE|                     ;       READ16          Reads data & checksum
18DE|                     ;       VFYCKSUM        Verifies checksum that was read
18DE|                     ;       BADDADDR        Handles all errors during read
18DE|                     ;**
18DE|
1671* DD18
18DE| 18DE                READBF   .EQU  *              ; Read w/o checksum verify entry point
18DE| A9 FF                LDA   #0FF
18E0| D0**                 BNE   READ01
18E2|
1663* E118
18E2| 18E2                READ     .EQU  *
18E2| A9 00                LDA   #00
18E0* 02
18E4| 85 68                READ01   STA   RWCSMFLG       ; Save in global flag
18E6| 20 5E17              JSR   SEEK            ; SEEK TO TRACK, ZERO ERROR CNTS, INIT RETRY
18E9| 20 4912              READ1    JSR   RDADR          ; FIND SECTOR
18EC| B0**                 BCS   READ3           ; ERROR, GO AND CHECK IT
18EE| 20 0018              JSR   READ16          ; READ THE SECTOR
18F1| B0**                 BCS   READ3
18F3| A6 68                LDX   RWCSMFLG       ; Fetch the checksum flag
18F5| D0**                 BNE   READ4           ; If >< 0 then ignore checksum verify
18F7| 20 ****              JSR   VFYCKSUM        ; Verify the checksum
18FA| 90**                 BCC   READ4           ; If carry clear then all is fine
18FC| E6 4A                INC   CSERROR         ; Increment the checksum error counter
18FE|
18F1* 0B
18EC* 10
18FE| 20 ****              READ3    JSR   BADADDR        ; FIND OUT WHAT WENT WRONG ( ecC = 1 )
1901| 90E6                BCC   READ1           ; TRY ONCE MORE
1903| A9 17                LDA   #SERRRD         ; NO GOOD, SIGNAL READ ERROR
```

```
18FA* 09
18FB* 0E
1905| 60                    READ4     RTS
1906|
PAGE -  71  LISA      FILE: READ.TEXT  Sony DRIVER FOR LISA


1906|                       .PAGE
1906|                       ;**
1906|                       ;                                BADADDR
1906|                       ;--
1906|                       ;
1906|                       ; REGISTERS
1906|                       ;  IN
1906|                       ;       All =     ANY VALUE
1906|                       ;  OUT
1906|                       ;       All =     DESTROYED
1906|                       ;       CARRY =   RETRY STATUS (=CLEAR, DON'T RETRY; =SET, PLEASE RETRY)
1906|                       ;
1906|                       ; CALLS
1906|                       ;       RECALBRT  RECALLIBRATE THE DRIVE
1906|                       ;       SEEKALT   SEEK TO TRACK/SIDE
1906|                       ;--
PAGE -  72  LISA      FILE: READ.TEXT  Sony DRIVER FOR LISA


1906|                       .PAGE
1906|
18FF* 0619
1906| 1906                  BADADDR   .EQU  *                 ; ENTRY POINT OF BADADDR  ( ccC = 1 )
1906| C6 26                           DEC   RETRYCNT          ; IF WE'VE TRIED ENOUGH TIMES TO FIND DATA
1908| F0**                           BEQ   BADADDR2
190A| 18                             clc
190B| A5 4B                          LDA   RASTRT            ; OR WE CAN'T FIND START BITSLIP
190D| 65 4E                          adc   RATRK             ; OR WE'RE ON THE WRONG TRK
190F| F0**                           beq   BadGood           ; try again
1911| C5 27                          cmp   RECALCNT          ; if >= recallibration count then abort
1913| F0**                           beq   BadBad            ; ccC = 1 when A = Memory
1915| C5 7B                          cmp   RtyFlg            ; see if new RaStrt or RaTrk error
1917| F0**                           beq   BadGood           ; if equal then no change from last time
1919| 85 7B                          sta   RtyFlg            ; a change -- save new error count
191B| D0**                           bne   BadAddr4          ; recalibrate head location
191D|
1908* 13
191D| C6 27                BadAddr2  dec   RecalCnt
191F| F0**                           beq   BadBad            ; abort upon timeout ( ccC still = 1 )
1921|
191B* 04
1921| 20 ****              BadAddr4  JSR   RECALBRT          ; BUT DONT BOTHER WITH SPD ERRORS AT THIS TIME
1924| 20 6417                        JSR   SEEKALT           ; AND GET BACK TO WERE WE WERE ON TARGET TRK
1927| A5 19                          LDA   MAXRETRY          ; PICK UP THE MAXIMUM NUMBER OF RETRYS COUNT
1929| 85 26                          STA   RETRYCNT          ; AND REFRESH THE COUNTER
1917* 12
190F* 1A
192B| 18                  BadGood   CLC                     ; WE'RE TRYING AGAIN MATE!
191F* 0B
1913* 17
192C| 60                  BadBad    RTS
192D|
192D|
192D|                       .INCLUDE CRECKSUM
PAGE -  73  LISA      FILE: CRECKSUM.TEXT  Sony DRIVER FOR LISA


192D|                       .PAGE
192D|                       ;**
192D|                       ;       CRECKSUM   ==   Create checksum using 524 byte i/o buffer
192D|                       ;
192D|                       ;       ALGORITHM:
192D|                       ;               A sector is composed of 524 user data bytes and a 3 byte
192D|                       ;       checksum.  These are translated into 6 bit nibbles which are used to
192D|                       ;       Look up GCR codwords to be written to the disk.  The data is encoded
192D|                       ;       as follows:  CSUMA, CSUMB, & CSUMC are "registers" used for accumulating
192D|                       ;       the checksum.  BYTEA, BYTEB, & BYTEC contain 3 bytes from the data
192D|                       ;       buffer.
192D|                       ;
192D|                       ;       1.      Rotate CSUMC left
192D|                       ;               CSUMC[65432107] <- CSUMC[76543210]
192D|                       ;               Carry <- CSUMC[7]
192D|                       ;       2.      CSUMA <- CSUMA + BYTEA + Carry from step 1
192D|                       ;       3.      BYTEA <- BYTEA xor CSUMC
192D|                       ;       4       CSUMC <- CSUMC + BYTEB + Carry from step 2
192D|                       ;       5.      BYTEB <- BYTEB xor CSUMA
192D|                       ;       6       CSUMB <- CSUMB + BYTEC + Carry from step 3
192D|                       ;       7.      BYTEC <- BYTEC xor CSUMC
192D|                       ;
192D|                       ;       Propagation of carry among three checksum bytes:
192D|                       ;
192D|                       ;       ----------------------------
192D|                       ;      /\      \/                  \/          NOTE: Carry out of CSUMC is
192D|                       ;       --CSUMC <--CSUMB <--CSUMA <--                from rotate.
192D|                       ;
192D|                       ;  REGISTERS:
192D|                       ;    IN:
192D|                       ;       ALL =   ANY VALUE
192D|                       ;
```

```
192D|                          ;   OUT:
192D|          - - -           ;       ALL =   DESTROYED
192D|                          ;
192D|                          ;--
PAGE -  74  LISA       FILE: CRECKSUM.TEXT  Sony DRIVER FOR LISA


192D|                          .Page
192D|
192D| 192D                     CRECKSUM  .EQU  *                       ; Entry point for all callers
192D| A9 00                               LDA   #00
192F| 85 61                               STA   CKSUM1
1931| 85 62                               STA   CKSUM2                ; Zero only two bytes
1933| 85 43                               STA   INXPTRL               ; Init pointer for 5 cycle index fetch
1935| 85 44                               STA   INXPTRH
1937| E6 44                               INC   INXPTRH               ; Start on page 1
1939| A0 F4                               LDY   #0F4                  ; Last 12 bytes in page 1
193B|
193B| 193B                     CRTOP     .EQU  *                       ; Initially "A" = 0
193B| 0A                                 ASL   A                     ; Move high bit to carry
193C| 08                                 PHP                         ; Save Status bits on stack
193D| 69 00                              ADC   #00                   ; Move carry to low bit ( 8 bit rotate )
193F| 85 63                               STA   CKSUM3
1941| 28                                 PLP                         ; Restore Status bits
1942| B1 43                              LDA   @INXPTRL,Y            ; First of three bytes in loop
1944| AA                                 TAX
1945| 45 63                              EOR   CKSUM3                ; Combine checksum w/ data
1947| 91 43                              STA   @INXPTRL,Y            ; Data ==> buffer
1949| 8A                                 TXA
194A| 65 61                              ADC   CKSUM1                ; Add with above carry
194C| 85 61                               STA   CKSUM1
194E| C8                                 INY
194F| D0**                              BNE   $20                   ; End of Page 2 data
1951| E6 44                              INC   INXPTRH               ; Point to page 3
1953|
194F* 02
1953| B1 43                    $20       LDA   @INXPTRL,Y            ; Second of three bytes
1955| AA                                 TAX
1956| 65 62                              ADC   CKSUM2                ; Add to second checksum byte
1958| 85 62                               STA   CKSUM2
195A| 8A                                 TXA
195B| 45 61                              EOR   CKSUM1                ; Combine checksum w/ data
195D| 91 43                              STA   @INXPTRL,Y            ; Data ==> buffer
195F| C8                                 INY
1960| F0**                              BEQ   $60                   ; End of Page 3 data
1962|
1962| B1 43                              LDA   @INXPTRL,Y            ; Third of three bytes
1964| AA                                 TAX                         ; Save data value
1965| 45 62                              EOR   CKSUM2                ; Combine checksum w/ data
1967| 91 43                              STA   @INXPTRL,Y            ; Data ==> buffer
1969| 8A                                 TXA                         ; Restore data value
196A| 65 63                              ADC   CKSUM3                ; Add to third checksum byte; leave in A
196C| C8                                 INY
196D| D0CC                              BNE   CRTOP                 ; Not at a page boundary, loop
196F| E6 44                              INC   INXPTRH               ; Start w/ Page 2 data
1971| D0C8                              BNE   CRTOP                 ; Branch always taken
1973|
1960* 11
1973| 60                       $60       RTS                         ; END OF CREATING A CHECKSUM
1974|
1974|
PAGE -  75  LISA       FILE: VFYCKSUM.TEXT  Sony DRIVER FOR LISA


1974|                          .INCLUDE VFYCKSUM
PAGE -  76  LISA       FILE: VFYCKSUM.TEXT  Sony DRIVER FOR LISA


1974|                          .PAGE
1974|                          ;**
1974|                          ;               VFYCKSUM  ==  Verify checksum that was just read.
1974|                          ;
1974|                          ;       VFYCKSUM will undo the mess that CRECKSUM created by xor'ing the
1974|                          ;       checksum into the data bytes.  See the CRECKSUM description for the
1974|                          ;       algorithm used to understand what is being undone here.
1974|                          ;
1974|                          ; REGISTERS
1974|                          ;   IN
1974|                          ;       ALL     ==      Any value
1974|                          ;
1974|                          ;  OUT
1974|                          ;       ALL     ==      DESTROYED
1974|                          ;       ccC     ==      0 if checksum matches
1974|                          ;                       1 if checksum does not match
1974|                          ;
1974|                          ;  CALLS
1974|                          ;       NONE
1974|                          ;
1974|                          ;--
1974|
18F8* 7419
1974| 1974                     VFYCKSUM  .EQU  *                       ; Entry point for all callers
1974| A9 00                              LDA   #00                   ; Init temporay checksums to zero
1976| 85 64                              STA   TCKSM1
1978| 85 65                              STA   TCKSM2
197A| 85 43                              STA   INXPTRL
```

```
197C  85 44                       STA   INXPTRH
197E  E6 44                       INC   INXPTRH                ; Start at page 1
1980  A0 F4                       LDY   #0F4                   ; Only last 12 bytes
1982
1982  1982                 VFTOP  .EQU  *
1982  0A                          ASL   A                      ; Rotate the third checksum byte
1983  08                          PHP                          ; Save Status bits
1984  69 00                       ADC   #00                    ; Put carry into low bit (8 bit rotate)
1986  85 66                       STA   TCKSM3                 ; Save 3rd cksum byte
1988  28                          PLP                          ; Restore Status bits
1989  B1 43                       LDA   @INXPTRL,Y             ; Read 1rst byte of 3 byte loop
198B  45 66                       EOR   TCKSM3                 ; Restore original data
198D  91 43                       STA   @INXPTRL,Y
198F  65 64                       ADC   TCKSM1
1991  85 64                       STA   TCKSM1                 ; And update the checksum
1993  C8                          INY
1994  D0**                       BNE   $20
1996  E6 44                       INC   INXPTRH                ; Next page
1998
1994* 02
1998  B1 43                 $20   LDA   @INXPTRL,Y             ; 2nd byte of 3 byte loop
199A  45 64                       EOR   TCKSM1                 ; Restore original data
199C  91 43                       STA   @INXPTRL,Y
199E  65 65                       ADC   TCKSM2
19A0  85 65                       STA   TCKSM2                 ; And update the checksum
19A2  C8                          INY
PAGE -  77  LISA       FILE: VFYCKSUM.TEXT  Sony DRIVER FOR LISA


19A3  F0**                        BEQ   $60                    ; Last byte to sum
19A5
19A5  B1 43                       LDA   @INXPTRL,Y             ; 3rd byte of 3 byte loop
19A7  45 65                       EOR   TCKSM2                 ; Restore original data
19A9  91 43                       STA   @INXPTRL,Y
19AB  65 66                       ADC   TCKSM3                 ; Update the checksum and leave in A
19AD  C8                          INY
19AE  D0D2                        BNE   VFTOP                  ; Not at page boundary, loop
19B0  E6 44                       INC   INXPTRH                ; Next page
19B2  D0CE                        BNE   VFTOP                  ; Branch always taken
19B4
19A3* 0F
19B4  C5 62                 $60   CMP   CKSUM2                 ; 'A' reg has TCKSM2
19B6  D0**                        BNE   $80
19B8  A5 66                       LDA   TCKSM3
19BA  C5 63                       CMP   CKSUM3
19BC  D0**                        BNE   $80
19BE  A5 64                       LDA   TCKSM1
19C0  C5 61                       CMP   CKSUM1
19C2  D0**                        BNE   $80
19C4  18                          CLC
19C5  60                          RTS
19C6
19C2* 02
19BC* 08
19B6* 0E
19C6  38                    $80   SEC
19C7  60                          RTS
19C8
19C8
19C8                       .INCLUDE WRITE
PAGE -  78  LISA       FILE: WRITE.TEXT  Sony DRIVER FOR LISA


19C8                       .PAGE
19C8                       ;**
19C8                       ;
19C8                       ;                                WRITE
19C8                       ;
19C8                       ;       WRITE will write to a Drive/Side/Track/Sector after the sector
19C8                       ;       header has been found by "RDADR".  Depending upon the command, it
19C8                       ;       will perform a checksum operation on the data.  If it is unable to
19C8                       ;       read the address header mark after 100. tries it will recalibrate
19C8                       ;       and try once more.
19C8                       ;
19C8                       ;  REGISTER
19C8                       ;    IN
19C8                       ;       ALL     ==      Any value
19C8                       ;
19C8                       ;   OUT
19C8                       ;       ALL     ==      DESTROYED
19C8                       ;
19C8                       ;--
19C8
19C8
1665* C719
19C8  19C8                 WRITE  .EQU  *
19C8  20 2D19                     JSR   CRECKSUM               ; Create a checksum
1673* CA19
19CB  19CB                 WRITEBF .EQU *                      ; Write data using host supplied checksum
19CB  20 7715                     JSR   PRENIB                 ; Setup data for reading across page boundaries
19CE  20 5E17                     JSR   SEEK
19D1  20 4912              WRITE3 JSR   RDADR                  ; FIND SECTOR
19D4  90**                        BCC   WRITE16                ; WRITE IF OK & return to caller from WRITE16
19D6  20 0619                     JSR   BADADDR
19D9  90F6                        BCC   WRITE3
19DB  A9 18                       LDA   #SERRWR                ; WRITE ERROR
```

```
19DD: 60                              RTS
19DE:
19DE:
19DE:                         .INCLUDE Write16
PAGE -  79  LISA      FILE: WRITE16.TEXT  Sony DRIVER FOR LISA


19DE:                         .page
19DE:                         ;**
19DE:                         ;         Write16
19DE:                         ;
19DE:                         ; $10       ldx  Q6L            ; 1       1         Assume worst case
19DE:                         ;           bpl  $10            ; 3       4         branch taken
19DE:                         ;           ldx  Q6L            ; 4       8         bit will be set
19DE:                         ;           bpl  $10            ; 2       10        fall thru
19DE:                         ;           sta  Q6H            ; 4       14        put the data into reg
19DE:                         ; 31 cycles max after bit is set before UnderRun bit is set.  Using above code
19DE:                         ; as start of time critical code,
19DE:                         ; total time = ((32-14)*31) = 49 cycles
19DE:                         ;--
19DE:
PAGE -  80  LISA      FILE: WRITE16.TEXT  Sony DRIVER FOR LISA


19DE:                         .page
19DE: 00 00 00 00 00 00 00    .org      1A00              ; precise page alignment
19D4* 2A
1A00: 1A00                    Write16   .equ *
1A00: A6 34                             ldx  IIOB+Sector
1A02: BD 0010                           lda  Nib1,x
1A05: 85 6D                             sta  Sv4
1A07: A0 00                             ldy  #0           ; no speed check bytes
1A09: 84 44                             sty  InxPtrH
1A0B: 98                                tya               ; flag to tell sync20 to use data mark values
1A0C: C8                                iny               ; write only one set of self-sync bytes
1A0D: 20 3A15                           jsr  Sync20       ; 6       6     turn on write circuitry & write self-sync bytes
1A10: A0 F4                             ldy  #0F4         ; 2       8
1A12: A9 AD                             lda  #0AD         ; 2       10    last byte of data header field
1A14: D0**                              bne  Pg1x         ; 3       13
1A16:
1A16: 1A16                    Pg1       .equ *            ; Code to write out last 12 bytes of Page #1
1A16: 86 6D                             stx  Sv4          ; 3
1A14* 02
1A18: AE 0C08                 Pg1x      ldx  Q6L          ; 4       17
1A1B: 10FB                              bpl  Pg1x         ; 2,3     19/20
1A1D: 8D 0D08                           sta  Q6H          ; 4       14        worst case possible
1A20:
1A20: BE 0201                           ldx  Page01+2,y   ; 4       18
1A23: BD 0010                           lda  Nib1,x       ; 4       22
1A26: 85 6C                             sta  Sv3          ; 3       25
1A28: 8A                                txa               ; 2       27
1A29: 4A                                lsr  a            ; 2       29
1A2A: 4A                                lsr  a            ; 2       31
1A2B: 85 6E                             sta  Temp1        ; 3       34
1A2D: BE 0101                           ldx  Page01+1,y   ; 4       38
1A30: BD 0010                           lda  Nib1,x       ; 4       42
1A33: 85 6B                             sta  Sv2          ; 3       45
1A35: 8A                                txa               ; 2       47
1A36: 29 C0                             and  #0C0         ; 2       49
1A38: 05 6E                             ora  Temp1        ; 3       52
1A3A: 4A                                lsr  a            ; 2       54
1A3B: A6 6D                             ldx  Sv4          ; 3       57        61-14 = 47
1A3D: 8E 0D08                           stx  Q6H          ; 4       61        Write last byte of previous loop
1A40:
1A40: 4A                                lsr  a            ; 2       2
1A41: 85 6E                             sta  Temp1        ; 3       5
1A43: B9 0001                           lda  Page01,y     ; 4       9
1A46: 29 C0                             and  #0C0         ; 2       11
1A48: 05 6E                             ora  Temp1        ; 3       14
1A4A: 4A                                lsr  a            ; 2       16
1A4B: 4A                                lsr  a            ; 2       18
1A4C: AA                                tax               ; 2       20
1A4D: BD 0010                           lda  Nib1,x       ; 4       24
1A50: 8D 0D08                           sta  Q6H          ; 4       28
1A53:
1A53: BE 0001                           ldx  Page01,y     ; 4       4
1A56: C8                                iny               ; 2       6
1A57: C8                                iny               ; 2       8
PAGE -  81  LISA      FILE: WRITE16.TEXT  Sony DRIVER FOR LISA


1A58: BD 0010                           lda  Nib1,x       ; 4       12
1A5B: AE 0C08                 $15       ldx  Q6L          ; 4       16
1A5E: 10FB                              bpl  $15          ; 2,3     18
1A60: 8D 0D08                           sta  Q6H          ; 4       22
1A63:
1A63: A5 6B                             lda  Sv2          ; 3       3         Second byte of loop
1A65: A6 6C                             ldx  Sv3          ; 3       6         Third byte
1A67: C8                                iny               ; 2       8
1A68: D0AC                              bne  Pg1          ; 2,3     10        fetch rest of 12 bytes
1A6A:                                                     ;                   fall thru & start on Page #2
1A6A:                         ;**
1A6A:                         ;
1A6A:                         ;--
1A6A:
1A6A: 1A6A                    Pg2       .equ *            ; Code to write out first 255 bytes of Page #2
```

```
1A6A| 86 6D                        stx  Sv4          ; 3
1A6C| AE 0C08              $40     ldx  Q6L          ; 4
1A6F| 10FB                        bpl  $40          ; 2,3
1A71| 8D 0D08                     sta  Q6H          ; 4       14        worst case possible
1A74|
1A74| BE 0202                     ldx  Page02+2,y   ; 4       18
1A77| BD 0010                     lda  Nibl,x       ; 4       22
1A7A| 85 6C                        sta  Sv3          ; 3       25
1A7C| 8A                          txa               ; 2       27
1A7D| 4A                          lsr  a            ; 2       29
1A7E| 4A                          lsr  a            ; 2       31
1A7F| 85 6E                       sta  Temp1        ; 3       34
1A81| BE 0102                     ldx  Page02+1,y   ; 4       38
1A84| BD 0010                     lda  Nibl,x       ; 4       42
1A87| 85 6B                       sta  Sv2          ; 3       45
1A89| 8A                          txa               ; 2       47
1A8A| 29 C0                       and  #0C0         ; 2       49
1A8C| 05 6E                       ora  Temp1        ; 3       52
1A8E| 4A                          lsr  a            ; 2       54
1A8F| A6 6D                       ldx  Sv4          ; 3       57        61-14 = 47
1A91| 8E 0D08                     stx  Q6H          ; 4       61        Write last byte of previous loop
1A94|
1A94| 4A                          lsr  a            ; 2       2
1A95| 85 6E                       sta  Temp1        ; 3       5
1A97| B9 0002                     lda  Page02,y     ; 4       9
1A9A| 29 C0                       and  #0C0         ; 2       11
1A9C| 05 6E                       ora  Temp1        ; 3       14
1A9E| 4A                          lsr  a            ; 2       16
1A9F| 4A                          lsr  a            ; 2       18
1AA0| AA                          tax               ; 2       20
1AA1| BD 0010                     lda  Nibl,x       ; 4       24
1AA4| 8D 0D08                     sta  Q6H          ; 4       28        Composite of 3 bytes
1AA7|
1AA7| BE 0002                     ldx  Page02,y     ; 4       4
1AAA| C8                          iny               ; 2       6
1AAB| C8                          iny               ; 2       8
1AAC| BD 0010                     lda  Nibl,x       ; 4       12
1AAF| AE 0C08              $55     ldx  Q6L          ; 4       16
1AB2| 10FB                        bpl  $55          ; 2,3     18
1AB4| 8D 0D08                     sta  Q6H          ; 4       22
PAGE -  82  LISA      FILE: WRITE16.TEXT  Sony DRIVER FOR LISA


1AB7|
1AB7| A5 6B                       lda  Sv2          ; 3       3         Second byte of loop
1AB9| A6 6C                       ldx  Sv3          ; 3       6         Third byte
1ABB| C8                          iny               ; 2       8
1ABC| C0 FF                       cpy  #Pg2Len      ; 2       10
1ABE| D0AA                        bne  Pg2          ; 2,3     12        fetch rest of 255 bytes
1AC0|
1AC0| 20 ****                    jsr  WrAX         ; 6       26/6      write Nibls in A and X
1AC3| AE 0103                     ldx  Page03+1     ; 4       10
1AC6| BD 0010                     lda  Nibl,x       ; 4       14
1AC9| 85 6D                       sta  Sv4          ; 3       17
1ACB| A5 5E                       lda  CpBy01       ; 3       20
1ACD| 20 ****                    jsr  WrNibl       ; 6       26/6
1AD0|
1AD0| A0 02                       ldy  #02          ; 2       8       for use in Pg3
1AD2| AE FF02                     ldx  Page02+0FF   ; 4       12
1AD5| 20 ****                    jsr  WrByteX      ; 6       18/6
1AD8|
1AD8| AE 0003                     ldx  Page03       ; 4       10
1ADB| BD 0010                     lda  Nibl,x       ; 4       14
1ADE| D0**                        bne  Pg3x         ; 3       17
1AE0|
1AC1* E01A
1AE0| 1AE0              WrAX      .equ *            ;  write nibls that are in A & X(Sv3)
1AE0| 20 ****                    jsr  WrNibl       ; 6       32/6
1AE3| A5 6C                       lda  Sv3          ; 3       9
1AE5| 4C ****                    jmp  WrNibl       ; 3       12/6
1AE8|
1AE8|                           ;**
1AE8|                           ;
1AE8|                           ;--
1AE8|
1AE8| 1AE8              WrByte    .equ *            ; nibblize byte in Areg before starting
1AE8| AA                          tax               ; 2
1AD6* E91A
1AE9| 1AE9              WrByteX   .equ *            ; use byte in Xreg
1AE9| BD 0010                     lda  Nibl,x       ; 4
1AEC|
1AE6* EC1A
1AE1* EC1A
1ACE* EC1A
1AEC| 1AEC              WrNibl    .equ *            ; wait for handshake bit, then write data in Areg
1AEC| AE 0C08                     ldx  Q6L          ; 4
1AEF| 10FB                        bpl  WrNibl       ; 2,3
1AF1| 8D 0D08                     sta  Q6H          ; 4
1AF4| 60                          rts               ; 6
1AF5|
1AF5| 00 00 00 00 00 00 00  .align    0100
1B00| 1B00              Pg3       .equ *            ; Code to write out first 254 bytes of Page #3
1B00| 86 6D                       stx  Sv4          ; 3
1ADE* 22
1B02| AE 0C08              Pg3x      ldx  Q6L          ; 4
1B05| 10FB                        bpl  Pg3x         ; 2,3
```

```
1B07| 8D 0D08                        sta  Q6H             ; 4       14        worst case possible
1B0A|
PAGE -  83  LISA      FILE: WRITE16.TEXT  Sony DRIVER FOR LISA


1B0A| BE 0203                        ldx  Page03+2,y      ; 4       18
1B0D| BD 0010                        lda  Nibl,x          ; 4       22
1B10| 85 6C                         sta  Sv3             ; 3       25
1B12| 8A                           txa                  ; 2       27
1B13| 4A                           lsr  a               ; 2       29
1B14| 4A                           lsr  a               ; 2       31
1B15| 85 6E                         sta  Temp1           ; 3       34
1B17| BE 0103                        ldx  Page03+1,y      ; 4       38
1B1A| BD 0010                        lda  Nibl,x          ; 4       42
1B1D| 85 6B                         sta  Sv2             ; 3       45
1B1F| 8A                           txa                  ; 2       47
1B20| 29 C0                         and  #0C0            ; 2       49
1B22| 05 6E                         ora  Temp1           ; 3       52
1B24| 4A                           lsr  a               ; 2       54
1B25| A6 6D                         ldx  Sv4             ; 3       57        61-14 = 47
1B27| 8E 0D08                        stx  Q6H             ; 4       61        Write last byte of previous loop
1B2A|
1B2A| 4A                           lsr  a               ; 2       2
1B2B| 85 6E                         sta  Temp1           ; 3       5
1B2D| B9 0003                        lda  Page03,y        ; 4       9
1B30| 29 C0                         and  #0C0            ; 2       11
1B32| 05 6E                         ora  Temp1           ; 3       14
1B34| 4A                           lsr  a               ; 2       16
1B35| 4A                           lsr  a               ; 2       18
1B36| AA                           tax                  ; 2       20
1B37| BD 0010                        lda  Nibl,x          ; 4       24
1B3A| 8D 0D08                        sta  Q6H             ; 4       28        Composite of 3 bytes
1B3D|
1B3D| BE 0003                        ldx  Page03,y        ; 4       4
1B40| C8                           iny                  ; 2       6
1B41| C8                           iny                  ; 2       8
1B42| BD 0010                        lda  Nibl,x          ; 4       12
1B45| AE 0C08              $85       ldx  Q6L             ; 4       16
1B48| 10FB                        bpl  $85             ; 2,3     18
1B4A| 8D 0D08                        sta  Q6H             ; 4       22
1B4D|
1B4D| A5 6B                         lda  Sv2             ; 3       3         Second byte of loop
1B4F| A6 6C                         ldx  Sv3             ; 3       6         Third byte
1B51| C8                           iny                  ; 2       8
1B52| C0 FE                         cpy  #Pg3Len         ; 2       10
1B54| D0AA                        bne  Pg3             ; 2,3     12        fetch rest of 254 bytes
1B56|
1B56| 20 E01A                        jsr  WrAX            ; 6       26/6      write Nibls in A and X(Sv3)
1B59| AE FF03                        ldx  Page03+0FF      ; 4       10
1B5C| BD 0010                        lda  Nibl,x          ; 4       14
1B5F| 85 6C                         sta  Sv3             ; 3       17
1B61| A5 5F                         lda  CpBy02          ; 3       20
1B63| 20 EC1A                        jsr  WrNibl          ; 6       26/6
1B66| AE FE03                        ldx  Page03+0FE      ; 4       10
1B69| BD 0010                        lda  Nibl,x          ; 4       14
1B6C| 20 E01A                        jsr  WrAX            ; 6       20/6
1B6F|
1B6F| A0 61                         ldy  #CkSum1         ; 2       8         absolute addr of 3 cksum bytes
1B71| 84 43                         sty  InxPtrL         ; 3       11
1B73| A5 60                         lda  CpCkSum         ; 3       14
PAGE -  84  LISA      FILE: WRITE16.TEXT  Sony DRIVER FOR LISA


1B75| AE 0C08              $95       ldx  Q6L             ; 4       18
1B78| 10FB                        bpl  $95             ; 2,3     20
1B7A| 8D 0D08                        sta  Q6H             ; 4       24
1B7D| A0 00                         ldy  #0              ; 2       2         3 bytes to transfer
1B7F| B1 43              $99       lda  @InxPtrL,y      ; 6       8
1B81| 20 E81A                        jsr  WrByte          ; 3       11/6
1B84| C8                           iny                  ; 2       8
1B85| C0 03                         cpy  #3              ; 2       10
1B87| D0F6                        bne  $99             ; 2,3     12
1B89|                                                   ;  fall thru & write final BitSlip marks & end
1B89| A5 73                         lda  DatMk4          ; 3
1B8B| 20 EC1A                        jsr  WrNibl          ; 6
1B8E| A5 74                         lda  DatMk5          ; 3
1B90| 20 EC1A                        jsr  WrNibl          ; 6
1B93|
1B93| 1B93                ShtOff    .equ *               ; write 2 Bitslip and return /WrUnderRun
1B93| A9 FF                         lda  #0FF            ; 2
1B95| 20 EC1A                        jsr  WrNibl          ; 6
1B98| 20 EC1A                        jsr  WrNibl          ; 6
1B9B| 18                           clc                  ; 2
1B9C| AD 0C08                        lda  Q6L             ; 4       Bit6 = underrun bit
1B9F| 29 40                         and  #40             ; 2       leave only Bit6
1BA1| D0**                        bne  $38             ; 2,3     If = 1 then no underrun occurred
1BA3| A9 1F                         lda  #ErrWrt         ; obscure error code #
1BA5| 38                           sec                  ; 2
1BA1* 03
1BA6| 8D 0D08              $38       sta  Q6H             ; 4       Put into Write Load state
1BA9| AE 0E08                        ldx  Q7L             ; 4       Now into Write Protect - Sense state
1BAC| 60                           rts                  ; 6
1BAD|
1BAD|
1BAD|                     .INCLUDE FORMAT
PAGE -  85  LISA      FILE: FORMAT.TEXT  Sony DRIVER FOR LISA
```

```
1BAD|                          .PAGE
1BAD|                          ;**
1BAD|                          ;                                FORMAT
1BAD|                          ;
1BAD|                          ;--
1BAD|                          ; REGISTERS
1BAD|                          ;  IN
1BAD|                          ;       All =       ANY VALUE
1BAD|                          ;  OUT
1BAD|                          ;       All =       DESTROYED
1BAD|                          ;
1BAD|                          ; CALLS
1BAD|                          ;       CLRBUF          Clears 524 byte buffer and other variables
1BAD|                          ;       SELSIDE         Selects side
1BAD|                          ;       RECALL          Recalibrates disk to optical index signal
1BAD|                          ;       WRITRK          Writes no. of sectors in TEMPSEC
1BAD|                          ;       VERTRK          Verifies no. of sectors in TEMPSEC
1BAD|                          ;       SEEKALT         Seeks to track in IIOB+TRACK and sets speed
1BAD|                          ;       WRSYNTRK        Writes bitslip FF's and A9's then sector 0
1BAD|                          ;       WADR16          Writes sector #, both header and data fields
1BAD|                          ;       RDADR           Read sector header
1BAD|                          ;       READ16          Reads data fields
1BAD|                          ;       VFYCKSUM        Verifies checksum that was read
1BAD|                          ;       PRENIB          Prenibblizes certain bytes for timing purposes
1BAD|                          ;
1BAD|                          ;**
PAGE -  86  LISA      FILE: FORMAT.TEXT  Sony DRIVER FOR LISA


1BAD|                          .PAGE
1BAD|
1BAD| 1BAD                    FORMAT    .EQU  *               ; FORMAT entry point
1BAD| 1BAD                    FORMTRAK  .EQU  *
1BAD| 20 ****                           JSR   CLRBUF          ; Clear the buffer to be written
1BB0| A9 07                             lda   #7
1BB2| 85 2B                             sta   FmtCap
1BB4| 20 ****                           jsr   RcCalMtr        ; Turn on motor & start from a known point (track 0)
1BB7| B0**                             bcs   FormRts         ; abort upon error from Recal
1BB9|
1BB9| 20 6417                 FORMTOP   JSR   SEEKALT         ; Go to the track in IIOB+TRACK & set speed
1BBC| A4 23                             ldy   CurClass
1BBE| B9 4611                           LDA   SECPRTRK,Y      ; Fetch number of sectors in current track
1BC1| 85 67                            STA   TEMPSEC         ; SAVE IT FOR WRITRK ROUTINE
1BC3|
1BC3| 20 ****                           JSR   WRITRK          ; Write a track full of sectors
1BC6|
1BC6| A5 31                            LDA   IIOB+COMMAND    ; If a format track
1BC8| C9 05                            CMP   #FRMTTRK        ; then we're done-exit the routine
1BCA| F0**                             beq   VT00            ; Command = FormatTrack (5)
1BCC|
1BCC| A5 0A                            lda   Iob+NoSides     ; 1 = single side, 2 = double sided
1BCE| C9 02                            cmp   #2
1BD0| D0**                             bne   $39             ; if <> 2 then go to incr to next track
1BD2| A9 20                            lda   #20
1BD4| 85 33                            sta   Iiob+Side
1BD6| 20 ****                          jsr   WriTrk          ; now write the second side
1BD9| A9 00                            lda   #00
1BDB| 85 33                            sta   Iiob+Side       ; restore to side #0
1BDD|
1BDD* 0B
1BDD| E6 35                 $39       INC   IIOB+TRACK
1BDF| A5 35                           LDA   IIOB+TRACK
1BE1| C9 50                           CMP   #MAXTRACK+1
1BE3| 90D4                           BCC   FORMTOP         ; UNTIL WE RUN OUT OF TRACKS
1BE5| C6 35                           dec   Iiob+Track
1BE7| D0**                           bne   VT00            ; skip turning the motor on
1BE9|
1BE9| 1BE9                 VERIFY    .EQU  *               ; VERIFY entry point
1BE9| A9 4F                           lda   #MaxTrack
1BEB| 85 35                           sta   Iiob+Track      ; from inside of disk to outer edge
1BED| 1BED                 VERTRACK  .EQU  *
1BED| 20 ****                          jsr   RcCalMtr        ; Turn on motor & start from a known point (track 0)
1BF0| B0**                           bcs   FormRts         ; abort upon error from Recal
1BE7* 09
1BCA* 26
1BF2| 1BF2                 VT00      .equ  *
1BF2| 20 6417                        JSR   SEEKALT         ; Go to the track in IIOB+TRACK & set speed
1BF5| A4 23                          ldy   CurClass
1BF7| B9 4611                        LDA   SECPRTRK,Y      ; Fetch number of sectors in current track
1BFA| 85 67                          STA   TEMPSEC         ; SAVE IT FOR WRITRK ROUTINE
1BFC|
1BFC| 20 ****                        JSR   VERTRK
1BFF| B0**                           BCS   FORMERR         ; Error, exit from routine
PAGE -  87  LISA      FILE: FORMAT.TEXT  Sony DRIVER FOR LISA


1C01|
1C01| A5 31                          LDA   IIOB+COMMAND    ; If a verify track
1C03| C9 05                          CMP   #FrmtTrk        ; then we're done-exit the routine
1C05| B0**                           bcs   $41             ; Command = FormatTrack (5) or VerifyTrack (6)
1C07|
1C07| A5 0A                          lda   Iob+NoSides     ; 1 = single side, 2 = double sided
1C09| C9 02                          cmp   #2
1C0B| D0**                           bne   $39
```

```
1C0D  A9 20                           lda   #20
1C0F  85 33 -- --                     sta   IIeb+Side
1C11  20 **** **                      jsr   VerTrk          ; now verify side 1, track x
1C14  B0**                            bcs   FormErr         ; Error, exit from routine
1C16  A9 00                           lda   #00
1C18  85 33                           sta   IIeb+Side
1C1A
1C0B* 0D
1C1A  C6 35                   $39     dec   IIOB+TRACK
1C1C  10D4                            bpl   VT00
1C1E
1C05* 17
1C1E  18                      $41     CLC
1C1F  60                              RTS
1C20
1C14* 0A
1BFF* 1F
1C20  A5 35                   FORMERR LDA   IIOB+TRACK      ; Current track number
1C22  8D D101                         STA   TRKNUMB         ; Save for host's usage
1C25  A5 33                           LDA   IIOB+SIDE       ; Current side
1C27  29 20                           and   #20             ; leave bit 5 -- side bit
1C29  F0**                            beq   $17
1C2B  A9 01                           lda   #1
1C29* 02
1C2D  8D D201                 $17     STA   SIDNUMB         ; Save for host's usage
1C30  A9 15                           LDA   #SERRFRMT       ; Format error code
1BF0* 40
1BB7* 79
1C32  60                      FORMRTS RTS
1C33
1C33  1C33                    TooSml  .equ  *
1C33  00 05 0A 0F                     .byte 0., 5., 10., 15.        ; not used
1C37  14 19 1E 23                     .byte 20., 25., 30., 35.      ; 4:7
1C3B  28 2D 32 37                     .byte 40., 45., 50., 55.      ; 8:11
1C3F  3C 41 46 4B                     .byte 60., 65., 70., 75.      ; 12:15
1C43
1C43  1C43                    JustRit .equ  *
1C43  6C 63 5A 51 48                  .byte 108., 99., 90., 81., 72. ; sectors in class * 9 bytes per sector
1C48
1C48                          ;**
1C48                          ;       WRITRK will physically format a disk in a 2:1 interleave.  It will
1C48                          ;       write Sector 0 , Sector X, Sector 1, Sector Y, etc.
1C48                          ;--
1C48
1BD7* 481C
1BC4* 481C
1C48  1C48                    WRITRK  .EQU  *               ; Entry for writing a track of 524 byte sectors
PAGE -  88  LISA      FILE: FORMAT.TEXT  Sony DRIVER FOR LISA


1C48  20 E612                         jsr   SelSide         ; Select proper side
1C4B  A5 67                           LDA   TEMPSEC         ; Total number of sectors on current track
1C4D  85 78                           STA   TOTCNT          ; Init total count of sectors written
1C4F  4A                              LSR   A               ; Divide by 2 and put remainder into the carry
1C50  69 00                           ADC   #0              ; Round up by adding carry
1C52  85 76                           STA   HIHCNT          ; Init counter for high sector values
1C54  A9 00                           LDA   #00
1C56  85 77                           STA   CNTPTR          ; Init pointer for which count to use
1C58  85 75                           STA   LOWCNT          ; For counting up from Sector 0
1C5A  85 34                           STA   IIOB+SECTOR     ; Start w/ sector 0
1C5C
1C5C  20 0012                         JSR   WRSYNTRK        ; Write 20 usec nibbles and A9's before sector 0
1C5F  C6 78                           DEC   TOTCNT          ; Subtract 1 from total sector count
1C61  E6 75                           INC   LOWCNT          ; Increment value to next low sector ( 1 )
1C63  A6 77                   $23     LDX   CNTPTR          ; Pointer to which cnt to use -- low or high
1C65  F0**                            BEQ   $35             ; If = 0 then increment to 1
1C67  A2 FF                           LDX   #0FF            ; If = 1 then decrement to 0
1C65* 02
1C69  E8                      $35     INX
1C6A  86 77                           STX   CNTPTR
1C6C  B5 75                           LDA   LOWCNT,X        ; Fetch sector number
1C6E  85 34                           STA   IIOB+SECTOR
1C70  F6 75                           INC   LOWCNT,X        ; Increment to next sector number
1C72  20 0412                         JSR   WADR16          ; Write address and data fields
1C75  C6 78                           DEC   TOTCNT          ; Decrement total sector cnt
1C77  D0EA                            BNE   $23             ; When = 0 then all sectors are written
1C79
1C79  A9 00                           lda   #00
1C7B  85 34                           sta   IIeb+Sector
1C7D  20 4912                         jsr   RdAdr           ; read sctr 0 and count bytes
1C80
1C80  B0**                            bcs   DecrG1
1C82  A5 5D                           lda   RangeH
1C84  C9 08                           cmp   #RdAdrTmt       ; if = then less than 256 bytes were counted
1C86  D0**                            bne   IncrG1          ; > 256 bytes so increase inter-sector gap
1C88
1C88  A4 2B                           ldy   FmtGap
1C8A  B9 331C                         lda   TooSml,y
1C8D  C5 5C                           cmp   RangeL
1C8F  B0**                            bcs   DecrG1          ; a > normal cnt with current Gap amount
1C91
1C91  A4 23                           ldy   CurClass
1C93  79 431C                         adc   JustRit,y
1C96  C5 5C                           cmp   RangeL
1C98  B0**                            bcs   WrTkDone        ; A > RangeL so all ok
1C9A
```

```
1C86* 12
1C9A  A9 0E                   IncrG1    lda   #14.
1C9C  C5 2B                             cmp   FmtGap          ; limit to 14. selfsync groups of five
1C9E  F0**                              beq   WrTkDone
1CA0  E6 2B                             inc   FmtGap          ; incr count for next track
1CA2
1C9E* 02
1C98* 08
1CA2  1CA2                    WrTkDone  .equ  *
PAGE -  89  LISA       FILE: FORMAT.TEXT  Sony DRIVER FOR LISA


1CA2  60                                rts
1CA3
1C8F* 12
1C80* 21
1CA3  1CA3                    DecrG1    .equ  *
1CA3  A9 04                             lda   #4.
1CA5  C5 2B                             cmp   FmtGap
1CA7  F0F9                              beq   WrTkDone        ; minimum gap is 4 counts of 5 20usec bytes
1CA9  C6 2B                             dec   FmtGap
1CAB  D09B                              bne   WriTrk
1CAD
1CAD                          ;**
1CAD                          ;
1CAD                          ;--
1C12* AD1C
1BFD* AD1C
1CAD  1CAD                    VERTRK    .equ  *
1CAD  20 E612                           jsr   SelSide         ; Select proper side
1CB0  A9 00                             LDA   #00
1CB2  85 34                             STA   IIOB+SECTOR
1CB4  8D D001                           STA   SCTRCNT
1CB7  A5 19                   VFYTRK1   LDA   MAXRETRY
1CB9  85 26                             STA   RETRYCNT
1CBB
1CBB  20 4912                 VFYTRK2   JSR   RDADR           ; Read sector address field
1CBE  B0**                              BCS   VFYERR
1CC0  20 0018                           JSR   READ16          ; Read sector data
1CC3  B0**                              BCS   VFYERR
1CC5  20 7419                           JSR   VFYCKSUM        ; Verify the checksum
1CC8  B0**                              BCS   VFYERR          ; Must be 0
1CCA  E6 34                   VFRTRY    INC   IIOB+SECTOR
1CCC  A5 34                             LDA   IIOB+SECTOR
1CCE  C5 67                             CMP   TEMPSEC
1CD0  D0E5                              BNE   VFYTRK1         ; If equal then carry will be set
1CD2  AD D001                           LDA   SCTRCNT
1CD5  D0**                              BNE   $74
1CD7  18                                CLC
1CD5* 01
1CD8  60                      $74       RTS
1CD9
1CC8* 0F
1CC3* 14
1CBE* 19
1CD9  C6 26                   VFYERR    DEC   RETRYCNT
1CDB  D0DE                              BNE   VFYTRK2
1CDD  AC D001                           LDY   SCTRCNT
1CE0  A5 34                             LDA   IIOB+SECTOR     ; Current sector number
1CE2  99 D301                           STA   SCTRSAV,Y       ; Save for host usage
1CE5  EE D001                           INC   SCTRCNT
1CE8  D0E0                              BNE   VFRTRY          ; Go to next sector and try again
1CEA
1BAE* EA1C
1CEA  1CEA                    CLRBUF    .equ  *
1CEA  A9 00                             LDA   #00             ; Clear the total buffer area
1CEC  A8                                TAY
PAGE -  90  LISA       FILE: FORMAT.TEXT  Sony DRIVER FOR LISA


1CED  99 0002                 $05       STA   PAGE02,Y        ; Zero part of the data buffer
1CF0  99 0003                           STA   PAGE03,Y        ; Zero the rest of the data buffer
1CF3  C8                                INY
1CF4  D0F7                              BNE   $05
1CF6  A0 0B                             LDY   #BUFR12SZ       ; For 12 byte header buffer
1CF8  99 F401                 $10       STA   BUFR12,Y
1CFB  88                                DEY
1CFC  10FA                              BPL   $10
1CFE  85 61                             STA   CKSUM1          ; Clear the three checksum bytes
1D00  85 62                             STA   CKSUM2
1D02  85 63                             STA   CKSUM3
1D04  4C 7715                           JMP   PRENIB          ; Create 5 composite bytes & return to caller
1D07
1D07
1D07                          .INCLUDE Recal
PAGE -  91  LISA       FILE: RECAL.TEXT  Sony DRIVER FOR LISA


1D07                          .PAGE
1D07  1D07                    CleScDly  .equ  *
1D07  C9 0D                             cmp   #13.
1D09  B0**                              bcs   $43             ; if 'a' >= 13 then only wait for head settling
1D0B  A8                                tay                   ; save number of steps
1D0C  A9 00                             lda   #0              ; ecC already = '0'
1D0E  69 02                   $31       adc   #2              ; use 10 msec/step instead of 12 msec
1D10  88                                dey
```

```
1D11| D0FB                          bne   $31
1D13| 85 6E                        sta   Temp1
1D15| A5 15                        lda   ScDly
1D17| 38                          sec
1D18| E5 6E                        sbc   Temp1
1D1A| B0**                        bcs   $65             ; if ccC=1 then result is positive
1D09* 11
1D1C| A5 16                  $43   lda   HeaDelay      ; assume only wait for head settling
1D1E| 60                          rts
1D1A* 03
1D1F| C5 16                  $65   cmp   HeaDelay      ; see if head settling time also included
1D21| 90F9                        bcc   $43
1D23| 60                          rts
1D24|
1D24|                       ;**
1D24|                       ;                         RECALBRT
1D24|                       ;--
1BEE* 241D
1BB5* 241D
1D24| 1D24                  RcCalMtr .EQU *
1D24| A5 21                        lda   MtrOn
1D26| D0**                        bne   RcCal1        ; if motor is not on then turn it on & wait 400ms
1D28| 20 4113                      jsr   TMOn
1D2B|
1D2B| 1D2B                  RECALBRT .EQU *           ; ENTRY POINT FOR RECALBRT
1D26* 03
1D2B| 1D2B                  RECALL   .EQU *
1D2B| A0 00                        ldy   #0
1D2D| 84 22                        sty   CurTrack
1D2F| 20 9213                      jsr   SetSpdy       ; set CurClass to '0' also
1D32| 88                          dey
1D33| 84 0C                        sty   HostSeek      ; tell 68K that I am seeking
1D35| A2 00                        ldx   #InWard
1D37| 20 5013                      jsr   SetDretX
1D3A| A2 04                        ldx   #RclStep
1D3C| 86 79                        stx   Temp3         ; 3 steps away from track 00 for bad power supply
1D3E| 20 6313                $06   jsr   DoStep        ; step away from track 0
1D41| C6 79                        dec   Temp3
1D43| D0F9                        bne   $06           ; loop 2 times for 3 total steps
1D45| A9 08                        lda   #TurnRnd      ; must wait for stepper to come to it's senses
1D47| 20 ****                     jsr   Wait
1D4A|
1D4A| A2 64                        ldx   #TmOutRcl
1D4C| 86 79                        stx   Temp3         ; timeout if unable to leave track 0 being True
1D4E| 20 1313                $17   jsr   ReadIndx      ; read track 0 line
1D51| B0**                        bcs   $21           ; if ccC=1 then /Trk00 is false
PAGE -  92  LISA      FILE: RECAL.TEXT  Sony DRIVER FOR LISA


1D53| 20 6313                      jsr   DoStep        ; step away from Trk 00 ( direction is already set )
1D56| C6 79                        dec   Temp3         ; maximum of 80 steps
1D58| D0F4                        bne   $17           ; loop while less than 80 steps
1D5A| F0**                        beq   $72           ; abort upon underflow
1D5C|
1D51* 09
1D5C| A2 01                  $21   ldx   #OutWard
1D5E| 20 5013                      jsr   SetDretX      ; seek away from spindle toward Track 00
1D61| A2 64                        ldx   #TmOutRcl
1D63| 86 6F                        stx   Temp2
1D65| C6 6F                  $43   dec   Temp2         ; for each step save a counter
1D67| F0**                        beq   $78           ; abort upon underflow
1D69| 20 6313                      jsr   DoStep
1D6C| 20 1313                      jsr   ReadIndx
1D6F| B0F4                        bcs   $43
1D71| A9 64                        lda   #TmOutRcl
1D73| E5 6F                        sbc   Temp2         ; result will be positive & = number of steps
1D75| 20 071D                      jsr   CleScDly      ; calc speed change delay time, return in 'A'
1D78| 20 ****                     jsr   Wait          ; now wait about 150 ms
1D7B| 20 ****                     jsr   SpdChk        ; check/adjust the speed at track 0 & return error
1D7E| A2 00                  $68   ldx   #0
1D80| 86 0C                        stx   HostSeek      ; Done seeking
1D82| 60                          rts
1D83|
1D5A* 27
1D83| A9 10                  $72   lda   #DErrTk0      ; unable to leave track 0 behind
1D85| D0**                        bne   $82
1D67* 1E
1D87| A9 0D                  $78   lda   #DErrCal      ; timeout during recal constant
1D85* 02
1D89| 85 0B                  $82   sta   Iob+DrvError
1D8B| 38                          sec
1D8C| B0F0                        bcs   $68           ; cleanup flag then exit
1D8E|
PAGE -  93  LISA      FILE: RECAL.TEXT  Sony DRIVER FOR LISA


1D8E|                             .PAGE
1D8E|
1D8E| 1D8E                  CLPENTY  .EQU *           ; Entry point used by clamp command
1D8E| 1D8E                  CLAMP    .EQU *           ; ENTRY POINT FOR CLAMP
1D8E| 60                          RTS
1D8F|
1D8F|
PAGE -  94  LISA      FILE: RECAL.TEXT  Sony DRIVER FOR LISA
```

```
1D8F|                       .PAGE
1D8F|                       ;**
1D8F|                       ;
1D8F|                       ;                                   USWAIT
1D8F|                       ;
1D8F|                       ;                          * * * * * * * * * * * * * * * * *
1D8F|                       ;                          *  THE FOLOWING CODES ASUMES  *
1D8F|                       ;                          *  A CYCLE TIME OF 0.5 USEC    *
1D8F|                       ;                          * * * * * * * * * * * * * * * * *
1D8F|                       ;
1D8F|                       ; DELAYS A SPECIFIED NUMBER OF 100 USEC INTERVALS FOR TIMING PURPOSES
1D8F|                       ; A CALL TO USWAIT TAKES  E X A C T L Y  (A-REG * 100USEC) TO COMPLETE,
1D8F|                       ; INCLUDING THE JSR-RTS, THERFORE THE FOLOWING CODE TAKES 301 USEC TO
1D8F|                       ; EXECUTE:
1D8F|                       ;
1D8F|                       ;               LDA #3            1 USEC  (  2 CYCLES)
1D8F|                       ;               JSR USWAIT      300 USEC  (600 CYCLES)
1D8F|                       ;--
1D8F|                       ;
1D8F|                       ; REGISTERS
1D8F|                       ;  IN
1D8F|                       ;       A =       NUMBER OF 100 USEC INTERVALS TO DELAY
1D8F|                       ;       X =       ANY VALUE
1D8F|                       ;       Y =       ANY VALUE
1D8F|                       ;  OUT
1D8F|                       ;       A =       DESTROYED
1D8F|                       ;       X =       DESTROYED
1D8F|                       ;       Y =       UNCHANGED
1D8F|                       ;--
1D8F|
1D8F| 1D8F                  WAITALT   .EQU  *               ; Alternate entry point for DoStep
1D8F| A5 1B                           LDA   StpDly          ; Load delay time
1D91|
1D91| 1D91                  USWAIT    .EQU  *               ; ENTRY POINT FOR USWAIT
1D91| A2 23                           LDX   #023            ; (2)
1D93| CA                    USWAIT1   DEX                   ; (2)
1D94| D0FD                           BNE   USWAIT1         ; (3,2)
1D96| A2 26                           LDX   #026            ; (2)
1D98| EA                             NOP                   ; (2)
1D99| 38                             SEC                   ; (2)
1D9A| E9 01                          SBC   #001            ; (2)
1D9C| D0F5                          BNE   USWAIT1         ; (3,2)
1D9E| EA                             NOP                   ; (2)
1D9F| 60                             RTS                   ; (6)
1DA0|
PAGE -  95  LISA      FILE: RECAL.TEXT  Sony DRIVER FOR LISA


1DA0|                       .PAGE
1DA0|                       ;**
1DA0|                       ;
1DA0|                       ;                                     WAIT
1DA0|                       ;
1DA0|                       ; WAIT A-REG TIMES 5 MILLISECONDS.
1DA0|                       ;
1DA0|                       ;--
1DA0|                       ;
1DA0|                       ; REGISTERS
1DA0|                       ;  IN
1DA0|                       ;       A =       NUMBER IF 5 MSEC INTERVALS TO WAIT
1DA0|                       ;       X =       ANY VALUE
1DA0|                       ;       Y =       ANY VALUE
1DA0|                       ;  OUT
1DA0|                       ;       A =       DESTROYED
1DA0|                       ;       X =       DESTROYED
1DA0|                       ;       Y =       '0'
1DA0|                       ;
1DA0|                       ; CALLS
1DA0|                       ;       USWAIT    WAIT A-REG NUMBER OF 100 USEC INTERVALS
1DA0|                       ;--
1DA0|
1DA0|
1DA0|
1D79* A01D
1D48* A01D
1DA0| 1DA0                  WAIT      .EQU  *               ; ENTRY POINT FOR WAIT
1DA0| A8                             TAY
1DA1| A9 32                  WAIT1     LDA   #50.            ; CALL USWAIT FOR 50*100 = 5000 USEC
1DA3| 20 911D                          JSR   USWAIT
1DA6| 88                              DEY
1DA7| D0F8                           BNE   WAIT1
1DA9| 60                             RTS
1DAA|
1DAA|
1DAA|                       .INCLUDE Npwm
PAGE -  96  LISA      FILE: NPWM.TEXT  Sony DRIVER FOR LISA


1DAA|                       .page
1DAA|                       ;**
1DAA|                       ;--
1DAA|
1DAA| 1DAA                  TimIt     .equ *
1DAA| A9 05                          lda  #5               ; time 6 pulses
1DAC| 85 6F                          sta  Temp2
1DAE| 18                             clc                   ; should stay clear unless explicitly set
```

```
1DAF| 20 ****                    jsr  WtZero        ; look for low, then first high
1DB2| B0**                      bcs  RsOut         ; 2,3     10        no edge within timeout period
1DB4| 20 ****                    jsr  RsEdge        ; 6       8         now at known location
1DB7| B0**                      bcs  RsOut         ; 2,3     10        no edge within timeout period
1DB9| A9 00                      lda  #0            ; 2       12
1DBB| 85 5C                      sta  RangeL        ; 3       15
1DBD| 85 5D                      sta  RangeH        ; 3       18        zero both bytes of counter
1DBF|
1DBF|                        ;**
1DBF|                        ;    WtZero & RsEdge both have 36 cycles(18.0 usec) per tick of RangeL:H
1DBF|                        ;--
1DBF|
1DBF| 20 ****               $20      jsr  WtZero        ; look for low
1DC2| B0**                      bcs  RsOut         ; 2,3
1DC4| 20 ****                    jsr  RsEdge        ; wait for next edge on pulse
1DC7| B0**                      bcs  RsOut         ; 2,3
1DC9| C6 6F                      dec  Temp2         ; 5                 when less than zero then exit
1DCB| 10F2                      bpl  $20           ; 2,3     15*(2+2+5+3) = 180 cycles
1DCD|
1DC7* 04
1DC2* 09
1DB7* 14
1DB2* 19
1DCD| 1DCD               RsOut    .equ *
1DCD| 60                        rts                ; 6                       common exit point
1DCE|
1DC0* CE1D
1DB0* CE1D
1DCE| 1DCE               WtZero   .equ *
1DCE| 24 45                      bit  K000          ; 3           ccV set to Zero
1DD0| 50**                      bvc  RsEd          ; 3       6
1DD2|
1DC5* D21D
1DB5* D21D
1DD2| 1DD2               RsEdge   .equ *            ;             look for a rising edge
1DD2| 24 46                      bit  K0FF          ; 3       3   ccV eq 0 == low, ccV eq 1 == high
1DD4| 70**                      bvs  RsEd          ; 3       6
1DD4* 00
1DD0* 04
1DD6| 1DD6               RsEd     .equ *
1DD6| A0 00                      ldy  #00           ; 2       8
1DD8| A2 A0                      ldx  #0A0          ; 2       10  constant timeout for pulse counting
1DDA|
1DDA| 88                  $10      dey                ; 2       2
1DDB| D0**                      bne  $20           ; 2,3     4,5
1DDD| CA                        dex                ; 2       6
PAGE -  97  LISA      FILE: NPWM.TEXT  Sony DRIVER FOR LISA


1DDE| D0**                      bne  $30           ; 2,3     8,9
1DE0| 38                        sec                ; 2       10  when Xreg = 0 then return error
1DE1| 60                        rts
1DE2|
1DDB* 05
1DE2| EA                  $20      nop                ; 2       7
1DE3| EA                        nop                ; 2       9   waste time
1DE4|
1DDE* 04
1DE4| E6 5C                $30      inc  RangeL        ; 5       14
1DE6| D0**                      bne  $40           ; 2,3     16,17
1DE8| E6 5D                      inc  RangeH        ; 5       21
1DEA| D0**                      bne  $50           ; 3       24  always taken ??
1DEC|
1DE6* 04
1DEC| 48                  $40      pha                ; 3       20  waste more time
1DED| 68                        pla                ; 4       24
1DEE|
1DEA* 02
1DEE| AD 0E08              $50      lda  07L           ; 4       28        Bit7=0 ==> tach low, else tach high
1DF1| 50**                      bvc  $70           ; 2,3     30,31     ccV=0 ==> look for low
1DF3| 10**                      bpl  $90           ; 2,3     32,33     if high then fall thru & exit
1DF5| EA                        nop                ; 2       34
1DF6| 60                  $60      rts                ; 6
1DF7|
1DF1* 04
1DF7| 10FD                $70      bpl  $60           ; 2,3     33,34     if low then exit
1DF9|
1DF3* 04
1DF9| 90DF                $90      bcc  $10           ; 3       36  always taken?? -- 18.0 usec/loop
1DFB|
1DFB| 1DFB               ChkLrg   .equ *
1DFB| A9 05                      lda  #Lrge         ; 5
1DFD| 85 6E                      sta  Temp1         ; for speed adjustment
1DFF| A9 00                      lda  #WLow
1E01| F0**                      beq  ChkComm
1E03|
1E03| 1E03               ChkSml   .equ *
1E03| A9 01                      lda  #Sml          ; 1
1E05| 85 6E                      sta  Temp1
1E07| A9 14                      lda  #TLow
1E09|
1E09|                        ;**
1E09|                        ;  ChkComm
1E09|                        ;     if H < TL
1E09|                        ;       then AdjPlus
1E09|                        ;     else if H > TL
```

```
1E09|                          ;           then TstHih
1E09|                          ;         else if L < TL
1E09|                          ;                 then AdjPlus
1E09|                          ;               else if L = TL
1E09|                          ;                       then OK
1E09|                          ;  TstHih
1E09|                          ;    if H > TH
1E09|                          ;      then AdjMinus
PAGE -  98  LISA       FILE: NPWM.TEXT  Sony DRIVER FOR LISA


1E09|                          ;    else if H <> TH
1E09|                          ;           then OK
1E09|                          ;    else if L <= TH
1E09|                          ;           then OK
1E09|                          ;         else AdjMinus
1E09|                          ;--
1E09|
1E01* 06
1E09| 1E09                     ChkComm   .equ *
1E09| 85 43                              sta  InxPtrL          ; ptr to which constraints ( low or high )
1E0B| A5 23                              lda  CurClass         ; current track class (0:4)
1E0D| 0A                                asl  a                ; *2 to address word array
1E0E| A8                                tay
1E0F| A5 5D                              lda  RangeH
1E11| D1 43                              cmp  @InxPtrL,y       ; high must be greater than or equal to lowtable,y
1E13| 90**                              bcc  AdjPlus          ; high byte low, must spin slower
1E15| D0**                              bne  TstHih           ; high byte greater -- must see if w/in high boundary
1E17| C8                                iny
1E18| A5 5C                              lda  RangeL
1E1A| D1 43                              cmp  @InxPtrL,y       ; must be greater than or equal to table,y
1E1C| 90**                              bcc  AdjPlus          ; high bytes = and low byte low - slow it down
1E1E| F0**                              beq  ChkOk            ; both bytes are equal so within range
1E20| 18                                clc
1E15* 0A
1E21| 1E21                     TstHih    .equ *                ; ccC=1 if branched to here so add +1 for not 'iny'
1E21| 98                                tya
1E22| 69 09                              adc  #TblJmp          ; 9.
1E24| A8                                tay                   ; index now points at high side of range
1E25| B1 43                              lda  @InxPtrL,y
1E27| C5 5D                              cmp  RangeH           ; high byte must be <= high boundary
1E29| 90**                              bcc  AdjMinus         ; high byte is too high, spin disk faster
1E2B| D0**                              bne  ChkOk            ; if <> then low byte must be w/in range
1E2D| C8                                iny                   ; point @ low byte of boundary value
1E2E| B1 43                              lda  @InxPtrL,y
1E30| C5 5C                              cmp  RangeL           ; Low byte must be <= high boundary
1E32| 90**                              bcc  AdjMinus         ; high byte is too high, spin disk faster
1E34|
1E2B* 07
1E1E* 14
1E34| 1E34                     ChkOk     .equ *
1E34| 18                                clc
1E35| 60                                rts
1E36|
1E1C* 18
1E13* 21
1E36| 1E36                     AdjPlus   .equ *                ; ccC = 0 already
1E36| A4 23                              ldy  CurClass
1E38| B9 10 00                           lda  MSpdTbl,y
1E3B| 65 6E                              adc  Temp1            ; add adjustment value
1E3D| 99 10 00                           sta  MSpdTbl,y
1E40| 38                                sec
1E41| 60                                rts
1E42|
1E32* 0E
1E29* 17
PAGE -  99  LISA       FILE: NPWM.TEXT  Sony DRIVER FOR LISA


1E42| 1E42                     AdjMinus  .equ *
1E42| A4 23                              ldy  CurClass
1E44| B9 10 00                           lda  MSpdTbl,y
1E47| 38                                sec
1E48| E5 6E                              sbc  Temp1            ; subtract adjustment value
1E4A| 99 10 00                           sta  MSpdTbl,y
1E4D| 38                                sec
1E4E| 60                                rts
1E4F|
1D7C* 4F1E
1E4F| 1E4F                     SpdChk    .equ *                ;   Main entry point for speed check
1E4F| A9 11                              lda  #WHih            ; '11'x
1E51| 85 44                              sta  InxPtrH
1E53| 20 4913                           jsr  SetTach          ; make sure in Tach Sense mode
1E56| 20 AA1D                           jsr  Timit            ; time 6 pulses & abort on timeout
1E59| B0**                              bcs  SpdErr
1E5B| 20 FB1D                           jsr  ChkLrg           ; must be within +/- 2%
1E5E| 90**                              bcc  SpdDone
1E60| A9 65                              lda  #101.            ; 100 times total for attempts to adjust
1E62| 85 29                              sta  Counter
1E64|
1E64| C6 29                     $10       dec  Counter
1E66| F0**                              beq  SpdErr           ; try for 100 times & abort upon inablility to adjust
1E68| 20 8F13                           jsr  SetSpeed
1E6B| A5 15                              lda  ScDly
1E6D| 20 A01D                           jsr  Wait
1E70| 20 AA1D                           jsr  Timit
```

```
1E73| B0**                          bcs  SpdErr
1E75| 20 FB1D- --                    jsr  ChkLrg
1E78| B0EA                          bcs  $10            ; loop until speed within +/- 2%
1E7A|
1E7A| 38                    $32     sec                 ; make sure error flag is set
1E7B| C6 29                         dec  Counter
1E7D| F0**                          beq  SpdErr         ; try for 100 times & abort upon inability to adjust
1E7F| 20 031E                       jsr  ChkSml         ; now loop until within +/- .5%
1E82| 90**                          bcc  SpdDone
1E84| 20 8F13                       jsr  SetSpeed
1E87| A5 15                         lda  ScDly
1E89| 20 A01D                       jsr  Wait
1E8C| 20 AA1D                       jsr  Timit
1E8F| 90E9                         bcc  $32
1E91|
1E7D* 12
1E73* 1C
1E66* 29
1E59* 36
1E91| 1E91                SpdErr   .equ *
1E91| A9 1B                         lda  #SerrTmt
1E93|
1E82* 0F
1E5E* 33
1E93| 1E93                SpdDone  .equ *
1E93| 60                           rts
1E94|
1E94|
PAGE - 100  LISA      FILE: HIMEM.TEXT  Sony DRIVER FOR LISA


1E94|                         .INCLUDE HIMEM
PAGE - 101  LISA      FILE: HIMEM.TEXT  Sony DRIVER FOR LISA


1E94|                         .PAGE
1E94|                         ;**
1E94|                         ;
1E94|                         ;                              HIMEM
1E94|                         ;                        WELCOME TO VECTOR CITY!
1E94|                         ; HIMEM CONSTANTS
1E94|                         ;
1E94|                         ;--
1E94|
1E94| 00 00 00 00 00 00 00          .ORG    02000-00D       ; LENGTH OF HIGH MEMORY CONSTANTS
1FF3| 43 38 33 41 50 50 4C  CPYRT    .ASCII  "C83APPLE"      ; THIS'L SHOW EM WE MEAN BUSINESS!
1FFA| 45
1FFB| 4C                    RSTJMP   .BYTE   04C             ; A jump vector to the RESTART code
1FFC| 3214                          .WORD   RESTART         ; RESTART 6504 RESET VECTOR
1FFE| 00 00                       .Byte   0, 0            ; Used to hold checksum for ROM verification
2000|
2000|
2000|                         .END
PAGE - 102  LISA      FILE:   SYMBOLTABLE DUMP


AB - Absolute     LB - Label     UD - Undefined     MC - Macro
RF - Ref          DF - Def       PR - Proc          FC - Func
PB - Public       PV - Private   CS - Consts


ADJMINUS LB 1E42|  ADJPLUS  LB 1E36|  ADM1     AB 00D5|  ADM2     AB 00AA|  ADM3     AB 0096|  ADRH     AB 0002|  ADRL
B 0001
ADRMK1   AB 0038|  ADRMK2   AB 0039|  ADRMK3   AB 003A|  ADRMK4   AB 003B|  ADRMK5   AB 003C|  ADRSLEN  AB 0004|  AOK
B 160F
BADADDR  LB 1906|  BADADDR2 LB 191D|  BADADDR4 LB 1921|  BADBAD   LB 192C|  BADGOOD  LB 192B|  BITSLP1  AB 00DE|  BITSLP2
B 00AA
BOOTH    AB 081D|  BOOTL    AB 081C|  BSCNT    AB 0049|  BSERR    LB 189F|  BSTBL    LB 1534|  BUFR12   AB 01F4|  BUFR12SZ
B 000B
C1BT01   AB 02FF|  C1BT02   AB 0300|  C1BT03   AB 0301|  C2BT01   AB 03FE|  C2BT02   AB 03FF|  CA0      AB 0800|  CA1
B 0802
CA2      AB 0804|  CALL     LB 1739|  CHKCOMM  LB 1E09|  CHKDRV   LB 140E|  CHKLRG   LB 1DFB|  CHKOK    LB 1E34|  CHKSML
B 1E03
CKSUM1   AB 0061|  CKSUM2   AB 0062|  CKSUM3   AB 0063|  CLAMP    LB 1D8E|  CLAMPED  AB 0020|  CLCSCDLY LB 1D07|  CLPENTY
B 1D8E
CLRBUF   LB 1CEA|  CLREXIT  LB 1747|  CLRIMSK  LB 1753|  CLRIST   LB 173F|  CLSSTBL  LB 1162|  CLSTSCMD AB 0085|  CMD
B 159E
CMD1     LB 15A2|  CMD3     LB 15C6|  CMD5     LB 15CA|  CMD6     LB 15ED|  CMDCLNUP LB 15E6|  CMDJMP   LB 15F6|  CMDLENG
B 003F
CMDNOERR LB 15E4|  CMDNUMB  AB 0007|  CMDRTS   LB 15EC|  CMDX     AB 007D|  CNFMVAL  AB 00FF|  CNTENA   AB 0810|  CNTPTR
B 0077
COMMAND  AB 0001|  COUNTER  AB 0029|  CPBY01   AB 005E|  CPBY02   AB 005F|  CPCKSUM  AB 0060|  CPYRT    LB 1FF3|  CRECKSUM
B 192D
CRTOP    LB 193B|  CSERR    LB 18A4|  CSERROR  AB 004A|  CSMFND   AB 0050|  CSUM     AB 0055|  CURCLASS AB 0023|  CURTRACK
B 0022
DATMK1   AB 0070|  DATMK2   AB 0071|  DATMK3   AB 0072|  DATMK4   AB 0073|  DATMK5   AB 0074|  DDM3     AB 00AD|  DECRG1
B 1CA3
DELAY    AB 0069|  DERRCAL  AB 000D|  DERRTK0  AB 0010|  DIPINTR  AB 002D|  DIRECT   AB 0059|  DISH     AB 0819|  DISKID
B 0009
DISL     AB 0818|  DNIBL    LB 1100|  DNWHWT   LB 1434|  DOSEEK   LB 1384|  DOSTDONE LB 1383|  DOSTEP   LB 1363|  DOSTRB
B 1321
DRENA    AB 080A|  DRIVE    AB 0002|  DRVCONN  AB 0024|  DRVERROR AB 000B|  EIGHTY   AB 0001|  ENBLTEST LB 170E|  ENBLTST1
B 1710
ERRHDR   AB 001E|  ERRLEN   AB 0007|  ERRSTAT  AB 0008|  ERRWRT   AB 001F|  ESAD     LB 17C7|  FATALERR LB 14DD|  FDIRH
B 081F
FDIRL    AB 081E|  FMTCNFM  AB 0007|  FMTGAP   AB 002B|  FMTTYPE  AB 0025|  FORMAT   LB 1BAD|  FORMERR  LB 1C20|  FORMRTS
```

```
B 1C32
FORMTOP  LB 1BB9|  FORMTRAK LB 1BAD|  FR3TO1   LB 1167|  FRMTDSK  AB 0003|  FRMTTRK  AB 0005|  GERRCLM  AB 0007|  GERRCMD
B 0001
GERRDRV  AB 0002|  GERRENA  AB 0008|  GERRFMPR AB 000A|  GERRINTR AB 0009|  GERRMSK  AB 0006|  GERRSEC  AB 0004|  GERRSID
B 0003
GERRTRK  AB 0005|  GETDIP   LB 14E9|  GETDIP2  LB 150F|  GETDIP3  LB 1515|  GETDIP4  LB 1515|  GETDIP5  LB 1519|  GLOBALS
B 0040
GOBYTE   AB 0000|  HARDABOR LB 1407|  HARDINIT LB 13BF|  HEADELAY AB 0016|  HIGH     AB 0001|  HIHCNT   AB 0076|  HOLDINX
B 002A
HOSTSEEK AB 000C|  IIOB     AB 0030|  IMALIVE  AB 0028|  IMSK     AB 002C|  INCRG1   LB 1C9A|  INDEXH   AB 005B|  INDEXL
B 005A
INIT     LB 1715|  INWARD   AB 0000|  INXPTRH  AB 0044|  INXPTRL  AB 0043|  IOB      AB 0000|  IOBSIZE  AB 0007|  IOSPACE
B 0800
IST      AB 002F|  IWMERROR AB 000E|  IWMMODE  AB 001F|  JSTMTR   LB 17AB|  JUSTRIT  LB 1C43|  K000     AB 0045|  KOFF
B 0046
LISA     PR ----|  LOCALS   AB 0068|  LOOP     LB 1482|  LOOP1    LB 148C|  LOOP2    LB 149A|  LOW      AB 0000|  LOW6
B 003F
LOWCNT   AB 0075|  LRGE     AB 0005|  LSTRB    AB 0806|  LSTUSED  AB 00BF|  LWCMDNO  AB 0083|  MASK     AB 0001|  MAXCLASS
B 0004
MAXCMD   AB 0009|  MAXDDLY  AB 0017|  MAXRECAL AB 001A|  MAXRETRY AB 0019|  MAXSECNT AB 000C|  MAXSPEED AB 0038|  MAXTRACK
B 004F
MINSECNT AB 0008|  MINSPEED AB 00D4|  MINTRACK AB 0000|  MONDLY   AB 001C|  MSPDTBL  AB 0010|  MTENA    AB 0808|  MTRFLG
B 0057
MTRON    AB 0021|  NIBL     LB 1000|  NIBLRETR AB 0020|  NOSIDES  AB 000A|  NOTAOK   LB 160A|  NULLCMD  AB 0080|  OFF
B 0000
OKTOGO   AB 002E|  ON       AB 0001|  ONESEC   AB 00C8|  OUTWARD  AB 0001|  PAGE01   AB 0100|  PAGE02   AB 0200|  PAGE03
B 0300
PERRINT  AB 000C|  PERRROM  AB 000B|  PG1      LB 1A16|  PG1X     LB 1A18|  PG2      LB 1A6A|  PG2LEN   AB 00FF|  PG3
B 1B00
PG3LEN   AB 00FE|  PG3X     LB 1B02|  PRENIB   LB 1577|  PRKCLR0  LB 17CD|  PROGERR1 LB 17CA|  PSTV     LB 1791|  PWMENA
B 0816
PWMREG   AB 0820|  Q6H      AB 080D|  Q6L      AB 080C|  Q7H      AB 080F|  Q7L      AB 080E|  QKDLY    AB 001C|  RACSUM
B 004F
RAEND    AB 004C|  RAERR1   LB 12CD|  RAERR2   LB 12D2|  RAERR3   LB 12D7|  RAERR4   LB 12E1|  RAERR5   LB 12DC|  RAERR6
B 12E3
RAEXIT   LB 12C8|  RAEXIT1  LB 12C9|  RAMTEST  LB 14E8|  RANGEH   AB 005D|  RANGEL   AB 005C|  RASCTR   AB 004D|  RASLP1
B 1299
RASLP2   LB 12AE|  RASTRT   AB 004B|  RATRK    AB 004E|  RCLSTEP  AB 0004|  RD10     LB 1891|  RD9      LB 1888|  RDAD1
B 125F
RDAD2    LB 126F|  RDAD3    LB 1278|  RDAD4    LB 1287|  RDADR    LB 1249|  RDADRTMT AB 0008|  RDASN1   LB 126B|  RDASYN
B 1257
RDBF01   LB 180B|  RDBF02   LB 180D|  RDCSM01  LB 1854|  RDERR    LB 18A6|  RDERR2   LB 18A9|  RDEXIT   LB 189B|  RDS1
B 1308
RDS2     LB 1305|  RDSYNTOP LB 18AE|  RDWASTE  LB 189E|  READ     LB 18E2|  READ01   LB 18E4|  READ1    LB 18E9|  READ16
B 1800
READ3    LB 18FE|  READ4    LB 1905|  READBF   LB 18DE|  READDIP  LB 131E|  READINDX LB 1313|  READIT   LB 130B|  READWP
B 1302
RECALBRT LB 1D2B|  RECALCNT AB 0027|  RECALL   LB 1D2B|  RECALMTR LB 1D24|  RESTART  LB 1432|  RESTART1 LB 1452|  RESTART2
B 145B
RETRYCNT AB 0026|  RFLD     LB 1285|  RID      LB 1314|  ROMERR   LB 14DB|  ROMID    LB 1141|  ROMIDNUM AB 0018|  ROMTEST
B 14A4
RS1      LB 18B8|  RS2      LB 18C3|  RS3      LB 18CE|  RS4      LB 18D8|  RSED     LB 1DD6|  RSEDGE   LB 1DD2|  RSOUT
B 1DCD
RSTJMP   LB 1FFB|  RSYNC    LB 18B5|  RSYNC1   LB 18BD|  RTYFLG   AB 007B|  RW400    LB 1638|  RWCSMFLG AB 0068|  RWTS
B 1604
RWTS4    LB 1619|  RWTS40   LB 1624|  RWTS5    LB 164B|  RWTS7    LB 1654|  RWTSCMD  AB 0081|  RWTSJMP  LB 1663|  SAVADR
B 114B
SAVDAT   LB 1150|  SAVEH    AB 007F|  SAVEL    AB 007E|  SAVINDEX AB 0080|  SCDLY    AB 0015|  SCTRCNT  AB 01D0|  SCTRSAV
B 01D3
SDFND    AB 0052|  SECFND   AB 0053|  SECPRTRK LB 1146|  SECTOR   AB 0004|  SEEK     LB 175E|  SEEK1    LB 1784|  SEEK2
B 1797
SEEKALT  LB 1764|  SEEKEND  LB 17BD|  SEEKERR  AB 000D|  SELSIDE  LB 12E6|  SERRCLMP AB 0016|  SERRFRMT AB 0015|  SERRM1TK
B 001C
SERRNOA9 AB 001A|  SERRPROT AB 0014|  SERRRD   AB 0017|  SERRTMT  AB 001B|  SERRUCLM AB 0019|  SERRWR   AB 0018|  SETDRCT
B 1352
PAGE - 103  LISA      FILE:   SYMBOLTABLE DUMP


SETDRCTX LB 1350|  SETHIGH  LB 12FE|  SETIMSK  LB 174A|  SETLOW   LB 12FA|  SETRMODE LB 139B|  SETSEL   LB 12EF|  SETSPDY
B 1392
SETSPEED LB 138F|  SETTACH  LB 1349|  SHARERAM AB 0010|  SHAREROM LB 1139|  SHARESZ  AB 000D|  SHTOFF   LB 1B93|  SIDE
B 0003
SIDE0SEL AB 081A|  SIDE1SEL AB 081B|  SIDNUMB  AB 01D2|  SMAL     AB 0001|  SPDCHK   LB 1E4F|  SPDDONE  LB 1E93|  SPDERR
B 1E91
SPEED    AB 0006|  STACKST  AB 00CF|  STATUS   AB 0020|  STEPERR  AB 000F|  STPAMT   AB 0058|  STPDLY   AB 001B|  STSLP
B 0048
SV1      AB 006A|  SV2      AB 006B|  SV3      AB 006C|  SV4      AB 006D|  SYNC20   LB 153A|  TBLJMP   AB 0009|  TCKSM1
B 0064
TCKSM2   AB 0065|  TCKSM3   AB 0066|  TEMP1    AB 006E|  TEMP2    AB 006F|  TEMP3    AB 0079|  TEMP4    AB 007A|  TEMPSEC
B 0067
TESTGOB  LB 1132|  TESTTBL  LB 1128|  THINHIH  LB 111E|  THINLOW  LB 1114|  TIMIT    LB 1DAA|  TLOW     AB 0014|  TMON
B 1341
TMOUTRCL AB 0064|  TOOSML   LB 1C33|  TOTCNT   AB 0078|  TRACK    AB 0005|  TRKCLSS  LB 1155|  TRKFLG   AB 0056|  TRKFND
B 0054
TRKNUMB  AB 01D1|  TRMTOFF  LB 132A|  TRNMTRON LB 1331|  TSHIH    LB 1E21|  TTCDRIVE AB 0001|  TTCFCFM  AB 0020|  TTCFVPM
B 0080
TTCMASK  AB 0010|  TTCSECTR AB 0004|  TTCSIDE  AB 0002|  TTCTRACK AB 0008|  TTCWPRT  AB 0040|  TURNMTR  LB 1336|  TURNRND
B 0008
UNCLAMP  LB 13A8|  UPDINT   LB 151C|  UPDINT2  LB 1526|  UPDINT3  LB 1530|  USWAIT   LB 1D91|  USWAIT1  LB 1D93|  UU6
B 007C
VALID1   LB 167D|  VALID2   LB 168C|  VALID20  LB 168E|  VALID3   LB 1696|  VALIDATE LB 1677|  VALIDCF  LB 16E3|  VALIDDR
B 1698
VALIDFV  LB 16F4|  VALIDMA  LB 16D9|  VALIDSE  LB 16C1|  VALIDSI  LB 16AF|  VALIDTR  LB 16CF|  VALIDWP  LB 16EB|  VALJMP
B 16FE
VERIFY   LB 1BE9|  VERTRACK LB 1BED|  VERTRK   LB 1CAD|  VFRTRY   LB 1CCA|  VFTOP    LB 1982|  VFYCKSUM LB 1974|  VFYERR
B 1CD9
```

```
VFYTRK1  LB 1CB7|  VFYTRK2  LB 1CB8|  VLDNOTST LB 1692|  VOLFND   AB 0051|  VRFYDSK  AB 0004|  VRFYTRK  AB 0006|  VTOO
B 1BF2
WADR16   LB 1204|  WAIT     LB 1DA0|  WAIT1    LB 1DA1|  WAITALT  LB 1D8F|  WAITROM  LB 17D5|  WAITROM1 LB 17D8|  WAITROM2
B 17DE
WHIH     AB 0011|  WIDEHIH  LB 110A|  WIDELOW  LB 1100|  WLOW     AB 0000|  WRADR01  LB 1206|  WRAX     LB 1AE0|  WRBYTE
B 1AE8
WRBYTEX  LB 1AE9|  WRITE    LB 19C8|  WRITE16  LB 1A00|  WRITE3   LB 19D1|  WRITEBF  LB 19CB|  WRITRK   LB 1C48|  WRNIBL
B 1AEC
WRSYNTRK LB 1200|  WRTBFCMD AB 0008|  WRTCMD   AB 0001|  WRTKDONE LB 1CA2|  WSTTM    LB 1248|  WTHIH    AB 0042|  WTLOW
B 0040
WTMID    AB 0041|  WTZERO   LB 1DCE|  XFER1    LB 1465|  ZERO     AB 0000|
PAGE - 104  LISA      FILE:   Sony DRIVER FOR LISA


Current minimum space is    32767 words.
1246* 001A
1232* E81A
1227* E91A
121F* E91A
1575* EC1A
1570* EC1A
156B* EC1A
1566* EC1A
123C* EC1A
1237* EC1A
1218* EC1A
123F* 931B
1669* AC1B
166D* AC1B
166B* E81B
166F* EC1B
17A4* 071D
1504* 241D
1922* 2B1D
1675* 8D1D
1374* 8F1D
17B4* A01D
1501* A01D
144C* A01D
13CB* A01D
13B4* A01D
1347* A01D
17B8* 4F1E

Assembly complete:       3671 lines
     0   Errors flagged on this Assembly
ESY.TEXT=sDRIVE2/SY$P3&2/SY$P:SY.TEXT"".sR""sx/ "Q1".,R3".xtJ"sh0x0 &2/SY$P:SY.TEXT"".sR""sx/ "Q1".,R3".xt
```