


```

0000 1: ;-----
0000 2: ;
0000 3: ;      Copyright 2006 Intergraded Logic Systems
0000 4: ;      Source Code is Copyright Stephen J. Car
0000 5: ;
0000 6: ;
0000 7: ;      Please do not share this source!
0000 8: ;-----
0000 9: ;      Low Level Formatter for mio for real.dos v1.0)
0000 10: ;
0000 11: ;-----
0000 12: ;      Real.dos hdfmtmio.com
0000 13: ;      This program is to replace
0000 14: ;      hdfmtmio.com
0000 15: ;
0000 16: ;      Code History.  Init design,  November 5, 2005
0000 17: ;
0000 18: ;
0000 19: ;
0000 20: ;
0000 21: ;
0000 22: ;
0000 23: ;      ~~~~~
0000 24: ;-----
0000 25: ; Notes: o This source code MAY NOT be placed for download
0000 26: ;      o "mea" is a macro that loads the address of the
0000 27: ;      into the pointer specified by the second field.
0000 28: ;-----
0000 29: ; Assembler: MADMAC (tm) ST Cross Assembler (Atari Corp)
0000 30: ;      XASM  st and IBM VERSIONS
0000 31: ;-----
0000 32: ;=====
0000 33:
0000 34: com_file_name:      .macro
0000 35: ;      dc.b  13,"filename.ext",$9b
0000 36: ;      dc.b  12,"Hdfmtmio.com",$9b
0000 37: ;      .endm
0000 38:
0000 39: ;
0000 40: ;~~~~~
0000 41: ; File revision history. Version Number in hex
0000 42: ;
0000 43: file_ver:      equ  $12
0000 44: ;~~~~~
0000 45: ; the Month in dec for the first time this revision
0000 46: ;was compiled
0000 47: ;
0000 48: c_month:      equ  10
0000 49: ;~~~~~
0000 50: ; the day in dec for the first time this revision
0000 51: ;was compiled
0000 52: c_day:      equ  13
0000 53: ;~~~~~

```

```
0000 54: ; the year in dec for the first time this revision
0000 55: ;was compiled
0000 56: c_year:          equ 2012
0000 57: ;~~~~~
0000 58: ; Type of compiler used in program
0000 59: ;
0000 60: ; 0 = unknown Compiler
0000 61: ; 1 = xasm
0000 62: ; 2 = mac_65
0000 63: ; 3 = basic
0000 64: ; 4 = compiled basic xl
0000 65: ; 5 = C65
0000 66: ; 6 = Action
0000 67: ;
0000 68: ; We can add more as time goes on!
0000 69: ;
0000 70: ;
0000 71: ;
0000 72: xasm:             equ 1
0000 73: mac_65:           equ 2
0000 74: basic:            equ 3
0000 75: basicxl:          equ 4
0000 76: c65:             equ 5
0000 77: action:           equ 6
0000 78:
0000 79: file_compiler:     equ xasm
0000 80: ;~~~~~
0000 81: ; is this relocatable code ?
0000 82: ; anyother value other than 1 or 2 would be unknown
0000 83: r..yes:            equ 1
0000 84: r..no:             equ 2
0000 85: ;
0000 86: ;~~~~~
0000 87: ; Gotta define if it can be relocated
0000 88: l_relocatable:     equ r..no
0000 89: ;
0000 90: ;
0000 91: ;~~~~~
0000 92: ; Gotta define if it can be relocated
0000 93: r..crl:            equ $7d
0000 94: r..crlf:           equ $9b
0000 95: r..space:          equ $20
0000 96:
0000 97: l_frstscreenbyte: equ r..space
0000 98: ;~~~~~
0000 99: ; The language the output file is in.
0000 100: ;
0000 101: ;
0000 102: ; 0 = Undefined
0000 103: ; 1 = English
0000 104: ; 2 = German
0000 105: ;
0000 106: r..language:       equ 1
```

```

0000 107: ;~~~~~
0000 108:
0000 109:
0000 110: ; MIO registers
0000 111: ;-----
0000 112: mio.RAMPAGE equ $D1E0 ; write ram page address
0000 113: daIn equ $D1E1 ; HD data input
0000 114: daout equ $D1E1 ; HD & printer data out
0000 115: inputs equ $D1E2 ; monitor handshake lines
0000 116: miscout equ $D1E2 ; several misc outputs (ram
0000 117: irqstat equ $D1FF ; read IRQ status
0000 118: setrom equ $D1FF ; write ROM bank address
0000 119: rststro equ $D1E0 ; strobe reset low (goes hi
0000 120:
0000 121: ; Bit definition of INPUTS
0000 122: ;-----
0000 123: prfault equ $10 ; printer fault
0000 124: prbusy equ $40 ; printer busy
0000 125: hdc d equ $01 ; hard disk command/data
0000 126: hdmsg equ $02 ; MSG input
0000 127: hdio equ $04 ; hard disk input/output control
0000 128: hdbusy equ $20 ; hard disk busy line
0000 129: hdreq equ $80 ; hard disk request line
0000 130:
0000 131: ; Bit definitions of MISCOUT
0000 132: ;-----
0000 133: selstro equ $10 ; select LOW bit...
0000 134: ramen equ $20 ; ram enable bit
0000 135:
0000 136:
0000 137: ; Configurable operating parameters
0000 138: ;-----
0000 139: ;memkey equ $D600 ; 16 byte key... indicates
0000 140: mio.dridata equ $D610 ; data table describing dis
0000 141: ; +00: First Logical block address
0000 142: ; +03: Last+1 Logical block address
0000 143: ; +06: Drive number
0000 144: ; Bit[2__0] = SCSI/SASI ID if Ha
0000 145: ; = Drive number if Re
0000 146: ; Bit[7] : 1 = Drive is a Hard D
0000 147: ; Bit[6] : 1 = Drive is a RAM Di
0000 148: ; Bit[5] : 1 = Reassign to SIO t
0000 149: ; if Bit[7__5] = 0, then un
0000 150: ; +07: flags (SCSI)
0000 151: ; Bit[7__5] = logical unit numbe
0000 152: ; Bit[4] : 1 = SASI type interfa
0000 153: ; Bit[3] : 1 = Disk Write Locked
0000 154:
0000 155:
0000 156: mio.prunit equ $D654 ; Printer device # (0=disab
0000 157:
0000 158: dritype equ $D680 ; data describing each hard
0000 159: ; +00: Total # cylinders on Hard D

```

```

0000 160:          ; +02: Number of Heads on Hard Dri
0000 161:          ; +03: Cylinder to start reduced w
0000 162:          ; +05: Precomp value (usually 0)
0000 163:          ; +07: ECC burst length (usually $
0000 164:
0000 165:
0000 166: prirq equ   $D6C8 ; IRQ enable for parallel spooler
0000 167:
0000 168:
0000 169:
0000 170: ; Hard Disk variables
0000 171: ; -----
0000 172: devidx equ   $D6D0 ; current device ID
0000 173: direct equ   $D6D1 ; read/write direction
0000 174: loops equ   $D6D2
0000 175: unitID equ   $D6D3 ; SCSI bus ID for drive access
0000 176: LUN equ     $D6D4 ; logical unit number
0000 177: byxfer equ   $D6D5 ; length of buffer for read/write o
0000 178: bloklen equ   $D6D6 ; length of data block
0000 179: tsec equ    $D6D8 ; 2 bytes -- temp for RAMDISK
0000 180: comfram equ   $D6DA ; 6 byte command frame
0000 181: driptr equ   $30 ; ptr to drive buffer
0000 182: curpage equ   $D6FC ; RAMpage shadow
0000 183: Smisc equ    $D6FD ; MISCOUT shadow
0000 184: fieldln equ   $40C ; field length for filling
0000 185:
0000 186: zp equ      $D3
0000 187: dptr equ    $d5
0000 188:
0000 189: tp equ      $d5
0000 190: ntp equ     $d7
0000 191:
0000 192:
0000 193:
0000 194:          .include equates
0000 195:          .include globals
0000 196:          .include macros
0000 197:
3000 198:          .org $3000
3000 199:
3000 200: win_start:
3000 201: header_info:
3000 202:          .include header
32a7 203:
32a7 204: ; Entry of format program
32a7 205: ; -----
32a7 206: start
32a7 a960 207: start2 lda   #$60
32a9 8da732 208:      sta   start2
32ac ba 209:      tsx
32ad 8e444c 210:      stx   stack
32b0 211:
32b0 20f64c 212:      jsr   print

```

```

32b3 202020 213:    dc.b  "    MIO Hard Disk Physical Format"
32d5 9b9b9b 214:    dc.b  $9B,$9B,$9B,-1
32d9      215:
32d9 20fd34 216:    jsr   build      ; build table containing a
32dc ad2a4c 217:    lda   curopt      ; of possible controllers
32df f003 218:    beq   f.prer
32e1 4ca233 219:    jmp   f.somopt
32e4 20f64c 220: f.prer jsr   print
32e7 596f75 221:    dc.b  "You must first use configuration menu",$9B
330d 746f20 222:    dc.b  "to map 'Dn:' to Hard Drives --- Note",$9B
3332 746861 223:    dc.b  "that 'Start Sec#' and 'End Sec#' are",$9B
3357 696e69 224:    dc.b  "initially set after the Hard Drives",$9B
337b 617265 225:    dc.b  "are formatted. Press SELECT+RESET.",$9B,-1
339f 4c724a 226:    jmp   abort
33a2      227:
33a2 204f35 228: f.somopt jsr   options      ; display list of c
33a5 20f64c 229:    jsr   print
33a8 9b5365 230:    dc.b  $9B,"Select Hard Drive to Format: ",-1
33c7 209f4d 231: f.geopt2 jsr   inkey
33ca c91b 232:    cmp   #27
33cc d017 233:    bne   f.isnoab
33ce 20f64c 234: f.fabort jsr   print
33d1 9b9b55 235:    dc.b  $9B,$9B,"User Aborted",$9B,$9B,-1
33e2 4c724a 236:    jmp   abort
33e5 aa 237: f.isnoab tax
33e6 38 238:    sec
33e7 e931 239:    sbc   #'1'
33e9 90dc 240:    bcc   f.geopt2
33eb cd2a4c 241:    cmp   curopt
33ee b0d7 242:    bcs   f.geopt2
33f0 8d294c 243:    sta   opt          ; set unit to be formatted
33f3 8a 244:    txa
33f4 20dc4c 245:    jsr   prch          ; print selection #
33f7 20f64c 246:    jsr   print
33fa 9b9bff 247:    dc.b  $9B,$9B,-1
33fd      248:
33fd 20764f 249:    jsr   seram          ; set to system RAM to read
3400 205a4b 250:    jsr   sedrive        ; search for this drive
3403 208b4b 251:    jsr   geparam        ; get params from this driv
3406      252:
3406 ac294c 253:    ldy   opt
3409 b9344c 254:    lda   optIDt,y
340c 8d4d4e 255:    sta   busID          ; set unit ID for HD comman
340f b92c4c 256:    lda   optLUNt,y      ; and LUN
3412 8d4e4e 257:    sta   Unit
3415      258:
3415 20f64c 259:    jsr   print
3418 9b9b20 260:    dc.b  $9B,$9B," ***** Drive Selection *****",$9
343b      261:
343b 20f64c 262:    jsr   print
343e 312e20 263:    dc.b  "1. Adaptec ACB 4000A/4070",$9B
3459 322e20 264:    dc.b  "2. Iomega Alpha/Beta Series",$9B
3476 332e20 265:    dc.b  "3. XEBEC 1410/1410A, WD 1002-SHD",$9B

```

```

3498 342e20 266:    dc.b  "4. Seagate Embedded drives", $9B, -1
34b5 20f64c 267:    jsr  print
34b8 9b5365 268:    dc.b  $9B, "Select Your Drive: ", -1
34cd      269:
34cd 209f4d 270: f.inke4    jsr  inkey
34d0 e931   271:    sbc  #'1'
34d2 90f9   272:    bcc  f.inke4
34d4 c904   273:    cmp  #4
34d6 b0f5   274:    bcs  f.inke4
34d8      275:
34d8 48     276:    pha
34d9 6931   277:    adc  #'1'
34db 20dc4c 278:    jsr  prch
34de 20f64c 279:    jsr  print
34e1 9b9bff 280:    dc.b  $9B, $9B, -1
34e4 20be4a 281:    jsr  yousure      ; make sure he is sure...
34e7 68     282:    pla
34e8 0a     283:    asl
34e9 aa     284:    tax
34ea bdf434 285:    lda  f.drtyp+1,x
34ed 48     286:    pha
34ee bdf334 287:    lda  f.drtyp,x
34f1 48     288:    pha
34f2 60     289:    rts
34f3      290:
34f3      291: f.drtyp
34f3 8637   292:    dc.w  adaptec-1
34f5 6b41   293:    dc.w  iomega-1
34f7 e941   294:    dc.w  xebec-1
34f9 f936   295:    dc.w  sea_mfm-1
34fb f936   296:    dc.w  sea_rll-1
34fd      297: ;
34fd      298:
34fd      299:
34fd      300: ;    Build ID/LUN table from 8 drives
34fd      301: ;    -----
34fd      302: build
34fd a908   303:    lda  #8
34ff 8d0103 304:    sta  dunit
3502 20764f 305:    jsr  seram      ; set to system RAM
3505 a000   306:    ldy  #0
3507 8c2a4c 307:    sty  curopt
350a      308: f.unloop
350a 20184c 309:    jsr  unitX      ; calc unit #
350d bd16d6 310:    lda  mio.dridata+6,x
3510 1037   311:    bpl  f.nedri1
3512 2907   312:    and  #7
3514 ac2a4c 313:    ldy  curopt
3517 99344c 314:    sta  optIDt,y
351a bd17d6 315:    lda  mio.dridata+7,x
351d 29e0   316:    and  #$E0
351f 992c4c 317:    sta  optLUNt,y
3522 a900   318:    lda  #0

```

```

3524 993c4c 319:      sta  multity
3527 88      320: f.cnewopt  dey
3528 301c 321:      bmi  f.newopt      ; this is first occurrence,
352a bd16d6 322:      lda  mio.dridata+6,x
352d 2907 323:      and  #7
352f d9344c 324:      cmp  optIDt,y
3532 d0f3 325:      bne  f.cnewopt
3534 bd17d6 326:      lda  mio.dridata+7,x
3537 29e0 327:      and  #$E0
3539 d92c4c 328:      cmp  optLUNt,y
353c d0e9 329:      bne  f.cnewopt
353e a9ff 330:      lda  #-1
3540 993c4c 331:      sta  multity      ; indicate at least two dri
3543 ce2a4c 332:      dec  curopt
3546 ee2a4c 333: f.newopt  inc  curopt
3549 ce0103 334: f.nedril  dec  dunit
354c d0bc 335:      bne  f.unloop      ; to next drive...
354e 60      336:      rts
354f          337: ;
354f          338:
354f          339:
354f          340: ;   Display list of controllers to user
354f          341: ;   -----
354f          342: options
354f 20f64c 343:      jsr  print
3552 202020 344:      dc.b  "   Intf ID LUN Cyln Head Drive #s",$9B
3575 202020 345:      dc.b  "   -----",$9B,-1
3599          346:
3599 a900 347:      lda  #0
359b 8d294c 348:      sta  opt      ; show options...
359e ad294c 349: f.optlp  lda  opt
35a1 18      350:      clc
35a2 6931 351:      adc  #'1'
35a4 20dc4c 352:      jsr  prch
35a7 20f64c 353:      jsr  print
35aa 202d20 354:      dc.b  " - ",-1
35ae 20764f 355:      jsr  seram      ; set to system RAM to read
35b1 205a4b 356:      jsr  sedrive     ; search for ID/LUN drive..
35b4 208b4b 357:      jsr  geparam     ; get drive parameters
35b7 20a64b 358:      jsr  prininfo    ; print an info line...
35ba          359:
35ba ad0103 360: f.tagain  lda  dunit
35bd 0930 361:      ora  #'0'
35bf 20dc4c 362:      jsr  prch
35c2 20764f 363:      jsr  seram
35c5 207e4b 364:      jsr  csedri      ; continue search on drive
35c8 d02a 365:      bne  f.neopt      ; not found... advance to
35ca          366:
35ca bd17d6 367:      lda  mio.dridata+7,x ; Check for consistency of
35cd 2910 368:      and  #$10      ; ~~~~~~
35cf cd2b4c 369:      cmp  sasif      ; sasi flag
35d2 d031 370:      bne  f.incons    ; inconsistant data
35d4 bd80d6 371:      lda  dritype,x

```



```

35d7 cd274c 372:    cmp  cyln+1
35da d029 373:    bne  f.incons
35dc bd81d6 374:    lda  dritype+1,x
35df cd264c 375:    cmp  cyln
35e2 d021 376:    bne  f.incons
35e4 bd82d6 377:    lda  dritype+2,x
35e7 cd284c 378:    cmp  heads
35ea d019 379:    bne  f.incons
35ec a92c 380:    lda  #','
35ee 20dc4c 381:    jsr  prch      ; print comma, then drive #
35f1 4cba35 382:    jmp  f.tagain
35f4      383:
35f4 a99b 384: f.neopt  lda  #$9B
35f6 20dc4c 385:    jsr  prch
35f9 ee294c 386:    inc  opt
35fc ad294c 387:    lda  opt      ; loop for next choice
35ff cd2a4c 388:    cmp  curopt
3602 d09a 389:    bne  f.optlp
3604 60 390:    rts
3605      391:
3605 20f64c 392: f.incons  jsr  print
3608 9b9b9b 393:    dc.b  $9B,$9B,$9B
360b 486172 394:    dc.b  "Hard Drive is partitioned into",$9B
362a 736576 395:    dc.b  "several units, but the units do",$9B
364a 6e6f74 396:    dc.b  "not contain consistant data about",$9B
366c 746865 397:    dc.b  "the Hard Drive. The datum that must",$9B
3691 626520 398:    dc.b  "be consistant are: Cylinders, Heads,",$9B
36b6 616e64 399:    dc.b  "and Interface type. Press",$9B
36d1 53454c 400:    dc.b  "SELECT+RESET for configuration menu.",$9B,
36f7 4c724a 401:    jmp  abort
36fa      402: ;
36fa      403: ;
36fa      404: ;
36fa      405: ;=====
36fa      406: ;   Format and verify Seagate ST225N
36fa      407: ;=====
36fa      408: ;
36fa      409: ;   Format Seagate Drive
36fa      410: ;   -----
36fa      411: sea_mfm
36fa      412: sea_rll
36fa      413: ;   sec
36fa      414: ;   lda  cyln
36fa      415: ;   sbc  #2
36fa      416: ;   sta  cyln      ; for SEAGATE... subtract 2
36fa      417: ;   lda  cyln+1
36fa      418: ;   sbc  #0
36fa      419: ;   sta  cyln+1
36fa      420: ;
36fa      421: ;   lda  heads      ; need cyln * heads * 32
36fa      422: ;   sta  mula
36fa      423: ;   lda  cyln
36fa      424: ;   sta  mulb

```

```

36fa 425: ; lda  cyn+1
36fa 426: ; sta  mulb+1
36fa 427: ; jsr  multiply    ; res = mula * mulb
36fa 428: ; ldx  #5          ; now multiply by 32 for to
36fa 429: ;..tlop3  asl   res
36fa 430: ; rol   res+1
36fa 431: ; rol   res+2
36fa 432: ; dex
36fa 433: ; bne   ..tlop3
36fa 434: ;
36fa 435: ; sec
36fa 436: ; lda   res
36fa 437: ; sbc   #100
36fa 438: ; sta   res
36fa 439: ; lda   res+1
36fa 440: ; sbc   #0          ; For SEAGATE subtract 100
36fa 441: ; sta   res+1        ; reassigned blocks
36fa 442: ; lda   res+2
36fa 443: ; sbc   #0
36fa 444: ; sta   res+2
36fa 445: ;
36fa 446: ; lda   res
36fa 447: ; sta   nulogblo+2    ; set # logical blocks
36fa 448: ; lda   res+1
36fa 449: ; sta   nulogblo+1
36fa 450: ; lda   res+2
36fa 451: ; sta   nulogblo
36fa 452:
36fa 20be4d 453: jsr  setcom
36fd 06 454:  dc.b  6          ; Seagate Mode Select
36fe 150000 455:  dc.b  $15,0,0,0,$0C,0
3704 456:
3704 20f74d 457: jsr  hard          ; perform hard disk I/O
3707 2e37 458:  dc.w  modsel      ; buffer is MODSEL
3709 0c00 459:  dc.w  12          ; bytes
370b ff 460:  dc.b  -1         ; direction
370c 1003 461:  bpl  f.okhd1
370e 4c274a 462:  jmp  erconfig
3711 463:
3711 20be4d 464: f.okhd1  jsr  setcom
3714 06 465:  dc.b  6          ; standard format drive
3715 040000 466:  dc.b  4,0,0,0,0,0
371b 20f74d 467: jsr  hard          ; perform format... no data
371e 000000 468:  dc.w  0,0
3722 00 469:  dc.b  0
3723 1003 470:  bpl  f.okhd2
3725 4c4e4a 471:  jmp  erformat
3728 472:
3728 208646 473: f.okhd2  jsr  readcap    ; read drive capaci
372b 474: ; jsr  verify    ; verify all logical blocks
372b 4cdc46 475:  jmp  xformat
372e 476:
372e 477:          ; Mode Select data block

```

```

372e 478: ; -----
372e 000000 479: modsel dc.b 0,0,0,8
3732 00 480: dc.b 0
3733 481: nulogblo
3733 000000 482: dc.b 0,0,0 ; # logical blocks
3736 000001 483: dc.b 0,0,1,0 ; block length
373a 484: ;
373a 485:
373a 486:
373a 487: ;
373a 488: ;=====
373a 489: ; Format and Verify Rodime RO 650/652
373a 490: ;=====
373a 491:
373a 492: ; Format a Rodime RO 650 Hard Disk
373a 493: ; -----
373a 494: rodime
373a 20be4d 495: jsr setcom
373d 06 496: dc.b 6 ; rodime mode select
373e 150000 497: dc.b $15,0,0,0,25,0
3744 20f74d 498: jsr hard
3747 6e37 499: dc.w rmodsel
3749 1900 500: dc.w 25
374b ff 501: dc.b -1
374c 1003 502: bpl f.rokhd1
374e 4c274a 503: jmp erconfig
3751 504:
3751 20be4d 505: f.rokhd1 jsr setcom
3754 06 506: dc.b 6 ; standard format disk
3755 040000 507: dc.b 4,0,0,0,0,0
375b 20f74d 508: jsr hard
375e 000000 509: dc.w 0,0
3762 00 510: dc.b 0
3763 1003 511: bpl f.rokhd2
3765 4c4e4a 512: jmp erformat
3768 513:
3768 208646 514: f.rokhd2 jsr readcap ; read drive capaci
376b 515: ; jsr verify
376b 4cdc46 516: jmp xformat
376e 517:
376e 518: ; Mode Select data block fo
376e 519: ; -----
376e 000000 520: rmodsel dc.b 0,0,0,0
3772 0313 521: dc.b $03,19 ; Page 3, page len
3774 000000 522: dc.b 0,0,0,0,0,0,0,0,0
377e 0100 523: dc.b 1,0 ; Bytes/sector
3780 000000 524: dc.b 0,0,0,0,0,0,0
3787 525: ;
3787 526:
3787 527:
3787 528: ;
3787 529: ;=====
3787 530: ; Format and verify Adaptec ACB-4000

```

```

3787      531: ;=====
3787      532:
3787      533: ;   Format an Adaptec ACB-4000 Hard Disk
3787      534: ;   -----
3787      535: adaptec
3787 a9ff  536:   lda  #-1
3789 2c    537:   dc.b $2C
378a a900 538: adaptec  lda  #0
378c 8d353c 539:   sta  f.fixed
378f      540:
378f 20f64c 541:   jsr  print
3792 9b9b53 542:   dc.b $9B,$9B,"Select stepping mode:",$9b,$9b
37ab 20302e 543:   dc.b " 0. Nonbuffered (ST506)","$9b
37c3 20312e 544:   dc.b " 1. Slow buffered (ST4096)","$9b
37de 20322e 545:   dc.b " 2. Fast buffered (others)","$9B,$9b
37fa 576869 546:   dc.b "Which?",-1
3801      547:
3801 209f4d 548: f.wail jsr  inkey
3804 c930  549:   cmp  #'0'
3806 90f9  550:   bcc  f.wail
3808 c933  551:   cmp  #'3'
380a b0f5  552:   bcs  f.wail
380c 48    553:   pha
380d 290f  554:   and  #15
380f 8d513c 555:   sta  f.stepra    ; set step rate...
3812 68    556:   pla
3813 20dc4c 557:   jsr  prch
3816 20f64c 558:   jsr  print
3819 9b9bff 559:   dc.b $9B,$9B,-1
381c      560:
381c a200  561:   ldx  #0
381e a96c  562:   lda  #$6C
3820 9d6c40 563: f.filit sta  f.sdata+20,x    ; set data pattern
3823 e8    564:   inx
3824 d0fa  565:   bne  f.filit
3826      566:
3826 ad284c 567:   lda  heads
3829 8d4b3c 568:   sta  f.ahead
382c ad264c 569:   lda  cyln
382f 8d4a3c 570:   sta  f.acyln+1    ; set # heads and cylinders
3832 ad274c 571:   lda  cyln+1
3835 8d493c 572:   sta  f.acyln
3838      573:
3838 a90a  574:   lda  #10
383a 8d3a3c 575:   sta  f.tpass    ; allow 4 total passes...
383d      576:
383d a900  577:   lda  #0
383f 8d393c 578:   sta  f.times    ; # errors in etable...
3842 a958  579:   lda  #low f.etable ; set address of error tabl
3844 8df339 580:   sta  f.eaddr
3847 a93c  581:   lda  #high f.etable
3849 8df439 582:   sta  f.eaddr+1
384c      583:

```

```

384c 2c353c 584:    bit   f.fixed
384f 101a 585:    bpl   f.remov
3851 20be4d 586:    jsr   setcom
3854 06 587:    dc.b  6
3855 150000 588:    dc.b  $15,0,0,0,22,0
385b a901 589:    lda   #1
385d 8d483c 590:    sta   f.fortc
3860 20f74d 591:    jsr   hard
3863 3c3c 592:    dc.w  f.modsel
3865 1600 593:    dc.w  22
3867 ff 594:    dc.b  -1
3868 4c8238 595:    jmp   f.adap44
386b 596:
386b 20be4d 597: f.remov    jsr   setcom
386e 06 598:    dc.b  6
386f 150000 599:    dc.b  $15,0,0,0,24,0
3875 a902 600:    lda   #2
3877 8d483c 601:    sta   f.fortc
387a 20f74d 602:    jsr   hard
387d 3c3c 603:    dc.w  f.modsel
387f 1800 604:    dc.w  24
3881 ff 605:    dc.b  -1
3882 606:
3882 1003 607: f.adap44    bpl   f.format
3884 4c274a 608:    jmp   erconfig
3887 609:
3887 20be4d 610: f.format    jsr   setcom
388a 06 611:    dc.b  6          ; initially try an interlea
388b 040000 612:    dc.b  4,0,0,0,2,0 ; fill with data pattern $6
3891 20f74d 613:    jsr   hard
3894 000000 614:    dc.w  0,0        ; format standard...
3898 00 615:    dc.b  0
3899 1003 616:    bpl   f.vtop
389b 4c4e4a 617:    jmp   erformat
389e 618:
389e 208646 619: f.vtop jsr   readcap    ; get # blocks on drive...
38a1 ad7d4a 620:    lda   res
38a4 8dd94c 621:    sta   bycou
38a7 ad7e4a 622:    lda   res+1
38aa 8dda4c 623:    sta   bycou+1
38ad ad7f4a 624:    lda   res+2
38b0 8ddb4c 625:    sta   bycou+2
38b3 20f64c 626:    jsr   print
38b6 9b9b54 627:    dc.b  $9B,$9B,"Total blocks to verify: ",-1
38d1 208c4c 628:    jsr   prbycou
38d4 20f64c 629:    jsr   print
38d7 9bff 630:    dc.b  $9B,-1
38d9 631:
38d9 a900 632:    lda   #0
38db 8d5739 633:    sta   f.fblock+1
38de 8d5839 634:    sta   f.fblock+2    ; zero first block # to tes
38e1 8d5939 635:    sta   f.fblock+3
38e4 8d3b3c 636:    sta   f.terror    ; # errors this time around

```

```

38e7      637:
38e7 38    638: f.top sec          ; Set # blocks to verify in
38e8 ad7d4a 639:   lda  res          ; ~~~~~
38eb ed5939 640:   sbc  f.fblock+3
38ee 8d363c 641:   sta  f.blocks      ; # blocks left to test
38f1 ad7e4a 642:   lda  res+1
38f4 ed5839 643:   sbc  f.fblock+2
38f7 8d373c 644:   sta  f.blocks+1
38fa aa    645:   tax
38fb ad7f4a 646:   lda  res+2
38fe ed5739 647:   sbc  f.fblock+1
3901 8d383c 648:   sta  f.blocks+2
3904 d00d   649:   bne  f.max          ; if >65536, then set max
3906 e010   650:   cpx  #$10
3908 b009   651:   bcs  f.max
390a ad363c 652:   lda  f.blocks
390d ae373c 653:   ldx  f.blocks+1     ; get # blocks to test
3910 4c1739 654:   jmp  f.set
3913 a900   655: f.max lda  #0
3915 a210   656:   ldx  #$10
3917      657:
3917 8d5c39 658: f.set sta  f.nblock+2 ; set # blocks to test...
391a 8d6340 659:   sta  f.sdata+11
391d 8e5b39 660:   stx  f.nblock+1
3920 8e6240 661:   stx  f.sdata+10
3923 0d5b39 662:   ora  f.nblock+1
3926 d003   663:   bne  f.okt
3928 4cc43a 664:   jmp  f.done1       ; if no blocks... then exit
392b      665:
392b 20f64c 666: f.okt jsr  print
392e 9c9d41 667:   dc.b 156,157,"At Block #",-1
393b ad5939 668:   lda  f.fblock+3
393e 8dd94c 669:   sta  bycou          ; Display block number veri
3941 ad5839 670:   lda  f.fblock+2     ; ~~~~~
3944 8dda4c 671:   sta  bycou+1
3947 ad5739 672:   lda  f.fblock+1
394a 8ddb4c 673:   sta  bycou+2
394d 208c4c 674:   jsr  prbycou
3950      675:
3950 20be4d 676:   jsr  setcom
3953 0a     677:   dc.b 10
3954 3110   678:   dc.b $31,$10
3956 000000 679: f.fblock dc.b 0,0,0    ; starting block ad
395a 008000 680: f.nblock dc.b 0,$80,$00,0 ; search 32768 bloc
395e 20f74d 681:   jsr  hard
3961 5840   682:   dc.w f.sdata      ; block
3963 1401   683:   dc.w 20+256       ; # bytes
3965 ff     684:   dc.b -1           ; write
3966 aa     685:   tax
3967 2906   686:   and  #6
3969 d01e   687:   bne  f.cksen       ; must either be bad sector
396b      688:
396b 18     689:   clc               ; Search failed (all data g

```

```

396c ad5939 690:   lda   f.fblock+3   ; -----
396f 6d5c39 691:   adc   f.nblock+2   ; set next first block...
3972 8d5939 692:   sta   f.fblock+3
3975 ad5839 693:   lda   f.fblock+2
3978 6d5b39 694:   adc   f.nblock+1
397b 8d5839 695:   sta   f.fblock+2
397e ad5739 696:   lda   f.fblock+1
3981 6900 697:   adc   #0
3983 8d5739 698:   sta   f.fblock+1
3986 4ce738 699:   jmp   f.top        ; repeat until all tested..
3989      700:
3989      701:           ; Returns error from verify
3989 ad484e 702: f.cksen   lda   senseb   ; ~~~~~
398c 3029 703:   bmi   f.loblo   ; jump if have logical bloc
398e 20f64c 704:   jsr   print
3991 9b9b53 705:   dc.b  $9B,$9B,"Sense data has no logical block",$
39b4 4c724a 706:   jmp   abort
39b7      707:
39b7 ad494e 708: f.loblo   lda   senseb+1   ; get sector number
39ba 8deb39 709:   sta   f.xlat   ; ~~~~~
39bd 8ddb4c 710:   sta   bycou+2
39c0 ad4a4e 711:   lda   senseb+2
39c3 8dec39 712:   sta   f.xlat+1
39c6 8dda4c 713:   sta   bycou+1
39c9 ad4b4e 714:   lda   senseb+3
39cc 8ded39 715:   sta   f.xlat+2
39cf 8dd94c 716:   sta   bycou
39d2      717:
39d2 20f64c 718:   jsr   print
39d5 9c9d44 719:   dc.b  156,157,"Defect at #",-1
39e3 208c4c 720:   jsr   prbycou
39e6 20be4d 721:   jsr   setcom
39e9 06 722:   dc.b  6          ; translate logical block a
39ea 0f 723:   dc.b  $0F        ; to cylindar, offset
39eb 000000 724: f.xlat dc.b  0,0,0
39ee 0000 725:   dc.b  0,0
39f0 20f74d 726:   jsr   hard
39f3 0000 727: f.eaddr   dc.w  0          ; ptr to error tabl
39f5 0800 728:   dc.w  8          ; return 8 bytes...
39f7 00 729:   dc.b  0
39f8 2902 730:   and   #2
39fa f022 731:   beq   f.fix2      ; if error, then must calcu
39fc 20f64c 732:   jsr   print
39ff 206572 733:   dc.b  " err in header - skipping!",$9B,-1
3a1b 4ca83a 734:   jmp   f.ok4
3a1e      735:
3a1e 20f64c 736: f.fix2 jsr   print      ; Print physical location o
3a21 202043 737:   dc.b  " C=",-1      ; ~~~~~
3a26 adf339 738:   lda   f.eaddr
3a29 85d5 739:   sta   tp
3a2b adf439 740:   lda   f.eaddr+1
3a2e 85d6 741:   sta   tp+1
3a30 a000 742:   ldy   #0

```

```

3a32 a202 743:    ldx  #2
3a34 b1d5 744: f.lop1 lda  (tp),y
3a36 9dd94c 745:    sta  bycou,x
3a39 c8    746:    iny
3a3a ca    747:    dex
3a3b 10f7 748:    bpl  f.lop1
3a3d 208c4c 749:    jsr  prbycou
3a40 20f64c 750:    jsr  print
3a43 20483d 751:    dc.b  " H=", -1
3a47 a003 752:    ldy  #3
3a49 b1d5 753:    lda  (tp),y
3a4b 8dd94c 754:    sta  bycou
3a4e a900 755:    lda  #0
3a50 8dda4c 756:    sta  bycou+1
3a53 8ddb4c 757:    sta  bycou+2
3a56 208c4c 758:    jsr  prbycou
3a59 20f64c 759:    jsr  print
3a5c 20423d 760:    dc.b  " B=", -1
3a60 a005 761:    ldy  #5
3a62 a202 762:    ldx  #2
3a64 b1d5 763: f.lop2 lda  (tp),y
3a66 9dd94c 764:    sta  bycou,x
3a69 c8    765:    iny
3a6a ca    766:    dex
3a6b 10f7 767:    bpl  f.lop2
3a6d 208c4c 768:    jsr  prbycou
3a70 20f64c 769:    jsr  print
3a73 9bff 770:    dc.b  $9B, -1
3a75      771:
3a75 adf339 772:    lda  f.eaddr
3a78 18    773:    clc
3a79 6908 774:    adc  #8
3a7b 8df339 775:    sta  f.eaddr
3a7e adf439 776:    lda  f.eaddr+1    ; advance ptr to next addre
3a81 6900 777:    adc  #0
3a83 8df439 778:    sta  f.eaddr+1
3a86 ee3b3c 779:    inc  f.terror    ; # errors this pass
3a89 ee393c 780:    inc  f.times    ; total errors in table
3a8c 101a 781:    bpl  f.ok4
3a8e 20f64c 782:    jsr  print
3a91 9b9b54 783:    dc.b  $9B,$9B,"Too many defects",$9B,-1
3aa5 4c724a 784:    jmp  abort
3aa8      785:
3aa8 aded39 786: f.ok4 lda  f.xlat+2
3aab 18    787:    clc
3aac 6901 788:    adc  #1
3aae 8d5939 789:    sta  f.fblock+3    ; set starting block to 1 b
3ab1 aded39 790:    lda  f.xlat+1    ; error block...
3ab4 6900 791:    adc  #0
3ab6 8d5839 792:    sta  f.fblock+2
3ab9 adeb39 793:    lda  f.xlat+0
3abc 6900 794:    adc  #0
3abe 8d5739 795:    sta  f.fblock+1

```



```

3ac1 4ce738 796:    jmp    f.top
3ac4      797:
3ac4      798:
3ac4 ad3b3c 799: f.done1    lda    f.terror    ; Pass done, check
3ac7 d01d 800:    bne    f.p1done    ; ~~~~~
3ac9 20f64c 801:    jsr    print
3acc 9b9b46 802:    dc.b  $9B,$9B,"Format completed",$9B,-1
3ae0 208646 803: f.good jsr    readcap
3ae3 4cdc46 804:    jmp    xformat
3ae6      805:
3ae6 20f64c 806: f.p1done    jsr    print    ; Finished pass 1 v
3ae9      807:    ; ~~~~~
3ae9 9b5665 808:    dc.b  $9B,"Verify Completed With Errors",$9B,$9B,
3b09      809:
3b09 20f64c 810:    jsr    print
3b0c 526566 811:    dc.b  "Reformatting...",-1
3b1c      812:
3b1c adf339 813:    lda    f.eaddr
3b1f 38 814:    sec
3b20 e958 815:    sbc    #low f.etable
3b22 8d573c 816:    sta    f.ndef+1    ; set length of defect list
3b25 adf439 817:    lda    f.eaddr+1
3b28 e93c 818:    sbc    #high f.etable
3b2a 8d563c 819:    sta    f.ndef
3b2d      820:
3b2d ad573c 821:    lda    f.ndef+1
3b30 18 822:    clc
3b31 6904 823:    adc    #4
3b33 8d503b 824:    sta    f.loft
3b36 ad563c 825:    lda    f.ndef
3b39 6900 826:    adc    #0
3b3b 8d513b 827:    sta    f.loft+1
3b3e      828:
3b3e 20da3b 829:    jsr    sort    ; sort the defects in the l
3b41 20be4d 830:    jsr    setcom
3b44 06 831:    dc.b  6    ; Format with complete defe
3b45 041c00 832:    dc.b  $04,$1C,0,0,2,0
3b4b 20f74d 833:    jsr    hard
3b4e 543c 834:    dc.w  f.defects
3b50 0000 835: f.loft dc.w  0    ; length of defect table
3b52 ff 836:    dc.b  -1
3b53 1003 837:    bpl    f.fmtok2
3b55 4c4e4a 838:    jmp    erformat
3b58      839:
3b58 ce3a3c 840: f.fmtok2    dec    f.tpass
3b5b f003 841:    beq    f.exit
3b5d 4c9e38 842:    jmp    f.vtop
3b60 20f64c 843: f.exit jsr    print
3b63 9b9b4e 844:    dc.b  $9B,$9B,"Not all defects were found, you"
3b84 9b6e65 845:    dc.b  $9B,"need to map the sectors out in"
3ba3 9b7468 846:    dc.b  $9B,"the VTOC after you build the"
3bc0 9b6469 847:    dc.b  $9B,"directory structure!",$9B,-1
3bd7 4ce03a 848:    jmp    f.good

```

```

3bda      849:
3bda      850:
3bda      851: ;   Sort the defects listed in the defect list
3bda      852: ;   -----
3bda a958   853: sort: lda   #low f.etable
3bdc a23c   854:     ldx   #high f.etable
3bde      855:
3bde 85d5   856: f.sortop  sta   tp
3be0 86d6   857:     stx   tp+1
3be2      858:
3be2 18     859: f.sear clc
3be3 6908   860:     adc   #8
3be5 9001   861:     bcc   f.ncx1
3be7 e8     862:     inx
3be8 cdf339 863: f.ncx1 cmp  f.eaddr
3beb d005   864:     bne   f.noten
3bed ecf439 865:     cpx   f.eaddr+1    ; if search ptr hit end, th
3bf0 f023   866:     beq   f.next1      ; tp
3bf2      867:
3bf2 85d7   868: f.noten   sta   ntp
3bf4 86d8   869:     stx   ntp+1
3bf6 202a3c 870:     jsr   compare      ; looking for an ntp<tp
3bf9 a5d7   871:     lda   ntp
3bfb a6d8   872:     ldx   ntp+1         ; advance to next if ntp>=t
3bfd b0e3   873:     bcs   f.sear       ; (compare is ntp-tp)
3bff      874:
3bff a007   875:     ldy   #7
3c01 b1d5   876: f.excha   lda   (tp),y
3c03 48     877:     pha
3c04 b1d7   878:     lda   (ntp),y      ; exchange items in list...
3c06 91d5   879:     sta   (tp),y
3c08 68     880:     pla
3c09 91d7   881:     sta   (ntp),y
3c0b 88     882:     dey
3c0c 10f3   883:     bpl   f.excha
3c0e a5d7   884:     lda   ntp
3c10 a6d8   885:     ldx   ntp+1
3c12 4ce23b 886:     jmp   f.sear       ; continue
3c15      887:
3c15 18     888: f.next1   clc
3c16 a5d5   889:     lda   tp           ; tp is now lowest, advance
3c18 a6d6   890:     ldx   tp+1
3c1a 6908   891:     adc   #8
3c1c 9001   892:     bcc   f.ncx2
3c1e e8     893:     inx
3c1f cdf339 894: f.ncx2 cmp  f.eaddr
3c22 d0ba   895:     bne   f.sortop
3c24 ecf439 896:     cpx   f.eaddr+1
3c27 d0b5   897:     bne   f.sortop
3c29 60     898:     rts              ; if tp=end, then all done
3c2a      899:
3c2a a007   900: compare   ldy   #7
3c2c 38     901:     sec

```

```

3c2d b1d7 902: f.comp2    lda  (ntp),y
3c2f f1d5 903:    sbc  (tp),y
3c31 88 904:    dey
3c32 10f9 905:    bpl  f.comp2
3c34 60 906:    rts
3c35 907:
3c35 908:
3c35 909: f.fixed    ds.b  1
3c36 910: f.blocks    ds.b  3
3c39 911: f.times    ds.b  1
3c3a 912: f.tpass    ds.b  1
3c3b 913: f.terror    ds.b  1
3c3c 914:
3c3c 915:                ; Mode Select data block fo
3c3c 000000 916: f.modsel    dc.b  0,0,0,8    ; ~~~~~
3c40 000000 917:    dc.b  0,0,0,0,0,1,0
3c48 01 918: f.fortc    dc.b  1
3c49 0000 919: f.acyln    dc.w  0
3c4b 00 920: f.ahead    dc.b  0
3c4c 02 921:    dc.b  615 /256    ;high 615
3c4d 67 922:    dc.b  615 & $ff    ;low 615    ; write cur
3c4e 02 923:    dc.b  615 /256    ;
3c4f 67 924:    dc.b  615 & $ff    ;; write precomp
3c50 00 925:    dc.b  0
3c51 02 926: f.stepra    dc.b  2
3c52 0020 927:    dc.b  $00,32
3c54 928:
3c54 929:                ; Defect list for adaptec 4
3c54 0000 930: f.defects    dc.w  0    ; ~~~~~
3c56 0000 931: f.ndef dc.w  0    ; # of defects (filled in)
3c58 932: f.etable    ds.b  8*128
4058 933:
4058 934:                ; Search data record for ad
4058 935:                ; ~~~~~
4058 000001 936: f.sdata    dc.b  0,0,1,0    ; rec size = 256
405c 000000 937:    dc.b  0,0,0,0    ; offset = 0
4060 000000 938:    dc.b  0,0,0,0    ; # records (filled in)
4064 0106 939:    dc.b  1,6    ; pattern length = 256+6
4066 000000 940:    dc.b  0,0,0,0    ; disp in record = 0
406a 0100 941:    dc.b  1,0    ; pattern length = 256
406c 942:    ds.b  256    ; pattern data
416c 943: ;
416c 944:
416c 945:
416c 946: ;
416c 947: ;=====
416c 948: ;   Format and verify Iomega Series
416c 949: ;=====
416c 950:
416c 951: ;   Format an Iomega Hard Disk
416c 952: ;   -----
416c 953: iomega
416c 20be4d 954:    jsr  setcom

```

```

416f 06 955: dc.b 6
4170 040000 956: dc.b 4,0,0,0,0,0 ; format disk command
4176 20f74d 957: jsr hard
4179 000000 958: dc.w 0,0
417d 00 959: dc.b 0
417e 1003 960: bpl f.iokhd1
4180 4c4e4a 961: jmp erformat
4183 962:
4183 20d541 963: f.iokhd1 jsr delay5
4186 20d541 964: jsr delay5
4189 965:
4189 20d541 966: f.iokhd2 jsr delay5
418c 20be4d 967: jsr setcom
418f 030000 968: dc.b 3,0,0,0,4,0 ; request sense command
4195 20f74d 969: jsr hard
4198 484e 970: dc.w senseb
419a 0400 971: dc.w 4 ; get 4 byte sense data
419c 00 972: dc.b 0
419d 1030 973: bpl f.ioksen1
419f 974:
419f 2908 975: and #8
41a1 f008 976: beq f.baer11 ; jump if unit not busy..
41a3 20f64c 977: JSR PRINT
41a6 2eff 978: dc.b " ",-1
41a8 4c8941 979: jmp f.iokhd2
41ab 980:
41ab 8a 981: f.baer11 txa
41ac 20f64c 982: jsr print
41af 9b9b53 983: dc.b $9B,$9B,"Sense Request Failed!",$9B,$9B,-1
41c9 20b149 984: jsr error
41cc 4c724a 985: jmp abort
41cf 986:
41cf 208646 987: f.ioksen1 jsr readcap ; read drive capaci
41d2 4cdc46 988: jmp xformat
41d5 989:
41d5 a514 990: delay5 lda $14
41d7 c514 991: f.del5b cmp $14
41d9 f0fc 992: beq f.del5b
41db a613 993: ldx $13
41dd e8 994: inx
41de e8 995: inx
41df e8 996: inx
41e0 e8 997: inx
41e1 e413 998: f.del6 cpx $13
41e3 d0fc 999: bne f.del6
41e5 c514 1000: cmp $14
41e7 d0f8 1001: bne f.del6
41e9 60 1002: rts
41ea 1003: ;
41ea 1004:
41ea 1005:
41ea 1006: ;
41ea 1007: ;=====

```

```

41ea 1008: ; Format XEBEC 1410/1410A
41ea 1009: ;=====
41ea 1010:
41ea 1011: ; Format XEBEC type drive
41ea 1012: ; -----
41ea 1013: xebec
41ea a900 1014: lda #0
41ec 8d4f4c 1015: sta curbad
41ef a901 1016: lda #1
41f1 203f43 1017: jsr setcfg ; set configuration for for
41f4 1018:
41f4 a900 1019: lda #0
41f6 8d0442 1020: sta f.block
41f9 8d0542 1021: sta f.block+1
41fc 8d0642 1022: sta f.block+2
41ff 1023:
41ff 20be4d 1024: f.confmt jsr setcom
4202 06 1025: dc.b 6
4203 04 1026: dc.b 4
4204 000000 1027: f.block dc.b 0,0,0,1,7 ; format command
4209 1028:
4209 20f74d 1029: jsr hard ; format...
420c 000000 1030: dc.w 0,0
4210 00 1031: dc.b 0
4211 300c 1032: bmi f.ffail
4213 2902 1033: and #2
4215 d003 1034: bne f.iser11
4217 4c7042 1035: jmp f.passlok
421a 1036:
421a ad484e 1037: f.iser11 lda senseb
421d 301c 1038: bmi f.senok1
421f 20f64c 1039: f.ffail jsr print
4222 9b9b46 1040: dc.b $9B,$9B,"Format Failed!",$9B,$9B,-1
4235 20b149 1041: jsr error
4238 4c724a 1042: jmp abort
423b 1043:
423b ae4f4c 1044: f.senok1 ldx curbad
423e ad4b4e 1045: lda senseb+3
4241 9d784c 1046: sta badt1,x
4244 18 1047: clc
4245 6920 1048: adc #32
4247 8d0642 1049: sta f.block+2
424a ad4a4e 1050: lda senseb+2
424d 9d644c 1051: sta badt2,x
4250 6900 1052: adc #0
4252 8d0542 1053: sta f.block+1
4255 ad494e 1054: lda senseb+1
4258 9d504c 1055: sta badt3,x
425b 6900 1056: adc #0
425d 8d0442 1057: sta f.block
4260 8dec4d 1058: sta command+1
4263 ee4f4c 1059: inc curbad
4266 ad4f4c 1060: lda curbad

```

```

4269 c914 1061:    cmp  #20
426b b0b2 1062:    bcs  f.fail
426d 4cff41 1063:    jmp  f.confmt
4270      1064:
4270 a900 1065: f.pass1ok  lda  #0          ; Finished first fo
4272 8d4e4c 1066:    sta  tbad          ; in bad tracks...
4275 cd4f4c 1067:    cmp  curbad
4278 f052 1068:    beq  f.pass2ok      ; if at end of bad tracks,
427a 20f64c 1069:    jsr  print
427d 9b9bff 1070:    dc.b $9B,$9B,-1
4280      1071:
4280 20fa42 1072: f.totop1  jsr  onesub
4283 20be4d 1073:    jsr  setcom
4286 06 1074:    dc.b 6
4287 07 1075:    dc.b 7
4288 000000 1076: f.btrack  dc.b 0,0,0,1,7    ; set format bad tr
428d      1077:
428d 20f74d 1078:    jsr  hard          ; format bad track...
4290 000000 1079:    dc.w 0,0
4294 00 1080:    dc.b 0
4295 1026 1081:    bpl  f.netrz
4297 20f64c 1082: f.fail3  jsr  print
429a 9b9b46 1083:    dc.b $9B,$9B,"Format BAD Track Failed!",$9B,$9B,
42b7 20b149 1084:    jsr  error
42ba 4c724a 1085:    jmp  abort
42bd      1086:
42bd 2902 1087: f.netrz  and  #2
42bf d0d6 1088:    bne  f.fail3
42c1 ee4e4c 1089:    inc  tbad
42c4 ad4e4c 1090:    lda  tbad
42c7 cd4f4c 1091:    cmp  curbad
42ca d0b4 1092:    bne  f.totop1
42cc      1093:
42cc ad284c 1094: f.pass2ok  lda  heads      ; need cyln * heads
42cf 8d794a 1095:    sta  mula
42d2 ad264c 1096:    lda  cyln
42d5 8d7a4a 1097:    sta  mulb
42d8 ad274c 1098:    lda  cyln+1
42db 8d7b4a 1099:    sta  mulb+1
42de 20804a 1100:    jsr  multiply        ; res = mula * mulb
42e1 a205 1101:    ldx  #5              ; now multiply by 32 for to
42e3 0e7d4a 1102: f.tlop4  asl  res
42e6 2e7e4a 1103:    rol  res+1
42e9 2e7f4a 1104:    rol  res+2
42ec ca 1105:    dex
42ed d0f4 1106:    bne  f.tlop4        ; now RES = total number fr
42ef      1107:
42ef 209e43 1108:    jsr  vxebec          ; now verify entire disk...
42f2      1109:          ; blocks as needed...
42f2 a90b 1110:    lda  #11
42f4 203f43 1111:    jsr  setcfg          ; reconfigure drive...
42f7 4cdc46 1112:    jmp  xformat
42fa      1113:

```

```

42fa ae4e4c 1114: onesub ldx  tbad
42fd bd504c 1115:  lda  badt3,x
4300 8ddb4c 1116:  sta  bycou+2
4303 8d8842 1117:  sta  f.btrack
4306 bd644c 1118:  lda  badt2,x
4309 8dda4c 1119:  sta  bycou+1
430c 8d8942 1120:  sta  f.btrack+1
430f bd784c 1121:  lda  badt1,x
4312 8dd94c 1122:  sta  bycou
4315 8d8a42 1123:  sta  f.btrack+2
4318 20f64c 1124:  jsr  print
431b 466f72 1125:  dc.b  "Formatting Bad Track: at #",-1
4336 208c4c 1126:  jsr  prbycou
4339 20f64c 1127:  jsr  print
433c 9bff 1128:  dc.b  $9B,-1
433e 60 1129:  rts
433f 1130: ;
433f 1131:
433f 1132:
433f 1133: ; Set drive configuration
433f 1134: ; -----
433f 1135: setcfg
433f 8d9d43 1136:  sta  f.cfgbuf+7  ; set ECC to 1 or 11
4342 ad264c 1137:  lda  cyln
4345 8d9743 1138:  sta  f.cfgbuf+1
4348 ad274c 1139:  lda  cyln+1
434b 8d9643 1140:  sta  f.cfgbuf+0
434e ad284c 1141:  lda  heads
4351 8d9843 1142:  sta  f.cfgbuf+2
4354 1143:
4354 20be4d 1144:  jsr  setcom
4357 06 1145:  dc.b  6
4358 0c0000 1146:  dc.b  $0C,0,0,0,0,0
435e a900 1147:  lda  #0
4360 8dec4d 1148:  sta  command+1  ; No LUNIT in command frame
4363 20f74d 1149:  jsr  hard
4366 9643 1150:  dc.w  f.cfgbuf
4368 0800 1151:  dc.w  8
436a ff 1152:  dc.b  -1
436b 1028 1153:  bpl  f.ok
436d 20f64c 1154:  jsr  print
4370 9b9b43 1155:  dc.b  $9B,$9B,"Could not configure drive!",$9B,$9
438f 20b149 1156:  jsr  error
4392 4c724a 1157:  jmp  abort
4395 60 1158: f.ok rts
4396 1159:
4396 000000 1160: f.cfgbuf  dc.b  0,0,0
4399 008000 1161:  dc.b  $00,$80,$00,$00,$0B
439e 1162: ;
439e 1163:
439e 1164:
439e 1165: ; Verify XEBEC format
439e 1166: ; -----

```

```

439e    1167: ; in:
439e    1168: ;   res   = number logical blocks to verify
439e    1169: vxebec
439e a901 1170:     lda   #1
43a0 8df002 1171:     sta   752
43a3 a901 1172:     lda   #1
43a5 8d4d4c 1173:     sta   count
43a8 20f64c 1174:     jsr   print
43ab 9b9b56 1175:     dc.b  $9B,$9B,"Verifying Sector #",-1
43c1    1176:
43c1 a900 1177:     lda   #0
43c3 8dd143 1178:     sta   f.rblock
43c6 8dd243 1179:     sta   f.rblock+1
43c9 8dd343 1180:     sta   f.rblock+2
43cc    1181:
43cc 20be4d 1182: f.neread  jsr   setcom
43cf 06    1183:     dc.b  6
43d0 08    1184:     dc.b  8
43d1 000000 1185: f.rblock  dc.b  0,0,0,1,7    ; read command
43d6    1186:
43d6 add143 1187:     lda   f.rblock
43d9 8ddb4c 1188:     sta   bycou+2
43dc add243 1189:     lda   f.rblock+1    ; set sector number...
43df 8dda4c 1190:     sta   bycou+1
43e2 add343 1191:     lda   f.rblock+2
43e5 8dd94c 1192:     sta   bycou
43e8 ce4d4c 1193:     dec   count
43eb d01b 1194:     bne   f.nocou
43ed a908 1195:     lda   #8
43ef 8d4d4c 1196:     sta   count
43f2 a907 1197:     lda   #7
43f4 8d0c04 1198:     sta   fieldln
43f7 208c4c 1199:     jsr   prbycou
43fa 208e4d 1200:     jsr   clend
43fd 20f64c 1201:     jsr   print
4400 1e1e1e 1202:     dc.b  30,30,30,30,30,30,30,-1
4408    1203:
4408 20f74d 1204: f.nocou   jsr   hard
440b 7044 1205:     dc.w  f.secbuf
440d 0001 1206:     dc.w  256
440f 00    1207:     dc.b  0
4410 303b 1208:     bmi   f.xokver1
4412 2902 1209:     and   #2
4414 f006 1210:     beq   f.okver1
4416 207045 1211:     jsr   remap
4419 4c2944 1212:     jmp   f.com11
441c    1213:
441c eed343 1214: f.okver1  inc   f.rblock+2
441f d008 1215:     bne   f.com11
4421 eed243 1216:     inc   f.rblock+1
4424 d003 1217:     bne   f.com11
4426 eed143 1218:     inc   f.rblock
4429    1219:

```



```

4429 add143 1220: f.com11    lda  f.rblock
442c cd7f4a 1221:      cmp  res+2
442f d09b 1222:      bne  f.neread
4431 add243 1223:      lda  f.rblock+1
4434 cd7e4a 1224:      cmp  res+1
4437 d093 1225:      bne  f.neread
4439 add343 1226:      lda  f.rblock+2
443c cd7d4a 1227:      cmp  res
443f d08b 1228:      bne  f.neread
4441 a900 1229:      lda  #0
4443 8df002 1230:      sta  752
4446 20f64c 1231:      jsr  print
4449 9b9bff 1232:      dc.b  $9B,$9B,-1
444c 60 1233:      rts          ; if at end, then all verif
444d 1234:
444d 20f64c 1235: f.xokver1 jsr  print
4450 9b9b46 1236:      dc.b  $9B,$9B,"Format verify failed!",$9B,$9B,-1
446a 20b149 1237:      jsr  error
446d 4c724a 1238:      jmp  abort
4470 1239: f.secbuf  ds.b  256
4570 1240:
4570 add143 1241: remap lda  f.rblock
4573 8ddb4c 1242:      sta  bycou+2
4576 add243 1243:      lda  f.rblock+1    ; set sector number...
4579 8dda4c 1244:      sta  bycou+1
457c add343 1245:      lda  f.rblock+2
457f 8dd94c 1246:      sta  bycou
4582 1247:
4582 208c4c 1248:      jsr  prbycou
4585 20f64c 1249:      jsr  print
4588 206973 1250:      dc.b  " is BAD",$9B,-1
4591 1251:
4591 1252:          ; Reassign track for SASI
4591 1253:          ; -----
4591 20e549 1254: f.badasg jsr  showse    ; get sense data an
4594 1255:
4594 add143 1256:      lda  f.rblock
4597 8dad45 1257:      sta  f.newlog
459a add243 1258:      lda  f.rblock+1
459d 8dae45 1259:      sta  f.newlog+1
45a0 add343 1260:      lda  f.rblock+2
45a3 29e0 1261:      and  #$E0
45a5 8daf45 1262:      sta  f.newlog+2
45a8 20be4d 1263:      jsr  setcom
45ab 06 1264:      dc.b  6
45ac 0e 1265:      dc.b  $0E
45ad 000000 1266: f.newlog  dc.b  0,0,0,1,7
45b2 1267:
45b2 ad7d4a 1268:      lda  res
45b5 38 1269:      sec
45b6 e920 1270:      sbc  #32
45b8 8d7d4a 1271:      sta  res
45bb 8dd94c 1272:      sta  bycou

```

```

45be 8d4e46 1273:    sta  newtrk+2
45c1 ad7e4a 1274:    lda  res+1
45c4 e900  1275:    sbc  #0
45c6 8d7e4a 1276:    sta  res+1
45c9 8dda4c 1277:    sta  bycou+1
45cc 8d4d46 1278:    sta  newtrk+1
45cf ad7f4a 1279:    lda  res+2
45d2 e900  1280:    sbc  #0
45d4 8d7f4a 1281:    sta  res+2
45d7 8ddb4c 1282:    sta  bycou+2
45da 8d4c46 1283:    sta  newtrk
45dd      1284:
45dd 20f64c 1285:    jsr  print
45e0 202052 1286:    dc.b " Reassigning track to #",-1
45fa 208c4c 1287:    jsr  prbycou
45fd 20f64c 1288:    jsr  print
4600 9bff  1289:    dc.b $9B,-1
4602      1290:
4602 20f74d 1291:    jsr  hard
4605 4c46  1292:    dc.w newtrk      ; send new track number as
4607 0300  1293:    dc.w 3
4609 ff    1294:    dc.b -1
460a 3029  1295:    bmi  f.badasx    ; jump if bad reassign trac
460c 2902  1296:    and  #2
460e d081  1297:    bne  f.badasg
4610      1298:
4610 add343 1299:    lda  f.rblock+2
4613 29e0  1300:    and  #$E0        ; start at beginning of tra
4615 8dd343 1301:    sta  f.rblock+2
4618      1302:
4618 20f64c 1303:    jsr  print
461b 566572 1304:    dc.b "Verifying Sector #",-1
462f a901  1305:    lda  #1
4631 8d4d4c 1306:    sta  count
4634 60    1307:    rts
4635      1308:
4635 20f64c 1309: f.badasx jsr  print
4638 436f6d 1310:    dc.b "Command failed!",$9B,-1
4649 4c724a 1311:    jmp  abort
464c      1312:
464c      1313: newtrk ds.b 3
464f      1314: ;
464f      1315:
464f      1316: ;
464f      1317: ;=====
464f      1318: ; General high level disk subroutines
464f      1319: ;=====
464f      1320:
464f      1321: ; Request sense
464f      1322: ; -----
464f      1323: getsense
464f 20be4d 1324:    jsr  setcom
4652 030000 1325:    dc.b 3,0,0,0,4,0

```

```

4658 20f74d 1326:    jsr    hard
465b 484e 1327:    dc.w    senseb
465d 0400 1328:    dc.w    4
465f 00 1329:    dc.b    0          ; get sense data
4660 1023 1330:    bpl    f.oksen1
4662 20f64c 1331:    jsr    print
4665 9b9b53 1332:    dc.b    $9B,$9B,"Sense Request Failed!",$9B,$9B,-1
467f 20b149 1333:    jsr    error
4682 4c724a 1334:    jmp    abort
4685 60 1335: f.oksen1    rts
4686      1336: ;
4686      1337:
4686      1338:
4686      1339: ;    Read drive capacity
4686      1340: ;    -----
4686      1341: ; out:
4686      1342: ;    res    = last+1 logical block on hard disk
4686      1343: readcap
4686 20be4d 1344:    jsr    setcom
4689 0a 1345:    dc.b    10
468a 250000 1346:    dc.b    $25,0,0,0,0
468f 000000 1347:    dc.b    0,0,0,0,0
4694 20f74d 1348:    jsr    hard
4697 454c 1349:    dc.w    captabl
4699 0800 1350:    dc.w    8
469b 00 1351:    dc.b    0          ; read capacity
469c 1024 1352:    bpl    f.okcap1
469e 20f64c 1353:    jsr    print
46a1 9b9b43 1354:    dc.b    $9B,$9B,"Could not read capacity",$9B,-1
46bc 20b149 1355:    jsr    error
46bf 4c724a 1356:    jmp    abort
46c2      1357:
46c2 18 1358: f.okcap1    clc
46c3 ad484c 1359:    lda    captabl+3
46c6 6901 1360:    adc    #1
46c8 8d7d4a 1361:    sta    res
46cb ad474c 1362:    lda    captabl+2
46ce 6900 1363:    adc    #0
46d0 8d7e4a 1364:    sta    res+1
46d3 ad464c 1365:    lda    captabl+1
46d6 6900 1366:    adc    #0
46d8 8d7f4a 1367:    sta    res+2
46db 60 1368:    rts
46dc      1369: ;
46dc      1370:
46dc      1371:
46dc      1372: ;
46dc      1373: ;=====
46dc      1374: ;    Exit from Hard Disk Format
46dc      1375: ;=====
46dc      1376:
46dc      1377: ;    Finish up with output to user
46dc      1378: ;    -----

```

```

46dc      1379: ; in:
46dc      1380: ;   res   = total # sectors
46dc      1381:
46dc      1382: xformat
46dc 20f64c 1383:      jsr   print
46df 9b9b46 1384:      dc.b  '$9B,$9B,"Formatted Successfully...",$9B,$9B
46fd      1385:
46fd 20764f 1386:      jsr   seram      ; set to system RAM to read
4700 205a4b 1387:      jsr   sedrive    ; search for ID/LUN drive..
4703 ac294c 1388:      ldy   opt
4706 b93c4c 1389:      lda   multit,y
4709 f003   1390:      beq   f.nomult
470b 4c3a48 1391:      jmp   f.ismulti   ; jump if multiple drives a
470e      1392:
470e 20f64c 1393: f.nomult      jsr   print
4711 456e74 1394:      dc.b  "Entire hard disk allocated to D",-1
4731 ad0103 1395:      lda   dunit
4734 0930   1396:      ora   #'0'
4736 20dc4c 1397:      jsr   prch
4739 20f64c 1398:      jsr   print
473c 3a9b9b 1399:      dc.b  ":",$9B,$9B,-1
4740      1400:
4740 ad7f4a 1401:      lda   res+2
4743 f065   1402:      beq   f.notrunc
4745 20f64c 1403:      jsr   print
4748 447269 1404:      dc.b  "Drive is larger than 16Meg... Thus a",$9B
476d 706f72 1405:      dc.b  "portion of the drive will be unused.",$9B,
4794      1406:
4794 20764f 1407:      jsr   seram      ; set to system RAM to read
4797 ae254c 1408:      ldx   device
479a a9ff   1409:      lda   #255
479c 9d15d6 1410:      sta   mio.dridata+5,x
479f 9d14d6 1411:      sta   mio.dridata+4,x
47a2 a900   1412:      lda   #0
47a4 9d13d6 1413:      sta   mio.dridata+3,x
47a7 4cc247 1414:      jmp   f.cstart
47aa      1415:
47aa 20764f 1416: f.notrunc      jsr   seram      ; set to system RAM
47ad ae254c 1417:      ldx   device
47b0 ad7d4a 1418:      lda   res
47b3 9d15d6 1419:      sta   mio.dridata+5,x
47b6 ad7e4a 1420:      lda   res+1
47b9 9d14d6 1421:      sta   mio.dridata+4,x
47bc ad7f4a 1422:      lda   res+2
47bf 9d13d6 1423:      sta   mio.dridata+3,x
47c2 a901   1424: f.cstart      lda   #1
47c4 9d12d6 1425:      sta   mio.dridata+2,x
47c7 a900   1426:      lda   #0
47c9 9d11d6 1427:      sta   mio.dridata+1,x
47cc 9d10d6 1428:      sta   mio.dridata+0,x
47cf      1429:
47cf 20f64c 1430:      jsr   print
47d2 4e6f77 1431:      dc.b  "Now you should use the Configuration",$9B

```

```

47f7 4d656e 1432:    dc.b  "Menu to save the drive parameters",$9B
4819 63616c 1433:    dc.b  "calculated by this program.",$9B,$9B,-1
4837 4c5349 1434:    jmp   f.shows
483a      1435:
483a 20f64c 1436: f.ismulti jsr   print
483d 546865 1437:    dc.b  "The Hard Drive is to be partitioned",$9B
4861 696e74 1438:    dc.b  "into several drives. You must enter",$9B
4885 746865 1439:    dc.b  "the Configuration Menu and set values",$9B
48ab 666f72 1440:    dc.b  "for Start & End Sector #s. (Refer to",$9B
48d1 746865 1441:    dc.b  "the section 'Partitioning Hard',$9B
48f0 447269 1442:    dc.b  "Drives' for more help.) After you",$9B
4913 686176 1443:    dc.b  "have set these parameters, you should",$9B
4939 736176 1444:    dc.b  "save the configuration.",$9B,$9B,-1
4953      1445:
4953 20f64c 1446: f.shows jsr   print
4956 466f72 1447:    dc.b  "For future reference, there are",$9B,-1
4977 ad7d4a 1448:    lda   res
497a 8dd94c 1449:    sta   bycou
497d ad7e4a 1450:    lda   res+1
4980 8dda4c 1451:    sta   bycou+1
4983 ad7f4a 1452:    lda   res+2
4986 8ddb4c 1453:    sta   bycou+2
4989 208c4c 1454:    jsr   prbycou      ; print byte count
498c 20f64c 1455:    jsr   print
498f 207365 1456:    dc.b  " sectors on this Hard Drive.",$9B,$9B,-1
49ae 4c724a 1457:    jmp   abort
49b1      1458: ;
49b1      1459:
49b1      1460: ;   Print out error code
49b1      1461: ;   -----
49b1      1462: error
49b1 48      1463:    pha
49b2 20f64c 1464:    jsr   print
49b5 457272 1465:    dc.b  "Error stats are: A=",-1
49c9 20284d 1466:    jsr   prby
49cc 98      1467:    tya
49cd 20f64c 1468:    jsr   print
49d0 20593d 1469:    dc.b  " Y=",-1
49d4 20104a 1470:    jsr   error_prby
49d7 20f64c 1471:    jsr   print
49da 9b9bff 1472:    dc.b  $9B,$9B,-1
49dd 68      1473:    pla
49de 2902 1474:    and   #2
49e0 f02d 1475:    beq   f.skerr1
49e2      1476:
49e2 204f46 1477:    jsr   getsense
49e5      1478:
49e5 a200 1479: showse:    ldx   #0
49e7 20f64c 1480:    jsr   print
49ea 53656e 1481:    dc.b  "Sense data is: ",-1
49fa bd484e 1482: f.tlopa    lda   senseb,x
49fd 20104a 1483:    jsr   error_prby
4a00 20f64c 1484:    jsr   print

```

```

4a03 20ff 1485:      dc.b " ",-1
4a05 e8 1486:      inx
4a06 e004 1487:      cpx #4
4a08 d0f0 1488:      bne f.tlopa
4a0a 20f64c 1489:      jsr print
4a0d 9bff 1490:      dc.b $9B,-1
4a0f 60 1491: f.skerr1  rts
4a10      1492: ;
4a10      1493:
4a10      1494: error_prby
4a10 48 1495:      pha
4a11 4a 1496:      lsr
4a12 4a 1497:      lsr
4a13 4a 1498:      lsr
4a14 4a 1499:      lsr
4a15 20194a 1500:      jsr .prnib
4a18 68 1501:      pla
4a19 290f 1502: .prnib and #15
4a1b 18 1503:      clc
4a1c 6930 1504:      adc #'0'
4a1e c93a 1505:      cmp #'.'
4a20 9002 1506:      bcc f.les1
4a22 6906 1507:      adc #6
4a24 4cdc4c 1508: f.les1 jmp prch
4a27      1509: ;
4a27      1510:
4a27      1511: ; Various error exits
4a27      1512: ; -----
4a27      1513: erconfig
4a27 20f64c 1514:      jsr print
4a2a 9b9b43 1515:      dc.b $9B,$9B,"Could not Configure Drive.", $9B,-1
4a48 20b149 1516:      jsr error
4a4b 4c724a 1517:      jmp abort
4a4e      1518: ;
4a4e      1519:
4a4e      1520: erformat
4a4e 20f64c 1521:      jsr print
4a51 9b9b43 1522:      dc.b $9B,$9B,"Could Not Format Drive.", $9B,-1
4a6c 20b149 1523:      jsr error
4a6f 4c724a 1524:      jmp abort
4a72      1525: ;
4a72      1526:
4a72      1527: ; Abort from formatter
4a72      1528: ; -----
4a72 ae444c 1529: abort ldx stack
4a75 9a 1530:      txs
4a76 4c894f 1531:      jmp xseram
4a79      1532:
4a79      1533:
4a79      1534: ;
4a79      1535: ;=====
4a79      1536: ; Misc Math and Print routines
4a79      1537: ;=====

```

```

4a79      1538:
4a79      1539: ;   perform 8 x 16 multiply
4a79      1540: ;   -----
4a79      1541: ; in:
4a79      1542: ;   mula   = 8 bit multiplier
4a79      1543: ;   mulb   = 16 bit multiplicand
4a79      1544: ; out:
4a79      1545: ;   res    = result (24 bit)
4a79      1546:
4a79 00    1547: mula dc.b  0
4a7a 000000 1548: mulb dc.b  0,0,0
4a7d 000000 1549: res  dc.b  0,0,0
4a80      1550:
4a80 a208 1551: multiply   ldx  #8
4a82 a900 1552:      lda  #0
4a84 8d7d4a 1553:      sta  res
4a87 8d7e4a 1554:      sta  res+1
4a8a 8d7f4a 1555:      sta  res+2
4a8d 8d7c4a 1556:      sta  mulb+2
4a90 4e794a 1557: tmul lsr  mula
4a93 901c 1558:      bcc  noadq
4a95 18    1559:      clc
4a96 ad7d4a 1560:      lda  res
4a99 6d7a4a 1561:      adc  mulb
4a9c 8d7d4a 1562:      sta  res
4a9f ad7e4a 1563:      lda  res+1
4aa2 6d7b4a 1564:      adc  mulb+1
4aa5 8d7e4a 1565:      sta  res+1
4aa8 ad7f4a 1566:      lda  res+2
4aab 6d7c4a 1567:      adc  mulb+2
4aae 8d7f4a 1568:      sta  res+2
4ab1 0e7a4a 1569: noadq asl  mulb
4ab4 2e7b4a 1570:      rol  mulb+1
4ab7 2e7c4a 1571:      rol  mulb+2
4aba ca    1572:      dex
4abb d0d3 1573:      bne  tmul
4abd 60    1574:      rts
4abe      1575:
4abe      1576:
4abe 20f64c 1577: yousure   jsr  print
4ac1 436175 1578:      dc.b  "Caution: This Destroys All Data!",$9B,$9B
4ae3 2d2d2d 1579:      dc.b  "-----> Are You Sure ?",-1
4afb 209f4d 1580: again jsr  inkey
4afe c959 1581:      cmp  #'Y'
4b00 f016 1582:      beq  yes
4b02 c979 1583:      cmp  #'y'
4b04 f012 1584:      beq  yes
4b06 c94e 1585:      cmp  #'N'
4b08 f004 1586:      beq  no
4b0a c96e 1587:      cmp  #'n'
4b0c d0ed 1588:      bne  again
4b0e 20f64c 1589: no   jsr  print
4b11 4e9b9b 1590:      dc.b  "N",$9B,$9B,-1

```

```

4b15 4c724a 1591:    jmp    abort
4b18      1592:
4b18 20f64c 1593: yes  jsr    print
4b1b 599b9b 1594:    dc.b  "Y",$9B,$9B,$9B,"Formatting Unit Number ",-
4b37 ad4d4e 1595:    lda    busID
4b3a 0930   1596:    ora    #'0'
4b3c 20dc4c 1597:    jsr    prch
4b3f 20f64c 1598:    jsr    print
4b42 2cff   1599:    dc.b  ",",-1
4b44 ad4e4e 1600:    lda    Unit
4b47 4a     1601:    lsr
4b48 4a     1602:    lsr
4b49 4a     1603:    lsr
4b4a 4a     1604:    lsr
4b4b 4a     1605:    lsr
4b4c 0930   1606:    ora    #'0'
4b4e 20dc4c 1607:    jsr    prch
4b51 20f64c 1608:    jsr    print
4b54 202e2e 1609:    dc.b  "...",-1
4b59 60     1610:    rts
4b5a      1611:
4b5a      1612:
4b5a      1613: ;
4b5a      1614: ;=====
4b5a      1615: ;   Misc routines used by Main entry
4b5a      1616: ;=====
4b5a      1617:
4b5a      1618: ;   Search for DUNIT that contains OPT info
4b5a      1619: ; -----
4b5a a901   1620: sedrive  lda    #1
4b5c 8d0103 1621:    sta    dunit
4b5f ac294c 1622:    ldy    opt
4b62 20184c 1623: nedri2 jsr    unitX      ; get unit index
4b65 bd16d6 1624:    lda    mio.dridata+6,x
4b68 1014   1625:    bpl    csedri
4b6a 2907   1626:    and    #7
4b6c ac294c 1627:    ldy    opt
4b6f d9344c 1628:    cmp    optIDt,y
4b72 d00a   1629:    bne    csedri
4b74 bd17d6 1630:    lda    mio.dridata+7,x
4b77 29e0   1631:    and    #$E0
4b79 d92c4c 1632:    cmp    optLUNt,y
4b7c f00c   1633:    beq    fodri      ; jump if drive found...
4b7e ee0103 1634: csedri inc    dunit
4b81 ad0103 1635:    lda    dunit
4b84 c909   1636:    cmp    #9
4b86 d0da   1637:    bne    nedri2
4b88 a9ff   1638:    lda    #-1      ; not found status
4b8a 60     1639: fodri rts
4b8b      1640:
4b8b      1641:
4b8b      1642: ;   Get physical parameters
4b8b      1643: ; -----

```



```

4b8b bd17d6 1644: geparam    lda    mio.dridata+7,x
4b8e 2910 1645:    and    #$10
4b90 8d2b4c 1646:    sta    sasif        ; sasi flag
4b93 bd80d6 1647:    lda    dritype,x
4b96 8d274c 1648:    sta    cyln+1
4b99 bd81d6 1649:    lda    dritype+1,x
4b9c 8d264c 1650:    sta    cyln
4b9f bd82d6 1651:    lda    dritype+2,x
4ba2 8d284c 1652:    sta    heads
4ba5 60 1653:    rts
4ba6 1654:
4ba6 1655:
4ba6 1656: ;    Print drive info
4ba6 1657: ;    -----
4ba6 ad2b4c 1658: prinfo lda    sasif
4ba9 f00c 1659:    beq    isnosa
4bab 20f64c 1660:    jsr    print
4bae 534153 1661:    dc.b   "SASI",-1
4bb4 4cc04b 1662:    jmp    info2
4bb7 20f64c 1663: isnosa jsr    print
4bba 534353 1664:    dc.b   "SCSI",-1
4bc0 ac294c 1665: info2 ldy    opt
4bc3 b9344c 1666:    lda    optIDt,y
4bc6 0930 1667:    ora    #'0'
4bc8 20dc4c 1668:    jsr    prch
4bcb 1669:
4bcb 20f64c 1670:    jsr    print
4bce 202020 1671:    dc.b   " ",-1
4bd2 b92c4c 1672:    lda    optLUNt,y
4bd5 4a 1673:    lsr
4bd6 4a 1674:    lsr
4bd7 4a 1675:    lsr
4bd8 4a 1676:    lsr
4bd9 4a 1677:    lsr
4bda 0930 1678:    ora    #'0'
4bdc 20dc4c 1679:    jsr    prch
4bdf 20f64c 1680:    jsr    print
4be2 2020ff 1681:    dc.b   " ",-1
4be5 1682:
4be5 ad264c 1683:    lda    cyln
4be8 8dd94c 1684:    sta    bycou
4beb ad274c 1685:    lda    cyln+1
4bee 8dda4c 1686:    sta    bycou+1
4bf1 a900 1687:    lda    #0
4bf3 8ddb4c 1688:    sta    bycou+2
4bf6 a904 1689:    lda    #4
4bf8 8d0c04 1690:    sta    fieldln
4bfb 208c4c 1691:    jsr    prbycou        ; print # cylinders
4bfe 208e4d 1692:    jsr    clend
4c01 20f64c 1693:    jsr    print
4c04 2020ff 1694:    dc.b   " ",-1
4c07 ad284c 1695:    lda    heads
4c0a 0930 1696:    ora    #'0'

```

```

4c0c 20dc4c 1697:    jsr  prch
4c0f 20f64c 1698:    jsr  print
4c12 202020 1699:    dc.b  " ",-1
4c17 60     1700:    rts
4c18         1701:
4c18         1702:
4c18         1703: ;   Get Drive index from DUNIT
4c18         1704: ;   -----
4c18         1705: unitX
4c18 ae0103 1706:    ldx   dunit
4c1b ca     1707:    dex
4c1c 8a     1708:    txa
4c1d 0a     1709:    asl
4c1e 0a     1710:    asl
4c1f 0a     1711:    asl
4c20 aa     1712:    tax
4c21 8e254c 1713:    stx   device
4c24 60     1714:    rts
4c25         1715: ;
4c25         1716:
4c25         1717: device ds.b  1
4c26         1718: cyln  ds.b  2
4c28         1719: heads ds.b  1
4c29         1720: opt   ds.b  1
4c2a         1721: curopt ds.b  1
4c2b         1722: sasif ds.b  1
4c2c         1723: optLUNt ds.b  8
4c34         1724: optIDt ds.b  8
4c3c         1725: multit ds.b  8
4c44         1726: stack ds.b  1
4c45         1727: captabl ds.b  8
4c4d         1728: count ds.b  1
4c4e         1729: tbad  ds.b  1
4c4f         1730: curbad ds.b  1
4c50         1731: badt3 ds.b  20
4c64         1732: badt2 ds.b  20
4c78         1733: badt1 ds.b  20
4c8c         1734:
4c8c         1735:
4c8c         1736: ;   Print byte count
4c8c         1737: ;   -----
4c8c         1738: prbycou:
4c8c a207   1739:    ldx   #7           ; setup for 8 digit output
4c8e 8ed04c 1740: f.ndig stx   f.place
4c91 a218   1741:    ldx   #24          ; 24 bit division
4c93 a900   1742:    lda   #0
4c95 0ed94c 1743: f.repdt   asl   bycou
4c98 2eda4c 1744:    rol   bycou+1
4c9b 2edb4c 1745:    rol   bycou+2
4c9e 2a     1746:    rol
4c9f 4ed94c 1747:    lsr   bycou
4ca2 c90a   1748:    cmp   #10
4ca4 9002   1749:    bcc   f.nx2x

```

```

4ca6 e90a 1750:    sbc  #10
4ca8 2ed94c 1751: f.nx2x rol  bycou
4cab ca 1752:    dex
4cac d0e7 1753:    bne  f.repdt
4cae 0930 1754:    ora  #'0'      ; convert remainder to digi
4cb0 aed04c 1755:    ldx  f.place
4cb3 9dd14c 1756:    sta  f.decout,x
4cb6 ca 1757:    dex
4cb7 300b 1758:    bmi  f.coutd
4cb9 add94c 1759:    lda  bycou
4cbc 0dda4c 1760:    ora  bycou+1    ; if not zero, then have mo
4cbf 0ddb4c 1761:    ora  bycou+2
4cc2 d0ca 1762:    bne  f.ndig
4cc4      1763:
4cc4 e8 1764: f.coutd    inx
4cc5 bdd14c 1765:    lda  f.decout,x
4cc8 20dc4c 1766:    jsr  prch      ; print char ASCII buffer
4ccb e007 1767:    cpx  #7
4ccd d0f5 1768:    bne  f.coutd
4ccf 60 1769: f.xcle rts
4cd0      1770:
4cd0      1771: f.place    ds.b 1
4cd1      1772: f.decout ds.b 8
4cd9      1773: bycou    ds.b 3
4cdc      1774:
4cdc      1775:
4cdc      1776: ;    Print character
4cdc      1777: ;    -----
4cdc      1778: ; in:
4cdc      1779: ;    A    = character to print
4cdc      1780: ; out:
4cdc      1781: ;    all registers preserved
4cdc      1782:
4cdc      1783: prch
4cdc 20574d 1784:    jsr  saver
4cdf 20e54c 1785:    jsr  zout
4ce2 4c874d 1786:    jmp  resall
4ce5      1787:
4ce5 8dbd4d 1788: zout sta  ztemp
4ce8 ad4703 1789:    lda  $347
4ceb 48 1790:    pha
4cec ad4603 1791:    lda  $346
4cef 48 1792:    pha
4cf0 a200 1793:    ldx  #0
4cf2 adbd4d 1794:    lda  ztemp
4cf5 60 1795:    rts
4cf6      1796:
4cf6      1797:
4cf6      1798:
4cf6      1799: ;    Print text
4cf6      1800: ;    -----
4cf6      1801: ; out:
4cf6      1802: ;    all registers preserved

```

```

4cf6      1803: ; notes:
4cf6      1804: ;   This print routine will print all characters
4cf6      1805: ;   following the JSR PRINT until a delimiter of
4cf6      1806: ;   -1 ($FF) is reached. PRINT will return to the
4cf6      1807: ;   point one byte beyond the delimiter.
4cf6      1808:
4cf6      1809: print
4cf6 20574d 1810:      jsr   saver
4cf9 ba    1811:      tsx
4cfa bd0501 1812:      lda   $105,x
4cfd 8d094d 1813:      sta   .zoch+1
4d00 bd0601 1814:      lda   $106,x
4d03 8d0a4d 1815:      sta   .zoch+2
4d06      1816:
4d06 a001 1817:      ldy   #1
4d08 b9ffff 1818: .zoch lda   $ffff,y
4d0b c9ff 1819:      cmp   #$ff
4d0d f006 1820:      beq   .zecho
4d0f 20dc4c 1821:      jsr   prch
4d12 c8    1822:      iny
4d13 d0f3 1823:      bne   .zoch
4d15 98    1824: .zecho tya
4d16 18    1825:      clc
4d17 7d0501 1826:      adc   $105,x
4d1a 9d0501 1827:      sta   $105,x
4d1d bd0601 1828:      lda   $106,x
4d20 6900 1829:      adc   #0
4d22 9d0601 1830:      sta   $106,x
4d25 4c874d 1831:      jmp   resall
4d28      1832:
4d28      1833:
4d28      1834:
4d28      1835: ;   Print byte
4d28      1836: ;   -----
4d28      1837: ; in:
4d28      1838: ;   A       = byte to print
4d28      1839: ; out:
4d28      1840: ;   all registers preserved
4d28      1841:
4d28 20574d 1842: prby jsr   saver
4d2b 20314d 1843:      jsr   zoutbyt
4d2e 4c874d 1844:      jmp   resall
4d31      1845:
4d31 48    1846: zoutbyt pha
4d32 4a    1847:      lsr
4d33 4a    1848:      lsr
4d34 4a    1849:      lsr
4d35 4a    1850:      lsr
4d36 203a4d 1851:      jsr   .zob
4d39 68    1852:      pla
4d3a 290f 1853: .zob and   #$0f
4d3c 0930 1854:      ora   #$30
4d3e c93a 1855:      cmp   #$3a

```

```

4d40 9002 1856:    bcc  .zkkkk
4d42 6906 1857:    adc  #6
4d44 20dc4c 1858: .zkkkk jsr  prch
4d47 60    1859:    rts
4d48      1860:
4d48      1861:
4d48      1862:
4d48      1863: ;    Print word
4d48      1864: ;    -----
4d48      1865: ; in:
4d48      1866: ;    XA    = word to print
4d48      1867: ; out:
4d48      1868: ;    all registers preserved
4d48      1869:
4d48 20574d 1870: prwo jsr  saver
4d4b 48    1871:    pha
4d4c 8a    1872:    txa
4d4d 20314d 1873:    jsr  zoutbyt
4d50 68    1874:    pla
4d51 20314d 1875:    jsr  zoutbyt
4d54 4c874d 1876:    jmp  resall
4d57      1877:
4d57      1878:
4d57      1879:
4d57      1880: ;    SAVE & RESTORE registers
4d57      1881: ;    -----
4d57      1882:
4d57 08    1883: saver php
4d58 48    1884:    pha
4d59 48    1885:    pha
4d5a 48    1886:    pha
4d5b 08    1887:    php
4d5c 48    1888:    pha
4d5d 8a    1889:    txa
4d5e 48    1890:    pha
4d5f ba    1891:    tsx
4d60 bd0901 1892:    lda  $109,x
4d63 9d0501 1893:    sta  $105,x
4d66 bd0701 1894:    lda  $107,x
4d69 9d0901 1895:    sta  $109,x
4d6c bd0101 1896:    lda  $101,x
4d6f 9d0701 1897:    sta  $107,x
4d72 bd0801 1898:    lda  $108,x
4d75 9d0401 1899:    sta  $104,x
4d78 bd0601 1900:    lda  $106,x
4d7b 9d0801 1901:    sta  $108,x
4d7e 98    1902:    tya
4d7f 9d0601 1903:    sta  $106,x
4d82 68    1904:    pla
4d83 aa    1905:    tax
4d84 68    1906:    pla
4d85 28    1907:    plp
4d86 60    1908:    rts

```

```

4d87      1909:
4d87 68    1910: resall pla
4d88 a8    1911:      tay
4d89 68    1912:      pla
4d8a aa    1913:      tax
4d8b 68    1914:      pla
4d8c 28    1915:      plp
4d8d 60    1916:      rts
4d8e      1917:
4d8e      1918: ;      Clear to end of field
4d8e      1919: ;      -----
4d8e      1920: clend:
4d8e ad9e4d 1921:      lda  .fieldln
4d91 300a  1922:      bmi  ..xcle
4d93 f008  1923:      beq  ..xcle
4d95 a920  1924:      lda  #' '
4d97 20dc4c 1925:      jsr  prch
4d9a 4c8e4d 1926:      jmp  clend
4d9d 60    1927: ..xcle rts
4d9e      1928:
4d9e 00    1929: .fieldln: dc.b 0      ; this may need to be removed
4d9f      1930:
4d9f      1931:
4d9f      1932:
4d9f      1933: ;      Get character from keyboard
4d9f      1934: ;      -----
4d9f 20a54d 1935: inkey jsr  inke2
4da2 a900  1936: xlda lda  #0
4da4 60    1937:      rts
4da5      1938:
4da5 20574d 1939: inke2 jsr  saver
4da8 20b44d 1940:      jsr  gekey
4dab 8da34d 1941:      sta  xlda+1
4dae 20764f 1942:      jsr  seram
4db1 4c874d 1943:      jmp  resall
4db4      1944:
4db4 ad25e4 1945: gekey lda  $E425
4db7 48    1946:      pha
4db8 ad24e4 1947:      lda  $E424
4dbb 48    1948:      pha
4dbc 60    1949:      rts
4dbd      1950:
4dbd      1951: ztemp ds.b 1
4dbe      1952:
4dbe      1953:
4dbe      1954: ;=====
4dbe      1955: ;      Main Hard Drive Calls
4dbe      1956: ;=====
4dbe      1957:
4dbe      1958: ;      Set command frame for hard drive I/O
4dbe      1959: ;      -----
4dbe      1960: setcom:
4dbe 68    1961:      pla

```

```

4dbf 85d3 1962:    sta  zp
4dc1 68 1963:    pla
4dc2 85d4 1964:    sta  zp+1
4dc4 a001 1965:    ldy  #1
4dc6 b1d3 1966:    lda  (zp),y
4dc8 8df54d 1967:    sta  comlen
4dcb aa 1968:    tax
4dcc c8 1969: f.setop    iny
4dcd b1d3 1970:    lda  (zp),y
4dcf 99e94d 1971:    sta  command-2,y
4dd2 ca 1972:    dex
4dd3 d0f7 1973:    bne  f.setop
4dd5 98 1974:    tya
4dd6 18 1975:    clc
4dd7 65d3 1976:    adc  zp
4dd9 aa 1977:    tax
4dda a900 1978:    lda  #0
4ddc 65d4 1979:    adc  zp+1
4dde 48 1980:    pha
4ddf 8a 1981:    txa
4de0 48 1982:    pha
4de1 adec4d 1983:    lda  command+1
4de4 0d4e4e 1984:    ora  unit      ; LUN
4de7 8dec4d 1985:    sta  command+1
4dea 60 1986:    rts
4deb      1987:
4deb      1988: command: ds.b 10
4df5      1989: comlen ds.b 2    ; length of command
4df7      1990: ;
4df7      1991:
4df7      1992:
4df7      1993: ; Hard drive command execute
4df7      1994: ; -----
4df7      1995: hard:
4df7 68 1996:    pla
4df8 85d3 1997:    sta  zp
4dfa 68 1998:    pla
4dfb 85d4 1999:    sta  zp+1
4dfd a001 2000:    ldy  #1
4dff b1d3 2001:    lda  (zp),y
4e01 85d5 2002:    sta  dptr
4e03 c8 2003:    iny
4e04 b1d3 2004:    lda  (zp),y
4e06 85d6 2005:    sta  dptr+1
4e08 c8 2006:    iny
4e09 b1d3 2007:    lda  (zp),y
4e0b 8d464e 2008:    sta  blockln
4e0e c8 2009:    iny
4e0f b1d3 2010:    lda  (zp),y
4e11 8d474e 2011:    sta  blockln+1
4e14 c8 2012:    iny
4e15 b1d3 2013:    lda  (zp),y
4e17 8d4c4e 2014:    sta  hdirect

```

```

4e1a 98 2015: tya
4e1b 18 2016: clc
4e1c 65d3 2017: adc zp
4e1e aa 2018: tax
4e1f a900 2019: lda #0
4e21 65d4 2020: adc zp+1
4e23 48 2021: pha
4e24 8a 2022: txa
4e25 48 2023: pha
4e26 204f4e 2024: jsr xhdio
4e29 301a 2025: bmi f.bad
4e2b 48 2026: pha
4e2c 2902 2027: and #2
4e2e f012 2028: beq f.nosts ; jump if no extended sense
4e30 20be4d 2029: jsr setcom
4e33 06 2030: dc.b 6
4e34 030000 2031: dc.b 3,0,0,0,4,0
4e3a 20f74d 2032: jsr hard
4e3d 484e 2033: dc.w senseb
4e3f 0400 2034: dc.w 4
4e41 00 2035: dc.b 0
4e42 68 2036: f.nosts pla ; A = status from m
4e43 c000 2037: cpy #0
4e45 60 2038: f.bad rts
4e46 2039:
4e46 2040: blockln ds.b 2
4e48 2041: senseb: ds.b 4
4e4c 2042: hdirect ds.b 1 ; direction...
4e4d 2043: ;
4e4d 2044:
4e4d 2045:
4e4d 2046: ;
4e4d 2047: ;=====
4e4d 2048: ; Hard Drive support functions
4e4d 2049: ;=====
4e4d 2050:
4e4d 2051: busid: ds.b 1 ; SCSI controller ID (unitID)
4e4e 2052: unit: ds.b 1 ; unit number on controller (LUN)
4e4f 2053:
4e4f 2054: ; Perform general Hard Disk I/O
4e4f 2055: ; -----
4e4f 2056: xhdio
4e4f 78 2057: SEI
4e50 20194f 2058: jsr hdready ; check if HD is ready
4e53 b02d 2059: bcs f.balog ; exit if error...
4e55 ade2d1 2060: LDA INPUTS
4e58 2904 2061: AND #HDIO ; IF INPUT TO US, THEN MUST
4e5a f01e 2062: BEQ f.comer ; DRIVE (HARDWARE BUSY)..
4e5c 2063:
4e5c 20eb4e 2064: jsr sencom ; send command frame
4e5f 2ce2d1 2065: f.wareq bit inputs ; wait for REQ-
4e62 30fb 2066: bmi f.wareq
4e64 ade2d1 2067: lda inputs ; now must start data phase

```



```

4e67 2901 2068:    and  #hdcd      ; then must have status...
4e69 f00f 2069:    beq  f.comer    ; status
4e6b 208c4e 2070:    jsr  rwdata      ; perform data phase... rea
4e6e 2ce2d1 2071: f.wareb    bit  inputs
4e71 30fb 2072:    bmi  f.wareb    ; wait for request (status
4e73 ade2d1 2073:    lda  inputs
4e76 2901 2074:    and  #hdcd      ; must be command (actually
4e78 d00b 2075:    bne  f.daer     ; jump if still thinking da
4e7a 20024f 2076: f.comer    jsr  hdstat
4e7d 18 2077:    clc
4e7e 58 2078:    CLI
4e7f a001 2079:    ldy  #1
4e81 60 2080:    rts
4e82 a08a 2081: f.balog    ldy  #$8A      ; give timeout if n
4e84 2c 2082:    dc.b $2C
4e85 a0e0 2083: f.daer ldy  #$E0      ; give interface error
4e87 58 2084:    CLI
4e88 c000 2085:    cpy  #0
4e8a 38 2086:    sec
4e8b 60 2087:    rts
4e8c      2088: ;
4e8c      2089:
4e8c      2090:
4e8c      2091: ;   Read/Write sector buffer
4e8c      2092: ;   -----
4e8c      2093: rwdata
4e8c ade2d1 2094:    lda  inputs
4e8f 4d4c4e 2095:    eor  hdirect
4e92 2904 2096:    and  #hdio
4e94 d0f6 2097:    bne  rwdata      ; wait until I/O is correct
4e96 ee474e 2098:    inc  blockln+1
4e99 2c4c4e 2099:    bit  hdirect
4e9c 1026 2100:    bpl  f.rtop     ; jump if reading data
4e9e      2101:
4e9e ce474e 2102: f.wtop dec  blockln+1
4ea1 3047 2103:    bmi  f.fini      ; jump if finished...
4ea3 d006 2104:    bne  f.fullw     ; jump if a full page left
4ea5 ae464e 2105:    ldx  blockln
4ea8 f040 2106:    beq  f.fini
4eaa 2c 2107:    dc.b $2C
4eab a200 2108: f.fullw    ldx  #0
4ead a000 2109:    ldy  #0
4eaf 2ce2d1 2110: f.wdata    bit  inputs      ; wait for request.
4eb2 30fb 2111:    bmi  f.wdata
4eb4 b1d5 2112:    lda  (dptr),y
4eb6 49ff 2113:    eor  #-1
4eb8 8de1d1 2114:    sta  daout       ; send data to HD
4ebb c8 2115:    iny
4ebc ca 2116:    dex
4ebd d0f0 2117:    bne  f.wdata
4ebf e6d6 2118:    inc  dptr+1
4ec1 4c9e4e 2119:    jmp  f.wtop
4ec4      2120:

```

```

4ec4 ce474e 2121: f.rtop dec    blockln+1
4ec7 3021  2122:    bmi  f.fini    ; jump if finished...
4ec9 d006  2123:    bne  f.fullr   ; jump if a full page left
4ecb ae464e 2124:    ldx  blockln
4ece f01a  2125:    beq  f.fini
4ed0 2c    2126:    dc.b $2C
4ed1 a200  2127: f.fullr ldx  #0
4ed3 a000  2128:    ldy  #0
4ed5 2ce2d1 2129: f.rdata bit  inputs    ; wait for request.
4ed8 30fb  2130:    bmi  f.rdata
4eda ade1d1 2131:    lda  dain      ; read data from HD
4edd 49ff  2132:    eor  #-1
4edf 91d5  2133:    sta  (dptr),y
4ee1 c8    2134:    iny
4ee2 ca    2135:    dex
4ee3 d0f0  2136:    bne  f.rdata
4ee5 e6d6  2137:    inc  dptr+1
4ee7 4cc44e 2138:    jmp  f.rtop
4eea 60    2139: f.fini rts
4eeb      2140: ;
4eeb      2141:
4eeb      2142:
4eeb      2143: ;   Send command frame to SCSI device
4eeb      2144: ;   -----
4eeb      2145: sencom
4eeb acf54d 2146:    ldy  comlen
4eee a200  2147:    ldx  #0
4ef0 bdeb4d 2148: f.neou lda  command,x
4ef3 49ff  2149:    eor  #-1
4ef5 8de1d1 2150:    sta  daout     ; set output data...
4ef8 2ce2d1 2151: ..wareq bit  inputs
4efb 30fb  2152:    bmi  ..wareq
4efd e8    2153:    inx
4efe 88    2154:    dey
4eff d0ef  2155:    bne  f.neou
4f01 60    2156:    rts
4f02      2157: ;
4f02      2158:
4f02      2159:
4f02      2160: ;   Read hard drive status
4f02      2161: ;   -----
4f02      2162: hdstat
4f02 ade1d1 2163:    lda  dain
4f05 a200  2164:    ldx  #0
4f07 2ce2d1 2165: f.replp bit  inputs
4f0a 1005  2166:    bpl  f.ldain
4f0c ca    2167:    dex
4f0d d0f8  2168:    bne  f.replp
4f0f f003  2169:    BEQ  f.LDAIN3
4f11 ace1d1 2170: f.ldain ldy  dain    ; remove second sta
4f14 49ff  2171: f.LDAIN3 eor  #-1
4f16 291f  2172:    and  #$1F      ; strip LUN bit... FIX 9/3/
4f18 60    2173:    rts

```

```

4f19    2174: ;
4f19    2175:
4f19    2176:
4f19    2177: ;   Perform selection of hard disk
4f19    2178: ;   -----
4f19    2179: hready
4f19 ade2d1 2180:   lda   inputs
4f1c 2920 2181:   and   #hdbusy   ; if not busy, then go on t
4f1e f023 2182:   beq   f.hderr   ; else jump to error
4f20    2183:
4f20 ae4d4e 2184:   ldx   busID
4f23 bd5d4f 2185:   lda   f.scsix,x   ; set unit ID...
4f26 49ff 2186:   eor   #-1
4f28 8de1d1 2187:   sta   daout       ; put on data bus...
4f2b ade2d1 2188: f.waiou   lda   inputs
4f2e 2904 2189:   and   #hdio       ; wait until assured that w
4f30 f0f9 2190:   beq   f.waiou
4f32 a930 2191:   lda   #ramen+selstro ; strobe SELECT low until n
4f34 8de2d1 2192:   sta   miscout
4f37 a264 2193:   ldx   #100        ; Must assert BUSY within 2
4f39 ade2d1 2194: f.buslp   lda   inputs
4f3c 2920 2195:   and   #hdbusy     ; wait until HD goes busy..
4f3e f00a 2196:   beq   f.conreq     ; if BUSY, then continue wi
4f40 ca 2197:   dex           ; check... else keep loopi
4f41 d0f6 2198:   bne   f.buslp
4f43 a920 2199: f.hderr   lda   #ramen
4f45 8de2d1 2200:   sta   miscout
4f48 38 2201:   sec           ; return with error...
4f49 60 2202:   rts
4f4a    2203:
4f4a a920 2204: f.conreq   lda   #ramen   ; restore MISCOUT
4f4c 8de2d1 2205:   sta   miscout
4f4f 20654f 2206:   jsr   waireq       ; wait for request
4f52 b0ef 2207:   bcs   f.hderr      ; if never happened... the
4f54 ade2d1 2208:   lda   inputs
4f57 2901 2209:   and   #hdcd        ; Must be asking for comman
4f59 d0e8 2210:   bne   f.hderr
4f5b 18 2211:   clc
4f5c 60 2212:   rts
4f5d    2213:
4f5d 010204 2214: f.scsix    dc.b $01,$02,$04,$08,$10,$20,$40,$80
4f65    2215: ;
4f65    2216:
4f65    2217:
4f65    2218: ;   Wait for request with timeout
4f65    2219: ;   -----
4f65    2220: waireq
4f65 a200 2221:   ldx   #0
4f67 2ce2d1 2222: f.waire   bit   inputs   ; jump if is reques
4f6a 1008 2223:   bpl   f.isreqw
4f6c 88 2224:   dey
4f6d d0f8 2225:   bne   f.waire       ; continue checking...
4f6f ca 2226:   dex

```

```
4f70 d0f5 2227:    bne  f.waire
4f72 38 2228:    sec
4f73 60 2229:    rts
4f74 18 2230: f.isreqw    clc
4f75 60 2231:    rts
4f76    2232: ;
4f76    2233:
4f76    2234:
4f76    2235: ; Set to system RAM
4f76    2236: ; -----
4f76 78 2237: seram: SEI
4f77 a200 2238:    ldx  #0
4f79 8ee0d1 2239:    stx  mio.RAMPAGE
4f7c a920 2240:    lda  #ramen    ; set to system RAM...
4f7e 8de2d1 2241:    sta  miscout
4f81 8dfdd6 2242:    sta  Smisc    ; shadow MISCOUT
4f84 8efcd6 2243:    stx  curpage
4f87 58 2244:    CLI
4f88 60 2245:    rts
4f89    2246:
4f89    2247: ; Exit from system ram
4f89    2248: ; -----
4f89 78 2249: xseram: SEI
4f8a adc8d6 2250:    lda  prirq
4f8d 8dfdd6 2251:    sta  Smisc    ; set shadow from PRIRQ
4f90 8de2d1 2252:    sta  miscout
4f93 58 2253:    CLI
4f94 60 2254:    rts
4f95    2255:
4f95    2256: win_end:    ds.b  0
4f95    2257:
4f95    2258:
4f95    2259: ;rest of the code
```

End assembly: no errors