


```

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: ;      title 'Mio diagnostic tool for the mio'
0000 10: ;
0000 11: ;-----
0000 12: ;      miodiag.s
0000 13: ;
0000 14: ;      ~~~~~
0000 15: ;-----
0000 16: ; Notes: o This source code MAY NOT be placed for download
0000 17: ;      o "mea" is a macro that loads the address of the
0000 18: ;      into the pointer specified by the second field.
0000 19: ;-----
0000 20: ; Assembler: MADMAC (tm) ST Cross Assembler (Atari Corp)
0000 21: ;      XASM st and IBM VERSIONS
0000 22: ;-----
0000 23:
0000 24: savmsg equ $58
0000 25: zoch equ $80
0000 26: oup equ $82
0000 27: memp equ $84
0000 28:
0000 29: zp equ $D3
0000 30: dptr equ $d5
0000 31:
0000 32: brkflg equ $11 ; break flag
0000 33:
0000 34:
0000 35: ; MIO registers
0000 36: ;-----
0000 37: rampage equ $D1E0 ; write ram page address
0000 38: dain equ $D1E1 ; HD data input
0000 39: daout equ $D1E1 ; HD & printer data out
0000 40: inputs equ $D1E2 ; monitor handshake lines
0000 41: miscout equ $D1E2 ; several misc outputs (ramdisk,
0000 42: irqstat equ $D1FF ; read IRQ status
0000 43: setrom equ $D1FF ; write ROM bank address
0000 44: rststro equ $D1E0 ; strobe reset low (goes high on
0000 45:
0000 46: ; Bit definition of INPUTS
0000 47: ;-----
0000 48: prfault equ $10 ; printer fault
0000 49: prbusy equ $40 ; printer busy
0000 50: hdcd equ $01 ; hard disk command/data
0000 51: hdmsg equ $02 ; MSG input
0000 52: hdio equ $04 ; hard disk input/output control
0000 53: hdbusy equ $20 ; hard disk busy line

```

```

0000 54: hdreq equ $80 ; hard disk request line
0000 55:
0000 56: ; Bit definitions of MISCOUT
0000 57: ; -----
0000 58: ramhi equ $0F ; ram high 4 bits of address (ma
0000 59: selstro equ $10 ; select LOW bit...
0000 60: ramen equ $20 ; ram enable bit
0000 61: prstro equ $40 ; printer strobe output
0000 62: prirqen equ $80 ; enable printer IRQ on NOT busy
0000 63:
0000 64:
0600 65: .org $600
0600 66:
0600 67: ztemp ds.b 1
0601 68: zch ds.b 1
0602 69: loop ds.b 2
0604 70: comp ds.b 2
0606 71: curbank ds.b 1
0607 72: curpage ds.b 1
0608 73: chp ds.b 1
0609 74: chp2 ds.b 1
060a 75: curlin ds.b 1
060b 76: xsave ds.b 1
060c 77: maxbank ds.b 1
060d 78: lcount ds.b 1
060e 79: keydel ds.b 1
060f 80: srtimr ds.b 1
0610 81: chl ds.b 1
0611 82: check ds.b 2
0613 83: ocheck ds.b 2
0615 84:
0615 85:
0615 86: ; HardIO variables
0615 87: ; -----
0615 88: command ds.b 10
061f 89: comlen ds.b 2 ; length of command
0621 90:
0621 91: blockln ds.b 2
0623 92: senseb ds.b 4
0627 93: hdirect ds.b 1 ; direction...
0628 94: busid ds.b 1 ; SCSI controller ID (unit
0629 95: unit ds.b 1 ; unit number on controller (
062a 96:
062a 97:
062a 98: screen: equ $9C40 ; start of screen
062a 99: ;screen: equ $7c40
062a 100:
062a 101:
062a 102: datsc equ screen+280 ; 7 lines on top of main pa
062a 103:
062a 104:
a000 105: .org $a000
a000 106:

```

```

a000 107: ; defseg romarea,start=$A000
a000 108: ; seg romarea
a000 109:
a000 a9d2 110: lda #low carini
a002 850c 111: sta 12
a004 a9a9 112: lda #high carini
a006 850d 113: sta 13
a008 4cd2a9 114: jmp carini
a00b 115:
a00b 116:
a00b 117: ; Start of menu program
a00b 118: ; -----
a00b 119: menu
a00b a2ff 120: ldx #$ff
a00d 9a 121: txs
a00e 200aa9 122: jsr clear
a011 123:
a011 20f3a8 124: jsr inidat
a014 20fca8 125: jsr add40
a017 a000 126: ldy #0
a019 20b7a8 127: jsr print
a01c 442e20 128: dc.b "D. Display Page B. Burn In",-1
a03b 20fca8 129: jsr add40
a03e 20b7a8 130: jsr print
a041 522e20 131: dc.b "R. Read Location L. Write Latch",-1
a064 20fca8 132: jsr add40
a067 20b7a8 133: jsr print
a06a 572e20 134: dc.b "W. Write Location F. Firmware Test",
a08f 20fca8 135: jsr add40
a092 20b7a8 136: jsr print
a095 482e20 137: dc.b "H. HardDisk Read P. PIO HD Read",-1
a0b8 20eea7 138: ink2 jsr dstatus
a0bb 2000a7 139: ink jsr getkey
a0be 90f8 140: bcc ink2
a0c0 f0f9 141: beq ink
a0c2 c944 142: cmp #'D'
a0c4 f022 143: beq jdis
a0c6 c952 144: cmp #'R'
a0c8 f01b 145: beq jrloc
a0ca c957 146: cmp #'W'
a0cc f01d 147: beq jwloc
a0ce c942 148: cmp #'B'
a0d0 f01c 149: beq jburn
a0d2 c94c 150: cmp #'L'
a0d4 f01b 151: beq jwlat
a0d6 c948 152: cmp #'H'
a0d8 f01a 153: beq jhardr
a0da c946 154: cmp #'F'
a0dc f019 155: beq jfirm
a0de c950 156: cmp #'P'
a0e0 f018 157: beq jpio
a0e2 4cbba0 158: jmp ink
a0e5 4c1aa6 159: jrloc jmp csload

```

```

a0e8 4cb5a6 160: jdis  jmp  dispage
a0eb 4c39a6 161: jwloc  jmp  cssave
a0ee 4c24a4 162: jburn  jmp  burnin
a0f1 4c07a6 163: jwlat  jmp  tlatch
a0f4 4c7aa1 164: jhardr jmp  hardr
a0f7 4c50a5 165: jfirm  jmp  firm
a0fa 4cfda0 166: jpio   jmp  pio
a0fd      167:
a0fd      168:
a0fd      169:
a0fd      170:
a0fd      171: ;*****
a0fd      172: ;      Hard Disk Call through PIO
a0fd      173: ;*****
a0fd      174:
a0fd      175: pio
a0fd 2065e4 176:      jsr  $e465
a100 2050e4 177:      jsr  $e450
a103 209be4 178:      jsr  $e49b
a106      179:
a106 200aa9 180:      jsr  clear
a109 2075a8 181:      jsr  disform
a10c a900   182: ..fistar lda  #0
a10e 8d0206 183:      sta  loop
a111 8d0306 184:      sta  loop+1
a114 8d0406 185:      sta  comp
a117 8d0506 186:      sta  comp+1
a11a 20eea7 187:      jsr  dstatus      ; display status screen
a11d 2035a1 188: ..again jsr  piorea    ; read page at $D600 int
a120 2048a8 189:      jsr  disp      ; display page
a123 ee0206 190:      inc  loop
a126 d003   191:      bne  ..cx
a128 ee0306 192:      inc  loop+1
a12b 20eea7 193: ..cx   jsr  dstatus
a12e 2000a7 194:      jsr  getkey
a131 90d9   195:      bcc  ..fistar
a133 b0e8   196:      bcs  ..again
a135      197:
a135      198:
a135      199:
a135      200: piorea
a135 20f0a1 201:      jsr  firmge
a138 ad0706 202:      lda  curpage
a13b 8d0a03 203:      sta  $30a
a13e ad0606 204:      lda  curbank
a141 8d0b03 205:      sta  $30b
a144 a900   206:      lda  #0
a146 8d0803 207:      sta  $308
a149 a901   208:      lda  #1
a14b 8d0903 209:      sta  $309
a14e a940   210:      lda  #$40
a150 8d0303 211:      sta  $303
a153 a952   212:      lda  #$52

```

```

a155 8d0203 213:      sta  $302
a158 a931  214:      lda  #$31
a15a 8d0003 215:      sta  $300
a15d a901  216:      lda  #1
a15f 8d0103 217:      sta  $301
a162 a900  218:      lda  #0
a164 8d0403 219:      sta  $304
a167 a907  220:      lda  #7
a169 8d0503 221:      sta  $305
a16c a940  222:      lda  #$40
a16e 8d0603 223:      sta  $306
a171 2059e4 224:      jsr  $e459
a174 1003   225:      bpl  ..ok1
a176 ee0506 226:      inc  comp+1
a179 60     227: ..ok1  rts
a17a      228:
a17a      229:
a17a      230:
a17a      231:
a17a      232:
a17a      233: ;*****
a17a      234: ;      Hard Disk Read Test
a17a      235: ;*****
a17a      236:
a17a      237: hardr
a17a a900   238:      lda  #0
a17c 8d2806 239:      sta  busID
a17f 8d2906 240:      sta  unit
a182      241:
a182      242: ;      jsr  setcom
a182      243: ;      dc.b  6,0,0,0,0,0,0
a182      244: ;      jsr  hard
a182      245: ;      dc.w  0,0
a182      246: ;      dc.b  0
a182      247:
a182 200aa9 248:      jsr  clear
a185 2075a8 249:      jsr  disform
a188 a900   250: ..fistar lda  #0
a18a 8d0206 251:      sta  loop
a18d 8d0306 252:      sta  loop+1
a190 8d0406 253:      sta  comp
a193 8d0506 254:      sta  comp+1
a196      255:
a196 20f0a1 256:      jsr  firmge
a199 a000   257:      ldy  #0
a19b b184   258: ..lop2  lda  (memp),y
a19d 990040 259:      sta  $4000,y
a1a0 c8     260:      iny
a1a1 d0f8   261:      bne  ..lop2
a1a3 2082a5 262:      jsr  showchk
a1a6      263:
a1a6 20eea7 264:      jsr  dstatus      ; display status screen
a1a9 20f0a1 265: ..again jsr  firmge

```

```

a1ac 2046a2 266:    jsr  hardpg      ; read page at $D600 into
a1af 2048a8 267:    jsr  disp      ; display page
a1b2 a000 268:    ldy  #0
a1b4 b184 269:    ..lop  lda  (memp),y
a1b6 990040 270:    sta  $4000,y
a1b9 c8 271:      iny
a1ba d0f8 272:    bne  ..lop
a1bc 2082a5 273:    jsr  showchk
a1bf ad1306 274:    lda  ocheck
a1c2 cd1106 275:    cmp  check
a1c5 d008 276:    bne  ..err
a1c7 ad1406 277:    lda  ocheck+1
a1ca cd1206 278:    cmp  check+1
a1cd f003 279:    beq  ..ok
a1cf ee0406 280:    ..err  inc  comp      ; an error
a1d2      281:
a1d2 ad1106 282:    ..ok  lda  check
a1d5 8d1306 283:    sta  ocheck
a1d8 ad1206 284:    lda  check+1
a1db 8d1406 285:    sta  ocheck+1
a1de ee0206 286:    inc  loop
a1e1 d003 287:    bne  ..cx
a1e3 ee0306 288:    inc  loop+1
a1e6 20eea7 289:    ..cx  jsr  dstatus
a1e9 2000a7 290:    jsr  getkey
a1ec 909a 291:    bcc  ..fistar
a1ee b0b9 292:    bcs  ..again
a1f0      293:
a1f0      294:
a1f0      295:  firmge
a1f0 a000 296:    ldy  #0
a1f2 ad0706 297:    lda  curpage
a1f5 2907 298:    and  #7
a1f7 09d8 299:    ora  #$D8
a1f9 8585 300:    sta  mem+1
a1fb a900 301:    lda  #0
a1fd 8584 302:    sta  mem
a1ff ae0606 303:    ldx  curbank
a202 bdf7a5 304:    lda  mymask,x
a205 8dffd1 305:    sta  $D1FF
a208 60 306:    rts
a209      307:
a209      308:
a209      309:  showsen
a209      310:    ;    lda  #low screen+40*6
a209 a930 311:    lda  #low screen+240
a20b 8582 312:    sta  oup
a20d a99d 313:    lda  #high screen+240
a20f 8583 314:    sta  oup+1
a211 a000 315:    ldy  #0
a213 20b7a8 316:    jsr  print
a216 53454e 317:    dc.b  "SENSE: ",-1
a21e ad2306 318:    lda  senseb

```

```

a221 2096a8 319:    jsr  prby
a224 20b7a8 320:    jsr  print
a227 20ff  321:    dc.b  "-,-1
a229 ad2406 322:    lda  senseb+1
a22c 2096a8 323:    jsr  prby
a22f 20b7a8 324:    jsr  print
a232 20ff  325:    dc.b  "-,-1
a234 ad2506 326:    lda  senseb+2
a237 2096a8 327:    jsr  prby
a23a 20b7a8 328:    jsr  print
a23d 20ff  329:    dc.b  "-,-1
a23f ad2606 330:    lda  senseb+3
a242 2096a8 331:    jsr  prby
a245 60  332:    rts
a246 333:
a246 334:
a246 335:
a246 336: hardpg
a246 2081a2 337:    jsr  setcom
a249 06  338:    dc.b  6
a24a 080000 339:    dc.b  8,0,0,0,1,0
a250 ad0706 340:    lda  curpage
a253 8d1806 341:    sta  command+3
a256 ad0606 342:    lda  curbank
a259 8d1706 343:    sta  command+2
a25c 344:
a25c 20aea2 345:    jsr  hard
a25f 004000 346:    dc.w  $4000,256
a263 00  347:    dc.b  0
a264 1006 348:    bpl  ..ok11
a266 ee0506 349:    inc  comp+1      ; failed totally
a269 4c73a2 350:    jmp  ..skip
a26c 351:
a26c 2902 352:    ..ok11 and  #2
a26e f003 353:    beq  ..ok12
a270 ee0406 354:    inc  comp      ; bad return status
a273 355:    ..ok12
a273 a200 356:    ..skip ldx  #0
a275 bd0040 357:    ..loop lda  $4000,x
a278 49ff 358:    eor  #$ff
a27a 9d0040 359:    sta  $4000,x
a27d e8  360:    inx
a27e d0f5 361:    bne  ..loop
a280 60  362:    rts
a281 363:
a281 364:
a281 365:
a281 366:
a281 367:
a281 368: ;=====
a281 369: ;   Hard Drive support functions
a281 370: ;=====
a281 371:

```



```

a281      372: ;      Set command frame for hard drive I/O
a281      373: ;      -----
a281      374: setcom
a281 68    375:      pla
a282 85d3  376:      sta  zp
a284 68    377:      pla
a285 85d4  378:      sta  zp+1
a287 a001  379:      ldy  #1
a289 b1d3  380:      lda  (zp),y
a28b 8d1f06 381:      sta  comlen
a28e aa    382:      tax
a28f c8    383: ..setop iny
a290 b1d3  384:      lda  (zp),y
a292 991306 385:      sta  command-2,y
a295 ca    386:      dex
a296 d0f7  387:      bne  ..setop
a298 98    388:      tya
a299 18    389:      clc
a29a 65d3  390:      adc  zp
a29c aa    391:      tax
a29d a900  392:      lda  #0
a29f 65d4  393:      adc  zp+1
a2a1 48    394:      pha
a2a2 8a    395:      txa
a2a3 48    396:      pha
a2a4 ad1606 397:      lda  command+1
a2a7 0d2906 398:      ora  unit      ; LUN
a2aa 8d1606 399:      sta  command+1
a2ad 60    400:      rts
a2ae      401:
a2ae      402:
a2ae      403:
a2ae      404: ;      Hard drive command execute
a2ae      405: ;      -----
a2ae      406: hard
a2ae 68    407:      pla
a2af 85d3  408:      sta  zp
a2b1 68    409:      pla
a2b2 85d4  410:      sta  zp+1
a2b4 a001  411:      ldy  #1
a2b6 b1d3  412:      lda  (zp),y
a2b8 85d5  413:      sta  dptr
a2ba c8    414:      iny
a2bb b1d3  415:      lda  (zp),y
a2bd 85d6  416:      sta  dptr+1
a2bf c8    417:      iny
a2c0 b1d3  418:      lda  (zp),y
a2c2 8d2106 419:      sta  blockln
a2c5 c8    420:      iny
a2c6 b1d3  421:      lda  (zp),y
a2c8 8d2206 422:      sta  blockln+1
a2cb c8    423:      iny
a2cc b1d3  424:      lda  (zp),y

```

```

a2ce 8d2706 425:    sta    hdirect
a2d1 98      426:    tya
a2d2 18      427:    clc
a2d3 65d3    428:    adc    zp
a2d5 aa      429:    tax
a2d6 a900    430:    lda    #0
a2d8 65d4    431:    adc    zp+1
a2da 48      432:    pha
a2db 8a      433:    txa
a2dc 48      434:    pha
a2dd 20fda2  435:    jsr    xhdio
a2e0 301a    436:    bmi    ..bad
a2e2 48      437:    pha
a2e3 2902    438:    and    #2
a2e5 f012    439:    beq    ..nosts    ; jump if no extended sen
a2e7 2081a2  440:    jsr    setcom
a2ea 06      441:    dc.b    6
a2eb 030000  442:    dc.b    3,0,0,0,4,0
a2f1 20aea2  443:    jsr    hard
a2f4 2306    444:    dc.w    senseb
a2f6 0400    445:    dc.w    4
a2f8 00      446:    dc.b    0
a2f9 68      447:    ..nosts pla    ; A = status from main o
a2fa c000    448:    cpy    #0
a2fc 60      449:    ..bad    rts
a2fd        450:
a2fd        451:
a2fd        452:
a2fd        453:    ;    Perform general Hard Disk I/O
a2fd        454:    ;    -----
a2fd        455:    xhdio
a2fd 78      456:    SEI
a2fe 20c7a3  457:    jsr    hdready    ; check if HD is ready
a301 b02d    458:    bcs    ..balog    ; exit if error...
a303 ade2d1  459:    LDA    INPUTS
a306 2904    460:    AND    #HDIO      ; IF INPUT TO US, THEN MU
a308 f01e    461:    BEQ    ..comer    ; DRIVE (HARDWARE BUSY).
a30a        462:
a30a 2099a3  463:    jsr    sencom     ; send command frame
a30d 2ce2d1  464:    ..wareq bit    inputs    ; wait for REQ-
a310 30fb    465:    bmi    ..wareq
a312 ade2d1  466:    lda    inputs     ; now must start data pha
a315 2901    467:    and    #hdcd      ; then must have status.
a317 f00f    468:    beq    ..comer    ; status
a319 203aa3  469:    jsr    rwdata     ; perform data phase... r
a31c 2ce2d1  470:    ..wareb bit    inputs
a31f 30fb    471:    bmi    ..wareb    ; wait for request (statu
a321 ade2d1  472:    lda    inputs
a324 2901    473:    and    #hdcd      ; must be command (actual
a326 d00b    474:    bne    ..daer     ; jump if still thinking
a328 20b0a3  475:    ..comer jsr    hdstat
a32b 18      476:    clc
a32c 58      477:    CLI

```

```

a32d a001 478:      ldy  #1
a32f 60 479:      rts
a330 a08a 480: ..balog ldy  #$8A      ; give timeout if not RE
a332 2c 481:      dc.b $2C
a333 a0e0 482: ..daer ldy  #$E0      ; give interface error
a335 58 483:      CLI
a336 c000 484:      cpy  #0
a338 38 485:      sec
a339 60 486:      rts
a33a      487:
a33a      488:
a33a      489:
a33a      490: ;      Read/Write sector buffer
a33a      491: ;      -----
a33a      492: rwdata
a33a ade2d1 493:      lda  inputs
a33d 4d2706 494:      eor  hdirect
a340 2904 495:      and  #hdio
a342 d0f6 496:      bne  rwdata      ; wait until I/O is corre
a344 ee2206 497:      inc  blockln+1
a347 2c2706 498:      bit  hdirect
a34a 1026 499:      bpl  ..rtop      ; jump if reading data
a34c      500:
a34c ce2206 501: ..wtop dec  blockln+1
a34f 3047 502:      bmi  ..fini      ; jump if finished...
a351 d006 503:      bne  ..fullw      ; jump if a full page lef
a353 ae2106 504:      ldx  blockln
a356 f040 505:      beq  ..fini
a358 2c 506:      dc.b $2C
a359 a200 507: ..fullw ldx  #0
a35b a000 508:      ldy  #0
a35d 2ce2d1 509: ..wdata bit  inputs      ; wait for request...
a360 30fb 510:      bmi  ..wdata
a362 b1d5 511:      lda  (dptr),y
a364 49ff 512:      eor  #-1
a366 8de1d1 513:      sta  daout      ; send data to HD
a369 c8 514:      iny
a36a ca 515:      dex
a36b d0f0 516:      bne  ..wdata
a36d e6d6 517:      inc  dptr+1
a36f 4c4ca3 518:      jmp  ..wtop
a372      519:
a372 ce2206 520: ..rtop dec  blockln+1
a375 3021 521:      bmi  ..fini      ; jump if finished...
a377 d006 522:      bne  ..fullr      ; jump if a full page lef
a379 ae2106 523:      ldx  blockln
a37c f01a 524:      beq  ..fini
a37e 2c 525:      dc.b $2C
a37f a200 526: ..fullr ldx  #0
a381 a000 527:      ldy  #0
a383 2ce2d1 528: ..rdata bit  inputs      ; wait for request...
a386 30fb 529:      bmi  ..rdata
a388 ade1d1 530:      lda  dain      ; read data from HD

```

```

a38b 49ff 531:    eor    #-1
a38d 91d5 532:    sta    (dptr),y
a38f c8 533:    iny
a390 ca 534:    dex
a391 d0f0 535:    bne    ..rdata
a393 e6d6 536:    inc    dptr+1
a395 4c72a3 537:    jmp    ..rtop
a398 60 538: ..fini rts
a399      539:
a399      540:
a399      541:
a399      542: ;    Send command frame to SCSI device
a399      543: ;    -----
a399      544: sencom
a399 ac1f06 545:    ldy    comlen
a39c a200 546:    ldx    #0
a39e bd1506 547: ..neou lda    command,x
a3a1 49ff 548:    eor    #-1
a3a3 8de1d1 549:    sta    daout    ; set output data...
a3a6 2ce2d1 550: ..wareq bit    inputs
a3a9 30fb 551:    bmi    ..wareq
a3ab e8 552:    inx
a3ac 88 553:    dey
a3ad d0ef 554:    bne    ..neou
a3af 60 555:    rts
a3b0      556:
a3b0      557:
a3b0      558:
a3b0      559: ;    Read hard drive status
a3b0      560: ;    -----
a3b0      561: hdstat
a3b0 ade1d1 562:    lda    dain
a3b3 a200 563:    ldx    #0
a3b5 2ce2d1 564: ..replp bit    inputs
a3b8 1005 565:    bpl    ..ldain
a3ba ca 566:    dex
a3bb d0f8 567:    bne    ..replp
a3bd f003 568:    BEQ    ..LDAIN3
a3bf ace1d1 569: ..ldain ldy    dain    ; remove second status b
a3c2 49ff 570: ..LDAIN3 eor    #-1
a3c4 291f 571:    and    #$1F    ; strip LUN bit... FIX 9/
a3c6 60 572:    rts
a3c7      573:
a3c7      574:
a3c7      575:
a3c7      576: ;    Perform selection of hard disk
a3c7      577: ;    -----
a3c7      578: hdready
a3c7 ade2d1 579:    lda    inputs
a3ca 2920 580:    and    #hdbusy    ; if not busy, then go on
a3cc f023 581:    beq    ..hderr    ; else jump to error
a3ce      582:
a3ce ae2806 583:    ldx    busID

```

```

a3d1 bd0ba4 584:    lda    ..scsix,x    ; set unit ID...
a3d4 49ff 585:    eor    #-1
a3d6 8de1d1 586:    sta    daout    ; put on data bus...
a3d9 ade2d1 587:    ..waiou lda    inputs
a3dc 2904 588:    and    #hdio    ; wait until assured that
a3de f0f9 589:    beq    ..waiou
a3e0 a930 590:    lda    #ramen+selstro ; strobe SELECT low until
a3e2 8de2d1 591:    sta    miscout
a3e5 a264 592:    ldx    #100    ; Must assert BUSY within
a3e7 ade2d1 593:    ..buslp lda    inputs
a3ea 2920 594:    and    #hdbusy    ; wait until HD goes busy
a3ec f00a 595:    beq    ..conreq    ; if BUSY, then continue
a3ee ca 596:    dex    ; check... else keep loo
a3ef d0f6 597:    bne    ..buslp
a3f1 a920 598:    ..hderr lda    #ramen
a3f3 8de2d1 599:    sta    miscout
a3f6 38 600:    sec    ; return with error...
a3f7 60 601:    rts
a3f8 602:
a3f8 a920 603:    ..conreq lda    #ramen    ; restore MISCOUT
a3fa 8de2d1 604:    sta    miscout
a3fd 2013a4 605:    jsr    waireq    ; wait for request
a400 b0ef 606:    bcs    ..hderr    ; if never happened... t
a402 ade2d1 607:    lda    inputs
a405 2901 608:    and    #hdcd    ; Must be asking for comm
a407 d0e8 609:    bne    ..hderr
a409 18 610:    clc
a40a 60 611:    rts
a40b 612:
a40b 010204 613:    ..scsix dc.b $01,$02,$04,$08,$10,$20,$40,$80
a413 614:
a413 615:
a413 616:
a413 617: ;    Wait for request with timeout
a413 618: ;    -----
a413 619: waireq
a413 a200 620:    ldx    #0
a415 2ce2d1 621:    ..waire bit    inputs    ; jump if is request
a418 1008 622:    bpl    ..isreqw
a41a 88 623:    dey
a41b d0f8 624:    bne    ..waire    ; continue checking...
a41d ca 625:    dex
a41e d0f5 626:    bne    ..waire
a420 38 627:    sec
a421 60 628:    rts
a422 18 629:    ..isreqw clc
a423 60 630:    rts
a424 631:
a424 632:
a424 633:
a424 634:
a424 635: ; *****
a424 636: ;    Burn in test

```

```

a424      637: ,*****
a424      638:
a424      639: burnin
a424 200aa9 640:      jsr  clear
a427 20eea7 641:      jsr  dstatus
a42a      642:
a42a 20f3a8 643:      jsr  inidat
a42d 20fca8 644:      jsr  add40
a430 20b7a8 645:      jsr  print
a433 205369 646:      dc.b  " Size: ",-1
a43b      647:
a43b a924   648:      lda  #$24
a43d 8de2d1 649:      sta  $D1E2
a440 a9aa   650:      lda  #$AA
a442 8d00d6 651:      sta  $D600
a445 4e00d6 652:      lsr  $D600
a448 ad00d6 653:      lda  $D600
a44b c955   654:      cmp  #$55
a44d f00c   655:      beq  ismeg
a44f 20b7a8 656:      jsr  print
a452 323536 657:      dc.b  "256K",-1
a457 a904   658:      lda  #4
a459 d00b   659:      bne  burn2
a45b 20b7a8 660: ismeg jsr  print
a45e 31204d 661:      dc.b  "1 Meg",-1
a464 a910   662:      lda  #16
a466 8d0c06 663: burn2 sta  maxbank
a469      664:
a469 20fca8 665:      jsr  add40
a46c 20b7a8 666:      jsr  print
a46f 537461 667:      dc.b  "Status: ",-1 ; Y=8 at screen+40*8
a478      668:
a478 209ca6 669:      jsr  clearm
a47b      670:
a47b a900   671:      lda  #0
a47d 8d0206 672:      sta  loop
a480 8d0306 673:      sta  loop+1
a483 8d0406 674:      sta  comp
a486 8d0506 675:      sta  comp+1
a489 20eea7 676:      jsr  dstatus
a48c      677:
a48c a910   678: vtop  lda  #16
a48e 8d0d06 679:      sta  lcount
a491 ae0c06 680:      ldx  maxbank
a494 ca     681:      dex
a495 8a     682:      txa
a496 2d0ad2 683:      and  $d20a
a499 8d0606 684:      sta  curbank
a49c ad0ad2 685:      lda  $d20a
a49f 8d0706 686:      sta  curpage
a4a2 ee0706 687: vtop  inc  curpage
a4a5 207da6 688:      jsr  setram      ; set rambank
a4a8 208ca6 689:      jsr  setpg

```

```

a4ab b184 690:    lda  (memp),y
a4ad f015 691:    beq  mwrit      ; if 0, then must write
a4af 2c0ad2 692:    bit  $d20a
a4b2 3010 693:    bmi  mwrit      ; if rnd neg, then write
a4b4 202ba5 694:    jsr  verify
a4b7 f00e 695:    beq  nlop
a4b9 ee0406 696:    inc  comp
a4bc d009 697:    bne  nlop
a4be ee0506 698:    inc  comp+1
a4c1 4cc7a4 699:    jmp  nlop
a4c4 20fea4 700: mwrit jsr  write
a4c7 ce0d06 701: nlop  dec  lcount
a4ca d0d6 702:    bne  vtop
a4cc ee0206 703:    inc  loop
a4cf d003 704:    bne  ok22
a4d1 ee0306 705:    inc  loop+1
a4d4 20eea7 706: ok22 jsr  dstatus
a4d7 adfc02 707:    lda  $2fc
a4da c9ff 708:    cmp  #-1
a4dc f0ae 709:    beq  vtop
a4de a9ff 710:    lda  #-1
a4e0 8dfc02 711:    sta  $2fc
a4e3 4c0ba0 712:    jmp  menu
a4e6      713:
a4e6      714:
a4e6      715:
a4e6      716: rslw
a4e6 a980 717:    lda  #low screen+320
a4e8 8582 718:    sta  oup
a4ea a99d 719:    lda  #high screen+320
a4ec 8583 720:    sta  oup+1
a4ee a008 721:    ldy  #8
a4f0 9006 722:    bcc  isr
a4f2 20b7a8 723:    jsr  print
a4f5 57ff 724:    dc.b  "W",-1
a4f7 60 725:    rts
a4f8 20b7a8 726: isr  jsr  print
a4fb 56ff 727:    dc.b  "V",-1
a4fd 60 728:    rts
a4fe      729:
a4fe      730:
a4fe      731:
a4fe      732: write
a4fe ad0ad2 733:    lda  $d20a
a501 2903 734:    and  #3
a503 18 735:    clc
a504 6901 736:    adc  #1
a506 9184 737:    sta  (memp),y      ; set new type...
a508 48 738:    pha
a509 38 739:    sec
a50a 20e6a4 740:    jsr  rslw
a50d 68 741:    pla
a50e c902 742:    cmp  #2

```

```

a510 9006 743:      bcc  w1
a512 f004 744:      beq  w2
a514 c904 745:      cmp  #4
a516 9000 746:      bcc  w3
a518      747: w1
a518      748: w2
a518      749: w3
a518 ad0606 750: w4      lda  curbank
a51b 18 751:      clc
a51c 6d0706 752:      adc  curpage
a51f a000 753:      ldy  #0
a521 9900d6 754: w4lo    sta  $d600,y
a524 18 755:      clc
a525 6901 756:      adc  #1
a527 c8 757:      iny
a528 d0f7 758:      bne  w4lo
a52a 60 759:      rts
a52b      760:
a52b      761:
a52b      762:
a52b      763: verify
a52b 48 764:      pha
a52c 18 765:      clc
a52d 20e6a4 766:      jsr  rslw
a530 68 767:      pla
a531 c902 768:      cmp  #2
a533 9006 769:      bcc  r1
a535 f004 770:      beq  r2
a537 c904 771:      cmp  #4
a539 9000 772:      bcc  r3
a53b      773: r1
a53b      774: r2
a53b      775: r3
a53b ad0606 776: r4      lda  curbank
a53e 18 777:      clc
a53f 6d0706 778:      adc  curpage
a542 a000 779:      ldy  #0
a544 d900d6 780: r4lo    cmp  $d600,y
a547 d006 781:      bne  nok4
a549 18 782:      clc
a54a 6901 783:      adc  #1
a54c c8 784:      iny
a54d d0f5 785:      bne  r4lo
a54f 60 786: nok4    rts
a550      787:
a550      788:
a550      789:
a550      790:
a550      791: ;*****
a550      792: ;      Test firmware
a550      793: ;*****
a550      794:
a550      795: firm

```



```

a550 200aa9 796:      jsr  clear
a553 2075a8 797:      jsr  disform
a556 a900   798: ..top  lda   #0
a558 8d0206 799:      sta  loop
a55b 8d0306 800:      sta  loop+1
a55e 8d0406 801:      sta  comp
a561 8d0506 802:      sta  comp+1
a564 20eea7 803:      jsr  dstatus      ; display status screen
a567 20c1a5 804: ..again jsr  firmpg      ; read page at $D600 int
a56a 2048a8 805:      jsr  dispg      ; display page
a56d ee0206 806:      inc  loop
a570 d003   807:      bne  ..cx
a572 ee0306 808:      inc  loop+1
a575 20eea7 809: ..cx   jsr  dstatus
a578 2082a5 810:      jsr  showchk
a57b 2000a7 811:      jsr  getkey
a57e 90d6   812:      bcc  ..top
a580 b0e5   813:      bcs  ..again
a582      814:
a582      815:
a582      816:
a582      817: showchk
a582 a930   818:      lda  #low screen+240
a584 8582   819:      sta  oup
a586 a99d   820:      lda  #high screen+240
a588 8583   821:      sta  oup+1
a58a a000   822:      ldy  #0
a58c 20b7a8 823:      jsr  print
a58f 434845 824:      dc.b  "CHECKSUM: ",-1
a59a a200   825:      ldx  #0
a59c 8e1106 826:      stx  check
a59f 8e1206 827:      stx  check+1
a5a2 bd0040 828: ..loop  lda  $4000,x
a5a5 18     829:      clc
a5a6 6d1106 830:      adc  check
a5a9 8d1106 831:      sta  check
a5ac 9003   832:      bcc  ..nc
a5ae ee1206 833:      inc  check+1
a5b1 e8     834: ..nc   inx
a5b2 d0ee   835:      bne  ..loop
a5b4 ad1206 836:      lda  check+1
a5b7 2096a8 837:      jsr  prby
a5ba ad1106 838:      lda  check
a5bd 2096a8 839:      jsr  prby
a5c0 60     840:      rts
a5c1      841:
a5c1      842:
a5c1      843:
a5c1      844: firmpg
a5c1 a000   845:      ldy  #0
a5c3 ad0706 846:      lda  curpage
a5c6 2907   847:      and  #7
a5c8 09d8   848:      ora  #$D8

```

```

a5ca 8585 849:      sta  memp+1
a5cc a900 850:      lda  #0
a5ce 8584 851:      sta  memp
a5d0 ae0606 852:     ldx  curbank
a5d3      853:
a5d3 bdf7a5 854: firtop lda  mymask,x
a5d6 8dffdl 855:      sta  $D1FF
a5d9 b184 856:      lda  (memp),y
a5db 990040 857:      sta  $4000,y
a5de bdf7a5 858:      lda  mymask,x
a5e1 8dffdl 859:      sta  $D1FF
a5e4 b184 860:      lda  (memp),y
a5e6 d90040 861:      cmp  $4000,y
a5e9 f008 862:      beq  ok123
a5eb ee0406 863:      inc  comp
a5ee d003 864:      bne  ok123
a5f0 ee0506 865:      inc  comp+1
a5f3 c8 866: ok123 iny
a5f4 d0dd 867:      bne  firtop
a5f6 60 868:      rts
a5f7      869:
a5f7      870:
a5f7 000102 871: mymask dc.b  $00,$01,$02,$04,$08,$10,$20,$40,$80
a600 000000 872:      dc.b  $00,$00,$00,$00,$00,$00,$00,$00
a607      873:
a607      874:
a607      875:
a607 876: ;*****
a607 877: ;    Latch test
a607 878: ;*****
a607 879:
a607 880: tlatch
a607 a200 881:      ldx  #0
a609 207da6 882: tlat2 jsr  setram
a60c ca 883:      dex
a60d d0fa 884:      bne  tlat2
a60f 2000a7 885:      jsr  getkey
a612 b0f3 886:      bcs  tlatch
a614 20eea7 887:      jsr  dstatus
a617 4c07a6 888:      jmp  tlatch
a61a      889:
a61a      890:
a61a      891:
a61a 892: ;*****
a61a 893: ;    Continous single location load/save
a61a 894: ;*****
a61a 895:
a61a 896: csload
a61a 2058a6 897:      jsr  getloc
a61d 8e0b06 898:      stx  xsave
a620 ae0b06 899: lpp1 ldx  xsave
a623 a000 900:      ldy  #0
a625 bd00d6 901: lpp2 lda  $D600,x

```

```

a628 88 902:    dey
a629 d0fa 903:    bne  llpp2
a62b 2000a7 904:    jsr  getkey
a62e b0f0 905:    bcs  llpp1
a630 207da6 906:    jsr  setram
a633 20eea7 907:    jsr  dstatus
a636 4c20a6 908:    jmp  llpp1
a639      909:
a639      910:
a639      911:
a639      912: ;    Continous single location save
a639      913: ;    -----
a639      914: cssave
a639 2058a6 915:    jsr  getloc
a63c 8e0b06 916:    stx  xsave
a63f ae0b06 917: llpp4  ldx  xsave
a642 a000 918:    ldy  #0
a644 9d00d6 919: llpp3  sta  $D600,x
a647 88 920:    dey
a648 d0fa 921:    bne  llpp3
a64a 2000a7 922:    jsr  getkey
a64d b0f0 923:    bcs  llpp4
a64f 207da6 924:    jsr  setram
a652 20eea7 925:    jsr  dstatus
a655 4c3fa6 926:    jmp  llpp4
a658      927:
a658      928:
a658      929: getloc
a658 a930 930:    lda  #low screen+240    ;120
a65a 8582 931:    sta  oup
a65c a99d 932:    lda  #high screen+240    ;120
a65e 8583 933:    sta  oup+1
a660 a000 934:    ldy  #0
a662 20b7a8 935:    jsr  print
a665 4c6f63 936:    dc.b  "Location?",-1
a66f 20a6a7 937:    jsr  inbyte
a672 48 938:    pha
a673 a008 939:    ldy  #8
a675 a93a 940:    lda  #':'
a677 20a9a8 941:    jsr  prch
a67a 68 942:    pla
a67b aa 943:    tax
a67c 60 944:    rts
a67d      945:
a67d      946:
a67d      947:
a67d      948:
a67d      949: ; *****
a67d      950: ;    Memory sub-routines
a67d      951: ; *****
a67d      952:
a67d      953: setram
a67d ad0606 954:    lda  curbank

```

```

a680 0920 955:    ora  #$20
a682 8de2d1 956:    sta  $D1E2
a685 ad0706 957:    lda  curpage
a688 8de0d1 958:    sta  $D1E0
a68b 60 959:    rts
a68c 960:
a68c 961:
a68c 962:
a68c 963: ;    Set memory page
a68c 964: ;    -----
a68c 965: setpg
a68c a900 966:    lda  #0
a68e 8584 967:    sta  memp
a690 a910 968:    lda  #$10
a692 18 969:    clc
a693 6d0606 970:    adc  curbank
a696 8585 971:    sta  memp+1
a698 ac0706 972:    ldy  curpage
a69b 60 973:    rts
a69c 974:
a69c 975:
a69c 976:
a69c 977: ;    clear memory map
a69c 978: ;    -----
a69c 979: clearm
a69c a900 980:    lda  #0
a69e 8584 981:    sta  memp
a6a0 a910 982:    lda  #$10
a6a2 8585 983:    sta  memp+1
a6a4 a000 984: lop4 ldy  #0
a6a6 98 985:    tya
a6a7 9184 986: lop3 sta  (memp),y
a6a9 c8 987:    iny
a6aa d0fb 988:    bne  lop3
a6ac e685 989:    inc  memp+1
a6ae a585 990:    lda  memp+1
a6b0 c920 991:    cmp  #$20
a6b2 d0f0 992:    bne  lop4
a6b4 60 993:    rts
a6b5 994:
a6b5 995:
a6b5 996:
a6b5 997:
a6b5 998:
a6b5 999: ; *****
a6b5 1000: ;    Continuous display of RAM page
a6b5 1001: ; *****
a6b5 1002:
a6b5 1003: dispage
a6b5 200aa9 1004:    jsr  clear    ; clear screen
a6b8 2075a8 1005:    jsr  disform  ; display form page
a6bb 207da6 1006: ..start jsr  setram
a6be a900 1007:    lda  #0

```

```

a6c0 8d0206 1008:      sta  loop
a6c3 8d0306 1009:      sta  loop+1
a6c6 8d0406 1010:      sta  comp
a6c9 8d0506 1011:      sta  comp+1
a6cc 20eea7 1012:      jsr  dstatus      ; display status screen
a6cf 20e7a6 1013: ..again jsr  readpg      ; read page at $D600 int
a6d2 2048a8 1014:      jsr  disp      ; display page
a6d5 ee0206 1015:      inc  loop
a6d8 d003 1016:      bne  ..cx
a6da ee0306 1017:      inc  loop+1
a6dd 20eea7 1018: ..cx   jsr  dstatus
a6e0 2000a7 1019:      jsr  getkey
a6e3 90d6 1020:      bcc  ..start
a6e5 b0e8 1021:      bcs  ..again
a6e7 1022:
a6e7 1023:
a6e7 1024:
a6e7 1025: ;      Read memory page
a6e7 1026: ;      -----
a6e7 1027: readpg
a6e7 a200 1028:      ldx  #0
a6e9 bd00d6 1029: ..top  lda  $D600,x
a6ec 9d0040 1030:      sta  $4000,x
a6ef dd00d6 1031:      cmp  $D600,x
a6f2 f008 1032:      beq  ..cx
a6f4 ee0406 1033:      inc  comp
a6f7 d003 1034:      bne  ..cx
a6f9 ee0506 1035:      inc  comp+1
a6fc e8 1036: ..cx   inx
a6fd d0ea 1037:      bne  ..top
a6ff 60 1038:      rts
a700 1039:
a700 1040:
a700 1041:
a700 1042:
a700 1043: ;*****
a700 1044: ;      Input routines
a700 1045: ;*****
a700 1046:
a700 1047: ;      Get key and execute
a700 1048: ;      -----
a700 1049: ; out:
a700 1050: ;      c=0 if changed BANK/PAGE
a700 1051: ;      c=1,Z=1 if no key
a700 1052: ;      c=1,Z=0 if key (nothing done)
a700 1053:
a700 1054: getkey
a700 adfc02 1055:      lda  $2FC
a703 c9ff 1056:      cmp  #-1
a705 f027 1057:      beq  noke
a707 293f 1058:      and  #$3F
a709 aa 1059:      tax
a70a bd66a7 1060:      lda  chtabl,x      ; get char from table

```

```

a70d a0ff 1061:    ldy  #$FF
a70f 8cfc02 1062:    sty  $2FC
a712 c92d 1063:    cmp  #'-'
a714 f024 1064:    beq  upa
a716 c93d 1065:    cmp  #'='
a718 f026 1066:    beq  downa
a71a c91b 1067:    cmp  #27
a71c f045 1068:    beq  esc1
a71e c92a 1069:    cmp  #'*'
a720 f013 1070:    beq  righta
a722 c92b 1071:    cmp  #'+'
a724 f00a 1072:    beq  lefta
a726 c93c 1073:    cmp  #'<'
a728 f023 1074:    beq  left8
a72a c93e 1075:    cmp  #'>'
a72c f02a 1076:    beq  right8
a72e 38 1077: noke  sec
a72f 60 1078:    rts
a730 1079:
a730 ce0706 1080: lefta  dec  curpage
a733 18 1081:    clc
a734 60 1082:    rts
a735 ee0706 1083: righta  inc  curpage
a738 18 1084:    clc
a739 60 1085:    rts
a73a ee0606 1086: upa  inc  curbank
a73d 4c43a7 1087:    jmp  o11
a740 ce0606 1088: downa  dec  curbank
a743 ad0606 1089: o11  lda  curbank
a746 290f 1090:    and  #15
a748 8d0606 1091:    sta  curbank
a74b 18 1092:    clc
a74c 60 1093:    rts
a74d ad0706 1094: left8  lda  curpage
a750 38 1095:    sec
a751 e908 1096:    sbc  #8
a753 8d0706 1097:    sta  curpage
a756 18 1098:    clc
a757 60 1099:    rts
a758 ad0706 1100: right8  lda  curpage
a75b 18 1101:    clc
a75c 6908 1102:    adc  #8
a75e 8d0706 1103:    sta  curpage
a761 18 1104:    clc
a762 60 1105:    rts
a763 4c0ba0 1106: esc1  jmp  menu
a766 1107:
a766 1108:
a766 4c4a3b 1109: chtabl  dc.b  "LJ;",0,0,"K+*"
a76e 4f0050 1110:    dc.b  "O",0,"PU",13,"I="
a776 560043 1111:    dc.b  "V",0,"C",0,0,"BXZ"
a77e 340033 1112:    dc.b  "4",0,"36",$1B,"521"
a786 2c202e 1113:    dc.b  ",.N",0,"M/", $81

```

```

a78e 520045 1114:    dc.b  "R",0,"EY",9,"TWQ"
a796 390030 1115:    dc.b  "9",0,"07",8,"8<"
a79e 464844 1116:    dc.b  "FHD",0,$82,"GSA"
a7a6      1117:
a7a6      1118: inbyte
a7a6 20c3a7 1119:    jsr  inbin
a7a9 8d0806 1120:    sta  chp
a7ac 209fa8 1121:    jsr  prnib
a7af 20c3a7 1122:    jsr  inbin
a7b2 8d0906 1123:    sta  chp2
a7b5 209fa8 1124:    jsr  prnib
a7b8 ad0806 1125:    lda  chp
a7bb 0a     1126:    asl
a7bc 0a     1127:    asl
a7bd 0a     1128:    asl
a7be 0a     1129:    asl
a7bf 0d0906 1130:    ora  chp2
a7c2 60     1131:    rts
a7c3      1132:
a7c3      1133:
a7c3      1134: inbin
a7c3 20dba7 1135:    jsr  inchar
a7c6 c930   1136:    cmp  #'0'
a7c8 90f9   1137:    bcc  inbin
a7ca c93a   1138:    cmp  #'.'
a7cc 900a   1139:    bcc  num
a7ce c947   1140:    cmp  #'G'
a7d0 b0f1   1141:    bcs  inbin
a7d2 c941   1142:    cmp  #'A'
a7d4 90ed   1143:    bcc  inbin
a7d6 e907   1144:    sbc  #7
a7d8 290f   1145: num    and    #15
a7da 60     1146:    rts
a7db      1147:
a7db      1148:
a7db      1149:
a7db      1150: inchar
a7db adfc02 1151:    lda  $2FC
a7de c9ff   1152:    cmp  #-1
a7e0 f0f9   1153:    beq  inchar
a7e2 293f   1154:    and  #$3F
a7e4 aa     1155:    tax
a7e5 bd66a7 1156:    lda  chtabl,x    ; get char from table
a7e8 a2ff   1157:    ldx  #$FF
a7ea 8efc02 1158:    stx  $2FC
a7ed 60     1159:    rts
a7ee      1160:
a7ee      1161:
a7ee      1162:
a7ee      1163:
a7ee      1164: ;*****
a7ee      1165: ;    Display routines
a7ee      1166: ;*****

```

```

a7ee      1167:
a7ee      1168: ;      display status line
a7ee      1169: ;      -----
a7ee      1170: dstatus
a7ee a908  1171:      lda    #low screen+200      ;40*5
a7f0 8582  1172:      sta    oup
a7f2 a99d  1173:      lda    #high screen+200    ;40*5
a7f4 8583  1174:      sta    oup+1
a7f6 a000  1175:      ldy    #0
a7f8 20b7a8 1176:      jsr    print
a7fb 42414e 1177:      dc.b   "BANK:",-1
a801 ad0606 1178:      lda    curbank
a804 2096a8 1179:      jsr    prby
a807 20b7a8 1180:      jsr    print
a80a 202050 1181:      dc.b   " PAGE:",-1
a812 ad0706 1182:      lda    curpage
a815 2096a8 1183:      jsr    prby
a818 20b7a8 1184:      jsr    print
a81b 20204c 1185:      dc.b   " LOOP:",-1
a823 ad0306 1186:      lda    loop+1
a826 2096a8 1187:      jsr    prby
a829 ad0206 1188:      lda    loop
a82c 2096a8 1189:      jsr    prby
a82f 20b7a8 1190:      jsr    print
a832 202045 1191:      dc.b   " ERROR:",-1
a83b ad0506 1192:      lda    comp+1
a83e 2096a8 1193:      jsr    prby
a841 ad0406 1194:      lda    comp
a844 2096a8 1195:      jsr    prby
a847 60     1196:      rts
a848      1197:
a848      1198:
a848      1199:
a848      1200: ;      display memory page
a848      1201: ;      -----
a848      1202: dispg
a848 20f3a8 1203:      jsr    inidat
a84b a900   1204:      lda    #0
a84d 8d0a06 1205:      sta    curlin
a850 ad0a06 1206: dispg2  lda    curlin
a853 290f   1207:      and    #15
a855 d007   1208:      bne    sk12
a857 20fca8 1209:      jsr    add40
a85a a004   1210:      ldy    #4
a85c d008   1211:      bne    sk11
a85e ad0a06 1212: sk12    lda    curlin
a861 2903   1213:      and    #3
a863 d001   1214:      bne    sk11
a865 c8     1215:      iny          ; skip a space if start o
a866 ae0a06 1216: sk11    ldx    curlin
a869 bd0040 1217:      lda    $4000,x
a86c 2096a8 1218:      jsr    prby          ; print value
a86f ee0a06 1219:      inc    curlin

```



```

a872 d0dc 1220:    bne  disp2
a874 60  1221:    rts
a875     1222:
a875     1223:
a875     1224:
a875     1225: ;    display address line starts
a875     1226: ;    -----
a875     1227: disform
a875 20f3a8 1228:    jsr  inidat
a878 20fca8 1229:    jsr  add40
a87b a900  1230:    lda  #0
a87d 8d0a06 1231: tloop sta  curlin
a880 a000  1232:    ldy  #0
a882 2096a8 1233:    jsr  prby
a885 a93a  1234:    lda  #'.'
a887 20a9a8 1235:    jsr  prch
a88a 20fca8 1236:    jsr  add40
a88d ad0a06 1237:    lda  curlin
a890 18    1238:    clc
a891 6910  1239:    adc  #$10
a893 d0e8  1240:    bne  tloop
a895 60    1241:    rts
a896     1242:
a896     1243:
a896     1244:
a896     1245: ;    print byte directly to screen
a896     1246: ;    -----
a896     1247: ; in:
a896     1248: ;    oup  = ptr into screen
a896     1249: ;    Y    = offset from ptr
a896     1250: prby
a896 48    1251:    pha
a897 4a    1252:    lsr
a898 4a    1253:    lsr
a899 4a    1254:    lsr
a89a 4a    1255:    lsr
a89b 209fa8 1256:    jsr  prnib
a89e 68    1257:    pla
a89f 290f  1258: prnib and  #$0F
a8a1 0930  1259:    ora  #$30
a8a3 c93a  1260:    cmp  #$3A
a8a5 9002  1261:    bcc  prch
a8a7 6906  1262:    adc  #6
a8a9     1263:
a8a9 c960  1264: prch  cmp  #$60
a8ab b006  1265:    bcs  prch2
a8ad e91f  1266:    sbc  #$1F
a8af b002  1267:    bcs  prch2
a8b1 6960  1268:    adc  #$60
a8b3 9182  1269: prch2 sta  (oup),y
a8b5 c8    1270:    iny
a8b6 60    1271:    rts
a8b7     1272:

```

```

a8b7 1273:
a8b7 1274:
a8b7 1275: ;      Print string of text
a8b7 1276: ;      -----
a8b7 1277: print
a8b7 ba 1278:      tsx
a8b8 bd0101 1279:      lda  $101,x
a8bb 8580 1280:      sta  zoch
a8bd bd0201 1281:      lda  $102,x
a8c0 8581 1282:      sta  zoch+1
a8c2 1283:
a8c2 8c0806 1284:      sty  chp
a8c5 a001 1285:      ldy  #1
a8c7 b180 1286: xzoch lda  (zoch),y
a8c9 c9ff 1287:      cmp  #$ff
a8cb f012 1288:      beq  zecho
a8cd 8c0906 1289:      sty  chp2
a8d0 ac0806 1290:      ldy  chp
a8d3 20a9a8 1291:      jsr  prch
a8d6 8c0806 1292:      sty  chp
a8d9 ac0906 1293:      ldy  chp2
a8dc c8 1294:      iny
a8dd d0e8 1295:      bne  xzoch
a8df 98 1296: zecho  tya
a8e0 ac0806 1297:      ldy  chp
a8e3 18 1298:      clc
a8e4 7d0101 1299:      adc  $101,x
a8e7 9d0101 1300:      sta  $101,x
a8ea bd0201 1301:      lda  $102,x
a8ed 6900 1302:      adc  #0
a8ef 9d0201 1303:      sta  $102,x
a8f2 60 1304:      rts
a8f3 1305:
a8f3 1306:
a8f3 1307:
a8f3 1308: inidat
a8f3 a930 1309:      lda  #low datsc-40
a8f5 8582 1310:      sta  oup
a8f7 a99d 1311:      lda  #high datsc-40
a8f9 8583 1312:      sta  oup+1
a8fb 60 1313:      rts
a8fc 1314:
a8fc 1315:
a8fc 1316:
a8fc 1317: add40
a8fc a582 1318:      lda  oup
a8fe 18 1319:      clc
a8ff 6928 1320:      adc  #40
a901 8582 1321:      sta  oup
a903 9002 1322:      bcc  nc1
a905 e683 1323:      inc  oup+1
a907 a000 1324: nc1  ldy  #0
a909 60 1325:      rts

```

```

a90a 1326:
a90a 1327:
a90a 1328:
a90a 1329: ; clear screen
a90a 1330: ; -----
a90a 1331: clear
a90a a940 1332: lda #low screen
a90c 8582 1333: sta oup
a90e a99c 1334: lda #high screen
a910 8583 1335: sta oup+1
a912 a218 1336: ldx #24
a914 a027 1337: c112 ldy #39
a916 a900 1338: lda #0
a918 9182 1339: c111 sta (oup),y
a91a 88 1340: dey
a91b 10fb 1341: bpl c111
a91d 20fca8 1342: jsr add40
a920 ca 1343: dex
a921 d0f1 1344: bne c112
a923 1345:
a923 a940 1346: lda #low screen
a925 8582 1347: sta oup
a927 a99c 1348: lda #high screen
a929 8583 1349: sta oup+1
a92b a000 1350: ldy #0
a92d 20b7a8 1351: jsr print
a930 1352: ; dc.b "Multi I/O Diagnostics Apr 04",-1
a930 1353:
a930 204d75 1354: dc.b " Multi I/O Diagnostics Feb 13
a958 202020 1355: dc.b " Copyright Intergraded Logic Sys
a980 1356: ; dc.b "
a980 202020 1357: dc.b " Telnet a real Atari BBS
a9a8 207463 1358: dc.b " tcpipexpress.com port 8888, 8889,
a9d0 ff 1359: dc.b -1
a9d1 60 1360: rts
a9d2 1361:
a9d2 1362:
a9d2 1363:
a9d2 1364:
a9d2 1365: ;*****
a9d2 1366: ; initialize system (quickly)
a9d2 1367: ;*****
a9d2 1368:
a9d2 1369: carini
a9d2 20cfaa 1370: jsr ihw ; IHW -- initialize hardware regi
a9d5 1371:
a9d5 a959 1372: lda #low irqhan
a9d7 8d1602 1373: sta $216
a9da a9aa 1374: lda #high irqhan
a9dc 8d1702 1375: sta $217
a9df 1376:
a9df a996 1377: lda #low nmihan
a9e1 8d2202 1378: sta $222

```

```

a9e4 a9aa 1379:   lda  #high nmihan
a9e6 8d2302 1380:   sta  $223
a9e9      1381:
a9e9 a937 1382:   lda  #low dlist
a9eb 8d02d4 1383:   sta  $d402
a9ee a9aa 1384:   lda  #high dlist
a9f0 8d03d4 1385:   sta  $d403      ; dlist
a9f3      1386:
a9f3 a922 1387:   lda  #$22
a9f5 8d00d4 1388:   sta  $d400      ; dmactl
a9f8 a901 1389:   lda  #1
a9fa 8d01d4 1390:   sta  $d401      ; chactl
a9fd a9e0 1391:   lda  #$e0
a9ff 8d09d4 1392:   sta  $d409      ; chbase
aa02      1393:
aa02 a980 1394:   lda  #$80      ;74
aa04 8d18d0 1395:   sta  $d018      ; color in PF2
aa07 a9ff 1396:   lda  #$ff      ;0a
aa09 8d17d0 1397:   sta  $d017      ; lum in PF1
aa0c      1398:
aa0c a940 1399:   lda  #$40
aa0e 8d0ed4 1400:   sta  $d40e      ; NMIEN
aa11 a940 1401:   lda  #$40
aa13 8510 1402:   sta  $10
aa15 8d0ed2 1403:   sta  $d20e      ; IRQEN
aa18      1404:
aa18 a913 1405:   lda  #$13
aa1a 8d0fd2 1406:   sta  $d20f      ; key board enable
aa1d 8d0ad2 1407:   sta  $d20a
aa20      1408:
aa20 a9ff 1409:   lda  #-1
aa22 8dfc02 1410:   sta  $2fc
aa25      1411:
aa25 a900 1412:   lda  #0
aa27 8d0e06 1413:   sta  keydel
aa2a 8d0f06 1414:   sta  srtimr
aa2d 8d0706 1415:   sta  curpage
aa30 8d0606 1416:   sta  curbank
aa33 58 1417:   CLI
aa34 4c0ba0 1418:   jmp  menu      ; finally enter MENU
aa37      1419: ;
aa37      1420:
aa37      1421:
aa37 707070 1422: dlist  dc.b  $70,$70,$70,$42
aa3b 409c 1423:   dc.w  $9c40
aa3d 100202 1424:   dc.b  $10,2,2,2,2,$10,2,2,$20
aa46 020202 1425:   dc.b  2,2,2,2,2,2,2,2,2,2,2,2,2,2,$41
aa57 37aa 1426:   dc.w  dlist
aa59      1427:
aa59      1428:
aa59      1429:
aa59      1430: ;   IRQ handler for key board
aa59      1431: ;   -----

```

```

aa59      1432: irqhan
aa59 48    1433:      pha
aa5a ad0ed2 1434:      lda  $d20E ; irqst
aa5d 2940   1435:      and  #$40
aa5f d00d   1436:      bne  noi
aa61 a9bf   1437:      lda  #$BF
aa63 8d0ed2 1438:      sta  $d20E
aa66 a510   1439:      lda  $10
aa68 8d0ed2 1440:      sta  $d20E
aa6b 2070aa 1441:      jsr  keybd
aa6e 68     1442: noi    pla
aa6f 40     1443:      rti
aa70      1444:
aa70      1445:
aa70      1446:
aa70      1447: keybd
aa70 8a     1448:      txa
aa71 48     1449:      pha
aa72 98     1450:      tya
aa73 48     1451:      pha
aa74 ad09d2 1452:      lda  $d209 ; kbcode
aa77 cd1006 1453:      cmp  ch1
aa7a d005   1454:      bne  kir1
aa7c ae0e06 1455:      ldx  keydel
aa7f d00b   1456:      bne  kir8 ; must delay more
aa81 8dfc02 1457: kir1   sta  $2fc
aa84 8d1006 1458:      sta  ch1
aa87 a903   1459:      lda  #3
aa89 8d0e06 1460:      sta  keydel
aa8c a918   1461: kir8   lda  #24
aa8e 8d0f06 1462:      sta  srtimr
aa91 68     1463:      pla
aa92 a8     1464:      tay
aa93 68     1465:      pla
aa94 aa     1466:      tax
aa95 60     1467:      rts
aa96      1468:
aa96      1469:
aa96      1470:
aa96      1471: ;      NMI handler for key board
aa96      1472: ;      -----
aa96      1473: nmihan
aa96 ad0fd2 1474:      lda  $d20f ; SKSTAT
aa99 2904   1475:      and  #4
aa9b f008   1476:      beq  ivn11 ; jump if key down
aa9d ad0e06 1477:      lda  keydel ; jump if key up and counted down
aaa0 f003   1478:      beq  ivn11
aaa2 ce0e06 1479:      dec  keydel
aaa5 ad0f06 1480: ivn11   lda  srtimr ; repeat timer
aaa8 f01f   1481:      beq  ivn13 ; jump if expired
aaaa ad0fd2 1482:      lda  $d20f
aaad 2904   1483:      and  #4
aaaf d013   1484:      bne  ivn12 ; jump if key up

```

```

aab1 ce0f06 1485:      dec  srtimr
aab4 d013  1486:      bne  ivn13
aab6 a903  1487:      lda  #3
aab8 8d0f06 1488:      sta  srtimr ; reset repeat counter
aabb ad09d2 1489:      lda  $d209
aabe 8dfc02 1490:      sta  $2fc
aac1 4cc9aa 1491:      jmp  ivn13
aac4 a900  1492: ivn12  lda  #0
aac6 8d0f06 1493:      sta  srtimr
aac9 68    1494: ivn13  pla
aaca a8    1495:      tay
aacb 68    1496:      pla
aacc aa    1497:      tax
aacd 68    1498:      pla
aace 40    1499:      rti
aacf      1500:
aacf      1501:
aacf      1502:
aacf      1503: ;      initialize hardware
aacf      1504: ;      -----
aacf      1505: ihw
aacf a900  1506:      lda  #0
aad1 aa    1507:      tax
aad2 8d03d3 1508:      sta  $D303
aad5 9d00d0 1509: ihw1  sta  $d000,x
aad8 9d00d4 1510:      sta  $d400,x
aadb 9d00d2 1511:      sta  $d200,x
aade e001  1512:      cpx  #1
aae0 f003  1513:      beq  ihw2
aae2 9d00d3 1514:      sta  $d300,x
aae5 e8    1515: ihw2  inx
aae6 d0ed  1516:      bne  ihw1
aae8      1517:
aae8 a93c  1518:      lda  #$3c
aaea 8d03d3 1519:      sta  $d303
aaed a9ff  1520:      lda  #$ff
aaef 8d01d3 1521:      sta  $d301
aaf2 a938  1522:      lda  #$38
aaf4 8d02d3 1523:      sta  $d302
aaf7 8d03d3 1524:      sta  $d303
aafa a900  1525:      lda  #0
aafc 8d00d3 1526:      sta  $d300
aaff a9ff  1527:      lda  #$ff
ab01 8d01d3 1528:      sta  $d301
ab04 a93c  1529:      lda  #$3c
ab06 8d02d3 1530:      sta  $d302
ab09 8d03d3 1531:      sta  $d303
ab0c ad01d3 1532:      lda  $d301
ab0f ad00d3 1533:      lda  $d300
ab12      1534:
ab12 a922  1535:      lda  #$22
ab14 8d0fd2 1536:      sta  $d20f ; SKCTL
ab17      1537:

```

```
ab17 a9a0 1538:    lda  #$a0
ab19 8d05d2 1539:    sta  $d205
ab1c 8d07d2 1540:    sta  $d207 ; audc3,4
ab1f      1541:
ab1f a928 1542:    lda  #$28
ab21 8d08d2 1543:    sta  $d208
ab24      1544:
ab24 a9ff 1545:    lda  #$ff
ab26 8d0dd2 1546:    sta  $d20d ; serout
ab29 60    1547:    rts
ab2a      1548:
ab2a      1549:
ab2a      1550: ;You would need to set a cart of for this.
ab2a      1551:
ab2a      1552: ; .org $bff0 ; seg segbff0
ab2a      1553: ;
ab2a      1554: ; sta $D5E0 ; switch to bank 0 (initially set
ab2a      1555: ; jmp carini
ab2a      1556: ; dc.w 0,0
ab2a      1557: ; dc.w 0 ; run address
ab2a      1558: ; dc.b 0 ; (0 indicatess cartridge presen
ab2a      1559: ; dc.b $80 ; special execution desired
ab2a      1560: ; dc.w $BFF0
ab2a      1561: ;
ab2a      1562: win_end: ds.b 0
ab2a      1563: ;
ab2a      1564: ;
ab2a      1565: ;
```

End assembly: no errors