



```
0000 1: ;-----
0000 2: ;
0000 3: ;      Copyright 2012 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 'Drive programming Stuff'
0000 10: ;-----
0000 11: ;      make_bad.s
0000 12: ;With the super achiver with bit writer
0000 13: ;      ~~~~~
0000 14: ;-----
0000 15: ;      Real.dos Make_bad
0000 16: ;
0000 17: ;
0000 18: ;      Code History.  Init design, 7/5/2012    sjc
0000 19: ;
0000 20: ;
0000 21: ;
0000 22: ;      ~~~~~
0000 23: ;-----
0000 24: ; Notes: o This source code MAY NOT be placed for download
0000 25: ;      o "mea" is a macro that loads the address of the
0000 26: ;      into the pointer specified by the second field.
0000 27: ;-----
0000 28: ; Assembler: MADMAC (tm) ST Cross Assembler (Atari Corp)
0000 29: ;      XASM  st and IBM VERSIONS
0000 30: ;-----
0000 31:
0000 32: com_file_name:      .macro
0000 33: ;      dc.b 13,"filename.ext",$9b
0000 34: ;      dc.b 12,"Make_bad.com",$9b
0000 35: ;      .endm
0000 36:
0000 37: ;
0000 38: ;~~~~~
0000 39: ; File revision history. Version Number in hex
0000 40: ;
0000 41: file_ver:      equ $10
0000 42: ;~~~~~
0000 43: ; the Month in dec for the first time this revision
0000 44: ;was compiled
0000 45: ;
0000 46: c_month:      equ 10
0000 47: ;~~~~~
0000 48: ; the day in dec for the first time this revision
0000 49: ;was compiled
0000 50: c_day:      equ 16
0000 51: ;~~~~~
0000 52: ; the year in dec for the first time this revision
0000 53: ;was compiled
```

```

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

```

```

0000      107:
0000      108:
0000      109: d_sector: equ 365
0000      110: b_code: equ fi_sector+1
0000      111: start_fuzz: equ 64
0000      112: end_fuzz: equ 127
0000      113: ;
0000      114: ;
0000      115: ;
0000      116:
0000      117:
0000      118: .include equates
0000      119: .include globals
0000      120: .include macros
0000      121:
0000      122:
5000      123: .org $5000
5000      124:
5000      125:
5000      126: ;
5000      127: ;
5000      128: win_start:
5000      129: header_info:
5000      130:
5000      131: .include header ; t
52a7      132:
52a7 4c4359 133: jmp top
52aa      134: ;-----
52aa      135: ;prints Routeen macro
52aa      136: ;this is a slow print to the screen
52aa      137: ;-----
52aa      138: ;
52aa 68      139: sprintsi: pla
52ab 85b0     140: sta $b0
52ad 68      141: pla
52ae 85b1     142: sta $b1
52b0 a001     143: ldy #$01
52b2 b1b0     144: e8894: lda ($b0),y
52b4 c9ff     145: cmp #$ff
52b6 f009     146: beq m88a1
52b8 20d252   147: jsr sprint
52bb 20cb52   148: jsr l88ab
52be 4cb252   149: jmp e8894
52c1 20cb52   150: m88a1: jsr l88ab
52c4 a5b1     151: lda $b1
52c6 48      152: pha
52c7 a5b0     153: lda $b0
52c9 48      154: pha
52ca 60      155: rts
52cb e6b0     156: l88ab: inc $b0
52cd d002     157: bne l88b1
52cf e6b1     158: inc $b1
52d1 60      159: l88b1: rts

```

```

52d2      160: ;
52d2      161: ;
52d2      162: ;-----;
52d2      163: ; PRINT BYTE
52d2      164: ;-----;
52d2      165: echo:
52d2 85b8  166: sprint: sta $b8
52d4 a904  167:      lda #$04
52d6 8d4a03 168:      sta $034a
52d9 86b6  169:      stx $b6
52db 84b7  170:      sty $b7
52dd a206  171:      ldx #$06
52df bd01e4 172:      lda $e400+1,x
52e2 85b3  173:      sta $b3
52e4 bd00e4 174:      lda $e400,x
52e7 85b2  175:      sta $b2
52e9 a5b8  176:      lda $b8
52eb 20fa52 177:      jsr 188da
52ee a6b6  178:      ldx $b6
52f0 a4b7  179:      ldy $b7
52f2 a90c  180:      lda #$0c
52f4 8d4a03 181:      sta $034a
52f7 a5b8  182:      lda $b8
52f9 60     183:      rts
52fa      184: ;
52fa e6b2  185: 188da: inc $b2
52fc 6cb200 186:      jmp ($b2)
52ff 60     187:      rts
5300      188: ;-----;
5300      189: ; GET KEY ROUTEEN      ;
5300      190: ;this routeen get's one key and;
5300      191: ;return it to the accumulator ;
5300      192: ;-----;
5300      193: ;
5300 86b6  194: getkey: stx $b6
5302 84b7  195:      sty $b7
5304 200c53 196:      jsr g_fff
5307 a6b6  197:      ldx $b6
5309 a4b7  198:      ldy $b7
530b 60     199:      rts
530c ad25e4 200: g_fff: lda $e420+5
530f 48     201:      pha
5310 ad24e4 202:      lda $e420+4
5313 48     203:      pha
5314 60     204:      rts
5315      205: ;
5315      206: ;-----;
5315      207: ;CHANGES THE CHARTER TO UPCASE;
5315      208: ;-----;
5315      209: upcase:
5315 c961  210:      cmp #$61
5317 9006  211:      bcc upcase_exit
5319 c97a  212:      cmp #$7a

```

```

531b b002 213:      bcs upcase_exit
531d e91f 214:      sbc #$1f
531f 60   215: upcase_exit: rts
5320      216: ;
5320      217: ;-----;
5320      218: ;CHANGES THE CHARTER TO LOCASE;
5320      219: ;-----;
5320      220: locase:
5320 c941 221:      cmp #$41
5322 9006 222:      bcc locase_exit
5324 c95a 223:      cmp #$5a
5326 b002 224:      bcs locase_exit
5328 691f 225:      adc #$1f
532a 60   226: locase_exit: rts
532b      227: ;
532b      228: ;-----;
532b      229: ;  ERROR NUMBER          ;
532b      230: ;this take's the value of the a reg;
532b      231: ;and put a hex number to the screen;
532b      232: ;-----;
532b      233: ;
532b 00   234: h_cardd: dc.b $00
532c      235: ;
532c      236: ; print a hex word value
532c      237: ; low byte in a reg x is high byte
532c      238: ;
532c 8d2b53 239: h_card: sta h_cardd
532f 8a   240:      txa
5330 203a53 241:      jsr drive_error
5333 ad2b53 242:      lda h_cardd
5336 203a53 243:      jsr drive_error
5339 60   244:      rts
533a      245: ;
533a      246: ;
533a 48   247: drive_error: pha
533b 4a   248:      lsr
533c 4a   249:      lsr
533d 4a   250:      lsr
533e 4a   251:      lsr
533f 204553 252:      jsr l8913
5342 68   253:      pla
5343 290f 254:      and #$0f
5345 c90a 255: l8913: cmp #$0a
5347 b004 256:      bcs l891b
5349 0930 257:      ora #$30
534b d002 258:      bne l891d
534d 6936 259: l891b: adc #$36
534f 20d252 260: l891d: jsr sprint
5352 60   261:      rts
5353      262: ;
5353      263: ;
5353      264: ;-----;
5353      265: ; jsr printsi          ;

```

```

5353      266: ;this prints an inline string;
5353      267: ;   terminated by $ff   ;
5353      268: ;-----;
5353      269: ;   jsr prints
5353      270: ;   .BYTE "print this",$9B,$FF
5353      271: echosi:
5353 68      272: prints: pla
5354 8d6453 273:      sta pstr+1
5357 68      274:      pla
5358 8d6553 275:      sta pstr+2
535b ee6453 276: prsl:   inc pstr+1
535e d003 277:      bne pstr
5360 ee6553 278:      inc pstr+2
5363 adffff 279: pstr:   lda $ffff
5366 c9ff 280:      cmp #$ff
5368 f006 281:      beq estri
536a 20a953 282:      jsr fast_output
536d 4c5b53 283:      jmp prsl
5370 ad6553 284: estri:  lda pstr+2
5373 48      285:      pha
5374 ad6453 286:      lda pstr+1
5377 48      287:      pha
5378 60      288:      rts
5379      289: ;
5379      290: ;-----;
5379      291: ;actual screen handler ;
5379      292: ;screen offset definitions;
5379      293: ;-----;
5379 0000 294: scr_offset: dc.w 0
537b 2800 295:      dc.w 40
537d 5000 296:      dc.w 80
537f 7800 297:      dc.w 120
5381 a000 298:      dc.w 160
5383 c800 299:      dc.w 200
5385 f000 300:      dc.w 240
5387 1801 301:      dc.w 280
5389 4001 302:      dc.w 320
538b 6801 303:      dc.w 360
538d 9001 304:      dc.w 400
538f b801 305:      dc.w 440
5391 e001 306:      dc.w 480
5393 0802 307:      dc.w 520
5395 3002 308:      dc.w 560
5397 5802 309:      dc.w 600
5399 8002 310:      dc.w 640
539b a802 311:      dc.w 680
539d d002 312:      dc.w 720
539f f802 313:      dc.w 760
53a1 2003 314:      dc.w 800
53a3 4803 315:      dc.w 840
53a5 7003 316:      dc.w 880
53a7 9803 317:      dc.w 920
53a9      318: ;

```

```
53a9 855a 319: fast_output: sta $5a
53ab c920 320:      cmp #32
53ad 9004 321:      bcc do_look
53af c97d 322:      cmp #125
53b1 9013 323:      bcc do_here
53b3 a000 324: do_look: ldy #0
53b5 b9bc54 325: lookup: lda char_table,y
53b8 f00c 326:      beq do_here
53ba c55a 327:      cmp $5a
53bc f003 328:      beq go_cio
53be c8 329:      iny
53bf d0f4 330:      bne lookup
53c1 a55a 331: go_cio: lda $5a
53c3 4ccc54 332:      jmp putlocal
53c6 a000 333: do_here: ldy #0
53c8 a55d 334:      lda $5d
53ca 915e 335:      sta ($5e),y
53cc a55a 336:      lda $5a
53ce c99b 337:      cmp #$9b
53d0 f044 338:      beq docr
53d2 209b54 339: notcr: jsr get_adr
53d5 a55a 340:      lda $5a
53d7 297f 341:      and #$7f
53d9 c920 342:      cmp #32
53db 9007 343:      bcc add64
53dd c960 344:      cmp #96
53df 9009 345:      bcc sub32
53e1 4ced53 346:      jmp asis
53e4 18 347: add64: clc
53e5 6940 348:      adc #64
53e7 4ced53 349:      jmp asis
53ea 38 350: sub32: sec
53eb e920 351:      sbc #32
53ed 245a 352: asis:  bit $5a
53ef 1002 353:      bpl xxlate
53f1 0980 354:      ora #$80
53f3 a000 355: xxlate: ldy #0
53f5 915e 356:      sta ($5e),y
53f7 e655 357:      inc $55
53f9 e663 358:      inc $63
53fb a555 359:      lda $55
53fd c928 360:      cmp #40
53ff b019 361:      bcs next_row
5401 c8 362:      iny
5402 b15e 363:      lda ($5e),y
5404 855d 364:      sta $5d
5406 0980 365:      ora #$80
5408 aef002 366:      ldx 752
540b d002 367:      bne nocurs1
540d 915e 368:      sta ($5e),y
540f e65e 369: nocurs1: inc $5e
5411 d002 370:      bne nooav
5413 e65f 371:      inc $5e+1
```



---

```
5415 60      372: nooav: rts
5416 a900    373: docr:  lda #0
5418 8563    374:      sta $63
541a a552    375: next_row: lda $52
541c 8555    376:      sta $55
541e e654    377:      inc $54
5420 a454    378:      ldy $54
5422 c018    379:      cpy #24
5424 b013    380:      bcs scroll
5426 209b54  381:      jsr get_adr
5429 a000    382:      ldy #0
542b b15e    383:      lda ($5e),y
542d 855d    384:      sta $5d
542f aef002  385:      ldx 752
5432 d004    386:      bne nocurs2
5434 0980    387:      ora #$80
5436 915e    388:      sta ($5e),y
5438 60      389: nocurs2: rts
5439 c654    390: scroll:  dec $54
543b a558    391:      lda $58
543d 8568    392:      sta $68
543f 18      393:      clc
5440 6928    394:      adc #40
5442 855e    395:      sta $5e
5444 a559    396:      lda $58+1
5446 8569    397:      sta $68+1
5448 6900    398:      adc #0
544a 855f    399:      sta $5e+1
544c a000    400:      ldy #0
544e a203    401:      ldx #3
5450 b15e    402: scloop:  lda ($5e),y
5452 9168    403:      sta ($68),y
5454 c8      404:      iny
5455 d0f9    405:      bne scloop
5457 e65f    406:      inc $5e+1
5459 e669    407:      inc $68+1
545b ca      408:      dex
545c d0f2    409:      bne scloop
545e b15e    410: scloop2: lda ($5e),y
5460 9168    411:      sta ($68),y
5462 c8      412:      iny
5463 c098    413:      cpy #152
5465 90f7    414:      bcc scloop2
5467 a558    415:      lda $58
5469 18      416:      clc
546a 6998    417:      adc #920&$ff
546c 855e    418:      sta $5e
546e a559    419:      lda $58+1
5470 6903    420:      adc #920/256
5472 855f    421:      sta $5e+1
5474 a000    422:      ldy #0
5476 98      423:      tya
5477 915e    424: clear:  sta ($5e),y
```

```

5479 c8 425: iny
547a c028 426: cpy #40
547c 90f9 427: bcc clear
547e aef002 428: ldx 752
5481 d006 429: bne nocurs3
5483 a980 430: lda #$80
5485 a455 431: ldy $55
5487 915e 432: sta ($5e),y
5489 a900 433: nocurs3: lda #0
548b 855d 434: sta $5d
548d a55e 435: lda $5e
548f 18 436: clc
5490 6555 437: adc $55
5492 855e 438: sta $5e
5494 a55f 439: lda $5e+1
5496 6900 440: adc #0
5498 855f 441: sta $5e+1
549a 60 442: rts
549b a554 443: get_adr: lda $54
549d 0a 444: asl
549e a8 445: tay
549f a558 446: lda $58
54a1 18 447: clc
54a2 797953 448: adc scr_offset,y
54a5 855e 449: sta $5e
54a7 a559 450: lda $58+1
54a9 797a53 451: adc scr_offset+1,y
54ac 855f 452: sta $5e+1
54ae a55e 453: lda $5e
54b0 18 454: clc
54b1 6555 455: adc $55
54b3 855e 456: sta $5e
54b5 a55f 457: lda $5e+1
54b7 6900 458: adc #0
54b9 855f 459: sta $5e+1
54bb 60 460: rts
54bc 461: char_table:
54bc 1b1c1d 462: dc.b 27,28,29,30,31,125,126,127
54c4 9c9d9e 463: dc.b 156,157,158,159,253,254,255,0
54cc a20b 464: putlocal: ldx #11
54ce 8e4203 465: stx $0342
54d1 a200 466: ldx #0
54d3 8e4803 467: stx $0348
54d6 8e4903 468: stx $0349
54d9 4c56e4 469: jmp $e456
54dc 470: ;
54dc 471: ;
54dc 472: ;-----;
54dc 473: ;THIS ROUTEEN WILL PAUSE 1 SEC. ;
54dc 474: ; PAUSE JIFFIY ;
54dc 475: ;USES LOAD THE A REGISTER JSR PAUSE;
54dc 476: ;-----;
54dc 477: ;

```

```
54dc      478: pause:
54dc 8dfc54 479:      sta pause_data
54df a900  480:      lda #$00
54e1 8dfe54 481:      sta pause_data+2
54e4 a900  482: p_l1:  lda #$00
54e6 8514  483:      sta $14
54e8 18    484:      clc
54e9      485: pause_loop:
54e9 a514  486:      lda $14
54eb cdfd54 487:      cmp pause_data+1
54ee 90f9  488:      bcc pause_loop
54f0 eefe54 489:      inc pause_data+2
54f3 adfe54 490:      lda pause_data+2
54f6 cdfc54 491:      cmp pause_data
54f9 90e9  492:      bcc p_l1
54fb 60    493:      rts
54fc 003c00 494: pause_data: dc.b $00,60,$00
54ff      495: ;-----
54ff      496: ;   JIFFIY PAUSE
54ff      497: ;-----
54ff      498: ;
54ff 8d0f55 499: jiffiy: sta jiy_h
5502 18    500:      clc
5503 a900  501:      lda #$00
5505 8514  502:      sta $14
5507 a514  503: jif_loop: lda $14
5509 cd0f55 504:      cmp jiy_h
550c 90f9  505:      bcc jif_loop
550e 60    506:      rts
550f 00    507: jiy_h:  dc.b 0
5510      508: ;
5510      509: ;
5510      510: ;
5510      511: ;-----
5510      512: ;   JSR ECHOS
5510      513: ;LOAD THE LOW BYTE INTO fi_pr_ptr
5510      514: ;LOAD THE HIGH BYTE INTO fi_pr_ptr+1
5510      515: ;JSR ECHOS.BYTE $00,"NAME"
5510      516: ;-----
5510      517: ;
5510      518: prints:
5510      519: echos:
5510 48    520:      pha
5511 48    521:      pha
5512 18    522:      clc
5513 ad4255 523:      lda fi_pr_ptr
5516 8d3255 524:      sta e_print+1
5519 8d3b55 525:      sta a_print+1
551c ad4355 526:      lda fi_pr_ptr+1
551f 8d3355 527:      sta e_print+2
5522 8d3c55 528:      sta a_print+2
5525 a900  529:      lda #$00
5527 8d4455 530:      sta fi_pr_ptr+2
```

```
552a ac4455 531: w_print: ldy fi_pr_ptr+2
552d c8 532: iny
552e 8c4455 533: sty fi_pr_ptr+2
5531 b9ffff 534: e_print: lda $ffff,y
5534 20d252 535: jsr sprint
5537 ac4455 536: ldy fi_pr_ptr+2
553a ccffff 537: a_print: cpy $ffff
553d 90eb 538: bcc w_print
553f 68 539: pla
5540 68 540: pla
5541 60 541: rts
5542 000000 542: fi_pr_ptr: dc.b 0,0,0
5545 543: ;
5545 544: ;
5545 545: ;-----
5545 546: ; JSR GETSTRING
5545 547: ;LOAD THE LOW BYTE INTO fi_s_ptr
5545 548: ;THE HIGH BYTE INTO fi_s_ptr+1
5545 549: ;LOAD THE MAX LENGHT INTO Y REG.
5545 550: ;-----
5545 551: ;
5545 552: ;
5545 553: getstring:
5545 a99b 554: lda #$9b
5547 8d9055 555: sta x1_get+1
554a 8c9f55 556: sty x_get+1
554d e0ff 557: cpx #$ff
554f d003 558: bne wq_get
5551 8e9055 559: stx x1_get+1
5554 adc955 560: wq_get: lda fi_s_ptr
5557 8d8d55 561: sta g_get+1
555a 8db355 562: sta b_get+1
555d 8d9855 563: sta s_get+1
5560 8db655 564: sta w_get+1
5563 8da855 565: sta t_get+1
5566 8dac55 566: sta back_s+1
5569 adca55 567: lda fi_s_ptr+1
556c 8d8e55 568: sta g_get+2
556f 8db455 569: sta b_get+2
5572 8d9955 570: sta s_get+2
5575 8db755 571: sta w_get+2
5578 8da955 572: sta t_get+2
557b 8dad55 573: sta back_s+2
557e a001 574: ldy #$01
5580 18 575: get_loop: clc
5581 200053 576: jsr getkey
5584 c91c 577: cmp #$1c ;CURSER UP
5586 9004 578: bcc g_get
5588 c920 579: cmp #$20
558a 90f4 580: bcc get_loop
558c 99ffff 581: g_get: sta $ffff,y
558f c99b 582: x1_get: cmp #$9b
5591 f011 583: beq g_exit
```

```

5593 c97e 584:      cmp #$7e
5595 f014 585:      beq back_s
5597 eeffff 586: s_get: inc $ffff
559a 20d252 587:      jsr sprint
559d c8 588:      iny
559e c0ff 589: x_get: cpy #$ff
55a0 f002 590:      beq g_exit
55a2 d0dc 591:      bne get_loop
55a4 20d252 592: g_exit: jsr sprint
55a7 eeffff 593: t_get: inc $ffff
55aa 60 594:      rts
55ab adffff 595: back_s: lda $ffff
55ae c900 596:      cmp #$00
55b0 f0ce 597:      beq get_loop
55b2 ceffff 598: b_get: dec $ffff
55b5 adffff 599: w_get: lda $ffff
55b8 c900 600:      cmp #$00
55ba 90c4 601:      bcc get_loop
55bc 98 602:      tya
55bd 88 603:      dey
55be a97e 604:      lda #$7e
55c0 20d252 605:      jsr sprint
55c3 c900 606:      cmp #$00
55c5 d0b9 607:      bne get_loop
55c7 f0b7 608:      beq get_loop
55c9      609:
55c9      610:
55c9 0000 611: fi_s_ptr: dc.w 0
55cb      612:
55cb 0000 613: fi_d_ptr: dc.w 0
55cd 000000 614: fi_test_byte: dc.b 0,0,0,0
55d1      615: ;
55d1      616: ;
55d1      617: ;-----
55d1      618: ;COMPARE STRING'S fi_s_ptr TO fi_d_ptr
55d1      619: ;-----
55d1      620: ;
55d1 18 621: cmpstr: clc
55d2 a901 622:      lda #$01
55d4 8dd055 623:      sta fi_test_byte+3
55d7 adc955 624:      lda fi_s_ptr
55da 8d1a56 625:      sta c_sptr+1
55dd 8d0a56 626:      sta cmp_word+1
55e0 8d0256 627:      sta cmp_check1+1
55e3 adca55 628:      lda fi_s_ptr+1
55e6 8d1b56 629:      sta c_sptr+2
55e9 8d0b56 630:      sta cmp_word+2
55ec 8d0356 631:      sta cmp_check1+2
55ef adcb55 632:      lda fi_d_ptr
55f2 8d2356 633:      sta c_dptr+1
55f5 8d0556 634:      sta cmp_check2+1
55f8 adcc55 635:      lda fi_d_ptr+1
55fb 8d2456 636:      sta c_dptr+2

```

```

55fe 8d0656 637:      sta cmp_check2+2
5601 adffff 638: cmp_check1: lda $ffff
5604 cdffff 639: cmp_check2: cmp $ffff
5607 d026 640:      bne w_cmpstr
5609 adffff 641: cmp_word: lda $ffff
560c 8dcf55 642:      sta fi_test_byte+2
560f a000 643:      ldy #$00
5611 c8 644: cmp_loop: iny
5612 18 645:      clc
5613 cccf55 646:      cpy fi_test_byte+2
5616 f01c 647:      beq cmp_exit
5618 18 648:      clc
5619 b9ffff 649: c_sptr:  lda $ffff,y
561c 202053 650:      jsr locase
561f 8dcd55 651:      sta fi_test_byte
5622 b9ffff 652: c_dptr:  lda $ffff,y
5625 202053 653:      jsr locase
5628 cdc55 654:      cmp fi_test_byte
562b d002 655:      bne w_cmpstr
562d f0e2 656:      beq cmp_loop
562f      657: w_cmpstr:
562f a900 658:      lda #$00
5631 8dd055 659:      sta fi_test_byte+3
5634 add055 660: cmp_exit: lda fi_test_byte+3
5637 c901 661:      cmp #$01
5639 60 662:      rts
563a      663: ;
563a      664: ;
563a      665: ;-----
563a      666: ; LENGHT TO 0 BYTE STRING
563a      667: ; SRC STRING fi_s_ptr DES STRING fi_d_ptr
563a      668: ;-----
563a      669: ;
563a      670: toascii:
563a adc955 671:      lda fi_s_ptr
563d 8d6b56 672:      sta src_asc+1
5640 8d6356 673:      sta src_hold+1
5643 adca55 674:      lda fi_s_ptr+1
5646 8d6c56 675:      sta src_asc+2
5649 8d6456 676:      sta src_hold+2
564c adcb55 677:      lda fi_d_ptr
564f 8d6656 678:      sta des_hold+1
5652 8d7456 679:      sta src_e+1
5655 adcc55 680:      lda fi_d_ptr+1
5658 8d6756 681:      sta des_hold+2
565b 8d7556 682:      sta src_e+2
565e a001 683:      ldy #$01
5660 a200 684:      ldx #$00
5662 b9ffff 685: src_hold: lda $ffff,y
5665 9dffff 686: des_hold: sta $ffff,x
5668 c8 687:      iny
5669 e8 688:      inx
566a ccffff 689: src_asc: cpy $ffff

```

```

566d f002 690:      beq src_exit
566f d0f1 691:      bne src_hold
5671 a900 692: src_exit: lda #$00
5673 9dffff 693: src_e:  sta $ffff,x
5676 60 694:      rts
5677      695: ;
5677      696: ;-----
5677      697: ; 0 TO LENGHT STRING
5677      698: ;-----
5677      699: tostr:
5677 adc955 700:      lda fi_s_ptr
567a 8da356 701:      sta tostr_sr+1
567d adca55 702:      lda fi_s_ptr+1
5680 8da456 703:      sta tostr_sr+2
5683 adcb55 704:      lda fi_d_ptr
5686 8daa56 705:      sta tostr_dr+1
5689 8daf56 706:      sta tostr_hold+1
568c 8d9e56 707:      sta tostr_ho+1
568f adcc55 708:      lda fi_d_ptr+1
5692 8dab56 709:      sta tostr_dr+2
5695 8db056 710:      sta tostr_hold+2
5698 8d9f56 711:      sta tostr_ho+2
569b a000 712:      ldy #$00
569d 8cffff 713: tostr_ho: sty $ffff
56a0 a201 714:      ldx #$01
56a2 b9ffff 715: tostr_sr: lda $ffff,y
56a5 c900 716:      cmp #$00
56a7 f00c 717:      beq tostr_exit
56a9 9dffff 718: tostr_dr: sta $ffff,x
56ac c8 719:      iny
56ad e8 720:      inx
56ae eeffff 721: tostr_hold: inc $ffff
56b1 f0ef 722:      beq tostr_sr
56b3 d0ed 723:      bne tostr_sr
56b5      724: tostr_exit:
56b5 60 725:      rts
56b6      726: ;
56b6      727: ;
56b6      728: ;-----
56b6      729: ; JSR CONCAT APPENDS
56b6      730: ; SPRT TO fi_d_ptr
56b6      731: ;-----
56b6      732: ;
56b6      733: concat:
56b6 adc955 734:      lda fi_s_ptr
56b9 8de756 735:      sta con_s+1
56bc 8df256 736:      sta con_hold+1
56bf adca55 737:      lda fi_s_ptr+1
56c2 8de856 738:      sta con_s+2
56c5 8df356 739:      sta con_hold+2
56c8 adcb55 740:      lda fi_d_ptr
56cb 8dea56 741:      sta con_d+1
56ce 8de156 742:      sta con_de+1

```

```

56d1 8ded56 743:      sta con_dd+1
56d4 adcc55 744:      lda fi_d_ptr+1
56d7 8deb56 745:      sta con_d+2
56da 8de256 746:      sta con_de+2
56dd 8dee56 747:      sta con_dd+2
56e0 acffff 748: con_de: ldy $ffff
56e3 c8     749:      iny
56e4 a201   750:      ldx #$01
56e6 bdffff 751: con_s:  lda $ffff,x
56e9 99ffff 752: con_d:  sta $ffff,y
56ec eeffff 753: con_dd:  inc $ffff
56ef c8     754:      iny
56f0 e8     755:      inx
56f1 ecffff 756: con_hold: cpx $ffff
56f4 30f0   757:      bmi con_s
56f6 60     758:      rts
56f7       759: ;
56f7       760: ;
56f7       761: ;-----
56f7       762: ;JSR MOVESTR
56f7       763: ;-----
56f7       764: movestr:
56f7 adc955 765:      lda fi_s_ptr
56fa 8d1957 766:      sta mov_s+1
56fd 8d1657 767:      sta mov_h+1
5700 adca55 768:      lda fi_s_ptr+1
5703 8d1a57 769:      sta mov_s+2
5706 8d1757 770:      sta mov_h+2
5709 adcb55 771:      lda fi_d_ptr
570c 8d1c57 772:      sta mov_d+1
570f adcc55 773:      lda fi_d_ptr+1
5712 8d1d57 774:      sta mov_d+2
5715 acffff 775: mov_h:  ldy $ffff
5718 b9ffff 776: mov_s:  lda $ffff,y
571b 99ffff 777: mov_d:  sta $ffff,y
571e c000   778:      cpy #$00
5720 f005   779:      beq mov_ew
5722 88     780:      dey
5723 d0f3   781:      bne mov_s
5725 f0f1   782:      beq mov_s
5727 60     783: mov_ew: rts
5728       784: ;-----
5728       785: ;JSR GETCARD
5728       786: ;
5728       787: ;-----
5728 002020 788: gtbyte: dc.b $00,"    "
5730 202020 789: qstrin: dc.b "    "
5737 0306   790: gtwr:  dc.b $03,$06
5739       791: getbyte:
5739 ad3757 792:      lda gtwr
573c 8d5557 793:      sta gt1+1
573f a000   794:      ldy #$00
5741 8c2857 795:      sty gtbyte

```



```
5744 cd3757 796:      cmp gtwr
5747 f00b 797:      beq gt1
5749          798: getcard:
5749 ad3857 799:      lda gtwr+1
574c 8d5557 800:      sta gt1+1
574f a000 801:      ldy #$00
5751 8c2857 802:      sty gtbyte
5754 c0ff 803: gt1:   cpy #$ff
5756 f03e 804:      beq gt3
5758 200053 805:      jsr getkey
575b 201553 806:      jsr upcase
575e c99b 807:      cmp #$9b
5760 f034 808:      beq gt3
5762 c97e 809:      cmp #$7e ; BACKSPACE
5764 f01b 810:      beq gt2
5766 c930 811:      cmp #'0'
5768 90ea 812:      bcc gt1
576a c93a 813:      cmp #':'
576c 10e6 814:      bpl gt1
576e ac2857 815:      ldy gtbyte
5771 c8 816:      iny
5772 8c2857 817:      sty gtbyte
5775 992857 818:      sta gtbyte,y
5778 20d252 819:      jsr echo
577b c9ff 820:      cmp #$ff
577d d0d5 821:      bne gt1
577f f0d3 822:      beq gt1
5781 ac2857 823: gt2:   ldy gtbyte ; BACKSPACE
5784 c000 824:      cpy #$00
5786 f0cc 825:      beq gt1
5788 a97e 826:      lda #$7e
578a 20d252 827:      jsr echo
578d ce2857 828:      dec gtbyte
5790 c9ff 829:      cmp #$ff
5792 d0c0 830:      bne gt1
5794 f0be 831:      beq gt1
5796          832: ;
5796          833: ; -MATH SHIT FOR CARD CONVERSION-
5796          834: gt3:
5796 a99b 835:      lda #$9b
5798 ac2857 836:      ldy gtbyte
579b c8 837:      iny
579c 992857 838:      sta gtbyte,y
579f 8c2857 839:      sty gtbyte
57a2 a928 840:      lda # low gtbyte
57a4 8dc955 841:      sta fi_s_ptr
57a7 a957 842:      lda # high gtbyte
57a9 8dca55 843:      sta fi_s_ptr+1
57ac a930 844:      lda # low qstrin
57ae 8dcb55 845:      sta fi_d_ptr
57b1 a957 846:      lda # high qstrin
57b3 8dcc55 847:      sta fi_d_ptr+1
57b6 203a56 848:      jsr toascii
```

```

57b9 a930 849:    lda # low qstrin
57bb 85f3 850:    sta $f3
57bd a957 851:    lda # high qstrin
57bf 85f4 852:    sta $f3+1
57c1 a900 853:    lda #$00
57c3 85f2 854:    sta $f2
57c5 2000d8 855:    jsr $d800
57c8 20d2d9 856:    jsr $d9d2
57cb a5d4 857:    lda $d4
57cd a6d5 858:    ldx $d4+1
57cf 60 859:    rts
57d0 860:
57d0 861:
57d0 862: pr_byte:
57d0 a200 863:    ldx #$00
57d2 864: pr_card:
57d2 85d4 865:    sta $d4
57d4 86d5 866:    stx $d4+1
57d6 a900 867:    lda #$00
57d8 85f2 868:    sta $f2
57da 20aad9 869:    jsr $d9aa
57dd 20e6d8 870:    jsr $d8e6
57e0 a000 871:    ldy #$00
57e2 872: .aprbloop:
57e2 b1f3 873:    lda ($f3),y
57e4 08 874:    php
57e5 297f 875:    and #$7f
57e7 c8 876:    iny
57e8 20d252 877:    jsr echo
57eb 28 878:    plp
57ec 10f4 879:    bpl .aprbloop
57ee 60 880:    rts
57ef 881: ;
57ef 882: ;
57ef 6d01 883: start: dc.w $016d
57f1 1004 884: stop:  dc.w $0410
57f3 885: ;
57f3 886: ;~~~~~
57f3 887: ;the code to really makes things Work with the Archiver an
57f3 888: ;
57f3 889: ;
57f3 890: ;
57f3 891: load_block:
57f3 a001 892:    ldy #$01
57f5 ad0758 893:    lda d_num
57f8 9188 894:    sta (pr_ptr),y
57fa a000 895:    ldy #$00
57fc 896: load_block1:
57fc b188 897:    lda (pr_ptr),y
57fe 990003 898:    sta $0300,y
5801 c8 899:    iny
5802 c00c 900:    cpy #$0c
5804 d0f6 901:    bne load_block1

```

```

5806 60    902:      rts
5807      903: ;
5807 00    904: d_num:  dc.b 0
5808      905: ;
5808      906: super_archiver:
5808 310273 907:      dc.b $31,2,115,0,0,0,0,0,0,2,0
5814      908: ;
5814      909: open_chip:
5814      910: ; .BYTE $31,2,$4F,0,0,0,0,0,0,$CD,$AB
5814 310275 911:      dc.b $31,2,$75,$80
5818 2058  912:      dc.w megabytes
581a 0b00  913:      dc.w $0b
581c 00    914:      dc.b 0
581d 01    915:      dc.b 1
581e 2c00  916:      dc.w $2c
5820      917: ;
5820      918: ;
5820      919: ;
5820      920: megabytes:
5820      921: ;
5820 a02cb9 922:      dc.b $a0,$2c,$b9,$2b,$10,$99,$a5,$f4
5828 8810f7 923:      dc.b $88,$10,$f7,$a0,$08,$b9,$57,$10
5830 99ea32 924:      dc.b $99,$ea,$32,$88,$10,$f7,$a0,$27
5838 b96310 925:      dc.b $b9,$63,$10,$99,$33,$33,$88,$10
5840 f7a595 926:      dc.b $f7,$a5,$95,$09,$80,$29,$fe,$85
5848 951860 927:      dc.b $95,$18,$60,$ad,$c5,$f4,$a5,$95
5850 1022a5 928:      dc.b $10,$22,$a5,$91,$0a,$66,$98,$0a
5858 26970a 929:      dc.b $26,$97,$0a,$26,$97,$0a,$66,$99
5860 a59129 930:      dc.b $a5,$91,$29,$07,$85,$91,$a0,$02
5868 d00200 931:      dc.b $d0,$02,$00,$02,$a5,$97,$f0,$04
5870 a90185 932:      dc.b $a9,$01,$85,$97,$84,$9a,$60,$ad
5878 001049 933:      dc.b $00,$10,$49,$ff,$24,$94,$30,$36
5880 8d8002 934:      dc.b $8d,$80,$02,$8d,$03,$04,$bd,$01
5888 1049ff 935:      dc.b $10,$49,$ff,$ec,$7e,$10,$f0,$10
5890 e8e486 936:      dc.b $e8,$e4,$86,$90,$e7,$ad,$80,$02
5898 09048d 937:      dc.b $09,$04,$8d,$80,$02,$4c,$0b,$f3
58a0 ad8002 938:      dc.b $ad,$80,$02,$49,$04,$8d,$80,$02
58a8 1890e5 939:      dc.b $18,$90,$e5,$83,$09,$45,$00,$80
58b0 000080 940:      dc.b $00,$00,$80
58b3      941: ;
58b3      942: ;
58b3      943: ;
58b3      944: kill_sector:
58b3 31    945:      dc.b $31 ;$300
58b4 02    946:      dc.b 2 ;$301
58b5 71    947:      dc.b "q" ;$302
58b6 80    948:      dc.b $80 ;$303
58b7 c358  949:      dc.w message ;$304
58b9 0f    950:      dc.b $0f ;$306
58ba 00    951:      dc.b 0 ;$307
58bb 8000  952:      dc.w 128 ;$308 256 for dd
58bd 6d01  953: fi_sector: dc.w d_sector ;$30A
58bf 00    954:      dc.b 0 ;$30C

```

```

58c0 00 955:      dc.b 0  ;$30D
58c1 00 956:      dc.b 0  ;$30E
58c2 00 957:      dc.b 0  ;$30F
58c3      958: ;
58c3      959: message:
58c3 546869 960:      dc.b "This is "
58cb 612074 961:      dc.b "a test o"
58d3 662074 962:      dc.b "f the ba"
58db 642077 963:      dc.b "d writte"
58e3 722070 964:      dc.b "r progra"
58eb 6d2e20 965:      dc.b "m. By St"
58f3 657068 966:      dc.b "ephen J."
58fb 204361 967:      dc.b " Carden "
5903      968: ;
5903 000000 969:      dc.b 0,0,0,0,0,0,0
590b 000000 970:      dc.b 0,0,0,0,0,0,0
5913 000000 971:      dc.b 0,0,0,0,0,0,0
591b 000000 972:      dc.b 0,0,0,0,0,0,0
5923 000000 973:      dc.b 0,0,0,0,0,0,0
592b 000000 974:      dc.b 0,0,0,0,0,0,0
5933 000000 975:      dc.b 0,0,0,0,0,0,0
593b 000000 976:      dc.b 0,0,0,0,0,0
5941 40 977:      dc.b start_fuzz
5942 7f 978:      dc.b end_fuzz
5943      979: ;
5943      980: top:
5943 205353 981:      jsr printsi
5946 9b2054 982:      dc.b $9b," This Program is for a Super A
596c 9b2020 983:      dc.b $9b," With a bit Writter"
5987 9b2046 984:      dc.b $9b," Fuzz's the sector of your cho
59aa 9b9b9b 985:      dc.b $9b,$9b,$9b,$9b
59ae 205768 986:      dc.b " What drive ",$FF
59bb 203957 987:      jsr getbyte
59be 8d0758 988:      sta d_num
59c1      989: ;
59c1      990: back_trak:
59c1      991: ;
59c1 adef57 992:      lda start
59c4 8dbd58 993:      sta fi_sector
59c7 adf057 994:      lda start+1
59ca 8dbe58 995:      sta fi_sector+1
59cd 205353 996:      jsr printsi
59d0 9b2049 997:      dc.b $9b," Insert Destination Disk", $9B,$FF
59ec 200053 998:      jsr getkey
59ef      999:      mea super_archiver,pr_ptr
59ef a908 ----:      lda #low super_archiver
59f1 8588 ----:      sta pr_ptr
59f3 a958 ----:      lda #high super_archiver
59f5 8589 ----:      sta pr_ptr + 1
59f7 20f357 1000:     jsr load_block
59fa 2059e4 1001:     jsr $e459
59fd ad0303 1002:     lda $0303
5a00 c901 1003:     cmp #$01

```

```

5a02 f003 1004:      beq top_1
5a04 4c4359 1005:      jmp top
5a07      1006: ;
5a07      1007: top_1:
5a07      1008:      mea open_chip,pr_ptr
5a07 a914 ----:      lda #low open_chip
5a09 8588 ----:      sta pr_ptr
5a0b a958 ----:      lda #high open_chip
5a0d 8589 ----:      sta pr_ptr + 1
5a0f 20f357 1009:      jsr load_block
5a12 2059e4 1010:      jsr $e459
5a15 205353 1011:      jsr printsi
5a18 2020ff 1012:      dc.b " ",$FF
5a1b ad0303 1013:      lda $0303
5a1e 20d057 1014:      jsr pr_byte
5a21      1015: kill_next:
5a21 20975a 1016:      jsr add_one
5a24      1017:      mea kill_sector,pr_ptr
5a24 a9b3 ----:      lda #low kill_sector
5a26 8588 ----:      sta pr_ptr
5a28 a958 ----:      lda #high kill_sector
5a2a 8589 ----:      sta pr_ptr + 1
5a2c 20f357 1018:      jsr load_block
5a2f ade903 1019:      lda b_code
5a32 4980 1020:      eor #$80
5a34 8d0b03 1021:      sta $030b
5a37 2059e4 1022:      jsr $e459
5a3a 98 1023:      tya
5a3b 48 1024:      pha
5a3c 205353 1025:      jsr printsi
5a3f 9b2020 1026:      dc.b $9b," ",$FF
5a45 adbd58 1027:      lda fi_sector
5a48 aebe58 1028:      ldx fi_sector+1
5a4b 20d257 1029:      jsr pr_card
5a4e      1030: ;
5a4e 205353 1031:      jsr printsi
5a51 205365 1032:      dc.b " Sector Killed ",$FF
5a61 68 1033:      pla
5a62 20d057 1034:      jsr pr_byte
5a65 20a55a 1035:      jsr check_end
5a68 c900 1036:      cmp #$00
5a6a f003 1037:      beq A22
5a6c 4c215a 1038:      jmp kill_next
5a6f      1039: ;
5a6f      1040: A22:
5a6f 205353 1041:      jsr printsi
5a72 9b9b 1042:      dc.b $9b,$9b
5a74 20444f 1043:      dc.b " DO IT AGAIN? [y,N]?", $FF
5a89 200053 1044:      jsr getkey
5a8c 201553 1045:      jsr upcase
5a8f c959 1046:      cmp #'Y'
5a91 d003 1047:      bne A11
5a93 4cc159 1048:      jmp back_trak

```

```
5a96      1049: A11:
5a96 60    1050:      rts
5a97      1051: ;
5a97      1052: add_one:
5a97 eebd58 1053:      inc fi_sector
5a9a adbd58 1054:      lda fi_sector
5a9d c900   1055:      cmp #$00
5a9f d003   1056:      bne add_one1
5aa1 eebe58 1057:      inc fi_sector+1
5aa4      1058: add_one1:
5aa4 60    1059:      rts
5aa5      1060: ;
5aa5      1061: ;
5aa5      1062: check_end:
5aa5 adbe58 1063:      lda fi_sector+1
5aa8 cdf257 1064:      cmp stop+1 ; #$04
5aab d00b   1065:      bne check_end1
5aad adbd58 1066:      lda fi_sector
5ab0 cdf157 1067:      cmp stop ; #$10
5ab3 d003   1068:      bne check_end1
5ab5 a900   1069:      lda #$00
5ab7 60    1070:      rts
5ab8      1071: check_end1:
5ab8 a901   1072:      lda #$01
5aba 60    1073:      rts
5abb      1074: ;
5abb      1075: win_end:    ds.b  0
```

End assembly: no errors