```
0000:
0000:                           .MACRO  ifFlatCable
0000:                           CMPBIM  cardType,1
0000:                           JE      %1
0000:                           .ENDM
0000:
0000: 0000              dotOn   .EQU    0                       ;for printing dots when reading disk
0000: 0001              dotOff  .EQU    1
0000:
0000:                           .PROC   CIIBOOT
0000:                   constBoot
0000: 0000                      .WORD   0                       ;length of code file in bytes
0002:                                                           ;must be zapped in by hand due to
0002:                                                           ;some stupid assemsbler restriction
0002:
0002:                   ;***************************************************************
0002:                   ;
0002:                   ; CONSTBOOT -- CONSTELLATION II BOOT FOR THE IBM/PC
0002:                   ;
0002:                   ;***************************************************************
0002: FC                        CLD
0003:
0003: 8C CB                     MOV     BX,CS                   ;set up segment registers
0005: 8E DB                     MOV     DS,BX
0007: 8E C3                     MOV     ES,BX
0009:
0009: A2 ****                   MOV     saveHost,AL             ;server host number to use
000C:
000C: FA                        CLI                             ;disable interrupts while fixing stack
000D: 33 C0                     XOR     AX,AX
000F: 8E D0                     MOV     SS,AX
0011: BC 00 7C                  MOV     SP,7C00H                ;stack just below us
0014:
0014: 06                        PUSH    ES
0015: 8E C0                     MOV     ES,AX
0017: BF 6C 00                  MOV     DI,1BH*4                ;fix up interrupt vectors  1B and 1C
001A: A1 ****                   MOV     AX,iRetAdd              ;to point to iRet on rom
001D: 8B 1E ****                MOV     BX,iRetAdd+2
0021: AB                        STOSW
0022: 93                        XCHG    AX,BX
0023: AB                        STOSW
0024: 93                        XCHG    AX,BX
0025: AB                        STOSW
0026: 93                        XCHG    AX,BX
0027: AB                        STOSW
0028: 07                        POP     ES
0029:
0029: FB                        STI                             ;enable interrupts
002A:
002A: B4 00                     MOV     AH,0                    ;identify card type
002C: FF 1E ****                CALLL   (romIo)
0030: A2 ****                   MOV     cardType,AL
0033: 88 26 ****                MOV     userHost,AH             ;save xporter host number
0037:
```

```
0037: B4 00                         MOV     AH,0                    ;init video to 80x25 BW
0039: B0 02                         MOV     AL,2
003B: CD 10                         INT     10H
003D:                       bootStart
003D: C6 06 **** 01                 MOVBIM  dots,dotOff
0042: A0 ****                       MOV     AL,saveHost
0045: A2 ****                       MOV     serverHost,AL
0048:
0048: E8 ****                       CALL    logon
004B:
004B:                               ifFlatCable $10
0052: E8 ****                       CALL    findServer
0055: 72E6                          JC      bootStart
0057:                       $10
0057: C6 06 **** 00                 MOVBIM  dots,dotOn
005C: E8 ****                       CALL    home
005F: E8 ****                       CALL    findBtDisk
0062: 72D9                          JC      bootStart
0064:
0064: E8 ****                       CALL    findMtDisk
0067:
0067:                               ifFlatCable $20
006E: E8 ****                       CALL    omniHello
0071: E9 ****                       JMP     $30
0074:                       $20
0074: E8 ****                       CALL    flatHello
0077:                       $30
0077: E8 ****                       CALL    boot
007A: EBC1                          JMP     bootStart
007C:
007C:                               .INCLUDE CONST2COT1.TEXT
007C:                       ;*****************************************************************
007C:                       ;
007C:                       ; EQUATES
007C:                       ;
007C:                       ;*****************************************************************
007C: 0007                  defAttr         .EQU    7               ;default attribute byte for video i/o
007C: 0080                  tabBit          .EQU    80H             ;tab bit for printStr
007C: 000D                  cr              .EQU    0DH             ;carriage return
007C: 000A                  lf              .EQU    0AH             ;line feed
007C: 0007                  bell            .EQU    7               ;bell
007C: 0000                  echo            .EQU    0               ;echo char when inputing from keyboard
007C: 0001                  noEcho          .EQU    1
007C: 0008                  bs              .EQU    8               ;back space
007C: 000A                  nameLen         .EQU    10              ;length of user name
007C: 0008                  passLen         .EQU    8               ;length of user password
007C: 0004                  addLen          .EQU    4               ;length of abs disk address
007C: 0032                  rdCmd           .EQU    32H             ;disk read sector command
007C: 0200                  sectSize        .EQU    512             ;size of one sector
007C: 0004                  ucsdDirBlks     .EQU    4               ;number of blocks in UCSD directory
007C: 0500                  drvrAdd         .EQU    500H            ;ucsd attached driver loaded at this address
007C: 0600                  biosAdd         .EQU    600H            ;msDos bios loaded at this address
007C:
007C: 0004                  ibmDos          .EQU    4               ;O/S type in Network User
007C: 0011                  UCSD40          .EQU    17.             ;UCSD 4.0 user type
```

```
007C: 0003                  msDosVol        .EQU    3               ;volume type in drive volume table
007C:
007C: 0E00                  xyGetUser       .EQU    0E00H           ;gotoxy(0,14)
007C: 1000                  xyGetPass       .EQU    1000H           ;gotoxy(0,16)
007C: 1200                  xyErrMsg        .EQU    1200H           ;gotoxy(0,18)
007C: 1200                  xyInvUsr        .EQU    1200H           ;   "     "
007C: 1200                  xyInvPass       .EQU    1200H           ;   "     "
007C:
007C: 000A                  mntMax          .EQU    10.             ;maximum volumes that can be mounted
007C:
007C: 0000                  true            .EQU    0
007C: 0001                  false           .EQU    1
007C:
007C:
007C:                       ;*****************************************************************
007C:                       ;
007C:                       ; CONSTANTS -- messages...etc.
007C:                       ;
007C:                       ;*****************************************************************
007C:                       constMsg
007C: 0D 0A 0A 0A 0A                        .BYTE   cr,lf,lf,lf,lf
0081: 90                                    .BYTE   16.+tabBit
0082: 2A                                    .ASCII  "*"
0083: 0D 0A                                 .BYTE   cr,lf
0085:
0085: 94                                    .BYTE   20.+tabBit
0086: 2A                                    .ASCII  "*"
0087: 0D 0A                                 .BYTE   cr,lf
0089:
0089: 9A                                    .BYTE   26.+tabBit
008A: 43 20 4F 20 52 20 56                  .ASCII  "C O R V U S   S Y S T E M S"
00A5: 0D 0A 0A                              .BYTE   cr,lf,lf
00A8:
00A8: 96                                    .BYTE   22.+tabBit
00A9: 2A 20 20 20 20 20 20                  .ASCII  "*          CONSTELLATION II"
00C3: 0D 0A 0A                              .BYTE   cr,lf,lf
00C6:
00C6: 92                                    .BYTE   tabBit+18.
00C7: 2A 20 20 20 20 2A 20                  .ASCII  "*    *             V 1.5"
00DF: 00                                    .BYTE   0
00E0:
00E0: 50 6C 65 61 73 65 20  nameMsg         .ASCII  "Please enter your name: "
00F8: 00                                    .BYTE   0
00F9:
00F9: 50 6C 65 61 73 65 20  passMsg         .ASCII  "Please enter your password: "
0115: 00                                    .BYTE   0
0116:
0116: 43 61 6E 20 6E 6F 74  badDiMsg        .ASCII  "Can not read the drive information table"
013E: 00                                    .BYTE   0
013F:
013F: 44 72 69 76 65 20 69  invDiMsg        .ASCII "Drive information table is not initialized"
0169: 00                                    .BYTE   0
016A:
016A: 49 6E 76 61 6C 69 64  invUsrMs        .ASCII "Invalid user name"
017B: 00                                    .BYTE   0
```

```
017C:
017C: 49 6E 76 61 6C 69 64  invPassMsg       .ASCII "Invalid password"
018C: 00                                     .BYTE   0
018D:
018D: 55 73 65 72 73 20 62  noBootMsg        .ASCII  "Users boot volume not found"
01A8: 00                                     .BYTE   0
01A9:
01A9: 55 73 65 72 73 20 6F  badOsMsg         .ASCII  "Users operating system is not supported"
01D0: 00                                     .BYTE   0
01D1:
01D1: 0D 0A 0A              waitMsg          .BYTE   cr,lf,lf
01D4: 50 72 65 73 73 20 61                   .ASCII  "Press any key to continue "
01EE: 07 00                                  .BYTE   bell,0
01F0:
01F0:                       ;noServerMsg      .ASCII "Users home disk server is not online "
01F0:                       ;                 .BYTE   0
01F0:
01F0: 55 43 53 44 20 70 2D  noInterpMsg      .ASCII "UCSD p-system interpreter file not found"
0218: 00                                     .BYTE   0
0219:
0219: 55 43 53 44 20 70 2D  noDrvrMsg        .ASCII "UCSD p-system driver file not found"
023C: 00                                     .BYTE   0
023D:
023D: 4D 53 2D 44 4F 53 20  noBiosMsg        .ASCII "MS-DOS bios file not found"
0257: 00                                     .BYTE   0
0258:
0258: 4D 53 2D 44 4F 53 20  noDosMsg         .ASCII "MS-DOS dos file not found"
0271: 00                                     .BYTE   0
0272:
0272: 55 6E 69 74 20 35 20  noUnit5msg       .ASCII "Unit 5 is not mounted"
0287: 00                                     .BYTE   0
0288:
0288: 55 43 53 44 20 76 6F  badUnit5Msg      .ASCII "UCSD volume on unit 5 can not be booted "
02B0: 28 66 6C 69 70 70 65                   .ASCII "(flipped directory)"
02C3: 00                                     .BYTE   0
02C4:
02C4: 50 6C 61 63 65 20 4D  putFlopInMsg     .ASCII "Place MSDOS floppy in drive A:"
02E2: 00                                     .BYTE  0
02E3:
02E3: 42 6F 6F 74 20 70 61  noBootParmMsg    .ASCII "Boot parameter file not found"
0300: 00                                     .BYTE   0
0301:
0301: 45 72 72 6F 72 20 69  readErrMsg       .ASCII "Error in reading disk block"
031C: 00                                     .BYTE   0
031D:
031D: 0D 0A 00              crLfMsg          .BYTE   0DH,0AH,0
0320:
0320: 00 00 00 08           diAdd            .BYTE   0,0,0,8       ;block address of drive info table
0324:
0324:
0324: 0F                    ucsdInterp       .BYTE 15.
0325: 49 42 4D 2E 55 43 53                   .ASCII  "IBM.UCSD.INTERP"
0334:
0334: 0F                    bootParmFile     .BYTE   15.
0335: 49 42 4D 2E 55 43 53                   .ASCII  "IBM.UCSD.BTPARM"
```

```
0344:
0344: 0F                        drvrFileName    .BYTE   15.
0345: 49 42 4D 2E 55 43 53                      .ASCII  "IBM.UCSD.DRIVER"
0354:
0354: 0E                        msDosBiosName   .BYTE 14.
0355: 49 42 4D 2E 4D 53 44                      .ASCII  "IBM.MSDOS.BIOS"
0363:
0363: 0D                        msDosName       .BYTE 13.
0364: 49 42 4D 2E 4D 53 44                      .ASCII  "IBM.MSDOS.DOS"
0371:
0371: 43 4F 52 56 55 53 20      memMark         .ASCII  "CORVUS CONST II"
0380: 000F                      memMarkLen      .EQU  15.
0380:
0380: 43 4F 52 54 41 62         dosMark         .ASCII  "CORTAb"
0386: 0006                      dosMarkLen      .EQU  6
0386:
0386: 43 6F 72 76 75 73 20      badMntMsg       .ASCII "Corvus disk driver not found in interpreter"
03B1: 00                                        .BYTE 0
03B2:
03B2: 43 6F 72 76 75 73 20      badBiosMsg      .ASCII "Corvus disk driver not found in bios"
03D6: 00                                        .BYTE 0
03D7:
03D7: 0600                      romIo           .WORD   6                       ;ROM io address
03D9: 00DF                                      .WORD   0DF00H                  ;ROM seg address
03DB:
03DB: 0900                      iRetAdd         .WORD   9                       ;address of dummy Ireturn in ROM
03DD: 00DF                                      .WORD   0DF00H
03DF:                           ;
03DF:                           ; CONSTANTS FOR encrypt/decrypt routines
03DF:                           ;
03DF: 0900                      maskMod         .WORD   9
03E1:
03E1: 1F00 9200 8700 1A00       nameBase        .WORD   31,146,135,26,99,0,7,78,125,98
03E9: 6300 0000 0700 4E00
03F1: 7D00 6200
03F5:
03F5: 0200 0100 0100 0100       nameIncr        .WORD   2,1,1,1,1,1,2,1,1,1
03FD: 0100 0100 0200 0100
0405: 0100 0100
0409:
0409: 1900 2400 7A00 9600       pswBase         .WORD   25,36,122,150,43,18,96,112
0411: 2B00 1200 6000 7000
0419:
0419: 0200 0100 FFFF 0100       pswIncr         .WORD   2,1,-1,1,2,1,-1,1
0421: 0200 0100 FFFF 0100
0429:
0429:                           ;***************************************************************
0429:                           ;
0429:                           ; RECORD LAYOUTS
0429:                           ;
0429:                           ;***************************************************************
0429:
0429:                           ;***************************************************************
0429:                           ;
0429:                           ; DRIVE INFORMATION TABLE
```

```
0429!                           ;
0429!                           ;*****************************************************************
0429!                                           .ASECT
0000!                                           .ORG    0
0000!                           diTable
0000!
0000!                           diDrive         .BLOCK  nameLen         ;drive name (unencrypted)
000A!                           diDrivePass     .BLOCK  passLen         ;drive password (encrypted)
0012!
0012!                           diServer        .BLOCK  nameLen         ;disk server name (unencrypted)
001C!                           diServPass      .BLOCK  passLen         ; "     "    password (encrypted)
0024!
0024!                           diCrvAdd        .BLOCK  addLen          ;abs addr of corvus volume
0028!                           diCrvSize       .BLOCK  addLen          ;size of corvus volume in blocks
002C!
002C!                           diBootAdd       .BLOCK  addLen          ;abs addr of cold boot table
0030!                           diNaAdd         .BLOCK  addLen          ; "   "   "  network active talbles
0034!
0034!                           diDrvInit       .BYTE                   ;drive init flag
0035!                           diRes1          .BYTE                   ;reserved.....
0036!                           diDrvNo         .WORD                   ;drive number
0038!                           diOnline        .BYTE                   ;drive online indicator
0039!                           diRes2          .BYTE                   ;reserved.....
003A!
003A!                           diNuBlks        .WORD                   ;Network User table size (blocks)
003C!                           diDvBlks        .WORD                   ;Drive Volume   "    "      "
003E!                           diDuBlks        .WORD                   ;Drive Name     "    "      "
0040!                           diDaBlks        .WORD                   ;Drive Access   "    "      "
0042!                           diSbBlks        .WORD                   ;Sys Boot       "    "      "
0044!
0044!                           diSpecial       .WORD                   ;version code
0046!
0046!                           diNuAddr        .BLOCK  addLen          ;Network User  table addr (blocks)
004A!                           diDvAddr        .BLOCK  addLen          ;Drive Volume    "    "      "
004E!                           diDuAddr        .BLOCK  addLen          ;Drive Name      "    "      "
0052!                           diDaAddr        .BLOCK  addLen          ;Drive Access    "    "      "
0056!                           diSbAddr        .BLOCK  addLen          ;System Boot     "    "      "
005A!
005A!                           diVersion       .BYTE                   ;Drive info table version
005B!                           diEnd
005B!
005B! 005B                      diRecSize       .EQU    diEnd
005B! 0001                      diRPB           .EQU    1               ;1 record per block
005B!                                           .PSECT
0429!
0429!
0429!                           ;*****************************************************************
0429!                           ;
0429!                           ; NETWORK ACTIVE TABLE
0429!                           ;
0429!                           ;*****************************************************************
0429!                                           .ASECT
005B!                                           .ORG    0
0000!                           naTable
0000!                           naHost          .BLOCK  nameLen         ;host name
```

```
000A!                       naHostNo        .BYTE                   ;host number
000B!                       naHostKind      .BYTE                   ;host kind
000C!                       naRes           .BLOCK  4               ;reserver
0010!                       naEnd
0010!
0010! 0010                  naRecSize       .EQU    naEnd
0010! 0020                  naRPB           .EQU    32              ;records per block
0010!                                       .PSECT
0429!
0429!                       ;*****************************************************************
0429!                       ;
0429!                       ; DRIVE VOLUME TABLE
0429!                       ;
0429!                       ;*****************************************************************
0429!                                       .ASECT
0010!                                       .ORG    0
0000!                       dvTable
0000!                       dvVolName       .BLOCK  nameLen         ;volume name (encrypted)
000A!                       dvStartBlk      .BLOCK  addLen          ;first block of volume (abs)
000E!                       dvEndBlk        .BLOCK  addLen          ;last block of volume (abs)
0012!                       dvVolType       .BYTE                   ;volume type
0013!                       dvWriteable     .BYTE                   ;=1 if volume is R/W
0014!                       dvGlbAccess     .BYTE                   ;=1 if volume is visable
0015!                       dvVolOff        .BYTE                   ;begining offset of volume
0016!                       dvRes1          .BLOCK  10              ;reserved
0020!                       dvEnd
0020!
0020! 0020                  dvRecSize       .EQU    dvEnd
0020! 0010                  dvRPB           .EQU    16              ;16 records per block
0020!                                       .PSECT
0429!
0429!
0429!
0429!                       ;*****************************************************************
0429!                       ;
0429!                       ; DRIVE ACCESS TABLE
0429!                       ;
0429!                       ;*****************************************************************
0429!                                       .ASECT
0020!                                       .ORG    0
0000!                       daTable
0000!                       daUserid        .WORD                   ;user identifier
0002!                       daVolAddr       .BLOCK  addLen          ;start block of volume
0006!                       daMntInfo       .BYTE                   ;OS dependent mount unit
0007!                       daReadBoot      .BYTE                   ;read only and boot flag
0008!
0008! 0010                  daROflag        .EQU    10H             ;bit for read only flag
0008! 0001                  daBootflat      .EQU    01H             ;bit for users boot volume
0008!                       daEnd
0008!
0008! 0008                  daRecSize       .EQU    daEnd
0008! 0040                  daRPB           .EQU    64              ;64 records per block
0008!                                       .PSECT
0429!
0429!                       ;*****************************************************************
```

```
0429!                       ;
0429!                       ; NETWORK USER TABLE
0429!                       ;
0429!                       ;*************************************************************************
0429!                                   .ASECT
0008!                                   .ORG    0
0000!                       nuTable
0000!                       nuUser          .BLOCK  nameLen         ;user name (encrypted)
000A!                       nuUserPass      .BLOCK  passLen         ;user password (encrypted)
0012!                       nuServer        .BLOCK  nameLen         ;home disk server name (unencrypted)
001C!                       nuOpsys         .WORD                   ;operating system type
001E!                       nuHostType      .BYTE                   ;host station type
001F!                       nuRes1          .BYTE                   ;reserver
0020!                       nuEnd
0020!
0020! 0020                  nuRecSize       .EQU    nuEnd
0020! 0010                  nuRPB           .EQU    16              ;16 records per block
0020!                                   .PSECT
0429!
0429!                       ;*************************************************************************
0429!                       ;
0429!                       ; DRIVE USER TABLE
0429!                       ;
0429!                       ;*************************************************************************
0429!                                   .ASECT
0020!                                   .ORG    0
0000!                       duTable
0000!                       duUser          .BLOCK  nameLen         ;user name (encrypted)
000A!                       duPass          .BLOCK  passLen         ;user password (encrypted)
0012!                       duUserId        .WORD                   ;user identifier
0014!                       duMounted       .WORD                   ;# of user volumes mounted
0016!                       duAccess        .WORD                   ;# of vols in user access
0018!                       duBootInfo      .WORD                   ;currently unused
001A!                       duRes1          .BLOCK  6               ;reserved
0020!                       duEnd
0020!
0020! 0020                  duRecSize       .EQU    duEnd
0020! 0010                  duRPB           .EQU    16              ;16 entries per block
0020!                                   .PSECT
0429!
0429!
0429!                       ;*************************************************************************
0429!                       ;
0429!                       ; UCSD PASCAL DISK DIRECTORY ENTRY
0429!                       ;
0429!                       ;*************************************************************************
0429!                                   .ASECT
0020!                                   .ORG    0
0000!                       ucsdDir
0000!
0000!                       ucsd1stBlk      .WORD                   ;first block in file
0002!                       ucsdLastBlk     .WORD                   ;last block in file
0004!                                   .BLOCK  2               ;of no interest to us
0006!                       ucsdName        .BLOCK  16.             ;name of file (1st byte is len)
0016!                                   .BLOCK  4               ;of no interest
```

```
001A!
001A!                           ucsdEnd
001A! 001A                      ucsdDirSize     .EQU    ucsdEnd
001A!
001A! 0010                      ucsdNumFiles    .EQU    ucsdDir+10H     ;only in entry zero
001A!
001A!                           .PSECT
0429!                           ;*****************************************************************
0429!                           ;
0429!                           ; UCSD MOUNT TABLE
0429!                           ;
0429!                           ;*****************************************************************
0429!                           .ASECT
001A!                           .ORG    0
0000!                           mntTbl
0000!                           mntDrvWp        .BYTE                   ;drive = left 4 bits write prot = right
0001!                                                                   ;             4 bits
0001!                           mntDskSrvr      .BYTE                   ;disk server network address
0002!                           mntBlock        .BLOCK  4               ;4 byte extended block address
0006!                           mntNumBlks      .WORD                   ;num blocks in volume
0008!
0008!                           mntTblEnd
0008! 0008                      mntRecSize      .EQU    mntTblEnd
0008!                           .PSECT
0429!
0429!                           ;-----------------------------------------------------------------
0429!                           ;
0429!                           ; TABLES IN MSDOS BIOS THAT MUST BE FILLED IN MY THE BOOTING PROCESS
0429!                           ;
0429!                           ;
0429!                           ;DRIVE MOUNT TABLE -- INDEXED BY DOS LOGICAL UNIT 0..max logical units
0429!                           ;
0429!                           ;   rWflag         : 0..1          1=readOnly 0=readWrite
0429!                           ;   kindFlag       : 0..1          0=corvus 1=floppy
0429!                           ;   unused         : 0..1
0429!                           ;   physUnit       : 0..31         if floppy this is physical drive (1-2)
0429!                           ;                                   if corvus this is index into volume table
0429!                           ;                                   0=not mounted
0429!                           ;
0429!                           ;VOLUME TABLE -- indexed by PhysUnit from DRIVE MOUNT TABLE
0429!                           ;
0429!                           ;   loAddByte      : 0..255        lo byte of 20 bit volume address
0429!                           ;   midAddByte     : 0..255        mid byte   "    "    "       "
0429!                           ;   hiAddByte      : 0..255        hi   "     "    "    "       "
0429!                           ;   driveNum       : 0..255        drive number (corvus drive number)
0429!                           ;   server         : 0..255        disk server network address
0429!                           ;
0429!                           ;DISK CONFIGURATION TABLES
0429!                           ;
0429!                           ;   numEntries     : 0..255        number of entries used in configuration table
0429!                           ;....
0429!                           ;   logicalUnit    : 0..255        DOS logical unit 0..max logical units
0429!                           ;   parmPoint      : 0..FFFFH      pointer to disk parameter table
0429!                           ;
0429!                           ;   (* to allow mounting and remounting there should be two entries for
```

```
0429!                          ;      each dos logical unit. If it is a floppy the first should point to
0429!                          ;      the single sided entry and the second should point to the double
0429!                          ;      sided entry *)
0429!                          ;
0429!                          ;DISK PARAMETER TABLE
0429!                          ;
0429!                          ;   sectSize       : 0..FFFFH      sector size in bytes
0429!                          ;   sectPerBlk     : 0..255        sectors per block
0429!                          ;   resvSect       : 0..FFFFH      reserved sectors
0429!                          ;   numFats        : 0..255        number of FAT's
0429!                          ;   dirEntries     : 0..FFFFH      number of directory entries
0429!                          ;   numSects       : 0..FFFFH      number of sectors/disk
0429!                          ;
0429!                          ;LOGICAL UNIT MAPPING TABLE -- INDEXED BY DOS LOGICAL UNIT 0..max logical units
0429!                          ;
0429!                          ;   index          : 0..255        index into disk configuration tables
0429!                          ;
0429!                          ;-------------------------------------------------------------------------------
0429!
0429!
0429!                          ;*******************************************************************
0429!                          ;
0429!                          ; RAM VARIABLES
0429!                          ;
0429!                          ;*******************************************************************
0429! 00 00 00 00 00 00 00    diRec           .BLOCK  diRecSize       ;drive info table
0484! 00 00 00 00 00 00 00    nuRec           .BLOCK  nuRecSize       ;network user table
04A4! 00 00 00 00 00 00 00    daREc           .BLOCK  daRecSize       ;drive access table
04AC! 00 00 00 00 00 00 00    duRec           .BLOCK  duRecSize       ;drive user table
04CC! 00 00 00 00 00 00 00    dvRec           .BLOCK  dvRecSize       ;drive volume table
04EC! 00 00 00 00 00 00 00    dvBtRec         .BLOCK  dvRecSize       ; "      "     "
050C!
050C!                          ;
050C!                          ; THE FOLLOWING ARE USED BY THE UCSD BOOT PROCESS
050C!                          ;
050C!
050C! ****                    mntDvPoint      .WORD   mntDvTab        ;pointer to next entry in mount tables
050E! ****                    mntDaPoint      .WORD   mntDaTab
0510! ****                    mntDrvPoint     .WORD   mntDrvTab
0512! 0000                    mntCount        .WORD   0               ;count of entries in mnt tables
0514! 0000                    mntTabAdd       .WORD   0
0516! 0000                    mntTabSeg       .WORD   0
0518! 00 00 00 00 00 00 00    hold5dvRec      .BLOCK  dvRecSize       ;unit 9 drive volume rec
0538! 00                      hold5drive      .BYTE   0               ;unit 9 drive
0539! 0000                    interpAdd       .WORD   0
053B! 0000 5000               drvrInitAdd     .WORD   0,drvrAdd/10H   ;for long call to driver initalize
053F!
053F!                          ;
053F!                          ; THE FOLLOWING ARE USED BY THE MSDOS BOOT PROCESS
053F!                          ;
053F! 0000                    dosSeg          .WORD   0               ;segment address of MS/DOS bios
0541! 0000                    biosLen         .WORD   0               ;length in bytes of loaded MS/DOS bios
0543! 0000                    configAdd       .WORD   0               ;address of DOS bios configuration table
0545! 0000                    sConfigAdd      .WORD   0               ;address of DOS bios configuration table
0547! 0000                    fparmAdd        .WORD   0               ;address of DOS bios floppy disk paramters
```

```
0549: 0000                    cParmAdd        .WORD   0               ;address of DOS bios Corvus  "     "
054B: 0000                    cParmCadd       .WORD   0               ;address of # entries used in above table
054D: 0000                    volAdd          .WORD   0               ;address of DOS bios volume table
054F: 0000                    mapAdd          .WORD   0               ;address of DOS bios logical unit map table
0551: 0000                    logUnit         .WORD   0               ;logical unit number
0553: 0000                    mountCount      .WORD   0               ;work for DOS mounting
0555: 0000                    flopCount       .WORD   0               ;number of floppys mounted
0557: 0000                    mapCount        .WORD   0               ;index into config table
0559: 0000                    maxFlop         .WORD   0               ;actual number of floppies in system
055B: 00                      aMounted        .BYTE   0               ;true if unit a is mounted
055C:
055C:
055C: 0000                    temp            .WORD   0               ;temp word may be used by any subroutine
055E: 0000                    numBlocks       .WORD   0               ;temp for misc routines
0560: 0000                    holdNumBlocks   .WORD   0
0562: 0000                    holdNumRead     .WORD   0               ;used by read UCSD file routine
0564:
0564: 0000                    getStrLen       .WORD   0               ;used by getStr routine
0566: 00                      getStrEcho      .BYTE   0               ; "    "   "      "
0567:
0567: 00 00 00 00 00 00 00    userName        .BLOCK  nameLen
0571: 00 00 00 00 00 00 00    userPassword    .BLOCK  passLen
0579: 00                      userHost        .BYTE   0
057A:
057A: 00 00 00 00 00 00 00    cryptName       .BLOCK  nameLen          ;encrypted name
0584: 00 00 00 00 00 00 00    cryptPassword   .BLOCK  passLen          ;encrypted password
058C:
058C: 00 00 00 00 00 00 00    tempDName       .BLOCK  nameLen          ;temps
0596: 00 00 00 00 00 00 00    tempEName       .BLOCK  nameLen
05A0:
05A0: 00                      drvsOnline      .BYTE   0               ;number of drives on system
05A1:
05A1: 00                      cardType        .BYTE   0
05A2: 00                      dots            .BYTE   0
05A3:
05A3: 00                      serverHost      .BYTE   0               ;server host number
05A4: 00                      saveHost        .BYTE   0
05A5: 00 00 00 00 00 00 00    serverName      .BLOCK  nameLen
05AF: 00                      drive           .BYTE   0               ;current drive number
05B0: 00 00 00 00             block           .BLOCK  addLen          ;current block to read
05B4: 00 00 00 00             holdBlock       .BLOCK  addLen
05B8: 00                      ioRetCode       .BYTE   0               ;return code from read
05B9: 00                      dirIsFlipped    .BYTE   0
05BA: 00                      errOk           .BYTE   0               ;<> 0 if read errors are ok
05BB: 00 00 00 00             drvCmd          .BLOCK  4               ;4 byte disk command
05BF:
05BF: 0000                    retLen          .WORD                   ;used by block read routines
05C1:
05C1:                         ;
05C1:                         ; THE FOLLOWING ARE CONSTELLATION II SERVER COMMANDS
05C1:                         ;
05C1:
05C1: 01 FE 03 00 00          whereAreYou     .BYTE   1,0FEH,3,0,0    ;where are you constellation packet
05C6: 00                      waySource       .BYTE   0               ;host number of sender
05C7: 00 FF                                   .BYTE   0,0FFH          ;anybody respond
```

```
05C9: 00 00 00 00 00 00 00  wayName         .BLOCK  nameLen          ;server we are looking for
05D3:
05D3: 00 00 00 00 00        myIdIs          .BLOCK  5               ;we only care about host number
05D8: 00                    miiSource       .BYTE   0               ;host number of responder
05D9: 00 00                                 .BLOCK  2
05DB: 00 00 00 00 00 00 00                  .BLOCK  nameLen
05E5:
05E5: 01 FE 00 00 00        hello           .BYTE   1,0FEH,0,0,0    ;hello on network message
05EA: 00                    helloSource     .BYTE   0               ;our host number
05EB: 00                                    .BYTE   0
05EC: 00                    helloType       .BYTE   0               ;our station type
05ED: 00 00 00 00 00 00 00  helloName       .BLOCK  nameLen
05F7:
05F7: 34 03                 addActive       .BYTE   34H,3           ;add active disk cmd
05F9: 00 00 00 00 00 00 00  addActName      .BLOCK  nameLen         ;our user name
0603: 00 00 00 00 00 00                     .BLOCK  6               ;not used
0609:
0609:
0609:
0609:
0609:                       ;*****************************************************************
0609:                       ;
0609:                       ; SAVE_REGS -- saves all 8086 registers on the stack
0609:                       ;
0609:                       ;*****************************************************************
0609:                       saveRegs
0609: 8F 06 5C05                    POP     temp                    ;save return address
060D: 50                            PUSH    AX
060E: 53                            PUSH    BX
060F: 51                            PUSH    CX
0610: 52                            PUSH    DX
0611: 56                            PUSH    SI
0612: 57                            PUSH    DI
0613: FF 36 5C05                    PUSH    temp                    ;restore return address
0617: C3                            RET
0618:
0618:                       ;*****************************************************************
0618:                       ;
0618:                       ; REST_REGS -- restores all 8086 registers from the stack
0618:                       ;
0618:                       ;*****************************************************************
0618:                       restRegs
0618: 8F 06 5C05                    POP     temp                    ;save return address
061C: 5F                            POP     DI
061D: 5E                            POP     SI
061E: 5A                            POP     DX
061F: 59                            POP     CX
0620: 5B                            POP     BX
0621: 58                            POP     AX
0622: FF 36 5C05                    PUSH    temp                    ;restore return address
0626: C3                            RET
0627:
0627:                       ;*****************************************************************
0627:                       ;
```

```
0627:                           ; HOME -- clears the screen and positions cursor at 0,0
0627:                           ;
0627:                           ;*************************************************************
0627:                           home
0627: E8 DFFF                           CALL    saveRegs
062A: B4 06                             MOV     AH,6                    ;specify scroll i/o call
062C: B0 00                             MOV     AL,0                    ;clear window
062E: B9 00 00                          MOV     CX,0                    ;upper left corner
0631: BA 4F 18                          MOV     DX,184FH                ;lower right corner 24,79 (18H,4FH)
0634: B7 07                             MOV     BH,defAttr              ;attribute byte of white on black
0636: CD 10                             INT     10H                     ;call i/o routines
0638:
0638: B4 02                             MOV     AH,2                    ;specify set cursor position i/o call
063A: 33 D2                             XOR     DX,DX                   ;set cursor to 0,0
063C: B7 00                             MOV     BH,0                    ;use screen page zero
063E: CD 10                             INT     10H                     ;call i/o routines
0640:
0640: E8 D5FF                           CALL    restRegs                ;restore all registers
0643: C3                                RET
0644:
0644:                           ;*************************************************************
0644:                           ;
0644:                           ; GOTOXY -- positions cursor at (DH,DL)
0644:                           ;
0644:                           ;*************************************************************
0644:                           gotoxy
0644: E8 C2FF                           CALL    saveRegs                ;save all registers on stack
0647: B4 02                             MOV     AH,2                    ;specify set cursor position i/o call
0649: B7 00                             MOV     BH,0                    ;use screen page zero
064B: CD 10                             INT     10H                     ;call i/o routines
064D:
064D: E8 C8FF                           CALL    restRegs                ;restore registers
0650: C3                                RET
0651:
0651:
0651:                           ;*************************************************************
0651:                           ;
0651:                           ; PRINT_STR -- prints the string pointed to by SI. String is
0651:                           ;              terminated by 0H. Backspace, car ret, bell, and
0651:                           ;              line feed are handled. If the MSB of a byte
0651:                           ;              is on, the lower 7 bits are used as a horiz tab
0651:                           ;              value.
0651:                           ;
0651:                           ;*************************************************************
0651:                           printStr
0651: E8 B5FF                           CALL    saveRegs                ;save regs on stack
0654:                           $10
0654: AC                                LODSB                           ;load one byte
0655: 3C 00                             CMP     AL,0                    ;is it end of string?
0657: 74**                              JE      $99                     ; yes exit
0659: A8 80                             TEST    AL,80H                  ;is it a tab?
065B: 75**                              JNZ     $30                     ; yes, process it
065D:
065D: B4 0E                             MOV     AH,14                   ;specify write tty mode char
065F: B7 00                             MOV     BH,0                    ;use screen page zero
```

```
0661: B3 07                         MOV     BL,defAttr              ;use white on black attribute byte
0663: CD 10                         INT     10H                     ;call i/o routines
0665:
0665: EBED                          JMP     $10                     ;get next byte
0667:                       $30
0667: 50                            PUSH    AX                      ;save tab count
0668: B4 03                         MOV     AH,3                    ;specify read cursor
066A: B7 00                         MOV     BH,0                    ;use screen page zero
066C: CD 10                         INT     10H                     ;call i/o routines
066E:
066E: 58                            POP     AX                      ;restore tab count
066F: 24 7F                         AND     AL,7FH                  ;mask off msb
0671: 02 D0                         ADD     DL,AL                   ;add tab value to current col
0673: B4 02                         MOV     AH,2                    ;specify set cursor position
0675: CD 10                         INT     10H                     ;call i/o routines
0677: EBDB                          JMP     $10                     ;get next char
0679:                       $99
0679: E8 9CFF                       CALL    restRegs                ;restore registers
067C: C3                            RET
067D:
067D:                       ;*****************************************************************
067D:                       ;
067D:                       ; GETSTR -- get string from keyboard
067D:                       ;           SI is address of string
067D:                       ;           CX is length of string
067D:                       ;           BL is echo char or print '.'
067D:                       ;           string will be padded with blanks
067D:                       ;
067D:                       ;*****************************************************************
067D:                       GETSTR
067D: E8 B9FF                       CALL    saveRegs                ;save all registers
0680: 8B FE                         MOV     DI,SI                   ;set up add for fill with blanks
0682: 89 0E 6405                    MOV     getStrLen,CX            ;put count in memory word
0686: 88 1E 6605                    MOV     getStrEcho,BL           ;put echo in memory byte
068A: B0 20                         MOV     AL,20H                  ;space
068C: F3 AA                  REP    STOSB                           ;fill with blanks
068E: 8B FE                         MOV     DI,SI                   ;point at start of string
0690: 33 C9                         XOR     CX,CX                   ;zero CX for count
0692:                       $10
0692: B4 00                         MOV     AH,0                    ;specify read keyboard
0694: CD 16                         INT     16H                     ;call keyboard i/o routines
0696:
0696: 3C 0D                         CMP     AL,cr                   ;is it carriage return
0698: 74**                          JE      $99                     ;yes all done
069A:
069A: 3C 08                         CMP     AL,bs                   ;is it a backspace?
069C: 75**                          JNE     $15                     ; no
069E: 83 F9 00                      CMP     CX,0                    ;is count = zero
06A1: 74EF                          JE      $10                     ; yes, ignore backspace
06A3:
06A3: B4 0E                         MOV     AH,14                   ;specify tty write(backspace)
06A5: B7 00                         MOV     BH,0                    ;use screen page zero
06A7: CD 10                         INT     10H                     ;call i/o routines
06A9:
06A9: 51                            PUSH    CX
```

```
06AA: B4 0A                         MOV     AH,10                   ;specify write char at curr position
06AC: B0 20                         MOV     AL,20H                  ;write a blank
06AE: B9 01 00                      MOV     CX,1                    ;write one char
06B1: CD 10                         INT     10H                     ;call i/o routines
06B3: 59                            POP     CX
06B4:
06B4: 4F                            DEC     DI                      ;set address pointers back
06B5: 49                            DEC     CX                      ;dec counter
06B6: B0 20                         MOV     AL,20H                  ;move blank to backspaced area
06B8: 88 45 00                      MOV     (DI+0),AL
06BB: EBD5                          JMP     $10                     ;get another byte
06BD:                       $15
06BD: 3B 0E 6405                    CMP     CX,getStrLen            ;is string full
06C1: 74CF                          JE      $10                     ; yes wait for bs or cr
06C3:
06C3: 3C 20                         CMP     AL,20H                  ;is char < space?
06C5: 72CB                          JB      $10                     ; yes get another
06C7:
06C7: 3C 7E                         CMP     AL,7EH                  ;is char above 7E?
06C9: 77C7                          JA      $10                     ; yes, get another
06CB:
06CB: 3C 61                         CMP     AL,61H                  ;is it less than 'a'?
06CD: 72**                          JB      $17                     ;yes
06CF:
06CF: 3C 7A                         CMP     AL,7AH                  ;is it greater than 'z'?
06D1: 77**                          JA      $17
06D3:
06D3: 2C 20                         SUB     AL,20H                  ;make it upper case
06D5:                       $17
06D5: AA                            STOSB                           ;save char
06D6: 41                            INC     CX                      ;add one to count
06D7: 80 3E 6605 00                 CMPBIM  getStrEcho,echo         ;are we susposed to echo
06DC: 74**                          JE      $20                     ; yes
06DE: B0 2E                         MOV     AL,2EH                  ;print a dot instead
06E0:                       $20
06E0: B4 0E                         MOV     AH,14                   ;specify write one char
06E2: B7 00                         MOV     BH,0                    ;use screen page zero
06E4: CD 10                         INT     10H                     ;call i/o routines
06E6: EBAA                          JMP     $10                     ;get another character
06E8:                       $99
06E8: E8 2DFF                       CALL    restRegs                ;restore registers
06EB: C3                            RET
06EC:
06EC:
06EC:                       ;*************************************************************
06EC:                       ;
06EC:                       ; PRINT_AND_WAIT -- clears the screen , print the message in SI
06EC:                       ;                   and waits for a keypress
06EC:                       ;
06EC:                       ;*************************************************************
06EC:                       printAndWait
06EC: E8 38FF                       CALL    home
06EF: E8 5FFF                       CALL    printStr
06F2: E8 ****                       CALL    waitForKey
06F5: C3                            RET
```

```
06F6:                   ;*************************************************************
06F6:                   ;
06F6:                   ; PUTLOGO -- clears screen and displays const II logo
06F6:                   ;
06F6:                   ;*************************************************************
06F6:                   PUTLOGO
06F6: E8 2EFF                   CALL    home
06F9: BE 7C00                   LEA     SI,constMsg
06FC: E8 52FF                   CALL    printStr
06FF: C3                        RET
0700:
0700:                   ;*************************************************************
0700:                   ;
0700:                   ; GETUSERID -- prompt the user for his user id
0700:                   ;
0700:                   ;*************************************************************
0700:                   GETUSERID
0700: BA 00 0E                  MOV     DX,xyGetUser
0703: E8 3EFF                   CALL    GOTOXY
0706: BE E000                   LEA     SI,nameMsg              ;"Please enter your name..."
0709: E8 45FF                   CALL    printStr                ;print it on screen
070C:
070C: BE 6705                   LEA     SI,userName             ;for get string
070F: B9 0A 00                  MOV     CX,nameLen              ;length of name field
0712: B3 00                     MOV     BL,echo                 ;echo chars when we get them
0714: E8 66FF                   CALL    getStr
0717: C3                        RET
0718:                   ;*************************************************************
0718:                   ;
0718:                   ;  GETPASSWORD -- prompt the user for his password
0718:                   ;
0718:                   ;*************************************************************
0718:                   GETPASSWORD
0718: E8 EEFE                   CALL    saveRegs
071B: BA 00 10                  MOV     DX,xyGetPass            ;gotoxy(0,14)
071E: E8 23FF                   CALL    GOTOXY
0721: BE F900                   LEA     SI,passMsg              ;"Please enter your password "
0724: E8 2AFF                   CALL    printStr
0727:
0727: BE 7105                   LEA     SI,userPassword         ;for get string
072A: B9 08 00                  MOV     CX,passLen              ;length of password
072D: B3 01                     MOV     BL,noEcho               ;only print dots
072F: E8 4BFF                   CALL    getStr
0732: E8 E3FE                   CALL    restRegs
0735: C3                        RET
0736:                   ;*************************************************************
0736:                   ;
0736:                   ; ENCRYPT_NAME -- encrypts userName into cryptName
0736:                   ;
0736:                   ;*************************************************************
0736:                   encryptName
0736:
0736: E8 D0FE                   CALL    saveRegs                ;save registers onto stack
0739: 33 DB                     XOR     BX,BX                   ;bx = index into userName
```

```
073B: 33 FF                         XOR     DI,DI                   ;bx+di = index into crypt tables
073D: 33 F6                         XOR     SI,SI                   ;si = mask value
073F:                       $10
073F: 8A 8F 6705                    MOV     CL,userName(BX)
0743: 32 ED                         XOR     CH,CH
0745: 83 E9 20                      SUB     CX,32
0748:
0748: 8B C1                         MOV     AX,CX                   ;move to ax for multiply
074A: F7 AD F503                    IMUL    nameIncr(DI)
074E: 8B C8                         MOV     CX,AX                   ;back to CX
0750:
0750: 03 8D E103                    ADD     CX,nameBase(DI)
0754: 03 CE                         ADD     CX,SI                   ;add mask value
0756:
0756: 88 8F 7A05                    MOV     cryptName(BX),CL        ;save in crypted form
075A:
075A: 83 FB 00                      CMP     BX,0                    ;is it 1st char?
075D: 75**                          JNE     $15
075F:
075F: 8A 8F 6705                    MOV     CL,userName(BX)         ;make new maskVal
0763: 32 ED                         XOR     CH,CH
0765:
0765: 83 E9 20                      SUB     CX,32
0768: 8B C1                         MOV     AX,CX                   ;to AX for divide
076A: 99                            CWD                             ;entend sign to DX
076B: F7 3E DF03                    IDIV    maskMod
076F: 8B F2                         MOV     SI,DX                   ;move remainder into maskValue
0771:
0771:                       $15
0771: 43                            INC     BX                      ;point to next char
0772: 47                            INC     DI
0773: 47                            INC     DI
0774: 83 FB 0A                      CMP     BX,nameLen              ;are we finished?
0777: 75C6                          JNE     $10                     ;no do next char
0779:
0779: E8 9CFE                       CALL    restRegs                ;restore registers from stack
077C: C3                            RET
077D:                       ;*****************************************************************
077D:                       ;
077D:                       ; DECRYPT_NAME -- decrypts tempEName into tempDName
077D:                       ;
077D:                       ;*****************************************************************
077D:                       deCryptName
077D:
077D: E8 89FE                       CALL    saveRegs                ;save registers onto stack
0780: 33 DB                         XOR     BX,BX                   ;bx = index into userName
0782: 33 FF                         XOR     DI,DI                   ;bx+di = index into crypt tables
0784: 33 F6                         XOR     SI,SI                   ;si = mask value
0786:                       $10
0786: 8A 8F 9605                    MOV     CL,tempEName(BX)
078A: 32 ED                         XOR     CH,CH
078C:
078C: 2B CE                         SUB     CX,SI                   ;minus mask value
078E: 2B 8D E103                    SUB     CX,nameBase(DI)
0792:
```

```
07B2: 8B C1                         MOV     AX,CX                   ;to AX for DIV
07B4: 99                            CWD
07B5: F7 BD F503                    IDIV    nameIncr(DI)
07B9: 8B C8                         MOV     CX,AX                   ;quotient back to CX
07BB:
07BB: 83 C1 20                      ADD     CX,32
07BE:
07BE: 88 8F 8C05                    MOV     tempDName(BX),CL        ;save in crypted form
07C2:
07C2: 83 FB 00                      CMP     BX,0                    ;is it 1st char?
07C5: 75**                          JNE     $15
07C7:
07C7: 83 E9 20                      SUB     CX,32
07CA: 8B C1                         MOV     AX,CX                   ;to AX for divide
07CC: 99                            CWD                             ;entend sign to DX
07CD: F7 3E DF03                    IDIV    maskMod
07D1: 8B F2                         MOV     SI,DX                   ;move remainder into maskValue
07D3:
07D3:                       $15
07D3: 43                            INC     BX                      ;point to next char
07D4: 47                            INC     DI
07D5: 47                            INC     DI
07D6: 83 FB 0A                      CMP     BX,nameLen              ;are we finished?
07D9: 75CB                          JNE     $10                     ;no do next char
07DB:
07DB: E8 5AFE                       CALL    restRegs                ;restore registers from stack
07DE: C3                            RET
07DF:
07DF:
07DF:                       ;*****************************************************************
07DF:                       ;
07DF:                       ; ENCRYPT_PASSWORD -- encrypts userPassword into cryptPassword
07DF:                       ;
07DF:                       ;*****************************************************************
07DF:                       encryptPassword
07DF:
07DF: E8 47FE                       CALL    saveRegs                ;save registers onto stack
07E2: 33 DB                         XOR     BX,BX                   ;bx = index into userPassword
07E4: 33 FF                         XOR     DI,DI                   ;di = index into crypt tables
07E6: 33 F6                         XOR     SI,SI                   ;si = mask value
07E8:                       $10
07E8: 8A 8F 7105                    MOV     CL,userPassword(BX)
07EC: 32 ED                         XOR     CH,CH
07EE: 83 E9 20                      SUB     CX,32
07F1:
07F1: 8B C1                         MOV     AX,CX                   ;move to ax for multiply
07F3: F7 AD 1904                    IMUL    pswIncr(DI)
07F7: 8B C8                         MOV     CX,AX                   ;back to CX
07F9:
07F9: 03 8D 0904                    ADD     CX,pswBase(DI)
07FD: 03 CE                         ADD     CX,SI                   ;add mask value
07FF:
07FF: 88 8F 8405                    MOV     cryptPassword(BX),CL    ;save in crypted form
0803:
0803: 83 FB 00                      CMP     BX,0                    ;is it 1st char?
```

```
07E6: 75**                          JNE     $15
07E8:
07E8: 8A 8F 7105                    MOV     CL,userPassword(BX)     ;make new maskVal
07EC: 32 ED                         XOR     CH,CH
07EE:
07EE: 83 E9 20                      SUB     CX,32
07F1: 8B C1                         MOV     AX,CX                   ;to AX for divide
07F3: 99                            CWD                             ;entend sign to DX
07F4: F7 3E DF03                    IDIV    maskMod
07F8: 8B F2                         MOV     SI,DX                   ;move remainder into maskValue
07FA:
07FA:                       $15
07FA: 43                            INC     BX                      ;point to next char
07FB: 47                            INC     DI
07FC: 47                            INC     DI
07FD: 83 FB 08                      CMP     BX,passLen              ;are we finished?
0800: 75C6                          JNE     $10                     ;no do next char
0802:
0802: E8 13FE                       CALL    restRegs                ;restore registers from stack
0805: C3                            RET
0806:                       ;*************************************************************
0806:                       ;
0806:                       ; FLIP -- flip msb and lsb of word pointed to by BX
0806:                       ;
0806:                       ;*************************************************************
0806:                       FLIP
0806: 8B 07                         MOV     AX,(BX)
0808: 86 C4                         XCHG    AL,AH
080A: 89 07                         MOV     (BX),AX
080C: C3                            RET
080D:                       ;*************************************************************
080D:                       ;
080D:                       ; FLIP DRIVE INFO TABLE (address of table in SI)
080D:                       ;
080D:                       ;*************************************************************
080D:                       FLIP_DI
080D: E8 F9FD                       CALL    saveRegs
0810:
0810: 8D 5C 36                      LEA     BX,(SI+diDrvNo)
0813: E8 F0FF                       CALL    flip
0816:
0816: 8D 5C 3A                      LEA     BX,(SI+diNuBlks)
0819: E8 EAFF                       CALL    flip
081C:
081C: 8D 5C 3C                      LEA     BX,(SI+diDvBlks)
081F: E8 E4FF                       CALL    flip
0822:
0822: 8D 5C 3E                      LEA     BX,(SI+diDuBlks)
0825: E8 DEFF                       CALL    flip
0828:
0828: 8D 5C 40                      LEA     BX,(SI+diDaBlks)
082B: E8 D8FF                       CALL    flip
082E:
082E: 8D 5C 42                      LEA     BX,(SI+diSbBlks)
```

```
0831: E8 D2FF                   CALL    flip
0834:
0834: 8D 5C 44                  LEA     BX,(SI+diSpecial)
0837: E8 CCFF                   CALL    flip
083A:
083A: E8 DBFD                   CALL    restRegs
083D: C3                        RET
083E:
083E:                   ;*****************************************************************
083E:                   ;
083E:                   ; FLIP DRIVE ACCESS TABLE (address in SI)
083E:                   ;
083E:                   ;*****************************************************************
083E:                   FLIP_DA
083E: E8 C8FD                   CALL    saveRegs
0841: B9 40 00                  MOV     CX,daRPB                ;records per block in CX
0844:                   $10
0844: 8D 5C 00                  LEA     BX,(SI+daUserId)
0847: E8 BCFF                   CALL    flip
084A:
084A: 83 C6 08                  ADD     SI,daRecSize
084D: E2F5                      LOOP    $10
084F:
084F: E8 C6FD                   CALL    restRegs
0852: C3                        RET
0853:
0853:                   ;*****************************************************************
0853:                   ;
0853:                   ; FLIP NETWORK USER TABLE (address in SI)
0853:                   ;
0853:                   ;*****************************************************************
0853:                   FLIP_NU
0853: E8 B3FD                   CALL    saveRegs
0856: B9 10 00                  MOV     CX,nuRPB
0859:                   $10
0859: 8D 5C 1C                  LEA     BX,(SI+nuOpSys)
085C: E8 A7FF                   CALL    flip
085F:
085F: 83 C6 20                  ADD     SI,nuRecSize
0862: E2F5                      LOOP    $10
0864:
0864: E8 B1FD                   CALL    restRegs
0867: C3                        RET
0868:
0868:                   ;*****************************************************************
0868:                   ;
0868:                   ; FLIP DRIVE USER TABLE (address in SI)
0868:                   ;
0868:                   ;*****************************************************************
0868:                   FLIP_DU
0868: E8 9EFD                   CALL    saveRegs
086B: B9 10 00                  MOV     CX,duRPB                ;records per block
086E:                   $10
086E: 8D 5C 12                  LEA     BX,(SI+duUserId)
0871: E8 92FF                   CALL    flip
```

```
0874:
0874: 8D 5C 14                  LEA     BX,(SI+duMounted)
0877: E8 8CFF                   CALL    flip
087A:
087A: 8D 5C 16                  LEA     BX,(SI+duAccess)
087D: E8 86FF                   CALL    flip
0880:
0880: 8D 5C 18                  LEA     BX,(SI+duBootInfo)
0883: E8 80FF                   CALL    flip
0886:
0886: 83 C6 20                  ADD     SI,duRecSize
0889: E2E3                      LOOP    $10
088B:
088B: E8 8AFD                   CALL    restRegs
088E: C3                        RET
088F:
088F:                   ;*************************************************************************
088F:                   ;
088F:                   ; FLIP UCSD DIRECTORY IN dirBuffer
088F:                   ;
088F:                   ;*************************************************************************
088F:                   FLIP_DIR
088F: E8 77FD                   CALL    saveRegs
0892:
0892: BB ****                   LEA     BX,dirBuffer+ucsdNumFiles       ;only in directory entry zero
0895: E8 6EFF                   CALL    flip
0898:
0898: BE ****                   LEA     SI,dirBuffer
089B: B9 4E 00                  MOV     CX,78.                          ;78 entries in directory
089E:                   $10
089E: 8D 5C 00                  LEA     BX,(SI+ucsd1stBlk)
08A1: E8 62FF                   CALL    flip
08A4:
08A4: 8D 5C 02                  LEA     BX,(SI+ucsdLastBlk)
08A7: E8 5CFF                   CALL    flip
08AA:
08AA: 83 C6 1A                  ADD     SI,ucsdDirSize
08AD: E2EF                      LOOP    $10
08AF:
08AF: E8 66FD                   CALL    restRegs
08B2: C3                        RET
08B3:                   ;*************************************************************************
08B3:                   ;
08B3:                   ; MAKEDRVCMD -- converts the vars drive and block to a 4 byte read block
08B3:                   ;               command located in drvCmd
08B3:                   ;
08B3:                   ;*************************************************************************
08B3:                   makeDrvCmd
08B3: E8 53FD                   CALL    saveRegs                ;save regsiters on stack
08B6: C6 06 BB05 32             MOVBIM  drvCmd,rdCmd            ; drive read block command
08BB: A0 B105                   MOV     AL,block+1              ; get msb 4 bits of 20bit block address
08BE: B1 04                     MOV     CL,4                    ;shift 4 bits left
08C0: D2 E0                     SHL     AL,CL
08C2: 0A 06 AF05                OR      AL,drive                ;put drive in ls 4 bits
08C6: A2 BC05                   MOV     drvCmd+1,AL
```

```
08C9: A1 B205                       MOV     AX,block+2
08CC: 86 E0                         XCHG    AH,AL
08CE: A3 BD05                       MOV     drvCmd+2,AX             ;put ls 16 bit block address in cmd
08D1: E8 44FD                       CALL    restRegs                ;restore registers
08D4: C3                            RET
08D5:
08D5:                       ;*****************************************************************************
08D5:                       ;
08D5:                       ; CHKNUMDRIVES -- determines the number of drives online, places the
08D5:                       ;                 number in the var readblock
08D5:                       ;
08D5:                       ;*****************************************************************************
08D5:                       chkNumDrives
08D5: C6 06 BA05 01                 MOVBIM  errOk,1                 ;read errors ok
08DA: B0 00                         MOV     AL,0                    ;set block address to zero
08DC: B9 04 00                      MOV     CX,addlen
08DF: BF B005                       LEA     DI,block
08E2: F3 AA                     REP STOSB
08E4: C6 06 AF05 01                 MOVBIM  drive,1                 ;start at drive one
08E9:                       $10
08E9: E8 ****                       CALL    readBlock               ;read a block
08EC: 80 3E B805 00                 CMPBIM  ioRetCode,0             ;was return code = zero
08F1: 75**                          JNE     $20                     ; no, end of drives
08F3: FE 06 AF05                    INCMB   drive                   ;try next drive
08F7: EBF0                          JMP     $10
08F9:                       $20
08F9: A0 AF05                       MOV     AL,drive
08FC: FE C8                         DEC     AL
08FE: A2 A005                       MOV     drvsOnLine,AL
0901: C6 06 BA05 00                 MOVBIM  errOk,0                 ;errors not ok
0906: C3                            RET
0907:
0907:                       ;*****************************************************************************
0907:                       ;
0907:                       ; WAITFORKEY -- prints 'press any key to continue' and waits for key press
0907:                       ;
0907:                       ;*****************************************************************************
0907:                       waitForKey
0907: E8 FFFC                       CALL    saveRegs                ;save register on stack
090A: BE D101                       LEA     SI,waitMsg
090D: E8 41FD                       CALL    printStr
0910:
0910: B4 00                         MOV     AH,0                    ;specify read keyboard
0912: CD 16                         INT     16H                     ;call keyboard i/o routines
0914:
0914: E8 01FD                       CALL    restRegs                ;restore registers
0917: C3                            RET
0918:                       ;*****************************************************************************
0918:                       ;
0918:                       ; ADD_TO_BLOCK -- ADDS AX REGISTER TO BLOCK
0918:                       ;
0918:                       ;*****************************************************************************
0918:                       addToBlock
0918: 53                            PUSH    BX
```

```
0919: 8B D8                         MOV     BX,AX
091B:
091B: A1 B205                       MOV     AX,block+2
091E: 86 E0                         XCHG    AH,AL
0920: 03 C3                         ADD     AX,BX
0922: 86 E0                         XCHG    AH,AL
0924: A3 B205                       MOV     block+2,AX
0927:
0927: A1 B005                       MOV     AX,block
092A: 86 E0                         XCHG    AH,AL
092C: 15 00 00                      ADC     AX,0
092F: 86 E0                         XCHG    AH,AL
0931: A3 B005                       MOV     block,AX
0934:
0934: 5B                            POP     BX
0935: C3                            RET
0936:
0936:                       ;*************************************************************************
0936:                       ;
0936:                       ; NEXT BLOCK -- adds 1 to the 4 byte abs disk address in var block
0936:                       ;
0936:                       ;*************************************************************************
0936:                       nextBlock
0936: 50                            PUSH    AX
0937: B8 01 00                      MOV     AX,1
093A: E8 DBFF                       CALL    addToBlock
093D: 58                            POP     AX
093E: C3                            RET
093F:
093F:
093F:                       ;*************************************************************************
093F:                       ;
093F:                       ; LOGON -- read drive info table, prompts user for userid and password,
093F:                       ;             loads users Network User entry into nuRec
093F:                       ;
093F:                       ;*************************************************************************
093F:                       logon
093F: E8 E5FC                       CALL    home                    ;clear screen
0942: C6 06 AF05 01                 MOVBIM  drive,1                 ;read DI table from drive 1
0947: BE 2003                       LEA     SI,diAdd                ;block address of DI table
094A: BF B005                       LEA     DI,block                ;param for read block routine
094D: B9 04 00                      MOV     CX,addLen               ;length of abs block address
0950: F3 A4                   REP   MOVSB                           ;move address to parm area
0952:
0952: E8 ****                       CALL    readBlock               ;do the read
0955: 80 3E B805 00                 CMPBIM  ioRetCode,0             ;was read ok?
095A: 74**                          JE      $30                     ; yes
095C:                       $10
095C: BE 1601                       LEA     SI,badDiMsg             ;bad DI table message
095F: E8 EFFC                       CALL    printStr                ;print it
0962:                       $15
0962: EBFE                          JMP     $15                     ;and loop
0964:                       $20
0964: BE 3F01                       LEA     SI,invDiMsg             ;not inited Di table msg
0967: E8 E7FC                       CALL    printStr
```

```
096A: EBF6                          JMP     $15                     ;loop forever
096C:                         $30
096C: BE ****                       LEA     SI,buffer               ;address of block from read
096F: E8 98FE                       CALL    flipDi                  ;make sex correct
0972: 81 7C 44 BE 07                CMP     (SI+diSpecial),1982     ;is DI table inited?
0977: 75EB                          JNE     $20                     ; no,print err and hang
0979: 80 7C 34 01                   CMPBIM  (SI+diDrvInit),1
097D: 75E5                          JNE     $20
097F: 80 7C 38 01                   CMPBIM  (SI+diOnline),1
0983: 75DF                          JNE     $20
0985:
0985: BF 2904                       LEA     DI,diRec                ;move buffer to record area
0988: B9 5B 00                      MOV     CX,diRecSize
098B: F3 A4                    REP  MOVSB
098D:
098D:                         logoni
098D: E8 66FD                       CALL    putLogo                 ;put logon msg on screen
0990: E8 6DFD                       CALL    getUserId               ;get user name
0993: E8 A0FD                       CALL    encryptName             ;encrypt it for search
0996:
0996: BB 2904                       LEA     BX,diRec                ;address of DI record
0999: 8D 77 46                      LEA     SI,(BX+diNuAddr)         ;move add of NU table to parm area
099C: BF B005                       LEA     DI,block
099F: B9 04 00                      MOV     CX,addLen
09A2: F3 A4                    REP  MOVSB
09A4:
09A4: 8B 4F 3A                      MOV     CX,(BX+diNuBlks)         ;number of blocks in NU table
09A7: 89 0E 5E05                    MOV     numBlocks,CX
09AB:
09AB:                         logon2
09AB: 83 3E 5E05 00                 CMP     numBlocks,0             ;are we all done?
09B0: 75**                          JNE     logon3                  ;no read next block in NU
09B2: BA 00 12                      MOV     DX,xyInvUsr             ;print invalid user message
09B5: E8 8CFC                       CALL    gotoxy
09B8: BE 6A01                       LEA     SI,invUsrMsg
09BB: E8 93FC                       CALL    printStr
09BE: E8 46FF                       CALL    waitForKey              ;wait for a keypress
09C1: EBCA                          JMP     logon1                  ;try again
09C3:                         logon3
09C3: E8 ****                       CALL    readBlock               ;read one block of NU
09C6: BE ****                       LEA     SI,buffer
09C9: E8 87FE                       CALL    flipNu                  ;make sex correct
09CC: BB ****                       LEA     BX,buffer               ;address of buffer
09CF: BA 10 00                      MOV     DX,nuRPB                ;# of entries in NU block
09D2:                         logon4
09D2: 8D 77 00                      LEA     SI,(BX+nuUser)           ;address of user name in buffer
09D5: BF 7A05                       LEA     DI,cryptName            ;address of name user entered
09D8: B9 0A 00                      MOV     CX,nameLen              ;compare length
09DB: F3 A6                    REPZ CMPSB                           ;compare them
09DD: 74**                          JZ      logon5                  ;found match
09DF:
09DF: 83 C3 20                      ADD     BX,nuRecSize            ;point to next entry in table
09E2: 4A                            DEC     DX                      ;any more left in buffer
09E3: 75ED                          JNZ     logon4                  ;try again
09E5: E8 4EFF                       CALL    nextBlock               ;point to next block
```

```
09E8: FF 0E 5E05                        DEC     numBlocks
09EC: EBBD                              JMP     logon2
09EE:                           logon5
09EE: BF 7105                           LEA     DI,userPassword         ;see if user has password
09F1: B9 08 00                          MOV     CX,passLen
09F4: B0 20                             MOV     AL,20H                  ;fill with blanks
09F6: F3 AA                       REP   STOSB
09F8:
09F8: E8 C4FD                           CALL    encryptPassword         ;encrypt it for search
09FB: 8D 77 0A                          LEA     SI,(BX+duPass)           ;set up registers for match
09FE: BF 8405                           LEA     DI,cryptPass
0A01: B9 08 00                          MOV     CX,passLen
0A04: F3 A6                       REPZ  CMPSB                           ;do they match?
0A06: 74**                              JZ      logon6                  ; yes, user has no password
0A08:
0A08: E8 0DFD                           CALL    getPassword
0A0B: E8 B1FD                           CALL    encryptPassword         ;encrypt it for search
0A0E: 8D 77 0A                          LEA     SI,(BX+duPass)           ;set up registers for match
0A11: BF 8405                           LEA     DI,cryptPass
0A14: B9 08 00                          MOV     CX,passLen
0A17: F3 A6                       REPZ  CMPSB                           ;do they match?
0A19: 74**                              JZ      logon6                  ; yes, user has no password
0A1B:
0A1B: BA 00 12                          MOV     DX,xyInvPass            ;set x,y for inv password msg
0A1E: E8 23FC                           CALL    gotoxy
0A21: BE 7C01                           LEA     SI,invPassMsg
0A24: E8 2AFC                           CALL    printStr                ;print error message
0A27: E8 DDFE                           CALL    waitForKey              ;wait for a key press
0A2A: E9 60FF                           JMP     logon1                  ;and try again
0A2D:                           logon6
0A2D: 8B F3                             MOV     SI,BX                   ;si = add of user NU entry
0A2F: BF 8404                           LEA     DI,nuRec                ;di = record to store NU in
0A32: B9 20 00                          MOV     CX,nuRecSize
0A35: F3 A4                       REP   MOVSB                           ;move entry to record area
0A37:
0A37: 83 3E A004 04                     CMP     nuRec+nuOpSys,ibmDos    ;we support only ibmDos and UCSD40
0A3C: 74**                              JE      logon7
0A3E: 83 3E A004 11                     CMP     nuRec+nuOpSys,UCSD40
0A43: 74**                              JE      logon7
0A45:
0A45: BE A901                           LEA     SI,badOsMsg
0A48: E8 A1FC                           CALL    printAndWait
0A4B: E9 3FFF                           JMP     logon1
0A4E:
0A4E:                           logon7
0A4E: C3                                RET                             ;and return
0A4F:
0A4F:                           ;*****************************************************
0A4F:                           ;
0A4F:                           ; FIND_SERVER  -- finds disk server with name that is in NU table
0A4F:                           ;                  carry set if server is not found. Places server
0A4F:                           ;                  host number in serverHost
0A4F:                           ;
0A4F:                           ;                IF a disk server doesnt respond we assume that it is a
0A4F:                           ;                  old disk server that doesnt have the const II protocol
```

```
0A4F:                          ;                  and we continue with issueing an error message.
0A4F:
0A4F:                          ;*****************************************************************
0A4F:                          findServer
0A4F: BE 9604                          LEA     SI,nuRec+nuServer       ;users server name
0A52: BF 9605                          LEA     DI,tempEname
0A55: B9 0A 00                         MOV     CX,nameLen
0A58: F3 A4                       REP  MOVSB
0A5A:
0A5A: E8 20FD                          CALL    deCryptName             ;decrypt server name
0A5D: BE 8C05                          LEA     SI,tempDname
0A60: BF C905                          LEA     DI,wayName              ;WhereAreYouName
0A63: B9 0A 00                         MOV     CX,nameLen
0A66: F3 A4                       REP  MOVSB
0A68:
0A68: A0 7905                          MOV     AL,userHost             ;our host number
0A6B: A2 C605                          MOV     waySource,AL            ;put in where are you message
0A6E:
0A6E: BE C105                          LEA     SI,whereAreYou          ;get ready to send packet
0A71: B9 12 00                         MOV     CX,18.
0A74: BF D305                          LEA     DI,myIdIs
0A77: BA 12 00                         MOV     DX,18.
0A7A: B3 01                            MOV     BL,1                    ;wait .86 seconds
0A7C: B7 03                            MOV     BH,3                    ;try 3 times
0A7E: B0 FF                            MOV     AL,0FFH                 ;broadcast message
0A80: B4 03                            MOV     AH,3                    ;network xmit/recv cmd
0A82:
0A82: FF 1E D703                       CALLL   (romIo)
0A86: 0A C0                            OR      AL,AL                   ;was return ok
0A88: 74**                             JZ      $10                     ; yes
0A8A:
0A8A:                                  ;LEA     SI,noServerMsg         ;THIS IS NOT USED CAUSE THIS BOOT MUST WORK
0A8A:                                  ;CALL    printAndWait           ; WITH OLD DISK SERVERS
0A8A:                                  ;STC                             ;set Error
0A8A: F8                               CLC
0A8B: C3                               RET
0A8C:                          $10
0A8C: A0 D805                          MOV     AL,miiSource            ;now start using this server
0A8F: A2 A305                          MOV     serverHost,AL
0A92: F8                               CLC                             ;set server found
0A93: C3                               RET
0A94:
0A94:
0A94:                          ;*****************************************************************
0A94:                          ;
0A94:                          ; OMNI_HELLO -- broadcast a hello message on the network
0A94:                          ;
0A94:                          ;*****************************************************************
0A94:                          omniHello
0A94: BE 6705                          LEA     SI,userName
0A97: BF ED05                          LEA     DI,helloName
0A9A: B9 0A 00                         MOV     CX,nameLen
0A9D: F3 A4                       REP  MOVSB
0A9F:
```

```
0A9F: A0 7905                       MOV     AL,userHost
0AA2: A2 EA05                       MOV     helloSource,AL
0AA5:
0AA5: A0 A204                       MOV     AL,nuRec+nuHostType
0AA8: A2 EC05                       MOV     helloType,AL
0AAB:
0AAB: BE E505                       LEA     SI,hello
0AAE: B9 12 00                      MOV     CX,18.
0AB1: B7 01                         MOV     BH,1                    ;send it once
0AB3: B0 FF                         MOV     AL,0FFH                 ;broadcast message
0AB5: B4 02                         MOV     AH,2                    ;xmit data to server
0AB7: FF 1E D703                    CALLL   (romIo)
0ABB: C3                            RET
0ABC:
0ABC:                       ;*************************************************************************
0ABC:                       ;
0ABC:                       ; FLAT_HELLO -- sends an add active command to the disk server
0ABC:                       ;
0ABC:                       ;*************************************************************************
0ABC:                       flatHello
0ABC: BE 6705                       LEA     SI,userName
0ABF: BF F905                       LEA     DI,addActName
0AC2: B9 0A 00                      MOV     CX,nameLen
0AC5: F3 A4                    REP  MOVSB
0AC7:
0AC7: BE F705                       LEA     SI,addActive
0ACA: B9 12 00                      MOV     CX,18.
0ACD: BA 00 00                      MOV     DX,0                    ;no bytes back
0AD0: B4 01                         MOV     AH,1                    ;xmit/recv to drive
0AD2: FF 1E D703                    CALLL   (romIo)
0AD6: C3                            RET
0AD7:
0AD7:                       ;*************************************************************************
0AD7:                       ;
0AD7:                       ;FIND_BT_DISK -- 1. Searches all drives starting at drive 1 looking
0AD7:                       ;                   for a driveUser entry that contains the flag
0AD7:                       ;                   indicating that this drive contains the volume
0AD7:                       ;                   that is the boot volume for this user.
0AD7:                       ;                2. Searches the driveAccess table on this drive
0AD7:                       ;                   for the entry that is the boot volume.
0AD7:                       ;                3  Searches the driveVolume table for the entry
0AD7:                       ;                   that is the boot volume
0AD7:                       ;                4. Returns with carry set if boot volume was not
0AD7:                       ;                   found, else carry is cleared
0AD7:                       ;                5. Moves the found entry into the var dvBtRec
0AD7:                       ;*************************************************************************
0AD7:                       findBtDisk
0AD7: E8 FBFD                       CALL    chkNumDrives                    ;find number of drives online
0ADA: C6 06 AF05 00                 MOVBIM  drive,0                         ;init drive to zero
0ADF:                       findBt01
0ADF: FE 06 AF05                    INCMB   drive                           ;look at next drive
0AE3: A0 AF05                       MOV     AL,drive                        ;have we done all drives?
0AE6: 3A 06 A005                    CMP     AL,drvsOnLine
0AEA: 76**                          JNA     findBt02                        ;no now read driveInfo table
0AEC:
```

```
0AEC: BE 9D01                    LEA     SI,noBootMsg                ;boot volume not found
0AEF: E8 FAFB                    CALL    printAndWait                ;wait for keypress
0AF2: F9                         STC                                 ;set return code
0AF3: C3                         RET
0AF4:                  findBt02
0AF4: BE 2003                    LEA     SI,diAdd                    ;block add of DI table
0AF7: BF B005                    LEA     DI,block
0AFA: B9 04 00                   MOV     CX,addLen
0AFD: F3 A4                 REP  MOVSB                               ;set up for readBlock
0AFF:
0AFF: E8 ****                    CALL    readBlock                   ;read the drive info table
0B02: 80 3E B805 00              CMPBIM  ioRetCode,0                 ;did we read it ok?
0B07: 75D6                       JNE     findBt01                    ; no, try next drive
0B09:
0B09: BE ****                    LEA     SI,buffer                   ;table was read in here
0B0C: E8 FEFC                    CALL    flipDi                      ;flip sex
0B0F:
0B0F: 81 7C 44 BE 07             CMP     (SI+diSpecial),1982         ;is drive info valid?
0B14: 75C9                       JNE     findBt01                    ;no, try next drive
0B16: 80 7C 34 01                CMPBIM  (SI+diDrvInit),1
0B1A: 75C3                       JNE     findBt01
0B1C: 80 7C 38 01                CMPBIM  (SI+diOnline),1
0B20: 75BD                       JNE     findBt01
0B22:
0B22: BF 2904                    LEA     DI,diRec                    ;move to record area
0B25: B9 5B 00                   MOV     CX,diRecSize
0B28: F3 A4                 REP  MOVSB
0B2A:
0B2A: A1 6704                    MOV     AX,diRec+diDuBlks           ;number of blocks in DU
0B2D: A3 5E05                    MOV     numBlocks,AX
0B30:
0B30: BE 7704                    LEA     SI,diRec+DiDuAddr
0B33: BF B005                    LEA     DI,block
0B36: B9 04 00                   MOV     CX,addLen                   ;set up address of DU
0B39: F3 A4                 REP  MOVSB
0B3B:
0B3B:                  findBt03
0B3B:
0B3B: 83 3E 5E05 00              CMP     numBlocks,0                 ;have we searched entire DU
0B40: 75**                       JNE     $05
0B42: EB9B                       JMP     findBt01                    ; yes, try next drive
0B44:                  $05
0B44: E8 ****                    CALL    readBlock                   ;read driveUser table
0B47: BE ****                    LEA     SI,buffer                   ;flip
0B4A: E8 1BFD                    CALL    flipDu                      ; word sex
0B4D:
0B4D: BB ****                    LEA     BX,buffer                   ;address of buffer
0B50: BA 10 00                   MOV     DX,duRPB                    ;records per block
0B53:                  $10
0B53: 8D 77 00                   LEA     SI,(BX+duUser)              ;compare user id
0B56: BF 7A05                    LEA     DI,cryptName
0B59: B9 0A 00                   MOV     CX,nameLen
0B5C: F3 A6                 REPZ CMPSB
0B5E: 74**                       JZ      findBt04                    ;found match in DriveUser
0B60:
```

```
0B60: 83 C3 20                      ADD     BX,duRecSize                ;next in buffer
0B63: 4A                            DEC     DX                          ;any left in buffer
0B64: 75ED                          JNZ     $10                         ; yes, look at next entry
0B66:
0B66: E8 CDFD                       CALL    nextBlock                   ;add 1 to address
0B69: FF 0E 5E05                    DEC     numBlocks                   ;dec number of blocks left
0B6D: EBCC                          JMP     findBt03                    ;read next block
0B6F:                       findBt04
0B6F: 83 7F 18 01                   CMP     (BX+duBootInfo),1           ;is boot volume on this drive
0B73: 74**                          JE      $01
0B75: E9 67FF                       JMP     findBt01                    ; no, try next drive
0B78:                       $01
0B78: 8B F3                         MOV     SI,BX                       ;address of matching DU
0B7A: BF AC04                       LEA     DI,duRec                    ;address of record area
0B7D: B9 20 00                      MOV     CX,duRecSize
0B80: F3 A4                    REP  MOVSB                               ;move DU rec to record area
0B82:
0B82: 8B 0E 6904                    MOV     CX,diRec+diDaBlks           ;num blocks in DA table
0B86: 89 0E 5E05                    MOV     numBlocks,CX
0B8A:
0B8A: BE 7B04                       LEA     SI,diRec+diDaAddr
0B8D: BF B005                       LEA     DI,block
0B90: B9 04 00                      MOV     CX,addLen
0B93: F3 A4                    REP  MOVSB
0B95:
0B95:                       findBt05
0B95: 83 3E 5E05 00                 CMP     numBlocks,0                 ;any left in DA table
0B9A: 75**                          JNE     $01
0B9C: E9 40FF                       JMP     findBt01                    ; no, try next drive
0B9F:                       $01
0B9F: E8 ****                       CALL    readBlock                   ;read DA entry
0BA2: BE ****                       LEA     SI,buffer
0BA5: E8 96FC                       CALL    flipDa                      ;flip word sex
0BA8:
0BA8: 8B DE                         MOV     BX,SI                       ;BX = add of DA entry
0BAA: BA 40 00                      MOV     DX,daRPB                    ;records per block
0BAD:                       $10
0BAD: 8B 47 00                      MOV     AX,(BX+daUserid)            ;userid in DA entry
0BB0: 3B 06 BE04                    CMP     AX,duRec+duUserid           ;does userid match
0BB4: 75**                          JNE     $20                         ;no, goto next entry
0BB6:
0BB6: F7 47 07 01 00                TEST    (BX+daReadBoot),daBootFlag  ;is this the boot entry
0BBB: 75**                          JNZ     findBt06                    ; yes
0BBD:                       $20
0BBD: 83 C3 08                      ADD     BX,daRecSize                ;next entry
0BC0: 4A                            DEC     DX                          ;any left
0BC1: 75EA                          JNZ     $10                         ;yes
0BC3:
0BC3: E8 70FD                       CALL    nextBlock
0BC6: FF 0E 5E05                    DEC     numBlocks
0BCA: EBC9                          JMP     findBt05                    ;get next block maybe
0BCC:                       findBt06
0BCC: 8B F3                         MOV     SI,BX                       ;add of found DA entry
0BCE: BF A404                       LEA     DI,daRec
0BD1: B9 08 00                      MOV     CX,daRecSize
```

```
OBD4: F3 A4                     REP  MOVSB                                  ;move DA to record area
OBD6:
OBD6: 8B 0E 6504                     MOV     CX,diRec+diDvBlks              ;get ready to read drive Volume
OBDA: 89 0E 5E05                     MOV     numBlocks,CX
OBDE:
OBDE: BE 7304                        LEA     SI,diRec+diDvAddr
OBE1: BF B005                        LEA     DI,block
OBE4: B9 04 00                       MOV     CX,addLen
OBE7: F3 A4                     REP  MOVSB
OBE9:
OBE9:                           findBt07
OBE9: 83 3E 5E05 00                  CMP     numBlocks,0                    ;any blocks left in DV
OBEE: 75**                           JNE     $01
OBF0: E9 ECFE                        JMP     findBt01                       ; no, try next drive
OBF3:                           $01
OBF3: E8 ****                        CALL    readBlock
OBF6:
OBF6: BB ****                        LEA     BX,buffer                      ;BX=add of DV entry
OBF9: BA 10 00                       MOV     DX,dvRPB                       ;entries per block
OBFC:                           $10
OBFC: 8D 77 0A                       LEA     SI,(BX+dvStartBlock)
OBFF: BF A604                        LEA     DI,daRec+daVolAddr
0C02: B9 04 00                       MOV     CX,addLen
0C05: F3 A6                     REPZ CMPSB                                  ;is this the volume?
0C07: 74**                           JZ      findBt08                       ;yes, finally here
0C09:
0C09: 83 C3 20                       ADD     BX,dvRecSize                   ;point to next entry
0C0C: 4A                             DEC     DX                             ;any left
0C0D: 75ED                           JNZ     $10                            ; yes
0C0F:
0C0F: E8 24FD                        CALL    nextBlock
0C12: FF 0E 5E05                     DEC     numBlocks
0C16: EBD1                           JMP     findBt07
0C18:                           findBt08
0C18: 8B F3                          MOV     SI,BX                          ;add of DV entry
0C1A: BF EC04                        LEA     DI,dvBtRec                     ;add of record area
0C1D: B9 20 00                       MOV     CX,dvRecSize
0C20: F3 A4                     REP  MOVSB                                  ;move it to record area
0C22:
0C22: F8                             CLC                                    ;set no error
0C23: C3                             RET
0C24:                           ;*****************************************************************
0C24:                           ;
0C24:                           ;FIND_MT_DISK -- 1. Searches all drives starting at drive 1 looking
0C24:                           ;                   for a driveUser entry for the logged on user
0C24:                           ;                2. Searches the driveAccess table on this drive
0C24:                           ;                   for the mounted entrys for this user.
0C24:                           ;                3. Searches the driveVolume table for the entry
0C24:                           ;                   that matches the drive access volume.
0C24:                           ;                4. Moves the found entrys into a mount table
0C24:                           ;*****************************************************************
0C24:                           findMtDisk
0C24: A0 AF05                        MOV     AL,drive
0C27: 50                             PUSH    AX                             ;save current drive
0C28:
```

```
OC28: BE ****                       LEA     SI,mntDvTab                 ;set up table pointers
OC2B: 89 36 0C05                    MOV     mntDvPoint,SI
OC2F:
OC2F: BE ****                       LEA     SI,mntDaTab
OC32: 89 36 0E05                    MOV     mntDaPoint,SI
OC36:
OC36: BE ****                       LEA     SI,mntDrvTab
OC39: 89 36 1005                    MOV     mntDrvPoint,SI
OC3D:
OC3D: C6 06 AF05 00                 MOVBIM  drive,0                     ;init drive to zero
OC42:                       findMt01
OC42: FE 06 AF05                    INCMB   drive                       ;look at next drive
OC46: A0 AF05                       MOV     AL,drive                    ;have we done all drives?
OC49: 3A 06 A005                    CMP     AL,drvsOnLine
OC4D: 76**                          JNA     findMt02                    ;no now read driveInfo table
OC4F:
OC4F: 58                            POP     AX                          ;restore current drive
OC50: A2 AF05                       MOV     drive,AL
OC53:
OC53: C3                            RET                                 ;all done, have searched all drives
OC54:                       findMt02
OC54: BE 2003                       LEA     SI,diAdd                    ;block add of DI table
OC57: BF B005                       LEA     DI,block
OC5A: B9 04 00                      MOV     CX,addLen
OC5D: F3 A4                    REP  MOVSB                               ;set up for readBlock
OC5F:
OC5F: E8 ****                       CALL    readBlock                   ;read the drive info table
OC62: 80 3E B805 00                 CMPBIM  ioRetCode,0                 ;did we read it ok?
OC67: 75D9                          JNE     findMt01                    ; no, try next drive
OC69:
OC69: BE ****                       LEA     SI,buffer                   ;table was read in here
OC6C: E8 9EFB                       CALL    flipDi                      ;flip sex
OC6F:
OC6F: 81 7C 44 BE 07                CMP     (SI+diSpecial),1982         ;is drive info valid?
OC74: 75CC                          JNE     findMt01                    ;no, try next drive
OC76: 80 7C 34 01                   CMPBIM  (SI+diDrvInit),1
OC7A: 75C6                          JNE     findMt01
OC7C: 80 7C 38 01                   CMPBIM  (SI+diOnline),1
OC80: 75C0                          JNE     findMt01
OC82:
OC82: BF 2904                       LEA     DI,diRec                    ;move to record area
OC85: B9 5B 00                      MOV     CX,diRecSize
OC88: F3 A4                    REP  MOVSB
OC8A:
OC8A: A1 6704                       MOV     AX,diRec+diDuBlks           ;number of blocks in DU
OC8D: A3 5E05                       MOV     numBlocks,AX
OC90:
OC90: BE 7704                       LEA     SI,diRec+DiDuAddr
OC93: BF B005                       LEA     DI,block
OC96: B9 04 00                      MOV     CX,addLen                   ;set up address of DU
OC99: F3 A4                    REP  MOVSB
OC9B:
OC9B:                       findMt03
OC9B:
OC9B: 83 3E 5E05 00                 CMP     numBlocks,0                 ;have we searched entire DU
```

```
0CA0: 75**                          JNE     $05
0CA2: EB9E                          JMP     findMt01                    ; yes, try next drive
0CA4:                       $05
0CA4: E8 ****                       CALL    readBlock                   ;read driveUser table
0CA7: BE ****                       LEA     SI,buffer                   ;flip
0CAA: E8 BBFB                       CALL    flipDu                      ; word sex
0CAD:
0CAD: BB ****                       LEA     BX,buffer                   ;address of buffer
0CB0: BA 10 00                      MOV     DX,duRPB                    ;records per block
0CB3:                       $10
0CB3: 8D 77 00                      LEA     SI,(BX+duUser)              ;compare user id
0CB6: BF 7A05                       LEA     DI,cryptName
0CB9: B9 0A 00                      MOV     CX,nameLen
0CBC: F3 A6                   REPZ  CMPSB
0CBE: 74**                          JZ      findMt04                    ;found match in DriveUser
0CC0:
0CC0: 83 C3 20                      ADD     BX,duRecSize                ;next in buffer
0CC3: 4A                            DEC     DX                          ;any left in buffer
0CC4: 75ED                          JNZ     $10                         ; yes, look at next entry
0CC6:
0CC6: E8 6DFC                       CALL    nextBlock                   ;add 1 to address
0CC9: FF 0E 5E05                    DEC     numBlocks                   ;dec number of blocks left
0CCD: EBCC                          JMP     findMt03                    ;read next block
0CCF:                       findMt04
0CCF: 8B F3                         MOV     SI,BX                       ;address of matching DU
0CD1: BF AC04                       LEA     DI,duRec                    ;address of record area
0CD4: B9 20 00                      MOV     CX,duRecSize
0CD7: F3 A4                   REP   MOVSB                               ;move DU rec to record area
0CD9:
0CD9: 8B 0E 6904                    MOV     CX,diRec+diDaBlks           ;num blocks in DA table
0CDD: 89 0E 5E05                    MOV     numBlocks,CX
0CE1:
0CE1: BE 7B04                       LEA     SI,diRec+diDaAddr
0CE4: BF B005                       LEA     DI,block
0CE7: B9 04 00                      MOV     CX,addLen
0CEA: F3 A4                   REP   MOVSB
0CEC:
0CEC:                       findMt05
0CEC: 83 3E 5E05 00                 CMP     numBlocks,0                 ;any left in DA table
0CF1: 75**                          JNE     $01
0CF3: E9 4CFF                       JMP     findMt01                    ; no, try next drive
0CF6:                       $01
0CF6: E8 ****                       CALL    readBlock                   ;read DA entry
0CF9: BE ****                       LEA     SI,buffer
0CFC: E8 3FFB                       CALL    flipDa                      ;flip word sex
0CFF:
0CFF: 8B DE                         MOV     BX,SI                       ;BX = add of DA entry
0D01: BA 40 00                      MOV     DX,daRPB                    ;records per block
0D04:                       $10
0D04: 83 7F 00 FF                   CMP     (BX+daUserId),0FFFFH        ;is this table entry null?
0D08: 74**                          JE      $20                         ;ignore it
0D0A: A1 BE04                       MOV     AX,duRec+duUserid           ;our users id
0D0D: 39 47 00                      CMP     (BX+daUserId),AX            ;do they match
0D10: 74**                          JE      $15                         ;yes
0D12: 72**                          JB      $20                         ;file is ordered by userid
```

```
OD14: E9 2BFF                       JMP     findMtO1                    ;check next drive
OD17:                       $15
OD17: 80 7F 06 00                   CMPBIM  (BX+daMntInfo),0            ;is anything mounted
OD1B: 74**                          JE      $20                         ;no goto next entry
OD1D:
OD1D: E8 ****                       CALL    findMtO6                    ;load volume info into volume
OD20:                                                                   ; table
OD20:                       $20
OD20: 83 C3 08                      ADD     BX,daRecSize                ;next entry
OD23: 4A                            DEC     DX                          ;any left
OD24: 75DE                          JNZ     $10                         ;yes
OD26:
OD26: E8 ODFC                       CALL    nextBlock
OD29: FF OE 5E05                    DEC     numBlocks
OD2D: EBBD                          JMP     findMtO5                    ;get next block maybe
OD2F:
OD2F:
OD2F:                       findMtO6
OD2F: E8 D7F8                       CALL    saveRegs
OD32:
OD32: A1 5E05                       MOV     AX,numBlocks                ;save vars used by findMt
OD35: A3 6005                       MOV     holdNumBlocks,AX
OD38:
OD38: BE B005                       LEA     SI,block
OD3B: BF B405                       LEA     DI,holdBlock
OD3E: B9 04 00                      MOV     CX,addLen
OD41: F3 A4                     REP MOVSB
OD43:
OD43: BE ****                       LEA     SI,buffer
OD46: BF ****                       LEA     DI,holdBuffer
OD49: B9 00 02                      MOV     CX,512.
OD4C: F3 A4                     REP MOVSB
OD4E:
OD4E: 8B F3                         MOV     SI,BX                       ;add of found DA entry
OD50: BF A404                       LEA     DI,daRec
OD53: B9 08 00                      MOV     CX,daRecSize
OD56: F3 A4                     REP MOVSB                               ;move DA to record area
OD58:
OD58: 8B 0E 6504                    MOV     CX,diRec+diDvBlks           ;get ready to read drive Volume
OD5C: 89 0E 5E05                    MOV     numBlocks,CX
OD60:
OD60: BE 7304                       LEA     SI,diRec+diDvAddr
OD63: BF B005                       LEA     DI,block
OD66: B9 04 00                      MOV     CX,addLen
OD69: F3 A4                     REP MOVSB
OD6B:
OD6B:                       findMtO7
OD6B: 83 3E 5E05 00                 CMP     numBlocks,0                 ;any blocks left in DV
OD70: 74**                          JE      findMtEnd
OD72: E8 ****                       CALL    readBlock
OD75:
OD75: BB ****                       LEA     BX,buffer                   ;BX=add of DV entry
OD78: BA 10 00                      MOV     DX,dvRPB                    ;entries per block
OD7B:                       $10
OD7B: 8D 77 0A                      LEA     SI,(BX+dvStartBlock)
```

```
OD7E: BF A604                       LEA     DI,daRec+daVolAddr
OD81: B9 04 00                      MOV     CX,addLen
OD84: F3 A6                   REPZ  CMPSB                           ;is this the volume?
OD86: 74**                          JZ      findMtOB                ;yes, finally here
OD88:
OD88: 83 C3 20                      ADD     BX,dvRecSize            ;point to next entry
OD8B: 4A                            DEC     DX                      ;any left
OD8C: 75ED                          JNZ     $10                     ; yes
OD8E:
OD8E: E8 A5FB                       CALL    nextBlock
OD91: FF 0E 5E05                    DEC     numBlocks
OD95: EBD4                          JMP     findMt07
OD97:                         findMtOB
OD97: 80 7F 14 01                   CMPBIM  (BX+dvGlbAccess),1      ;is this volume visible?
OD9B: 75**                          JNE     findMtEnd               ;no, forget it
OD9D: E8 ****                       CALL    findMtLoad              ;load into table
ODA0:                         findMtEnd
ODA0: BE B405                       LEA     SI,holdBlock            ;restore saved stuff
ODA3: BF B005                       LEA     DI,block
ODA6: B9 04 00                      MOV     CX,addLen
ODA9: F3 A4                    REP  MOVSB
ODAB:
ODAB: BE ****                       LEA     SI,holdBuffer
ODAE: BF ****                       LEA     DI,buffer
ODB1: B9 00 02                      MOV     CX,512.
ODB4: F3 A4                    REP  MOVSB
ODB6:
ODB6: A1 6005                       MOV     AX,holdNumBlocks
ODB9: A3 5E05                       MOV     numBlock,AX
ODBC:
ODBC: E8 59F8                       CALL    restregs
ODBF: C3                            RET
ODC0:
ODC0:                         findMtLoad
ODC0: 83 3E 1205 0A                 CMP     mntCount,mntMax         ;is table full
ODC5: 73**                          JAE     $10                     ;yes
ODC7:
ODC7: 8B F3                         MOV     SI,BX                   ;add of DV entry
ODC9: 8B 3E 0C05                    MOV     DI,mntDvPoint           ;add of record area
ODCD: B9 20 00                      MOV     CX,dvRecSize
ODD0: F3 A4                    REP  MOVSB                           ;move it to record area
ODD2:
ODD2: BE A404                       LEA     SI,daRec
ODD5: 8B 3E 0E05                    MOV     DI,mntDaPoint
ODD9: B9 08 00                      MOV     CX,daRecSize
ODDC: F3 A4                    REP  MOVSB
ODDE:
ODDE: 8B 1E 1005                    MOV     BX,mntDrvPoint
ODE2: A0 AF05                       MOV     AL,drive
ODE5: 88 07                         MOV     (BX),AL
ODE7:
ODE7: 83 06 0C05 20                 ADD     mntDvPoint,dvRecSize
ODEC: 83 06 0E05 08                 ADD     mntDaPoint,daRecSize
ODF1: FF 06 1005                    INC     mntDrvPoint
ODF5: FF 06 1205                    INC     mntCount
```

```
ODF9!                           $10
ODF9! C3                                RET
ODFA!
ODFA!                           ;*************************************************************
ODFA!                           ;
ODFA!                           ; READ_BLOCK -- reads block specified by drive, block
ODFA!                           ;               returns data in VAR buffer
ODFA!                           ;               returns return code in VAR ioRetCode
ODFA!                           ;               returns length (including return code) in VAR retLen
ODFA!                           ;
ODFA!                           ;*************************************************************
ODFA!                           readBlock
ODFA! E8 0CF8                           CALL    saveRegs                ;save registers on stack
ODFD!
ODFD! E8 B3FA                           CALL    makeDrvCmd              ;makes read command
OE00!
OE00! B4 01                             MOV     AH,1                    ;drive send/recv command
OE02! BE BB05                           LEA     SI,drvCmd               ;data to send to drive
OE05! B9 04 00                          MOV     CX,4                    ;length to send to drive
OE08!
OE08! BF ****                           LEA     DI,buffer               ;where to put result of read
OE0B! BA 00 02                          MOV     DX,sectSize             ;read one sector
OE0E!
OE0E! A0 A305                           MOV     AL,serverHost           ;server to use
OE11!
OE11! B3 00                             MOV     BL,0                    ;wait till server answers
OE13! B7 FF                             MOV     BH,255                  ;retry until server answers
OE15!
OE15! FF 1E D703                        CALLL   (romIo)
OE19!
OE19! A8 80                             TEST    AL,80H                  ;any errors?
OE1B! 75**                              JNZ     $05                     ;yes
OE1D! 32 C0                             XOR     AL,AL                   ;if no errors force retcode to zero
OE1F!                           $05
OE1F! A2 B805                           MOV     ioRetCode,AL
OE22! 89 0E BF05                        MOV     retLen,CX
OE26!
OE26! 80 3E A205 00                     CMPBIM  dots,dotOn
OE2B! 75**                              JNE     $10
OE2D!
OE2D! B4 0E                             MOV     AH,14.                  ;write byte i/o call
OE2F! B0 2E                             MOV     AL,2EH                  ;print a dot
OE31! BB 00 00                          MOV     BX,0
OE34! CD 10                             INT     10H
OE36!                           $10
OE36! 80 3E B805 00                     CMPBIM  ioRetCode,0             ;was all ok
OE3B! 74**                              JE      $20
OE3D! 80 3E BA05 00                     CMPBIM  errOk,0                 ;are errors ok
OE42! 75**                              JNE     $20                     ;yes
OE44!
OE44! E8 E0F7                           CALL    home
OE47! BE 0103                           LEA     SI,readErrMsg
OE4A! E8 04F8                           CALL    printStr
OE4D!                           $15
OE4D! EBFE                              JMP     $15                     ;and hang
```

```
0E4F:                           $20
0E4F: E8 C6F7                           CALL    restRegs                ;restore registers
0E52: C3                                RET
0E53:
0E53:                           ;*******************************************************************
0E53:                           ;
0E53:                           ; BOOT - call the correct boot procedure depending on the entry
0E53:                           ;        in the NETWORK USER table
0E53:                           ;
0E53:                           ;*******************************************************************
0E53:                           boot
0E53: 83 3E A004 11                     CMP     nuRec+nuOpSys,ucsd40
0E58: 75**                              JNE     $10
0E5A: E9 ****                           JMP     ucsdBoot
0E5D:                           $10
0E5D: 83 3E A004 04                     CMP     nuRec+nuOpSys,ibmDos
0E62: 75**                              JNE     $20
0E64: E9 ****                           JMP     msDosBoot
0E67:                           $20
0E67: C3                                RET
0E68:                           ;*******************************************************************
0E68:                           ;
0E68:                           ; READ_DIR      --  reads a UCSD dir into var dirBuffer
0E68:                           ;                   BX points to drive volume record
0E68:                           ;
0E68:                           ;*******************************************************************
0E68: E8 9EF7                   readDir         CALL    saveRegs
0E6B:
0E6B: 8D 77 0A                          LEA     SI,(BX+dvStartBlk)      ;add of beginning of volume
0E6E: BF B005                           LEA     DI,block
0E71: B9 04 00                          MOV     CX,addLen
0E74: F3 A4                       REP   MOVSB                           ;move add to parm for read
0E76:
0E76: 8A 47 15                          MOV     AL,(BX+dvVolOff)        ;offset to start of userLen
0E79: 32 E4                             XOR     AH,AH
0E7B: E8 9AFA                           CALL    addToBlock
0E7E:
0E7E: B8 02 00                          MOV     AX,2                    ;offset to directory
0E81: E8 94FA                           CALL    addToBlock
0E84:
0E84: BA 04 00                          MOV     DX,ucsdDirBlocks        ;# of blocks in directory
0E87: BF ****                           LEA     DI,dirBuffer            ;where to read directory
0E8A:                           $10
0E8A: E8 6DFF                           CALL    readBlock               ;read one block of directory
0E8D: BE ****                           LEA     SI,buffer               ;add of block read
0E90: B9 00 02                          MOV     CX,sectSize             ;number of bytes to move
0E93: F3 A4                       REP   MOVSB                           ;move block to directory buff
0E95:
0E95: E8 9EFA                           CALL    nextBlock               ;to next block in directory
0E98: 4A                                DEC     DX                      ;any more block to read
0E99: 75EF                              JNZ     $10                     ; yes
0E9B:
0E9B: C6 06 B905 01                     MOV8IM  dirIsFlipped,false
0EA0: 81 3E **** FF 00                  CMP     dirBuffer+ucsdLastBlk,0FFH ;is the directory flipped?
0EA6: 72**                              JB      $20                     ;NO
```

```
OEA8: E8 E4F9                           CALL    flipDir                 ;fix directory
OEAB: C6 06 B905 00                     MOVBIM  dirIsFlipped,true
OEB0:                           $20
OEB0: E8 65F7                           CALL    restRegs
OEB3: C3                                RET
OEB4:
OEB4:                           ;*******************************************************
OEB4:                           ;
OEB4:                           ; READ_UCSD_FILE --  reads a UCSD file into memory
OEB4:                           ;                    SI points to file name
OEB4:                           ;                    BX points to drive volume record
OEB4:                           ;                    DI points to area for file to be loaded
OEB4:                           ;                    if carry set then error
OEB4:                           ;                    directory assumed to be in dirBuffer
OEB4:                           ;                    returns with number of blocks read in DX
OEB4:                           ;
OEB4:                           ;*******************************************************
OEB4:                           readUcsdFile
OEB4: E8 52F7                           CALL    saveRegs
OEB7:
OEB7: 57                                PUSH    DI
OEB8: 53                                PUSH    BX
OEB9: 56                                PUSH    SI
OEBA:
OEBA: 8B 16 ****                        MOV     DX,dirBuffer+ucsdNumFiles
OEBE: BB ****                           LEA     BX,dirBuffer+ucsdDirSize ;start at entry one
OEC1:
OEC1: 42                                INC     DX                      ;check for zero files
OEC2: E9 ****                           JMP     $40
OEC5:                           $30
OEC5: 8D 7F 06                          LEA     DI,(BX+ucsdName)        ;name in directory
OEC8: 5E                                POP     SI                      ;get pointer to name
OEC9: 56                                PUSH    SI                      ;put back on stack
OECA: 8A 4C 00                          MOV     CL,(SI+0)               ;length of name
OECD: FE C1                             INC     CL                      ;must compare length also
OECF: 32 ED                             XOR     CH,CH
OED1: F3 A6                     REPZ    CMPSB                           ;does this entry match
OED3: 74**                              JZ      readU01                 ; yes
OED5:
OED5: 83 C3 1A                          ADD     BX,ucsdDirSize          ;point to next entry
OED8:                           $40
OED8: 4A                                DEC     DX                      ;any more left?
OED9: 75EA                              JNZ     $30
OEDB:
OEDB: 5E                                POP     SI                      ;take off stack
OEDC: 5B                                POP     BX
OEDD: 5F                                POP     DI
OEDE: E8 37F7                           CALL    restRegs
OEF1: F9                                STC                             ;set error return
OEE2: C3                                RET
OEE3:                           readU01
OEE3:                                   ;we have found the entry for the file, we now prepare
OEE3:                                   ;to read it into memory
OEE3:
OEE3: 8B D3                             MOV     DX,BX                   ;save dir entry pointer in DX
```

```
0EE5: 5E                                    POP     SI                      ;trash file name off stack
0EE6: 5B                                    POP     BX                      ;get pointer to dv Rec
0EE7: 8D 77 0A                              LEA     SI,(BX+dvStartBlk)
0EEA: BF B005                               LEA     DI,block
0EED: B9 04 00                              MOV     CX,addLen
0EF0: F3 A4                           REP   MOVSB                           ;move add to block var
0EF2:
0EF2: 32 E4                                 XOR     AH,AH
0EF4: 8A 47 15                              MOV     AL,(BX+dvVolOff)
0EF7: E8 1EFA                               CALL    addToBlock              ;add in volume offset
0EFA:
0EFA: 8B DA                                 MOV     BX,DX                   ;restore pointer to file entry
0EFC: 8B 47 00                              MOV     AX,(BX+ucsdistBlk)      ;1st block of file
0EFF: 50                                    PUSH    AX                      ;save it
0F00: E8 15FA                               CALL    addToBlock              ;add into drive address
0F03: 58                                    POP     AX                      ;restore it
0F04:
0F04: 8B 57 02                              MOV     DX,(BX+ucsdlastBlk)     ;calc number of blocks
0F07:
0F07: 2B D0                                 SUB     DX,AX                   ;DX = number of blocks
0F09: 89 16 6205                            MOV     holdNumRead,DX          ;save number of blocks
0F0D:
0F0D: 5F                                    POP     DI                      ;where to put file
0F0E:                         $30
0F0E: E8 E9FE                               CALL    readBlock
0F11: BE ****                               LEA     SI,buffer
0F14: B9 00 02                              MOV     CX,sectSize
0F17: F3 A4                           REP   MOVSB
0F19:
0F19: E8 1AFA                               CALL    nextBlock
0F1C: 4A                                    DEC     DX                      ;any more blocks to read
0F1D: 75EF                                  JNZ     $30                     ; yes
0F1F:
0F1F: E8 F6F6                               CALL    restRegs
0F22: 8B 16 6205                            MOV     DX,holdNumRead
0F26: F8                                    CLC                             ;set no error
0F27: C3                                    RET
0F28:
0F28:                         ;*************************************************************
0F28:                         ;
0F28:                         ; UCSD_BOOT -- boots up the UCSD 4.0 P-system
0F28:                         ;
0F28:                         ;*************************************************************
0F28:                         ucsdBoot
0F28: BB EC04                               LEA     BX,dvBtRec              ;for read directory routine
0F2B: E8 3AFF                               CALL    readDir
0F2E:
0F2E: BE 4403                               LEA     SI,drvrFileName         ;address of attached driver
0F31: BB EC04                               LEA     BX,dvBtRec
0F34: BF 00 05                              MOV     DI,drvrAdd              ;where to load driver
0F37: E8 7AFF                               CALL    readUCSDfile            ;load in driver
0F3A: 73**                                  JNC     ucsdBt00                ;went ok
0F3C:
0F3C: BE 1902                               LEA     SI,noDrvrMsg            ;no driver file message
0F3F: E8 AAF7                               CALL    printAndWait
```

```
OF42: C3                                RET
OF43:                           ucsdBtOO
OF43: 52                                PUSH    DX                      ;save # blocks loaded
OF44:
OF44: BE 00 05                          MOV     SI,drvrAdd
OF47: 03 74 04                          ADD     SI,(SI+04)              ;SI = address of interupt routine
OF4A: BF C0 03                          MOV     DI,4*0F0H               ;store at interrupt F0H
OF4D: AD                                LODSW
OF4E: AB                                STOSW
OF4F: B8 50 00                          MOV     AX,drvrAdd/10H          ;segment address
OF52: AB                                STOSW                           ;interrupt vector is now fixed
OF53:
OF53: BE 00 05                          MOV     SI,drvrAdd
OF56: 89 36 3905                        MOV     interpAdd,SI
OF5A:
OF5A: 8B 44 02                          MOV     AX,(SI+02)              ;init entry point and interp load
OF5D: 01 06 3905                        ADD     interpAdd,AX
OF61: A3 3B05                           MOV     drvrInitAdd,AX
OF64:
OF64: E8 C0F6                           CALL    home
OF67: FF 1E 3B05                        CALLL   (drvrInitAdd)           ;initalize the driver
OF6B:
OF6B: BE 2403                           LEA     SI,ucsdInterpName       ;file name to load
OF6E: BB EC04                           LEA     BX,dvBtRec              ;volume record
OF71: 8B 3E 3905                        MOV     DI,interpAdd            ;where to load it
OF75: E8 3CFF                           CALL    readUCSDfile            ;read it
OF78: 5A                                POP     DX                      ;number blocks driver read
OF79: 73**                              JNC     ucsdBtO1                ;everything went ok
OF7B:
OF7B: BE F001                           LEA     SI,noInterpMsg          ;error message for user
OF7E: E8 6BF7                           CALL    printAndWait
OF81: C3                                RET
OF82:                           ucsdBtO1
OF82: E8 ****                           CALL    ucsdMount               ;load mount table
OF85: 73**                              JNB     ucsdBtO2                ;went ok
OF87: BE 8603                           LEA     SI,badMntMsg            ;bad mount message
OF8A: E8 5FF7                           CALL    printAndWait
OF8D: C3                                RET
OF8E:                           ucsdBtO2
OF8E:                                   ;the interperter is now loaded into memory
OF8E:                                   ;we must now read the boot parameter file this contains the stack
OF8E:                                   ;and parameters need for initilization by pascal it is maintained
OF8E:                                   ;by the NCI turnkey program
OF8E:
OF8E: BE 3403                           LEA     SI,bootParmFile         ;address of file name
OF91: BB 1805                           LEA     BX,hold5dvRec           ;unit 4 drive volume record
OF94: BF ****                           LEA     DI,bootData             ;where to put it
OF97: A0 3805                           MOV     AL,hold5drive
OF9A: A2 AF05                           MOV     drive,AL
OF9D: E8 C8FE                           CALL    readDir
OFA0:
OFA0: 80 3E B905 00                     CMPBIM  dirIsFlipped,true
OFA5: 75**                              JNE     $05                     ;will not booted with flipped dir
OFA7:
OFA7: E8 7DF6                           CALL    home
```

```
OFAA: BE 8802                       LEA     SI,badUnit5Msg
OFAD: E8 3CF7                       CALL    printAndWait
OFB0: C3                            RET
OFB1:                         $05
OFB1: E8 00FF                       CALL    readUcsdFile
OFB4: 73**                          JNC     $10                         ;read it ok
OFB6:
OFB6: BE E302                       LEA     SI,noBootParmMsg
OFB9: E8 30F7                       CALL    printAndWait
OFBC: C3                            RET
OFBD:                               ; we must init the stack and registers and take off
OFBD:                         $10
OFBD: C7 06 **** 05 00              MOV     bootUnit,5                  ;force boot from unit 5
OFC3:
OFC3: E8 ****                       CALL    ucsdMemCheck                ;stack manip stuff
OFC6: 73**                          JNC     $20
OFC8: C3                            RET                                 ;error return
OFC9:                         $20
OFC9:
OFC9: BE 1D03                       LEA     SI,crLfMsg
OFCC: E8 82F6                       CALL    printStr
OFCF:
OFCF: BE ****                       LEA     SI,boot7F80                 ;move 64 words at 7F80 to
OFD2: BF 10 02                      MOV     DI,210H                     ;210H
OFD5: B9 40 00                      MOV     CX,0040H
OFD8: F3 A5                   REP   MOVSW
OFDA:
OFDA: BE ****                       LEA     SI,boot7C0C                 ;floppy disk paramters
OFDD: BF 00 02                      MOV     DI,200H                     ;store at 200H
OFE0: B9 0B 00                      MOV     CX,0BH                      ;11 bytes
OFE3: F3 A4                   REP   MOVSB                               ;move them
OFE5:
OFE5: BF 78 00                      MOV     DI,1EH*4                    ;vector address
OFE8: B8 00 02                      MOV     AX,200H
OFEB: AB                            STOSW
OFEC: B8 00 00                      MOV     AX,0
OFEF: AB                            STOSW
OFF0:
OFF0: 32 E4                         XOR     AH,AH
OFF2: CD 13                         INT     13H                         ;initalize the diskette system
OFF4:
OFF4: FA                            CLI                                 ;disable interrupts
OFF5:
OFF5: 8B 1E 3905                    MOV     BX,interpAdd                ;set stack at interp base
OFF9:
OFF9: BC ****                       LEA     SP,bootUnit                 ;this is in boot parm file
OFFC: 2B E3                         SUB     SP,BX                       ;adjust for different seg base
OFFE:
OFFE: B1 04                         MOV     CL,4                        ;divide interp add by 16.
1000: D3 EB                         SHR     BX,CL
1002:
1002: 8E D3                         MOV     SS,BX                       ;this is interperters base
1004: FB                            STI                                 ;enable interrupts
1005:
1005: 8E DB                         MOV     DS,BX
```

```
1007: 8E C3                         MOV     ES,BX
1009:
1009: 1E                            PUSH    DS
100A:
100A: 33 DB                         XOR     BX,BX                   ;1st byte of interp contains
100C:                                                               ;entry point
100C: 8B 07                         MOV     AX,(BX)                 ;ax = entry point
100E: 50                            PUSH    AX                      ;onto stack
100F: CB                            RETL                            ;boot!! round about way
1010:
1010:
1010:                       ;*******************************************************************
1010:                       ;
1010:                       ; UCSD_MEM_CHECK -- this code is ripped off straight from the NCI boot
1010:                       ;           for the p-system, some of it i dont understand what they are
1010:                       ;           are doing. This was from a disassembly
1010:                       ;
1010:                       ;*******************************************************************
1010:                       ucsdMemCheck
1010: CD 12                             INT     12H                 ;determine memory size
1012:
1012: 3D 40 00                          CMP     AX,64.
1015: BB FE FF                          MOV     BX,-2               ;highest address
1018: 75**                              JNZ     $10                 ;not exactly 64k
101A: 2B 1E 3905                        SUB     BX,interpAdd        ;subtrace address of interperter
101E: E9 ****                           JMP     $20
1021:
1021:                       $10
1021: 8B 16 ****                        MOV     DX,boot7F70
1025: B1 04                             MOV     CL,4
1027: D3 E2                             SHL     DX,CL
1029:
1029: B9 00 08                          MOV     CX,0800H            ;we have over 64k
102C: 3B D1                             CMP     DX,CX               ;800-interp address
102E: 77**                              JA      $15
1030: 8B CA                             MOV     CX,DX
1032:                       $15
1032: 2B 0E 3905                        SUB     CX,interpAdd
1036: 73**                              JAE     $20
1038: 03 D9                             ADD     BX,CX
103A:                       $20
103A: 89 1E ****                        MOV     boot7F6E,BX         ;save highest address interp relative
103E: C7 06 **** FF FF                  MOV     boot7F74,0FFFFH
1044: F8                                CLC
1045: C3                                RET
1046:
1046:
1046:                       ;*******************************************************************
1046:                       ;
1046:                       ; UCSD_MOUNT -- finds the mount table in the interperter just loaded into memory
1046:                       ;           and patches in the info for the boot volume
1046:                       ;
1046:                       ;*******************************************************************
1046:                       ucsdMount
1046: B1 09                             MOV     CL,9                ;convert num blks to bytes
```

```
1048: D3 E2                         SHL     DX,CL                   ;DX = num bytes to search
104A: B8 50 00                      MOV     AX,drvrAdd/10H
104D: 8E C0                         MOV     ES,AX                   ;search at interp base
104F: BB 00 00                      MOV     BX,0                    ;start looking here
1052:                       $10
1052: BE 7103                       LEA     SI,memMark              ;searching for this in memory
1055: 8B FB                         MOV     DI,BX                   ;we are looking here
1057: B9 0F 00                      MOV     CX,memMarkLen
105A: F3 A6                   REPZ  CMPSB                           ;have we found it
105C: 74**                          JZ      $20                     ; yes
105E: 43                            INC     BX
105F: 4A                            DEC     DX
1060: 75F0                          JNZ     $10                     ;look again
1062:
1062: 8C D8                         MOV     AX,DS                   ;restore ES reg
1064: 8E C0                         MOV     ES,AX
1066:
1066: F9                            STC                             ;set error return
1067: C3                            RET
1068:                       $20                                     ;found Mnt tbl
1068: 83 C3 0F                      ADD     BX,memMarkLen           ;go past marker
106B: 89 1E 1405                    MOV     mntTabAdd,BX            ;save addresses
106F: 8C C3                         MOV     BX,ES
1071: 89 1E 1605                    MOV     mntTabSeg,BX
1075:
1075: 8C D8                         MOV     AX,DS                   ;restore ES
1077: 8E C0                         MOV     ES,AX
1079:
1079: B0 04                         MOV     AL,4
107B: E8 ****                       CALL    ucsdMntFind             ;find and mount unit 4
107E:
107E: B0 05                         MOV     AL,5
1080: E8 ****                       CALL    ucsdMntFind             ;  "   "   "     "   5
1083: 73**                          JNC     $30                     ;unit five was found
1085:
1085: BE 7202                       LEA     SI,noUnit5msg           ;must be mounted
1088: E8 61F6                       CALL    printAndWait
108B: 58                            POP     AX                      ;return to outermost level
108C: C3                            RET
108D:                       $30
108D: BE CC04                       LEA     SI,dvRec                ;save unit 9 dvRec
1090: BF 1805                       LEA     DI,hold5dvRec
1093: B9 20 00                      MOV     CX,dvRecSize
1096: F3 A4                   REP   MOVSB
1098:
1098: A0 AF05                       MOV     AL,drive
109B: A2 3805                       MOV     hold5drive,AL
109E:
109E: B0 09                         MOV     AL,9
10A0: E8 ****                       CALL    ucsdMntFind             ;  "   "   "     "   9
10A3:
10A3: B0 0A                         MOV     AL,10.
10A5: E8 ****                       CALL    ucsdMntFind             ;  "   "   "     "   10
10A8:
10A8: B0 0B                         MOV     AL,11.
```

```
10AA: E8 ****                           CALL    ucsdMntFind                     ;  "   "   "     "   11
10AD:
10AD: B0 0C                             MOV     AL,12.
10AF: E8 ****                           CALL    ucsdMntFind                     ;  "   "   "     "   12
10B2:
10B2: 8B 3E 1405                        MOV     DI,mntTabAdd
10B6: 06                                PUSH    ES
10B7: 8E 06 1605                        MOV     ES,mntTabSeg
10BB:
10BB: BE 7A05                           LEA     SI,cryptName                    ;put user name in mount table
10BE: B9 0A 00                          MOV     CX,nameLen
10C1: F3 A4                       REP   MOVSB
10C3:
10C3: BE 8405                           LEA     SI,cryptPassword                ;put password in mount table
10C6: B9 08 00                          MOV     CX,passLen
10C9: F3 A4                       REP   MOVSB
10CB:
10CB: A0 A305                           MOV     AL,serverHost                   ;put disk server number in table
10CE: AA                                STOSB
10CF:
10CF: 07                                POP     ES                              ;restore ES
10D0: F8                                CLC
10D1: C3                                RET
10D2:
10D2:                           ;*******************************************************************
10D2:                           ; UCSD_MNT_FIND -- finds and mounts the UCSD unit in AL
10D2:                           ;
10D2:                           ;*******************************************************************
10D2:                           ucsdMntFind
10D2: BB ****                           LEA     BX,mntDrvTab
10D5: 89 1E 1005                        MOV     mntDrvPoint,BX
10D9:
10D9: BB ****                           LEA     BX,mntDvTab
10DC: 89 1E 0C05                        MOV     mntDvPoint,BX
10E0:
10E0: BB ****                           LEA     BX,mntDaTab
10E3:
10E3: 8B 0E 1205                        MOV     CX,mntCount
10E7:                           $10
10E7: 38 47 06                          CMP     (BX+daMntInfo),AL               ;is this the correct unit
10EA: 74**                              JE      $20
10EC:
10EC: FF 06 1005                        INC     mntDrvPoint
10F0: 83 06 0C05 20                     ADD     mntDvPoint,dvRecSize
10F5: 83 C3 08                          ADD     BX,daRecSize
10F8: E2ED                              LOOP    $10                             ;loop until searched all
10FA: E9 ****                           JMP     $30                             ;didnt find it
10FD:                           $20
10FD: 8B F3                             MOV     SI,BX
10FF: BF A404                           LEA     DI,daRec
1102: B9 08 00                          MOV     CX,daRecSize
1105: F3 A4                          REP  MOVSB
1107:
1107: 8B 36 0C05                        MOV     SI,mntDvPoint
```

```
110B: BF CC04                       LEA     DI,dvRec
110E: B9 20 00                      MOV     CX,dvRecSize
1111: F3 A4                   REP   MOVSB
1113:
1113: 8B 1E 1005                    MOV     BX,mntDrvPoint
1117: 8A 07                         MOV     AL,(BX)
1119: A2 AF05                       MOV     drive,AL
111C:
111C: E8 ****                       CALL    ucsdMntOne
111F: F8                            CLC                                     ;set unit found
1120: 73**                          JNC     $40                             ;always
1122:                         $30
1122: 06                            PUSH    ES
1123: 8B 3E 1405                    MOV     DI,mntTabAdd                    ;zero entry nothing is mounted
1127: 8E 06 1605                    MOV     ES,mntTabSeg
112B: 33 C0                         XOR     AX,AX
112D: B9 08 00                      MOV     CX,mntRecSize
1130: F3 AA                    REP  STOSB
1132: 07                            POP     ES
1133: F9                            STC                                     ;set mount not found
1134:                         $40
1134: 9C                            PUSHF
1135: 83 06 1405 08                 ADD     mntTabAdd,mntRecSize            ;set pointer to next entry
113A: 9D                            POPF
113B: C3                            RET
113C:
113C:                         ;*****************************************************************
113C:                         ;
113C:                         ; UCSD_MNT_ONE -- creates one UCSD mount entry at mntTabSeg:(mntTabAdd)
113C:                         ;
113C:                         ;*****************************************************************
113C:                         ucsdMntOne
113C: E8 CAF4                         CALL    saveRegs
113F: 06                              PUSH    ES
1140: 8B 1E 1405                      MOV     BX,mntTabAdd
1144: A1 1605                         MOV     AX,mntTabSeg
1147: 8E C0                           MOV     ES,AX
1149:
1149: 8A 26 AF05                      MOV     AH,drive                        ;build mount table
114D: B1 04                           MOV     CL,4
114F: D2 E4                           SHL     AH,CL
1151:
1151: 8A 2E DF04                      MOV     CH,dvRec+dvWriteable            ;write protect stuff
1155: 80 E5 0F                        AND     CH,0FH
1158:
1158: F6 06 A804 10                   TESTBIM daRec+daReadBoot,daRoFlag       ;is it read only
115D: 74**                            JZ      $10                             ;no
115F: 32 ED                           XOR     CH,CH                           ;set read only
1161:                         $10
1161: 0A E5                           OR      AH,CH                           ;or into drive
1163:
1163: 26 88 67 00                     MOV     ES:(BX+mntDrvWp),AH             ;move into mount table
1167:
1167: 8A 26 A305                      MOV     AH,serverHost                   ;put in disk server add
```

```
116B: 26 88 67 01                   MOV     ES:(BX+mntDskSrvr),AH
116F:
116F: A1 D804                       MOV     AX,dvRec+dvStartBlk+2           ;put volume add in mount table
1172: 26 89 47 04                   MOV     ES:(BX+mntBlock+2),AX
1176: A1 D604                       MOV     AX,dvRec+dvStartBlk
1179: 26 89 47 02                   MOV     ES:(BX+mntBlock),AX
117D:
117D: A1 DC04                       MOV     AX,dvRec+dvEndBlk+2             ;put num blocks in mount table
1180: 86 E0                         XCHG    AH,AL
1182: 8B 0E D804                    MOV     CX,dvRec+dvStartBlk+2
1186: 86 E9                         XCHG    CH,CL
1188:
1188: 2B C1                         SUB     AX,CX
118A: 40                            INC     AX
118B: 86 E0                         XCHG    AH,AL
118D: 26 89 47 06                   MOV     ES:(BX+mntNumBlks),AX
1191:
1191: 07                            POP     ES
1192: E8 83F4                       CALL    restRegs
1195: C3                            RET
1196:
1196:                               .INCLUDE CONSTBOOT2.TEXT                ;MSDOS BOOT STUFF
1196:                       ;*******************************************************
1196:                       ;
1196:                       ; MS_DOS_BOOT -- boots up the IBM/PC MS-DOS operating system
1196:                       ;
1196:                       ;*******************************************************
1196:                       msDosBoot
1196: BB EC04                           LEA     BX,dvBtRec              ;for reading directory
1199: E8 CCFC                           CALL    readDir
119C:
119C: BB EC04                           LEA     BX,dvBtRec
119F: BE 5403                           LEA     SI,msDosBiosName        ;file to load
11A2: BF 00 06                          MOV     DI,biosAdd
11A5: E8 0CFD                           CALL    readUCSDfile            ;read it into memory
11A8: 73**                              JNC     msDosBt1                ;everything went ok
11AA:
11AA: BE 3D02                           LEA     SI,noBiosMsg            ;error message for user
11AD: E8 3CF5                           CALL    printAndWait
11B0: C3                                RET
11B1:                       msDosBt1
11B1: FF 36 AF05                        PUSH    drive
11B5: E8 ****                           CALL    msDosMount              ;load mount table
11B8:
11B8: 58                                POP     AX                      ;restore drive
11B9: A2 AF05                           MOV     drive,AL
11BC:
11BC: 73**                              JNC     msDosBt2                ;went ok
11BE: BE B203                           LEA     SI,badBiosMsg           ;bad mount message
11C1: E8 28F5                           CALL    printAndWait            ;print it
11C4: C3                                RET                             ;and hang
11C5:                       msDosBt2
11C5: BB EC04                           LEA     BX,dvBtRec
11C8: BE 6303                           LEA     SI,msDosName
11CB: BF 00 06                          MOV     DI,biosAdd
```

```
11CE: 03 3E 4105                        ADD     DI,biosLen              ;load right after bios
11D2: E8 DFFC                           CALL    readUCSDfile            ;read it into memory
11D5: 73**                              JNC     msDosBt3                ;all ok
11D7:
11D7: BE 5802                           LEA     SI,noDosMsg
11DA: E8 0FF5                           CALL    printAndWait
11DD: C3                                RET
11DE:
11DE:                   msDosBt3
11DE:                                   ;the bios is now loaded into memory
11DE:                                   ; we must init the stack and registers and take off
11DE:
11DE: E8 46F4                           CALL    home
11E1:
11E1: 80 3E 5B05 00                     CMPBIM  aMounted,true           ;is 'A' mounted?
11E6: 74**                              JE      $10                     ;yes
11E8: BE C402                           LEA     SI,putFlopInMsg
11EB: E8 FEF4                           CALL    printAndWait
11EE: E8 36F4                           CALL    home
11F1:                   $10
11F1: BB 60 00                          MOV     BX,biosAdd/10H                  ;CS reg for interp
11F4: 8E DB                             MOV     DS,BX
11F6: 8E C3                             MOV     ES,BX
11F8: EA 0000 6000                      JMPL    0,biosAdd/10H                   ;go to it
11FD:
11FD:                   ;*****************************************************************
11FD:                   ;
11FD:                   ; MSDOS_MOUNT -- finds the mount table in the bios just loaded into memory
11FD:                   ;           and patches in the info for mounted volumes
11FD:                   ;
11FD:                   ;*****************************************************************
11FD:                   msDosMount
11FD: B1 09                             MOV     CL,9                            ;convert num blks to bytes
11FF: D3 E2                             SHL     DX,CL                           ;DX = num bytes to search
1201: B8 60 00                          MOV     AX,biosAdd/10H
1204: 8E C0                             MOV     ES,AX                           ;search at interp base
1206: BB 00 00                          MOV     BX,0                            ;start looking here
1209:                   $10
1209: BE 8003                           LEA     SI,DosMark                      ;searching for this in memory
120C: 8B FB                             MOV     DI,BX                           ;we are looking here
120E: B9 06 00                          MOV     CX,dosMarkLen
1211: F3 A6                       REPZ  CMPSB                                   ;have we found it
1213: 74**                              JZ      $20                             ; yes
1215: 43                                INC     BX
1216: 4A                                DEC     DX
1217: 75F0                              JNZ     $10                             ;look again
1219:
1219: 8C D8                             MOV     AX,DS                           ;restore ES reg
121B: 8E C0                             MOV     ES,AX
121D:
121D: F9                                STC                                     ;set error return
121E: C3                                RET
121F:                   $20                                                     ;found Mnt tbl
121F: 8C C0                             MOV     AX,ES
1221: 8E D8                             MOV     DS,AX
```

```
1223: 2E A3 3F05                MOV     CS:dosSeg,AX
1227:
1227: 83 C3 09                  ADD     BX,dosMarkLen+3         ;go past marker
122A: 8B F3                     MOV     SI,BX
122C:
122C:
122C: AD                        LODSW
122D: 2E A3 4505                MOV     CS:sConfigAdd,AX        ;address of config tables
1231: 8B F8                     MOV     DI,AX
1233: 50                        PUSH    AX
1234: B0 00                     MOV     AL,0                    ;zero out number in use
1236: AA                        STOSB
1237: 58                        POP     AX
1238:
1238: 40                        INC     AX
1239: 2E A3 4305                MOV     CS:configAdd,AX
123D:
123D: AD                        LODSW
123E: 2E A3 4705                MOV     CS:fParmAdd,AX          ;address of floppy disk parm tables
1242:
1242: AD                        LODSW
1243: 2E A3 4905                MOV     CS:cParmAdd,AX          ;address of corvus parm tables
1247: 8B F8                     MOV     DI,AX                   ;byte before is number of entries
1249: 4F                        DEC     DI                      ;in table used
124A: 2E 89 3E 4B05             MOV     CS:cParmCadd,DI
124F: 32 C0                     XOR     AL,AL
1251: AA                        STOSB                           ;init it to zero
1252:
1252: AD                        LODSW
1253: 2E A3 4D05                MOV     CS:volAdd,AX            ;address of volume table
1257:
1257: AD                        LODSW
1258: 2E A3 1405                MOV     CS:mntTabAdd,AX         ;address of mount table
125C:
125C: 83 C6 06                  ADD     SI,6
125F: AD                        LODSW
1260: 2E A3 4F05                MOV     CS:mapAdd,AX            ;logical unit mapping table
1264:
1264: 2E A0 7905                MOV     AL,CS:userHost          ;store xporter number
1268: 8B FE                     MOV     DI,SI
126A: AA                        STOSB
126B:
126B: 46                        INC     SI
126C: AD                        LODSW                           ;address of user name
126D: 8B F8                     MOV     DI,AX                   ;user name add in DI
126F:
126F: 83 C6 02                  ADD     SI,2                    ;point to bios len in blocks
1272: AC                        LODSB
1273: 86 E0                     XCHG    AH,AL
1275: D0 E4                     SHL     AH,1
1277: 32 C0                     XOR     AL,AL
1279: 2E A3 4105                MOV     CS:biosLen,AX           ;bios len in bytes
127D:
127D: 83 C6 02                  ADD     SI,2                    ;point to disk server byte
1280: 57                        PUSH    DI
```

```
12B1: 2E A0 A305                    MOV     AL,CS:serverHost
12B5: 8B FE                         MOV     DI,SI
12B7: AA                            STOSB
12B8: 5F                            POP     DI
12B9:
12B9: BE 7A05                       LEA     SI,cryptName                ;DI already set up
12BC: B9 0A 00                      MOV     CX,nameLen
12BF: 8C C8                         MOV     AX,CS
12C1: 8E D8                         MOV     DS,AX                       ;restore DS
12C3: F3 A4                   REP   MOVSB                               ;move name to interperter
12C5:
12C5: 8C C8                         MOV     AX,CS
12C7: 8E C0                         MOV     ES,AX                       ;restore ES
12C9:
12C9: 33 C0                         XOR     AX,AX
12CB: A3 5105                       MOV     logUnit,AX                  ;for logical mapping table
12CE: A3 5305                       MOV     mountCount,AX               ;number of drive mounted
12D1: A3 5505                       MOV     flopCount,AX                ;number of floppys mounted
12D4: A3 5705                       MOV     mapCount,AX                 ;index into config table for map
12D7: A3 5905                       MOV     maxFlop,AX                  ;number of actual floppies
12DA: C6 06 5B05 00                 MOVBIM  aMounted,true               ;assume a is mounted
12DF:
12DF: CD 11                         INT     11H                         ;equipment config call
12E1: A8 01                         TEST    AL,1                        ;any floppies attached?
12E3: 74**                          JZ      $30                         ;no
12E5: D0 C0                         ROL     AL,1                        ;shift num floppies around
12E7: D0 C0                         ROL     AL,1
12E9: 25 03 00                      AND     AX,3                        ;mask out other bits
12EC: 40                            INC     AX                          ;make 1-4
12ED: 3D 01 00                      CMP     AX,1                        ;is there just one fBoppy
12F0: 75**                          JNE     $25                         ; no
12F2: 40                            INC     AX                          ;if just one physical floppy mount
12F3:                         $25                                       ;two so user can copy floppies
12F3: A3 5905                       MOV     maxFlop,AX                  ;save it
12F6:                         $30
12F6: FF 06 5105                    INC     logUnit
12FA: 83 3E 5105 0B                 CMP     logUnit,11.                 ;are we all done
12FF: 74**                          JE      $40                         ;yes, have mounted 'a'-'j'
1301: E8 ****                       CALL    dosMntFind
1304: EBF0                          JMP     $30
1306:                         $40
1306: F8                            CLC
1307: C3                            RET
1308:
1308:                         ;*****************************************************************
1308:                         ;
1308:                         ; DOS_MNT_FIND -- finds and mounts the DOS unit in logUnit
1308:                         ;
1308:                         ;*****************************************************************
1308:                         dosMntFind
1308: BB ****                           LEA     BX,mntDrvTab
130B: 89 1E 1005                        MOV     mntDrvPoint,BX
130F:
130F: BB ****                           LEA     BX,mntDvTab
1312: 89 1E 0C05                        MOV     mntDvPoint,BX
```

```
12E6:
12E6: BB ****                       LEA     BX,mntDaTab
12E9:
12E9: 8B 0E 1205                    MOV     CX,mntCount
12ED:                         $10
12ED: A1 5105                       MOV     AX,logUnit
12F0: 38 47 06                      CMP     (BX+daMntInfo),AL           ;is this the correct unit
12F3: 74**                          JE      $20
12F5:
12F5: FF 06 1005                    INC     mntDrvPoint
12F9: 83 06 0C05 20                 ADD     mntDvPoint,dvRecSize
12FE: 83 C3 08                      ADD     BX,daRecSize
1301: E2EA                          LOOP    $10                         ;loop until searched all
1303: E9 ****                       JMP     $30                         ;didnt find it
1306:                         $20
1306: 8B F3                         MOV     SI,BX
1308: BF A404                       LEA     DI,daRec
130B: B9 08 00                      MOV     CX,daRecSize
130E: F3 A4                    REP  MOVSB
1310:
1310: 8B 36 0C05                    MOV     SI,mntDvPoint
1314: BF CC04                       LEA     DI,dvRec
1317: B9 20 00                      MOV     CX,dvRecSize
131A: F3 A4                    REP  MOVSB
131C:
131C: 80 3E DE04 03                 CMPBIM  dvRec+dvVolType,msDosVol    ;is this a msDos volume
1321: 75**                          JNE     $30                         ;no
1323: 8B 1E 1005                    MOV     BX,mntDrvPoint
1327: 8A 07                         MOV     AL,(BX)
1329: A2 AF05                       MOV     drive,AL
132C:
132C: E8 ****                       CALL    dosMntOne
132F: C3                            RET
1330:                         $30
1330: 83 3E 5105 01                 CMP     logUnit,1                   ;is this drive 'A'
1335: 75**                          JNE     $35                         ;NO
1337: C6 06 5B05 01                 MOVBIM  aMounted,false
133C:                         $35
133C: E8 ****                       CALL    dosMntFloppy
133F: C3                            RET
1340:                         ;*****************************************************************************
1340:                         ;
1340:                         ; DOS_MNT_FLOPPY
1340:                         ;
1340:                         ;*****************************************************************************
1340:                         dosMntFloppy
1340: 06                            PUSH    ES
1341: 8E 06 3F05                    MOV     ES,dosSeg
1345: 8B 3E 1405                    MOV     DI,mntTabAdd                ;get ready to store in DOS
1349:
1349: A1 5505                       MOV     AX,flopCount                ;floppys mounted so far
134C: 3B 06 5905                    CMP     AX,maxFlop                  ;any left
1350:                               ;JE      $10                         ;no
1350: FE C0                         INC     AL
1352: 0C 40                         OR      AL,040H                     ;put in r/w and floppy
```

```
1354:
1354: AA                        STOSB
1355:
1355: 8B 1E 4505                MOV     BX,sConfigAdd           ;num of entries in config table
1359: 26 80 07 02                ADDBIM  ES:(BX),2               ;a floppy has two entries
135D:
135D: 8B 3E 4305                MOV     DI,configAdd            ;next entry in config table
1361: A1 5105                    MOV     AX,logUnit
1364: FE C8                      DEC     AL                      ;access info is 1-x
1366: AA                         STOSB
1367: 8A D8                      MOV     BL,AL                   ;save it
1369: A1 4705                    MOV     AX,fParmAdd             ;address of ss floppy parms
136C: AB                         STOSW
136D: 8B C8                      MOV     CX,AX                   ;save it
136F: 8A C3                      MOV     AL,BL                   ;each floppy has two entries
1371: AA                         STOSB
1372: 8B C1                      MOV     AX,CX
1374: 05 0A 00                   ADD     AX,10.                  ;point to double sided entry
1377: AB                         STOSW
1378: 83 06 4305 06              ADD     configAdd,6             ;point to next entry in table
137D:
137D: 8B 3E 4F05                MOV     DI,mapAdd
1381: A1 5705                    MOV     AX,mapCount             ;index into config table for
1384: AA                         STOSB                           ;this drive
1385: 83 06 5705 02              ADD     mapCount,2
138A:
138A: FF 06 5505                 INC     flopCount               ;number of floppys mounted
138E: E9 ****                    JMP     $20
1391:                           $10
1391: B0 00                      MOV     AL,0                    ;unmount unit
1393: AA                         STOSB
1394:                           $20
1394: FF 06 1405                 INC     mntTabAdd
1398: FF 06 4F05                 INC     mapAdd
139C: 07                         POP     ES
139D: C3                         RET
139E:
139E:                           ;*****************************************************************
139E:                           ;
139E:                           ; DOS_MNT_ONE -- mounts the logical unit in logUnit
139E:                           ;
139E:                           ;*****************************************************************
139E:                           dosMntOne
139E: 06                         PUSH    ES
139F: 8E 06 3F05                 MOV     ES,dosSeg
13A3:
13A3: FF 06 5305                 INC     mountCount              ;number of corvus's mounted
13A7: A1 5305                    MOV     AX,mountCount
13AA: 8A 1E DF04                 MOV     BL,dvRec+dvWriteable    ;=1 if writeable
13AE: F6 06 A804 10              TESTBIM daRec+daReadBoot,daRoFlag       ;is this read only
13B3: 74**                       JZ      $1                      ;no
13B5: B3 00                      MOV     BL,0                    ;mark as read only
13B7:                           $1
13B7: 82 FB 01                   CMP     BL,1                    ;is it writeable
13BA: 74**                       JE      $2                      ;yes
```

```
13BC: 0C 80                         OR      AL,80H                      ;mark as write protect
13BE:                         #2
13BE: 8B 3E 1405                    MOV     DI,mntTabAdd                ;put in DOS mount table
13C2: AA                            STOSB
13C3: FF 06 1405                    INC     mntTabAdd                   ;point to next entry
13C7:
13C7: 8B 3E 4D05                    MOV     DI,volAdd                   ;point to DOS volume table
13CB: A0 D904                       MOV     AL,dvRec+dvStartBlk+3
13CE: 02 06 E104                    ADD     AL,dvRec+dvVolOff
13D2: AA                            STOSB
13D3: A0 D804                       MOV     AL,dvRec+dvStartBlk+2
13D6: 14 00                         ADC     AL,0
13D8: AA                            STOSB
13D9: A0 D704                       MOV     AL,dvRec+dvStartBlk+1
13DC: 14 00                         ADC     AL,0
13DE: AA                            STOSB
13DF: A0 AF05                       MOV     AL,drive
13E2: AA                            STOSB
13E3: A0 A305                       MOV     AL,serverHost
13E6: AA                            STOSB
13E7: 83 06 4D05 05                 ADD     volAdd,5                    ;point to next entry in table
13EC:
13EC: 8B 1E 4505                    MOV     BX,sConfigAdd               ;number of entries in use
13F0: 26 83 07 02                   ADD     ES:(BX),2
13F4:
13F4: 8B 3E 4305                    MOV     DI,configAdd                ;next entry in config table
13F8: A1 5105                       MOV     AX,logUnit                  ;logical unit
13FB: FE C8                         DEC     AL
13FD: 8B 1E 4905                    MOV     BX,cParmAdd                 ;pointer to disk parameter table
1401: AA                            STOSB                               ;store logical unit
1402: 93                            XCHG    AX,BX
1403: AB                            STOSW                               ;store pointer to parameter table
1404: 93                            XCHG    AX,BX
1405: AA                            STOSB
1406: 93                            XCHG    AX,BX
1407: AB                            STOSW
1408: 83 06 4305 06                 ADD     configAdd,6                 ;length of 2 entries
140D:
140D: 8B 3E 4F05                    MOV     DI,mapAdd                   ;address of mapping table
1411: A1 5705                       MOV     AX,mapCount                 ;index into config table
1414: AA                            STOSB
1415:
1415: FF 06 4F05                    INC     mapAdd                      ;point to next entry
1419: 83 06 5705 02                 ADD     mapCount,2
141E:
141E: 07                            POP     ES
141F: BE D604                       LEA     SI,dvRec+dvStartBlk
1422: BF B005                       LEA     DI,block
1425: B9 04 00                      MOV     CX,addLen
1428: F3 A4                   REP   MOVSB
142A:
142A: B8 03 00                      MOV     AX,3                        ;parameter table info is in block 3
142D: E8 E8F4                       CALL    addToBlock
1430: E8 C7F9                       CALL    readBlock                   ;read block with parm info
1433:
```

```
1433: 06                        PUSH    ES
1434: 8E 06 3F05                MOV     ES,dosSeg
1438:
1438: BE ****                   LEA     SI,buffer
143B: 8B 3E 4905                MOV     DI,cParmAdd
143F:
143F: AD                        LODSW                           ;sector size in bytes
1440: AB                        STOSW
1441:
1441: AD                        LODSW                           ;sectors per block
1442: AA                        STOSB
1443:
1443: AD                        LODSW                           ;reserved sectors
1444: AB                        STOSW
1445:
1445: AD                        LODSW                           ;number of FAT's                ;length of 2 entries
1446:
1446: 8B 3E 4F05                MOV     DI,mapAdd               ;address of mapping table
144A: A1 5705                   MOV     AX,mapCount             ;index into config table
144D: AA                        STOSB
144E:
144E: FF 06 4F05                INC     mapAdd                  ;point to next entry
1452: 83 06 5705 02             ADD     mapCount,2
1457:
1457: 07                        POP     ES
1458: BE D604                   LEA     SI,dvRec+dvStartBlk
145B: BF B005                   LEA     DI,block
145E: B9 04 00                  MOV     CX,addLen
1461: F3 A4                 REP MOVSB
1463:
1463: B8 03 00                  MOV     AX,3                    ;parameter table info is in block 3
1466:
1466:
1466: 0140              mntDvSize   .EQU dvRecSize*mntMax
1466: 0050              mntDaSize   .EQU daRecSize*mntMax
1466: 000A              mntDrvSize  .EQU mntMax
1466:
1466: 1466              buffer      .EQU    $                   ;used for read routines
1466: 1666              holdBuffer  .EQU buffer+512.            ;used for nested table searches
1466:
1466: 1866              mntDvTab    .EQU holdBuffer+512.        ;hold mounted drive volume tables
1466: 19A6              mntDaTab    .EQU mntDvTab+mntDvSize     ; "      "      "   access   "
1466: 19F6              mntDrvTab   .EQU mntDaTab+mntDaSize     ; "      "      "            "
1466:
1466: 1A00              bootData    .EQU mntDrvTab+mntDrvSize   ;will read IBM.UCSD.BTPARM here
1466: 1D66              bootUnit    .EQU bootData+366H          ;boot unit in boot.data fie
1466: 1A0C              boot7C0C    .EQU bootData+0CH           ;floppy disk paramters in boot
1466: 1D6E              boot7F6E    .EQU bootData+36EH          ;dont know what these are boot the
1466: 1D70              boot7F70    .EQU bootData+370H          ;from NCI uses them .. something
1466: 1D72              boot7F72    .EQU bootData+372H          ;to do with the 'turnkey' program
1466: 1D74              boot7F74    .EQU bootData+374H          ; and the memory configuration
1466: 1D80              boot7F80    .EQU bootData+380H
1466:
1466: 1E00              dirBuffer   .EQU bootData+1024.
1466:                           .END
```

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

```
ADDACTIV LB 05F7  ADDACTNA LB 05F9  ADDLEN   AB 0004  ADDTOBLO LB 0918  AMOUNTED LB 055B  BADBIOSM LB 0382  BADDIMSG LB 0116
BADMNTMS LB 0386  BADOSMSG LB 01A9  BADUNIT5 LB 0288  BELL     AB 0007  BIOSADD  AB 0600  BIOSLEN  LB 0541  BLOCK    LB 0580
BOOT     LB 0E53  BOOT7COC LB 1A0C  BOOT7F6E LB 1D6E  BOOT7F70 LB 1D70  BOOT7F72 LB 1D72  BOOT7F74 LB 1D74  BOOT7F80 LB 1D80
BOOTDATA LB 1A00  BOOTPARM LB 0334  BOOTSTAR LB 003D  BOOTUNIT LB 1D66  BS       AB 0008  BUFFER   LB 1466  CARDTYPE LB 05A1
CHKNUMDR LB 08D5  CIIBOOT  PR ----  CONFIGAD LB 0543  CONSTBOO LB 0000  CONSTMSG LB 007C  CPARMADD LB 0549  CPARMCAD LB 054B
CR       AB 000D  CRLFMSG  LB 031D  CRYPTNAM LB 057A  CRYPTPAS LB 0584  DABOOTFL AB 0001  DAEND    AB 0008  DAMNTINF AB 0006
DAREADBO AB 0007  DAREC    LB 04A4  DARECSIZ AB 0008  DAROFLAG AB 0010  DARPB    AB 0040  DATABLE  AB 0000  DAUSERID AB 0000
DAVOLADD AB 0002  DECRYPTN LB 077D  DEFATTR  AB 0007  DIADD    LB 0320  DIBOOTAD AB 002C  DICRVADD AB 0024  DICRVSIZ AB 0028
DIDAADDR AB 0052  DIDABLKS AB 0040  DIDRIVE  AB 0000  DIDRIVEP AB 000A  DIDRVINI AB 0034  DIDRVNO  AB 0036  DIDUADDR AB 004E
DIDUBLKS AB 003E  DIDVADDR AB 004A  DIDVBLKS AB 0038  DIEND    AB 005B  DINAADD  AB 0030  DINUADDR AB 0046  DINUBLKS AB 003A
DIONLINE AB 0038  DIRBUFFE LB 1E00  DIREC    LB 0429  DIRECSIZ AB 005B  DIRES1   AB 0035  DIRES2   AB 0039  DIRISFLI LB 05B9
DIRPB    AB 0001  DISBADDR AB 0056  DISBBLKS AB 0042  DISERVER AB 0012  DISERVPA AB 001C  DISPECIA AB 0044  DITABLE  AB 0000
DIVERSIO AB 005A  DOSMARK  LB 0380  DOSMARKL AB 0006  DOSMNTFI LB 12D8  DOSMNTFL LB 1340  DOSMNTON LB 139E  DOSSEG   LB 053F
DOTOFF   AB 0001  DOTON    AB 0000  DOTS     LB 05A2  DRIVE    LB 05AF  DRVCMD   LB 05BB  DRVRADD  AB 0500  DRVRFILE LB 0344
DRVRINIT LB 053B  DRVSONLI LB 05A0  DUACCESS AB 0016  DUBOOTIN AB 0018  DUEND    AB 0020  DUMOUNTE AB 0014  DUPASS   AB 000A
DUREC    LB 04AC  DURECSIZ AB 0020  DURES1   AB 001A  DURPB    AB 0010  DUTABLE  AB 0000  DUUSER   AB 0000  DUUSERID AB 0012
DVBTREC  LB 04EC  DVEND    AB 0020  DVENDBLK AB 000E  DVGLBACC AB 0014  DVREC    LB 04CC  DVRECSIZ AB 0020  DVRES1   AB 0016
DVRPB    AB 0010  DVSTARTB AB 000A  DVTABLE  AB 0000  DVVOLNAM AB 0000  DVVOLOFF AB 0015  DVVOLTYP AB 0012  DVWRITEA AB 0013
ECHO     AB 0000  ENCRYPTN LB 0736  ENCRYPTP LB 073F  ERROK    LB 05BA  FALSE    AB 0001  FINDBTO1 LB 0ADF  FINDBTO2 LB 0AF4
FINDBTO3 LB 0B3B  FINDBTO4 LB 0B6F  FINDBTO5 LB 0B95  FINDBTO6 LB 0BCC  FINDBTO7 LB 0BE9  FINDBTO8 LB 0C18  FINDBTDI LB 0AD7
FINDMTO1 LB 0C42  FINDMTO2 LB 0C54  FINDMTO3 LB 0C98  FINDMTO4 LB 0CCF  FINDMTO5 LB 0CEC  FINDMTO6 LB 0D2F  FINDMTO7 LB 0D6B
FINDMTO8 LB 0D97  FINDMTDI LB 0C24  FINDMTEN LB 0DA0  FINDMTLO LB 0DC0  FINDSERV LB 0A4F  FLATHELL LB 0ABC  FLIP     LB 0806
FLIPDA   LB 083E  FLIPDI   LB 080D  FLIPDIR  LB 083F  FLIPDU   LB 0868  FLIPNU   LB 0853  FLOPCOUN LB 0555  FPARMADD LB 0547
GETPASSW LB 0718  GETSTR   LB 067D  GETSTREC LB 0566  GETSTRLE LB 0564  GETUSERI LB 0700  GOTOXY   LB 0644  HELLO    LB 05E5
HELLONAM LB 05ED  HELLOSOU LB 05EA  HELLOTYP LB 05EC  HOLD5DRI LB 0538  HOLD5DVR LB 0518  HOLDBLOC LB 05B4  HOLDBUFF LB 1666
HOLDNUMB LB 0560  HOLDNUMR LB 0562  HOME     LB 0627  IBMDOS   AB 0004  IFFLATCA MC ----  INTERPAD LB 0539  INVDIMSG LB 013F
INVPASSM LB 017C  INVUSRMS LB 016A  IORETCOD LB 05B8  IRETADD  LB 03DB  LF       AB 000A  LOGON    LB 093F  LOGON1   LB 098D
LOGON2   LB 09AB  LOGON3   LB 09C3  LOGON4   LB 09D2  LOGON5   LB 09EE  LOGON6   LB 0A2D  LOGON7   LB 0A4E  LOGUNIT  LB 0551
MAKEDRVC LB 0883  MAPADD   LB 054F  MAPCOUNT LB 0557  MASKMOD  LB 03DF  MAXFLOP  LB 0559  MEMMARK  LB 0371  MEMMARKL AB 000F
MIISOURC LB 05D8  MNTBLOCK AB 0002  MNTCOUNT LB 0512  MNTDAPOI LB 050E  MNTDASIZ AB 0050  MNTDATAB LB 19A6  MNTDRVPO LB 0510
MNTDRVSI AB 000A  MNTDRVTA LB 19F6  MNTDRVWP AB 0000  MNTDSKSR AB 0001  MNTDVPOI LB 050C  MNTDVSIZ AB 0140  MNTDVTAB LB 1866
MNTMAX   AB 000A  MNTNUMBL AB 0006  MNTRECSI AB 0008  MNTTABAD LB 0514  MNTTABSE LB 0516  MNTTBL   AB 0000  MNTTBLEN AB 0008
MOUNTCOU LB 0553  MSDOSBIO LB 0354  MSDOSBOO LB 1196  MSDOSBT1 LB 11B1  MSDOSBT2 LB 11C5  MSDOSBT3 LB 11DE  MSDOSMOU LB 11FD
MSDOSNAM LB 0363  MSDOSVOL AB 0003  MYIDIS   LB 05D3  NAEND    AB 0010  NAHOST   AB 0000  NAHOSTKI AB 000B  NAHOSTNO AB 000A
NAMEBASE LB 03E1  NAMEINCR LB 03F5  NAMELEN  AB 000A  NAMEMSG  LB 00E0  NARECSIZ AB 0010  NARES    AB 000C  NARPB    AB 0020
NATABLE  AB 0000  NEXTBLOC LB 0936  NOBIOSMS LB 023D  NOBOOTMS LB 018D  NOBOOTPA LB 02E3  NODOSMSG LB 0258  NODRVRMS LB 0219
NOECHO   AB 0001  NOINTERP LB 01F0  NOUNITSM LB 0272  NUEND    AB 0020  NUHOSTTY AB 001E  NUMBLOCK LB 055E  NUOPSYS  AB 001C
NUREC    LB 0484  NURECSIZ AB 0020  NURES1   AB 001F  NURPB    AB 0010  NUSERVER AB 0012  NUTABLE  AB 0000  NUUSER   AB 0000
NUUSERPA AB 000A  OMNIHELL LB 0A94  PASSLEN  AB 0008  PASSMSG  LB 00F9  PRINTAND LB 06EC  PRINTSTR LB 0651  PSWBASE  LB 0409
PSWINCR  LB 0419  PUTFLOPI LB 02C4  PUTLOGO  LB 06F6  RDCMD    AB 0032  READBLOC LB 0DFA  READDIR  LB 0E68  READERRM LB 0301
READUO1  LB 0EE3  READUCSD LB 0EB4  RESTREGS LB 0618  RETLEN   LB 05BF  ROMIO    LB 03D7  SAVEHOST LB 05A4  SAVEREGS LB 0609
SCONFIGA LB 0545  SECTSIZE AB 0200  SERVERHO LB 05A3  SERVERNA LB 05A5  TABBIT   AB 0080  TEMP     LB 055C  TEMPDNAM LB 058C
TEMPENAM LB 0596  TRUE     AB 0000  UCSD1STB AB 0000  UCSD40   AB 0011  UCSDBOOT LB 0F28  UCSDBTOO LB 0F43  UCSDBTO1 LB 0F82
UCSDBTO2 LB 0F8E  UCSDDIR  AB 0000  UCSDDIRB AB 0004  UCSDDIRS AB 001A  UCSDEND  AB 001A  UCSDINTE LB 0324  UCSDLAST AB 0002
UCSDMEMC LB 1010  UCSDMNTF LB 10D2  UCSDMNTO LB 113C  UCSDMOUN LB 1046  UCSDNAME AB 0006  UCSDNUMF AB 0010  USERHOST LB 0579
USERNAME LB 0567  USERPASS LB 0571  VOLADD   LB 054D  WAITFORK LB 0907  WAITMSG  LB 01D1  WAYNAME  LB 05C9  WAYSOURC LB 05C6
WHEREARE LB 05C1  XYERRMSG AB 1200  XYGETPAS AB 1000  XYGETUSE AB 0E00  XYINVPAS AB 1200  XYINVUSR AB 1200
```

Assembly complete: 2863 lines
0 errors flagged on this assembly