

```

0000 ;
0000 ;
0000 ; This file is generated by The Interactive Disassembler (IDA)
0000 ; Licensed to: Unknown User ;- )
0000 ; Copyright (c) 1999 by DataRescue sa/nv, <ida@datarescue.com>
0000 ;
0000 ;
0000 ; File Name : smdv3.sms
0000 ; Format : Binary File
0000 ; Base Address: 0000h Range: 0000h - 2000h Loaded length: 2000h
0000 ;
0000 ; Processor: z80
0000 ; Target assembler: Table Driven Assembler (TASM) by Speech Technology Inc.
0000 ;
0000 ;
0000 ; Segment type: Pure code
0000 ; segment: 'seg000'
0000 ;
0000 loc_0000: ; CODE XREF: 198E□□j
0000 C3 68 00 ; 1996□□j
0000 ; Super Magic Drive v3 BIOS Disassembly
0000 ; by Mark McDougall (tcdev)
0000 ; http://pacedev.net
0000 ; Version 0.9 (incomplete)
0000 ;
0003 C3 3D 0A ; jp print_hexascii_byte
0006 ;
0006 C3 06 0A ; jp print_msg_at_HL
0009 ;
0009 C3 D5 09 ; jp VDP_cls
000C ;
000C C3 98 0A ; jp init_VDP_and_character_set
000F ;
000F C3 29 0B ; jp copy_cart_ROM_to_DRAM
0012 ;
0012 C3 55 1A ; jp handle_pc_command
0012 ;
0015 FF FF FF FF+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
0015 FF FF FF FF+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
0015 FF FF FF FF+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
0015 FF FF FF FF+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
0015 FF FF FF FF+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
0015 FF FF FF FF+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
0015 FF FF FF FF+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
0015 FF FF FF ED+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
0015 45 FF FF FF+ .db 0EDh, 45h
0068 ;
0068 RESET: ; CODE XREF: 0000□□j
0068 F3 di 1
0069 ED 56 im 1
006B AF xor a
006C 32 00 20 ld (2000h), a ; select page 0
006F 32 01 20 ld (2001h), a ; run BIOS in ROM
0072 3E AA ld a, 0AAh ; '-'
0074 32 F9 DF ld (0DF9h), a
0077 31 80 DF ld sp, 0DF80h
007A CD EC 08 call bring_FDC_out_of_reset
007D CD D5 09 call VDP_cls
0080 CD E6 1B call calc_dram_size
0083 CD 8D 00 call calc_carridge_size
0086 AF xor a
0087 32 00 DC ld (0DC00h), a ; clear SMD hdr - file size
008A C3 B6 0B jp MAIN
008D ; ██████████ S U B R O U T I N E ██████████
008D ;
008D ;
008D calc_carridge_size: ; CODE XREF: 0083□□p
008D 3A 09 20 ld a, (2009h) ; FDC digital input
0090 CB 77 bit 6, a ; cartridge present?
0092 20 04 jr nz, loc_0098 ; yes, skip
0094 AF xor a ; cartridge size = 0
0095 C3 2F 01 jp loc_012F
0098 ;
0098 ;
0098 loc_0098: ; CODE XREF: calc_carridge_size+5□□j
0098 AF xor a
0099 57 ld d, a
009A ;
009A ;
009A loc_009A: ; CODE XREF: calc_carridge_size+27□□j
009A 32 00 20 ld (2000h), a ; set rom/dram page
009D 5F ld e, a ; get page
009E 21 00 40 ld hl, 4000h ; cartridge rom bank
00A1 01 32 03 ld bc, 32h
00A4 AF xor a
00A5 ;
00A5 ;
00A5 loc_00A5: ; CODE XREF: calc_carridge_size+1B□□j
00A5 86 add a, (hl) ; calc checksum
00A6 ED A1 cpi ; next address
00A8 EA A5 00 jp pe, loc_00A5 ; loop for $332 bytes
00AB 21 00 DF ld hl, 0DF00h ; scratchpad
00AE 19 add hl, de ; offset for page
00AF 77 ld (hl), a ; store checksum for page
00B0 7B ld a, e ; get page
00B1 3C inc a ; next page
00B2 FE 40 cp 40h ; 'e' ; done 64 pages?
00B4 38 E4 jr c, loc_009A ; no, loop
00B6 21 00 DF ld hl, 0DF00h ; scratchpad
00B9 11 33 01 ld de, 133h
00BC 01 00 08 ld bc, 800h
00BF CD 26 01 call sub_0126 ; WTF???
00C2 21 10 DF ld hl, 0DF10h
00C5 11 33 01 ld de, 133h
00C8 01 10 30 ld bc, 3010h
00CB CD 26 01 call sub_0126
00CE 21 00 DF ld hl, 0DF00h
00D1 11 10 DF ld de, 0DF10h
00D4 01 10 10 ld bc, 1010h
00D7 CD 26 01 call sub_0126
00DA 21 10 DF ld hl, 0DF10h
00DD 11 18 DF ld de, 0DF18h
00E0 01 18 08 ld bc, 818h
00E3 CD 26 01 call sub_0126
00E6 21 20 DF ld hl, 0DF20h
00E9 11 33 01 ld de, 133h
00EC 01 20 20 ld bc, 2020h
00EF CD 26 01 call sub_0126
00F2 21 00 DF ld hl, 0DF00h
00F5 11 20 DF ld de, 0DF20h
00F8 01 20 20 ld bc, 2020h
00FB CD 26 01 call sub_0126
00FE 21 20 DF ld hl, 0DF20h
0101 11 28 DF ld de, 0DF28h
0104 01 28 08 ld bc, 828h
0107 CD 26 01 call sub_0126
010A 21 08 DF ld hl, 0DF08h
010D 11 28 DF ld de, 0DF28h
0110 01 28 18 ld bc, 1828h
0113 CD 26 01 call sub_0126
0116 21 20 DF ld hl, 0DF20h
0119 11 30 DF ld de, 0DF30h
011C 01 30 10 ld bc, 1030h
011F CD 26 01 call sub_0126
0122 0E 40 ld c, 40h ; 'e'
0124 18 08 jr loc_012E

```

```

0124 ; End of function calc_cartridge_size
0124
0126 ; ██████████ S U B R O U T I N E ██████████
0126
0126 sub_0126: ; CODE XREF: calc_cartridge_size+32□p
0126 1A ; calc_cartridge_size+3E□p ...
0126 ld a, (de) ; read back byte
0127 BE cp (hl) ; same?
0128 C0 ret nz ; no, return
0129 23 inc hl
012A 13 inc de ; next address
012B 10 F9 djnz sub_0126 ; loop until done
012D E1 pop hl
012E
012E 79 loc_012E: ; CODE XREF: calc_cartridge_size+97□j
012F ld a, c
012F
012F 32 F1 DF loc_012F: ; CODE XREF: calc_cartridge_size+8□j
0132 C9 ; cartridge size (16kB pages)
0132 ret
0132 ; End of function sub_0126
0132
0132 ;
0133 CE CE CE CE+ .db 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh
0133 CE CE CE CE+ .db 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh
0133 CE CE CE CE+ .db 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh
0133 CE CE CE CE+ .db 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh, 0CEh
0133 CE CE CE CE+ .db 0CEh, 0CEh, 0CEh
0133 ;
0163
0163 loc_0163: ; CODE XREF: detect_disk+23□j
0163 CD 85 09 call check_sig_trk80
0166 CD 7F 03 call read_t0_hl_s9
0169 3A ED DE ld a, (0DEBh)
016C E6 7F and 7Fh, '!' ; get ST1
016E 20 3C jr nz, loc_01AC ; mask off unwanted errors
0170 3A 00 DC ld a, (0DC00h) ; error, skip
0173 B7 or a ; 1st byte SMD header
0174 28 04 jr z, loc_017A ; zero?
0176 FE 41 cp 41h, 'A' ; yes, exit
0178 38 09 jr c, loc_0183 ; valid size?
017A ; yes, skip
017A 17A loc_017A: ; CODE XREF: 0174□j
017A AF xor a
017B 32 00 DC ld (0DC00h), a ; set file size in SMD header to 0
017E 3E 01 ld a, 1 ; "NO GAME"
0180 C3 66 0B jp print_msg_and_exit_option
0183 ;
0183
0183 loc_0183: ; CODE XREF: 0178□j
0184 3A F0 DF ld c, a ; file size
0187 B9 ld a, (0DF0h) ; DRAM size (16kB pages)
0188 30 09 cp c ; file size OK?
018A AF jr nc, loc_0193 ; yes, skip
018B 32 00 DC xor a ; zero SMD file size
018E 3E 07 ld (0DC00h), a ; "WAITING"
0190 C3 66 0B ld a, 7
0193 jp print_msg_and_exit_option
0193 ;
0193
0193 loc_0193: ; CODE XREF: 0188□j
0193 3E 02 ld a, 2 ; "LOADING"
0195 CD 09 0A call print_msg
0198 3E 01 ld a, 1
019A 32 E3 DF ld (0DFE3h), a
019D CD B5 01 call sub_01B5
01A0 3A ED DE ld a, (0DEBh) ; ST1
01A3 E6 7F and 7Fh, '!' ; mask of unwanted errors
01A5 20 05 jr nz, loc_01AC ; exit of error
01A7 3E 0D ld a, 0Dh ; "NO DISK"
01A9 C3 09 0A jp print_msg
01AC ;
01AC
01AC loc_01AC: ; CODE XREF: 016E□j
01AC AF ; 01A5□j
01AD 32 00 DC xor a
01B0 3E 05 ld (0DC00h), a ; "READ ERROR"
01B2 C3 66 0B ld a, 5
01B5 jp print_msg_and_exit_option
01B5 ;
01B5
01B5 ██████████ S U B R O U T I N E ██████████
01B5
01B5 sub_01B5: ; CODE XREF: 019D□p
01B5 CD 42 03 call set_1024_byte_geometry
01B8 AF xor a ; run BIOS in ROM
01B9 32 01 20 ld (2001h), a
01BC ED 47 ld i, a
01BE 32 E2 DF ld (0DFE2h), a
01C1 3A 00 DC ld a, (0DC00h)
01C4 32 9A DE ld (0DE9Ah), a ; number of tracks?
01C7 6F ld l, a
01C8 FE 31 cp 31h, '1'
01CA 38 07 jr c, loc_01D3
01CC D6 30 sub 30h, '0'
01CE 32 E2 DF ld (0DFE2h), a
01D1 2E 30 ld l, 30h, '0'
01D3
01D3 loc_01D3: ; CODE XREF: sub_01B5+15□j
01D3 26 00 ; sub_01B5+B2□j ...
01D3 ld h, 0
01D5 29 add hl, hl
01D6 29 add hl, hl
01D7 29 add hl, hl
01D8 29 add hl, hl
01D9 3A EC DF ld a, (0DFECh) ; bytes/sector (token)
01DC FE 03 cp 3 ; 1024?
01DE 28 01 jr z, loc_01E1 ; yes, skip
01E0 29 add hl, hl
01E1
01E1 loc_01E1: ; CODE XREF: sub_01B5+29□j
01E1 3E 01 ld a, 1
01E3 32 E9 DF ld (0DFE9h), a ; track
01E6 3E 00 ld a, 0
01E8 32 EA DF ld (0DFEAh), a ; head
01EB 3E 01 ld a, 1
01ED 32 EB DF ld (0DFEBh), a ; sector
01F0 CD F2 08 call enable_drive_1
01F3 CD 56 0B call delay
01F6
01F6 loc_01F6: ; CODE XREF: sub_01B5+85□j
01F6 3A E9 DF ld a, (0DFE9h) ; track
01F9 CD FF 0C call fdc_seek
01FC 3A E3 DF ld a, (0DFE3h)
01FF FE 02 cp 2
0201 ED 4B EB DF ld bc, (0DFEBh) ; sector, bytes/sector
0205 3A B9 DE ld a, (0DEB9h) ; end track sector number
0208 3C inc a ; next sector
0209 91 sub c ; #sectors remaining
020A 4F ld c, a
020B 06 00 ld b, 0
020D 3A EC DF ld a, (0DFECh) ; bytes/sector (token)
0210 FE 03 cp 3 ; 1024?
0212 3E 0F ld a, 0Fh
0214 28 02 jr z, loc_0218 ; yes, skip
0216 3E 1F ld a, 1Fh
0218
0218 loc_0218: ; CODE XREF: sub_01B5+5F□j

```

```

0218 A5          and     l
0219 ED 42      sbc     hl, bc
021B 30 04      jr     nc, loc_0221
021D 09          add     hl, bc
021E 4D          ld     c, l
021F 2E 00      ld     l, 0
0221
0221 41          loc_0221: ld     b, c ; CODE XREF: sub_01B5+66[]j
0222 4F          ld     c, a
0223 78          ld     a, b
0224 91          sub    c
0225 30 01      jr     nc, loc_0228
0227 AF          xor    a
0228
0228 4F          loc_0228: ld     c, a ; CODE XREF: sub_01B5+70[]j
0229 E5          push  hl
022A 3A E3 DF    ld     a, (0DFE3h)
022D FE 01      cp     l
022F CC 76 02   call  z, read_BC_bytes_to_DE
0232 3A E3 DF    ld     a, (0DFE3h)
0235 FE 02      cp     2
0237 E1          pop    hl
0238 7C          ld     a, h
0239 B5          or     l
023A 20 BA      jr     nz, loc_01F6
023C 3A E9 DF    ld     a, (0DFE9h) ; track
023F CB 3F      srl    a
0241 CD FF 0C   call  fdc_seek
0244 CD EC 08   call  bring_FDC_out_of_reset
0247 3A E2 DF    ld     a, (0DFE2h)
024A B7          or     a
024B C8          ret    z
024C 6F          ld     l, a
024D AF          xor    a
024E 32 E2 DF    ld     (0DFE2h), a
0251 E5          push  hl
0252 3E 04      ld     a, 4 ; "INSERT DISK 2"
0254 CD 09 0A   call  print_msg
0257 CD 92 0B   call  wait_for_controller
025A 3A E3 DF    ld     a, (0DFE3h)
025D FE 01      cp     l
025F 20 09      jr     nz, loc_026A
0261 3E 02      ld     a, 2 ; "LOADING"
0263 CD 09 0A   call  print_msg
0266 E1          pop    hl
0267 C3 D3 01   jp     loc_01D3
026A
026A
026A CD 2D 09   loc_026A: call  check_disk_WP ; CODE XREF: sub_01B5+AA[]j
026D 3E 03      ld     a, 3 ; "SAVING"
026F CD 09 0A   call  print_msg
0272 E1          pop    hl
0273 C3 D3 01   jp     loc_01D3
; End of function sub_01B5
0273
0273
0276
0276 ; ██████████ S U B R O U T I N E ██████████
0276
0276
0276 read_BC_bytes_to_DE: ; CODE XREF: sub_01B5+7A[]p
0276 3E 46      ld     a, 46h ; 'F' ; read_t0_hl_s9+1E[]p ...
0278 CD B6 02   call  send_9_byte_FDC_command ; FDC READ DATA command
027B
027B loc_027B: ; CODE XREF: read_BC_bytes_to_DE+35[]j
027B 79          ld     a, c
027C B8          cp     b
027D 20 0E      jr     nz, loc_028D ; if B=C then read to random page in DRAM
027F CD FA 0A   call  print_track_and_dec
0282 ED 57      ld     a, i ; random???
0284 32 00 20   ld     (2000h), a ; set rom/dram page
0287 3C          inc    a ; next bank
0288 ED 47      ld     i, a
028A 11 00 80   ld     de, 8000h ; copier DRAM
028D
028D loc_028D: ; CODE XREF: read_BC_bytes_to_DE+7[]j
028D C5          push  bc
028E ED 4B E0 DF ld     bc, (0DFE0h) ; bytes / sector
0292 21 0D 20   ld     hl, 200Dh ; FDC status register
0295
0295 loc_0295: ; CODE XREF: read_BC_bytes_to_DE+21[]j
0295 7E          ld     a, (hl) ; read FDC status
0296 07          rlca  ; ready?
0297 30 FC      jr     nc, loc_0295 ; no, loop
0299 2D          dec    l ; FDC data register
029A ED A0      ldi   ; read FDC data into DRAM
029C 07          rlca  ; execution mode?
029D 07          rlca  ; yes, skip
029E 30 10      jr     nc, loc_02B0
02A0
02A0 loc_02A0: ; CODE XREF: read_BC_bytes_to_DE+2C[]j
02A0 7E          ld     a, (hl) ; read_BC_bytes_to_DE+31[]j
02A1 07          rlca  ; ready?
02A2 30 FC      jr     nc, loc_02A0 ; no, loop
02A4 2D          dec    l ; FDC data register
02A5 ED A0      ldi   ; read FDC data into DRAM
02A7 EA A0 02   jp     pe, loc_02A0 ; loop until done all bytes
02AA C1          pop    bc
02AB 10 CE      djnz  loc_027B ; next 256 bytes
02AD C3 EC 02   jp     read_FDC_result
02B0
02B0
02B0 loc_02B0: ; CODE XREF: read_BC_bytes_to_DE+28[]j
02B0 C1          pop    bc ; DMA mode done???
02B1 1B          dec    de
02B2 1A          ld     a, (de) ; get ST1 from DMA operation???
02B3 C3 EF 02   jp     loc_02EF ; jumps into read_FDC_result
; End of function read_BC_bytes_to_DE
02B3
02B3
02B6
02B6 ; ██████████ S U B R O U T I N E ██████████
02B6
02B6
02B6 send_9_byte_FDC_command: ; CODE XREF: read_BC_bytes_to_DE+2[]p
02B6 CD F8 08   call  send_byte_to_FDC ; send FDC command byte
02B9 3A EA DF    ld     a, (0DFEAh) ; unit select
02BC 07          rlca
02BD 07          rlca
02BE CD F8 08   call  send_byte_to_FDC
02C1 3A E9 DF    ld     a, (0DFE9h) ; track number
02C4 CD F8 08   call  send_byte_to_FDC
02C7 3A EA DF    ld     a, (0DFEAh) ; drive head address
02CA CD F8 08   call  send_byte_to_FDC
02CD 3A EB DF    ld     a, (0DFEBh) ; sector number
02D0 CD F8 08   call  send_byte_to_FDC
02D3 3A EC DF    ld     a, (0DFECh) ; number of bytes per sector (token)
02D6 CD F8 08   call  send_byte_to_FDC
02D9 3A EB DF    ld     a, (0DFEBh) ; end of track sector number
02DC 3D          dec    a
02DD 80          add    a, b
02DE CD F8 08   call  send_byte_to_FDC
02E1 3A F2 DF    ld     a, (0DFF2h) ; in sector gap length
02E4 CD F8 08   call  send_byte_to_FDC
02E7 3E FF      ld     a, 0FFh ; data length
02E9 C3 F8 08   jp     send_byte_to_FDC
; End of function send_9_byte_FDC_command
02E9

```

```

02EC ; -----
02EC read_FDC_result: call read_byte_from_FDC ; CODE XREF: read_BC_bytes_to_DE+37□j
02EC CD 07 09
02EF loc_02EF: ld (0DEBCh), a ; CODE XREF: read_BC_bytes_to_DE+3D□j
02EF 32 BC DE ; ST0
02F2 CD 07 09 call read_byte_from_FDC
02F5 32 BD DE ld (0DEBDh), a ; ST1
02F8 CD 07 09 call read_byte_from_FDC
02FB 32 BE DE ld (0DEBEh), a ; ST2
02FE CD 07 09 call read_byte_from_FDC
0301 32 E9 DF ld (0DFE9h), a ; Track number
0304 CD 07 09 call read_byte_from_FDC
0307 32 EA DF ld (0DFEAh), a ; head number
030A CD 07 09 call read_byte_from_FDC
030D 32 EB DF ld (0DFEBh), a ; sector number
0310 CD 07 09 call read_byte_from_FDC
0313 32 EC DF ld (0DFECh), a ; bytes per sector (token)
0316 3A BD DE ld a, (0DEBDh) ; ST1
0319 CB 7F bit 7, a ; end of cylinder?
031B C8 ret z ; no, return
031C 3A EB DF ld a, (0DFEBh) ; sector number
031F ED 4B B9 DE ld bc, (0DEB9h)
0323 B9 cp c ; end of track?
0324 30 05 jr nc, increment_track
0326 3C inc a ; next sector
0327 32 EB DF ld (0DFEBh), a ; store new sector
032A C9 ret
; -----
032B increment_track: ; CODE XREF: 0324□j
032B 3E 01 ld a, 1
032D 32 EB DF ld (0DFEBh), a ; store sector number
0330 3A EA DF ld a, (0DFEAh) ; head number
0333 EE 01 xor l ; toggle
0335 32 EA DF ld (0DFEAh), a ; store new head number
0338 C0 ret nz ; return if same track
0339 21 E9 DF ld hl, 0DFE9h ; track number
033C 34 inc (hl) ; increment track
033D C9 ret
033E ; ██████████ S U B R O U T I N E ██████████
033E
033E
033E set_512_byte_geometry: ; CODE XREF: read_t0_h1_s9+3□p
033E 3E 00 ld a, 0 ; check_sig_trk80+3□p
0340 18 02 jr loc_0344
; -----
0342
0342 set_1024_byte_geometry: ; CODE XREF: sub_01B5□p
0342 3E 02 ld a, 2
0344 loc_0344: ; CODE XREF: set_512_byte_geometry+2□j
0344 6F ld l, a
0345 26 00 ld h, 0
0347 29 add hl, hl
0348 29 add hl, hl
0349 11 6F 03 ld de, 36Fh ; drive geometry table
034C 19 add hl, de
034D 7E ld a, (hl)
034E 23 inc hl
034F 32 EC DF ld (0DFECh), a ; bytes/sector (token)
0352 7E ld a, (hl)
0353 23 inc hl
0354 32 B9 DE ld (0DEB9h), a ; sectors/track?
0357 7E ld a, (hl)
0358 23 inc hl
0359 32 F2 DF ld (0DF2h), a
035C 7E ld a, (hl)
035D 23 inc hl
035E 32 F3 DF ld (0DF3h), a
0361 3A EC DF ld a, (0DFECh) ; bytes/sector (token)
0364 47 ld b, a
0365 21 80 00 ld hl, 128
0368 loc_0368: ; CODE XREF: set_512_byte_geometry+2B□j
0368 29 add hl, hl
0369 10 FD djnz loc_0368 ; calculate number of bytes
036B 22 E0 DF ld (0DFE0h), hl ; store number of bytes per sector
036E C9 ret
; End of function set_512_byte_geometry
036E
; -----
036E
036E .db 2 ; 512 bytes/sector
0370 09 .db 9 ; SPT
0371 1B .db 1Bh ;
0372 54 .db 54h ; T
0373 02 .db 2 ;
0374 12 .db 12h ;
0375 1B .db 1Bh ;
0376 54 .db 54h ; T
0377 03 .db 3 ; 1024 bytes/sector
0378 05 .db 5 ; SPT
0379 35 .db 35h ; 5
037A 74 .db 74h ; t
037B 03 .db 3 ;
037C 0A .db 0Ah ;
037D 35 .db 35h ; 5
037E 74 .db 74h ; t
037F
; ██████████ S U B R O U T I N E ██████████
037F
037F
037F read_t0_h1_s9: ; CODE XREF: 0166□p
037F CD F2 08 call enable_drive_1
0382 CD 3E 03 call set_512_byte_geometry
0385 3E 00 ld a, 0
0387 32 E9 DF ld (0DFE9h), a ; track
038A CD FF 0C call fdc_seek
038D 3E 01 ld a, 1
038F 32 EA DF ld (0DFEAh), a ; head
0392 3E 09 ld a, 9
0394 32 EB DF ld (0DFEBh), a ; sector
0397 11 00 DC ld de, 0DC00h ; ptr SMD header
039A 01 00 01 ld bc, 100h ; 256 bytes
039D CD 76 02 call read_BC_bytes_to_DE
03A0 C3 EC 08 jp bring_FDC_out_of_reset
; End of function read_t0_h1_s9
03A0
; -----
03A3 2A 2A 20+aMagicDriverV3: .text "**** MAGIC DRIVER V-3 *** "
03BC 50 52 4F 44+aProductManager: .text "PRODUCT MANAGER: LMM, "
03D2 44 45 42 55+aDebugEngineerJ: .text "DEBUG ENGINEER: JSI, "
03E7 42 49 4F 53+aBiosProgrammer: .text "BIOS PROGRAMMER: JSI, "
03FD 41 4C 4C 20+allRightsReser: .text "ALL RIGHTS RESERVED, "
0412 43 4F 50 59+aCopyright1991B: .text "COPYRIGHT 1991 BY JSI, "
0429 43 48 49 4E+aChinaCoachComp: .text "CHINA COACH COMPANY LTD., "
0443 46 52 4F 4E+aFrontFareast1: .text "FRONT FAREAST CORPORATION. 8/17/91 "
0466 03 .db 3 ;
0467 05 .db 5 ;
0468 03 .db 3 ;
0469 05 .db 5 ;
046A 02 .db 2 ; number of routines in main menu
046B 00 .db 0 ;
046C 08 .db 8 ; number of routines in utility menu
046D 00 .db 0 ;
046E 72 04 .dw 472h ; main menu routine table
0470 76 04 .dw 476h ; utility menu routine table
0472 6E 19 .dw 196Eh ; run file routine

```

```

0474 A0 19      .dw 19A0h      ; run ic card routine
0476 AE 19      .dw 19AEh      ; save ic card routine
0478 CF 19      .dw 19CFh      ; rename file routine
047A 0D 1A      .dw 1A0Dh      ; delete file routine
047C 38 1A      .dw 1A38h      ; format disk routine
047E 3D 19      .dw 193Dh      ; load data routine
0480 50 19      .dw 1950h      ; save data routine
0482 E4 18      .dw 18E4h
0484 FD 18      .dw 18FDh      ; dump cartridge routine
0486 30 06      .dw 630h
0488 40 06      .dw 640h      ; "NO GAME"
048A 4D 06      .dw 64Dh      ; "LOADING"
048C 5A 06      .dw 65Ah      ; "SAVING"
048E 67 06      .dw 667h      ; "INSERT DISK 2"
0490 77 06      .dw 677h      ; "READ ERROR"
0492 84 06      .dw 684h      ; "WRITE PROTECT"
0494 94 06      .dw 694h      ; "WAITING"
0496 94 06      .dw 694h      ; "WAITING"
0498 A1 06      .dw 6A1h      ; "NO IC CARD"
049A AE 06      .dw 6AEh      ; "UNKNOW DISK" (sic)
049C BC 06      .dw 6BCh      ; "NO DISK"
049E BC 06      .dw 6BCh      ; "NO DISK"
04A0 BC 06      .dw 6BCh      ; "NO DISK"
04A2 BC 06      .dw 6BCh      ; "NO DISK"
04A4 BC 06      .dw 6BCh      ; "NO DISK"
04A6 CA 06      .dw 6CAh      ; "NO GAME IN RAM"
04A8 D6 04      .dw 4D6h      ; main menu
04AA 71 05      .dw 571h      ; utility menu
04AC DB 06      .dw 6DBh      ; "ERR"
04AE E1 06      .dw 6E1h      ; "WRITE ERROR"
04B0 EF 06      .dw 6EFh      ; "NO FILE"
04B2 F9 06      .dw 6F9h      ; "FILE NOT FOUND"
04B4 0A 07      .dw 70Ah      ; "DUP FILE NAME"
04B6 1A 07      .dw 71Ah      ; "NOT ENOUGH SPACE"
04B8 2D 07      .dw 72Dh      ; "FILE ERR"
04BA 3A 07      .dw 73Ah      ; "NEW NAME: _____"
04BC CD 07      .dw 7CDh      ; "FILE LIST:"
04BE 23 08      .dw 823h      ; "RUN FILE"
04C0 2E 08      .dw 82Eh      ; "SAVE IC CARD"
04C2 3D 08      .dw 83Dh      ; "LOAD DATA"
04C4 49 08      .dw 849h      ; "SAVE DATA"
04C6 55 08      .dw 855h      ; "LOAD FILE"
04C8 61 08      .dw 861h      ; "SAVE FILE"
04CA 6D 08      .dw 86Dh      ; "RENAME FILE"
04CC 7B 08      .dw 87Bh      ; "DELETE FILE"
04CE 89 08      .dw 889h      ; "FORMAT DISK"
04D0 C2 08      .dw 8C2h      ; "RENAMING..."
04D2 D0 08      .dw 8D0h      ; "DELETING..."
04D4 DE 08      .dw 8DEh      ; "FORMATTING."
04D6 01        .db 1 ;
04D7 02        .db 2 ;
04D8 4D 41 47 49+aMagicDriverV_0: .text "MAGIC DRIVER V-3"
04E9 FF        .db 0FFh ;
04EA 01        .db 1 ;
04EB 14        .db 14h ;
04EC 01        .db 1 ;
04ED 31 39 39 31+a1991Jsi: .text "1991 JSI"
04F5 FF        .db 0FFh ;
04F6 05        .db 5 ;
04F7 05        .db 5 ;
04F8 52 55 4E 20+aRunFile: .text "RUN FILE"
0500 FF        .db 0FFh ;
0501 07        .db 7 ;
0502 05        .db 5 ;
0503 52 55 4E 20+aRunIcCard: .text "RUN IC CARD"
050E FF        .db 0FFh ;
050F 05        .db 5 ;
0510 12        .db 12h ;
0511 5B 1B 1A 5D+aMenu: .text "["
0511 2D 4D 45 4E+ .db 1Bh
0511 55        .db 1Ah
0511          .text "]"-MENU"
051A FF        .db 0FFh ;
051B 07        .db 7 ;
051C 12        .db 12h ;
051D 5B 18 19 5D+aChoose: .text "["
051D 2D 43 48 4F+ .db 18h
051D 4F 53 45 .db 19h
051D          .text "]"-CHOOSE"
0528 FF        .db 0FFh ;
0529 09        .db 9 ;
052A 12        .db 12h ;
052B 5B 43 5D 2D+aCAccept: .text "[C]-ACCEPT"
0535 FF        .db 0FFh ;
0536 0D        .db 0Dh ;
0537 16        .db 16h ;
0538 52 41 4D 20+aRamM: .text "RAM M"
053F FF        .db 0FFh ;
0540 0F        .db 0Fh ;
0541 16        .db 16h ;
0542 43 41 52 44+aCardM: .text "CARD M"
0549 FF        .db 0FFh ;
054A 14        .db 14h ;
054B 09        .db 9 ;
054C 01        .db 1 ;
054D 31 39 39 31+a1991CclH_k_: .text "1991 CCL H.K."
055A FF        .db 0FFh ;
055B 16        .db 16h ;
055C 07        .db 7 ;
055D 26 20 46 52+aProntFareast_0: .text "% FRONT FAREAST CO."
0570 00        .db 0 ;
0571 01        .db 1 ;
0572 02        .db 2 ;
0573 55 54 49 4C+aUtilityMenu: .text "UTILITY MENU"
057F FF        .db 0FFh ;
0580 01        .db 1 ;
0581 14        .db 14h ;
0582 01        .db 1 ;
0583 31 39 39 31+a1991Jsi_0: .text "1991 JSI"
058B FF        .db 0FFh ;
058C 05        .db 5 ;
058D 05        .db 5 ;
058E 53 41 56 45+aSaveIcCard: .text "SAVE IC CARD"
059A FF        .db 0FFh ;
059B 07        .db 7 ;
059C 05        .db 5 ;
059D 52 45 4E 41+aRenameFile: .text "RENAME FILE"
05A6 FF        .db 0FFh ;
05A9 09        .db 9 ;
05AA 05        .db 5 ;
05AB 44 45 4C 45+aDeleteFile: .text "DELETE FILE"
05B6 FF        .db 0FFh ;
05B7 0B        .db 0Bh ;
05B8 05        .db 5 ;
05B9 46 4F 52 4D+aFormatDisk: .text "FORMAT DISK"
05C4 FF        .db 0FFh ;
05C5 0D        .db 0Dh ;
05C6 05        .db 5 ;
05C7 4C 4F 41 44+aLoadData: .text "LOAD DATA"
05D0 FF        .db 0FFh ;
05D1 0F        .db 0Fh ;
05D2 05        .db 5 ;
05D3 53 41 56 45+aSaveData: .text "SAVE DATA"
05DC FF        .db 0FFh ;
05DD 11        .db 11h ;
05DE 05        .db 5 ;
05DF 4C 4F 41 44+aLoadFile: .text "LOAD FILE"
05E8 FF        .db 0FFh ;
05E9 13        .db 13h ;
05EA 05        .db 5 ;
05EB 53 41 56 45+aSaveFile: .text "SAVE FILE"

```

```

05F4 FF .db 0FFh ;
05F5 05 .db 5 ;
05F6 12 .db 12h ;
05F7 5B 1A 5D+aMenu_0: .text "["
05F7 2D 4D 45 4E+ .db 1Bh
05F7 55 .db 1Ah
05F7 .text "]"-MENU"
0600 FF .db 0FFh ;
0601 07 .db 7 ;
0602 12 .db 12h ;
0603 5B 18 19 5D+aChoose_0: .text "["
0603 2D 43 48 4F+ .db 18h
0603 4F 53 45 .db 19h
0603 .text "]"-CHOOSE"
060E FF .db 0FFh ;
060F 09 .db 9 ;
0610 12 .db 12h ;
0611 5B 43 5D 2D+aCAccept_0: .text "[C]-ACCEPT"
061B FF .db 0FFh ;
061C 0D .db 0Dh ;
061D 16 .db 16h ;
061E 52 41 4D 20+aRamM_0: .text "RAM M"
0625 FF .db 0FFh ;
0626 0F .db 0Fh ;
0627 16 .db 16h ;
0628 43 41 52 44+aCardM_0: .text "CARD M"
062F 00 .db 0 ;
0630 11 .db 11h ;
0631 03 .db 3 ;
0632 20 20 20 20+ .text " "
063F 00 .db 0 ;
0640 11 .db 11h ;
0641 03 .db 3 ;
0642 4E 4F 20 47+aNoGame: .text "NO GAME "
064C 00 .db 0 ;
064D 11 .db 11h ;
064E 03 .db 3 ;
064F 4C 4F 41 44+aLoading___: .text "LOADING..."
0659 00 .db 0 ;
065A 11 .db 11h ;
065B 03 .db 3 ;
065C 53 41 56 49+aSaving___: .text "SAVING ..."
0666 00 .db 0 ;
0667 11 .db 11h ;
0668 03 .db 3 ;
0669 49 4E 53 45+aInsertDisk2: .text "INSERT DISK 2"
0676 00 .db 0 ;
0677 11 .db 11h ;
0678 03 .db 3 ;
0679 52 45 41 44+aReadError: .text "READ ERROR"
0683 00 .db 0 ;
0684 11 .db 11h ;
0685 03 .db 3 ;
0686 57 52 49 54+aWriteProtect: .text "WRITE PROTECT"
0693 00 .db 0 ;
0694 11 .db 11h ;
0695 03 .db 3 ;
0696 57 41 49 54+aWaiting___: .text "WAITING..."
06A0 00 .db 0 ;
06A1 11 .db 11h ;
06A2 03 .db 3 ;
06A3 4E 4F 20 49+aNoIcCard: .text "NO IC CARD"
06AD 00 .db 0 ;
06AE 11 .db 11h ;
06AF 03 .db 3 ;
06B0 55 4E 4B 4E+aUnknowDisk: .text "UNKNOW DISK"
06BB 00 .db 0 ;
06BC 11 .db 11h ;
06BD 03 .db 3 ;
06BE 4E 4F 20 44+aNoDisk: .text "NO DISK "
06C9 00 .db 0 ;
06CA 11 .db 11h ;
06CB 03 .db 3 ;
06CC 4E 4F 20 47+aNoGameInRam: .text "NO GAME IN RAM"
06DA 00 .db 0 ;
06DB 11 .db 11h ;
06DC 03 .db 3 ;
06DD 45 52 52 aErr: .text "ERR"
06E0 00 .db 0 ;
06E1 11 .db 11h ;
06E2 03 .db 3 ;
06E3 57 52 49 54+aWriteError: .text "WRITE ERROR"
06EE 00 .db 0 ;
06EF 11 .db 11h ;
06F0 03 .db 3 ;
06F1 4E 4F 20 46+aNoFile: .text "NO FILE"
06F8 00 .db 0 ;
06F9 11 .db 11h ;
06FA 03 .db 3 ;
06FB 46 49 4C 45+aFileNotFound: .text "FILE NOT FOUND"
0709 00 .db 0 ;
070A 11 .db 11h ;
070B 03 .db 3 ;
070C 44 55 50 20+aDupFileName: .text "DUP FILE NAME"
0719 00 .db 0 ;
071A 11 .db 11h ;
071B 03 .db 3 ;
071C 4E 4F 54 20+aNotEnoughSpace: .text "NOT ENOUGH SPACE"
072C 00 .db 0 ;
072D 11 .db 11h ;
072E 03 .db 3 ;
072F 46 49 4C 45+aFileErr: .text "FILE ERR "
0739 00 .db 0 ;
073A 05 .db 5 ;
073B 03 .db 3 ;
073C 20 20 4E 45+aNewName_____.text " NEW NAME : _____"
0754 FF .db 0FFh ;
0755 07 .db 7 ;
0756 05 .db 5 ;
0757 30 31 32 33+a0123456789abcd: .text "0123456789ABCDEF"
0769 FF .db 0FFh ;
076A 09 .db 9 ;
076B 05 .db 5 ;
076C 49 4A 4B 4C+aIjklmnopqrstuv: .text "IJKLMNOPQRSTUVWXYZ"
077E FF .db 0FFh ;
077F 0C .db 0Ch ;
0780 05 .db 5 ;
0781 5B 18 19 1B+aChoose_1: .text "["
0781 1A 5D 2D 43+ .db 18h
0781 48 4F 4F 53+ .db 19h
0781 45 .db 1Bh
0781 .db 1Ah
0781 .text "]"-CHOOSE"
078E FF .db 0FFh ;
078F 0D .db 0Dh ;
0790 05 .db 5 ;
0791 5B 43 5D 2D+aCCharInsert: .text "[C]-CHAR INSERT"
07A0 FF .db 0FFh ;
07A1 0E .db 0Eh ;
07A2 05 .db 5 ;
07A3 5B 42 5D 2D+aBCharDelete: .text "[B]-CHAR DELETE"
07B2 FF .db 0FFh ;
07B3 0F .db 0Fh ;
07B4 05 .db 5 ;
07B5 5B 18 43 5D+aCAcceptBAabort: .text "["
07B5 2D 41 43 43+ .db 18h
07B5 45 50 54 20+ .text "[C]-ACCEPT ["
07B5 20 5B 18 42+ .db 18h
07B5 5D 2D 41 42+ .text "[B]-ABORT"
07CC 00 .db 0 ;
07CD 03 .db 3 ;

```

```

07CE 05 .db 5 ;
07CF 46 49 4C 45+aFileList: .text "FILE LIST:"
07D9 FF .db 0FFh ;
07DA 05 .db 5 ;
07DB 12 .db 12h ;
07DC 5B 1B 1A 5D+aPage: .text "["
07DC 2D 50 41 47+ .db 1Bh
07DC 45 .db 1Ah
07DC .text "]--PAGE"
07E5 FF .db 0FFh ;
07E6 07 .db 7 ;
07E7 12 .db 12h ;
07E8 5B 18 19 5D+aChoose_2: .text "["
07E8 2D 43 48 4F+ .db 18h
07E8 4F 53 45 .db 19h
07E8 .text "]--CHOOSE"
07F3 FF .db 0FFh ;
07F4 09 .db 9 ;
07F5 12 .db 12h ;
07F6 5B 43 5D 2D+aCAccept_1: .text "[C]-ACCEPT"
0800 FF .db 0FFh ;
0801 0B .db 0Bh ;
0802 12 .db 12h ;
0803 5B 42 5D 2D+aBAbort: .text "[B]-ABORT"
080C FF .db 0FFh ;
080D 0D .db 0Dh ;
080E 12 .db 12h ;
080F 46 49 4C 45+aFile: .text "FILE: "
0817 FF .db 0FFh ;
0818 0F .db 0Fh ;
0819 12 .db 12h ;
081A 46 52 45 45+aFreeM: .text "FREE: M"
0822 00 .db 0 ;
0823 01 .db 1 ;
0824 02 .db 2 ;
0825 52 55 4E 20+aRunFile_0: .text "RUN FILE"
082D 00 .db 0 ;
082E 01 .db 1 ;
082F 02 .db 2 ;
0830 53 41 56 45+aSaveIcCard_0: .text "SAVE IC CARD"
083C 00 .db 0 ;
083D 01 .db 1 ;
083E 02 .db 2 ;
083F 4C 4F 41 44+aLoadData_0: .text "LOAD DATA"
0848 00 .db 0 ;
0849 01 .db 1 ;
084A 02 .db 2 ;
084B 53 41 56 45+aSaveData_0: .text "SAVE DATA"
0854 00 .db 0 ;
0855 01 .db 1 ;
0856 02 .db 2 ;
0857 4C 4F 41 44+aLoadFile_0: .text "LOAD FILE"
0860 00 .db 0 ;
0861 01 .db 1 ;
0862 02 .db 2 ;
0863 53 41 56 45+aSaveFile_0: .text "SAVE FILE"
086C 00 .db 0 ;
086D 01 .db 1 ;
086E 02 .db 2 ;
086F 52 45 4E 41+aRenameFile_0: .text "RENAME FILE"
087A 00 .db 0 ;
087B 01 .db 1 ;
087C 02 .db 2 ;
087D 44 45 4C 45+aDeleteFile_0: .text "DELETE FILE"
0888 00 .db 0 ;
0889 01 .db 1 ;
088A 02 .db 2 ;
088B 46 4F 52 4D+aFormatDisk_0: .text "FORMAT DISK"
0896 FF .db 0FFh ;
0897 03 .db 3 ;
0898 05 .db 5 ;
0899 46 4F 52 4D+aFormatSelect: .text "FORMAT SELECT:"
08A7 FF .db 0FFh ;
08A8 05 .db 5 ;
08A9 05 .db 5 ;
08AA 31 2E 34 34+a1_44mHd: .text "1.44M (HD)"
08B4 FF .db 0FFh ;
08B5 07 .db 7 ;
08B6 05 .db 5 ;
08B7 37 32 30 4B+a720k2d: .text "720K (2D)"
08C1 00 .db 0 ;
08C2 11 .db 11h ;
08C3 03 .db 3 ;
08C4 52 45 4E 41+aRenaming___: .text "RENAMING..."
08CF 00 .db 0 ;
08D0 11 .db 11h ;
08D1 03 .db 3 ;
08D2 44 45 4C 45+aDeleting___: .text "DELETING..."
08DD 00 .db 0 ;
08DE 11 .db 11h ;
08DF 03 .db 3 ;
08E0 46 4F 52 4D+aFormatting_: .text "FORMATTING."
08EB 00 .db 0 ;
08EC ; ██████████ S U B R O U T I N E ██████████
08EC
08EC bring_FDC_out_of_reset: ; CODE XREF: 007A□p
08EC 3E 04 ld a, 4 ; sub_01B5+8F□p ...
08EE 32 0B 20 ld (200Bh), a ; bring FDC out of reset
08F1 C9 ret ; FDC digital output register
; End of function bring_FDC_out_of_reset
08F1
08F2 ; ██████████ S U B R O U T I N E ██████████
08F2
08F2 enable_drive_1: ; CODE XREF: sub_01B5+3B□p
08F2 3E 2D ld a, 2Dh ; '-' ; read_t0_h1_s9□p ...
08F4 32 0B 20 ld (200Bh), a ; enable drive 1, DRQ etc
08F7 C9 ret ; FDC digital output register
; End of function enable_drive_1
08F7
08F8 ; ██████████ S U B R O U T I N E ██████████
08F8
08F8 send_byte_to_FDC: ; CODE XREF: send_9_byte_FDC_command□p
08F8 F5 push af ; send_9_byte_FDC_command+8□p ...
08F9
08F9 loc_08F9: ; CODE XREF: send_byte_to_FDC+8□j
08F9 3A 0D 20 ld a, (200Dh) ; FDC status
08FC E6 C0 and 0C0h ; 'L'
08FE E0 cp 80h ; 'C' ; host send ready?
0900 20 F7 jr nz, loc_08F9 ; no, loop
0902 F1 pop af
0903 32 0C 20 ld (200Ch), a ; write to FDC data register
0906 C9 ret
; End of function send_byte_to_FDC
0906
0907 ; ██████████ S U B R O U T I N E ██████████
0907
0907 read_byte_from_FDC: ; CODE XREF: 02EC□p
0907 3A 0D 20 ld a, (200Dh) ; 02F2□p ...
090A E6 C0 and 0C0h ; 'L' ; FDC status

```

```

090C FE C0          cp      0C0h ; 'L' ; host rcv ready?
090E 20 F7          jr      nz, read_byte_from_FDC ; no, loop
0910 3A 0C 20       ld      a, (200Ch) ; read from FDC data register
0913 C9             ret
0913             ; End of function read_byte_from_FDC
0914
0914             ; ██████████ S U B R O U T I N E ██████████
0914
0914             FDC_sense_interrupt_status: ; CODE XREF: FDC_sense_interrupt_status+5□□j
0914 3A 09 20       ld      a, (2009h) ; fdc_seek+2C□□p
0917 CB 7F          bit     7, a
0919 28 F9          jr      z, FDC_sense_interrupt_status
091B 3E 08          ld      a, 8 ; FDC SENSE INTERRUPT STATUS command
091D CD F8 08       call    send_byte_to_FDC
0920 CD 07 09       call    read_byte_from_FDC
0923 32 BC DE       ld      (0DEBCh), a ; ST0
0926 CD 07 09       call    read_byte_from_FDC
0929 32 BF DE       ld      (0DEBFh), a ; present cylinder number
092C C9             ret
092C             ; End of function FDC_sense_interrupt_status
092D
092D             ; ██████████ S U B R O U T I N E ██████████
092D
092D             check_disk_WP: ; CODE XREF: sub_01B5+B5□□p
092D             ; 0964□□j ...
092D CD F2 08       call    enable_drive_1
0930 3E 04          ld      a, 4 ; FDC SENSE DRIVE STATUS command
0932 CD F8 08       call    send_byte_to_FDC
0935 3E 00          ld      a, 0 ; head / unit select
0937 CD F8 08       call    send_byte_to_FDC
093A CD 07 09       call    read_byte_from_FDC
093D 32 E8 DF       ld      (0DFE8h), a ; ST3
0940 3A E8 DF       ld      a, (0DFE8h)
0943 CB 77          bit     6, a ; WP?
0945 C8             ret     z ; no, return
0946 3E 06          ld      a, 6 ; "WRITE PROTECT"
0948 C3 66 0B       jp      print_msg_and_exit_option
0948             ; End of function check_disk_WP
0948
0948             ;
0948             ; -----
0948             ;
094B CD EC 08       call    bring_FDC_out_of_reset ; (I don't think this code is ever called?)
094E 3E 06          ld      a, 6 ; "WRITE PROTECT"
0950 CD 09 0A       call    print_msg
0953
0953             loc_0953: ; CODE XREF: 095A□□j
0953 CD 99 0B       call    handle_pc_cmd_read_controller
0956 3A 95 DF       ld      a, (0DF95h) ; get controller input
0959 B7             or      ; anything pressed?
095A 20 F7          jr      nz, loc_0953 ; yes, loop while pressed
095C
095C             loc_095C: ; CODE XREF: 0968□□j
095C CD 99 0B       call    handle_pc_cmd_read_controller
095F 3A 95 DF       ld      a, (0DF95h) ; get controller input
0962 CB 6F          bit     5, a ; button C?
0964 20 C7          jr      nz, check_disk_WP ; yes, skip
0966 CB 67          bit     4, a ; button B?
0968 28 F2          jr      z, loc_095C ; no, loop
096A ED 7B 8C DF    ld      sp, (0DF8Ch) ; exit option
096E C9             ret
096F
096F             ; ██████████ S U B R O U T I N E ██████████
096F
096F             check_disc_present: ; CODE XREF: detect_disk+A□□p
096F             ; detect_disk+37□□p ...
096F CD F2 08       call    enable_drive_1
0972 01 00 D0       ld      bc, 53248 ; number of times to retry
0975
0975             loc_0975: ; CODE XREF: check_disc_present+F□□j
0975 3A 09 20       ld      a, (2009h)
0978 CB 6F          bit     5, a ; disk in drive???
097A C8             ret     z ; yes, return
097B 0B             dec     bc
097C 78             ld      a, b
097D B1             or      c
097E 20 F5          jr      nz, loc_0975 ; loop until give up
0980 3E 0F          ld      a, 0Fh ; "NO DISK"
0982 C3 66 0B       jp      print_msg_and_exit_option
0982             ; End of function check_disc_present
0982
0982             ;
0982             ; -----
0982             ;
0985             ; ██████████ S U B R O U T I N E ██████████
0985
0985             check_sig_trk80: ; CODE XREF: 0163□□p
0985             ; detect_disk+2E□□p
0985 CD F2 08       call    enable_drive_1
0988 CD 3E 03       call    set_512_byte_geometry
098B 3E 50          ld      a, 80 ; track
098D 32 E9 DF       ld      (0DFE9h), a
0990 CD FF 0C       call    fdc_seek
0993 3E 00          ld      a, 0 ; set head
0995 32 EA DF       ld      (0DFEAh), a
0998 3E 01          ld      a, 1 ; set sector
099A 32 EB DF       ld      (0DFEBh), a
099D 11 00 DA       ld      de, 0DA00h ; scratch buffer
09A0 01 00 01       ld      bc, 100h
09A3 CD 76 02       call    read_BC_bytes_to_DE
09A6 CD EC 08       call    bring_FDC_out_of_reset
09A9 3A BD DE       ld      a, (0DEBDh) ; get ST1
09AC E6 35          and     35h ; '5' ; DE,OR,MISSING ADDRESS MARK
09AE 20 11          jr      nz, loc_09C1 ; error, jump (unknow disk)
09B0 21 00 DA       ld      hl, 0DA00h
09B3 11 E0 1B       ld      de, 1BE0h ; ptr FFE message
09B6 06 20          ld      b, 32 ; 32 chars
09B8
09B8             loc_09B8: ; CODE XREF: check_sig_trk80+39□□j
09B8 1A             ld      a, (de) ; get message byte
09B9 BE             cp      (hl) ; same?
09BA 20 05          jr      nz, loc_09C1 ; no, exit (UNKNOW DISK)
09BC 23             inc     hl
09BD 13             inc     de ; next byte
09BE 10 F8          djnz   loc_09B8 ; loop until done
09C0 C9             ret
09C1
09C1             ;
09C1             ; -----
09C1             ;
09C1             loc_09C1: ; CODE XREF: check_sig_trk80+29□□j
09C1 3E 0A          ld      a, 0Ah ; check_sig_trk80+35□□j
09C3 C3 66 0B       jp      print_msg_and_exit_option ; "UNKNOW DISK" (sic)
09C3             ; End of function check_sig_trk80
09C3
09C3             ; ██████████ S U B R O U T I N E ██████████
09C3
09C3             init_VDP_display_enabled: ; CODE XREF: VDP_cls+2D□□p
09C3 3E E0          ld      a, 0E0h ; 'ó' ; TMS9918 Mode, Display enable, VIRQ enable
09C8 18 02          jr      loc_09CC
09C8             ; End of function init_VDP_display_enabled
09C8
09C8             ; ██████████ S U B R O U T I N E ██████████
09C8
09C8             init_VDP_display_disabled: ; CODE XREF: VDP_cls□□p

```



```

09CA 3E A0                                ; init_VDP_and_character_set␣p ...
09CA                                ; TMS9918 Mode, VIRQ enable
09CC
09CC loc_09CC:                                ; CODE XREF: init_VDP_display_enabled+2␣j
09CC D3 BF                                out    (0BFh), a                ; write VDP control port
09CE 3E 81                                ld     a, 81h ; 'H'           ; mode set register #2
09D0 D3 BF                                out    (0BFh), a                ; write VDP control port
09D2 DB BF                                in     a, (0BFh)               ; read VDP control port
09D4 C9                                ret
09D4                                ; End of function init_VDP_display_disabled
09D4
09D5                                ; ██████████ S U B R O U T I N E ██████████
09D5
09D5 VDP_cls:                                    ; CODE XREF: 0009␣j
09D5 CD CA 09                               ; 007D␣p ...
09D5                                call   init_VDP_display_disabled
09D8 21 00 78                               ld     hl, 7800h
09DB 11 08 20                               ld     de, 2008h
09DE 01 00 03                               ld     bc, 768                 ; #bytes to clear
09E1 CD 5E 0A                               call   write_VDP_control_word
09E4
09E4 cls:                                    ; CODE XREF: VDP_cls+18␣j
09E4 7A                                ld     a, d
09E5 D3 BE                                out    (0BEh), a                ; write VDP data port
09E7 7B                                ld     a, e
09E8 D3 BE                                out    (0BEh), a                ; write VDP data port
09EA 0B                                dec    bc
09EB 78                                ld     a, b
09EC B1                                or     c                        ; done screen?
09ED 20 F5                                jr     nz, cls                  ; no, loop
09EF 21 00 7F                               ld     hl, 7F00h
09F2 16 E0                                ld     d, 0E0h ; 'Ó'
09F4 01 40 00                               ld     bc, 64
09F7 CD 5E 0A                               call   write_VDP_control_word
09FA
09FA loc_09FA:                                ; CODE XREF: VDP_cls+2B␣j
09FA 7A                                ld     a, d
09FB D3 BE                                out    (0BEh), a                ; write VDP data port
09FD 0B                                dec    bc
09FE 78                                ld     a, b
09FF B1                                or     c                        ; done?
0A00 20 F8                                jr     nz, loc_09FA            ; no, loop
0A02 CD C6 09                               call   init_VDP_display_enabled
0A05 C9                                ret
0A05                                ; End of function VDP_cls
0A05
0A06                                ; ██████████ S U B R O U T I N E ██████████
0A06
0A06 print_msg_at_HL:                            ; CODE XREF: 0006␣j
0A06 C5                                ; select_file+D6␣p
0A06                                push   bc
0A07 18 0E                                jr     loc_0A17
0A07                                ; End of function print_msg_at_HL
0A07
0A09                                ; ██████████ S U B R O U T I N E ██████████
0A09
0A09 print_msg:                                    ; CODE XREF: 0195␣p
0A09 C5                                ; 01A9␣j ...
0A09                                push   bc
0A0A CB 27                               sla    a
0A0C 4F                                ld     c, a
0A0D 06 00                                ld     b, 0
0A0F 21 86 04                               ld     hl, 486h
0A12 09                                add    hl, bc
0A13 7E                                ld     a, (hl)
0A14 23                                inc    hl
0A15 66                                ld     h, (hl)
0A16 6F                                ld     l, a
0A17                                ; hl = message
0A17 loc_0A17:                                    ; CODE XREF: print_msg_at_HL+1␣j
0A17 DB BF                                in     a, (0BFh)
0A19 4E                                ld     c, (hl)
0A1A 23                                inc    hl
0A1B 7E                                ld     a, (hl)
0A1C 23                                inc    hl
0A1D 07                                rlca
0A1E 07                                rlca
0A1F 07                                rlca
0A20 CB 39                               srl    c
0A22 1F                                rra
0A23 CB 39                               srl    c
0A25 1F                                rra
0A26 D3 BF                                out    (0BFh), a                ; write VDP control port
0A28 79                                ld     a, c
0A29 F6 78                                or     78h ; 'x'
0A2B D3 BF                                out    (0BFh), a                ; write VDP control port
0A2D
0A2D loc_0A2D:                                    ; CODE XREF: print_msg+30␣j
0A2D 7E                                ld     a, (hl)
0A2E 23                                inc    hl
0A2F B7                                or     a
0A30 28 09                                jr     z, loc_0A3B
0A32 FE FF                                cp     0FFh
0A34 28 E1                                jr     z, loc_0A17
0A36 CD 57 0A                               call   sub_0A57
0A39 18 F2                                jr     loc_0A2D
0A3B
0A3B loc_0A3B:                                    ; CODE XREF: print_msg+27␣j
0A3B C1                                pop    bc
0A3C C9                                ret
0A3C                                ; End of function print_msg
0A3C
0A3D                                ; ██████████ S U B R O U T I N E ██████████
0A3D
0A3D print_hexascii_byte:                            ; CODE XREF: 0003␣j
0A3D F5                                ; print_track_and_dec+2A␣p
0A3D                                push   af
0A3E 0F                                rrca
0A3F 0F                                rrca
0A40 0F                                rrca
0A41 0F                                rrca
0A42 CD 4D 0A                               call   print_hexascii_nibble
0A45 F1                                pop    af
0A46 CD 4D 0A                               call   print_hexascii_nibble
0A49 3E 20                                ld     a, 20h ; ' '
0A4B 18 0A                                jr     sub_0A57
0A4B                                ; End of function print_hexascii_byte
0A4B
0A4D                                ; ██████████ S U B R O U T I N E ██████████
0A4D
0A4D print_hexascii_nibble:                            ; CODE XREF: print_hexascii_byte+5␣p
0A4D E6 0F                                ; print_hexascii_byte+9␣p
0A4D                                and    0Fh
0A4F FE 0A                                cp     0Ah
0A51 38 02                                jr     c, loc_0A55
0A53 C6 07                                add    a, 7
0A55                                ; A-F
0A55 loc_0A55:                                    ; CODE XREF: print_hexascii_nibble+4␣j
0A55 C6 30                                add    a, 30h ; '0'
0A55                                ; convert_to_ascii

```

```

0A55      ; End of function print_hexascii_nibble
0A55
0A57      ; ██████████ SUBROUTINE ██████████
0A57
0A57      sub_0A57:                                ; CODE XREF: print_msg+2D□p
0A57 D3 BE      out      (0BEh), a                        ; print_hexascii_byte+E□j
0A57          ld      a, 8                          ; write VDP data port
0A59 3E 08      out      (0BEh), a                        ; write VDP data port
0A5B D3 BE      ret
0A5D C9
0A5D      ; End of function sub_0A57
0A5D
0A5E      ; ██████████ SUBROUTINE ██████████
0A5E
0A5E      write_VDP_control_word:                  ; CODE XREF: VDP_cls+C□p
0A5E DB BF      in      a, (0BFh)                       ; VDP_cls+22□p ...
0A5E          ld      a, 1                          ; read VDP control port
0A60 7D          out      (0BFh), a                    ; write VDP control port
0A61 D3 BF      ld      a, h
0A63 7C          out      (0BFh), a                    ; write VDP control port
0A64 D3 BF      ret
0A66 C9
0A66      ; End of function write_VDP_control_word
0A66
0A67      ; ██████████ SUBROUTINE ██████████
0A67
0A67      print_char:                              ; CODE XREF: OC0E□p
0A67 07          rlca                                ; OC1E□p ...
0A67          rlca                                ; A=col,B=char,C=row
0A68 07          rlca
0A69 07          rlca
0A6A CB 39      srl      c
0A6C 1F          rra
0A6D CB 39      srl      c
0A6F 1F          rra
0A70 D3 BF      out      (0BFh), a                    ; write VDP control port
0A72 79          ld      a, c
0A73 F6 78      or      78h ; 'x'
0A75 D3 BF      out      (0BFh), a                    ; write VDP control port
0A77 78          ld      a, b
0A78 D3 BE      out      (0BEh), a                    ; write VDP data port
0A7A C9          ret
0A7A      ; End of function print_char
0A7A
0A7B      ; ██████████ SUBROUTINE ██████████
0A7B
0A7B      sub_0A7B:                                ; CODE XREF: add_char_to_buf+E□p
0A7B 07          rlca
0A7C 07          rlca
0A7D 07          rlca
0A7E CB 39      srl      c
0A80 1F          rra
0A81 CB 39      srl      c
0A83 1F          rra
0A84 D3 BF      out      (0BFh), a                    ; write VDP control port
0A86 79          ld      a, c
0A87 F6 78      or      78h ; 'x'
0A89 D3 BF      out      (0BFh), a                    ; write VDP control port
0A8B DD E3      ex      (sp), ix
0A8D DD E3      ex      (sp), ix
0A8F DB BE      in      a, (0BEh)
0A91 DD E3      ex      (sp), ix
0A93 DD E3      ex      (sp), ix
0A95 DB BE      in      a, (0BEh)
0A97 C9          ret
0A97      ; End of function sub_0A7B
0A97
0A98      ; ██████████ SUBROUTINE ██████████
0A98
0A98      init_VDP_and_character_set:              ; CODE XREF: 000C□j
0A98 CD CA 09      call     init_VDP_display_disabled          ; 0BB9□p
0A98          ld      b, 4                          ; output 4 bytes
0A9D 0E 7F      ld      c, 7Fh ; '□'                       ; another port??
0A9F 21 E0 0A      ld      hl, 0AE0h
0AA2 ED B3      otir
0AA4 DB BF      in      a, (0BFh)
0AA6 06 16      ld      b, 22
0AA8 0E BF      ld      c, 0BFh ; 'j'
0AAA 21 E4 0A      ld      hl, 0AE4h
0AAD ED B3      otir
0AAF 21 10 C0      ld      hl, 0C010h
0AB2 CD 5E 0A      call     write_VDP_control_word
0AB5 06 02      ld      b, 2
0AB7 0E BE      ld      c, 0BEh ; 'Y'
0AB9 21 DE 0A      ld      hl, 0ADEh
0ABC ED B3      otir
0ABE 21 00 1C      ld      hl, 1C00h
0AC1 3E 00      ld      a, 0
0AC3 D3 BF      out      (0BFh), a                    ; write VDP control port
0AC5 3E 40      ld      a, 40h ; 'e'
0AC7 D3 BF      out      (0BFh), a                    ; write VDP control port
0AC9 0E 80      ld      c, 80h ; 'C'
0ACB          ; CODE XREF: init_VDP_and_character_set+43□j
0ACB          ld      b, 8                          ; 8 data bytes per character
0ACD
0ACD      loc_0ACD:                                ; CODE XREF: init_VDP_and_character_set+40□j
0ACD 7E          ld      a, (hl)
0ACE 23          inc     hl
0ACF D3 BE      out      (0BEh), a                    ; get character data byte
0AD1 AF          xor     a                                  ; next data ptr
0AD2 D3 BE      out      (0BEh), a                    ; out VDP data port
0AD4 D3 BE      out      (0BEh), a                    ; out VDP data port
0AD6 D3 BE      out      (0BEh), a                    ; out VDP data port
0AD8 10 F3      djnz   loc_0ACD                       ; loop for 8 bytes
0ADA 0D          dec     c
0ADB 20 EE      jr     nz, loc_0ACB                   ; next character
0ADD C9          ret
0ADD          ; loop through all 128 characters
0ADD
0ADD      ; End of function init_VDP_and_character_set
0ADD
0ADE 04 0F      ; ██████████
0ADE          .dw 0F04h
0AE0 9F BF DF FF      .db 9Fh, 0BFh, 0DFh, 0FFh
0AE4 16 80      ; out to port $7F
0AE4          .dw 8016h
0AE6 A0 81      ; Mode Set Register #1
0AE6          .dw 81A0h
0AE8 FF 82      ; Mode Set Register #2
0AE8          .dw 82FFh
0AEA FF 83      ; Pattern Name Table Address for Scroll A
0AEA          .dw 83FFh
0AEC FF 84      ; Pattern Name Table Address for Scroll B
0AEC          .dw 84FFh
0AEE FF 85      ; Sprite Attribute Table Base Address
0AEE          .dw 85FFh
0AF0 FB 86      ; ??
0AF0          .dw 86FBh
0AF2 00 87      ; Backdrop Colour
0AF2          .dw 8700h
0AF4 00 88      ; ??
0AF4          .dw 8800h
0AF6 00 89      ; ??
0AF6          .dw 8900h
0AF8 BF 8A      ; HIRQ register
0AF8          .dw 8ABFh
0AFA          ; ██████████ SUBROUTINE ██████████
0AFA
0AFA      print_track_and_dec:                      ; CODE XREF: read_BC_bytes_to_DE+9□p
0AFA          ; FDC_format_disc+5F□p ...
0AFA C5

```

```

OAF8 push bc
OAFB 3A 9A DE ld a, (0DE9Ah) ; number of tracks
OAFE 3D dec a ; -1
OAFF 32 9A DE ld (0DE9Ah), a ; save
OB02 3C inc a ; +1
OB03 01 00 00 ld bc, 0 ; tens = 0
OB06 ;
OB06 loc_0B06: ; CODE XREF: print_track_and_dec+13[]
OB06 FE 0A cp 10 ; <10?
OB08 38 05 jr c, loc_0B0F ; yes, print digit
OB0A D6 0A sub 10 ; -10
OB0C 04 inc b ; count tens
OB0D 18 F7 jr loc_0B06 ; loop until <10
OB0F ;
OB0F loc_0B0F: ; CODE XREF: print_track_and_dec+E[]
OB0F 4F ld c, a
OB10 DB BF in a, (0BFh) ; read VDP control port
OB12 3E 5C ld a, 5Ch ; '\
OB14 D3 BF out (0BFh), a ; write VDP control port
OB16 3E 7C ld a, 7Ch ; '|'
OB18 D3 BF out (0BFh), a ; write VDP control port
OB1A 78 ld a, b
OB1B CB 27 sla a
OB1D CB 27 sla a
OB1F CB 27 sla a
OB21 CB 27 sla a
OB23 B1 or c
OB24 CD 3D 0A call print_hexascii_byte
OB27 C1 pop bc
OB28 C9 ret
OB28 ; End of function print_track_and_dec
OB29 ;
OB29 copy_cart_ROM_to_DRAM: ; CODE XREF: 000F[]
OB29 AF xor a
OB2A 32 01 20 ld (2001h), a ; run BIOS from ROM w/cart & DRAM
OB2D ;
OB2D loc_0B2D: ; CODE XREF: 0B53[]
OB2D 32 00 20 ld (2000h), a ; set rom/dram page
OB30 ED 47 ld i, a ; zero bank count
OB32 21 00 60 ld hl, 6000h ; upper 8kB of cart ROM bank
OB35 11 00 A0 ld de, 0A000h ; upper 8kB of copier DRAM bank
OB38 01 00 20 ld bc, 8192
OB3B D9 exx
OB3C 21 00 40 ld hl, 4000h ; lower 8kB of cart ROM bank
OB3F 11 00 80 ld de, 8000h ; lower 8kB of copier DRAM bank
OB42 ;
OB42 loc_0B42: ; CODE XREF: 0B48[]
OB42 ED A0 ldi ; copy byte from lower 8kB ROM to DRAM
OB44 D9 exx
OB45 ED A0 ldi ; copy byte from upper 8kB ROM to DRAM
OB47 D9 exx
OB48 EA 42 0B jp pe, loc_0B42 ; loop for 8192 words (16kB)
OB4B ED 57 ld a, i ; get bank count
OB4D 3C inc a
OB4E ED 4B F1 DF ld bc, (0DFF1h) ; cartridge size (16kB pages)
OB52 B9 cp c ; done all banks?
OB53 38 D8 jr c, loc_0B2D ; no, loop
OB55 C9 ret
OB56 ;
OB56 ; SUBROUTINE
OB56 delay: ; CODE XREF: sub_01B5+3E[]p
OB56 C5 push bc
OB57 01 00 00 ld bc, 0
OB5A ;
OB5A loc_0B5A: ; CODE XREF: delay+7[]j
OB5A 0B dec bc
OB5B 78 ld a, b
OB5C B1 or c
OB5D 20 FB jr nz, loc_0B5A
OB5F ;
OB5F loc_0B5F: ; CODE XREF: delay+C[]j
OB5F 0B dec bc
OB60 78 ld a, b
OB61 B1 or c
OB62 20 FB jr nz, loc_0B5F
OB64 C1 pop bc
OB65 C9 ret
OB65 ; End of function delay
OB66 ;
OB66 print_msg_and_exit_option: ; CODE XREF: 0180[]j
OB66 F5 ; 0190[]j ...
OB66 CD EC 08 push af
OB67 F1 call bring_FDC_out_of_reset
OB68 21 AD DE pop af
OB69 7E ld hl, 0DEADh
OB6A 7E bit 7, (hl)
OB6B 28 15 jr z, exit_option
OB6C 09 0A call print_msg
OB6D ;
OB6D loc_0B75: ; CODE XREF: 0B7C[]j
OB6D CD 99 0B call handle_pc_cmd_read_controller
OB6E 3A 95 DF ld a, (0DF95h) ; read controller input
OB6F B7 or a ; anything pressed?
OB70 20 F7 jr nz, loc_0B75 ; yes, loop
OB71 ;
OB71 loc_0B7E: ; CODE XREF: 0B85[]j
OB71 CD 99 0B call handle_pc_cmd_read_controller
OB72 3A 95 DF ld a, (0DF95h) ; read controller input
OB73 B7 or a ; anything pressed?
OB74 28 F7 jr z, loc_0B7E ; yes, loop
OB75 ;
OB75 exit_option: ; CODE XREF: 0B70[]j
OB75 ED 7B 8C DF ld sp, (0DF8Ch) ; 0B8F[]j ...
OB76 C9 ret
OB77 ;
OB77 call bring_FDC_out_of_reset
OB78 C3 87 0B jp exit_option
OB79 ;
OB79 ; SUBROUTINE
OB79 wait_for_controller: ; CODE XREF: sub_01B5+A2[]p
OB79 CD 99 0B ; wait_for_controller+4[]j
OB79 call handle_pc_cmd_read_controller
OB7A B7 or a ; any buttons pressed?
OB7B 28 FA jr z, wait_for_controller ; no, loop
OB7C C9 ret
OB7C ; End of function wait_for_controller
OB7D ;
OB7D ; SUBROUTINE
OB7D handle_pc_cmd_read_controller: ; CODE XREF: 0953[]p
OB7D 3A 03 20 ; 095C[]p ...
OB7D ld a, (2003h) ; read PC I/O status
OB7E CB 7F bit 7, a ; PC busy?
OB7F 28 03 jr z, read_controller ; yes, skip
OB80 CD 55 1A call handle_pc_command
OB81 ;
OB81 read_controller: ; CODE XREF: handle_pc_cmd_read_controller+5[]j
OB81 DB DC in a, (0DCh) ; read controller
OB83 E6 3F and 3Fh, '?' ; mask off unwanted bits

```

```

0BA7 EE 3F          xor     3Fh ; '?'          ; active high
0BA9 32 95 DF      ld     (0DF95h), a        ; store
0BAC C9           ret
; End of function handle_pc_cmd_read_controller
0BAD
0BAD
0BAD
0BAD
0BAD DB DC      wait_for_no_controller_press: ; CODE XREF: wait_for_no_controller_press+6[]
0BAD          in     a, (0DCh)        ; get_menu_selection+5[]
0BAF E6 3F      and     3Fh ; '?'          ; read controller
0BB1 FE 3F      cp     3Fh ; '?'          ; mask off unwanted bits
0BB3 20 F8      jr     nz, wait_for_no_controller_press ; any presses?
0BB5 C9           ret                        ; yes, loop
0BB5
; End of function wait_for_no_controller_press
0BB6
;
0BB6
0BB6
0BB6 32 92 DF      MAIN: ld     (0DF92h), a        ; CODE XREF: 008A[]
0BB9 CD 98 0A    call   init_VDP_and_character_set ; store main/utility menu
0BBC
init_main_loop: ; CODE XREF: 0C3C[]
0BBC AF          xor     a
0BBD 32 91 DF      ld     (0DF91h), a        ; current menu option
0BC0 3A 92 DF      ld     a, (0DF92h)        ; main/utility menu
0BC3 FE 02      cp     z
0BC5 38 04      jr     c, main_loop      ; valid?
0BC7 AF          xor     a                  ; yes, skip
0BC8 32 92 DF      ld     (0DF92h), a        ; set to main
0BCB
main_loop:      ; CODE XREF: 0BC5[]
0BCB CB 27      ; CODE XREF: 0C68[]
0BCB          sla     a
0BCD 4F          ld     c, a
0BCE 06 00      ld     b, 0
0BD0 21 6A 04    ld     hl, 46Ah          ; table with number of routines in each menu
0BD3 09          add     hl, bc
0BD4 7E          ld     a, (hl)
0BD5 32 90 DF      ld     (0DF90h), a        ; get number of routines in this menu
0BD8 21 6E 04    ld     hl, 46Eh          ; store number of routines in menu
0BDB 09          add     hl, bc            ; table of routines for each menu
0BDC 7E          ld     a, (hl)
0BDD 32 8E DF      ld     (0DF8Eh), a        ; get menu routine table
0BE0 23          inc     hl
0BE1 7E          ld     a, (hl)
0BE2 32 8F DF      ld     (0DF8Fh), a        ; store table for menu routines
0BE5 21 66 04    ld     hl, 466h
0BE8 09          add     hl, bc
0BE9 7E          ld     a, (hl)
0BEA 32 93 DF      ld     (0DF93h), a        ; get ??? for current menu
0BED 23          inc     hl
0BEE 7E          ld     a, (hl)
0BEF 32 94 DF      ld     (0DF94h), a        ; get ??? for current menu
0BF2 AF          xor     a
0BF3 32 EF DE      ld     (0DEEFh), a
0BF6 CD D5 09    call   VDP_cls           ; main/utility menu
0BF9 3A 92 DF      ld     a, (0DF92h)
0BF0 C6 11      add     a, 11h
0BFE CD 09 0A    call   print_msg         ; DRAM size (16kB pages)
0C01 3A F0 DF      ld     a, (0DF0h)
0C04 0F          rrca
0C05 0F          rrca
0C06 0F          rrca
0C07 C6 30      add     a, 30h ; '0'
0C09 47          ld     b, a
0C0A 3E 1B      ld     a, 1Bh
0C0C 0E 0D      ld     c, 0Dh
0C0E CD 67 0A    call   print_char        ; print DRAM size
0C11 3A F1 DF      ld     a, (0DF1h)
0C14 0F          rrca
0C15 0F          rrca
0C16 0F          rrca
0C17 C6 30      add     a, 30h ; '0'
0C19 47          ld     b, a
0C1A 3E 1B      ld     a, 1Bh
0C1C 0E 0F      ld     c, 0Fh
0C1E CD 67 0A    call   print_char        ; print cartridge size
0C21 3E 80      ld     a, 80h ; 'C'
0C23 32 AD DE    ld     (0DEADh), a
0C26
loc_0C26:      ; CODE XREF: 0C32[]
0C26 CD 70 0C    call   get_menu_selection ; read controller input
0C29 3A 95 DF      ld     a, (0DF95h)
0C2C CB 6F      bit     5, a              ; button C pressed?
0C2E 20 0F      jr     nz, handle_meu_option ; yes, skip
0C30 E6 0C      and     0Ch              ; left/right?
0C32 28 F2      jr     z, loc_0C26       ; no, skip
0C34 3A 92 DF      ld     a, (0DF92h)
0C37 EE 01      xor     l
0C39 32 92 DF      ld     (0DF92h), a
0C3C C3 BC 0B    jp     init_main_loop    ; store main/utility menu flag
0C3F
;
0C3F
0C3F
0C3F CD D5 09      handle_meu_option:      ; CODE XREF: 0C2E[]
0C42 3A 91 DF      call   VDP_cls           ; get current menu option
0C45 CB 27      sla     a
0C47 4F          ld     c, a
0C48 06 00      ld     b, 0
0C4A 2A 8E DF      ld     hl, (0DF8Eh)
0C4D 09          add     hl, bc
0C4E 4D          ld     c, l
0C4F 44          ld     b, h
0C50 0A          ld     a, (bc)
0C51 6F          ld     l, a
0C52 03          inc     bc
0C53 0A          ld     a, (bc)
0C54 67          ld     h, a
0C55 3A 92 DF      ld     a, (0DF92h)
0C58 F5          push   af                ; hl = routine for this option
0C59 3A 91 DF      ld     a, (0DF91h)
0C5C F5          push   af                ; main/utility menu
0C5D CD 6B 0C    call   call_function_at_HL ; current menu option
0C60 F1          pop    af
0C61 32 91 DF      ld     (0DF91h), a
0C64 F1          pop    af                ; restore current menu option
0C65 32 92 DF      ld     (0DF92h), a
0C68 C3 CB 0B    jp     main_loop        ; restore main/utility menu
0C6B
;
0C6B
0C6B
0C6B
0C6B ED 73 8C DF    call_function_at_HL:    ; CODE XREF: 0C5D[]
0C6F E9          ld     (0DF8Ch), sp
0C6F          jp     (hl)
; End of function call_function_at_HL
0C6F
0C70
;
0C70
0C70
0C70
0C70
0C70 06 10      get_menu_selection:    ; CODE XREF: 0C26[]
0C70          ld     b, 10h           ; get_menu_selection+2B[] ...
0C72 CD A5 0C    call   print_menu_item_marker ; right-triangle symbol
0C75 CD AD 0B    call   wait_for_no_controller_press
0C78
loc_0C78:      ; CODE XREF: get_menu_selection+F[]

```



```

OD35
OD36
OD36 ; ██████████ S U B R O U T I N E ██████████
OD36
OD36 detect_disk: ; CODE XREF: get_file_cluster□p
OD36 AF ; sub_1389□p ...
OD36 xor a
OD37 32 B2 DE ld (0DEB2h), a ; clear media type
OD3A CD C8 0D call FDC_init_and_SPECIFY
OD3D CD 15 0D call FDC_RECALIBRATE_twice
OD40 CD 6F 09 call check_disc_present
OD43 AF xor a ; transfer rate 500kbps
OD44
OD44 loc_0D44: ; CODE XREF: detect_disk+16□j
OD44 CD F5 0D call FDC_READ_DATA_command
OD47 38 18 jr c, get_disk_info ; jump if successful
OD49 3C inc a ; next slower transfer rate
OD4A FE 03 cp 3 ; all exhausted?
OD4C 38 F6 jr c, loc_0D44 ; no, loop
OD4E 3A EF DE ld a, (0DEEFh)
OD51 B7 or a
OD52 28 08 jr z, loc_0D5C ; handle read error
OD54 CD C8 0D call FDC_init_and_SPECIFY
OD57 F1 pop af
OD58 F1 pop af
OD59 C3 63 01 jp loc_0163
OD5C
OD5C loc_0D5C: ; CODE XREF: detect_disk+1C□j
OD5C 3E 05 ld a, 5 ; "READ ERROR"
OD5E C3 66 0B jp print_msg_and_exit_option
OD61
OD61 get_disk_info: ; CODE XREF: detect_disk+11□j
OD61 CD C8 0D call FDC_init_and_SPECIFY
OD64 CD 85 09 call check_sig_trk80
OD67 CD C8 0D call FDC_init_and_SPECIFY
OD6A CD 15 0D call FDC_RECALIBRATE_twice
OD6D CD 6F 09 call check_disc_present
OD70 CD 9E 0E call flush_sector_if_required
OD73 3A 11 DA ld a, (0DA11h) ; get number of root directory entries
OD76 32 B0 DE ld (0DEB0h), a ; store max root directory entries
OD79 CB 3F srl a
OD7B CB 3F srl a
OD7D CB 3F srl a
OD7E CB 3F srl a
OD81 32 90 DE ld (0DE90h), a
OD84 3A 16 DA ld a, (0DA16h) ; get sectors for each FAT
OD87 CB 27 sla a
OD89 3C inc a
OD8A 32 AE DE ld (0DEAEh), a
OD8D 21 90 DE ld hl, 0DE90h
OD90 86 add a, (hl)
OD91 32 AF DE ld (0DEAFh), a
OD94 3A 18 DA ld a, (0DA18h) ; get sectors/track
OD97 32 B1 DE ld (0DEB1h), a ; store
OD9A 3A 0D DA ld a, (0DA0Dh)
OD9D CB 3F srl a
OD9F CB 3F srl a
ODA1 3A 15 DA ld a, (0DA15h) ; get media type
ODA4 1F rra
ODA5 CB 3F srl a
ODA7 E6 41 and 41h, 'A'
ODA9 32 B2 DE ld (0DEB2h), a ; store
ODAC 2A 13 DA ld hl, (0DA13h) ; number of sectors in filesystem
ODAF ED 4B AF DE ld bc, (0DEAFh)
ODB3 06 00 ld b, 0
ODB5 B7 or a
ODB6 ED 42 sbc hl, bc
ODB8 3A B2 DE ld a, (0DEB2h) ; get media type
ODBB CB 77 bit 6, a
ODBD 28 04 jr z, loc_0DC3
ODBF CB 3C srl h
ODC1 CB 1D rr l
ODC3
ODC3 loc_0DC3: ; CODE XREF: detect_disk+87□j
ODC3 23 inc hl
ODC4 22 84 DE ld (0DE84h), hl
ODC7 C9 ret
ODC7 ; End of function detect_disk
ODC7
ODC8
ODC8 ; ██████████ S U B R O U T I N E ██████████
ODC8
ODC8 FDC_init_and_SPECIFY: ; CODE XREF: detect_disk+4□p
ODC8 21 ED 0D ; detect_disk+1E□p ...
ODCB 11 B4 DE ld hl, 0DEDh ; FDC command parameter block init values
ODCE 01 08 00 ld de, 0DEB4h ; FDC command parameter block
ODD1 ED B0 ldir ; init FDC command block
ODD3 3E 29 ld a, 29h ; ';'
ODD5 32 0B 20 ld (200Bh), a ; reset FDC (drive 1 enabled)
ODD8 00 nop ; FDC digital output register
ODD9 00 nop
ODDA 00 nop
ODDB CD F2 08 call enable_drive_1
ODDE 3E 03 ld a, 3 ; FDC SPECIFY command
ODE0 CD F8 08 call send_byte_to_FDC
ODE3 3E DF ld a, 0DFh ; '█' ; step rate / unload time
ODE5 CD F8 08 call send_byte_to_FDC
ODE8 3E 03 ld a, 3 ; head load time
ODEA C3 F8 08 jp send_byte_to_FDC
ODEA ; End of function FDC_init_and_SPECIFY
ODEA
ODEA
ODED 01 .db 1 ; unit select
ODEE 00 .db 0 ; track number
ODEF 00 .db 0 ; drive head address
ODF0 01 .db 1 ; sector number
ODF1 02 .db 2 ; number of bytes / sector
ODF2 01 .db 1 ; end of track sector number
ODF3 1B .db 1Bh ; in sector gap length
ODF4 FF .db 0FFh ; data length
ODF5
ODF5 ; ██████████ S U B R O U T I N E ██████████
ODF5
ODF5 FDC_READ_DATA_command: ; CODE XREF: detect_disk+E□p
ODF5 32 0E 20 ld (200Eh), a ; FDC control register
ODF8 32 90 DE ld (0DE90h), a
ODFB CD C8 0D call FDC_init_and_SPECIFY
ODFE 3E 46 ld a, 46h ; 'F' ; FDC READ DATA command
OE00 32 B3 DE ld (0DEB3h), a
OE03 CD 32 0E call send_9_byte_FDC_cmd_0DEB3
OE06
OE06 loc_0E06: ; CODE XREF: FDC_READ_DATA_command+16□j
OE06 3A 0D 20 ld a, (200Dh) ; FDC status
OE09 CB 7F bit 7, a ; ready?
OE0B 28 F9 jr z, loc_0E06 ; no, loop
OE0D CB 27 sla a
OE0F CB 27 sla a
OE11 CB 27 sla a
OE13 3A 90 DE ld a, (0DE90h)
OE16 C9 ret
OE16 ; End of function FDC_READ_DATA_command
OE16
OE17
OE17 ; ██████████ S U B R O U T I N E ██████████
OE17

```



```

0EF1      ; ██████████ S U B R O U T I N E ██████████
0EF1
0EF1      FDC_write_512_bytes:                ; CODE XREF: flush_sector_if_required+23□p
0EF1 3A B2 DE                                ; write_dirent_from_buffer+D□□ ...
0EF1      ld      a, (0DEB2h)                    ; get media type
0EF4 CB 9F                                res      3, a
0EF6 32 B2 DE                                ld      (0DEB2h), a                ; set media type
0EF9 3A 9E DE                                ld      a, (0DE9Eh)                ; get track (write)
0EFC CD F1 0C                                call    sub_0CF1
0EFF 3A 9F DE                                ld      a, (0DE9Fh)                ; drive head address (write)
0F02 CB 27                                sla     a
0F04 CB 27                                sla     a
0F06 F6 01                                or      1
0F08 32 B4 DE                                ld      (0DEB4h), a                ; unit select
0F0B 3A A0 DE                                ld      a, (0DEA0h)                ; get sector (write)
0F0E 32 B9 DE                                ld      (0DEB9h), a                ; end of track sector number
0F11 11 00 DA                                ld      de, 0DA00h                 ; data buffer
0F14 21 0D 20                                ld      hl, 200Dh                 ; FDC status register
0F17 3E 45                                ld      a, 45h : 'E'              ; WRITE DATA
0F19 CD F8 08                                call    send_byte_to_FDC
0F1C 3A B4 DE                                ld      a, (0DEB4h)                ; unit select
0F1F CD F8 08                                call    send_byte_to_FDC
0F22 3A 9E DE                                ld      a, (0DE9Eh)                ; track number
0F25 CD F8 08                                call    send_byte_to_FDC
0F28 3A 9F DE                                ld      a, (0DE9Fh)                ; drive head address
0F2B CD F8 08                                call    send_byte_to_FDC
0F2E 3A A0 DE                                ld      a, (0DEA0h)                ; sector number
0F31 CD F8 08                                call    send_byte_to_FDC
0F34 3A B8 DE                                ld      a, (0DEB8h)                ; number of bytes per sector
0F37 CD F8 08                                call    send_byte_to_FDC
0F3A 3A B9 DE                                ld      a, (0DEB9h)                ; end track sector number
0F3D CD F8 08                                call    send_byte_to_FDC
0F40 3A BA DE                                ld      a, (0DEBAh)                ; in sector gap length
0F43 CD F8 08                                call    send_byte_to_FDC
0F46 3A BB DE                                ld      a, (0DEBBh)                ; data length
0F49 CD F8 08                                call    send_byte_to_FDC
0F4C
0F4C      loc_0F4C:                                ; CODE XREF: FDC_write_512_bytes+64□□
0F4C 1A                                ld      a, (de)                    ; get data byte
0F4D
0F4D      loc_0F4D:                                ; CODE XREF: FDC_write_512_bytes+5E□□
0F4D CB 7E                                bit     7, (hl)                    ; FDC ready?
0F4F 28 FC                                jr      z, loc_0F4D                ; no, loop
0F51 32 0C 20                                ld      (200Ch), a                ; write FDC data
0F54 1C                                inc     e                            ; next data byte
0F55 C2 4C 0F                                jp      nz, loc_0F4C                ; loop until 256 bytes done
0F58 14                                inc     d                            ; next data byte
0F59
0F59      loc_0F59:                                ; CODE XREF: FDC_write_512_bytes+71□□
0F59 1A                                ld      a, (de)                    ; get data byte
0F5A
0F5A      loc_0F5A:                                ; CODE XREF: FDC_write_512_bytes+6B□□
0F5A CB 7E                                bit     7, (hl)                    ; FDC ready?
0F5C 28 FC                                jr      z, loc_0F5A                ; no, loop
0F5E 32 0C 20                                ld      (200Ch), a                ; write FDC data
0F61 1C                                inc     e                            ; next data byte
0F62 C2 59 0F                                jp      nz, loc_0F59                ; loop until 2nd 256 bytes done
0F65 C3 3F 0E                                jp      read_FDC_result_ST0_2
0F65      ; End of function FDC_write_512_bytes
0F65
0F68      ; ██████████ S U B R O U T I N E ██████████
0F68
0F68      read_dirent:                                ; CODE XREF: read_dirent_into_buffer□p
0F68 3A A2 DE                                ; write_dirent_from_buffer□p
0F68      ld      a, (0DEA2h)                    ; current dirent number
0F6B CB 3F                                srl     a
0F6D CB 3F                                srl     a
0F6F CB 3F                                srl     a
0F71 CB 3F                                srl     a                            ; calc sector for dirent
0F73 21 AE DE                                ld      hl, 0DEAEh
0F76 86                                add     a, (hl)
0F77 32 8E DE                                ld      (0DE8Eh), a
0F7A AF                                xor     a
0F7B 32 8F DE                                ld      (0DE8Fh), a
0F7E CD DF 10                                call    sub_10DF
0F81 CD 9E 0E                                call    flush_sector_if_required
0F84 21 00 DA                                ld      hl, 0DA00h
0F87 ED 4B A2 DE                                ld      bc, (0DEA2h)                ; current dirent number
0F8B 06 00                                ld      b, 0
0F8D CB 21                                sla     c
0F8F CB 21                                sla     c
0F91 CB 21                                sla     c
0F93 CB 21                                sla     c
0F95 CB 21                                sla     c                            ; x32
0F97 CB 10                                r1     b
0F99 09                                add     hl, bc
0F9A C9                                ret
0F9A      ; End of function read_dirent
0F9A
0F9A      ; ██████████ S U B R O U T I N E ██████████
0F9B
0F9B      read_dirent_into_buffer:                ; CODE XREF: find_file_at_$FD80+13□p
0F9B CD 68 0F                                ; sub_1389+1C□p ...
0F9B      call    read_dirent
0F9E 11 CE DE                                ld      de, 0DECEh                ; directory entry buffer
0FA1 01 20 00                                ld      bc, 32
0FA4 ED B0                                ldir
0FA6 C9                                ret
0FA6      ; End of function read_dirent_into_buffer
0FA6
0FA7      ; ██████████ S U B R O U T I N E ██████████
0FA7
0FA7      write_dirent_from_buffer:                ; CODE XREF: sub_1389+6D□p
0FA7 CD 68 0F                                ; 1A07□p ...
0FA7      call    read_dirent
0FAA 5D                                ld      e, l
0FAB 54                                ld      d, h
0FAC 21 CE DE                                ld      hl, 0DECEh                ; directory entry buffer
0FAF 01 20 00                                ld      bc, 32
0FB2 ED B0                                ldir
0FB4 C3 F1 0E                                jp      FDC_write_512_bytes
0FB4      ; End of function write_dirent_from_buffer
0FB4
0FB7      ; ██████████ S U B R O U T I N E ██████████
0FB7
0FB7      sub_0FB7:                                ; CODE XREF: sub_110F+18□p
0FB7 2A 88 DE                                ; sub_11DD+18□p ...
0FB7      ld      hl, (0DE88h)
0FB7      ld      c, l
0FBB 44                                ld      b, h
0FBC CB 3C                                srl     h
0FBE CB 1D                                rr      l
0FC0 09                                add     hl, bc
0FC1 22 90 DE                                ld      (0DE90h), hl
0FC4 CB 41                                bit     0, c
0FC6 20 16                                jr      nz, loc_0FDE
0FC8 CD 08 10                                call    sub_1008
0FCB 32 80 DE                                ld      (0DE80h), a
0FCE 2A 90 DE                                ld      hl, (0DE90h)
0FD1 23                                inc     hl
0FD2 22 90 DE                                ld      (0DE90h), hl
0FD5 CD 08 10                                call    sub_1008

```



```

OFD8 E6 0F          and    0Fh
OFDA 32 81 DE      ld     (0DE81h), a
OFDD C9           ret
;
OFDE
OFDE
OFDE CD 08 10      loc_OFDE:      call    sub_1008          ; CODE XREF: sub_0FB7+F0j
OFE1 32 80 DE      ld     (0DE80h), a
OFE4 2A 90 DE      ld     hl, (0DE90h)
OFE7 23           inc     hl
OFE8 22 90 DE      ld     (0DE90h), hl
OFE8 CD 08 10      call    sub_1008
OFE8 32 81 DE      ld     (0DE81h), a
OFF1 2A 80 DE      ld     hl, (0DE80h)
OFF4 CB 3C         srl    h
OFF6 CB 1D         rr     l
OFF8 CB 3C         srl    h
OFFA CB 1D         rr     l
OFFC CB 3C         srl    h
OFFE CB 1D         rr     l
1000 CB 3C         srl    h
1002 CB 1D         rr     l
1004 22 80 DE      ld     (0DE80h), hl
1007 C9           ret
; End of function sub_0FB7
1007
1008
1008
1008
1008
1008
1008 3A 91 DE      sub_1008:      call    sub_1008          ; CODE XREF: sub_0FB7+110p
; sub_0FB7+1E0p ...
1008             ld     a, (0DE91h)
100B CB 3F         srl    a
100D C6 02         add    a, 2
100F 32 B7 DE      ld     (0DEB7h), a          ; save sector number
1012 AF           xor    a
1013 32 B5 DE      ld     (0DEB5h), a          ; save track number (0)
1016 32 B6 DE      ld     (0DEB6h), a          ; save drive head address (0)
1019 CD 9E 0E      call    flush_sector_if_required
101C 21 00 DA      ld     hl, 0DA00h
101F ED 4B 90 DE  ld     bc, (0DE90h)
1023 3E 01         ld     a, 1
1025 A0           and    b
1026 47           ld     b, a
1027 09           add    hl, bc
1028 7E           ld     a, (hl)
1029 C9           ret
; End of function sub_1008
1029
1029
102A
102A
102A
102A
102A 2A 88 DE      sub_102A:      call    sub_11DD+270p      ; CODE XREF: sub_11DD+270p
; sub_120A+170j ...
102A             ld     hl, (0DE88h)          ; cluster numbr of file in current directory entry
102D 4D           ld     c, 1
102E 44           ld     b, h
102F CB 3C         srl    h
1031 CB 1D         rr     l
1033 09           add    hl, bc
1034 22 90 DE      ld     (0DE90h), hl
1037 CB 41         bit    0, c
1039 20 17         jr    nz, loc_1052
103B 3A 80 DE      ld     a, (0DE80h)
103E 0E 00         ld     c, 0
1040 CD 7A 10      call    sub_107A
1043 2A 90 DE      ld     hl, (0DE90h)
1046 23           inc    hl
1047 22 90 DE      ld     (0DE90h), hl
104A 3A 81 DE      ld     a, (0DE81h)
104D 0E F0         ld     c, 0F0h ; '-'
104F C3 7A 10      jp     sub_107A
;
1052
1052
1052 2A 80 DE      loc_1052:      call    sub_102A+F0j      ; CODE XREF: sub_102A+F0j
1055 CB 25         sla    l
1057 CB 14         rl     h
1059 CB 25         sla    l
105B CB 14         rl     h
105D CB 25         sla    l
105F CB 14         rl     h
1061 CB 25         sla    l
1063 CB 14         rl     h
1065 E5           push   hl
1066 7D           ld     a, 1
1067 0E 0F         ld     c, 0Fh
1069 CD 7A 10      call    sub_107A
106C 2A 90 DE      ld     hl, (0DE90h)
106F 23           inc    hl
1070 22 90 DE      ld     (0DE90h), hl
1073 E1           pop    hl
1074 7C           ld     a, h
1075 0E 00         ld     c, 0
1077 C3 7A 10      jp     sub_107A
; End of function sub_102A
1077
107A
107A
107A
107A
107A
107A 32 A4 DE      sub_107A:      call    sub_102A+160p      ; CODE XREF: sub_102A+160p
; sub_102A+250j ...
107A             ld     (0DEA4h), a
107D 79           ld     a, c
107E 32 A5 DE      ld     (0DEA5h), a
1081 3A 91 DE      ld     a, (0DE91h)
1084 CB 3F         srl    a
1086 C6 02         add    a, 2
1088 32 B7 DE      ld     (0DEB7h), a          ; save sector number
108B AF           xor    a
108C 32 B5 DE      ld     (0DEB5h), a          ; save track number (0)
108F 32 B6 DE      ld     (0DEB6h), a          ; save drive head address (0)
1092 CD 9E 0E      call    flush_sector_if_required
1095 21 B2 DE      ld     hl, 0DEB2h          ; media type
1098 CB DE         set    3, (hl)
109A 21 00 DA      ld     hl, 0DA00h
109D ED 4B 90 DE  ld     bc, (0DE90h)
10A1 3E 01         ld     a, 1
10A3 A0           and    b
10A4 47           ld     b, a
10A5 09           add    hl, bc
10A6 7E           ld     a, (hl)
10A7 4D           ld     c, l
10A8 44           ld     b, h
10A9 21 A5 DE      ld     hl, 0DEA5h
10AC A6           and    (hl)
10AD 21 A4 DE      ld     hl, 0DEA4h
10B0 B6           or     (hl)
10B1 02           ld     (bc), a
10B2 C9           ret
; End of function sub_107A
10B2
10B2
10B3
10B3
10B3
10B3
10B3 2A 84 DE      sub_10B3:      ld     hl, (0DE84h)          ; CODE XREF: sub_110F+210p

```

```

10B6 ED 5B 88 DE      ld      de, (0DE88h)          ; cluster of file in current directory entry
10BA B7              or      a
10BB ED 52          sbc    hl, de
10BD 30 05          jr     nc, loc_10C4
10BF 3E 13          ld     a, 13h
10C1 C3 66 0B          jp     print_msg_and_exit_option
10C4                ; -----
10C4                loc_10C4:
10C4 2A 88 DE          ld     hl, (0DE88h)          ; CODE XREF: sub_10B3+A[]
10C7 2B            dec    hl                    ; cluster of file in current directory entry
10C8 2B            dec    hl
10C9 3A B2 DE          ld     a, (0DEB2h)          ; get media type
10CC CB 77          bit    6, a
10CE 28 04          jr     z, loc_10D4
10D0 CB 25          sla    l
10D2 CB 14          r1     h
10D4                loc_10D4:
10D4 ED 5B AF DE          ld     de, (0DEAFh)          ; CODE XREF: sub_10B3+1B[]
10D8 16 00          ld     c, 0
10DA 19          add    hl, de
10DB 22 8E DE          ld     (0DE8Eh), hl
10DE C9            ret
10DE                ; End of function sub_10B3
10DF                ; ----- S U B R O U T I N E -----
10DF                sub_10DF:
10DF 3A 8E DE          ; CODE XREF: read_dirent+16[]p
10DF                ; sub_110F+24[]p
10DF                ld     a, (0DE8Eh)
10E2 4F            ld     c, a
10E3 3A 8F DE          ld     a, (0DE8Fh)
10E6 67            ld     h, a
10E7 AF            xor    a
10E8 5F            ld     e, a
10E9 3A B1 DE          ld     a, (0DEB1h)
10EC 6F            ld     l, a
10ED 06 10          ld     b, 10h
10EF                loc_10EF:
10EF                ; CODE XREF: sub_10DF+1C[]j
10EF CB 21          sla    c
10F1 CB 14          r1     h
10F3 CB 13          r1     e
10F5 7B            ld     a, e
10F6 95            sub    l
10F7 38 02          jr     c, loc_10FB
10F9 0C            inc    c
10FA 5F            ld     e, a
10FB                loc_10FB:
10FB                ; CODE XREF: sub_10DF+18[]j
10FB 10 F2          djnz   loc_10EF
10FD 79            ld     a, c
10FE E6 01          and    l
1100 32 B6 DE          ld     (0DEB6h), a          ; save drive head address
1103 CB 39          srl    c
1105 1C            inc    e
1106 79            ld     a, c
1107 32 B5 DE          ld     (0DEB5h), a          ; save track number
110A 7B            ld     a, e
110B 32 B7 DE          ld     (0DEB7h), a          ; save sector number
110E C9            ret
110E                ; End of function sub_10DF
110F                ; ----- S U B R O U T I N E -----
110F                sub_110F:
110F                ; CODE XREF: 15C1[]p
110F 3A B2 DE          ; sub_15EF+18[]p ...
110F                ; get media type
1112 CB 7F          bit    7, a
1114 28 1A          jr     z, loc_1130
1116 CB 77          bit    6, a
1118 28 0D          jr     z, loc_1127
111A CB 6F          bit    5, a
111C 28 09          jr     z, loc_1127
111E 2A 8E DE          ld     hl, (0DE8Eh)
1121 23            inc    hl
1122 22 8E DE          ld     (0DE8Eh), hl
1125 18 0C          jr     loc_1133
1127                ; -----
1127                loc_1127:
1127                ; CODE XREF: sub_110F+9[]j
1127                ; sub_110F+D[]j
112A 2A 80 DE          call   sub_0FB7
112D 22 88 DE          ld     hl, (0DE80h)
1130                ; save cluster number of file in current directory entry
1130                loc_1130:
1130 CD B3 10          call   sub_10B3          ; CODE XREF: sub_110F+5[]j
1133                loc_1133:
1133 CD DF 10          call   sub_10DF          ; CODE XREF: sub_110F+16[]j
1136 3A B2 DE          ld     a, (0DEB2h)          ; get media type
1139 EE 20          xor    20h, ' '
113B F6 80          or     80h, 'C'
113D 32 B2 DE          ld     (0DEB2h), a          ; store media type
1140 CB 57          bit    2, a
1142 28 03          jr     z, loc_1147
1144 C3 4A 11          jr     loc_114A
1147                ; -----
1147                loc_1147:
1147                ; CODE XREF: sub_110F+33[]j
1147                ; -----
114A                loc_114A:
114A                ; CODE XREF: sub_110F+35[]j
114A                ; FDC READ DATA command
114A 3E 46          ld     a, 46h, 'F'
114C CD 17 0E          call   seek_and_FDC_command
114F 21 0D 20          ld     hl, 200Dh          ; FDC status
1152 ED 5B 82 DE          ld     de, (0DE82h)          ; get address for disk read data
1156 01 00 02          ld     bc, 512
1159                loc_1159:
1159                ; CODE XREF: sub_110F+4C[]j
1159 7E            ld     a, (hl)          ; sub_110F+51[]j
1159                ; read FDC status
115A 07            rlca          ; ready?
115B 30 FC          jr     nc, loc_1159          ; no, loop
115D 2D            dec    l          ; FDC data register
115E ED A0          ldi    ; read FDC data
1160 EA 59 11          jp     pe, loc_1159          ; loop for 512 bytes
1163 ED 53 82 DE          ld     (0DE82h), de          ; save new address for disk rad data
1167 C3 3F 0E          jp     read_FDC_result_ST0_2
116A                ; -----
116A                loc_116A:
116A                ; CODE XREF: sub_110F+38[]j
116A 3E 45          ld     a, 45h, 'E'          ; FDC WRITE DATA command
116C CD 17 0E          call   seek_and_FDC_command
116F 21 0D 20          ld     hl, 200Dh          ; FDC status register
1172 ED 5B 82 DE          ld     de, (0DE82h)          ; get address for disk read data
1176                loc_1176:
1176                ; CODE XREF: sub_110F+70[]j
1176 1A            ld     a, (de)          ; get FDC data byte
1177                loc_1177:
1177                ; CODE XREF: sub_110F+6A[]j
1177 CB 7E          bit    7, (hl)          ; FDC ready?
1179 28 FC          jr     z, loc_1177          ; no, loop
117B 32 0C 20          ld     (200Ch), a          ; write FDC data register

```

```

117E 1C          inc     e                ; next data byte
117F C2 76 11   jp      nz, loc_1176   ; loop until done
1182 14          inc     d
1183
1183          loc_1183:
1183 1A          ld      a, (de)        ; CODE XREF: sub_110F+7D□j
1184          ; get data byte
1184
1184          loc_1184:
1184 CB 7E          bit     7, (hl)        ; CODE XREF: sub_110F+77□j
1186 28 FC          jr      z, loc_1184   ; FDC ready?
1188 32 0C 20      ld      (200Ch), a     ; no, loop
118B 1C          inc     e                ; write FDC data register
118C C2 83 11     jp      nz, loc_1183   ; next data byte
118F 14          inc     d                ; loop until done
1190 ED 53 82 DE   ld      (0DE82h), de   ; save new address for disk read data
1194 C3 3F 0E       jp      read_FDC_result_ST0_2
1194          ; End of function sub_110F
1194
1197          ; ██████████ S U B R O U T I N E ██████████
1197
1197          find_file_at_$FD80:
1197 21 80 DF       ld      hl, 0DF80h     ; CODE XREF: get_file_cluster+8□p
1199 22 8C DE       ld      (0DE8Ch), hl  ; 19F2□p ...
119D C3 A6 11     jp      loc_11A6
11A0
11A0          find_file_at_$DEC3:
11A0 21 C3 DE       ld      hl, 0DEC3h    ; CODE XREF: sub_1389+B□p
11A3 22 8C DE       ld      (0DE8Ch), hl  ; 19E8□p
11A6          ; filename input buffer
11A6          loc_11A6:
11A6 AF          xor     a                ; CODE XREF: find_file_at_$FD80+6□j
11A7
11A7          loc_11A7:
11A7 32 A2 DE       ld      (0DEA2h), a    ; CODE XREF: find_file_at_$FD80+42□j
11AA CD 9B 0F     call   read_dirent_into_buffer
11AD 3A CE DE       ld      a, (0DECEh)    ; zero current dirent number
11B0 B7          or     a                ; 1st byte of directory entry
11B1 28 28        jr      z, loc_11DB    ; available entry?
11B3 FE E5        cp     0E5h ; '0'     ; yes, skip
11B5 28 1A        jr      z, next_dirent ; empty directory entry?
11B7 3A D9 DE       ld      a, (0DED9h)    ; yes, skip
11BA E6 1C        and    1Ch             ; file attribute
11BC 20 13        jr      nz, next_dirent ; directory, system file or volume label?
11BE ED 5B 8C DE   ld      de, (0DE8Ch)  ; yes, skip
11C2 21 CE DE       ld      hl, 0DECEh    ; filename
11C5 06 0B        ld      b, 11         ; directory entry
11C7          ; length of 8.3 filename
11C7          loc_11C7:
11C7 1A          ld      a, (de)        ; CODE XREF: find_file_at_$FD80+36□j
11C8 BE          cp     (hl)           ; same character?
11C9 20 06        jr      nz, next_dirent ; no, skip
11CB 13          inc     de
11CC 23          inc     hl
11CD 10 F8        djnz  loc_11C7        ; loop for length of filename
11CF 37          scf
11D0 C9          ret
11D1
11D1          next_dirent:
11D1 3A A2 DE       ld      a, (0DEA2h)    ; CODE XREF: find_file_at_$FD80+1E□j
11D4 3C          inc     a              ; find_file_at_$FD80+25□j ...
11D5 21 B0 DE       ld      hl, 0DEB0h    ; get current dirent number
11D8 BE          cp     (hl)           ; +1
11D9 20 CC        jr      nz, loc_11A7  ; maximum root directory entries
11DB          ; maximum number of entries?
11DB          loc_11DB:
11DB B7          or     a                ; CODE XREF: find_file_at_$FD80+1A□j
11DC C9          ret                    ; flag no entries left
11DC          ; End of function find_file_at_$FD80
11DC
11DD          ; ██████████ S U B R O U T I N E ██████████
11DD
11DD          sub_11DD:
11DD 3A E9 DE       ld      a, (0DEE9h)    ; CODE XREF: sub_11DD+2A□j
11E0 32 89 DE       ld      (0DE89h), a    ; 1A2A□p
11E3 FE 0F        cp     0Fh
11E5 20 08        jr      nz, loc_11EF   ; cluster number
11E7 3A E8 DE       ld      a, (0DEE8h)
11EA FE F7        cp     0F7h ; '7'
11EC 38 01        jr      c, loc_11EF
11EE C9          ret
11EF
11EF          loc_11EF:
11EF 3A E8 DE       ld      a, (0DEE8h)    ; CODE XREF: sub_11DD+8□j
11F2 32 88 DE       ld      (0DE88h), a    ; sub_11DD+F□j
11F5 CD B7 0F     call   sub_0FB7        ; cluster number
11F8 2A 80 DE       ld      hl, (0DE80h)   ; save cluster of current directory entry
11FB 22 E8 DE       ld      (0DEE8h), hl  ; cluster number
11FE 21 00 00     ld      hl, 0
1201 22 80 DE       ld      (0DE80h), hl
1204 CD 2A 10     call   sub_102A
1207 C3 DD 11     jp      sub_11DD
1207          ; End of function sub_11DD
1207
120A          ; ██████████ S U B R O U T I N E ██████████
120A
120A          sub_120A:
120A 2A 86 DE       ld      hl, (0DE86h)   ; CODE XREF: sub_120A+32□j
120D 22 88 DE       ld      (0DE88h), hl  ; sub_1389+6A□p
1210 2A 92 DE       ld      hl, (0DE92h)   ; save cluster of current directory entry
1213 2B          dec     hl
1214 22 92 DE       ld      (0DE92h), hl
1217 7C          ld      a, h
1218 B5          or     l
1219 20 09        jr      nz, loc_1224
121B 21 FF 0F     ld      hl, 0FFFh
121E 22 80 DE       ld      (0DE80h), hl
1221 C3 2A 10     jp      sub_102A
1224
1224          loc_1224:
1224 CD 3F 12     call   some_about_clusters ; CODE XREF: sub_120A+F□j
1227 2A 88 DE       ld      hl, (0DE88h)   ; get lister of current directory entry
122A 22 80 DE       ld      (0DE80h), hl
122D 2A 86 DE       ld      hl, (0DE86h)
1230 22 88 DE       ld      (0DE88h), hl  ; save cluster of current directory entry
1233 CD 2A 10     call   sub_102A
1236 2A 80 DE       ld      hl, (0DE80h)
1239 22 86 DE       ld      (0DE86h), hl
123C C3 0A 12     jp      sub_120A
123C          ; End of function sub_120A
123C
123F          ; ██████████ S U B R O U T I N E ██████████
123F
123F          some_about_clusters:
123F          ; CODE XREF: sub_120A+1A□p

```



```

1326 20 20 20 20+      .db 31h, 31h, 31h, 31h, 31h, 31h, 31h, 31h, 31h, 31h, 32h, 32h
1326 20 20 31 31+      .db 32h, 32h, 32h, 32h, 32h, 32h, 32h, 32h, 32h, 33h, 33h, 33h
1326 31 31 31 31+      .db 33h, 33h, 33h, 33h
134B 30 31 32 33+      .db 30h, 31h, 32h, 33h, 34h, 35h, 36h, 37h, 38h, 39h, 30h ; units
134B 34 35 36 37+      .db 31h, 32h, 33h, 34h, 35h, 36h, 37h, 38h, 39h, 30h, 31h
134B 38 39 30 31+      .db 32h, 33h, 34h, 35h, 36h, 37h, 38h, 39h, 30h, 31h, 32h
134B 32 33 34 35+      .db 33h, 34h, 35h, 36h
1370
1370 ; ██████████ S U B R O U T I N E ██████████
1370
1370 get_file_cluster: ; CODE XREF: sub_18E4+13□p
1370 CD 36 0D ; 194A□p
1370      call detect_disk
1373 21 B2 DE      ld hl, 0DEB2h ; media type
1376 CB D6      set 2, (hl)
1378 CD 97 11      call find_file_at_$FD80
137B 38 05      jr c, loc_1382 ; skip if found
137D 3E 16      ld a, 16h ; "FILE NOT FOUND"
137F C3 66 0B      jp print_msg_and_exit_option
1382
1382 loc_1382: ; CODE XREF: get_file_cluster+B□j
1382 2A E8 DE      ld hl, (0DEE8h) ; get cluster number
1385 22 88 DE      ld (0DEE8h), hl ; store
1388 C9      ret
1388 ; End of function get_file_cluster
1389
1389 ; ██████████ S U B R O U T I N E ██████████
1389
1389 sub_1389: ; CODE XREF: 1937□p
1389 CD 36 0D ; 1968□p
1389      call detect_disk
138C CD 2D 09      call check_disk_WP
138F 21 B2 DE      ld hl, 0DEB2h ; media type
1392 CB 96      res 2, (hl)
1394 CD A0 11      call find_file_at_$DEC3
1397 30 05      jr nc, loc_139E ; skip if not found
1399 3E 17      ld a, 17h ; "DUP FILE NAME"
139B C3 66 0B      jp print_msg_and_exit_option
139E
139E loc_139E: ; CODE XREF: sub_1389+E□j
139E CD 60 12      call check_disk_free_sectors
13A1 AF      xor a
13A2
13A2 loc_13A2: ; CODE XREF: sub_1389+31□j
13A2 32 A2 DE      ld (0DEA2h), a ; zero current dirent number
13A5 CD 9B 0F      call read_dirent_into_buffer
13A8 3A CE DE      ld a, (0DECEh) ; 1st byte of directory entry
13AB B7      or a ; available?
13AC 28 13      jr z, loc_13C1 ; yes, skip
13AE FE E5      cp 0E5h ; '0' ; deleted entry (free)?
13B0 28 0F      jr z, loc_13C1 ; yes, skip
13B2 3A A2 DE      ld a, (0DEA2h) ; get current dirent number
13B5 3C      inc a ; +1
13B6 21 B0 DE      ld hl, 0DEB0h ; maximum root directory entries
13B9 BE      cp (hl) ; any left?
13BA 38 E6      jr c, loc_13A2 ; yes, skip
13BC 3E 13      ld a, 13h ; "ERR"
13BE C3 66 0B      jp print_msg_and_exit_option
13C1
13C1 loc_13C1: ; CODE XREF: sub_1389+23□j
13C1 21 00 14 ; sub_1389+27□j
13C1      ld hl, 1400h ; template directory entry
13C4 11 CE DE      ld de, 0DECEh ; directory entry buffer
13C7 01 20 00      ld bc, 32 ; size of directory entry
13CA ED B0      ldir ; copy template
13CC 21 C3 DE      ld hl, 0DEC3h ; filename
13CF 11 CE DE      ld de, 0DECEh ; directory entry
13D2 01 0B 00      ld bc, 11 ; length of filename
13D5 ED B0      ldir ; copy to directory entry buffer
13D7 2A 96 DE      ld hl, (0DE96h) ; get number of sectors required
13DA CB 25      sla l
13DC CB 14      rl ; x2 = number of 256-byte blocks
13DE 22 EB DE      ld (0DEEBh), hl ; store file size
13E1 21 01 00      ld hl, 1
13E4 22 88 DE      ld (0DEE8h), hl ; save cluster of current directory entry
13E7 CD 3F 12      call some_about_clusters
13EA 2A 88 DE      ld hl, (0DEE8h) ; get cluster of current directory entry
13ED 22 86 DE      ld (0DEE6h), hl
13F0 22 E8 DE      ld (0DEE8h), hl ; store cluster number
13F3 CD 0A 12      call sub_120A
13F6 CD A7 0F      call write_dirent_from_buffer
13F9 2A E8 DE      ld hl, (0DEE8h) ; cluster number
13FC 22 88 DE      ld (0DEE8h), hl ; store
13FF C9      ret
13FF ; End of function sub_1389
13FF
13FF
1400 31 32 33 34+      .db 31h, 32h, 33h, 34h, 35h, 36h, 37h, 38h, 41h, 41h ; template directory entry
1400 35 36 37 38+      .db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 60h, 21h, 16h
1400 41 41 41 00+      .db 0, 0, 0, 0, 0, 0
1420
1420 ; ██████████ S U B R O U T I N E ██████████
1420
1420
1420 FDC_format_disc: ; CODE XREF: 1A4E□p
1420 21 00 DA      ld a, 0DA00h ; sector buffer & scratchpad
1423 AF      xor a
1424 06 00      ld b, 0 ; 256 bytes
1426
1426 loc_1426: ; CODE XREF: FDC_format_disc+8□j
1426 77      ld (hl), a
1427 23      inc hl
1428 10 FC      djnz loc_1426 ; zero 256 bytes of sector buffer & scratchpad
142A
142A loc_142A: ; CODE XREF: FDC_format_disc+C□j
142A 77      ld (hl), a
142B 23      inc hl
142C 10 FC      djnz loc_142A ; zero 2nd 256 bytes of sector buffer
142E 3E FF      ld a, 0FFh
1430 32 01 DA      ld (0DA01h), a
1433 32 02 DA      ld (0DA02h), a
1436 ED 4B A6 DE      ld bc, (0DEA6h) ; get selected disk format
143A 06 00      ld b, 0
143C DD 21 50 15      ld ix, 1550h ; format parameters table (4 entries)
1440 DD 09      add ix, bc
1442 DD 7E 00      ld a, (ix+0)
1445 32 0E 20      ld (200Eh), a ; FDC control register
1448 DD 7E 04      ld a, (ix+4)
144B 32 B2 DE      ld (0DEB2h), a ; store media type
144E DD 7E 08      ld a, (ix+8)
1451 32 A7 DE      ld (0DEA7h), a ; number of tracks
1454 DD 7E 0C      ld a, (ix+0Ch)
1457 32 A8 DE      ld (0DEA8h), a ; store SPT (format)
145A DD 7E 10      ld a, (ix+10h)
145D 32 00 DA      ld (0DA00h), a
1460 DD 7E 14      ld a, (ix+14h)
1463 32 A9 DE      ld (0DEA9h), a ; store in sector gap length (format)
1466 3A A7 DE      ld a, (0DEA7h) ; get number of tracks
1469 32 9A DE      ld (0DE9Ah), a ; store number of tracks
146C CD C9 0D      call FDC_init_and_SPECIFY
146F CD 15 0D      call FDC_RECALIBRATE_twice
1472 CD 6F 09      call check_disc_present
1475 CD 2D 09      call check_disk_WP
1478 AF      xor a ; start at track 0
1479

```



```

15C1
15C1 loc_15C1: call sub_110F ; CODE XREF: 15C8□j
15C1 CD 0F 11 ld hl, 0DE9Dh
15C4 21 9D DE dec (hl)
15C7 35 jr nz, loc_15C1
15C8 20 F7 jp bring_FDC_out_of_reset
15CA C3 EC 08
15CD
15CD
15CD loc_15CD: ; CODE XREF: sub_18E4+16□j
15CD 3E 06 ld a, 6 ; 193A□j
15CD ; SMD file type = 68k game
15CF CD BD 18 call sub_18BD
15D2 AF xor a
15D3 32 01 20 ld (2001h), a ; run BIOS in ROM
15D6 AF xor a
15D7 32 AA DE ld (0DEAAh), a
15DA 3A 00 DC ld a, (0DC00h) ; get SMD header size
15DD 32 AB DE ld (0DEABh), a
15E0 32 9A DE ld (0DE9Ah), a ; store number of tracks
15E3
15E3 loc_15E3: ; CODE XREF: 15EA□j
15E3 CD EF 15 call sub_15EF
15E6 21 AB DE ld hl, 0DEABh
15E9 35 dec (hl)
15EA 20 F7 jr nz, loc_15E3
15EC C3 EC 08 jp bring_FDC_out_of_reset
15EF
15EF ; ██████████ S U B R O U T I N E ██████████
15EF
15EF sub_15EF: ; CODE XREF: 15E3□p
15EF CD FA 0A call print_track_and_dec
15F2 2A 8A DE ld hl, (0DESAh)
15F5 22 82 DE ld (0DES2h), hl ; save address for disk read data
15F8 3A AA DE ld a, (0DEAAh) ; get rom/dram page
15FB 32 00 20 ld (2000h), a ; set rom/dram page
15FE 3C inc a ; +1 for next time
15FF 32 AA DE ld (0DEAAh), a ; store
1602 3E 20 ld a, 20h ; ' '
1604 32 9D DE ld (0DE9Dh), a
1607
1607 loc_1607: ; CODE XREF: sub_15EF+1F□j
1607 CD 0F 11 call sub_110F
160A 21 9D DE ld hl, 0DE9Dh
160D 35 dec (hl)
160E 20 F7 jr nz, loc_1607
1610 C9 ret
1610 ; End of function sub_15EF
1610
1611
1611 ; ██████████ S U B R O U T I N E ██████████
1611
1611 select_file: ; CODE XREF: sub_18E4+B□p
1611 AF ; 1942□p ...
1611 xor a ; main menu
1612 32 92 DF ld (0DF92h), a ; store flag
1615 21 00 DC ld hl, 0DC00h ; scratch buffer
1618 3E 20 ld a, 20h ; ' '
161A 06 00 ld b, 0
161C
161C loc_161C: ; CODE XREF: select_file+D□j
161C 77 ld (hl), a
161D 23 inc hl
161E 10 FC djnz loc_161C ; set 256 bytes to $20
1620
1620 loc_1620: ; CODE XREF: select_file+11□j
1620 77 ld (hl), a
1621 23 inc hl
1622 10 FC djnz loc_1620 ; set another 256 bytes to $20
1624 06 80 ld b, 80h ; 'Ç'
1626
1626 loc_1626: ; CODE XREF: select_file+17□j
1626 77 ld (hl), a
1627 23 inc hl
1628 10 FC djnz loc_1626 ; set another 128 bytes to $20
162A AF xor a
162B 32 8B DF ld (0DF8Bh), a
162E 32 91 DF ld (0DF91h), a
1631 DD 21 00 DC ld ix, 0DC00h ; scratch buffer
1635
1635 loc_1635: ; CODE XREF: select_file+58□j
1635 3A 91 DF ld a, (0DF91h)
1638 CB 27 sla a ; x2
163A C6 05 add a, 5 ; x2+5
163C DD 77 00 ld (ix+0), a
163F 3E 03 ld a, 3
1641 DD 77 01 ld (ix+1), a
1644 06 FF ld b, 0FFh
1646 3A 91 DF ld a, (0DF91h)
1649 3C inc a
164A 32 91 DF ld (0DF91h), a
164D FE 06 cp 6
164F 38 06 jr c, loc_1657
1651 06 00 ld b, 0
1653 AF xor a
1654 32 91 DF ld (0DF91h), a
1657
1657 loc_1657: ; CODE XREF: select_file+3E□j
1657 78 ld a, b
1658 DD 77 10 ld (ix+10h), a
165B 11 11 00 ld de, 11h
165E DD 19 add ix, de
1660 3A 8B DF ld a, (0DF8Bh)
1663 3C inc a
1664 32 8B DF ld (0DF8Bh), a
1667 FE 24 cp 24h ; '$'
1669 38 CA jr c, loc_1635
166B AF xor a
166C 32 A2 DE ld (0DEA2h), a ; zero current dirent number
166F 32 8B DF ld (0DF8Bh), a
1672 CD 36 0D call detect_disk
1675
1675 loc_1675: ; CODE XREF: select_file+B2□j
1675 CD 6E 17 call sub_176E
1678 30 4B jr nc, loc_16C5
167A 3A 8B DF ld a, (0DF8Bh)
167D 06 11 ld b, 11h
167F CD 93 17 call DC00_PLUS_B_x_A_STORE_DE90
1682 FD 2A 90 DE ld iy, (0DE90h)
1686 ED 4B 8B DF ld bc, (0DF8Bh)
168A 06 00 ld b, 0
168C DD 21 27 13 ld ix, 1327h ; tens
1690 DD 09 add ix, bc
1692 DD 7E 00 ld a, (ix+0)
1695 FD 77 02 ld (iy+2), a
1698 DD 21 4C 13 ld ix, 134Ch ; units
169C DD 09 add ix, bc
169E DD 7E 00 ld a, (ix+0)
16A1 FD 77 03 ld (iy+3), a
16A4 3E 2D ld a, 2Dh ; '-'
16A6 FD 77 04 ld (iy+4), a
16A9 3E 05 ld a, 5
16AB CD A0 17 call DE90_PLUS_A_STORE_DE90
16AE 21 CE DE ld hl, 0DECEh ; directory entry buffer
16B1 ED 5B 90 DE ld de, (0DE90h)
16B5 01 0B 00 ld bc, 11 ; length of 8.3 filename
16B8 ED B0 ldir
16BA 3A 8B DF ld a, (0DF8Bh)
16BD 3C inc a

```

```

16BE 32 8B DF      ld      (0DF8Bh), a
16C1 FE 24      cp      24h ; '$'
16C3 38 B0      jr      c, loc_1675
16C5
16C5 3E 1B      loc_16C5:      ; CODE XREF: select_file+67[]
                  ; "FILE LIST"
16C7 CD 09 0A      ld      a, 1Bh
16CA CD AF 12      call   print_msg
16CD CD EC 08      call   sub_12AF
16D0 3A 8B DF      call   bring_FDC_out_of_reset
16D3 B7          ld      a, (0DF8Bh)
16D4 20 09      jr      a
16D6 AF          jr      nz, loc_16DF
16D7 32 00 DC      xor     a
16DA 3E 15      ld      (0DC00h), a
16DC C3 66 0B      ld      a, 15h ; "NO FILE"
16DF          jp      print_msg_and_exit_option
16DF
16DF          ; -----
16DF 3A 92 DF      loc_16DF:      ; CODE XREF: select_file+C3[]
                  ; select_file+122[] ...
                  ; ?
16E2 06 66      ld      a, (0DF92h)
16E4 CD 93 17      ld      b, 66h ; 'f'
16E7 CD 06 0A      call   DC00_PLUS_B_x_A_STORE_DE90
16EA 3E 02      call   print_msg_at_HL
16EC 32 93 DF      ld      a, 2
16EF 3E 05      ld      (0DF93h), a
16F1 32 94 DF      ld      a, 5
16F4 3A 92 DF      ld      (0DF94h), a
16F7 06 06      ld      a, (0DF92h) ; main/utility menu
16F9 CD 93 17      ld      b, 6 ; multiplier
16FC 3A 8B DF      call   DC00_PLUS_B_x_A_STORE_DE90
16FF 21 90 DE      ld      a, (0DF8Bh)
1702 9E          ld      hl, 0DE90h
1703 FE 06      sbc     a, (hl)
1705 38 04      cp      6
1707 28 02      jr      c, loc_170B
1709 3E 06      jr      z, loc_170B
170B          ld      a, 6
170B          loc_170B:      ; CODE XREF: select_file+F4[]
                  ; select_file+F6[]
170B 32 90 DF      ld      (0DF90h), a
170E AF          xor     a
170F 32 91 DF      ld      (0DF91h), a
1712 CD 70 0C      call   get_menu_selection
1715 3A 95 DF      ld      a, (0DF95h) ; read controller input
1718 CB 67      bit     5, a ; button C?
171A 20 2F      jr      nz, loc_1743 ; yes, skip
171C CB 67      bit     4, a ; button B?
171E 20 1C      jr      nz, clear_SMD_size_and_exit_option ; yes, exit (abort?)
1720 3A 92 DF      ld      a, (0DF92h)
1723 3C          inc     a
1724 32 92 DF      ld      (0DF92h), a
1727 06 06      ld      b, 6 ; multiplier
1729 CD 93 17      call   DC00_PLUS_B_x_A_STORE_DE90
172C 3A 90 DE      ld      a, (0DE90h)
172F 21 8B DF      ld      hl, 0DF8Bh
1732 BE          cp      (hl)
1733 38 AA      jr      c, loc_16DF
1735 AF          xor     a
1736 32 92 DF      ld      (0DF92h), a
1739 C3 DF 16      jr      jp loc_16DF
173C          ; -----
173C          clear_SMD_size_and_exit_option: ; CODE XREF: select_file+10D[]
173C AF          xor     a
173D 32 00 DC      ld      (0DC00h), a
1740 C3 87 0B      jr      jp exit_option
1743          ; -----
1743          loc_1743:      ; CODE XREF: select_file+109[]
1743 3A 92 DF      ld      a, (0DF92h)
1746 06 06      ld      b, 6 ; multiplier
1748 CD 93 17      call   DC00_PLUS_B_x_A_STORE_DE90 ; HL = $DC00+($DF92)*6
174B 3A 91 DF      ld      a, (0DF91h) ; adder
174E CD A0 17      call   DE90_PLUS_A_STORE_DE90 ; HL=$DC00+($DF92)*6+($DF91)
1751 3A 90 DE      ld      a, (0DE90h) ; multiplier
1754 06 11      ld      b, 11 ; HL=$DC00+17*($DF92)*6+($DF91)
1756 CD 93 17      call   DC00_PLUS_B_x_A_STORE_DE90 ; adder
1759 3E 05      ld      a, 5 ; +5
175B CD A0 17      call   DE90_PLUS_A_STORE_DE90
175E 2A 90 DE      ld      hl, (0DE90h)
1761 11 80 DF      ld      de, 0DF80h
1764 01 0B 00      ld      bc, 0Bh
1767 ED B0      ldir
1769 AF          xor     a
176A 32 00 DC      ld      (0DC00h), a
176D C9          ret
176D          ; End of function select_file
176E
176E          ; ██████████ S U B R O U T I N E ██████████
176E
176E          sub_176E:      ; CODE XREF: select_file+64[]p
                  ; sub_176E+21[]
176E CD 9B 0F      call   read_dirent_into_buffer
1771 21 A2 DE      ld      hl, 0DEA2h ; current dirent number
1774 34          inc     (hl) ; +1
1775 3A CE DE      ld      a, (0DECEh) ; 1st byte of directory entry
1778 B7          or      a ; available?
1779 28 16      jr      z, loc_1791 ; yes, skip
177B FE E5      cp      0E5h ; '0' ; deleted (free)?
177D 28 09      jr      z, loc_1788 ; yes, skip
177F 3A D9 DE      ld      a, (0DED9h) ; file attribute
1782 E6 1C      and     1Ch ; directory, system file or volume label?
1784 20 02      jr      nz, loc_1788 ; yes, skip
1786 37          scf ; flag as?
1787 C9          ret
1788          ; -----
1788          loc_1788:      ; CODE XREF: sub_176E+F[]
                  ; sub_176E+16[]
1788          ld      a, (0DEA2h) ; get current dirent number
178B 21 B0 DE      ld      hl, 0DEB0h ; maximum root directory entries
178E BE          cp      (hl) ; any left?
178F 38 DD      jr      c, sub_176E ; yes, skip
1791          ; -----
1791          loc_1791:      ; CODE XREF: sub_176E+B[]
1791 B7          or      a ; flag no dirents left
1792 C9          ret
1792          ; End of function sub_176E
1793
1793          ; ██████████ S U B R O U T I N E ██████████
1793
1793          DC00_PLUS_B_x_A_STORE_DE90: ; CODE XREF: select_file+6E[]p
                  ; select_file+D3[]p ...
1793 21 00 DC      ld      hl, 0DC00h ; scratch buffer
1796 5F          ld      e, a
1797 16 00      ld      d, 0 ; DE=A
1799          ; -----
1799          loc_1799:      ; CODE XREF: DC00_PLUS_B_x_A_STORE_DE90+7[]
1799 19          add     hl, de
179A 10 FD      djnz   loc_1799 ; HL += Axh
179C 22 90 DE      ld      (0DE90h), hl ; store result
179F C9          ret
179F          ; End of function DC00_PLUS_B_x_A_STORE_DE90
17A0

```



```

17A0 ; ██████████ S U B R O U T I N E ██████████
17A0
17A0 DE90_PLUS_A_STORE_DE90: ; CODE XREF: select_file+9A□p
17A0 2A 90 DE ; select_file+13D□p ...
17A0 ld hl, (0DE90h)
17A3 5F ld e, a
17A4 16 00 ld d, 0
17A6 19 add hl, de ; HL += A
17A7 22 90 DE ld (0DE90h), hl ; store result
17AA C9 ret
17AA ; End of function DE90_PLUS_A_STORE_DE90
17AB
17AB ; ██████████ S U B R O U T I N E ██████████
17AB
17AB input_filename: ; CODE XREF: 192F□p
17AB 3E 1A ; 1955□p ...
17AB ; "NEW NAME"
17AB CD 09 0A ld a, 1Ah
17AB call print_msg
17B0 21 C3 DE ld hl, 0DEC3h ; name buffer
17B3 3E 20 ld a, 20h ; ' '
17B5 06 0B ld b, 11 ; space
17B7 ; length 8.3 filename
17B7 loc_17B7: ; CODE XREF: input_filename+E□j
17B7 77 ld (hl), a
17B8 23 inc hl
17B9 10 FC djnz loc_17B7 ; clear filename buffer
17BB AF xor a
17BC 32 91 DF ld a, (0DF91h), a ; number of chars in buffer = 0
17BF 3E 09 ld a, 9
17C1 32 93 DF ld a, (0DF93h), a
17C4 3E 0A ld a, 0Ah
17C6 32 94 DF ld a, (0DF94h), a
17C9
17C9 input_name_loop: ; CODE XREF: input_filename+61□j
17C9 06 1E ld b, 1Eh ; input_filename+66□j ...
17CB CD 75 18 call sub_1875
17CE
17CE loc_17CE: ; CODE XREF: input_filename+36□j
17CE CD 99 0B call handle_pc_cmd_read_controller
17D1 3A 95 DF ld a, (0DF95h) ; read controller input
17D4 CB 47 bit 0, a ; up?
17D6 28 08 jr z, loc_17E0 ; no, skip
17D8 CB 67 bit 4, a ; button B?
17DA 20 4A jr nz, abort_string_entry ; yes, skip (abort)
17DC CB 6F bit 5, a ; button C?
17DE 20 49 jr nz, accept_entry ; yes, skip (accept)
17E0
17E0 loc_17E0: ; CODE XREF: input_filename+2B□j
17E0 B7 or a ; anything pressed?
17E1 20 EB jr nz, loc_17CE ; yes, loop
17E3
17E3 loc_17E3: ; CODE XREF: input_filename+3F□j
17E3 CD 99 0B call handle_pc_cmd_read_controller
17E6 3A 95 DF ld a, (0DF95h) ; read controller input
17E9 B7 or a ; anything pressed?
17EA 28 F7 jr z, loc_17E3 ; no, loop waiting for input
17EC CB 67 bit 4, a ; button B?
17EE 20 24 jr nz, delete_char ; yes, skip (DELETE)
17F0 CB 6F bit 5, a ; button C?
17F2 20 1A jr nz, insert_char ; yes, skip (INSERT)
17F4 06 20 ld b, 20h ; ' '
17F6 CD 75 18 call sub_1875
17F9 3A 95 DF ld a, (0DF95h) ; read controller input
17FC CB 5F bit 3, a ; right?
17FE 20 1A jr nz, next_char ; yes, skip
1800 CB 57 bit 2, a ; left?
1802 20 1C jr nz, previous_char ; yes, skip
1804 3A 94 DF ld a, (0DF94h)
1807 EE 02 xor 2
1809 32 94 DF ld (0DF94h), a
180C 18 BB jr input_name_loop ; loop in filename input routine
180E
180E ;
180E insert_char: ; CODE XREF: input_filename+47□j
180E CD 30 18 call add_char_to_buf
1811 C3 C9 17 jp input_name_loop
1814
1814 ;
1814 delete_char: ; CODE XREF: input_filename+43□j
1814 CD 49 18 call delete_char_from_buf
1817 C3 C9 17 jp input_name_loop
181A
181A ;
181A next_char: ; CODE XREF: input_filename+53□j
181A 21 93 DF ld hl, 0DF93h
181D 34 inc (hl)
181E 20 A9 jr nz, input_name_loop
1820
1820 previous_char: ; CODE XREF: input_filename+57□j
1820 21 93 DF ld hl, 0DF93h
1823 35 dec (hl)
1824 20 A3 jr nz, input_name_loop
1826
1826 abort_string_entry: ; CODE XREF: input_filename+2F□j
1826 C3 87 0B jp exit_option
1829
1829 ;
1829 accept_entry: ; CODE XREF: input_filename+33□j
1829 3A 91 DF ld a, (0DF91h) ; get number of chars in buffer
182C B7 or a ; zero?
182D 28 9A jr z, input_name_loop ; yes, don't exit
182F C9 ret
182F ; End of function input_filename
1830
1830 ; ██████████ S U B R O U T I N E ██████████
1830
1830 add_char_to_buf: ; CODE XREF: input_filename+63□p
1830 3A 91 DF ld a, (0DF91h) ; get number of chars entered
1833 FE 0B cp 11 ; max?
1835 C8 ret z ; yes, return
1836 3A 94 DF ld a, (0DF94h)
1839 3D dec a
183A 4F ld c, a
183B 3A 93 DF ld a, (0DF93h)
183E CD 7B 0A call sub_0A7B
1841 CD 59 18 call save_char_to_buf_and_print
1844 21 91 DF ld hl, 0DF91h ; number of chars entered
1847 34 inc (hl) ; +1
1848 C9 ret
1848 ; End of function add_char_to_buf
1848
1848 ;
1848 ; ██████████ S U B R O U T I N E ██████████
1848
1848 delete_char_from_buf: ; CODE XREF: input_filename+69□p
1848 3A 91 DF ld a, (0DF91h) ; get number of chars entered
184C 3D dec a ; -1
184D FA 53 18 jp m, loc_1853 ; negative? yes, skip
1850 32 91 DF ld (0DF91h), a ; save number of chars entered
1853
1853 loc_1853: ; CODE XREF: delete_char_from_buf+4□j
1853 3E 20 ld a, 20h ; ' '
1855 CD 59 18 call save_char_to_buf_and_print
1858 C9 ret

```

```

1858 ; End of function delete_char_from_buf
1858
1859
1859 ; ██████████ S U B R O U T I N E ██████████
1859
1859 save_char_to_buf_and_print: ; CODE XREF: add_char_to_buf+110p
1859 ED 4B 91 DF ; delete_char_from_buf+C0p
1859 ld bc, (0DF91h)
185D 06 00 ld b, 0 ; offset of char in buffer
185F 21 C3 DE ld hl, 0DEC3h ; filename buffer address
1862 09 add hl, bc ; get char address
1863 77 ld (hl), a ; save new char
1864 FE 20 cp 20h ; ' '
1866 20 02 jr nz, loc_186A ; space?
1868 3E 5F ld a, 5Fh ; '_'
186A
186A loc_186A: ; CODE XREF: save_char_to_buf_and_print+D0j
186A 47 ld b, a ; save char
186B 3A 91 DF ld a, (0DF91h) ; filename buffer address
186E C6 10 add a, 10h ; offset to char in buffer
1870 0E 05 ld c, 5 ; calc column
1872 C3 67 0A jp print_char ; row
1872 ; End of function save_char_to_buf_and_print
1872
1875
1875 ; ██████████ S U B R O U T I N E ██████████
1875
1875 sub_1875: ; CODE XREF: input_filename+200p
1875 3A 93 DF ; input_filename+4B0p
1875 ld a, (0DF93h)
1878 FE 17 cp 17h
187A 20 02 jr nz, loc_187E
187C 3E 05 ld a, 5
187E
187E loc_187E: ; CODE XREF: sub_1875+50j
187E FE 04 cp 4
1880 20 02 jr nz, loc_1884
1882 3E 16 ld a, 16h
1884
1884 loc_1884: ; CODE XREF: sub_1875+B0j
1884 32 93 DF ld (0DF93h), a
1887 3A 94 DF ld a, (0DF94h)
188A 4F ld c, a
188B 3A 93 DF ld a, (0DF93h)
188E C3 67 0A jp print_char
188E ; End of function sub_1875
188E
1891
1891 ; ██████████ S U B R O U T I N E ██████████
1891
1891 check_cartridge_present: ; CODE XREF: 19A00p
1891 3A F1 DF ; 19AE0p
1891 ld a, (0DFF1h) ; cartridge size (16kB pages)
1894 B7 or a
1895 C0 ret nz ; return if non-zero
1896 3E 09 ld a, 9 ; "NO IC CARD"
1898 C3 66 0B jp print_msg_and_exit_option
1898 ; End of function check_cartridge_present
1898
1899
1899 ; ██████████ S U B R O U T I N E ██████████
1899
1899 init_SMD_header: ; CODE XREF: 195F0p
1899 32 A6 DE ; 19B80p
1899 ld (0DEA6h), a
189E 21 00 DC ld hl, 0DC00h ; SMD header
18A1 AF xor a
18A2 06 00 ld b, 0
18A4
18A4 loc_18A4: ; CODE XREF: init_SMD_header+B0j
18A4 77 ld (hl), a
18A5 23 inc hl
18A6 10 FC djnz loc_18A4 ; zero 256 bytes of SMD header
18A8
18A8 loc_18A8: ; CODE XREF: init_SMD_header+F0j
18A8 77 ld (hl), a
18A9 23 inc hl
18AA 10 FC djnz loc_18A8 ; clear 2nd 256 bytes
18AC 3E AA ld a, 0AAh ; '-'
18AE 32 08 DC ld (0DC08h), a ; identifier byte #1
18B1 3E BE ld a, 0BBh ; 'q'
18B3 32 09 DC ld (0DC09h), a ; identifier byte #2
18B6 3A A6 DE ld a, (0DEA6h)
18B9 32 0A DC ld (0DC0Ah), a ; type (68k game/SRAM)
18BC C9 ret
18BC ; End of function init_SMD_header
18BC
18BD
18BD ; ██████████ S U B R O U T I N E ██████████
18BD
18BD sub_18BD: ; CODE XREF: 15AE0p
18BD 32 A6 DE ; 15CF0p
18BD ld (0DEA6h), a ; save file type
18C0 21 00 DC ld hl, 0DC00h ; SMD header buffer
18C3 22 82 DE ld (0DE82h), hl ; save address for disk read data
18C6 CD 0F 11 call sub_110F
18C9 3E AA ld a, 0AAh ; '-'
18CB 21 08 DC ld hl, 0DC08h ; identifier byte #1
18CE BE cp (hl) ; valid?
18CF C0 ret nz ; no, return
18D0 3E BB ld a, 0BBh ; 'q'
18D2 21 09 DC ld hl, 0DC09h ; identifier byte #2
18D5 BE cp (hl) ; valid?
18D6 C0 ret nz ; no, return
18D7 3A A6 DE ld a, (0DEA6h)
18DA 21 0A DC ld hl, 0DC0Ah ; file type
18DD BE cp (hl) ; same?
18DE C8 ret z ; yes, return
18DF 3E 19 ld a, 19h ; "FILE ERR"
18E1 C3 66 0B jp print_msg_and_exit_option
18E1 ; End of function sub_18BD
18E1
18E4
18E4 ; ██████████ S U B R O U T I N E ██████████
18E4
18E4 sub_18E4: ; CODE XREF: 19730p
18E4 21 00 80 ld hl, 8000h ; copier DRAM
18E7 22 8A DE ld (0DE8Ah), hl ; store
18EA 3E FF ld a, 0FFh
18EC 32 EF DE ld (0DEEFh), a
18EF CD 11 16 call select_file
18F2 3E 02 ld a, 2 ; "LOADING"
18F4 CD 09 0A call print_msg
18F7 CD 70 13 call get_file_cluster
18FA C3 CD 15 jp loc_15CD
18FA ; End of function sub_18E4
18FA
18FD
18FD ; -----
18FD 21 00 80 ld hl, 8000h
1900 22 8A DE ld (0DE8Ah), hl
1903
1903 dump_cartridge: ; CODE XREF: 19CC0j
1903 3A 00 DC ld a, (0DC00h) ; get cartridge size (16kB pages)
1906 B7 or a ; zero?
1907 28 04 jr z, loc_190D ; yes, exit

```

```

1909 FE 41          cp      65          ; <= 1MB?
190B 38 05          jr      c, loc_1912      ; yes, skip
190D
190D
loc_190D:
190D 3E 10          ld      a, 10h          ; CODE XREF: 1907□□j
190F C3 66 0B      jp      print_msg_and_exit_option ; "NO GAME IN RAM"
;
1912
1912
loc_1912:
1912 2A 00 DC          ld      hl, (0DC00h)     ; CODE XREF: 190B□□j
1915 26 00          ld      h, 0            ; get SMD header size
1917 CB 25          sla     1
1919 CB 14          rl      h
191B CB 25          sla     1
191D CB 14          rl      h
191F CB 25          sla     1
1921 CB 14          rl      h
1923 CB 25          sla     1
1925 CB 14          rl      h
1927 CB 25          sla     1
1929 CB 14          rl      h
192B 2C            inc     l                ; x32
192C 22 96 DE          ld      (0DE96h), hl    ; x32+1
192F CD AB 17          call   input_filename   ; save number of sectors required
1932 3E 03          ld      a, 3            ; "SAVING"
1934 CD 09 0A          call   print_msg
1937 CD 89 13          call   sub_1389
193A C3 CD 15          jp      loc_15CD
;
193D
193D
load_data:
193D 3E 1E          ld      a, 1Eh          ; "LOAD DATA"
193F CD 09 0A          call   print_msg
1942 CD 11 16          call   select_file
1945 3E 02          ld      a, 2            ; "LOADING"
1947 CD 09 0A          call   print_msg
194A CD 70 13          call   get_file_cluster
194D C3 AC 15          jp      loc_15AC
;
1950
1950
save_data:
1950 3E 1F          ld      a, 1Fh          ; "SAVE DATA"
1952 CD 09 0A          call   print_msg
1955 CD AB 17          call   input_filename
1958 3E 03          ld      a, 3            ; "SAVING"
195A CD 09 0A          call   print_msg
195D 3E 07          ld      a, 7            ; file type = SRAM file
195F CD 9B 18          call   init_SMD_header
1962 21 41 00          ld      hl, 65          ; 65 sectors = header + 32KB
1965 22 96 DE          ld      (0DE96h), hl    ; save number of sectors required
1968 CD 89 13          call   sub_1389
196B C3 AC 15          jp      loc_15AC
;
196E
196E
run_file:
196E 3E 1C          ld      a, 1Ch          ; "RUN FILE"
1970 CD 09 0A          call   print_msg
1973 CD E4 18          call   sub_18E4
1976 3A 00 DC          ld      a, (0DC00h)
1979 B7            or      a
197A C8            ret
197B CD CA 09          call   init_VDP_display_disabled
197E 3A 01 DC          ld      a, (0DC01h)
1981 FE 01          cp      1
1983 28 0C          jr      z, run_z80_in_dram
1985 FE 02          cp      2
1987 28 10          jr      z, run_bios_in_rom
1989 3E 03          ld      a, 3
198B 32 01 20          ld      (2001h), a      ; run 68k code in DRAM
198E C3 00 00          jp      loc_0000        ; reset
;
1991
1991
run_z80_in_dram:
1991 3E 01          ld      a, 1            ; CODE XREF: 1983□□j
1993 32 01 20          ld      (2001h), a      ; run Z80 code in DRAM
1996 C3 00 00          jp      loc_0000        ; reset
;
1999
1999
run_bios_in_rom:
1999 AF          xor     a                ; CODE XREF: 1987□□j
199A 32 01 20          ld      (2001h), a      ; run BIOS in ROM
199D C3 00 80          jp      8000h           ; the bug Charles McDonald mentioned?
;
19A0
19A0
run_ic_card:
19A0 CD 91 18          call   check_cartridge_present
19A3 CD CA 09          call   init_VDP_display_disabled
19A6 3E 02          ld      a, 2
19A8 32 01 20          ld      (2001h), a      ; run 68k code in cartridge rom
19AB
loc_19AB:
19AB C3 AB 19          jp      loc_19AB        ; CODE XREF: 19AB□□j
;
19AE
19AE
save_ic_card:
19AE CD 91 18          call   check_cartridge_present
19B1 3E 1D          ld      a, 1Dh          ; "SAVE IC CARD"
19B3 CD 09 0A          call   print_msg
19B6 3E 06          ld      a, 6            ; file type = 68k game
19B8 CD 9B 18          call   init_SMD_header
19BB 3A F1 DF          ld      a, (0DF1h)      ; cartridge size (16kB pages)
19BE 32 00 DC          ld      (0DC00h), a     ; SMD hdr - size of file
19C1 3E 03          ld      a, 3
19C3 32 01 DC          ld      (0DC01h), a     ; SMD hdr (file data type) = 68000 program
19C6 21 00 40          ld      hl, 4000h
19C9 22 8A DE          ld      (0DE8Ah), hl
19CC C3 03 19          jp      dump_cartridge
;
19CF
19CF
rename_file:
19CF 3E 22          ld      a, 34           ; "RENAME FILE"
19D1 CD 09 0A          call   print_msg
19D4 CD 11 16          call   select_file
19D7 CD D5 09          call   VDP_cls
19DA CD AB 17          call   input_filename
19DD 3E 25          ld      a, 37           ; "RENAMING"
19DF CD 09 0A          call   print_msg
19E2 CD 36 0D          call   detect_disk
19E5 CD 2D 09          call   check_disk_WP
19E8 CD A0 11          call   find_file_at_$DEC3
19EB 30 05          jr      nc, loc_19F2
19ED 3E 17          ld      a, 17h
19EF C3 66 0B          jp      print_msg_and_exit_option
;
19F2
19F2
loc_19F2:
19F2 CD 97 11          call   find_file_at_$FD80 ; CODE XREF: 19EB□□j
19F5 38 05          jr      c, loc_19FC
19F7 3E 16          ld      a, 16h
19F9 C3 66 0B          jp      print_msg_and_exit_option
;
19FC
19FC
loc_19FC:
19FC 21 C3 DE          ld      hl, 0DEC3h      ; CODE XREF: 19F5□□j
19FF 11 CE DE          ld      de, 0DECCh      ; filename buffer
1A02 01 0B 00          ld      bc, 11          ; directory entry buffer
1A05 ED B0          ldir                    ; length of 8.3 filename
1A07 CD A7 0F          call   write_dirent_from_buffer
1A0A C3 EC 08          jp      bring_FDC_out_of_reset ; copy filename to directory entry buffer
1A0D
1A0D

```

```

IA0D      delete_file:                                ; "DELETE FILE"
IA0D 3E 23      ld      a, 23h ; '#'
IA0F CD 09 0A      call    print_msg
IA12 CD 11 16      call    select_file
IA15 3E 26      ld      a, 26h ; 'g'
IA17 CD 09 0A      call    print_msg
IA1A CD 36 0D      call    detect_disk
IA1D CD 2D 09      call    check_disk_WP
IA20 CD 97 11      call    find_file_at_$FD80
IA23 38 05      jr      c, loc_1A2A ; skip if found
IA25 3E 16      ld      a, 16h ; "FILE NOT FOUND"
IA27 C3 66 0B      jp      print_msg_and_exit_option
IA2A
IA2A
IA2A      loc_1A2A:                                ; CODE XREF: 1A23□j
IA2A CD DD 11      call    sub_11DD
IA2D 3E E5      ld      a, 0E5h ; 'O' ; mark file as deleted
IA2F 32 CE DE      ld      (0DECEh), a ; 1st byte of directory entry
IA32 CD A7 0F      call    write_dirent_from_buffer
IA35 C3 EC 08      jp      bring_FDC_out_of_reset
IA38
IA38      format_disk:                                ; "FORMAT DISK"
IA38 3E 24      ld      a, 36
IA3A CD C9 0C      call    format_confirm_or_abort
IA3D 3E 27      ld      a, 39 ; "FORMATTING"
IA3F CD 09 0A      call    print_msg
IA42 3A 91 DF      ld      a, (0DF91h) ; get current menu option
IA45 E6 01      and     1
IA47 EE 01      xor     1
IA49 F6 02      or      2 ; format = 2/3 (720/1.44)
IA4B 32 A6 DE      ld      (0DEA6h), a ; store as disk format
IA4E CD 20 14      call    FDC_format_disc
IA51 C3 EC 08      jp      bring_FDC_out_of_reset
IA54
IA54
IA54      locret_1A54:                                ; CODE XREF: handle_pc_command+5□j
IA54 C9      ret
IA55
IA55      ; ██████████ S U B R O U T I N E ██████████
IA55
IA55      handle_pc_command:                            ; CODE XREF: 0012□j
IA55 CD 40 1B      call    wait_if_busy_read_pc_data ; handle_pc_cmd_read_controller+7□p
IA58
IA58      loc_1A58:                                ; CODE XREF: handle_pc_command+C□j
IA58 FE D5      cp      0D5h ; '1' ; command identifier byte #1
IA5A 20 F8      jr      nz, locret_1A54 ; no, return
IA5C CD 40 1B      call    wait_if_busy_read_pc_data
IA5F FE AA      cp      0AAh ; '-' ; command identifier byte #2?
IA61 20 F5      jr      nz, loc_1A58 ; no, return
IA63 CD 40 1B      call    wait_if_busy_read_pc_data
IA66 FE 96      cp      96h ; '0' ; command identifier byte #3?
IA68 28 03      jr      z, valid_pc_command_header ; yes, continue
IA6A C3 2B 1B      jp      unknown_pc_command
IA6D
IA6D
IA6D      valid_pc_command_header:                    ; CODE XREF: handle_pc_command+13□j
IA6D 16 81      ld      d, 81h ; 'u'
IA6F 21 00 DF      ld      hl, 0DF00h ; command buffer
IA72 06 05      ld      b, 5 ; 5 command bytes
IA74
IA74      get_pc_cmd_packet:                            ; CODE XREF: handle_pc_command+26□j
IA74 CD 40 1B      call    wait_if_busy_read_pc_data
IA77 77      ld      (hl), a ; save byte
IA78 2C      inc     d ; next buffer address
IA79 AA      xor     d ; calc checksum
IA7A 57      ld      d, a ; store checksum
IA7B 10 F7      djnz   get_pc_cmd_packet ; loop for 5 bytes
IA7D CD 3B 1B      call    get_checksum_byte ; checksum ok?
IA80 28 03      jr      z, loc_1A85 ; yes, continue
IA82 C3 2B 1B      jp      unknown_pc_command
IA85
IA85
IA85      loc_1A85:                                ; CODE XREF: handle_pc_command+2B□j
IA85 2A 01 DF      ld      hl, (0DF01h) ; hl=address
IA88 ED 4B 03 DF      ld      bc, (0DF03h) ; bc=length
IA8C 3A 00 DF      ld      a, (0DF00h) ; command parameter
IA8F FE 00      cp      0 ; receive data from pc?
IA91 20 13      jr      nz, loc_1AA6 ; no, skip
IA93 16 81      ld      d, 81h ; 'u' ; init checksum
IA95
IA95      loc_1A95:                                ; CODE XREF: handle_pc_command+48□j
IA95 CD 40 1B      call    wait_if_busy_read_pc_data
IA98 77      ld      (hl), a ; store data at specified address
IA99 AA      xor     d ; calc checksum
IA9A 57      ld      d, a ; store checksum
IA9B ED A1      cpi    d, a ; next address
IA9D EA 95 1A      jp     pe, loc_1A95 ; loop until done
IAA0 CD 3B 1B      call    get_checksum_byte
IAA3 C3 2F 1B      jp     test_checksum
IAA6
IAA6
IAA6      loc_1AA6:                                ; CODE XREF: handle_pc_command+3C□j
IAA6 FE 01      cp      1 ; send data to pc?
IAA8 20 1D      jr      nz, loc_1AC7 ; no, skip
IAAA 16 81      ld      d, 81h ; 'u' ; init checksum
IAAC
IAAC      loc_1AAC:                                ; CODE XREF: handle_pc_command+61□j
IAAC 7E      ld      a, (hl) ; get byte at specified address
IAAD 5F      ld      e, a ; save byte
IAAE AA      xor     d ; calc checksum
IAAF 57      ld      d, a ; store checksum
IAB0 7B      ld      a, e ; restore data byte
IAB1 CD 4B 1B      call    write_byte_to_pc
IAB4 ED A1      cpi    d, a ; next address
IAB6 EA AC 1A      jp     pe, loc_1AAC ; loop until done
IAB9 7A      ld      a, d ; get checksum
IABA CD 4B 1B      call    write_byte_to_pc
IABD
IABD      loc_1ABD:                                ; CODE XREF: handle_pc_command+6D□j
IABD 3A 03 20      ld      a, (2003h) ; read PC busy status
IAC0 CB 7F      bit    7, a ; busy?
IAC2 20 F9      jr      nz, loc_1ABD ; yes, loop
IAC4 C3 35 1B      jp     flag_pc_command_ok
IAC7
IAC7
IAC7      loc_1AC7:                                ; CODE XREF: handle_pc_command+53□j
IAC7 FE 02      cp      2 ; upload VRAM command?
IAC9 20 20      jr      nz, loc_1AEB ; no, skip
IACB DB BF      in     a, (0BFh) ; read VDP control port
IACD 3A 01 DF      ld      a, (0DF01h) ; get 1st VDP command word byte
IAD0 D3 BF      out    (0BFh), a ; write VDP control port
IAD2 3A 02 DF      ld      a, (0DF02h) ; get 2nd VDP command word byte
IAD5 D3 BF      out    (0BFh), a ; write VDP control port
IAD7 16 81      ld      d, 81h ; 'u' ; init checksum
IAD9
IAD9      loc_1AD9:                                ; CODE XREF: handle_pc_command+8D□j
IAD9 CD 40 1B      call    wait_if_busy_read_pc_data
IADC D3 BE      out    (0BEh), a ; write VDP data port
IADE AA      xor     d ; calc checksum
IADE AA      ld      d, a ; store checksum
IADF 57      cpi    d, a ; next address
IAE0 ED A1      jp     pe, loc_1AD9 ; loop until done
IAE2 EA D9 1A      call    get_checksum_byte
IAE5 CD 3B 1B      call    test_checksum
IAE8 C3 2F 1B      jp     test_checksum
IAEB
IAEB
IAEB      loc_1AEB:                                ; CODE XREF: handle_pc_command+74□j
IAEB FE 03      cp      3 ; download VRAM command?

```

```

IAED 20 2A      jr      nz, loc_1B19      ; no, skip
IAEF DB BF      in      a, (0BFh)        ; read VDP control port
IAF1 3A 01 DF    ld      a, (0DF01h)       ; get 1st byte of VDP command word
IAF4 D3 BF      out     a, (0BFh)         ; write VDP control port
IAF6 3A 02 DF    ld      a, (0DF02h)       ; get 2nd byte of VDP command word
IAF9 D3 BF      out     a, (0BFh), a     ; write VDP control port
IAFB 16 81      ld      d, 81h; 'U'      ; init checksum
IAFD
IAFD DB BE      loc_1AFD:                ; CODE XREF: handle_pc_command+B3[]j
IAFD DB BE      in      a, (0BEh)        ; read VDP data port
IAFF 5F         ld      e, a
IB00 AA         xor     d, a              ; calc checksum
IB01 57         ld      d, a
IB02 7B         ld      a, e
IB03 CD 4B 1B   call   write_byte_to_pc
IB06 ED A1      cpi    a, 0              ; next address
IB08 EA PD 1A   jp     pe, loc_1AFD      ; loop until done
IB0B 7A         ld      a, d
IB0C CD 4B 1B   call   write_byte_to_pc
IB0F
IB0F 3A 03 20   loc_1B0F:                ; CODE XREF: handle_pc_command+BF[]j
IB0F 3A 03 20   ld      a, (2003h)       ; read PC busy flag
IB12 CB 7F      bit    7, a              ; busy?
IB14 20 F9      jr     nz, loc_1B0F      ; yes, loop
IB16 C3 35 1B   jp     flag_pc_command_ok
IB19
IB19
IB19 FE 04      loc_1B19:                ; CODE XREF: handle_pc_command+98[]j
IB19 FE 04      cp     4                  ; jump to address?
IB1B 20 01      jr     nz, loc_1B1E      ; no, skip
IB1D E9         jp     (hl)              ; jump to specified address
IB1E
IB1E
IB1E FE 05      loc_1B1E:                ; CODE XREF: handle_pc_command+C6[]j
IB1E FE 05      cp     5                  ; set rom/dram page?
IB20 20 09      jr     nz, unknown_pc_command ; no, skip
IB22 3A 01 DF    ld      a, (0DF01h)       ; get rom/dram page
IB25 32 00 20   ld      (2000h), a       ; set rom/dram page
IB28 C3 35 1B   jp     flag_pc_command_ok
IB2B
IB2B
IB2B 3E 01      unknown_pc_command:      ; CODE XREF: handle_pc_command+15[]j
IB2B 3E 01      ld      a, 1              ; handle_pc_command+2D[]j ...
IB2D 18 08      jr     loc_1B37          ; flag error
IB2F
IB2F
IB2F 28 04      test_checksum:           ; CODE XREF: handle_pc_command+4E[]j
IB2F 28 04      jr     z, flag_pc_command_ok ; handle_pc_command+93[]j
IB2F 28 04      ; jump if checksum OK
IB31
IB31 3E 02      loc_1B31:                ; CODE XREF: handle_pc_command+DE[]j
IB31 3E 02      ld      a, 2              ; loop forever!
IB33 18 FC      jr     loc_1B31
IB35
IB35
IB35 3E 00      flag_pc_command_ok:      ; CODE XREF: handle_pc_command+6F[]j
IB35 3E 00      ld      a, 0              ; handle_pc_command+C1[]j ...
IB35 3E 00      ; flag OK
IB37
IB37 32 05 DF    loc_1B37:                ; CODE XREF: handle_pc_command+D8[]j
IB37 32 05 DF    ld      (0DF05h), a       ; store success/failure flag
IB3A C9         ret
IB3A C9         ; End of function handle_pc_command
IB3A
IB3A
IB3B
IB3B
IB3B
IB3B
IB3B
IB3B
IB3B
IB3B
IB3B CD 40 1B   get_checksum_byte:       ; CODE XREF: handle_pc_command+28[]p
IB3B CD 40 1B   ; handle_pc_command+4B[]p ...
IB3B CD 40 1B   call   wait_if_busy_read_pc_data
IB3E AA         xor     d
IB3F C9         ret
IB3F C9         ; End of function get_checksum_byte
IB3F
IB40
IB40
IB40
IB40
IB40 3A 03 20   wait_if_busy_read_pc_data: ; CODE XREF: handle_pc_command[]p
IB40 3A 03 20   ; handle_pc_command+7[]p ...
IB40 3A 03 20   ld      a, (2003h)       ; read PC busy status
IB43 CB 7F      bit    7, a              ; busy?
IB45 28 F9      jr     z, wait_if_busy_read_pc_data ; yes, loop
IB47 3A 02 20   ld      a, (2002h)       ; read PC data
IB4A C9         ret
IB4A C9         ; End of function wait_if_busy_read_pc_data
IB4A
IB4A
IB4B
IB4B
IB4B
IB4B
IB4B
IB4B
IB4B
IB4B
IB4B F5         write_byte_to_pc:        ; CODE XREF: handle_pc_command+5C[]p
IB4B F5         ; handle_pc_command+65[]p ...
IB4B F5         push   af                ; store data byte
IB4C
IB4C
IB4C 3A 03 20   loc_1B4C:                ; CODE XREF: write_byte_to_pc+6[]j
IB4C 3A 03 20   ld      a, (2003h)       ; read PC busy status
IB4F CB 7F      bit    7, a              ; busy?
IB51 20 F9      jr     nz, loc_1B4C      ; yes, loop
IB53 F1         pop     af                ; restore data byte
IB54 32 02 20   ld      (2002h), a       ; write data (lo nibble) to PC
IB57 1F         rra
IB58 1F         rra
IB59 1F         rra
IB5A 1F         rra
IB5B F5         push   af                ; store data
IB5C 3A 02 20   ld      a, (2002h)       ; read PC data
IB5F
IB5F 3A 03 20   loc_1B5F:                ; CODE XREF: write_byte_to_pc+19[]j
IB5F 3A 03 20   ld      a, (2003h)       ; read PC busy status
IB62 CB 7F      bit    7, a              ; busy?
IB64 20 F9      jr     nz, loc_1B5F      ; yes, loop
IB66 F1         pop     af                ; restore data
IB67 32 02 20   ld      (2002h), a       ; write data (hi nibble) to PC
IB6A 3A 02 20   ld      a, (2002h)       ; read data from PC
IB6D C9         ret
IB6D C9         ; End of function write_byte_to_pc
IB6D
IB6D
IB6E
IB6E
IB6E
IB6E
IB6E
IB6E
IB6E
IB6E
IB6E AF         calc_dram_size:         ; CODE XREF: 0080[]p
IB6E AF         xor     a
IB6F
IB6F
IB6F 32 00 20   loc_1B6F:                ; CODE XREF: calc_dram_size+1B[]j
IB6F 32 00 20   ld      (2000h), a       ; select rom/dram page
IB72 ED 47      ld      i, a
IB74 21 01 80   ld      hl, 8001h
IB77 06 0E      ld      b, 14
IB79
IB79
IB79 ED 57      loc_1B79:                ; CODE XREF: calc_dram_size+13[]j
IB79 ED 57      ld      a, i              ; get watermark seed
IB7B AC         xor     h
IB7C AD         xor     l
IB7D 77         ld      (hl), a           ; munge for watermark
IB7E 29         add     hl, hl            ; store in copier DRAM bank
IB7F CB FC      set     7, h              ; add 1 to offset
IB81 10 F6      djnz   loc_1B79         ; set $8XXX range
IB81 10 F6      ; do 14 watermark bytes

```

```
1B83 ED 57          ld      a, i          ; get watermark seed
1B85 37            scf
1B86 17            rla
1B87 FE 40        cp      40h, '0'      ; done 6? banks?
1B89 38 E4        jr      c, loc_1B6F   ; no, loop
1B8B AF           xor     a
1B8C 18 18        jr      loc_1BA6
1B8E              ;
1B8E              ;
1B8E              ;
1B8E              ;
1B8E 32 00 20      check_dram_watermarks: ; CODE XREF: calc_dram_size+3C[]j
1B91 21 01 80      ld      (2000h), a     ; select rom/dram page
1B94 06 0E        ld      hl, 8001h     ; watermark offset
1B96              ld      b, 14         ; 14 watermarks
1B96              ;
1B96              ;
1B96              ;
1B96 18 18        loc_1B96:            ; CODE XREF: calc_dram_size+32[]j
1B96 ED 57        ld      a, i          ; get watermark seed
1B98 AC           xor     h
1B99 AD           xor     l
1B9A BE          cp      (hl)          ; munge for watermark
1B9B 20 0F        jr      nz, loc_1BAC   ; same as written?
1B9D 29          add     hl, hl         ; no, exit
1B9E CB FC        set     7, h          ; next watermark offset
1BA0 10 F4        djnz   loc_1B96       ; $8XXX range
1BA2 ED 57        ld      a, i          ; loop through 14 watermarks
1BA4 37          scf                  ; watermark seed
1BA5 17          rla
1BA6              ;
1BA6              ;
1BA6              ;
1BA6 ED 47        loc_1BA6:            ; CODE XREF: calc_dram_size+1E[]j
1BA8 FE 40        ld      i, a          ;
1BAA 38 E2        cp      40h, '0'      ; done all 6? banks?
1BAC              jr      c, check_dram_watermarks ; no, loop
1BAC              ;
1BAC              ;
1BAC              ;
1BAC ED 57        loc_1BAC:            ; CODE XREF: calc_dram_size+2D[]j
1BAE 3C          ld      a, i          ; get watermark seed
1BAF CB 3F        inc     a
1BB1 32 F0 DF      ld      (0DF00h), a   ; store DRAM size (16kB banks)
1BB4 C9           ret
1BB4              ; End of function calc_dram_size
1BB4              ;
1BB4              ;
1BB5 FF FF FF FF+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
1BB5 FF FF FF FF+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
1BB5 FF FF FF FF+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
1BB5 FF FF FF FF+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
1BB5 FF FF FF FF+ .db 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh, 0FFh
1BEO 46 52 4F 4E+aFrontFareastCocclh_k_.text "FRONT FAREAST COCCL H.K. "
1C00 4C 03 FF 00+char_data: .db 4Ch, 3, 0FFh, 0, 0, 0, 0, 0
1C08 3C 42 B9 A1+ .db 3Ch, 42h, 0B9h, 0A1h, 0B9h, 42h, 3Ch, 0
1C10 3C 7E DB FF+ .db 3Ch, 7Eh, 0DBh, 0FFh, 0C3h, 7Eh, 3Ch, 0
1C18 00 EE FE FE+ .db 0, 0EEh, 0FEh, 0FEh, 7Ch, 38h, 10h, 0
1C20 10 38 7C FE+ .db 10h, 38h, 7Ch, 0FEh, 0FEh, 7Ch, 38h, 10h, 0
1C28 00 3C 18 30+ .db 0, 3Ch, 18h, 0FEh, 0FEh, 8, 18h, 0
1C30 10 38 7C FE+ .db 10h, 38h, 7Ch, 0FEh, 0FEh, 10h, 38h, 0
1C38 00 00 18 3C+ .db 0, 0, 18h, 3Ch, 18h, 0, 0, 0
1C40 FF FF E7 C3+ .db 0FFh, 0FFh, 0E7h, 0C3h, 0E7h, 0FFh, 0FFh, 0FFh
1C48 00 3C 42 81+ .db 0, 3Ch, 42h, 81h, 81h, 42h, 3Ch, 0
1C50 FF C3 BD 7E+ .db 0FFh, 0C3h, 0BDh, 7Eh, 7Eh, 0BDh, 0C3h, 0FFh
1C58 1F 07 0D 7C+ .db 1Fh, 7, 0Dh, 7Ch, 0C6h, 0C6h, 7Ch, 0
1C60 7C FE D9 18+ .db 7Ch, 0FEh, 0D9h, 18h, 18h, 18h, 18h, 18h
1C68 00 18 18 30+ .db 0, 18h, 18h, 30h, 42h, 0FFh, 0C0h, 80h
1C70 08 FC 00 20+ .db 8, 0FCh, 0, 20h, 0F0h, 0, 6, 0FFh
1C78 00 00 80 E0+ .db 0, 0, 80h, 0E0h, 38h, 1Fh, 6, 0
1C80 00 00 60 78+ .db 0, 0, 60h, 78h, 7Eh, 78h, 60h, 0
1C88 00 00 0E 1E+ .db 0, 0, 6, 1Eh, 7Eh, 1Eh, 6, 0
1C90 18 7E 18 18+ .db 18h, 7Eh, 18h, 18h, 18h, 18h, 7Eh, 18h
1C98 66 66 66 66+ .db 66h, 66h, 66h, 66h, 66h, 0, 66h, 0
1CA0 FF B6 76 36+ .db 0FFh, 0B6h, 76h, 36h, 36h, 36h, 36h, 0
1CA8 7E C1 DC 22+ .db 7Eh, 0C1h, 0DCh, 22h, 22h, 1Fh, 83h, 7Eh
1CB0 00 00 00 7E+ .db 0, 0, 7Eh, 7Eh, 0, 0
1CB8 18 7E 18 18+ .db 18h, 7Eh, 18h, 18h, 7Eh, 18h, 0, 0FFh
1CC0 10 38 7C 10+ .db 10h, 38h, 7Ch, 10h, 10h, 10h, 10h, 0
1CC8 10 10 10 10+ .db 10h, 10h, 10h, 10h, 7Ch, 38h, 10h, 0
1CD0 00 04 06 7F+ .db 0, 4, 6, 7Fh, 6, 4, 0, 0
1CD8 00 20 60 FE+ .db 0, 20h, 60h, 0FEh, 60h, 20h, 0, 0
1CE0 00 00 00 C0+ .db 0, 0, 0, 0C0h, 0C0h, 0C0h, 0FFh, 0
1CE8 00 24 66 FF+ .db 0, 24h, 66h, 0FFh, 66h, 24h, 0, 0
1CF0 00 00 10 3E+ .db 0, 0, 10h, 3Eh, 7Ch, 0FEh, 0, 0
1CF8 00 00 00 FE+ .db 0, 0, 0, 0FEh, 7Ch, 38h, 10h, 0
1D00 00 00 00 00+ .db 0, 0, 0, 0, 0, 0, 0, 0
1D08 30 30 30 30+ .db 30h, 30h, 30h, 30h, 30h, 0, 30h, 0
1D10 66 66 00 00+ .db 66h, 66h, 0, 0, 0, 0, 0, 0
1D18 6C 6C FE 6C+ .db 6Ch, 6Ch, 0FEh, 6Ch, 0FEh, 6Ch, 6Ch, 0
1D20 10 7C D2 7C+ .db 10h, 7Ch, 0D2h, 7Ch, 86h, 7Ch, 10h, 0
1D28 F0 96 FC 18+ .db 0F0h, 96h, 0FCh, 18h, 3Eh, 7Eh, 0DEh, 0
1D30 30 48 30 78+ .db 30h, 48h, 30h, 78h, 0CCh, 0CCh, 78h, 0
1D38 0C 0C 18 00+ .db 0Ch, 0Ch, 18h, 0, 0, 0, 0, 0
1D40 10 60 C0 C0+ .db 10h, 60h, 0C0h, 0C0h, 0C0h, 60h, 10h, 0
1D48 10 0C 06 06+ .db 10h, 0Ch, 6, 6, 6, 0Ch, 10h, 0
1D50 00 54 38 FE+ .db 0, 54h, 38h, 0FEh, 38h, 54h, 0, 0
1D58 00 18 18 7E+ .db 0, 18h, 18h, 7Eh, 18h, 18h, 0, 0
1D60 00 00 00 00+ .db 0, 0, 0, 0, 0, 0, 18h, 70h
1D68 00 00 00 7E+ .db 0, 0, 0, 7Eh, 0, 0, 0, 0
1D70 00 00 00 18+ .db 0, 0, 0, 0, 0, 0, 18h, 0
1D78 02 06 0C 18+ .db 2, 6, 0Ch, 18h, 30h, 60h, 0C0h, 0
1D80 7C CE DE FE+ .db 7Ch, 0CEh, 0DEh, 0FEh, 0E6h, 0E6h, 7Ch, 0
1D88 18 38 78 18+ .db 18h, 38h, 78h, 18h, 18h, 18h, 3Ch, 0
1D90 7C C6 06 0C+ .db 7Ch, 0C6h, 6, 0Ch, 30h, 60h, 0FEh, 0
1D98 7C C6 06 3C+ .db 7Ch, 0C6h, 6, 3Ch, 6, 0C6h, 7Ch, 0
1DA0 0E 1E 3E 66+ .db 0Eh, 1Eh, 3Eh, 66h, 0FEh, 6, 6, 0
1DA8 FE C0 C0 FC+ .db 0FEh, 0C0h, 0C0h, 0FCh, 6, 6, 0FCh, 0
1DB0 7C C6 0C 18+ .db 7Ch, 0C6h, 0C0h, 0FCh, 0C6h, 0C6h, 7Ch, 0
1DB8 FE 06 0C 18+ .db 0FEh, 6, 0Ch, 18h, 30h, 60h, 60h, 0
1DC0 7C C6 C6 7C+ .db 7Ch, 0C6h, 0C6h, 7Ch, 0C6h, 0C6h, 7Ch, 0
1DC8 7C C6 C6 7E+ .db 7Ch, 0C6h, 0C6h, 7Eh, 6, 0C6h, 7Ch, 0
1DD0 00 30 00 00+ .db 0, 30h, 0, 0, 0, 30h, 0, 0
1DD8 00 30 00 00+ .db 0, 30h, 0, 0, 0, 30h, 20h, 0
1DE0 00 1C 30 60+ .db 0, 1Ch, 30h, 60h, 30h, 1Ch, 0, 0
1DE8 00 70 18 0C+ .db 0, 0, 7Eh, 0, 7Eh, 0, 0, 0
1DF0 00 70 18 0C+ .db 0, 70h, 18h, 0Ch, 18h, 70h, 0, 0
1DF8 7C C6 0C 18+ .db 7Ch, 0C6h, 0Ch, 18h, 30h, 0, 30h, 0
1E00 7C 82 9A AA+ .db 7Ch, 82h, 9Ah, 0AAh, 0AAh, 9Eh, 7Ch, 0
1E08 7C C6 C6 FE+ .db 7Ch, 0C6h, 0C6h, 0FEh, 0C6h, 0C6h, 0C6h, 0
1E10 FC 66 66 7C+ .db 0FCh, 66h, 66h, 7Ch, 66h, 66h, 0FCh, 0
1E18 7C C6 C0 C0+ .db 7Ch, 0C6h, 0C0h, 0C0h, 0C0h, 0C6h, 7Ch, 0
1E20 FC 66 66 66+ .db 0FCh, 66h, 66h, 66h, 66h, 66h, 0FCh, 0
1E28 FE 62 68 78+ .db 0FEh, 62h, 68h, 78h, 68h, 62h, 0FEh, 0
1E30 FE 62 68 78+ .db 0FEh, 62h, 68h, 78h, 68h, 60h, 0F0h, 0
1E38 7C C6 C6 C0+ .db 7Ch, 0C6h, 0C6h, 0C0h, 0DEh, 0C6h, 7Ch, 0
1E40 C6 C6 C6 FE+ .db 0C6h, 0C6h, 0C6h, 0FEh, 0C6h, 0C6h, 0C6h, 0
1E48 3C 18 18 18+ .db 3Ch, 18h, 18h, 18h, 18h, 18h, 3Ch, 0
1E50 1E 0C 0C 0C+ .db 1Eh, 0Ch, 0Ch, 0Ch, 0Ch, 0CCh, 78h, 0
1E58 C6 CC D8 F0+ .db 0C6h, 0CCh, 0D8h, 0F0h, 0D8h, 0CCh, 0C6h, 0
1E60 F0 60 60 60+ .db 0F0h, 60h, 60h, 60h, 60h, 62h, 0FEh, 0
1E68 C6 EE FE D6+ .db 0C6h, 0EEh, 0FEh, 0D6h, 0C6h, 0C6h, 0C6h, 0
1E70 C6 E6 FE DE+ .db 0C6h, 0E6h, 0FEh, 0DEh, 0C6h, 0C6h, 0C6h, 0
1E78 7C C6 C6 C6+ .db 7Ch, 0C6h, 0C6h, 0C6h, 0C6h, 0C6h, 7Ch, 0
1E80 FC 66 66 7C+ .db 0FCh, 66h, 66h, 7Ch, 60h, 60h, 0F0h, 0
1E88 7C C6 C6 C6+ .db 7Ch, 0C6h, 0C6h, 0C6h, 0C6h, 0C6h, 7Ch, 0Ch
1E90 FC 66 66 7C+ .db 0FCh, 66h, 66h, 7Ch, 66h, 66h, 0E6h, 0
1E98 7C C6 C0 7C+ .db 7Ch, 0C6h, 0C0h, 7Ch, 6, 0C6h, 7Ch, 0
1EA0 7E 5A 18 18+ .db 7Eh, 5Ah, 18h, 18h, 18h, 18h, 3Ch, 0
1EA8 C6 C6 C6 C6+ .db 0C6h, 0C6h, 0C6h, 0C6h, 0C6h, 0C6h, 7Ch, 0
1EB0 C6 C6 C6 C6+ .db 0C6h, 0C6h, 0C6h, 0C6h, 0C6h, 6Ch, 38h, 0
1EB8 C6 C6 C6 C6+ .db 0C6h, 0C6h, 0C6h, 0C6h, 0C6h, 0C6h, 0C6h, 0
1EC0 C6 6C 38 38+ .db 0C6h, 6Ch, 38h, 38h, 38h, 6Ch, 0C6h, 0
1EC8 66 66 66 3C+ .db 66h, 66h, 66h, 3Ch, 18h, 18h, 3Ch, 0
1ED0 FE C6 0C 18+ .db 0FEh, 0C6h, 0Ch, 18h, 30h, 66h, 0FEh, 0
1ED8 1C 18 18 18+ .db 1Ch, 18h, 18h, 18h, 18h, 18h, 1Ch, 0
1EE0 C0 60 30 18+ .db 0C0h, 60h, 30h, 18h, 0Ch, 6, 2, 0
1EE8 70 30 30 30+ .db 70h, 30h, 30h, 30h, 30h, 30h, 70h, 0
```

```
1EF0 00 00 10 38+ .db 0, 0, 10h, 38h, 6Ch, 0C6h, 0, 0
1EF8 00 00 00 00+ .db 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
1F00 00 00 00 7F+ .db 0, 0, 0, 7Fh, 0, 0, 0, 0
1F08 00 00 0E 0C+ .db 0, 0, 0Eh, 0Ch, 0Ch, 0Ch, 0Ch, 0Ch, 0Ch
1F10 0C 0C 0C 0C+ .db 0Ch, 0Ch, 0Ch, 0Ch, 0Ch, 0Ch, 0Ch, 0FFh, 0
1F18 00 00 00 FF+ .db 0, 0, 0, 0, 0FFh, 18h, 18h, 18h, 18h
1F20 18 18 18 18+ .db 18h, 18h, 18h, 18h, 18h, 1Fh, 18h, 18h, 18h
1F28 18 18 18 18+ .db 18h, 18h, 18h, 18h, 18h, 18h, 18h, 0FFh, 0
1F30 00 00 18 FC+ .db 0, 0, 18h, 0FCh, 0, 0, 0, 0
1F38 00 00 00 30+ .db 0, 0, 0, 30h, 0F8h, 0, 0, 0
1F40 00 00 00 00+ .db 0, 0, 0, 0, 4, 0Eh, 0FFh, 0
1F48 00 07 06 06+ .db 0, 7, 6, 6, 0Ch, 0Fh, 18h, 10h
1F50 20 40 00 FF+ .db 20h, 40h, 0, 0, 0FFh, 0, 0, 0, 0
1F58 01 01 03 03+ .db 1, 1, 3, 3, 3, 6, 0Ch, 30h, 0C0h
1F60 1C 18 18 18+ .db 1Ch, 18h, 18h, 18h, 18h, 0FFh, 18h, 18h
1F68 18 18 18 FF+ .db 18h, 18h, 18h, 0FFh, 0C6h, 0C6h, 0C6h, 0C6h
1F70 86 86 06 06+ .db 86h, 86h, 6, 6, 6, 6, 7, 3
1F78 00 00 00 00+ .db 0, 0, 0, 0, 18h, 0FCh, 0, 0
1F80 00 00 06 FF+ .db 0, 0, 6, 0FFh, 0, 0, 0, 0
1F88 00 02 02 02+ .db 0, 2, 2, 2, 3, 0FFh, 0Fh
1F90 00 40 7F 40+ .db 0, 40h, 7Fh, 40h, 0C3h, 0C3h, 3, 0FFh
1F98 06 06 07 00+ .db 6, 6, 7, 0, 0Fh, 0Ch, 0Fh, 0Ch
1FA0 0C 0F 0C 0C+ .db 0Ch, 0Fh, 0Ch, 0Ch, 0Fh, 3, 0Eh, 78h
1FA8 1C 18 FF 00+ .db 1Ch, 18h, 0FFh, 0, 0FFh, 18h, 18h, 0FFh
1FB0 30 30 FF 00+ .db 30h, 30h, 0FFh, 0, 0FFh, 0, 0FFh, 0
1FB8 00 FF 00 00+ .db 0, 0FFh, 0, 0, 0FFh, 80h, 0, 0
1FC0 00 02 FF 06+ .db 0, 2, 0FFh, 6, 0E4h, 60h, 62h, 0FFh
1FC8 C0 C0 C0 00+ .db 0C0h, 0C0h, 0C0h, 0, 0F8h, 38h, 0F0h, 30h
1FD0 30 F0 30 30+ .db 30h, 0F0h, 30h, 30h, 0F0h, 0E0h, 38h, 0Ch
1FD8 00 10 0C 06+ .db 0, 10h, 0Ch, 6, 6, 0FFh, 2, 1
1FE0 00 3F 00 00+ .db 0, 3Fh, 0, 0, 0Fh, 0, 0, 0FFh
1FE8 00 00 01 03+ .db 0, 0, 1, 3, 6, 18h, 0E0h, 0
1FF0 66 66 66 66+ .db 66h, 66h, 66h, 66h, 66h, 0FFh, 0, 81h
1FF8 C2 FF 18 18+ .db 0C2h, 0FFh, 18h, 18h, 0FFh, 18h, 18h, 0FFh
1FF8 FF 18 18 FF ; end of 'seg000'
1FF8
1FF8
1FF8 .end
```

SMD BIOS V3 Memory Map

```

=====
0000-1FFF ROM
2000-3FFF I/O
-----
2000 Select cartridge ROM / copier DRAM page
2001 Memory and mode control
2002 Parallel I/O data I/O
2003 Parallel I/O busy flag
2009 FDC digital input register? (RO)
      7 - floppy drive ready???
      6 - cartridge present
      5 - disk present in drive???
200B FDC digital output register (WO)
200C FDC data register (R/W)
200D FDC status register (RO)
200E FDC control register

4000-7FFF Bank #1 (lower 16K SRAM, or banked cart ROM)
-----
8000-BFFF Bank #2 (upper 16K SRAM, or banked copier DRAM)
-----
C000-DFFF RAM
-----
DA00-DBFF sector buffer & scratchpad
DC00-DDFF SMD header scratch buffer

DE82/DE83 current memory address for disk read data
DE88/DE89 cluster number of file in current directory entry
DE8A/DE8B dram buffer address
DE8C/DE8D ??? address
DE90 unit select (format) /
DE96 number of sectors in file
DE9A number of tracks (scratch)

DE9D
DE9E track (write)
DE9F drive head address (write)
DEA0 sector (write)
DEA1
DEA2
DEA6 scratch - disk format (0-3)/file type/???
DEA7 number of tracks (format)
DEA8 sectors/track (format)
DEA9 in sector gap length (format)
DEAA next rom/dram page

DEAD

DEB0 maximum number of root directory entries
DEB1 sectors/track
DEB2 media type
DEB3 fdc command
DEB4 unit select
DEB5 track number
DEB6 drive head address
DEB7 sector number
DEB8 number of bytes / sector
DEB9 end track sector number
DEBA in sector gap length
DEBB data length
DEBC ST0
DEBD ST1
DEBE ST2
DEBF present cylinder number (PCN)

DEC3-DECD filename input buffer
DECE-DEED directory entry buffer
DECE-DED8 filename
DED9 file attribute
DEDA reserved
DEDB file creation millisecond
DEDC-DEDD file creation time
DEDE-DEDF file creation date
DEE0-DEE1 file accessed date
DEE2-DEE3 high 16-bit cluster number
DEE4-DEE5 file modification time
DEE6-DEE7 file modification date
DEE8-DEE9 16-bit cluster number
DEEA-DEED file size in bytes
DEEF ???

DF8B

DF8E/DF8F table of routines in current menu
DF90 number of routines in current menu
DF91 current menu option
DF92 main/utility menu
DF93 ??? for current menu
DF94 ??? for current menu
DF95 controller input

DFE0/1 bytes per sector (actual)
DFE2
DFE3
DFE8 ST3
DFE9 track
DFEA head
DFEB sector
DFEC bytes/sector (token)
DFE0 available DRAM size (16kB pages)
DFE1 cartridge size (16kB pages)
DFE9

```

E000-FFFF RAM (mirror)

Port Map

```

=====
7F ???
BE VDP data port
BF VDP control port
DC controller input
  5 - button C
  4 - button B
  3 - right
  2 - left
  1 - down
  0 - up

```

Other notes/references

```

=====

```

Byte	Capacity	Media Size and Type
F0	2.88 MB	3.5-inch, 2-sided, 36-sector
F0	1.44 MB	3.5-inch, 2-sided, 18-sector
F9	720K	3.5-inch, 2-sided, 9-sector
F9	1.2 MB	5.25-inch, 2-sided, 15-sector
FD	360K	5.25-inch, 2-sided, 9-sector
FF	320K	5.25-inch, 2-sided, 8-sector
FC	180K	5.25-inch, 1-sided, 9-sector
FE	160K	5.25-inch, 1-sided, 8-sector
FE	250K	8-inch, 1-sided, single-density
FD	500K	8-inch, 2-sided, single-density
FE	1.2 MB	8-inch, 2-sided, double-density
F8	----	Hard disk