

```

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 : E:\Projects\NeoKong\arcade\dkong.bin
0000 ; Format : Binary File
0000 ; Base Address: 0000h Range: 0000h - 4000h Loaded length: 4000h
0000 ;
0000 ; Processor: z80
0000 ; Target assembler: ASxxxx by Alan R. Baldwin v1.5
0000 .area idaseg (ABS)
0000 .hd64 ; this is needed only for HD64180
0000 ;
0000 ; Segment type: Pure code
0000 ; segment 'ROM'
0000
0000 RESET: ; CODE XREF: 0000:00B2|j
0000 3E 00 ; DATA XREF: 0000:0FCD|o
0000 ld a, #0
0002 32 84 7D ld (nmi_mask), a
0005 C3 66 02 jp INIT
0008 ; SUBROUTINE
0008
0008 return_if_attract_mode: ; CODE XREF: flash_1UP_or_2UP+7|p
0008 3A 07 60 ; add_bonus_and_update_high_score+1|p ...
0008 ld a, (attract_mode_flag)
000B 0F rrca ; in attract mode?
000C D0 ret NC ; no, return
000D 33 inc sp
000E 33 inc sp ; discard return address
000F C9 ret
000F ; End of function return_if_attract_mode
0010 ; SUBROUTINE
0010
0010 return_if_mario_not_alive: ; CODE XREF: sub_0_3A2+3|p
0010 3A 00 62 ; sub_0_2C03+3|p ...
0010 ld a, (mario_alive_flag)
0013 0F rrca ; is mario alive?
0014 D8 ret C ; yes, return
0015 33 inc sp
0016 33 inc sp ; discard return address
0017 C9 ret
0017 ; End of function return_if_mario_not_alive
0018 ; SUBROUTINE
0018
0018 return_NOT_8bit_timeout: ; CODE XREF: return_NOT_16bit_timeout+4|j
0018 21 09 60 ; display_1UP+10|p ...
0018 ld hl, #eight_bit_countdown
001B 35 dec (hl)
001C C8 ret Z
001D 33 inc sp
001E 33 inc sp ; discard return address
001F C9 ret
001F ; End of function return_NOT_8bit_timeout
0020 ; SUBROUTINE
0020
0020 return_NOT_16bit_timeout: ; CODE XREF: 0000:0763|p
0020 21 08 60 ; 0000:084B|p
0020 ld hl, #sixteen_bit_countdown_msb
0023 35 dec (hl)
0024 28 F2 jr Z, return_NOT_8bit_timeout
0026
0026 pop_hl_ret: ; CODE XREF: print_message_A+1A|j
0026 E1 pop hl ; sub_0_1783+4|j
0027 C9 ret ; discard return address
0027 ; End of function return_NOT_16bit_timeout
0028 ; SUBROUTINE
0028
0028 jump_table_go_A: ; CODE XREF: 0000:00C9|p
0028 87 ; 0000:0701|p ...
0028 add a, a ; entries are words
0029 E1 pop hl ; return address is table base
002A 5F ld e, a
002B 16 00 ld d, #0 ; DE = offset
002D C3 32 00 jp loc_0_32 ; skip vector address
002D ; End of function jump_table_go_A
0030 ; SUBROUTINE
0030
0030 sub_0_30: ; CODE XREF: sub_0_3A2+2|p
0030 18 12 ; 0000:1668|p ...
0030 jr return_if_level_bit_not_set
0032 ;
0032
0032 loc_0_32: ; CODE XREF: jump_table_go_A+5|j
0032 19 add hl, de ; get address of entry
0033 5E ld e, (hl)
0034 23 inc hl
0035 56 ld d, (hl) ; DE = jump address
0036 EB ex de, hl ; HL = jump address
0037 E9 jp (hl) ; go

```

```

0038 ; -----
0038
0038 add_c_sprite_register_x10: ; CODE XREF: animate_kong_and_pauline+F|p
0038 11 04 00 ; animate_kong_and_pauline+65|p ...
0038 ld de, #4 ; every 4th byte
0038 06 0A ld b, #10 ; loop 10 times
003D
003D add_c_sprite_register_xB: ; CODE XREF: sub_0_30+11|j
003D 79 ; 0000:0D9A|p ...
003D ld a, c
003E 86 add a, (hl)
003F 77 ld (hl), a ; (HL)+=C
0040 19 add hl, de ; next byte
0041 10 FA djnz add_c_sprite_register_xB ; loop
0043 C9 ret
0044 ; -----
0044
0044 return_if_level_bit_not_set: ; CODE XREF: sub_0_30|j
0044 21 27 62 ld hl, #level_type
0047 46 ld b, (hl) ; get level type
0048
0048 loc_0_48: ; CODE XREF: sub_0_30+19|j
0048 0F rrca
0049 10 FD djnz loc_0_48 ; get bit of A for level
004B D8 ret C ; bit set, return
004C E1 pop hl ; discard return address
004D C9 ret
004D ; End of function sub_0_30
004D
004E ; SUBROUTINE
004E
004E copy_sprites_2_11_data: ; CODE XREF: animate_kong_and_pauline+4D|p
004E 11 08 69 ; animate_kong_and_pauline+77|p ...
004E ld de, #soft_sprite_ram+8 ; ptr sprite #2
0051 01 28 00 ld bc, #40 ; 10 4-byte sprites to copy
0054 ED B0 ldir ; copy 40 bytes of sprite data
0056 C9 ret
0056 ; End of function copy_sprites_2_11_data
0056
0057 ; SUBROUTINE
0057
0057 rand: ; CODE XREF: 0000:00B9|p
0057 3A 18 60 ; sub_0_2523+22|p ...
0057 ld a, (random_no)
005A 21 1A 60 ld hl, #gen_purpose_timer
005D 86 add a, (hl)
005E
005E loc_0_5E:
005E 21 19 60 ld hl, #random_no+1
0061 86 add a, (hl)
0062 32 18 60 ld (random_no), a
0065 C9 ret
0065 ; End of function rand
0065
0066 ; -----
0066
0066 nmi:
0066 F5 push af
0067 C5 push bc
0068 D5 push de
0069 E5 push hl
006A DD E5 push ix
006C FD E5 push iy
006E AF xor a
006F 32 84 7D ld (nmi_mask), a ; disable_nmi
0072 3A 00 7D ld a, (in2_snd_latch) ; IN2
0075 E6 01 and #1 ; bit 0 set?
0077 C2 00 40 jp NZ, 0x4000 ; yes, boom! (not valid code)
007A 21 38 01 ld hl, #dma_reg_tbl
007D CD 41 01 call dma_sprite_data_to_hw ; update sprites
0080 3A 07 60 ld a, (attract_mode_flag)
0083 A7 and a ; in attract mode?
0084 C2 B5 00 jp NZ, loc_0_B5 ; yes, skip reading inputs
0087 3A 26 60 ld a, (upright)
008A A7 and a
008B C2 98 00 jp NZ, loc_0_98
008E 3A 0E 60 ld a, (current_player_E)
0091 A7 and a ; player 2?
0092 3A 80 7C ld a, (in1) ; (cocktail)
0095 C2 9B 00 jp NZ, loc_0_9B ; yes, skip
0098
0098 loc_0_98: ; CODE XREF: 0000:008B|j
0098 3A 00 7C ld a, (in0) ; (upright)
009B
009B loc_0_9B: ; CODE XREF: 0000:0095|j
009B 47 ld b, a ; store IN0/1
009C E6 0F and #0xF ; joystick only
009E 4F ld c, a ; store
009F 3A 11 60 ld a, (last_raw_in) ; last raw input
00A2 2F cpl ; negate
00A3 A0 and b ; rising-edge detect
00A4 E6 10 and #0x10 ; button
00A6 17 rla
00A7 17 rla ; bit 7
00A8 17 rla ; add joystick bits
00A9 B1 or c ; raw controller input
00AA 60 ld h, b ; joystick and button press
00AB 6F ld l, a ; store
00AC 22 10 60 ld (controller_in), hl
00AF 78 ld a, b ; reset input?
00B0 CB 77 bit 6, a ; reset input?
00B2 C2 00 00 jp NZ, RESET
00B5
00B5 loc_0_B5: ; CODE XREF: 0000:0084|j
00B5 21 1A 60 ld hl, #gen_purpose_timer ; general purpose timer tick
00B8 35 dec (hl) ; randomise
00B9 CD 57 00 call rand
00BC CD 7B 01 call check_coin_inserted
00BF CD E0 00 call update_sounds
00C2 21 D2 00 ld hl, #nmi_exit ; IRQ resume address
00C5 E5 push hl
00C6 3A 05 60 ld a, (nmi_sequencer)

```

```

00C9 EF          rst      0x28          ; go!
00C9
00CA C3 01      ;          .dw init_machine_settings          ; Jump table (nmi sequencer)
00CC 3C 07      .dw chk_credits_and_vector_on_attrac
00CE B2 08      .dw vector_on_credit_sequencer
00D0 FE 06      .dw vector_on_ingame_sequencer
00D2
00D2
00D2          nmi_exit:
00D2 FD E1      pop      iy          ; DATA XREF: 0000:00C2|o
00D4 DD E1      pop      ix
00D6 E1          pop      hl
00D7 D1          pop      de
00D8 C1          pop      bc
00D9 3E 01      ld      a, #1
00DB 32 84 7D    ld      (nmi_mask), a          ; enable_nmi
00DE F1          pop      af
00DF C9          ret
00E0
00E0          ; ██████████ S U B R O U T I N E ██████████
00E0
00E0
00E0          update_sounds:
00E0 21 80 60      ld      hl, #digital_snd_tmr_walk          ; CODE XREF: 0000:00BF|p
00E3 11 00 7D    ld      de, #in2_snd_latch          ; base of digital sound triggers
00E6 3A 07 60      ld      a, (attract_mode_flag)
00E9 A7          and      a          ; in attract mode?
00EA C0          ret      NZ          ; yes, return
00EB 06 08      ld      b, #8          ; 8 digital sound triggers
00ED
00ED          loc_0_ED:
00ED 7E          ld      a, (hl)          ; CODE XREF: update_sounds+18|j
00EE A7          and      a          ; timer for this sound
00EF CA F5 00      jp      Z, loc_0_F5          ; done?
00F2 35          dec     (hl)          ; yes, skip
00F3 3E 01      ld      a, #1          ; decrement timer
00F5
00F5          loc_0_F5:
00F5 12          ld      (de), a          ; CODE XREF: update_sounds+F|j
00F6 1C          inc     e          ; set trigger state for this sound
00F7 2C          inc     l          ; next latch
00F8 10 F3      djnz   loc_0_ED          ; next timer
00FA 21 8B 60      ld      hl, #unk_0_608B
00FD 7E          ld      a, (hl)
00FE A7          and      a
00FF C2 08 01      jp      NZ, loc_0_108
0102 2D          dec     l
0103 2D          dec     l
0104 7E          ld      a, (hl)
0105 C3 0B 01      jp      set_bg_sound_music
0108
0108          loc_0_108:
0108 35          dec     (hl)          ; CODE XREF: update_sounds+1F|j
0109 2D          dec     l
010A 7E          ld      a, (hl)          ; get background sound/music
010B
010B          set_bg_sound_music:
010B 32 00 7C      ld      (in0), a          ; CODE XREF: update_sounds+25|j
010E 21 88 60      ld      hl, #music_something          ; background sound/music select
0111 AF          xor      a
0112 BE          cp      (hl)          ; any music to play?
0113 CA 18 01      jp      Z, loc_0_118          ; no, skip
0116 35          dec     (hl)          ; ???
0117 3C          inc     a          ; flag music start
0118
0118          loc_0_118:
0118 32 80 7D      ld      (dsw_audio_irq), a          ; CODE XREF: update_sounds+33|j
011B C9          ret          ; digital sound - dead
011B
011B          ; End of function update_sounds
011C
011C          ; ██████████ S U B R O U T I N E ██████████
011C
011C
011C          stop_sound:
011C 06 08          ld      b, #8          ; CODE XREF: check_coin_inserted+1A|p
011E AF          xor      a          ; 0000:02B5|p ...
011F 21 00 7D    ld      hl, #in2_snd_latch          ; sound latch
0122 11 80 60      ld      de, #digital_snd_tmr_walk          ; timers
0125
0125          loc_0_125:
0125 77          ld      (hl), a          ; CODE XREF: stop_sound+D|j
0126 12          ld      (de), a          ; kill latch
0127 2C          inc     l          ; kill timer
0128 1C          inc     e
0129 10 FA      djnz   loc_0_125          ; write 8 bytes
012B 06 04      ld      b, #4
012D
012D          loc_0_12D:
012D 12          ld      (de), a          ; CODE XREF: stop_sound+13|j
012E 1C          inc     e
012F 10 FC      djnz   loc_0_12D          ; another 4 copies
0131 32 80 7D    ld      (dsw_audio_irq), a          ; audio IRQ
0134 32 00 7C    ld      (in0), a          ; background music = NONE
0137 C9          ret
0137
0137          ; End of function stop_sound
0137
0137          ;
0138 53          dma_reg_tbl: .db 0x53          ; DATA XREF: 0000:007A|o
0138
0139 00 69          .dw soft_sprite_ram          ; DMA mode (TC stop, CH0,1)
013B 80 41      .dw 0x4180          ; CH0 address
013D 00 70      .dw SPRAM_start          ; CH0 terminal count (RD 0x180 bytes)
013F 80 81      .dw 0x8180          ; CH1 Address
0141
0141          ; ██████████ S U B R O U T I N E ██████████
0141
0141
0141          dma_sprite_data_to_hw:
0141 AF          xor      a          ; CODE XREF: 0000:007D|p
0142 32 85 7D    ld      (p8257_drq), a          ; deassert DRQ0&1
0145 7E          ld      a, (hl)          ; 0x53
0146 32 08 78    ld      (i8257_io+8), a          ; mode set
0149 23          inc     hl

```

```

014A 7E      ld      a, (hl)                ; 0x00
014B 32 00 78 ld      (i8257_io), a            ; CH0 DMA address LSB
014E 23      inc     hl                ;
014F 7E      ld      a, (hl)                ; 0x69
0150 32 00 78 ld      (i8257_io), a            ; CH0 DMA address MSB
0153 23      inc     hl                ;
0154 7E      ld      a, (hl)                ; 0x80
0155 32 01 78 ld      (i8257_io+1), a        ; terminal count LSB
0158 23      inc     hl                ;
0159 7E      ld      a, (hl)                ; 0x41
015A 32 01 78 ld      (i8257_io+1), a        ; terminal count MSB
015D 23      inc     hl                ;
015E 7E      ld      a, (hl)                ; 0x00
015F 32 02 78 ld      (i8257_io+2), a        ; CH1 DMA address LSB
0162 23      inc     hl                ;
0163 7E      ld      a, (hl)                ; 0x70
0164 32 02 78 ld      (i8257_io+2), a        ; CH1 DMA address MSB
0167 23      inc     hl                ;
0168 7E      ld      a, (hl)                ; 0x80
0169 32 03 78 ld      (i8257_io+3), a        ; CH1 terminal count LSB
016C 23      inc     hl                ;
016D 7E      ld      a, (hl)                ; 0x81
016E 32 03 78 ld      (i8257_io+3), a        ; CH1 terminal count MSB
0171 3E 01      ld      a, #1
0173 32 85 7D ld      (p8257_drq), a        ; assert DRQ0&1
0176 AF      xor      a
0177 32 85 7D ld      (p8257_drq), a        ; deassert DRQ0&1
017A C9      ret
017A          ; End of function dma_sprite_data_to_hw
017A
017B          ; ██████████ SUBROUTINE ██████████
017B
017B          check_coin_inserted:                ; CODE XREF: 0000:00BC|p
017B 3A 00 7D      ld      a, (in2_snd_latch)        ; read IN2
017E CB 7F      bit      7, a                ; coin?
0180 21 03 60      ld      hl, #coin_state
0183 C2 89 01      jp      NZ, coin_inserted        ; yes, skip
0186 36 01      ld      (hl), #1
0188 C9      ret
0189
0189          coin_inserted:                ; CODE XREF: check_coin_inserted+8|j
0189 7E      ld      a, (hl)
018A A7      and      a
018B C8      ret      Z                ; debounce
018C E5      push    hl
018D 3A 05 60      ld      a, (nmi_sequencer)
0190 FE 03      cp      #3                ; in credit sequence?
0192 CA 9D 01      jp      Z, loc_0_19D        ; yes, skip
0195 CD 1C 01      call   stop_sound
0198 3E 03      ld      a, #3                ; tmr = 3
019A 32 83 60      ld      (digital_snd_tmr_coin_spring), a
019D
019D          loc_0_19D:                ; CODE XREF: check_coin_inserted+17|j
019D E1      pop     hl
019E 36 00      ld      (hl), #0                ; flag coin intersted
01A0 2B      dec     hl
01A1 34      inc     (hl)                ; inc coins_not_credited
01A2 11 24 60      ld      de, #coinage+2        ; ptr coins/credit
01A5 1A      ld      a, (de)
01A6 96      sub     (hl)                ; sub coins_not_credited
01A7 C0      ret      NZ                ; not enough coins for a credit
01A8 77      ld      (hl), a                ; update coins_not_credited (0)
01A9 13      inc     de
01AA 2B      dec     hl                ; no_of_credits
01AB EB      ex      de, hl                ; DE=no_of_credits, HL=credits/coin
01AC 1A      ld      a, (de)                ; no_of_credits
01AD FE 90      cp      #0x90 ; 'É'        ; max credits?
01AF D0      ret      NC                ; yes, take coins and exit
01B0 86      add     a, (hl)                ; add number of credits per coin
01B1 27      daa
01B2 12      ld      (de), a                ; update number of credits
01B3 11 00 04      ld      de, #0x400        ; display_credits_if_attract_mode
01B6 CD 9F 30      call   queue_fg_vector_fn
01B9 C9      ret
01B9          ; End of function check_coin_inserted
01B9
01B9          ;
01BA 00 37 00      initial_scores_and_high_score:.db 0, 0x37, 0        ; DATA XREF: 0000:01C6|o
01BA          ; Initial score and high score on bootup
01BD AA AA AA      byte_0_1BD: .db 0xAA, 0xAA, 0xAA        ; DATA XREF: 0000:159D|o
01C0 50 76 00      .db 0x50, 0x76, 0
01C3
01C3          ;
01C3          init_machine_settings:                ; DATA XREF: 0000:00CA|o
01C3 CD 74 08      call   clear_visible_area_and_sprites
01C6 21 BA 01      ld      hl, #initial_scores_and_high_score        ; copy in ROM
01C9 11 B2 60      ld      de, #pl_score        ; RAM location
01CC 01 09 00      ld      bc, #9                ; 9 bytes to copy
01CF ED B0      ldir                ; copy scores to RAM
01D1 3E 01      ld      a, #1
01D3 32 07 60      ld      (attract_mode_flag), a        ; set attract mode flag
01D6 32 29 62      ld      (level), a
01D9 32 28 62      ld      (lives_left), a
01DC CD B8 06      call   display_lives_and_level
01DF CD 07 02      call   read_dips_and_high_score_tbl
01E2 3E 01      ld      a, #1
01E4 32 82 7D      ld      (flipscreen), a
01E7 32 05 60      ld      (nmi_sequencer), a                ; next sequence
01EA 32 27 62      ld      (level_type), a
01ED AF      xor      a
01EE 32 0A 60      ld      (main_sequencer), a        ; game screen sequencer
01F1 CD 53 0A      call   display_LUP
01F4 11 04 03      ld      de, #0x304
01F7 CD 9F 30      call   queue_fg_vector_fn        ; print_message_A
01FA 11 02 02      ld      de, #0x202
01FD CD 9F 30      call   queue_fg_vector_fn        ; display_score_or_high_score
0200 11 00 02      ld      de, #0x200
0203 CD 9F 30      call   queue_fg_vector_fn        ; display_score_or_high_score
0206 C9      ret
0207
0207          ; ██████████ SUBROUTINE ██████████
0207
0207
0207

```

```

0207      read_dips_and_high_score_tbl:
0207 3A 80 7D      ld      a, (dsw_audio_irq)      ; CODE XREF: 0000:01DF|p
020A 4F      ld      c, a      ; read DIPSW
020B 21 20 60      ld      hl, #lives_per_game      ; store
020E E6 03      and     #3      ; lives setting
0210 C6 03      add     a, #3      ; init no. of lives
0212 77      ld      (hl), a      ; store no. of lives
0213 23      inc     hl
0214 79      ld      a, c      ; DIPSW
0215 0F      rrca
0216 0F      rrca
0217 E6 03      and     #3      ; bonus life setting
0219 47      ld      b, a
021A 3E 07      ld      a, #7      ; 7,000?
021C CA 26 02      jp     Z, loc_0_226      ; yes, skip
021F 3E 05      ld      a, #5      ; 5,000?
0221
0221      loc_0_221:      ; CODE XREF: read_dips_and_high_score_tbl+1D|j
0221 C6 05      add     a, #5
0223 27      daa
0224 10 FB      djnz    loc_0_221      ; calculate 10/15/20K points
0226
0226      loc_0_226:      ; CODE XREF: read_dips_and_high_score_tbl+15|j
0226 77      ld      (hl), a      ; bonus_setting
0227 23      inc     hl
0228 79      ld      a, c      ; DIPSW
0229 01 01 01      ld      bc, #0x101      ; 1C P1
022C 11 02 01      ld      de, #0x102      ; 1C P2
022F E6 70      and     #0x70 ; 'p'      ; coinage setting
0231 17      rla
0232 17      rla
0233 17      rla
0234 17      rla      ; coinage 0-7
0235 CA 47 02      jp     Z, loc_0_247      ; 1C1C
0238 DA 41 02      jp     C, loc_0_241      ; 2-5 coins
023B 3C      inc     a      ; no. credits
023C 4F      ld      c, a      ; C = credits
023D 5A      ld      e, d      ; D = coins
023E C3 47 02      jp     loc_0_247
0241
0241      ;
0241      loc_0_241:      ; CODE XREF: read_dips_and_high_score_tbl+31|j
0241 C6 02      add     a, #2      ; no. coins
0243 47      ld      b, a      ; B = coins
0244 57      ld      d, a      ; D = coins
0245 87      add     a, a
0246 5F      ld      e, a      ; E = coins x2
0247
0247      loc_0_247:      ; CODE XREF: read_dips_and_high_score_tbl+2E|j
0247 72      ; read_dips_and_high_score_tbl+37|j
0247 77      ld      (hl), d
0248 23      inc     hl
0249 73      ld      (hl), e
024A 23      inc     hl
024B 70      ld      (hl), b
024C 23      inc     hl
024D 71      ld      (hl), c
024E 23      inc     hl
024F 3A 80 7D      ld      a, (dsw_audio_irq)      ; read DIPSW
0252 07      rlca      ; upright?
0253 3E 01      ld      a, #1
0255 DA 59 02      jp     C, loc_0_259      ; yes, skip
0258 3D      dec     a
0259
0259      loc_0_259:      ; CODE XREF: read_dips_and_high_score_tbl+4E|j
0259 77      ld      (hl), a      ; store cocktail/upright
025A 21 65 35      ld      hl, #high_score_tbl
025D 11 00 61      ld      de, #high_score_tbl_ram      ; destination in RAM
0260 01 AA 00      ld      bc, #0xAA ; '-'      ; length of table
0263 ED B0      ldir      ; copy to ram
0265 C9      ret
0265      ; End of function read_dips_and_high_score_tbl
0265
0266      ;
0266
0266      INIT:      ; CODE XREF: 0000:0005|j
0266 06 10      ld      b, #16
0268 21 00 60      ld      hl, #RAM_start      ; start of RAM
026B AF      xor     a      ; zero byte
026C
026C      loc_0_26C:      ; CODE XREF: 0000:0272|j
026C 4F      ld      c, a
026D
026D      loc_0_26D:      ; CODE XREF: 0000:0270|j
026D 77      ld      (hl), a      ; zero memory
026E 23      inc     hl      ; next location
026F 0D      dec     c
0270 20 FB      jr     NZ, loc_0_26D      ; clear 256 bytes
0272 10 F8      djnz    loc_0_26C      ; clear 4K bytes
0274 06 04      ld      b, #4
0276 21 00 70      ld      hl, #SPRAM_start      ; start of sprite RAM
0279
0279      loc_0_279:      ; CODE XREF: 0000:027F|j
0279 4F      ld      c, a
027A
027A      loc_0_27A:      ; CODE XREF: 0000:027D|j
027A 77      ld      (hl), a      ; zero memory
027B 23      inc     hl      ; next location
027C 0D      dec     c
027D 20 FB      jr     NZ, loc_0_27A      ; clear 256 bytes
027F 10 F8      djnz    loc_0_279      ; clear 1K bytes
0281 06 04      ld      b, #4
0283 3E 10      ld      a, #0x10      ; space character
0285 21 00 74      ld      hl, #VRAM_start      ; start of VRAM
0288
0288      loc_0_288:      ; CODE XREF: 0000:028F|j
0288 0E 00      ld      c, #0
028A
028A      loc_0_28A:      ; CODE XREF: 0000:028D|j
028A 77      ld      (hl), a      ; clear memory
028B 23      inc     hl      ; next location
028C 0D      dec     c
028D 20 FB      jr     NZ, loc_0_28A      ; clear 256 bytes
028F 10 F7      djnz    loc_0_288      ; clear 1K bytes
0291 21 C0 60      ld      hl, #fg_vector_fn_params
0294 06 40      ld      b, #64      ; fill 64 bytes

```

```

0296 3E FF      ld      a, #0xFF      ; fill byte
0298
0298          loc_0_298:
0298 77          ld      (hl), a      ; CODE XREF: 0000:029A|j
0299 23          inc     hl           ; set to $FF
029A 10 FC      djnz    loc_0_298    ; next location
029C 3E C0      ld      a, #0xC0 ; 'L' ; set 64 bytes
029E 32 B0 60   ld      (fg_fn_queue_tail), a ; init queue tail
02A1 32 B1 60   ld      (fg_fn_queue_head), a ; init queue head
02A4 AF        xor     a
02A5 32 83 7D   ld      (spritebank), a
02A8 32 86 7D   ld      (palette_bank), a ; b0=0
02AB 32 87 7D   ld      (palette_bank+1), a ; b1=0
02AE 3C        inc     a
02AF 32 82 7D   ld      (flipscreen), a
02B2 31 00 6C   ld      sp, #0x6C00
02B5 CD 1C 01   call   stop_sound
02B8 3E 01      ld      a, #1
02BA 32 84 7D   ld      (nmi_mask), a ; enable interrupts
02BD
02BD          main_loop:
02BD 26 60          ; CODE XREF: 0000:02D8|j
02BD          ; 0000:02E1|j
02BD          ; DATA XREF: ...
02BD          ; msb of queue
02BF 3A B1 60   ld      h, #0x60 ; ''
02C2 6F        ld      a, (fg_fn_queue_head) ; ptr head of queue
02C3 7E        ld      l, a
02C4 87        ld      a, (hl) ; get queue entry
02C5 30 1C      add     a, a ; empty?
02C7 CD 15 03   jr      NC, process_fg_fn_queue ; no, skip
02CA CD 50 03   call   flash_1UP_or_2UP
02CD 21 19 60   call   check_and_award_bonus
02D0 34        ld      hl, #random_no+1 ; random LSB
02D1 21 83 63   inc     (hl) ; INC
02D4 3A 1A 60   ld      hl, #unk_0_6383
02D7 BE        cp      (hl) ; same?
02D8 28 E3      jr      Z, main_loop ; yes, loop
02DA 77        ld      (hl), a ; generate LSB from timer
02DB CD 7F 03   call   difficulty_timer_tick
02DE CD A2 03   call   sub_0_3A2 ; fireball release
02E1 18 DA      jr      main_loop
02E3
02E3          ;
02E3          process_fg_fn_queue:
02E3          ; CODE XREF: 0000:02C5|j
02E3 E6 1F        and     #0x1F
02E5 5F        ld      e, a ; E=param1 (vector entry)
02E6 16 00      ld      d, #0 ; msb of vector table offset
02E8 36 FF      ld      (hl), #0xFF ; wipe param1
02EA 2C        inc     l
02EB 4E        ld      c, (hl) ; C=param2 (vector fn param)
02EC 36 FF      ld      (hl), #0xFF ; wipe param2
02EE 2C        inc     l
02EF 7D        ld      a, l ; new queue head
02F0 FE C0      cp      #0xC0 ; 'L' ; wrap?
02F2 30 02      jr      NC, loc_0_2F6 ; no, skip
02F4 3E C0      ld      a, #0xC0 ; 'L'
02F6
02F6          loc_0_2F6:
02F6          ; CODE XREF: 0000:02F2|j
02F6 32 B1 60   ld      (fg_fn_queue_head), a
02F9 79        ld      a, c ; vector fn param
02FA 21 BD 02   ld      hl, #main_loop
02FD E5        push    hl ; return address
02FE 21 07 03   ld      hl, #foreground_vector_table ; jump table
0301 19        add     hl, de ; entry index
0302 5E        ld      e, (hl)
0303 23        inc     hl
0304 56        ld      d, (hl) ; DE=vector address
0305 EB        ex      de, hl ; HL=vector address
0306 E9        jp      (hl) ; jump
0307
0307          foreground_vector_table: .dw add_bonus_and_update_high_score ; DATA XREF: 0000:02FE|o
0307          ; jump table
0309 9B 05      .dw zero_score_or_high_score
030B C6 05      .dw display_score_or_high_score
030D E9 05      .dw print_message_A
030F 11 06      .dw display_credits_if_attract_mode
0311 2A 06      .dw update_bonus_timer
0313 B8 06      .dw display_lives_and_level
0315
0315          ; SUBROUTINE
0315
0315          flash_1UP_or_2UP:
0315          ; CODE XREF: 0000:02C7|p
0315 3A 1A 60   ld      a, (gen_purpose_timer)
0318 47        ld      b, a ; save timer
0319 E6 0F      and     #0xF
031B C0        ret     NZ
031C CF        rst     8 ; return if attract mode
031D 3A 0D 60   ld      a, (current_player_D)
0320 CD 47 03   call   get_1UP_or_2UP_screen_location
0323 11 E0 FF   ld      de, #0xFFE0 ; column address offset
0326 CB 60      bit     4, b ; unhide 1UP/2UP?
0328 28 14      jr      Z, loc_0_33E ; yes, skip
032A 3E 10      ld      a, #0x10 ; " "
032C 77        ld      (hl), a ; wipe "1" or "2"
032D 19        add     hl, de ; next column
032E 77        ld      (hl), a ; wipe "U"
032F 19        add     hl, de ; next column
0330 77        ld      (hl), a ; wipe "P"
0331 3A 0F 60   ld      a, (two_players)
0334 A7        and     a ; 1 player?
0335 C8        ret     Z ; yes, return
0336 3A 0D 60   ld      a, (current_player_D)
0339 EE 01      xor     #1
033B CD 47 03   call   get_1UP_or_2UP_screen_location
033E
033E          loc_0_33E:
033E          ; CODE XREF: flash_1UP_or_2UP+13|j
033E 3C        inc     a ; "1" or "2"
033F 77        ld      (hl), a ; next column
0340 19        add     hl, de ; "U"
0341 36 25      ld      (hl), #0x25 ; '%' ; next column
0343 19        add     hl, de ; "P"
0344 36 20      ld      (hl), #0x20 ; ' '
0346 C9        ret
0346          ; End of function flash_1UP_or_2UP
0346

```

```

0347 ; SUBROUTINE
0347
0347 get_lUP_or_2UP_screen_location: ; CODE XREF: flash_lUP_or_2UP+B|p
0347 21 40 77 ; flash_lUP_or_2UP+26|p
0347 ld hl, #VRAM_start+0x340 ; ptr "lUP" screen location
0347 and a ; player 1?
0347 C8 ret Z ; yes, return
0347 21 E0 74 ld hl, #VRAM_start+0xE0 ; ptr "2UP" screen location
0347 C9 ret
0347 ; End of function get_lUP_or_2UP_screen_location
0347
0350 ; SUBROUTINE
0350
0350 check_and_award_bonus: ; CODE XREF: 0000:02CA|p
0350 3A 2D 62 ld a, (awarded_bonus_life)
0350 A7 and a ; already got bonus life?
0350 C0 ret NZ ; yes, return
0350 21 B3 60 ld hl, #p1_score+1
0350 3A 0D 60 ld a, (current_player_D)
0350 A7 and a ; player 1?
0350 28 03 jr Z, loc_0_361 ; yes, skip
0350 21 B6 60 ld hl, #p2_score+1
0361
0361 loc_0_361: ; CODE XREF: check_and_award_bonus+C|j
0361 7E ld a, (hl) ; get hundreds from score
0362 E6 F0 and #0xF0 ; only thousands
0364 47 ld b, a ; save
0365 23 inc hl ; next score byte
0366 7E ld a, (hl) ; get tens of thousands
0367 E6 0F and #0xF ; only tens of thousands
0369 B0 or b ; B = thousands (and tens of)
036A 0F rrca
036B 0F rrca
036C 0F rrca
036D 0F rrca ; swap nibbles
036E 21 21 60 ld hl, #bonus_setting
0371 BE cp (hl) ; reached bonus score?
0372 D8 ret C ; no, return
0373 3E 01 ld a, #1
0375 32 2D 62 ld (awarded_bonus_life), a ; flag that we've got the bonus
0378 21 28 62 ld hl, #lives_left
037B 34 inc (hl) ; extra life
037C C3 B8 06 jp display_lives_and_level
037C ; End of function check_and_award_bonus
037C
037F ; SUBROUTINE
037F
037F difficulty_timer_tick: ; CODE XREF: 0000:02DB|p
037F 21 84 63 ld hl, #unk_0_6384
0382 7E ld a, (hl) ; get LSB
0383 34 inc (hl) ; LSB tick
0384 A7 and a ; LSB overflow?
0385 C0 ret NZ ; no, return
0386 21 81 63 ld hl, #unk_0_6381
0389 7E ld a, (hl) ; get MSB
038A 47 ld b, a
038B 34 inc (hl) ; MSB tick
038C E6 07 and #7 ; expired?
038E C0 ret NZ ; no, return
038F 78 ld a, b
0390 0F rrca
0391 0F rrca
0392 0F rrca
0393 47 ld b, a
0394 3A 29 62 ld a, (level)
0397 80 add a, b ; adjust for level
0398 FE 05 cp #5 ; max?
039A 38 02 jr C, loc_0_39E ; no, skip
039C 3E 05 ld a, #5 ; set to max
039E
039E loc_0_39E: ; CODE XREF: difficulty_timer_tick+1B|j
039E 32 80 63 ld (unk_0_6380), a
03A1 C9 ret
03A1 ; End of function difficulty_timer_tick
03A1
03A2 ; SUBROUTINE
03A2
03A2 sub_0_3A2: ; CODE XREF: 0000:02DE|p
03A2 3E 03 ld a, #3
03A4 F7 rst 0x30 ; return if level bit not set
03A5 D7 rst 0x10 ; return if mario not alive
03A6 3A 50 63 ld a, (unk_0_6350)
03A9 0F rrca
03AA D8 ret C
03AB 21 B8 62 ld hl, #unk_0_62B8
03AE 35 dec (hl)
03AF C0 ret NZ
03B0 36 04 ld (hl), #4
03B2 3A B9 62 ld a, (unk_0_62B9)
03B5 0F rrca
03B6 D0 ret NC
03B7 21 29 6A ld hl, #soft_sprite_ram+0x129 ; sprite #173, flipy & code
03BA 06 40 ld b, #0x40 ; '@'
03BC DD 21 A0 66 ld ix, #unk_0_66A0
03C0 0F rrca
03C1 D2 E4 03 jp NC, loc_0_3E4
03C4 DD 36 09 02 ld 9(ix), #2
03C8 DD 36 0A 02 ld 0xA(ix), #2
03CC 04 inc b
03CD 04 inc b
03CE CD F2 03 call sub_0_3F2
03D1 21 BA 62 ld hl, #unk_0_62BA
03D4 35 dec (hl)
03D5 C0 ret NZ
03D6 3E 01 ld a, #1
03D8 32 B9 62 ld (unk_0_62B9), a
03DB 32 A0 63 ld (unk_0_63A0), a
03DE

```

```

03DE      loc_0_3DE:      ld      a, #0x10      ; CODE XREF: sub_0_3A2+4D↓j
03DE 3E 10      ld      (unk_0_62BA), a
03E0 32 BA 62      ret
03E3 C9
03E4
03E4      loc_0_3E4:      ld      9(ix), #2      ; CODE XREF: sub_0_3A2+1F↓j
03E4 DD 36 09 02      ld      0xA(ix), #0
03E8 DD 36 0A 00      call   sub_0_3F2
03EC CD F2 03      jp      loc_0_3DE
03EF C3 DE 03      ; End of function sub_0_3A2
03EF
03EF      ; ██████████ SUBROUTINE ██████████
03F2
03F2      sub_0_3F2:      ; CODE XREF: sub_0_3A2+2C↓p
03F2 70      ; sub_0_3A2+4A↓p
03F2      ld      (hl), b
03F3 3A 19 60      ld      a, (random_no+1)
03F6 0F      rrca
03F7 D8      ret      C
03F8 04      inc     b
03F9 70      ld      (hl), b
03FA C9      ret
03FA      ; End of function sub_0_3F2
03FA
03FB      ; ██████████ SUBROUTINE ██████████
03FB
03FB      animate_kong_and_pauline:      ; CODE XREF: 0000:19B0↓p
03FB 3A 27 62      ld      a, (level_type)
03FE FE 02      cp      #2      ; cement pies?
0400 C2 13 04      jp      NZ, loc_0_413      ; no, skip
0403 21 08 69      ld      hl, #soft_sprite_ram+8      ; sprite #2 y coord
0406 3A A3 63      ld      a, (unk_0_63A3)      ; get top conveyer speed/direction
0409 4F      ld      c, a      ; kong location adjustment
040A FF      rst      0x38      ; add +/-1 to y for 10 sprites
040B 3A 10 69      ld      a, (soft_sprite_ram+0x10)
040E D6 3B      sub     #59      ; sprite #4, y coord
0410 32 B7 63      ld      (unk_0_63B7), a
0413
0413      loc_0_413:      ; CODE XREF: animate_kong_and_pauline+5↓j
0413 3A 91 63      ld      a, (kong_thrash_flag)
0416 A7      and     a      ; thrashing arms?
0417 C2 26 04      jp      NZ, loc_0_426      ; yes, continue
041A 3A 1A 60      ld      a, (gen_purpose_timer)
041D A7      and     a      ; expired?
041E C2 86 04      jp      NZ, animate_pauline      ; no, animate Pauline
0421 3E 01      ld      a, #1      ; flag thrashing
0423 32 91 63      ld      (kong_thrash_flag), a
0426
0426      loc_0_426:      ; CODE XREF: animate_kong_and_pauline+1C↓j
0426 21 90 63      ld      hl, #kong_thrash_tmr
0429 34      inc     (hl)      ; inc
042A 7E      ld      a, (hl)      ; get timer
042B FE 80      cp      #128      ; finished thrashing?
042D CA 64 04      jp      Z, draw_kong_mouth_closed      ; yes, continue
0430 3A 93 63      ld      a, (barrel_deployment)
0433 A7      and     a      ; deployment in progress?
0434 C2 86 04      jp      NZ, animate_pauline      ; yes, skip (no thrashing)
0437 7E      ld      a, (hl)      ; get timer
0438 47      ld      b, a
0439 E6 1F      and     #31      ; time to thrash arms?
043B C2 86 04      jp      NZ, animate_pauline      ; no, skip (animate Pauline)
043E 21 CF 39      ld      hl, #dk_thrash_right_spr
0441 CB 68      bit     5, b      ; left/right depending on timer
0443 20 03      jr      NZ, do_kong_thrash
0445 21 F7 39      ld      hl, #dk_thrash_left_spr
0448
0448      do_kong_thrash:      ; CODE XREF: animate_kong_and_pauline+48↓j
0448 CD 4E 00      call   copy_sprites_2_11_data
044B 3E 03      ld      a, #3      ; tmr=3
044D 32 82 60      ld      (digital_snd_tmr_thump), a
0450
0450      loc_0_450:      ; CODE XREF: animate_kong_and_pauline+7A↓j
0450 3A 27 62      ld      a, (level_type)
0453 0F      rrca      ; level 2/4?
0454 D2 78 04      jp      NC, loc_0_478      ; yes, skip
0457 0F      rrca      ; level 3?
0458 DA 86 04      jp      C, animate_pauline      ; yes, skip
045B 21 0B 69      ld      hl, #soft_sprite_ram+0xB      ; sprite #2, x coord
045E 0E FC      ld      c, #0xFC ; '3'
0460 FF      rst      0x38      ; subtract 4 from x for 10 sprites
0461 C3 86 04      jp      animate_pauline
0464
0464      draw_kong_mouth_closed:      ; CODE XREF: animate_kong_and_pauline+32↓j
0464 AF      xor     a      ; zero kong_animation_tmr
0465 77      ld      (hl), a
0466 23      inc     hl
0467 77      ld      (hl), a
0468 3A 93 63      ld      a, (barrel_deployment)
046B A7      and     a      ; deployment in progress?
046C C2 86 04      jp      NZ, animate_pauline      ; no, continue
046F 21 5C 38      ld      hl, #dk_normal_spr
0472 CD 4E 00      call   copy_sprites_2_11_data
0475 C3 50 04      jp      loc_0_450
0478
0478      loc_0_478:      ; CODE XREF: animate_kong_and_pauline+59↓j
0478 21 08 69      ld      hl, #soft_sprite_ram+8      ; ptr sprite #2 (x coord)
047B 0E 44      ld      c, #0x44 ; 'D'
047D 0F      rrca      ; level 2?
047E D2 85 04      jp      NC, loc_0_485      ; yes, skip
0481 3A B7 63      ld      a, (unk_0_63B7)
0484 4F      ld      c, a
0485
0485      loc_0_485:      ; CODE XREF: animate_kong_and_pauline+83↓j
0485 FF      rst      0x38      ; add C to y coord of 10 sprites
0486
0486      animate_pauline:      ; CODE XREF: animate_kong_and_pauline+23↓j
0486 3A 90 63      ld      a, (kong_thrash_tmr)      ; animate_kong_and_pauline+39↓j ...

```



```

0489 4F          ld      c, a
048A 11 20 00     ld      de, #0x20 ; ' '
048D 3A 27 62     ld      a, (level_type)
0490 FE 04       cp      #4
0492 CA BE 04     jp      Z, display_help_rivets_level
0495 79          ld      a, c
0496 A7          and     a, c
0497 CA A1 04     jp      Z, wipe_help
049A 3E EF       ld      a, #0xEF ; ''
049C CB 71       bit     6, c
049E C2 A3 04     jp      NZ, display_or_wipe_help
04A1            ; rivets?
04A1            ; yes, skip
04A1            ; kong_thrash_tmr
04A1            ; finished?
04A1            ; yes, skip
04A1            ; "HELP!"
04A1            ; time to display help?
04A1            ; yes, skip
04A1            ; CODE XREF: animate_kong_and_pauline+9C↑j
04A1 3E 10       ld      a, #0x10
04A3            ; blank tiles
04A3            ; CODE XREF: animate_kong_and_pauline+A3↑j
04A3            ; screen position for HELP!
04A6 CD 14 05     call    display_3_tiles_HL
04A9 3A 05 69     ld      a, (soft_sprite_ram+5)
04AC            ; display/wipe HELP!
04AC            ; sprite #1, flipy & code
04AC            ; CODE XREF: animate_kong_and_pauline+F3↑j
04AC            ; animate_kong_and_pauline+10B↑j
04AC            ; sprite #1, flipy & code
04AF CB 71       ld      (soft_sprite_ram+5), a
04B1 C8          bit     6, c
04B2 47         ret     Z
04B3 79         ld      b, a
04B4 E6 07       ld      a, c
04B6 C0         and     #7
04B7 78         ret     NZ
04B8 EE 03       ld      a, b
04BA 32 05 69     xor     #3
04BD C9         ld      (soft_sprite_ram+5), a
04BE            ; sprite #1, flipy & code
04BE            ; toggle sprites 0x11/0x12 pauline running
04BE            ; sprite #1, flipy & code
04BE            ;
04BE            ; CODE XREF: animate_kong_and_pauline+97↑j
04BE            ; blank tiles
04BE            ; screen pos
04C0 21 23 76     ld      hl, #VRAM_start+0x223
04C3 CD 14 05     call    display_3_tiles_HL
04C6 21 83 75     ld      hl, #VRAM_start+0x183
04C9 CD 14 05     call    display_3_tiles_HL
04CC CB 71       bit     6, c
04CE CA 09 05     jp      Z, loc_0_509
04D1 3A 03 62     ld      a, (mario_y)
04D4 FE 80       cp      #0x80 ; 'Ç'
04D6 D2 F1 04     jp      NC, display_help_right
04D9 3E DF       ld      a, #0xDF ; '■'
04DB 21 23 76     ld      hl, #VRAM_start+0x223
04DE CD 14 05     call    display_3_tiles_HL
04E1            ; mario left/right side of screen?
04E1            ; right, skip
04E1            ; "HELP!" to the left
04E1            ; screen pos
04E1            ; display "HELP!"
04E1            ; CODE XREF: animate_kong_and_pauline+116↑j
04E1            ; sprite #0, flipy & code
04E4 F6 80       or      #0x80 ; 'Ç'
04E6 32 01 69     ld      (soft_sprite_ram+1), a
04E9 3A 05 69     ld      a, (soft_sprite_ram+5)
04EC F6 80       or      #0x80 ; 'Ç'
04EE C3 AC 04     jp      make_pauline_run
04F1            ; flipy
04F1            ; save
04F1            ; sprite #1, flipy & code
04F1            ; flipy
04F1            ; CODE XREF: animate_kong_and_pauline+DB↑j
04F1            ; "HELP!" to the right
04F3 21 83 75     ld      hl, #VRAM_start+0x183
04F6 CD 14 05     call    display_3_tiles_HL
04F9            ; screen pos
04F9            ; display "HELP!"
04F9            ; CODE XREF: animate_kong_and_pauline+113↑j
04F9            ; sprite #0, flipy & code
04FC E6 7F       and     #0x7F ; ' '
04FE 32 01 69     ld      (soft_sprite_ram+1), a
0501 3A 05 69     ld      a, (soft_sprite_ram+5)
0504 E6 7F       and     #0x7F ; ' '
0506 C3 AC 04     jp      make_pauline_run
0509            ; not flipped
0509            ; save
0509            ; sprite #1, flipy & code
0509            ; not flipped
0509            ; CODE XREF: animate_kong_and_pauline+D3↑j
0509            ; store tile
0509            ; next row/column
0509            ; prev tile
0509            ; loop for 3 tiles
0509            ; End of function animate_kong_and_pauline
0511            ;
0511            ; SUBROUTINE
0514            ;
0514            ; CODE XREF: animate_kong_and_pauline+AB↑p
0514            ; animate_kong_and_pauline+C8↑p ...
0514            ; 3 tiles
0514            ; CODE XREF: display_3_tiles_HL+5↑j
0516            ; store tile
0516            ; next row/column
0516            ; prev tile
0516            ; loop for 3 tiles
0516 77         ld      (hl), a
0517 19         add     hl, de
0518 3D         dec     a
0519 10 FB       djnz    loc_0_516
051B C9         ret
051B            ; End of function display_3_tiles_HL
051B            ;
051B            ; SUBROUTINE
051C            ;
051C            ; CODE XREF: 0000:0698↑p
051C            ; 0000:06A5↑j
051C            ; DATA XREF: ...
051C            ; return if attract mode
051C 4F         ld      c, a
051D CF         rst      8
051E CD 5F 05     call    current_player_score_DE
0521 79         ld      a, c
0522 81         add     a, c
0523 81         add     a, c
0524 4F         ld      c, a
0525 21 29 35     ld      hl, #bonus_points_tbl
0528 06 00       ld      b, #0
052A 09         add     hl, bc
052B A7         and     a, c
052C 06 03       ld      b, #3
052E            ; 3 bytes of score

```

```

052E      loc_0_52E:      ld      a, (de)      ; CODE XREF: add_bonus_and_update_high_score+18|j
052E 1A      adc      a, (hl)      ; get score BCD pair
052F 8E      daa      ; add bonus BCD pair
0530 27      daa      ; adjust for BCD
0531 12      ld      (de), a      ; update score BCD pair
0532 13      inc      de
0533 23      inc      hl      ; next byte
0534 10 F8   djnz     loc_0_52E      ; loop through score
0536 D5      push     de
0537 1B      dec      de      ; ptr score
0538 3A 0D 60 ld      a, (current_player_D)
053B CD 6B 05 call     display_player_A_score
053E D1      pop      de
053F 1B      dec      de
0540 21 BA 60 ld      hl, #high_score+2      ; MSB
0543 06 03   ld      b, #3      ; 3 bytes to compare
0545
0545      loc_0_545:      ld      a, (de)      ; CODE XREF: add_bonus_and_update_high_score+31|j
0546 BE      cp      (hl)      ; get byte from score
0547 D8      c      ; less than high score?
0548 C2 50 05 jp      NZ, new_high_score      ; yes, return
054B 1B      dec      de      ; greater, we have a high score
054C 2B      dec      hl
054D 10 F6   djnz     loc_0_545      ; same, check next byte
054F C9      ret      ; loop through 3 bytes
0550
0550
0550      new_high_score:      call     current_player_score_DE      ; CODE XREF: add_bonus_and_update_high_score+2C|j
0550 CD 5F 05   ld      hl, #high_score
0553 21 B8 60
0556
0556      update_high_score:      ld      a, (de)      ; CODE XREF: add_bonus_and_update_high_score+3E|j
0556 1A      ld      (hl), a      ; get score byte
0557 77      inc      de      ; copy to high score
0558 13      inc      hl      ; next location
0559 23      inc      hl      ; next location
055A 10 FA   djnz     update_high_score      ; loop through 3 bytes
055C C3 DA 05   jp      display_high_score
055C      ; End of function add_bonus_and_update_high_score
055C
055F      ;
055F      ; SUBROUTINE
055F
055F      current_player_score_DE:      ; CODE XREF: add_bonus_and_update_high_score+2|p
055F 11 B2 60      ; add_bonus_and_update_high_score+34|p
055F      ld      de, #p1_score
0562 3A 0D 60 ld      a, (current_player_D)
0565 A7      and      a      ; player one?
0566 C8      ret      Z      ; yes, return
0567 11 B5 60 ld      de, #p2_score
056A C9      ret
056A      ; End of function current_player_score_DE
056A
056B      ;
056B      ; SUBROUTINE
056B
056B      display_player_A_score:      ; CODE XREF: add_bonus_and_update_high_score+1F|p
056B DD 21 81 77      ; display_score_or_high_score+11|j
056B      ld      ix, #VRAM_start+0x381
056F A7      and      a
0570 28 0A   jr      Z, display_score_HL_at_IX
0572 DD 21 21 75 ld      ix, #VRAM_start+0x121
0576 18 04   jr      display_score_HL_at_IX
0578
0578
0578      display_score_at_hs_location:      ; CODE XREF: display_score_or_high_score+17|j
0578 DD 21 41 76      ; screen position for score
057C
057C      display_score_HL_at_IX:      ; CODE XREF: display_player_A_score+5|j
057C EB      ; display_player_A_score+B|j ...
057C      ex      de, hl
057D 11 E0 FF ld      de, #0xFFE0      ; column address delta
0580 01 04 03 ld      bc, #0x304      ; 3=6 digits
0583
0583      display_B_bcd_digit_pairs:      ; CODE XREF: display_player_A_score+25|j
0583      ld      a, (hl)      ; display_credits+11|j
0584 0F      rrca      ; get bcd digit pair
0585 0F      rrca
0586 0F      rrca
0587 0F      rrca      ; shift high nibble
0588 CD 93 05 call     display_score_digit
058B 7E      ld      a, (hl)      ; low nibble
058C CD 93 05 call     display_score_digit
058F 2B      dec      hl      ; next digit pair
0590 10 F1   djnz     display_B_bcd_digit_pairs      ; loop through 6 digits
0592 C9      ret
0592      ; End of function display_player_A_score
0592
0593      ;
0593      ; SUBROUTINE
0593
0593      display_score_digit:      ; CODE XREF: display_player_A_score+1D|p
0593 E6 0F      ; display_player_A_score+21|p
0593      and      #0xF      ; low nibble only
0595 DD 77 00 ld      0(ix), a      ; display digit
0598 DD 19      add      ix, de      ; next column
059A C9      ret
059A      ; End of function display_score_digit
059A
059B      ;
059B      ; SUBROUTINE
059B
059B      zero_score_or_high_score:      ; CODE XREF: zero_score_or_high_score+24|p
059B FE 03      ; DATA XREF: 0000:0309|o
059B      cp      #3      ; zero all scores?
059D D2 BD 05 jp      NC, loc_0_5BD      ; yes, skip
05A0 F5      push     af
05A1 21 B2 60 ld      hl, #p1_score
05A4 A7      and      a
05A5 CA AB 05 jp      Z, loc_0_5AB

```

```

05A8 21 B5 60      ld      hl, #p2_score
05AB
05AB      loc_0_5AB:
05AB      cp      #2          ; CODE XREF: zero_score_or_high_score+1|j
05AD C2 B3 05      jp      NZ, loc_0_5B3
05B0 21 B8 60      ld      hl, #high_score
05B3
05B3      loc_0_5B3:
05B3      xor      a          ; CODE XREF: zero_score_or_high_score+12|j
05B4 AF          ld      (hl), a
05B5 23          inc     hl
05B6 77          ld      (hl), a
05B7 23          inc     hl
05B8 77          ld      (hl), a
05B9 F1          pop     af
05BA C3 C6 05      jp      display_score_or_high_score
05BD
05BD      loc_0_5BD:
05BD      dec      a          ; CODE XREF: zero_score_or_high_score+2|j
05BD 3D          ; zero_score_or_high_score+29|j
05BE          push   af          ; next score to zero
05BF CD 9B 05      call   zero_score_or_high_score
05C2 F1          pop     af
05C3 C8          ret      Z          ; return when done
05C4 18 F7      jr      loc_0_5BD      ; zero next score
05C4      ; End of function zero_score_or_high_score
05C4
05C6      ; SUBROUTINE
05C6
05C6      display_score_or_high_score:
05C6 FE 03          ; CODE XREF: zero_score_or_high_score+1F|j
05C6          ; display_score_or_high_score+1C|p
05C6          ; DATA XREF: ...
05C6      cp      #3
05C8 CA E0 05      jp      Z, loc_0_5E0
05CB 11 B4 60      ld      de, #p1_score+2
05CE A7          and     a
05CF CA D5 05      jp      Z, loc_0_5D5
05D2 11 B7 60      ld      de, #p2_score+2
05D5
05D5      loc_0_5D5:
05D5 FE 02      cp      #2          ; CODE XREF: display_score_or_high_score+9|j
05D7 C2 6B 05      jp      NZ, display_player_A_score
05DA
05DA      display_high_score:
05DA 11 BA 60      ld      de, #high_score+2          ; CODE XREF: add_bonus_and_update_high_score+40|j
05DD C3 78 05      jp      display_score_at_hs_location
05E0
05E0      loc_0_5E0:
05E0 3D          ; CODE XREF: display_score_or_high_score+2|j
05E0      dec      a          ; display_score_or_high_score+21|j
05E1 F5          push   af
05E2 CD C6 05      call   display_score_or_high_score
05E5 F1          pop     af
05E6 C8          ret      Z
05E7 18 F7      jr      loc_0_5E0
05E7      ; End of function display_score_or_high_score
05E7
05E9      ; SUBROUTINE
05E9
05E9      print_message_A:
05E9          ; CODE XREF: display_credits+2|p
05E9 21 4B 36      ; display_start_1P_2P_get_selectio+18|p
05E9          ; DATA XREF: ...
05E9      ld      hl, #message_table
05EC 87          add     a, a          ; convert entry to offset
05ED F5          push   af
05EE E6 7F      and     #0x7F ; ' '          ; mask off 'wipe' bit
05F0 5F          ld      e, a
05F1 16 00      ld      d, #0          ; DE = offset
05F3 19          add     hl, de          ; pointer to entry
05F4 5E          ld      (hl), e
05F5 23          inc     hl
05F6 56          ld      d, (hl)          ; DE = entry (word)
05F7 EB          ex      de, hl
05F8 5E          ld      e, (hl)
05F9 23          inc     hl
05FA 56          ld      d, (hl)          ; DE = screen address to print
05FB 23          inc     hl          ; HL = message text
05FC 01 E0 FF      ld      bc, #0xFFE0      ; screen column address inc value
05FF EB          ex      de, hl          ; DE = text, HL = screen address
0600
0600      loc_0_600:
0600 1A          ld      a, (de)          ; CODE XREF: print_message_A+26|j
0601 FE 3F      cp      #0x3F ; '?'          ; get message character
0603 CA 26 00      jp      Z, pop_hl_ret      ; end of message?
0606 77          ld      (hl), a          ; yes, exit
0607 F1          pop     af          ; display character on screen
0608 30 02      jr      NC, loc_0_60C      ; restore original entry index
060A 36 10      ld      (hl), #0x10      ; not wiping, skip
060C          ; display space character on screen
060C      loc_0_60C:
060C F5          push   af          ; CODE XREF: print_message_A+1F|j
060D 13          inc     de          ; store original entry index
060E 09          add     hl, bc          ; next message character
060F 18 EF      jr      loc_0_600      ; next screen location
060F      ; End of function print_message_A
0611
0611      ;
0611      display_credits_if_attract_mode:
0611 3A 07 60      ld      a, (attract_mode_flag)          ; DATA XREF: 0000:030F|o
0614 0F          rrca          ; in attract mode?
0615 D0          ret      NC          ; no, return
0616
0616      ; SUBROUTINE
0616
0616      display_credits:
0616          ; CODE XREF: display_start_1P_2P_get_selectio+1B|p
0616 3E 05          ld      a, #5          ; 0000:141E|p ...
0616          ; "credit"
0618 CD E9 05      call   print_message_A

```

```

061B 21 01 60          ld      hl, #no_of_credits
061E 11 E0 FF          ld      de, #0xFFE0                ; column address delta
0621 DD 21 BF 74       ld      ix, #VRAM_start+0xBF        ; screen position of credits
0625 06 01             ld      b, #1                    ; 1=2 digits
0627 C3 83 05         jp      display_B_bcd_digit_pairs
0627                   ; End of function display_credits
0627
062A                   ;
062A
062A                   ; DATA XREF: 0000:0311|o
062A A7               and      a                        ; add bonus to score?
062B CA 91 06         jp      Z, loc_0_691              ; yes, skip
062E 3A 8C 63         ld      a, (bonus_timer)
0631 A7               and      a                        ; zero?
0632 C2 A8 06         jp      NZ, bonus_timer_tick      ; no, skip
0635 3A B8 63         ld      a, (bonus_timer_expired)
0638 A7               and      a                        ; expired?
0639 C0               ret      NZ                       ; yes, exit
063A 3A B0 62         ld      a, (bonus_timer_init_value) ; initialise bonus timer here
063D 01 0A 00         ld      bc, #0xA
0640
0640                   ; CODE XREF: 0000:0642|j
0640 04               inc      b
0641 91               sub      c
0642 C2 40 06         jp      NZ, loc_0_640
0645 78               ld      a, b
0646 07               rlca
0647 07               rlca
0648 07               rlca
0649 07               rlca
064A 32 8C 63         ld      (bonus_timer), a          ; set initial bonus timer value
064D 21 4A 38         ld      hl, #bonus_graphic_tiles
0650 11 65 74         ld      de, #VRAM_start+0x65      ; screen position for bonus
0653 3E 06             ld      a, #6                    ; 6 columns of tiles to display
0655
0655                   ; CODE XREF: 0000:0664|j
0655 DD 21 1D 00       ld      ix, #0x1D                ; column inc
0659 01 03 00         ld      bc, #3                    ; 3 tiles to display
065C ED B0             ldird ix, de                     ; display bonus tiles
065E DD 19             add     ix, de                     ; next column
0660 DD E5             push    ix
0662 D1               pop     de                         ; screen position
0663 3D               dec     a                          ; done?
0664 C2 55 06         jp      NZ, loc_0_655              ; no, loop
0667 3A 8C 63         ld      a, (bonus_timer)
066A
066A                   ; CODE XREF: 0000:06B5|j
066A 4F               ld      c, a
066B E6 0F             and     #0xF
066D 47               ld      b, a                      ; B=low nibble
066E 79               ld      a, c
066F 0F               rrca
0670 0F               rrca
0671 0F               rrca
0672 0F               rrca
0673 E6 0F             and     #0xF                      ; C=high nibble
0675 C2 89 06         jp      NZ, display_bonus_digits ; skip if more than 9s left
0678 3E 03             ld      a, #3
067A 32 89 60         ld      (bg_music), a
067D 3E 70             ld      a, #0x70 ; 'p'
067F 32 86 74         ld      (VRAM_start+0x86), a      ; purple '0'
0682 32 A6 74         ld      (VRAM_start+0xA6), a      ; '0'
0685 80               add     a, b                      ; 2nd digit to 'ascii'
0686 47               ld      b, a                      ; store
0687 3E 10             ld      a, #0x10                 ; <space>
0689
0689                   ; CODE XREF: 0000:0675|j
0689 32 E6 74         ld      (VRAM_start+0xE6), a      ; display 1st digit
068C 78               ld      a, b                      ; restore 2nd digit
068D 32 C6 74         ld      (VRAM_start+0xC6), a      ; display 2nd digit
0690 C9               ret
0691
0691                   ;
0691
0691                   ; CODE XREF: 0000:062B|j
0691 3A 8C 63         ld      a, (bonus_timer)
0694 47               ld      b, a
0695 E6 0F             and     #0xF
0697 C5               push    bc
0698 CD 1C 05         call    add_bonus_and_update_high_score
069B C1               pop     bc
069C 78               ld      a, b
069D 0F               rrca
069E 0F               rrca
069F 0F               rrca
06A0 0F               rrca
06A1 E6 0F             and     #0xF
06A3 C6 0A         add     a, #0xA
06A5 C3 1C 05         jp      add_bonus_and_update_high_score
06A8
06A8                   ;
06A8
06A8                   ; CODE XREF: 0000:0632|j
06A8 D6 01             sub     #1
06AA 20 05             jr      NZ, loc_0_6B1
06AC 21 B8 63         ld      hl, #bonus_timer_expired
06AF 36 01             ld      (hl), #1
06B1
06B1                   ; CODE XREF: 0000:06AA|j
06B1 27               daa
06B2 32 8C 63         ld      (bonus_timer), a
06B5 C3 6A 06         jp      display_bonus_timer
06B8
06B8                   ; SUBROUTINE
06B8
06B8                   ; CODE XREF: 0000:01DC|p
06B8 4F               ld      c, a                      ; check_and_award_bonus+2C|j
06B8                   ; DATA XREF: ...
06B8 CF               rst      8                        ; store alive flag
06BA 06 06             ld      b, #6                    ; return if attract mode
06BC 11 E0 FF         ld      de, #0xFFE0                ; max icons
06BF 21 83 77         ld      hl, #VRAM_start+0x383        ; column delta
06C2
06C2                   ; CODE XREF: display_lives_and_level+D|j
06C2 36 10             ld      (hl), #0x10                ; <space>
06C4 19               add     hl, de                     ; next column

```

```

06C5 10 FB      djnz    loc_0_6C2                ; wipe 6 icons
06C7 3A 28 62   ld      a, (lives_left)
06CA 91         sub      c                    ; decrement if mario alive
06CB CA D7 06   jp      z, loc_0_6D7                ; none to display, skip
06CE 47         ld      b, a                    ; number of lives
06CF 21 83 77   ld      hl, #VRAM_start+0x383    ; screen location
06D2           ;
06D2           ; CODE XREF: display_lives_and_level+1D|j
06D2 36 FF      ld      (hl), #0xFF              ; mario icon
06D4 19         add      hl, de                  ; next screen location
06D5 10 FB      djnz    loc_0_6D2                ; loop for no. of lives
06D7           ;
06D7           ; CODE XREF: display_lives_and_level+13|j
06D7 21 03 75   ld      hl, #VRAM_start+0x103
06DA 36 1C      ld      (hl), #0x1C              ; 'L'
06DC 21 E3 74   ld      hl, #VRAM_start+0xE3
06DF 36 34      ld      (hl), #0x34 ; '4'
06E1 3A 29 62   ld      a, (level)
06E4 FE 64      cp      #100                    ; too high?
06E6 38 05      jr      c, loc_0_6ED            ; no, skip
06E8 3E 63      ld      a, #99                    ; max out at 99
06EA 32 29 62   ld      (level), a                    ; adjust
06ED           ;
06ED           ; CODE XREF: display_lives_and_level+2E|j
06ED 01 0A FF   ld      bc, #0xFF0A
06F0           ;
06F0           ; CODE XREF: display_lives_and_level+3A|j
06F0 04         inc      b
06F1 91         sub      c
06F2 D2 F0 06   jp      NC, loc_0_6F0
06F5 81         add      a, c                    ; level tens digit
06F6 32 A3 74   ld      (VRAM_start+0xA3), a
06F9 78         ld      a, b                    ; level units digit
06FA 32 C3 74   ld      (VRAM_start+0xC3), a
06FD C9         ret
06FD           ; End of function display_lives_and_level
06FE           ;
06FE           ;
06FE           ;
06FE           ; DATA XREF: 0000:00D0|o
06FE 3A 0A 60   ld      a, (main_sequencer)
0701 EF         rst      0x28                    ; go!
0701           ;
0701           ; dw cls_and_set_screen_flip                ; Jump table
0702 86 09      .dw init_P1_ingame_data
0704 AB 09      .dw display_player_I_and_2P_score
0706 D6 09      .dw init_P2_ingame_data
0708 FE 09      .dw display_player_II_2UP_and_2P_sco
070A 1B 0A      .dw display_1UP_and_high_score
070C 37 0A      .dw wait_cls_and_check_seen_intro
070E 63 0A      .dw vector_on_intro_sequence
0710 76 0A      .dw draw_how_high_can_you_get
0712 DA 0B      .dw 0
0714 00 00      .dw wait_init_and_draw_level
0716 91 0C      .dw init_mario
0718 3C 12      .dw gameplay
071A 7A 19      .dw died_in_gameplay
071C 7C 12      .dw save_P1_ingame_data
071E F2 12      .dw save_P2_ingame_data
0720 44 13      .dw p1_game_over
0722 8F 13      .dw p2_game_over
0724 A1 13      .dw set_flip_and_current_P2
0726 AA 13      .dw set_flip_and_current_P1
0728 BB 13      .dw draw_name_registered
072A 1E 14      .dw do_initials_entry
072C 86 14      .dw mario_pauline_reunion
072E 15 16      .dw cls_and_set_seq_for_current_play
0730 6B 19      .dw 0
0732 00 00      .dw 0
0734 00 00      .dw 0
0736 00 00      .dw 0
0738 00 00      .dw 0
073A 00 00      .dw 0
073C           ;
073C           ;
073C           ; DATA XREF: 0000:00CC|o
073C 21 0A 60   ld      hl, #main_sequencer
073F 3A 01 60   ld      a, (no_of_credits)
0742 A7         and      a                    ; any credits?
0743 C2 5C 07   jp      NZ, inc_nmi_sequencer          ; yes, skip
0746 7E         ld      a, (hl)
0747 EF         rst      0x28                    ; go!
0747           ;
0747           ; dw insert_coin_screen                ; Jump Table (attract sequencer)
0748 79 07      .dw init_attract_mode_and_draw_level
074A 63 07      .dw init_mario
074C 3C 12      .dw attract_mode_gameplay
074E 77 19      .dw died_in_gameplay
0750 7C 12      .dw cls_and_next_sequence
0752 C3 07      .dw title_screen_flash
0754 CB 07      .dw title_screen_no_flash
0756 4B 08      .dw 0
0758 00 00      .dw 0
075A 00 00      .dw 0
075C           ;
075C           ;
075C           ; CODE XREF: 0000:0743|j
075C           ; reset game sequencer
075C 36 00      ld      (hl), #0
075E 21 05 60   ld      hl, #nmi_sequencer
0761 34         inc      (hl)                    ; inc nmi_sequencer
0762 C9         ret
0763           ;
0763           ;
0763           ; DATA XREF: 0000:074A|o
0763           ; wait for 16-bit countdown
0763 E7         rst      0x20
0764 AF         xor      a
0765 32 92 63   ld      (unk_0_6392), a
0768 32 A0 63   ld      (unk_0_63A0), a
076B 3E 01      ld      a, #1
076D 32 27 62   ld      (level_type), a
0770 32 29 62   ld      (level), a
0773 32 28 62   ld      (lives_left), a
0776 C3 92 0C   jp      init_and_draw_level
0779           ;
0779           ; DATA XREF: 0000:0748|o
0779           ;
0779 21 86 7D      ld      hl, #palette_bank
077C 36 00      ld      (hl), #0

```

```

077E 23          inc     hl
077F 36 00      ld      (hl), #0
0781 11 1B 03   ld      de, #0x31B
0784 CD 9F 30   call    queue_fg_vector_fn
0787 1C          inc     e
0788 CD 9F 30   call    queue_fg_vector_fn
0788 CD 65 09   call    queue_hs_table_for_display
078E 21 09 60   ld      hl, #eight_bit_countdown
0791 36 02      ld      (hl), #2
0793 23          inc     hl
0794 34          inc     (hl)
0795 CD 74 08   call    clear_visible_area_and_sprites
0798 CD 53 0A   call    display_lUP
079B 3A 0F 60   ld      a, (two_players)
079E FE 01      cp      #1
07A0 CC EE 09   call    Z, display_2UP
07A3 ED 5B 22 60 ld      de, (coinage)
07A7 21 6C 75   ld      hl, #VRAM_start+0x16C
07AA CD AD 07   call    display_coinage
07AD
07AD          display_coinage:
07AD 73          ld      (hl), e
07AE 23          inc     hl
07AF 23          inc     hl
07B0 72          ld      (hl), d
07B1 7A          ld      a, d
07B2 D6 0A      sub     #0xA
07B4 C2 BC 07   jp      NZ, loc_0_7BC
07B7 77          ld      (hl), a
07B8 3C          inc     a
07B9 32 8E 75   ld      (VRAM_start+0x18E), a
07BC
07BC          loc_0_7BC:
07BC 11 01 02   ld      de, #0x201
07BF 21 8C 76   ld      hl, #VRAM_start+0x28C
07C2 C9          ret
07C3
07C3          ;
07C3
07C3          cls_and_next_sequence:
07C3 CD 74 08   call    clear_visible_area_and_sprites
07C6 21 0A 60   ld      hl, #main_sequencer
07C9 34          inc     (hl)
07CA C9          ret
07CB
07CB          ;
07CB
07CB          title_screen_flash:
07CB 3A 8A 63   ld      a, (title_flash_tmr_1)
07CE FE 00      cp      #0
07D0 C2 2D 08   jp      NZ, loc_0_82D
07D3 3E 60      ld      a, #0x60
07D5 32 8A 63   ld      (title_flash_tmr_1), a
07D8 0E 5F      ld      c, #0x5F
07DA
07DA          loc_0_7DA:
07DA FE 00      cp      #0
07DC CA 3B 08   jp      Z, loc_0_83B
07DF 21 86 7D   ld      hl, #palette_bank
07E2 36 00      ld      (hl), #0
07E4 79          ld      a, c
07E5 CB 07      rlc     a
07E7 30 02      jr      NC, loc_0_7EB
07E9 36 01      ld      (hl), #1
07EB
07EB          loc_0_7EB:
07EB 23          inc     hl
07EC 36 00      ld      (hl), #0
07EE CB 07      rlc     a
07F0 30 02      jr      NC, loc_0_7F4
07F2 36 01      ld      (hl), #1
07F4
07F4          loc_0_7F4:
07F4 32 8B 63   ld      (title_flash_tmr_2), a
07F7 21 08 3D   ld      hl, #title_screen
07FA
07FA          display_donkey_kong_title:
07FA 3E B0      ld      a, #0xB0
07FC 46          ld      b, (hl)
07FD 23          inc     hl
07FE 5E          ld      e, (hl)
07FF 23          inc     hl
0800 56          ld      d, (hl)
0801
0801          loc_0_801:
0801 12          ld      (de), a
0802 13          inc     de
0803 10 FC      djnz   loc_0_801
0805 23          inc     hl
0806 7E          ld      a, (hl)
0807 FE 00      cp      #0
0809 C2 FA 07   jp      NZ, display_donkey_kong_title
080C 11 1E 03   ld      de, #0x31E
080F CD 9F 30   call    queue_fg_vector_fn
0812 13          inc     de
0813 CD 9F 30   call    queue_fg_vector_fn
0816 21 CF 39   ld      hl, #dk_thrash_right_spr
0819 CD 4E 00   call    copy_sprites_2_11_data
081C CD 24 3F   call    display_tm
081F 00          nop
0820 21 08 69   ld      hl, #soft_sprite_ram+8
0823 0E 44      ld      c, #68
0825 FF          rst     0x38
0826 21 0B 69   ld      hl, #soft_sprite_ram+0xB
0829 0E 78      ld      c, #120
082B FF          rst     0x38
082C C9          ret
082D
082D          ;
082D
082D          loc_0_82D:
082D 3A 8B 63   ld      a, (title_flash_tmr_2)
0830 4F          ld      c, a
0831 3A 8A 63   ld      a, (title_flash_tmr_1)
0834 3D          dec     a
0835 32 8A 63   ld      (title_flash_tmr_1), a
0838 C3 DA 07   jp      loc_0_7DA
083B
083B          ;
083B

```

```

083B      loc_0_83B:      ld      hl, #eight_bit_countdown      ; CODE XREF: 0000:07DC|j
083B 21 09 60      ld      (hl), #2
083E 36 02      inc     hl
0840 23      inc     hl      ; game_sequencer
0841 34      inc     (hl)
0842 21 8A 63      ld      hl, #title_flash_tmr_1
0845 36 00      ld      (hl), #0
0847 23      inc     hl
0848 36 00      ld      (hl), #0
084A C9      ret
084B
084B
084B      title_screen_no_flash:      ; DATA XREF: 0000:0756|o
084B E7      rst      0x20      ; wait for 16-bit countdown
084C 21 0A 60      ld      hl, #main_sequencer
084F 36 00      ld      (hl), #0      ; reset game sequencer
0851 C9      ret
0852
0852      ; ██████████ S U B R O U T I N E ██████████
0852
0852      clear_tiles_and_sprites:      ; CODE XREF: 0000:0986|p
0852 21 00 74      ; 0000:196B|p
0852      ld      hl, #VRAM_start
0855 0E 04      ld      c, #4      ; 4x256 bytes to clear
0857
0857      loc_0_857:      ; CODE XREF: clear_tiles_and_sprites+E|j
0857 06 00      ld      b, #0      ; 256 bytes to clear
0859 3E 10      ld      a, #0x10      ; space character
085B
085B      loc_0_85B:      ; CODE XREF: clear_tiles_and_sprites+B|j
085B 77      ld      (hl), a      ; display space
085C 23      inc     hl
085D 10 FC      djnz   loc_0_85B      ; clear 256 bytes
085F 0D      dec     c
0860 C2 57 08      jp     NZ, loc_0_857      ; do 1024 bytes
0863 21 00 69      ld      hl, #soft_sprite_ram
0866 0E 02      ld      c, #2      ; 2x192 bytes to clear
0868
0868      loc_0_868:      ; CODE XREF: clear_tiles_and_sprites+1E|j
0868 06 C0      ld      b, #192      ; 192 bytes to clear
086A AF      xor     a
086B
086B      loc_0_86B:      ; CODE XREF: clear_tiles_and_sprites+1B|j
086B 77      ld      (hl), a      ; clear soft sprite ram byte
086C 23      inc     hl      ; next address
086D 10 FC      djnz   loc_0_86B      ; clear 192 bytes
086F 0D      dec     c
0870 C2 68 08      jp     NZ, loc_0_868      ; clear 384 bytes
0873 C9      ret
0873      ; End of function clear_tiles_and_sprites
0873
0873      ; ██████████ S U B R O U T I N E ██████████
0873
0873      clear_visible_area_and_sprites:      ; CODE XREF: 0000:01C3|p
0873 21 04 74      ; 0000:0795|p ...
0873      ld      hl, #VRAM_start+4
0877 0E 20      ld      c, #32      ; 32 columns
0879
0879      loc_0_879:      ; CODE XREF: clear_visible_area_and_sprites+12|j
0879 06 1C      ld      b, #28      ; 28 rows
087B 3E 10      ld      a, #0x10      ; <space>
087D 11 04 00      ld      de, #4      ; bottm-to-top next column increment
0880
0880      loc_0_880:      ; CODE XREF: clear_visible_area_and_sprites+E|j
0880 77      ld      (hl), a      ; display space character
0881 23      inc     hl      ; next line
0882 10 FC      djnz   loc_0_880      ; loop screen height
0884 19      add     hl, de      ; next column
0885 0D      dec     c      ; done all columns?
0886 C2 79 08      jp     NZ, loc_0_879      ; no, loop
0889 21 22 75      ld      hl, #VRAM_start+0x122
088C 11 20 00      ld      de, #32
088F 0E 02      ld      c, #2
0891 3E 10      ld      a, #0x10      ; <space>
0893
0893      loc_0_893:      ; CODE XREF: clear_visible_area_and_sprites+29|j
0893 06 0E      ld      b, #14      ; 14 columns
0895
0895      loc_0_895:      ; CODE XREF: clear_visible_area_and_sprites+23|j
0895 77      ld      (hl), a      ; display space character
0896 19      add     hl, de      ; next column
0897 10 FC      djnz   loc_0_895      ; loop for 14 columns
0899 21 23 75      ld      hl, #VRAM_start+0x123
089C 0D      dec     c
089D C2 93 08      jp     NZ, loc_0_893      ; repeat at new location
08A0 21 00 69      ld      hl, #soft_sprite_ram
08A3 06 00      ld      b, #0      ; 256 bytes to clear
08A5 3E 00      ld      a, #0      ; clear to 0x00
08A7
08A7      loc_0_8A7:      ; CODE XREF: clear_visible_area_and_sprites+35|j
08A7 77      ld      (hl), a      ; clear soft sprite ram byte
08A8 23      inc     hl      ; next location
08A9 10 FC      djnz   loc_0_8A7      ; do 256 bytes
08AB 06 80      ld      b, #128      ; 128 bytes to clear
08AD
08AD      loc_0_8AD:      ; CODE XREF: clear_visible_area_and_sprites+3B|j
08AD 77      ld      (hl), a      ; clear soft sprite ram byte
08AE 23      inc     hl      ; next location
08AF 10 FC      djnz   loc_0_8AD      ; clear 128 bytes
08B1 C9      ret
08B1      ; End of function clear_visible_area_and_sprites
08B1
08B1      ; ██████████
08B2
08B2      vector_on_credit_sequencer:      ; DATA XREF: 0000:00CE|o
08B2 3A 0A 60      ld      a, (main_sequencer)
08B5 EF      rst      0x28      ; go!
08B5
08B5      ; ██████████
08B6 BA 08      .dw     display_1P_2P_start_screen      ; jump table
08B8 F8 08      .dw     process_1P_2P_start
08BA
08BA      ; ██████████
08BA      display_1P_2P_start_screen:      ; DATA XREF: 0000:08B6|o

```

```

08BA CD 74 08      call    clear_visible_area_and_sprites
08BD AF           xor      a
08BE 32 07 60      ld        (attract_mode_flag), a      ; clear attract mode flag
08C1 11 0C 03      ld        de, #0x30C                      ; print_message_0C
08C4 CD 9F 30      call    queue_fg_vector_fn
08C7 21 0A 60      ld        hl, #main_sequencer
08CA 34           inc        (hl)
08CB CD 65 09      call    queue_hs_table_for_display
08CE AF           xor      a
08CF 21 86 7D      ld        hl, #palette_bank
08D2 77           ld        (hl), a
08D3 2C           inc        l
08D4 77           ld        (hl), a      ; palette bank 0
08D5
08D5 ; ██████████ S U B R O U T I N E ██████████
08D5
08D5 display_start_1P_2P_get_selectio:      ; CODE XREF: 0000:08F8|p
08D5         ld        b, #4      ; mask for START1
08D7 1E 09         ld        e, #return_if_attract_mode+1 ; "ONLY 1 PLAYER BUTTON"
08D9 3A 01 60      ld        a, (no_of_credits)
08DC FE 01         cp        #1
08DE CA E4 08      jp        Z, loc_0_8E4
08E1 06 0C         ld        b, #0xC      ; mask for START1/START2
08E3 1C           inc        e      ; "1 or 2 PLAYERS"
08E4
08E4 loc_0_8E4:      ; CODE XREF: display_start_1P_2P_get_selectio+9|j
08E4         ld        a, (gen_purpose_timer)
08E7 E6 07         and        #7
08E9 C2 F3 08      jp        NZ, loc_0_8F3
08EC 7B           ld        a, e      ; message 9/10
08ED CD E9 05      call    print_message_A
08F0 CD 16 06      call    display_credits
08F3
08F3 loc_0_8F3:      ; CODE XREF: display_start_1P_2P_get_selectio+14|j
08F3         ld        a, (in2_snd_latch)
08F6 A0           and        b
08F7 C9           ret      ; only START1/START2
08F7 ; End of function display_start_1P_2P_get_selectio
08F7
08F7 ;
08F7
08F7 process_1P_2P_start:      ; DATA XREF: 0000:08B8|o
08F8         call    display_start_1P_2P_get_selectio
08FB FE 04         cp        #4      ; START1?
08FD CA 06 09      jp        Z, start_1_selected ; yes, skip
0900 FE 08         cp        #8      ; START2?
0902 CA 19 09      jp        Z, start_2_selected ; yes, skip
0905 C9           ret
0906
0906 start_1_selected:      ; CODE XREF: 0000:08FD|j
0906         call    dec_credits_and_display
0909 21 48 60      ld        hl, #p2_ingame_data
090C 06 08         ld        b, #8
090E AF           xor      a
090F
090F loc_0_90F:      ; CODE XREF: 0000:0911|j
090F         ld        (hl), a
0910 2C           inc        l
0911 10 FC         djnz    loc_0_90F
0913 21 00 00      ld        hl, #0
0916 C3 38 09      jp        start_game
0919
0919 start_2_selected:      ; CODE XREF: 0000:0902|j
0919         call    dec_credits_and_display
091C CD 77 09      call    dec_credits_and_display
091F 11 48 60      ld        de, #p2_ingame_data
0922 3A 20 60      ld        a, (lives_per_game)
0925 12           ld        (de), a
0926 1C           inc        e
0927 21 5E 09      ld        hl, #game_init_data
092A 01 07 00      ld        bc, #7
092D ED B0         ldir
092F 11 01 01      ld        de, #0x101      ; zero_score_or_high_score
0932 CD 9F 30      call    queue_fg_vector_fn
0935 21 00 01      ld        hl, #0x100      ; players=2, current_player=1
0938
0938 start_game:      ; CODE XREF: 0000:0916|j
0938         ld        (current_player_E), hl ; players and current player
093B CD 74 08      call    clear_visible_area_and_sprites
093E 11 40 60      ld        de, #p1_ingame_data
0941 3A 20 60      ld        a, (lives_per_game)
0944 12           ld        (de), a
0945 1C           inc        e
0946 21 5E 09      ld        hl, #game_init_data
0949 01 07 00      ld        bc, #7      ; 7 bytes
094C ED B0         ldir
094E 11 00 01      ld        de, #0x100      ; zero_score_or_high_score
0951 CD 9F 30      call    queue_fg_vector_fn
0954 AF           xor      a
0955 32 0A 60      ld        (main_sequencer), a
0958 3E 03         ld        a, #3
095A 32 05 60      ld        (nmi_sequencer), a
095D C9           ret
095D
095D game_init_data: .db 1      ; DATA XREF: 0000:0927|o
095E           ; 0000:0946|o
095E           ; Start of game level init data
095F 65 3A         .dw level_seq_1
0961 01 00 00 00 .db 1, 0, 0, 0
0965
0965 ; ██████████ S U B R O U T I N E ██████████
0965
0965 queue_hs_table_for_display:      ; CODE XREF: 0000:078B|p
0965         ld        de, #0x400      ; 0000:08CB|p
0965         call    queue_fg_vector_fn ; display_credits_if_attract_mode
0968 CD 9F 30      ld        de, #0x314      ; print_message_14 (1st high score)
096B 11 14 03      ld        b, #6      ; 1-5 and "RANK SCORE NAME"
096E 06 06
0970
0970 loc_0_970:      ; CODE XREF: queue_hs_table_for_display+F|j
0970         call    queue_fg_vector_fn
0973 1C           inc        e      ; next msg

```



```

0974 10 FA      djnz     loc_0_970      ; loop through messages
0976 C9      ret
0976      ; End of function queue_hs_table_for_display
0976
0977
0977      ; SUBROUTINE
0977
0977      dec_credits_and_display:      ; CODE XREF: 0000:0906|p
0977 21 01 60      ; 0000:0919|p ...
0977      ld      hl, #no_of_credits
0977      ld      a, #0x99 ; 'Ö'
0977      add     a, (hl)
0977      daa     ; decrement credits
0977      ld      (hl), a      ; save
0977 11 00 04      ld      de, #0x400      ; display_credits_if_attract_mode
0982 CD 9F 30      call    queue_fg_vector_fn
0985 C9      ret
0985      ; End of function dec_credits_and_display
0985
0986
0986      ;
0986
0986      cls_and_set_screen_flip:      ; DATA XREF: 0000:0702|o
0986 CD 52 08      call    clear_tiles_and_sprites
0989 CD 1C 01      call    stop_sound
098C 11 82 7D      ld      de, #flipscreen
098F 3E 01      ld      a, #1      ; default flipscreen
0991 12      ld      (de), a
0992 21 0A 60      ld      hl, #main_sequencer
0995 3A 0E 60      ld      a, (current_player_E)
0998 A7      and     a      ; player 2?
0999 C2 9F 09      jp      NZ, loc_0_99F      ; yes, skip
099C 36 01      ld      (hl), #1      ; ingame sequencer = 1
099E C9      ret
099F
099F
099F      loc_0_99F:      ; CODE XREF: 0000:0999|j
099F 3A 26 60      ld      a, (upright)      ; get cabinet type
09A2 3D      dec     a      ; upright?
09A3 CA A8 09      jp      Z, loc_0_9A8      ; yes, skip
09A6 AF      xor     a      ; disable flipscreen
09A7 12      ld      (de), a      ; to hardware
09A8
09A8      loc_0_9A8:      ; CODE XREF: 0000:09A3|j
09A8 36 03      ld      (hl), #3      ; ingame sequencer = 3
09AA C9      ret
09AB
09AB
09AB      init_P1_ingame_data:      ; DATA XREF: 0000:0704|o
09AB 21 40 60      ld      hl, #p1_ingame_data
09AE 11 28 62      ld      de, #lives_left      ; player_current_data
09B1 01 08 00      ld      bc, #8      ; 8 bytes to copy
09B4 ED B0      ldir
09B6 2A 2A 62      ld      hl, (seq_data)      ; ptr current sequence table
09B9 7E      ld      a, (hl)      ; get level type
09BA 32 27 62      ld      (level_type), a      ; store as current
09BD 3A 0F 60      ld      a, (two_players)
09C0 A7      and     a      ; 1 player?
09C1 21 09 60      ld      hl, #eight_bit_countdown
09C4 11 0A 60      ld      de, #main_sequencer
09C7 CA D0 09      jp      Z, loc_0_9D0      ; yes, skip
09CA 36 78      ld      (hl), #0x78 ; 'x'      ; set 8-bit countdown
09CC EB      ex      de, hl
09CD 36 02      ld      (hl), #2      ; next sequence (2)
09CF C9      ret
09D0
09D0
09D0      loc_0_9D0:      ; CODE XREF: 0000:09C7|j
09D0 36 01      ld      (hl), #1      ; set 8-bit countdown
09D2 EB      ex      de, hl
09D3 36 05      ld      (hl), #5      ; next sequence (5)
09D5 C9      ret
09D6
09D6
09D6      display_player_I_and_2P_score:      ; DATA XREF: 0000:0706|o
09D6 AF      xor     a
09D7 32 86 7D      ld      (palette_bank), a
09DA 32 87 7D      ld      (palette_bank+1), a      ; palette bank 0
09DD 11 02 03      ld      de, #0x302      ; display_message_02 "PLAYER (I)"
09E0 CD 9F 30      call    queue_fg_vector_fn
09E3 11 01 02      ld      de, #0x201      ; display_score_or_high_score (P2)
09E6 CD 9F 30      call    queue_fg_vector_fn
09E9 3E 05      ld      a, #5
09EB 32 0A 60      ld      (main_sequencer), a
09EE
09EE      ; SUBROUTINE
09EE
09EE      display_2UP:      ; CODE XREF: 0000:07A0|p
09EE 3E 02      ; 0000:0A2E|p
09EE      ; '2'
09F0 32 E0 74      ld      a, #2      (VRAM_start+0xE0), a
09F3 3E 25      ld      a, #0x25 ; '%'      ; 'U'
09F5 32 C0 74      ld      (VRAM_start+0xC0), a
09F8 3E 20      ld      a, #0x20 ; ' '      ; 'P'
09FA 32 A0 74      ld      (VRAM_start+0xA0), a
09FD C9      ret
09FD      ; End of function display_2UP
09FE
09FE
09FE      init_P2_ingame_data:      ; DATA XREF: 0000:0708|o
09FE 21 48 60      ld      hl, #p2_ingame_data
09A0 11 28 62      ld      de, #lives_left      ; player_current_data
09A4 01 08 00      ld      bc, #8      ; 8 bytes to copy
09A7 ED B0      ldir
09A9 2A 2A 62      ld      hl, (seq_data)      ; ptr current seq table
09AC 7E      ld      a, (hl)      ; get level type
09AD 32 27 62      ld      (level_type), a      ; store as current
09A0 3E 78      ld      a, #0x78 ; 'x'      ; init 8-bit countdown
09A2 32 09 60      ld      (eight_bit_countdown), a
09A5 3E 04      ld      a, #4      ; next sequence (4)
09A7 32 0A 60      ld      (main_sequencer), a
09A9 C9      ret
09AB
09AB

```

```

0A1B      display_player_II_2UP_and_2P_sco:                ; DATA XREF: 0000:070A|o
0A1B AF      xor      a
0A1C 32 86 7D      ld      (palette_bank), a
0A1F 32 87 7D      ld      (palette_bank+1), a
0A22 11 03 03      ld      de, #0x303
0A25 CD 9F 30      call   queue_fg_vector_fn
0A28 11 01 02      ld      de, #0x201
0A2B CD 9F 30      call   queue_fg_vector_fn
0A2E CD EE 09      call   display_2UP
0A31 3E 05      ld      a, #5
0A33 32 0A 60      ld      (main_sequencer), a
0A36 C9      ret
0A37
0A37
0A37      display_lup_and_high_score:                        ; DATA XREF: 0000:070C|o
0A37 11 04 03      ld      de, #0x304
0A3A CD 9F 30      call   queue_fg_vector_fn
0A3D 11 02 02      ld      de, #0x202
0A40 CD 9F 30      call   queue_fg_vector_fn
0A43 11 00 02      ld      de, #0x200
0A46 CD 9F 30      call   queue_fg_vector_fn
0A49 11 00 06      ld      de, #0x600
0A4C CD 9F 30      call   queue_fg_vector_fn
0A4F 21 0A 60      ld      hl, #main_sequencer
0A52 34      inc      (hl)
0A53
0A53      ; SUBROUTINE
0A53
0A53      display_lup:                                          ; CODE XREF: 0000:01F1|p
0A53 3E 01      ld      a, #1
0A55 32 40 77      ld      (VRAM_start+0x340), a
0A58 3E 25      ld      a, #0x25 ; '%'
0A5A 32 20 77      ld      (VRAM_start+0x320), a
0A5D 3E 20      ld      a, #0x20 ; ' '
0A5F 32 00 77      ld      (VRAM_start+0x300), a
0A62 C9      ret
0A63
0A63      wait_cls_and_check_seen_intro:                       ; DATA XREF: 0000:070E|o
0A63 DF      rst      0x18
0A64 CD 74 08      call   clear_visible_area_and_sprites
0A67 21 09 60      ld      hl, #eight_bit_countdown
0A6A 36 01      ld      (hl), #1
0A6C 2C      inc      l
0A6D 34      inc      (hl)
0A6E 11 2C 62      ld      de, #seen_intro
0A71 1A      ld      a, (de)
0A72 A7      and      a
0A73 C0      ret      NZ
0A74 34      inc      (hl)
0A75 C9      ret
0A76
0A76      vector_on_intro_sequence:                            ; DATA XREF: 0000:0710|o
0A76 3A 85 63      ld      a, (intro_sequencer)
0A79 EF      rst      0x28
0A79
0A7A 8A 0A      .dw draw_climb_screen
0A7C BF 0A      .dw draw_climbing_kong
0A7E E8 0A      .dw animate_kong_climbing_ladder
0A80 69 30      .dw wait_and_inc_sequence
0A82 06 0B      .dw draw_1st_girder_deformation
0A84 69 30      .dw wait_and_inc_sequence
0A86 68 0B      .dw draw_rest_of_deformations
0A88 B3 0B      .dw growl
0A8A
0A8A      draw_climb_screen:                                    ; DATA XREF: display_lup+27|o
0A8A AF      xor      a
0A8B 32 86 7D      ld      (palette_bank), a
0A8E 3C      inc      a
0A8F 32 87 7D      ld      (palette_bank+1), a
0A92 11 0D 38      ld      de, #draw_data_climb
0A95 CD A7 0D      call   draw_level_background
0A98 3E 10      ld      a, #0x10
0A9A 32 A3 76      ld      (VRAM_start+0x2A3), a
0A9D 32 63 76      ld      (VRAM_start+0x263), a
0AA0 3E D4      ld      a, #0xD4 ; 'E'
0AA2 32 AA 75      ld      (VRAM_start+0x1AA), a
0AA5 AF      xor      a
0AA6 32 AF 62      ld      (byte_0_62AF), a
0AA9 21 B4 38      ld      hl, #dk_intro_jump_up_data
0AAC 22 C2 63      ld      (ptr_current_jump_up_data), hl
0AAF 21 CB 38      ld      hl, #dk_intro_jump_left_data
0AB2 22 C4 63      ld      (ptr_current_jump_left_data), hl
0AB5 3E 40      ld      a, #0x40 ; '@'
0AB7 32 09 60      ld      (eight_bit_countdown), a
0ABA 21 85 63      ld      hl, #intro_sequencer
0ABD 34      inc      (hl)
0ABE C9      ret
0ABF
0ABF      draw_climbing_kong:                                ; DATA XREF: display_lup+29|o
0ABF DF      rst      0x18
0AC0 21 8C 38      ld      hl, #dk_climbing_spr
0AC3 CD 4E 00      call   copy_sprites_2_1l_data
0AC6 21 08 69      ld      hl, #soft_sprite_ram+8
0AC9 0E 30      ld      c, #48
0ACB FF      rst      0x38
0ACC 21 0B 69      ld      hl, #soft_sprite_ram+0xB
0ACF 0E 99      ld      c, #153
0AD1 FF      rst      0x38
0AD2 3E 1F      ld      a, #0x1F
0AD4 32 8E 63      ld      (byte_0_638E), a
0AD7 AF      xor      a
0AD8 32 0C 69      ld      (soft_sprite_ram+0xC), a
0ADB 21 8A 60      ld      hl, #unk_0_608A
0ADE 36 01      ld      (hl), #1
0AE0 23      inc      hl
0AE1 36 03      ld      (hl), #3
0AE3 21 85 63      ld      hl, #intro_sequencer
0AE6 34      inc      (hl)
0AE7 C9      ret
0AE8

```

```

0AE8
0AE8 animate_kong_climbing_ladder: ; DATA XREF: display_1UP+2B|o
0AE8 CD 6F 30 call animate_kong_climbing
0AEB 3A AF 62 ld a, (byte_0_62AF)
0AEE E6 0F and #0xF ; time to wipe ladder?
0AF0 CC 4A 30 call Z, wipe_ladder_as_kong_climbs ; yes, do so
0AF3 3A 0B 69 ld a, (soft_sprite_ram+0xB) ; sprite #2, x coord
0AF6 FE 5D cp #0x5D ; ']' ; done climbing?
0AF8 D0 ret NC ; on, return
0AF9 3E 20 ld a, #0x20 ; ' '
0AFB 32 09 60 ld (eight_bit_countdown), a
0AFE 21 85 63 ld hl, #intro_sequencer
0B01 34 inc (hl) ; next sequence (3)
0B02 22 C0 63 ld (ptr_current_sequence), hl
0B05 C9 ret
0B06
0B06
0B06 draw_1st_girder_deformation: ; DATA XREF: display_1UP+2F|o
0B06 3A 1A 60 ld a, (gen_purpose_timer)
0B09 0F rrca ; time to animate?
0B0A D8 ret C ; no, return
0B0B 2A C2 63 ld hl, (ptr_current_jump_up_data)
0B0E 7E ld a, (hl)
0B0F FE 7F cp #0x7F ; ' ' ; done jumping up?
0B11 CA 1E 0B jp Z, draw_pauline_and_kong ; yes, skip
0B14 23 inc hl
0B15 22 C2 63 ld (ptr_current_jump_up_data), hl
0B18 4F ld c, a
0B19 21 0B 69 ld hl, #soft_sprite_ram+0xB ; sprite #2,X coord
0B1C FF rst 0x38
0B1D C9 ret
0B1E
0B1E
0B1E draw_pauline_and_kong: ; CODE XREF: display_1UP+BE|j
0B1E 21 5C 38 ld hl, #dk_normal_spr
0B1E ; End of function display_1UP
0B21 CD 4E 00 call copy_sprites_2_1l_data
0B24 11 00 69 ld de, #soft_sprite_ram
0B27 01 08 00 ld bc, #8
0B2A ED B0 ldir ; place pauline on girder
0B2C 21 08 69 ld hl, #soft_sprite_ram+8 ; sprite #2, y coord
0B2F 0E 50 ld c, #0x50 ; 'P'
0B31 FF rst 0x38
0B32 21 0B 69 ld hl, #soft_sprite_ram+0xB ; sprite #2, x coord
0B35 0E FC ld c, #0xFC ; '3'
0B37 FF rst 0x38
0B38
0B38 loc_0_B38: ; CODE XREF: 0000:0B40|j
0B38 CD 4A 30 call wipe_ladder_as_kong_climbs
0B3B 3A 8E 63 ld a, (byte_0_638E)
0B3E FE 0A cp #0xA ; done wiping ladders?
0B40 C2 38 0B jp NZ, loc_0_B38 ; no, loop
0B43 3E 03 ld a, #3 ; tmr=3
0B45 32 82 60 ld (digital_snd_tmr_thump), a
0B48 11 2C 39 ld de, #draw_data_bend_girders_1
0B4B CD A7 0D call draw_level_background
0B4E 3E 10 ld a, #0x10
0B50 32 AA 74 ld (VRAM_start+0xAA), a
0B53 32 8A 74 ld (VRAM_start+0x8A), a
0B56 3E 05 ld a, #5
0B58 32 8D 63 ld (next_girder_to_deform), a
0B5B 3E 20 ld a, #0x20 ; ' '
0B5D 32 09 60 ld (eight_bit_countdown), a
0B60 21 85 63 ld hl, #intro_sequencer
0B63 34 inc (hl)
0B64 22 C0 63 ld (ptr_current_sequence), hl
0B67 C9 ret
0B68
0B68
0B68 draw_rest_of_deformations: ; DATA XREF: display_1UP+33|o
0B68 3A 1A 60 ld a, (gen_purpose_timer)
0B6B 0F rrca
0B6C D8 ret C
0B6D 2A C4 63 ld hl, (ptr_current_jump_left_data)
0B70 7E ld a, (hl)
0B71 FE 7F cp #0x7F ; ' '
0B73 CA 86 0B jp Z, loc_0_B86
0B76 23 inc hl
0B77 22 C4 63 ld (ptr_current_jump_left_data), hl
0B7A 21 0B 69 ld hl, #soft_sprite_ram+0xB ; sprite #2, x coord
0B7D 4F ld c, a
0B7E FF rst 0x38
0B7F 21 08 69 ld hl, #soft_sprite_ram+8 ; sprite #2, y coord
0B82 0E FF ld c, #0xFF
0B84 FF rst 0x38 ; subtract 1 from y coord for 10 sprites
0B85 C9 ret
0B86
0B86
0B86 loc_0_B86: ; CODE XREF: 0000:0B73|j
0B86 21 CB 38 ld hl, #dk_intro_jump_left_data
0B89 22 C4 63 ld (ptr_current_jump_left_data), hl
0B8C 3E 03 ld a, #3 ; tmr=3
0B8E 32 82 60 ld (digital_snd_tmr_thump), a
0B91 21 DC 38 ld hl, #draw_data_bend_girders_2
0B94 3A 8D 63 ld a, (next_girder_to_deform)
0B97 3D dec a
0B98 07 rlca
0B99 07 rlca
0B9A 07 rlca
0B9B 07 rlca
0B9C 5F ld e, a
0B9D 16 00 ld d, #0
0B9F 19 add hl, de
0BA0 EB ex de, hl
0BA1 CD A7 0D call draw_level_background
0BA4 21 8D 63 ld hl, #next_girder_to_deform
0BA7 35 dec (hl)
0BA8 C0 ret NZ
0BA9 3E B0 ld a, #0xB0 ; 'B'
0BAB 32 09 60 ld (eight_bit_countdown), a
0BAE 21 85 63 ld hl, #intro_sequencer
0BB1 34 inc (hl)
0BB2 C9 ret
0BB3
0BB3

```

```

0BB3      growl:      ld      hl, #unk_0_608A      ; DATA XREF: display_1UP+35|o
0BB3 21 8A 60      ld      a, (eight_bit_countdown)
0BB6 3A 09 60      cp      #0x90 ; 'E'
0BB9 FE 90      jr      NZ, loc_0_BC8
0BBB 20 0B      ld      (hl), #0xF
0BBD 36 0F      inc      hl
0BBF 23      ld      (hl), #3
0BC0 36 03      ld      hl, #soft_sprite_ram+0x19      ; sprite #6, flipy & code
0BC2 21 19 69      inc      (hl)
0BC5 34      jr      loc_0_BD1
0BC6 18 09      ;
0BC8      loc_0_BC8:      cp      #0x18      ; CODE XREF: 0000:0BBB|j
0BC8      jr      NZ, loc_0_BD1
0BCA 20 05      ld      hl, #soft_sprite_ram+0x19      ; sprite #6, flipy & code
0BCC 21 19 69      dec      (hl)
0BCF 35      nop
0BD0 00      loc_0_BD1:      ; CODE XREF: 0000:0BC6|j
0BD1      rst      0x18      ; 0000:0BCA|j
0BD1 DF      ; wait for 8-bit countdown
0BD1      xor      a
0BD2 AF      ld      (intro_sequencer), a
0BD3 32 85 63      inc      (hl)
0BD6 34      inc      hl
0BD7 23      inc      hl
0BD8 34      inc      hl
0BD9 C9      ret
0BDA      ;
0BDA      draw_how_high_can_you_get:      ; DATA XREF: 0000:0712|o
0BDA CD 1C 01      call    stop_sound
0BDD DF      rst      0x18      ; wait for 8-bit countdown
0BDE CD 74 08      call    clear_visible_area_and_sprites
0BE1 16 06      ld      d, #6      ; display_lives_and_level
0BE3 3A 00 62      ld      a, (mario_alive_flag)
0BE6 5F      ld      e, a
0BE7 CD 9F 30      call    queue_fg_vector_fn
0BEA 21 86 7D      ld      hl, #palette_bank
0BED 36 01      ld      (hl), #1
0BEF 23      inc      hl
0BF0 36 00      ld      (hl), #0      ; set palette #1
0BF2 21 8A 60      ld      hl, #unk_0_608A
0BF5 36 02      ld      (hl), #2
0BF7 23      inc      hl
0BF8 36 03      ld      (hl), #3
0BFA 21 A7 63      ld      hl, #height_counter
0BFD 36 00      ld      (hl), #0
0BFF 21 DC 76      ld      hl, #VRAM_start+0x2DC      ; display location for height strings
0C02 22 A8 63      ld      (disp_loc_for_height_string), hl
0C05 3A 2E 62      ld      a, (height)
0C08 FE 06      cp      #6      ; higher than max?
0C0A 38 05      jr      C, loc_0_C11      ; no, skip
0C0C 3E 05      ld      a, #5      ; set max height
0C0E 32 2E 62      ld      (height), a      ; update
0C11      loc_0_C11:      ; CODE XREF: 0000:0C0A|j
0C11 3A 2F 62      ld      a, (last_seq_lsb)
0C14 47      ld      b, a
0C15 3A 2A 62      ld      a, (seq_data)      ; lsb of current level sequence ptr
0C18 B8      cp      b      ; same as last time?
0C19 28 04      jr      Z, loc_0_C1F      ; yes, skip
0C1B 21 2E 62      ld      hl, #height
0C1E 34      inc      (hl)      ; inc height
0C1F      loc_0_C1F:      ; CODE XREF: 0000:0C19|j
0C1F 32 2F 62      ld      (last_seq_lsb), a      ; update
0C22 3A 2E 62      ld      a, (height)
0C25 47      ld      b, a
0C26 21 BC 75      ld      hl, #VRAM_start+0x1BC      ; display location for kong
0C29      loc_0_C29:      ; CODE XREF: 0000:0C7F|j
0C29 0E 50      ld      c, #0x50 ; 'P'      ; 1st tile for kong
0C2B      loc_0_C2B:      ; CODE XREF: 0000:0C40|j
0C2B 71      ld      (hl), c      ; display
0C2C 0C      inc      c      ; next tile
0C2D 2B      dec      hl      ; next location
0C2E 71      ld      (hl), c      ; display
0C2F 0C      inc      c      ; next tile
0C30 2B      dec      hl      ; next location
0C31 71      ld      (hl), c      ; display
0C32 0C      inc      c      ; next tile
0C33 2B      dec      hl      ; next location
0C34 71      ld      (hl), c      ; display
0C35 79      ld      a, c
0C36 FE 67      cp      #0x67 ; 'g'      ; last tile?
0C38 CA 43 0C      jp      Z, loc_0_C43      ; yes, skip (exit)
0C3B 0C      inc      c      ; next tile
0C3C 11 23 00      ld      de, #0x23 ; '#'      ; column offset
0C3F 19      add      hl, de      ; next column
0C40 C3 2B 0C      jp      loc_0_C2B      ; loop another column
0C43      loc_0_C43:      ; CODE XREF: 0000:0C38|j
0C43 3A A7 63      ld      a, (height_counter)
0C46 3C      inc      a
0C47 32 A7 63      ld      (height_counter), a
0C4A 3D      dec      a      ; 0-based
0C4B CB 27      sla      a      ; x4 for table entry
0C4D CB 27      sla      a
0C4F E5      push     hl
0C50 21 F0 3C      ld      hl, #how_high_strings
0C53 C5      push     bc
0C54 DD 2A A8 63      ld      ix, (disp_loc_for_height_string)      ; display location for height strings
0C58 4F      ld      c, a      ; table entry offset
0C59 06 00      ld      b, #0
0C5B 09      add      hl, bc      ; get ptr how high string
0C5C 7E      ld      a, (hl)      ; get 1st byte
0C5D DD 77 60      ld      (0x60(ix), a)      ; display
0C60 23      inc      hl
0C61 7E      ld      a, (hl)      ; get 2nd byte
0C62 DD 77 40      ld      (0x40(ix), a)      ; display
0C65 23      inc      hl
0C66 7E      ld      a, (hl)      ; get 3rd byte

```

```

0C67 DD 77 20      ld      0x20(ix), a      ; display
0C6A DD 36 E0 8B   ld      0xE0(ix), #0x8B ; 'i'      ; "m"
0C6E C1            pop      bc
0C6F DD E5        push     ix
0C71 E1            pop      hl
0C72 11 FC FF      ld      de, #0xFFFF      ; offset for next string
0C75 19            add      hl, de      ; display location for next string
0C76 22 A8 63      ld      (disp_loc_for_height_string), hl
0C79 E1            pop      hl
0C7A 11 5F FF      ld      de, #0xFF5F
0C7D 19            add      hl, de
0C7E 05            dec      b
0C7F C2 29 0C      jp      NZ, loc_0_C29
0C82 11 07 03      ld      de, #0x307      ; display_message_07 "HOW HIGH CAN YOU GET"
0C85 CD 9F 30      call     queue_fg_vector_fn
0C88 21 09 60      ld      hl, #eight_bit_countdown
0C8B 36 A0          ld      (hl), #0xA0 ; 'á'
0C8D 23            inc      hl
0C8E 34            inc      (hl)
0C8F 34            inc      (hl)
0C90 C9            ret
0C91
0C91
0C91
0C91 DF            rst      0x18      ; DATA XREF: 0000:0716|o
0C92                                ; wait for 8-bit countdown
0C92
0C92
0C92 init_and_draw_level:      ; CODE XREF: 0000:0776|j
0C92 CD 74 08      call     clear_visible_area_and_sprites
0C95 AF            xor      a
0C96 32 8C 63      ld      (bonus_timer), a      ; init bonus timer
0C99 11 01 05      ld      de, #0x501      ; update_bonus_timer (tick)
0C9C CD 9F 30      call     queue_fg_vector_fn
0C9F 21 86 7D      ld      hl, #palette_bank
0CA2 36 00          ld      (hl), #0
0CA4 23            inc      hl
0CA5 36 01          ld      (hl), #1      ; select palette bank 2
0CA7 3A 27 62      ld      a, (level_type)
0CAA 3D            dec      a      ; barrel level?
0CAB CA D4 0C      jp      Z, draw_barrel_level      ; yes, skip
0CAE 3D            dec      a      ; cement pie level?
0CAF CA DF 0C      jp      Z, draw_cement_pie_level      ; yes, skip
0CB2 3D            dec      a      ; elevator level?
0CB3 CA F2 0C      jp      Z, draw_elevator_level      ; yes, skip
0CB6 CD 43 0D      call     draw_rivet_level_top_support
0CB9 21 86 7D      ld      hl, #palette_bank
0CBC 36 01          ld      (hl), #1      ; select palette bank 3
0CBE 3E 0B          ld      a, #0xB
0CC0 32 89 60      ld      (bg_music), a
0CC3 11 8B 3C      ld      de, #rivet_level_tilemap_data
0CC6
0CC6
0CC6 CD A7 0D      call     draw_level_background      ; CODE XREF: 0000:0CDC|j
0CC9 3A 27 62      ld      a, (level_type)      ; 0000:0CEF|j ...
0CCC FE 04          cp      #4      ; draw screen
0CCE CC 00 0D      call     Z, draw_8_rivets      ; rivets?
0CD1 C3 A0 3F      jp      init_level_data_tmrs_spr      ; yes, call
0CD4
0CD4
0CD4
0CD4 11 E4 3A      ld      de, #barrel_level_tilemap_data      ; CODE XREF: 0000:0CAB|j
0CD7 3E 08          ld      a, #8
0CD9 32 89 60      ld      (bg_music), a
0CDC C3 C6 0C      jp      draw_level_tilemap
0CDF
0CDF
0CDF
0CDF 11 5D 3B      ld      de, #cement_pie_level_tilemap_data      ; CODE XREF: 0000:0CAF|j
0CE2 21 86 7D      ld      hl, #palette_bank
0CE5 36 01          ld      (hl), #1
0CE7 23            inc      hl
0CE8 36 00          ld      (hl), #0      ; select palette #1
0CEA 3E 09          ld      a, #9
0CEC 32 89 60      ld      (bg_music), a
0CEF C3 C6 0C      jp      draw_level_tilemap
0CF2
0CF2
0CF2
0CF2 CD 27 0D      call     draw_2_elevator_cables      ; CODE XREF: 0000:0CB3|j
0CF5 3E 0A          ld      a, #0xA
0CF7 32 89 60      ld      (bg_music), a
0CFA 11 E5 3B      ld      de, #elevator_level_tilemap_data
0CFD C3 C6 0C      jp      draw_level_tilemap
0D00
0D00
0D00
0D00
0D00
0D00 06 08          ld      b, #8      ; CODE XREF: 0000:0CCE|p
0D02 21 17 0D      ld      hl, #rivet_loc_tbl      ; 8 rivets
0D05
0D05
0D05
0D05 3E B8          ld      a, #0xB8 ; '@'      ; CODE XREF: draw_8_rivets+14|j
0D07 0E 02          ld      c, #2      ; top of rivet tile
0D09 5E            ld      e, (hl)      ; 2 tiles/rivet (vertical)
0D0A 23            inc      hl
0D0B 56            ld      d, (hl)
0D0C 23            inc      hl      ; get VRAM location
0D0D
0D0D
0D0D loc_0_D0D:      ; CODE XREF: draw_8_rivets+11|j
0D0D 12            ld      (de), a      ; draw rivet tile
0D0E 3D            dec      a      ; next rivet tile
0D0F 13            inc      de      ; next VRAM location
0D10 0D            dec      c      ; done a rivet?
0D11 C2 0D 0D      jp      NZ, loc_0_D0D      ; no, loop
0D14 10 EF          djnz     draw_rivet      ; loop through 8 rivets
0D16 C9            ret
0D16
0D16
0D16
0D17 CA 76          rivet_loc_tbl: .dw VRAM_start+0x2CA      ; DATA XREF: draw_8_rivets+2|o
0D17                                ; Rivets level, location of rivets
0D19 CF 76          .dw VRAM_start+0x2CF
0D1B D4 76          .dw VRAM_start+0x2D4
0D1D D9 76          .dw VRAM_start+0x2D9
0D1F 2A 75          .dw VRAM_start+0x12A

```

```

0D21 2F 75      .dw VRAM_start+0x12F
0D23 34 75      .dw VRAM_start+0x134
0D25 39 75      .dw VRAM_start+0x139
0D27
0D27           ; ██████████ SUBROUTINE ██████████
0D27
0D27
0D27
0D27 draw_2_elevator_cables:                                ; CODE XREF: 0000:0CF2|p
0D27 21 0D 77      ld      hl, #VRAM_start+0x30D
0D2A CD 30 0D      call    draw_elevator_cable
0D2D 21 0D 76      ld      hl, #VRAM_start+0x20D
0D2D           ; End of function draw_2_elevator_cables
0D2D
0D30
0D30           ; ██████████ SUBROUTINE ██████████
0D30
0D30
0D30
0D30 draw_elevator_cable:                                ; CODE XREF: draw_2_elevator_cables+3|p
0D30 06 11      ld      b, #17                            ; cable height 17 tiles
0D32
0D32 loc_0_D32:                                ; CODE XREF: draw_elevator_cable+5|j
0D32 36 FD      ld      (hl), #0xFD ; '2'                ; vertical bar tile left edge
0D34 23      inc      hl                                    ; next row
0D35 10 FB      djnz    loc_0_D32                        ; loop cable height
0D37 11 0F 00      ld      de, #0xFF
0D3A 19      add      hl, de
0D3B 06 11      ld      b, #17                            ; cable height 17 tiles
0D3D
0D3D loc_0_D3D:                                ; CODE XREF: draw_elevator_cable+10|j
0D3D 36 FC      ld      (hl), #0xFC ; '3'                ; vertical bar tile right edge
0D3F 23      inc      hl                                    ; next row
0D40 10 FB      djnz    loc_0_D3D                        ; loop cable height
0D42 C9      ret
0D42           ; End of function draw_elevator_cable
0D42
0D43
0D43           ; ██████████ SUBROUTINE ██████████
0D43
0D43
0D43
0D43 draw_rivet_level_top_support:                    ; CODE XREF: 0000:0CB6|p
0D43 21 87 76      ld      hl, #VRAM_start+0x287
0D46 CD 4C 0D      call    draw_support_bars
0D49 21 47 75      ld      hl, #VRAM_start+0x147
0D49           ; End of function draw_rivet_level_top_support
0D49
0D4C
0D4C           ; ██████████ SUBROUTINE ██████████
0D4C
0D4C
0D4C
0D4C draw_support_bars:                                ; CODE XREF: draw_rivet_level_top_support+3|p
0D4C 06 04      ld      b, #4                              ; 4 rows to draw
0D4E
0D4E loc_0_D4E:                                ; CODE XREF: draw_support_bars+5|j
0D4E 36 FD      ld      (hl), #0xFD ; '2'                ; vertical bar tile left edge
0D50 23      inc      hl                                    ; next row
0D51 10 FB      djnz    loc_0_D4E                        ; loop cable height
0D53 11 1C 00      ld      de, #0x1C
0D56 19      add      hl, de
0D57 06 04      ld      b, #4                              ; 4 rows to draw
0D59
0D59 loc_0_D59:                                ; CODE XREF: draw_support_bars+10|j
0D59 36 FC      ld      (hl), #0xFC ; '3'                ; vertical bar tile right edge
0D5B 23      inc      hl                                    ; next row
0D5C 10 FB      djnz    loc_0_D59                        ; loop cable height
0D5E C9      ret
0D5E           ; End of function draw_support_bars
0D5E
0D5F
0D5F           ; -----
0D5F
0D5F init_level_data_tmrs_spr_cont:                    ; CODE XREF: 0000:3FA3|j
0D5F CD 56 0F      call    initialise_level_data_and_timers
0D62 CD 41 24      call    sub_0_2441
0D65 21 09 60      ld      hl, #eight_bit_countdown
0D68 36 40      ld      (hl), #0x40 ; '@'
0D6A 23      inc      hl                                    ; main_sequencer
0D6B 34      inc      (hl)                                ; next sequence (2)
0D6C 21 5C 38      ld      hl, #dk_normal_spr
0D6F CD 4E 00      call    copy_sprites_2_11_data
0D72 11 00 69      ld      de, #soft_sprite_ram
0D75 01 08 00      ld      bc, #8
0D78 ED B0      ldir
0D7A 3A 27 62      ld      a, (level_type)
0D7D FE 04      cp      #4
0D7F 28 0A      jr      Z, adj_pauline_kong_for_rivets
0D81 0F      rrca
0D82 0F      rrca
0D83 D8      ret
0D84 21 0B 69      ld      hl, #soft_sprite_ram+0xB
0D87 0E FC      ld      c, #0xFC ; '3'
0D89 FF      rst      0x38
0D8A C9      ret
0D8A           ; -----
0D8B
0D8B
0D8B
0D8B adj_pauline_kong_for_rivets:                    ; CODE XREF: 0000:0D7F|j
0D8B 21 08 69      ld      hl, #soft_sprite_ram+8
0D8E 0E 44      ld      c, #68
0D90 FF      rst      0x38
0D91 11 04 00      ld      de, #4
0D94 01 10 02      ld      bc, #0x210
0D97 21 00 69      ld      hl, #soft_sprite_ram
0D9A CD 3D 00      call    add_c_sprite_register_xB
0D9D 01 F8 02      ld      bc, #0x2F8
0DA0 21 03 69      ld      hl, #soft_sprite_ram+3
0DA3 CD 3D 00      call    add_c_sprite_register_xB
0DA6 C9      ret
0DA7
0DA7           ; ██████████ SUBROUTINE ██████████
0DA7
0DA7
0DA7
0DA7 draw_level_background:                            ; CODE XREF: display_LUP+42|p
0DA7 1A      ld      a, (de)                                ; 0000:0B4B|p ...
0DA8 32 B3 63      ld      (segment_type), a
0DAB FE AA      cp      #0xAA ; '-'
0DAD C8      ret      Z
0DAE 13      inc      de
0DAE           ; next table address

```

```

0DAF 1A      ld      a, (de)      ; get byte
0DB0 67      ld      h, a        ; H=Y1
0DB1 44      ld      b, h        ; B=Y1
0DB2 13      inc      de        ; next table address
0DB3 1A      ld      a, (de)      ; get byte
0DB4 6F      ld      l, a        ; L=X1
0DB5 4D      ld      c, l        ; C=X1
0DB6 D5      push     de
0DB7 CD F0 2F call     get_tilemap_addr_from_coords
0DBA D1      pop      de
0DBB 22 AB 63 ld      (segment_addr_1), hl ; store vram address #1
0DBE 78      ld      a, b
0DBF E6 07      and     #7
0DC1 32 B4 63 ld      (tile_byte_1), a
0DC4 79      ld      a, c
0DC5 E6 07      and     #7
0DC7 32 AF 63 ld      (start_tile_index), a
0DCA 13      inc      de        ; next table entry
0DCB 1A      ld      a, (de)      ; Y2
0DCC 67      ld      h, a        ; H=Y2
0DCD 90      sub      b        ; calc delta Y
0DCE D2 D3 0D jp      NC, loc_0_DD3 ; no, skip
0DD1 ED 44      neg      NC        ; delta Y
0DD3
0DD3
0DD3 32 B1 63 loc_0_DD3:      ld      (dY), a        ; CODE XREF: draw_level_background+27|j
0DD6 13      inc      de        ; next entry
0DD7 1A      ld      a, (de)      ; X2
0DD8 6F      ld      l, a        ; L=X2
0DD9 91      sub      c        ; calc delta X
0DDA 32 B2 63 ld      (dX), a
0DDD 1A      ld      a, (de)      ; X2 (again)
0DDE E6 07      and     #7        ; TILE bits only
0DE0 32 B0 63 ld      (end_tile_index), a
0DE3 D5      push     de
0DE4 CD F0 2F call     get_tilemap_addr_from_coords
0DE7 D1      pop      de
0DE8 22 AD 63 ld      (segment_addr_2), hl ; store vram address #2
0DEB 3A B3 63 ld      a, (segment_type) ; flag
0DEE FE 02      cp      #2        ; >=2?
0DF0 F2 4F 0E jp      P, draw_girder_segment ; yes, skip
0DF3
0DF3
0DF3 3A B2 63 draw_ladder_segment: ld      a, (dX)
0DF6 D6 10      sub     #0x10        ; calc starting tile index adjustment
0DF8 47      ld      b, a
0DF9 3A AF 63 ld      a, (start_tile_index)
0DFC 80      add      a, b        ; adjust
0DFD 32 B2 63 ld      (dX), a
0E00 3A AF 63 ld      a, (start_tile_index)
0E03 C6 F0      add     a, #0xF0 ; '-'
0E05 2A AB 63 ld      hl, (segment_addr_1)
0E08 77      ld      (hl), a        ; display tile
0E09 2C      inc      l        ; next row
0E0A D6 30      sub     #0x30 ; '0'
0E0C 77      ld      (hl), a        ; matching ladder tile
0E0D 3A B3 63 ld      a, (segment_type) ; display it
0E10 FE 01      cp      #1        ; broken ladder?
0E12 C2 19 0E jp      NZ, next_tile_in_ladder_segment ; no, skip
0E15 AF      xor      a        ; flag end-of-ladder
0E16 32 B2 63 ld      (dX), a
0E19
0E19
0E19 3A B2 63 next_tile_in_ladder_segment: ; CODE XREF: draw_level_background+6B|j
                                ; draw_level_background+80|j
                                ld      a, (dX)
0E1C D6 08      sub     #8        ; finished ladder?
0E1E 32 B2 63 ld      (dX), a
0E21 DA 2A 0E jp      C, loc_0_E2A        ; yes, skip
0E24 2C      inc      l        ; next row
0E25 36 C0      ld      (hl), #0xC0 ; 'L'
0E27 C3 19 0E jp      next_tile_in_ladder_segment ; loop for ladder
0E2A
0E2A
0E2A
0E2A 3A B0 63 loc_0_E2A:      ld      a, (end_tile_index) ; CODE XREF: draw_level_background+7A|j
0E2D C6 D0      add     a, #0xD0 ; 'D'
0E2F 2A AD 63 ld      hl, (segment_addr_2) ; girder top, bottom of ladder
0E32 77      ld      (hl), a        ; vram address
0E33 3A B3 63 ld      a, (segment_type)
0E36 FE 01      cp      #1        ; broken ladder?
0E38 C2 3F 0E jp      NZ, loc_0_E3F        ; no, skip
0E3B 2D      dec      l        ; row above
0E3C 36 C0      ld      (hl), #0xC0 ; 'L'
0E3E 2C      inc      l        ; display full ladder tile
0E3F
0E3F
0E3F 3A B0 63 loc_0_E3F:      ld      a, (end_tile_index) ; CODE XREF: draw_level_background+91|j
0E42 FE 00      cp      #0        ; 2nd tile (below) req'd?
0E44 CA AB 0E jp      Z, loc_0_E4B        ; no, skip
0E47 C6 E0      add     a, #0xE0 ; 'O'
0E49 2C      inc      l        ; bottom of girder, no ladder below
0E4A 77      ld      (hl), a        ; next row
0E4B
0E4B
0E4B 13      inc      de        ; CODE XREF: draw_level_background+9D|j
0E4C C3 A7 0D jp      draw_level_background ; next entry
                                ; loop through level data
0E4F
0E4F
0E4F 3A B3 63 draw_girder_segment: ; CODE XREF: draw_level_background+49|j
                                ld      a, (segment_type)
0E52 FE 02      cp      #2        ; girder?
0E54 C2 E8 0E jp      NZ, draw_conveyor_segment ; no, skip
0E57 3A AF 63 ld      a, (start_tile_index)
0E5A C6 F0      add     a, #0xF0 ; '-'
0E5C 32 B5 63 ld      (current_tile_in_segment), a ; girder top (no ladder above)
0E5F 2A AB 63 ld      hl, (segment_addr_1) ; initialise girder segment tile
0E62
0E62
0E62 3A B5 63 next_tile_in_girder_segment: ; CODE XREF: draw_level_background+E5|j
                                ; draw_level_background+125|j ...
                                ld      a, (current_tile_in_segment)
0E65 77      ld      (hl), a        ; display it
0E66 23      inc      hl        ; next row
0E67 7D      ld      a, l
0E68 E6 1F      and     #0x1F        ; bottom of screen?
0E6A CA 78 0E jp      Z, loc_0_E78        ; yes, skip
0E6D 3A B5 63 ld      a, (current_tile_in_segment)

```

```

OE70 FE F0      cp      #0xF0 ; '-'          ; full girder?
OE72 CA 78 0E   jp      Z, loc_0_E78          ; yes, skip
OE75 D6 10      sub     #0x10          ; get matching bottom piece
OE77 77         id      (hl), a          ; display it
OE78
OE78            loc_0_E78:                ; CODE XREF: draw_level_background+C3|j
OE78 01 1F 00    ; draw_level_background+CB|j
OE78            ld      bc, #0x1F
OE7B 09         add     hl, bc          ; next column
OE7C 3A B1 63    ld      a, (dY)
OE7F D6 08      sub     #8            ; finished? (ignore [2:0])
OE81 DA CF 0E   jp      C, next_segment ; yes, skip
OE84 32 B1 63    ld      (dY), a
OE87 3A B2 63    ld      a, (dX)
OE8A FE 00      cp      #0            ; angled?
OE8C CA 62 0E   jp      Z, next_tile_in_girder_segment ; no, loop
OE8F 3A B5 63    ld      a, (current_tile_in_segment)
OE92 77         id      (hl), a          ; display it
OE93 23         inc     hl            ; next row
OE94 7D         ld      a, 1
OE95 E6 1F      and     #0x1F          ; bottom of screen?
OE97 CA A0 0E   jp      Z, loc_0_EA0      ; yes, skip
OE9A 3A B5 63    ld      a, (current_tile_in_segment)
OE9D D6 10      sub     #0x10          ; get matching bottom piece
OE9F 77         id      (hl), a          ; display it
OEAA
OEAA            loc_0_EA0:                ; CODE XREF: draw_level_background+F0|j
OEAA 01 1F 00    ld      bc, #0x1F
OEAA 09         add     hl, bc          ; next column
OEAA 3A B1 63    ld      a, (dY)
OEAA D6 08      sub     #8            ; finished? (ignore [2:0])
OEAA DA CF 0E   jp      C, next_segment ; yes, skip
OEAC 32 B1 63    ld      (dY), a
OEAF 3A B2 63    ld      a, (dX)
OEB2 CB 7F      bit     7, a          ; sloping up?
OEB4 C2 D3 0E   jp      NZ, girder_sloping_down ; no, skip
OEB7 3A B5 63    ld      a, (current_tile_in_segment)
OEBA 3C         inc     a            ; next tile
OEBB 32 B5 63    ld      (current_tile_in_segment), a
OEBE FE F8      cp      #0xF8 ; 'o'
OEC0 C2 C9 0E   jp      NZ, loc_0_EC9      ; time to wrap tile?
OEC3 23         inc     hl            ; no, skip
OEC4 3E F0      ld      a, #0xF0 ; '-' ; next row
OEC6 32 B5 63    ld      (current_tile_in_segment), a ; init current tile
OEC9
OEC9            loc_0_EC9:                ; CODE XREF: draw_level_background+119|j
OEC9 7D         ld      a, 1
OECA E6 1F      and     #0x1F          ; bottom of screen?
OECC C2 62 0E   jp      NZ, next_tile_in_girder_segment ; no, loop
OECF
OECF            next_segment:            ; CODE XREF: draw_level_background+DA|j
OECF 13         ; draw_level_background+102|j ...
OECF            inc     de            ; next entry
OED0 C3 A7 0D   jp      draw_level_background ; loop for all entries
OED3
OED3            ;
OED3            girder_sloping_down:      ; CODE XREF: draw_level_background+10D|j
OED3 3A B5 63    ld      a, (current_tile_in_segment)
OED6 3D         dec     a            ; next tile in sequence is -1
OED7 32 B5 63    ld      (current_tile_in_segment), a
OEDA FE F0      cp      #0xF0 ; '-'
OEDC F2 E5 0E   jp      P, loc_0_EE5      ; time to wrap tile?
OEDF 2B         dec     hl            ; no, skip
OEE0 3E F7      ld      a, #0xF7 ; ',' ; next row
OEE2 32 B5 63    ld      (current_tile_in_segment), a ; init current tile
OEE5
OEE5            loc_0_EE5:                ; CODE XREF: draw_level_background+135|j
OEE5 C3 62 0E   jp      next_tile_in_girder_segment ; loop
OEE8
OEE8            ;
OEE8            draw_conveyor_segment:    ; CODE XREF: draw_level_background+AD|j
OEE8 3A B3 63    ld      a, (segment_type)
OEEB FE 03      cp      #3
OEED C2 1B 0F   jp      NZ, draw_other_segments ; conveyor?
OEF0 2A AB 63    ld      hl, (segment_addr_1) ; no, skip
OEF3 3E B3      ld      a, #0xB3 ; '|'
OEF5 77         id      (hl), a          ; empty tile!?!
OEF6 01 20 00    ld      (hl), a          ; display it
OEF9 09         add     hl, bc          ; next column
OEFA 3A B1 63    ld      a, (dY)
OEFD D6 10      sub     #0x10          ; 2nd last tile?
OEFF
OEFF            next_tile_on_coneyor_segment: ; CODE XREF: draw_level_background+16A|j
OEFF DA 14 0F   jp      C, end_of_conveyor_segment ; yes, skip
OF02 32 B1 63    ld      (dY), a
OF05 3E B1      ld      a, #0xB1 ; '≡'
OF07 77         id      (hl), a          ; conveyor tile
OF08 01 20 00    ld      (hl), a          ; display it
OF0B 09         add     hl, bc          ; next column
OF0C 3A B1 63    ld      a, (dY)
OF0F D6 08      sub     #8
OF11 C3 FF 0E   jp      next_tile_on_coneyor_segment ; loop through conveyor
OF14
OF14            ;
OF14            end_of_conveyor_segment:  ; CODE XREF: draw_level_background+158|j
OF14 3E B2      ld      a, #0xB2 ; '■'
OF16 77         id      (hl), a          ; end of conveyor
OF17 13         inc     de            ; display it
OF18 C3 A7 0D   jp      draw_level_background ; return
OF1B
OF1B            ;
OF1B            draw_other_segments:      ; CODE XREF: draw_level_background+146|j
OF1B 3A B3 63    ld      a, (segment_type)
OF1E FE 07      cp      #7
OF20 F2 CF 0E   jp      P, next_segment ; valid segment?
OF23 FE 04      cp      #4            ; no, continue
OF25 CA 4C 0F   jp      Z, draw_blank_segment ; blank?
OF28 FE 05      cp      #5            ; yes, skip
OF2A CA 51 0F   jp      Z, draw_rivet_level_girder ; rivet level girder?
OF2D 3E FE      ld      a, #0xFE ; '■' ; yes, skip
OF2F            ; oil barrel stand (conveyor level)
OF2F            loc_0_F2F:                ; CODE XREF: draw_level_background+1A7|j
OF2F 32 B5 63    ld      (current_tile_in_segment), a ; draw_level_background+1AC|j
OF2F            ld      hl, (segment_addr_1)
OF32 2A AB 63    ld      hl, (segment_addr_1)
OF35

```



```

0F35      next_other_segment_tile:
0F35 3A B5 63      ld      a, (current_tile_in_segment)      ; CODE XREF: draw_level_background+19E|j
0F38 77      ld      (hl), a      ; display tile
0F39 01 20 00      ld      bc, #0x20 ; ' '
0F3C 09      add     hl, bc      ; next column
0F3D 3A B1 63      ld      a, (dY)
0F40 D6 08      sub     #8      ; done?
0F42 32 B1 63      ld      (dY), a
0F45 D2 35 0F      jp      NC, next_other_segment_tile      ; no, loop
0F48 13      inc     de      ; next entry
0F49 C3 A7 0D      jp      draw_level_background
0F4C
0F4C
0F4C
0F4C 3E E0      ld      a, #0xE0 ; 'Ó'      ; CODE XREF: draw_level_background+17E|j
0F4E C3 2F 0F      jp      loc_0_F2F      ; blank tile
0F51
0F51
0F51
0F51 3E B0      ld      a, #0xB0 ; 'Þ'      ; CODE XREF: draw_level_background+183|j
0F53 C3 2F 0F      jp      loc_0_F2F      ; rivet level girder
0F53      ; End of function draw_level_background
0F56
0F56
0F56
0F56      initialise_level_data_and_timers:      ; CODE XREF: 0000:0D5F|p
0F56 06 27      ld      b, #39
0F58 21 00 62      ld      hl, #mario_alive_flag
0F5B AF      xor     a
0F5C
0F5C      loc_0_F5C:      ; CODE XREF: 0000:0F5E|j
0F5C 77      ld      (hl), a
0F5D 2C      inc     l
0F5E 10 FC      djnz   loc_0_F5C      ; clear 39 bytes
0F60 0E 11      ld      c, #17
0F62 16 80      ld      d, #128
0F64 21 80 62      ld      hl, #unk_0_6280      ; $6280-$6AFF cleared
0F67
0F67      loc_0_F67:      ; CODE XREF: 0000:0F6D|j
0F67 42      ld      b, d      ; 128 bytes to clear
0F68
0F68      loc_0_F68:      ; CODE XREF: 0000:0F6A|j
0F68 77      ld      (hl), a      ; clear byte
0F69 23      inc     hl
0F6A 10 FC      djnz   loc_0_F68      ; clear 128 bytes
0F6C 0D      dec     c
0F6D 20 F8      jr     NZ, loc_0_F67      ; clear 17*128=2176($880) bytes
0F6F 21 9C 3D      ld      hl, #level_init_data
0F72 11 80 62      ld      de, #unk_0_6280
0F75 01 40 00      ld      bc, #64
0F78 ED B0      ldir
0F7A 3A 29 62      ld      a, (level)      ; init 64 bytes
0F7D 47      ld      b, a
0F7E A7      and     a
0F7F 17      rla
0F80 A7      and     a      ; level * 2
0F81 17      rla      ; level * 4
0F82 A7      and     a      ; level * 8
0F83 17      rla      ; level * 9
0F84 80      add     a, b      ; level * 10
0F85 80      add     a, b      ; level * 10 + 40
0F86 C6 28      add     a, #40      ; max?
0F88 FE 51      cp      #81      ; no, skip
0F8A 38 02      jr     C, loc_0_F8E      ; max out at 50(00) (BCD)
0F8C 3E 50      ld      a, #0x50 ; 'P'
0F8E
0F8E      loc_0_F8E:      ; CODE XREF: 0000:0F8A|j
0F8E 21 B0 62      ld      hl, #bonus_timer_init_value      ; 3 timers to initialise
0F91 06 03      ld      b, #3
0F93
0F93      loc_0_F93:      ; CODE XREF: 0000:0F95|j
0F93 77      ld      (hl), a      ; store timer value
0F94 2C      inc     l      ; next timer
0F95 10 FC      djnz   loc_0_F93      ; loop for 3 timers
0F97 87      add     a, a      ; level * 20 + 80
0F98 47      ld      b, a
0F99 3E DC      ld      a, #220
0F9B 90      sub     b      ; 220-(level*20+80)=140-level*20
0F9C FE 28      cp      #40      ; min?
0F9E 30 02      jr     NC, loc_0_FA2      ; no, skip
0FA0 3E 28      ld      a, #40      ; set min=40
0FA2
0FA2      loc_0_FA2:      ; CODE XREF: 0000:0F9E|j
0FA2 77      ld      (hl), a      ; set timer
0FA3 2C      inc     l      ; next timer
0FA4 77      ld      (hl), a      ; set timer
0FA5 21 09 62      ld      hl, #unk_0_6209
0FA8 36 04      ld      (hl), #4
0FAA 2C      inc     l
0FAB 36 08      ld      (hl), #8
0FAD 3A 27 62      ld      a, (level_type)
0FB0 4F      ld      c, a
0FB1 CB 57      bit     2, a      ; rivets level?
0FB3 20 16      jr     NZ, loc_0_FCB      ; yes, skip
0FB5 21 00 6A      ld      hl, #soft_sprite_ram+0x100      ; sprite #64, y coord
0FB8 3E 4F      ld      a, #0x4F ; 'O'      ; sprite X position
0FBA 06 03      ld      b, #3      ; 3 sprites to draw
0FBC
0FBC      erase_top_of_kong_ladder:      ; CODE XREF: 0000:0FC9|j
0FBC 77      ld      (hl), a      ; set sprite X pos
0FBD 2C      inc     l
0FBE 36 3A      ld      (hl), #0x3A ; ':'      ; set sprite tile (blank)
0FC0 2C      inc     l
0FC1 36 0F      ld      (hl), #0xF      ; set sprite colour
0FC3 2C      inc     l
0FC4 36 18      ld      (hl), #0x18      ; set sprite Y pos
0FC6 2C      inc     l
0FC7 C6 10      add     a, #0x10      ; next X pos
0FC9 10 F1      djnz   erase_top_of_kong_ladder      ; loop for 3 sprites
0FCB
0FCB      loc_0_FCB:      ; CODE XREF: 0000:0FB3|j
0FCB 79      ld      a, c      ; level type
0FCC EF      rst     0x28      ; go!
0FCC
0FCD 00 00      .dw RESET      ; Jump table
0FCF D7 0F      .dw init_ll_girder

```

```

0FD1 1F 10      .dw init_l2_cement
0FD3 87 10      .dw init_l3_elevator
0FD5 31 11      .dw init_l4_rivets
0FD7
0FD7
0FD7
init_l1_girder:
0FD7 21 DC 3D      ld      hl, #top_barrel_spr          ; DATA XREF: 0000:0FCF|o
0FDA 11 A8 69      ld      de, #soft_sprite_ram+0xA8          ; sprite #42, Y coord
0FDD 01 10 00      ld      bc, #0x10          ; data for 4 sprites
0FE0 ED B0          ldir          ; init
0FE2 21 EC 3D      ld      hl, #fireball_spr
0FE5 11 07 64      ld      de, #unk_0_6407
0FE8 0E 1C          ld      c, #0x1C          ; offset of each sprite
0FEA 06 05          ld      b, #5          ; do 5 sprites
0FEC CD 2A 12      call   init_data_for_B_sprites
0FEF 21 F4 3D      ld      hl, #girders_fireball_spr
0FF2 CD FA 11      call   init_fireball_sprite
0FF5 21 00 3E      ld      hl, #girder_oil_barrel_spr
0FF8 11 FC 69      ld      de, #soft_sprite_ram+0xFC          ; sprite #63
0FFB 01 04 00      ld      bc, #4          ; 1 sprite only
0FFE ED B0          ldir          ; init sprite
1000 21 0C 3E      ld      hl, #girder_hammer_locs
1003 CD A6 11      call   init_hammer_sprites
1006
1006
loc_0_1006:
1006 21 1B 10      ld      hl, #barrel_init_data
1009 11 07 67      ld      de, #unk_0_6707
100C 01 1C 08      ld      bc, #0x81C          ; 8 sprites, offset $1C
100F CD 2A 12      call   init_data_for_B_sprites
1012 11 07 68      ld      de, #unk_0_6807
1015 06 02          ld      b, #2          ; 2 sprites to copy
1017 CD 2A 12      call   init_data_for_B_sprites
101A C9            ret
101A
101B 00 00 02 02 barrel_init_data:.db 0, 0, 2, 2          ; DATA XREF: 0000:1006|o
101F
101F
init_l2_cement:
101F 21 EC 3D      ld      hl, #fireball_spr          ; DATA XREF: 0000:0FD1|o
1022 11 07 64      ld      de, #unk_0_6407
1025 01 1C 05      ld      bc, #0x51C          ; 5 sprites, offset 0x1c
1028 CD 2A 12      call   init_data_for_B_sprites
102B CD 86 11      call   init_spring_sprites
102E 21 18 3E      ld      hl, #cement_pie_spr
1031 11 A7 65      ld      de, #unk_0_65A7
1034 01 0C 06      ld      bc, #0x60C          ; 6 sprites, offset 0x0c
1037 CD 2A 12      call   init_data_for_B_sprites
103A DD 21 A0 65      ld      ix, #unk_0_65A0
103E 21 B8 69      ld      hl, #soft_sprite_ram+0xB8          ; sprite #46-51
1041 11 10 00      ld      de, #0x10          ; offset 0x10
1044 06 06          ld      b, #6          ; 6 sprites to init
1046 CD D3 11      call   set_B_sprites_data
1049 21 FA 3D      ld      hl, #cement_fireball_spr
104C CD FA 11      call   init_fireball_sprite
104F 21 04 3E      ld      hl, #cement_oil_barrel_spr
1052 11 FC 69      ld      de, #soft_sprite_ram+0xFC          ; sprite #63
1055 01 04 00      ld      bc, #4          ; init oil barrel sprite
1058 ED B0          ldir          ; init oil barrel sprite
105A 21 1C 3E      ld      hl, #cement_ladder_spr
105D 11 44 69      ld      de, #soft_sprite_ram+0x44          ; sprite #17-18
1060 01 08 00      ld      bc, #8          ; 8 bytes = 2 sprits
1063 ED B0          ldir          ; init oil barrel sprite
1065 21 24 3E      ld      hl, #cement_conveyor_spr
1068 11 E4 69      ld      de, #soft_sprite_ram+0xE4          ; sprite #57-62
106B 01 18 00      ld      bc, #0x18          ; 0x18 bytes = 6 sprites
106E ED B0          ldir          ; init oil barrel sprite
1070 21 10 3E      ld      hl, #cement_hammer_locs
1073 CD A6 11      call   init_hammer_sprites
1076 21 3C 3E      ld      hl, #cement_obj_spr          ; hat, purse & umbrella
1079 11 0C 6A      ld      de, #soft_sprite_ram+0x10C          ; sprites #67-69
107C 01 0C 00      ld      bc, #0xC          ; 12 bytes = 3 sprites
107F ED B0          ldir          ; init oil barrel sprite
1081 3E 01          ld      a, #1
1083 32 B9 62      ld      (unk_0_62B9), a
1086 C9            ret
1087
1087
init_l3_elevator:
1087 21 EC 3D      ld      hl, #fireball_spr          ; DATA XREF: 0000:0FD3|o
108A 11 07 64      ld      de, #unk_0_6407
108D 01 1C 05      ld      bc, #0x51C          ; 5 sprites, offset 0x1c
1090 CD 2A 12      call   init_data_for_B_sprites
1093 CD 86 11      call   init_spring_sprites
1096 21 00 66      ld      hl, #unk_0_6600
1099 11 10 00      ld      de, #0x10
109C 3E 01          ld      a, #1
109E 06 06          ld      b, #6
10A0
10A0
loc_0_10A0:
10A0 77            ld      (hl), a          ; CODE XREF: 0000:10A2|j
10A1 19            add     hl, de
10A2 10 FC          djnz   loc_0_10A0
10A4 0E 02          ld      c, #2
10A6 3E 08          ld      a, #8
10A8
10A8
loc_0_10A8:
10A8 06 03          ld      b, #3          ; CODE XREF: 0000:10B4|j
10AA 21 0D 66      ld      hl, #unk_0_660D
10AD
10AD
loc_0_10AD:
10AD 77            ld      (hl), a          ; CODE XREF: 0000:10AF|j
10AE 19            add     hl, de
10AF 10 FC          djnz   loc_0_10AD
10B1 3E 08          ld      a, #8
10B3 0D            dec     c
10B4 C2 A8 10      jp      NZ, loc_0_10A8
10B7 21 64 3E      ld      hl, #elevator_spr_locs
10BA 11 03 66      ld      de, #unk_0_6603
10BD 01 0E 06      ld      bc, #0x60E          ; 6 sprites, offset #0x0c
10C0 CD EC 11      call   init_objects_locations
10C3 21 60 3E      ld      hl, #elevator_spr
10C6 11 07 66      ld      de, #unk_0_6607
10C9 01 0C 06      ld      bc, #0x60C          ; 6 sprites, offset 0x0c
10CC CD 2A 12      call   init_data_for_B_sprites
10CF DD 21 00 66      ld      ix, #unk_0_6600
10D3 21 58 69      ld      hl, #soft_sprite_ram+0x58          ; sprites #22-27

```

```

10D6 06 06      ld      b, #6                ; 6 sprites
10D8 11 10 00   ld      de, #0x10           ; offset 0x10
10DB CD D3 11   call    set_B_sprites_data
10DE 21 48 3E   ld      hl, #elevator_obj_spr    ; hat, purse & umbrella
10E1 11 0C 6A   ld      de, #soft_sprite_ram+0x10C ; sprites 67-69
10E4 01 0C 00   ld      bc, #0xC              ; 0x0c bytes = 3 sprites
10E7 ED B0      ldir
10E9 DD 21 00 64 ld      ix, #unk_0_6400           ; fireball character data
10ED DD 36 00 01 ld      0(ix), #1
10F1 DD 36 03 58 ld      3(ix), #0x58 ; 'X'
10F5 DD 36 0E 58 ld      0xE(ix), #0x58 ; 'X'
10F9 DD 36 05 80 ld      5(ix), #0x80 ; 'C'
10FD DD 36 0F 80 ld      0xF(ix), #0x80 ; 'C'
1101 DD 36 20 01 ld      0x20(ix), #1                ; 2nd fireball
1105 DD 36 23 EB ld      0x23(ix), #0xEB ; 'Ü'
1109 DD 36 2E EB ld      0x2E(ix), #0xEB ; 'Ü'
110D DD 36 25 60 ld      0x25(ix), #0x60 ; ''
1111 DD 36 2F 60 ld      0x2F(ix), #0x60 ; ''
1115 11 70 69   ld      de, #soft_sprite_ram+0x70 ; sprite #28-31
1118 21 21 11   ld      hl, #elevator_cap_spr
111B 01 10 00   ld      bc, #0x10              ; 0x10 bytes = 4 sprites
111E ED B0      ldir
1120 C9          ret
1120
1121 37 45 0F 60+ elevator_cap_spr:.db 0x37, 0x45, 0xF, 0x60, 0x37, 0x45, 0x8F, 0xF7, 0x77
1121 37 45 8F F7+ ; DATA XREF: 0000:1118|o
1121 77 45 0F 60+ .db 0x45, 0xF, 0x60, 0x77, 0x45, 0x8F, 0xF7
1131
1131
1131      init_l4_rivets:                ; DATA XREF: 0000:0FD5|o
1131 21 F0 3D      ld      hl, #rivet_fireball_spr
1134 11 07 64      ld      de, #unk_0_6407
1137 01 1C 05      ld      bc, #0x51C              ; 5 sprites, offset 0x0c
113A CD 2A 12      call    init_data_for_B_sprites
113D 21 14 3E      ld      hl, #rivet_hammer_locs
1140 CD A6 11      call    init_hammer_sprites
1143 21 54 3E      ld      hl, #rivet_obj_spr
1146 11 0C 6A      ld      de, #soft_sprite_ram+0x10C ; sprite #67-69
1149 01 0C 00      ld      bc, #0xC              ; 0x0c bytes = 3 sprites
114C ED B0      ldir
114E 21 82 11      ld      hl, #rivet_unk_obj_locs
1151 11 A3 64      ld      de, #unk_0_64A3
1154 01 1E 02      ld      bc, #0x21E              ; 2 sprites, offset 0x20
1157 CD EC 11      call    init_objects_locations
115A 21 7E 11      ld      hl, #rivet_unk_sprites
115D 11 A7 64      ld      de, #unk_0_64A7
1160 01 1C 02      ld      bc, #0x21C              ; 2 sprites, offset $20
1163 CD 2A 12      call    init_data_for_B_sprites
1166 DD 21 A0 64      ld      ix, #unk_0_64A0
116A DD 36 00 01      ld      0(ix), #1
116E DD 36 20 01      ld      0x20(ix), #1
1172 21 50 69      ld      hl, #soft_sprite_ram+0x50 ; sprite #20-21
1175 06 02          ld      b, #2                ; 2 sprites
1177 11 20 00      ld      de, #0x20 ; ''              ; offset 0x20
117A CD D3 11      call    set_B_sprites_data
117D C9          ret
117D
117E 3F 0C 08 08 rivet_unk_sprites:.db 0x3F, 0xC, 8, 8 ; DATA XREF: 0000:115A|o
117E ; transparent squares over kong's legs
1182 73 50 8D 50 rivet_unk_obj_locs:.db 0x73, 0x50, 0x8D, 0x50 ; DATA XREF: 0000:114E|o
1186
1186      ; SUBROUTINE
1186
1186
1186      init_spring_sprites:                ; CODE XREF: 0000:102B|p
1186 ; 0000:1093|p
1186      ld      hl, #elevator_bouncing_spr
1189 11 07 65      ld      de, #unk_0_6507
118C 01 0C 0A      ld      bc, #0xA0C
118F CD 2A 12      call    init_data_for_B_sprites
1192 DD 21 00 65      ld      ix, #unk_0_6500
1196 21 80 69      ld      hl, #soft_sprite_ram+0x80 ; sprites 20-29
1199 06 0A          ld      b, #0xA
119B 11 10 00      ld      de, #0x10
119E CD D3 11      call    set_B_sprites_data
11A1 C9          ret
11A1
11A1      ; End of function init_spring_sprites
11A1
11A1
11A2 3B 00 02 02 elevator_bouncing_spr:.db 0x3B, 0, 2, 2 ; DATA XREF: init_spring_sprites|o
11A6
11A6      ; SUBROUTINE
11A6
11A6
11A6      init_hammer_sprites:                ; CODE XREF: 0000:1003|p
11A6 ; 0000:1073|p ...
11A6 ; object XPOS
11A6 ; 2 sprites, offset=14
11A9 01 0E 02      ld      de, #unk_0_6683
11AC CD EC 11      call    init_objects_locations
11AF 21 08 3E      ld      hl, #hammer_pickup_spr
11B2 11 87 66      ld      de, #unk_0_6687 ; object tile
11B5 01 0C 02      ld      bc, #0x20C ; 2 sprites, offset inc=0x0C
11B8 CD 2A 12      call    init_data_for_B_sprites
11BB DD 21 80 66      ld      ix, #unk_0_6680
11BF DD 36 00 01      ld      0(ix), #1
11C3 DD 36 10 01      ld      0x10(ix), #1
11C7 21 18 6A      ld      hl, #soft_sprite_ram+0x118 ; sprite #70
11CA 06 02          ld      b, #2
11CC 11 10 00      ld      de, #0x10
11CF CD D3 11      call    set_B_sprites_data
11D2 C9          ret
11D2
11D2      ; End of function init_hammer_sprites
11D2
11D3
11D3      ; SUBROUTINE
11D3
11D3
11D3      set_B_sprites_data:                ; CODE XREF: 0000:1046|p
11D3 ; 0000:10DB|p ...
11D3      ld      a, 3(ix)
11D6 77          ld      (hl), a                ; set sprite X
11D7 2C          inc      l
11D8 DD 7E 07      ld      a, 7(ix)
11DB 77          ld      (hl), a                ; set sprite tile
11DC 2C          inc      l
11DD DD 7E 08      ld      a, 8(ix)

```

```

11E0 77          ld      (hl), a                ; set sprite vflip/palette
11E1 2C          inc     l
11E2 DD 7E 05    ld      a, 5(ix)
11E5 77          ld      (hl), a                ; set sprite Y
11E6 2C          inc     l
11E7 DD 19       add     ix, de                ; next sprite data address
11E9 10 E8       djnz    set_B_sprites_data
11EB C9         ret
11EB           ; End of function set_B_sprites_data
11EC
11EC           ; ██████████ SUBROUTINE ██████████
11EC
11EC           init_objects_locations:          ; CODE XREF: 0000:10C0|p
11EC 7E          ; 0000:1157|p ...
11EC          ld      a, (hl)
11ED 12          ld      (de), a                ; copy byte 1
11EE 23          inc     hl                    ; next source byte
11EF 1C          inc     e
11F0 1C          inc     e                    ; skips destination byte
11F1 7E          ld      a, (hl)
11F2 12          ld      (de), a                ; copy byte 2
11F3 23          inc     hl                    ; next source byte
11F4 7B          ld      a, e
11F5 81          add     a, c
11F6 5F          add     e, a                ; add offset to destination
11F7 10 F3       djnz    init_objects_locations ; loop B times
11F9 C9         ret
11F9           ; End of function init_objects_locations
11FA
11FA           ; ██████████ SUBROUTINE ██████████
11FA
11FA           init_fireball_sprite:           ; CODE XREF: 0000:0FF2|p
11FA DD 21 A0 66 ; 0000:104C|p
11FA          ld      ix, #unk_0_66A0
11FE 11 28 6A    ld      de, #soft_sprite_ram+0x128 ; sprite #74
1201 DD 36 00 01 ;
1205 7E          ld      a, (hl)                ; Y pos
1206 DD 77 03    ld      3(ix), a
1209 12          ld      (de), a                ; sprite Y pos
120A 1C          inc     e                    ; next sprite register
120B 23          inc     hl                    ; next data byte
120C 7E          ld      a, (hl)                ; flipy,tile
120D DD 77 07    ld      7(ix), a
1210 12          ld      (de), a                ; sprite flipy,tile
1211 1C          inc     e                    ; next sprite register
1212 23          inc     hl                    ; next data byte
1213 7E          ld      a, (hl)                ; flipx,colour
1214 DD 77 08    ld      8(ix), a
1217 12          ld      (de), a                ; sprite flipx,colour
1218 1C          inc     e                    ; next sprite register
1219 23          inc     hl                    ; next data byte
121A 7E          ld      a, (hl)                ; X pos
121B DD 77 05    ld      5(ix), a
121E 12          ld      (de), a                ; sprite X pos
121F 23          inc     hl                    ; next data byte
1220 7E          ld      a, (hl)
1221 DD 77 09    ld      9(ix), a
1224 23          inc     hl                    ; next data byte
1225 7E          ld      a, (hl)
1226 DD 77 0A    ld      0xA(ix), a
1229 C9         ret
1229           ; End of function init_fireball_sprite
122A
122A           ; ██████████ SUBROUTINE ██████████
122A
122A           init_data_for_B_sprites:         ; CODE XREF: 0000:0FEC|p
122A E5          ; 0000:100F|p ...
122A          push    hl
122B C5          push    bc
122C 06 04       ld      b, #4                ; 4 bytes/sprite
122E
122E           loc_0_122E:                     ; CODE XREF: init_data_for_B_sprites+8|j
122E 7E          ld      a, (hl)
122F 12          ld      (de), a
1230 23          inc     hl
1231 1C          inc     e
1232 10 FA       djnz    loc_0_122E            ; copy data for 1 sprite
1234 C1         pop     bc
1235 E1         pop     hl                    ; restore source
1236 7B          ld      a, e
1237 81          add     a, c
1238 5F          add     e, a                ; next destination
1239 10 EF       djnz    init_data_for_B_sprites ; do B sprites
123B C9         ret
123B           ; End of function init_data_for_B_sprites
123B
123C           ;
123C
123C           init_mario:                     ; DATA XREF: 0000:0718|o
123C DF          ; 0000:074C|o
123C          rst      0x18                    ; wait for 8-bit countdown
123D 3A 27 62    ld      a, (level_type)
1240 FE 03       cp      #3                    ; elevators?
1242 01 16 E0    ld      bc, #0xE016           ; mario x,y coords
1245 CA 4B 12    jp      Z, loc_0_124B        ; yes, skip
1248 01 3F F0    ld      bc, #0xF03F           ; mario x,y coords
124B
124B           loc_0_124B:                     ; CODE XREF: 0000:1245|j
124B DD 21 00 62 ;
124F 21 4C 69    ld      ix, #mario_alive_flag
1252 DD 36 00 01 ;
1256 DD 71 03    ld      0(ix), #1            ; flag mario is alive
1259 71          ld      3(ix), c              ; mario y coord (X)
125A 2C          ld      (hl), c              ; sprite y = mario X
125B DD 36 07 80 ;
125F 36 80       ld      7(ix), #0x80 ; 'Ç'    ; sprite #19, flipy & code
1261 2C          inc     l                    ; flipy & tile=0
1262 DD 36 08 02 ;
1266 36 02       ld      (hl), #0x80 ; 'Ç'    ; flipy & tile=0
1268 2C          inc     l                    ; sprite #19, flipx & colour
1268          ld      8(ix), #2                ; no flipx, colour=2
1268          ld      (hl), #2                ; no flipx, colour=2
1268          ld      1
1268          ; sprite #19, x coord

```

```

1269 DD 70 05      ld      5(ix), b      ; mario x coord (Y)
126C 70           ld      (hl), b      ; x coord
126D DD 36 0F 01   ld      0xF(ix), #1
1271 21 0A 06      ld      hl, #main_sequencer
1274 34           inc      (hl)      ; next sequence (3)
1275 11 01 06      ld      de, #0x601    ; display_lives_and_level
1278 CD 9F 30      call    queue_fg_vector_fn
127B C9           ret
127C
127C
127C died_in_gameplay:      ; DATA XREF: 0000:071C|o
127C           ; 0000:0750|o
127C CD BD 1D      call    check_and_handle_bonus
127F 3A 9D 63      ld      a, (mario_death_state)
1282 EF           rst      0x28      ; go!
1282
1283 8B 12           .dw delay_before_spin      ; Jump Table
1285 AC 12           .dw mario_death_spin
1287 DE 12           .dw dead_mario_lying_down
1289 00 00           .dw 0
128B
128B
128B delay_before_spin:      ; DATA XREF: 0000:1283|o
128B DF           rst      0x18      ; wait for 8-bit countdown
128C 21 4D 69      ld      hl, #soft_sprite_ram+0x4D ; sprite #19, tile
128F 3E F0         ld      a, #0xF0 ; '-'
1291 CB 16         rl       (hl)      ; mario sprite << 1
1293 1F           rra
1294 77           ld      (hl), a
1295 21 9D 63      ld      hl, #mario_death_state
1298 34           inc      (hl)      ; next death_state
1299 3E 0D         ld      a, #0xD
129B 32 9E 63      ld      (death_spin_counter), a
129E 3E 08         ld      a, #8
12A0 32 09 60      ld      (eight_bit_countdown), a
12A3 CD BD 30      call    hide_object_sprites
12A6 3E 03         ld      a, #3
12A8 32 88 60      ld      (music_something), a
12AB C9           ret
12AC
12AC
12AC mario_death_spin:      ; DATA XREF: 0000:1285|o
12AC DF           rst      0x18      ; wait for 8-bit countdown
12AD 3E 08         ld      a, #8
12AF 32 09 60      ld      (eight_bit_countdown), a
12B2 21 9E 63      ld      hl, #death_spin_counter
12B5 35           dec      (hl)
12B6 CA CB 12      jp      Z, finish_death_spin
12B9 21 4D 69      ld      hl, #soft_sprite_ram+0x4D ; sprite #19 (mario)
12BC 7E           ld      a, (hl)      ; get flipy & code
12BD 1F           rra      ; lsb to C
12BE 3E 02         ld      a, #2      ; sprite #1 <<1
12C0 1F           rra      ; lsb to flipy
12C1 47           ld      b, a
12C2 AE           xor      (hl)
12C3 77           ld      (hl), a      ; invert tile & flipy
12C4 2C           inc      l      ; flipx & colour
12C5 78           ld      a, b
12C6 E6 80         and      #0x80 ; 'Ç'
12C8 AE           xor      (hl)      ; flipy only
12C9 77           ld      (hl), a      ; invert flip
12CA C9           ret
12CB
12CB
12CB finish_death_spin:      ; CODE XREF: 0000:12B6|j
12CB 21 4D 69      ld      hl, #soft_sprite_ram+0x4D ; sprite #19 (mario)
12CE 3E F4         ld      a, #0xF4 ; '¶'
12D0 CB 16         rl       (hl)      ; mario dead sprite <<1
12D2 1F           rra      ; flipy to C
12D3 77           ld      (hl), a      ; restore flipy
12D4 21 9D 63      ld      hl, #mario_death_state ; update sprite
12D7 34           inc      (hl)
12D8 3E 80         ld      a, #0x80 ; 'Ç'
12DA 32 09 60      ld      (eight_bit_countdown), a
12DD C9           ret
12DE
12DE
12DE dead_mario_lying_down: ; DATA XREF: 0000:1287|o
12DE DF           rst      0x18      ; wait for 8-bit countdown
12DF CD DB 30      call    sub_0_30DB
12E2 21 0A 60      ld      hl, #main_sequencer
12E5 3A 0E 60      ld      a, (current_player_E)
12E8 A7           and      a      ; player 1?
12E9 CA ED 12      jp      Z, loc_0_12ED ; yes, skip
12EC 34           inc      (hl)
12ED
12ED loc_0_12ED:      ; CODE XREF: 0000:12E9|j
12ED 34           inc      (hl)
12EE 2B           dec      hl
12EF 36 01         ld      (hl), #1      ; eight_bit_countdown
12F1 C9           ret
12F2
12F2
12F2 save_P1_ingame_data:      ; DATA XREF: 0000:071E|o
12F2 CD 1C 01      call    stop_sound
12F5 AF           xor      a
12F6 32 2C 62      ld      (seen_intro), a
12F9 21 28 62      ld      hl, #lives_left
12FC 35           dec      (hl)
12FD 7E           ld      a, (hl)
12FE 11 40 60      ld      de, #p1_ingame_data
1301 01 08 00      ld      bc, #8      ; 8 bytes to copy
1304 ED B0         ldir
1306 A7           and      a      ; mario alive?
1307
1307 loc_0_1307:      ; yes, skip
1307 C2 34 13      jp      NZ, loc_0_1334
130A 3E 01         ld      a, #1
130C 21 B2 60      ld      hl, #p1_score
130F CD CA 13      call    sub_0_13CA      ; flag P1 score
1312 21 D4 76      ld      hl, #VRAM_start+0x2D4
1315 3A 0F 60      ld      a, (two_players)
1318 A7           and      a      ; 2 players?
1319 28 07         jr      Z, loc_0_1322 ; no, skip
131B 11 02 03      ld      de, #0x302
131E CD 9F 30      call    queue_fg_vector_fn

```

```

1321 2B          dec     hl
1322
1322          loc_0_1322:                                ; CODE XREF: 0000:1319|j
1322          call    clear_14x5_HL
1325 11 00 03    ld      de, #0x300                      ; display_message_00 "GAME OVER"
1328 CD 9F 30    call    queue_fg_vector_fn
132B 21 09 60    ld      hl, #eight_bit_countdown
132E 36 C0       ld      (hl), #0xC0 ; 'L'
1330 23          inc     hl
1331 36 10       ld      (hl), #0x10
1333 C9          ret
1334
1334          ;
1334          loc_0_1334:                                ; CODE XREF: 0000:1307|j
1334          ld      c, #8
1336 3A 0F 60    ld      a, (two_players)
1339 A7          and     a
133A CA 3F 13    jp      Z, loc_0_133F                    ; two players?
133D 0E 17       ld      c, #0x17                        ; no, skip
133F          ; next sequence (23)
133F          loc_0_133F:                                ; CODE XREF: 0000:133A|j
133F 79          ld      a, c
1340 32 0A 60    ld      (main_sequencer), a
1343 C9          ret
1344
1344          ;
1344          save_P2_ingame_data:                        ; DATA XREF: 0000:0720|o
1344          call    stop_sound
1347 AF          xor     a
1348 32 2C 62    ld      (seen_intro), a
134B 21 28 62    ld      hl, #lives_left
134E 35          dec     (hl)
134F 7E          ld      a, (hl)
1350 11 48 60    ld      de, #p2_ingame_data
1353 01 08 00    ld      bc, #8
1356 ED B0       ldir
1358 A7          and     a
1359 C2 7F 13    jp      NZ, loc_0_137F                    ; mario alive?
135C 3E 03       ld      a, #3
135E 21 B5 60    ld      hl, #p2_score
1361 CD CA 13    call    sub_0_13CA
1364 11 03 03    ld      de, #0x303
1367 CD 9F 30    call    queue_fg_vector_fn
136A 11 00 03    ld      de, #0x300
136D CD 9F 30    call    queue_fg_vector_fn
1370 21 D3 76    ld      hl, #VRAM_start+0x2D3
1373 CD 26 18    call    clear_14x5_HL
1376 21 09 60    ld      hl, #eight_bit_countdown
1379 36 C0       ld      (hl), #0xC0 ; 'L'
137B 23          inc     hl
137C 36 11       ld      (hl), #0x11
137E C9          ret
137F
137F          ;
137F          loc_0_137F:                                ; CODE XREF: 0000:1359|j
137F 0E 17       ld      c, #0x17
1381 3A 40 60    ld      a, (p1_ingame_data)
1384 A7          and     a
1385 C2 8A 13    jp      NZ, loc_0_138A                    ; set to switch players?
1388 0E 08       ld      c, #8
138A          ; mario alive P1?
138A          loc_0_138A:                                ; yes, skip
138A 79          ld      a, c
138B 32 0A 60    ld      (main_sequencer), a
138E C9          ret
138F
138F          ;
138F          p1_game_over:                              ; CODE XREF: 0000:1385|j
138F DF          rst     0x18
1390 0E 17       ld      c, #0x17
1392 3A 48 60    ld      a, (p2_ingame_data)
1395          ; wait for 8-bit countdown
1395          ; set to switch players?
1395          loc_0_1395:                                ; DATA XREF: 0000:0722|o
1395          inc     (hl)
1396 A7          and     a
1397 C2 9C 13    jp      NZ, loc_0_139C                    ; adjust countdown
139A 0E 14       ld      c, #0x14
139C          ; mario alive P2?
139C          loc_0_139C:                                ; yes, skip
139C 79          ld      a, c
139D 32 0A 60    ld      (main_sequencer), a
13A0 C9          ret
13A1
13A1          ;
13A1          p2_game_over:                              ; CODE XREF: 0000:1397|j
13A1 DF          rst     0x18
13A2 0E 17       ld      c, #0x17
13A4 3A 40 60    ld      a, (p1_ingame_data)
13A7 C3 95 13    jp      loc_0_1395
13AA          ; next sequence (8)
13AA          ; CODE XREF: 0000:1385|j
13AA          loc_0_138A:                                ; DATA XREF: 0000:0724|o
13AA          rst     0x18
13AA          ; wait for 8-bit countdown
13AA          ; set to switch players?
13AA          loc_0_1395:                                ; DATA XREF: 0000:13A7|j
13AA          inc     (hl)
13AA          ; adjust countdown
13AA          ; mario alive P2?
13AA          loc_0_139C:                                ; yes, skip
13AA          ld      a, c
13AA          (main_sequencer), a
13AA          ret
13AA
13AA          ;
13AA          set_flip_and_current_P2:                    ; DATA XREF: 0000:0726|o
13AA          ld      a, (upright)
13AD 32 82 7D    ld      (flipscreen), a
13B0 AF          xor     a
13B1 32 0A 60    ld      (main_sequencer), a
13B4 21 01 01    ld      hl, #0x101
13B7 22 0D 60    ld      (current_player_D), hl
13BA C9          ret
13BB
13BB          ;
13BB          set_flip_and_current_P1:                    ; DATA XREF: 0000:0728|o
13BB AF          xor     a
13BC 32 0D 60    ld      (current_player_D), a
13BF 32 0E 60    ld      (current_player_E), a
13C2 32 0A 60    ld      (main_sequencer), a
13C5 3C          inc     a
13C6 32 82 7D    ld      (flipscreen), a
13C9 C9          ret
13CA
13CA          ;
13CA          sub_0_13CA:                                ; CODE XREF: 0000:130F|p
13CA          ld      de, #unk_0_61C6
13CA          ; 0000:1361|p

```

```

13CD 12      ld      (de), a
13CE CF      rst      8
13CF 13      inc      de
13D0 01 03 00      ld      bc, #3
13D3 ED B0      ldir
13D5 06 03      ld      b, #3
13D7 21 B1 61      ld      hl, #unk_0_61B1
13DA
13DA          loc_0_13DA:
13DA 1B      dec      de
13DB 1A      ld      a, (de)
13DC 0F      rrca
13DD 0F      rrca
13DE 0F      rrca
13DF 0F      rrca
13E0 E6 0F      and      #0xF
13E2 77      ld      (hl), a
13E3 23      inc      hl
13E4 1A      ld      a, (de)
13E5 E6 0F      and      #0xF
13E7 77      ld      (hl), a
13E8 23      inc      hl
13E9 10 EF      djnz     loc_0_13DA
13EB 06 0E      ld      b, #0xE
13ED
13ED          loc_0_13ED:
13ED 36 10      ld      (hl), #0x10
13EF 23      inc      hl
13F0 10 FB      djnz     loc_0_13ED
13F2 36 3F      ld      (hl), #0x3F ; '?'
13F4 06 05      ld      b, #5
13F6 21 A5 61      ld      hl, #hs_tbl_5th+0x1D
13F9 11 C7 61      ld      de, #unk_0_61C7
13FC
13FC          loc_0_13FC:
13FC 1A      ld      a, (de)
13FD 96      sub      (hl)
13FE 23      inc      hl
13FF 13      inc      de
1400 1A      ld      a, (de)
1401 9E      sbc      a, (hl)
1402 23      inc      hl
1403 13      inc      de
1404 1A      ld      a, (de)
1405 9E      sbc      a, (hl)
1406 D8      ret      C
1407 C5      push     bc
1408 06 19      ld      b, #0x19
140A
140A          loc_0_140A:
140A 4E      ld      c, (hl)
140B 1A      ld      a, (de)
140C 77      ld      (hl), a
140D 79      ld      a, c
140E 12      ld      (de), a
140F 2B      dec      hl
1410 1B      dec      de
1411 10 F7      djnz     loc_0_140A
1413 01 F5 FF      ld      bc, #0xFFFF
1416 09      add      hl, bc
1417 EB      ex      de, hl
1418 09      add      hl, bc
1419 EB      ex      de, hl
141A C1      pop      bc
141B 10 DF      djnz     loc_0_13FC
141D C9      ret
141D          ; End of function sub_0_13CA
141D
141E          ; -----
141E
141E          draw_name_registered:
141E CD 16 06      call     display_credits
1421 DF      rst      0x18
1422 CD 74 08      call     clear_visible_area_and_sprites
1425 3E 00      ld      a, #0
1427 32 0E 60      ld      (current_player_E), a
142A 32 0D 60      ld      (current_player_D), a
142D 21 1C 61      ld      hl, #high_score_tbl_ram+0x1C
1430 11 22 00      ld      de, #0x22 ; ''
1433 06 05      ld      b, #5
1435 3E 01      ld      a, #1
1437
1437          loc_0_1437:
1437 BE      cp      (hl)
1438 CA 59 14      jp      Z, display_name_registration_msgs
143B 19      add      hl, de
143C 10 F9      djnz     loc_0_1437
143E 21 1C 61      ld      hl, #high_score_tbl_ram+0x1C
1441 06 05      ld      b, #5
1443 3E 03      ld      a, #3
1445
1445          loc_0_1445:
1445 BE      cp      (hl)
1446 CA 4F 14      jp      Z, registration_set_P2
1449 19      add      hl, de
144A 10 F9      djnz     loc_0_1445
144C C3 75 14      jp      exit_name_entry
144F
144F          registration_set_P2:
144F 3E 01      ld      a, #1
1451 32 0E 60      ld      (current_player_E), a
1454 32 0D 60      ld      (current_player_D), a
1457 3E 00      ld      a, #0
1459
1459          display_name_registration_msgs:
1459 21 26 60      ld      hl, #upright
145C B6      or      (hl)
145D 32 82 7D      ld      (flipscreen), a
1460 3E 00      ld      a, #0
1462 32 09 60      ld      (eight_bit_countdown), a
1465 21 0A 60      ld      hl, #main_sequencer
1468 34      inc      (hl)
1469 11 0D 03      ld      de, #0x30D
146C 06 0C      ld      b, #0xC
146E

```

```

146E loc_0_146E: call queue_fg_vector_fn ; CODE XREF: 0000:1472|j
146E CD 9F 30 inc de
1471 13 djnz loc_0_146E
1472 10 FA ret
1474 C9
1475 ;
1475
1475 exit_name_entry: ; CODE XREF: 0000:144C|j
1475 3E 01 ld a, #1
1477 32 82 7D ld (flipscreen), a
147A 32 05 60 ld (nmi_sequencer), a
147D 32 07 60 ld (attract_mode_flag), a ; set attract mode flag
1480 3E 00 ld a, #0
1482 32 0A 60 ld (main_sequencer), a
1485 C9 ret
1486 ;
1486
1486 do_initials_entry: ; DATA XREF: 0000:072C|o
1486 CD 16 06 call display_credits
1489 21 09 60 ld hl, #eight_bit_countdown
148C 7E ld a, (hl)
148D A7 and a
148E C2 DC 14 jp NZ, loc_0_14DC
1491 32 86 7D ld (palette_bank), a
1494 32 87 7D ld (palette_bank+1), a ; set palette 0
1497 36 01 ld (hl), #1
1499 21 30 60 ld hl, #unk_0_6030
149C 36 0A ld (hl), #0xA
149E 23 inc hl
149F 36 00 ld (hl), #0
14A1 23 inc hl
14A2 36 10 ld (hl), #0x10
14A4 23 inc hl
14A5 36 1E ld (hl), #30 ; regi_seconds_cntr
14A7 23 inc hl
14A8 36 3E ld (hl), #62 ; regi_vblank_cntr
14AA 23 inc hl
14AB 36 00 ld (hl), #0 ; regi_current_char
14AD 21 E8 75 ld hl, #VRAM_start+0x1E8
14B0 22 36 60 ld (regi_entry_cursor_loc), hl ; init cursor loc for 1st character
14B3 21 1C 61 ld hl, #high_score_tbl_ram+0x1C
14B6 3A 0E 60 ld a, (current_player_E) ; 0/1
14B9 07 rlica ; 0/2
14BA 3C inc a ; 1/3
14BB 4F ld c, a ; P1/P2 high score flag
14BC 11 22 00 ld de, #0x22 ; '' ; score offset
14BF 06 04 ld b, #4 ; 4 scores to check
14C1
14C1 loc_0_14C1: ; CODE XREF: 0000:14C7|j
14C1 7E ld a, (hl) ; get flag
14C2 B9 cp c ; P1/P2 high score?
14C3 CA C9 14 jp Z, loc_0_14C9 ; yes, skip
14C6 19 add hl, de ; next entry
14C7 10 F8 djnz loc_0_14C1
14C9
14C9 loc_0_14C9: ; CODE XREF: 0000:14C3|j
14C9 22 38 60 ld (regi_ptr_hs_entry_flag), hl ; point to high score entry
14CC 11 F3 FF ld de, #0xFFFF3
14CF 19 add hl, de ; offset for name
14D0 22 3A 60 ld (regi_ptr_hs_entry_name), hl ; store ptr to name
14D3 06 00 ld b, #0
14D5 3A 35 60 ld a, (regi_current_char)
14D8 4F ld c, a
14D9 CD FA 15 call outline_letter ; high score initial select sprite
14DC
14DC loc_0_14DC: ; CODE XREF: 0000:148E|j
14DC 21 34 60 ld hl, #regi_vblank_cntr
14DF 35 dec (hl) ; done another second?
14E0 C2 FC 14 jp NZ, regi_read_controller_input ; no, skip
14E3 36 3E ld (hl), #62 ; reset to roughly 1s
14E5 2B dec hl ; regi_second_cntr
14E6 35 dec (hl) ; out of time?
14E7 CA C6 15 jp Z, regi_save_hs_name ; yes, skip
14EA 7E ld a, (hl) ; seconds left
14EB 06 FF ld b, #0xFF
14ED
14ED regi_show_seconds_left: ; CODE XREF: 0000:14F0|j
14ED 04 inc b
14EE D6 0A sub #0xA
14F0 D2 ED 14 jp NC, regi_show_seconds_left ; divide by 10
14F3 C6 0A add a, #0xA ; fix last subtraction (units)
14F5 32 52 75 ld (VRAM_start+0x152), a ; units digit (time left)
14F8 78 ld a, b
14F9 32 72 75 ld (VRAM_start+0x172), a ; tens digit (time left)
14FC
14FC regi_read_controller_input: ; CODE XREF: 0000:14E0|j
14FC 21 30 60 ld hl, #unk_0_6030 ; (not used???)
14FF 46 ld b, (hl)
1500 36 0A ld (hl), #0xA
1502 3A 10 60 ld a, (controller_in) ; edge-detected inputs
1505 CB 7F bit 7, a ; button pressed?
1507 C2 46 15 jp NZ, regi_jump_pressed ; yes, skip
150A E6 03 and #3 ; left/right only
150C C2 14 15 jp NZ, regi_left_right_pressed ; yes, skip
150F 3C inc a ; A=1
1510 77 ld (hl), a
1511 C3 8A 15 jp loc_0_158A
1514 ;
1514
1514 regi_left_right_pressed: ; CODE XREF: 0000:150C|j
1514 05 dec b
1515 CA 1D 15 jp Z, loc_0_151D
1518 78 ld a, b
1519 77 ld (hl), a
151A C3 8A 15 jp loc_0_158A
151D ;
151D
151D loc_0_151D: ; CODE XREF: 0000:1515|j
151D CB 4F bit 1, a ; left?
151F C2 39 15 jp NZ, regi_previous_character ; yes, skip
1522 3A 35 60 ld a, (regi_current_char)
1525 3C inc a ; next character
1526 FE 1E cp #0x1E ; last character?
1528 C2 2D 15 jp NZ, loc_0_152D ; no, skip
152B 3E 00 ld a, #0 ; set to 1st character
152D

```



```

152D      loc_0_152D:                                     ; CODE XREF: 0000:1528|j
152D 32 35 60                                           ; 0000:153E|j ...
152D      ld      (regi_current_char), a                ; save new character
1530 4F      ld      c, a                                ; prepare to display
1531 06 00      ld      b, #0
1533 CD FA 15      call  outline_letter
1536 C3 8A 15      jp      loc_0_158A
1539      ;
1539
1539      regi_previous_character:                         ; CODE XREF: 0000:151F|j
1539 3A 35 60      ld      a, (regi_current_char)
153C D6 01      sub      #1                                ; previous character
153E F2 2D 15      jp      P, loc_0_152D                            ; not 0, skip
1541 3E 1D      ld      a, #0x1D                          ; set to last character
1543 C3 2D 15      jp      loc_0_152D                            ; continue
1546      ;
1546
1546      regi_jump_pressed:                               ; CODE XREF: 0000:1507|j
1546 3A 35 60      ld      a, (regi_current_char)
1549 FE 1C      cp      #0x1C                            ; RUB?
154B CA 6D 15      jp      Z, regi_rub                            ; yes, skip
154E FE 1D      cp      #0x1D                            ; END?
1550 CA C6 15      jp      Z, regi_save_hs_name                ; yes, skip
1553 2A 36 60      ld      hl, (regi_entry_cursor_loc)        ; get current location
1556 01 88 75      ld      bc, #VRAM_start+0x188
1559 A7      and      a
155A ED 42      sbc      hl, bc
155C CA 8A 15      jp      Z, loc_0_158A
155F 09      add      hl, bc
1560 C6 11      add      a, #0x11                            ; convert to 'ascii'
1562      loc_0_1562:                                     ; display character
1562 77      ld      (hl), a
1563 01 E0 FF      ld      bc, #0xFFE0                            ; next column
1566 09      add      hl, bc
1567      ;
1567      regi_update_cursor:                               ; CODE XREF: 0000:1583|j
1567 22 36 60      ld      (regi_entry_cursor_loc), hl        ; store next location
156A C3 8A 15      jp      loc_0_158A
156D      ;
156D
156D      regi_rub:                                       ; CODE XREF: 0000:154B|j
156D 2A 36 60      ld      hl, (regi_entry_cursor_loc)
1570 01 20 00      ld      bc, #0x20 ; ' '                            ; previous column
1573 09      add      hl, bc                                ; adjust
1574 A7      and      a
1575 01 08 76      ld      bc, #VRAM_start+0x208
1578 ED 42      sbc      hl, bc                                ; first character?
157A C2 86 15      jp      NZ, loc_0_1586                            ; no, skip
157D 21 E8 75      ld      hl, #VRAM_start+0x1E8
1580      ;
1580      regi_erase_char:                               ; CODE XREF: 0000:1587|j
1580 3E 10      ld      a, #0x10                            ; space
1582 77      ld      (hl), a                                ; display
1583 C3 67 15      jp      regi_update_cursor
1586      ;
1586
1586      loc_0_1586:                                       ; CODE XREF: 0000:157A|j
1586 09      add      hl, bc
1587 C3 80 15      jp      regi_erase_char
158A      ;
158A
158A      loc_0_158A:                                       ; CODE XREF: 0000:1511|j
158A 21 32 60      ld      hl, #byte_0_6032                    ; 0000:151A|j ...
158D 35      dec      (hl)
158E C2 F9 15      jp      NZ, locret_0_15F9
1591 3A 31 60      ld      a, (byte_0_6031)
1594 A7      and      a
1595 C2 B8 15      jp      NZ, loc_0_15B8
1598 3E 01      ld      a, #1
159A 32 31 60      ld      (byte_0_6031), a
159D 11 BF 01      ld      de, #byte_0_1BD+2                    ; empty/dummy score
15A0      ;
15A0      loc_0_15A0:                                       ; CODE XREF: 0000:15C3|j
15A0 FD 2A 38 60      ld      iy, (regi_ptr_hs_entry_flag)        ; ptr high score
15A4 FD 6E 04      ld      l, 4(iy)
15A7 FD 66 05      ld      h, 5(iy)
15AA E5      push     hl
15AB DD E1      pop      ix                                ; display location
15AD CD 7C 05      call  display_score_HL_at_IX                ; display new high score in table
15B0 3E 10      ld      a, #0x10
15B2 32 32 60      ld      (byte_0_6032), a
15B5 C3 F9 15      jp      locret_0_15F9
15B8      ;
15B8
15B8      loc_0_15B8:                                       ; CODE XREF: 0000:1595|j
15B8 AF      xor      a
15B9 32 31 60      ld      (byte_0_6031), a
15BC ED 5B 38 60      ld      de, (regi_ptr_hs_entry_flag)        ; point to high score
15C0 13      inc      de
15C1 13      inc      de
15C2 13      inc      de
15C3 C3 A0 15      jp      loc_0_15A0
15C6      ;
15C6
15C6      regi_save_hs_name:                               ; CODE XREF: 0000:14E7|j
15C6 ED 5B 38 60      ld      de, (regi_ptr_hs_entry_flag)        ; point to high score
15C6 AF      xor      a
15CB 12      ld      (de), a                                ; unflag as P1/P2 high score
15CC 21 09 60      ld      hl, #eight_bit_countdown
15CF 36 80      ld      (hl), #0x80 ; 'Ç'
15D1 23      inc      hl                                ; main_sequencer
15D2 35      dec      (hl)                                ; -1
15D3 06 0C      ld      b, #0xC                            ; 12 chars to copy
15D5 21 E8 75      ld      hl, #VRAM_start+0x1E8
15D8 FD 2A 3A 60      ld      iy, (regi_ptr_hs_entry_name)
15DC 11 E0 FF      ld      de, #0xFFE0
15DF      ;
15DF      loc_0_15DF:                                       ; CODE XREF: 0000:15E6|j
15DF 7E      ld      a, (hl)                                ; get name character from screen
15E0 FD 77 00      ld      0(iy), a                            ; store in hs entry
15E3 FD 23      inc      iy                                ; next position
15E5 19      add      hl, de                                ; next column on screen
15E6 10 F7      djnz   loc_0_15DF                            ; loop through 12 chars

```

```

15E8 06 05          ld      b, #5
15EA 11 14 03      ld      de, #0x314                ; display_message_14 "REGI TIME"
15ED
15ED          loc_0_15ED:
15ED CD 9F 30      call     queue_fg_vector_fn        ; CODE XREF: 0000:15F1|j
15F0 13          inc      de                        ; next message
15F1 10 FA      djnz     loc_0_15ED                ; display high score table
15F3 11 1A 03      ld      de, #0x31A                ; display_message_1A "YOUR NAME WAS REGISTERED"
15F6 CD 9F 30      call     queue_fg_vector_fn
15F9
15F9          locret_0_15F9:
15F9          ; CODE XREF: 0000:158E|j
15F9 C9          ret                                ; 0000:15B5|j
15F9
15FA          ; ██████████ S U B R O U T I N E ██████████
15FA
15FA          outline_letter:
15FA          ; CODE XREF: 0000:14D9|p
15FA          ; 0000:1533|p
15FB E5          push     de
15FC CB 21      push     hl
15FE 21 0F 36      sla      c
1601 09          ld      hl, #letter_coords
1602 EB          add      hl, bc
1603 21 74 69      ex       de, hl
1606 1A          ld      hl, #soft_sprite_ram+0x74        ; sprite #29 for initials entry
1607 13          ld      a, (de)
1608 77          inc      de
1609 23          ld      (hl), a                ; X coordinate
160A 36 72      inc      hl
160C 23          ld      (hl), #0x72 ; 'x'                ; tile
160D 36 0C      inc      hl
160F 23          ld      (hl), #0xC                ; palette
1610 1A          ld      a, (de)
1611 77          ld      (hl), a                ; Y coordinate
1612 E1          pop      hl
1613 D1          pop      de
1614 C9          ret
1614          ; End of function outline_letter
1614
1615          ;
1615
1615          mario_pauline_reunion:
1615          ; DATA XREF: 0000:072E|o
1618 CD BD 30      call     hide_object_sprites
161B 0F          ld      a, (level_type)
161C D2 2F 16      rrca
161F 3A 88 63      jp      NC, loc_0_162F
1622 EF          ld      a, (unk_0_6388)
1622          rst      0x28                ; go!
1622          ;
1623 54 16          .dw loc_0_1654                ; Jump table
1625 70 16          .dw loc_0_1670
1627 8A 16          .dw loc_0_168A
1629 32 17          .dw loc_0_1732
162B 57 17          .dw loc_0_1757
162D 8E 17          .dw loc_0_178E
162F
162F          loc_0_162F:
162F          ; CODE XREF: 0000:161C|j
162F 0F          rrca
1630 D2 41 16      jp      NC, loc_0_1641
1633 3A 88 63      ld      a, (unk_0_6388)
1636 EF          rst      0x28                ; go!
1636          ;
1637 A3 16          .dw loc_0_16A3                ; Jump table
1639 BB 16          .dw loc_0_16BB
163B 32 17          .dw loc_0_1732
163D 57 17          .dw loc_0_1757
163F 8E 17          .dw loc_0_178E
1641
1641          loc_0_1641:
1641          ; CODE XREF: 0000:1630|j
1641 CD BD 1D      call     check_and_handle_bonus
1644 3A 88 63      ld      a, (unk_0_6388)
1647 EF          rst      0x28                ; go!
1647          ;
1648 B6 17          .dw unk_0_17B6                ; Jump table
164A 69 30      .dw wait_and_inc_sequence
164C 39 18      .dw loc_0_1839
164E 6F 18      .dw loc_0_186F
1650 80 18      .dw loc_0_1880
1652 C6 18      .dw loc_0_18C6
1654
1654          loc_0_1654:
1654          ; DATA XREF: 0000:1623|o
1654 CD 08 17      call     sub_0_1708
1657 21 5C 38      ld      hl, #dk_normal_spr
165A CD 4E 00      call     copy_sprites_2_11_data
165D 3E 20      ld      a, #0x20 ; '-'
165F 32 09 60      ld      (eight_bit_countdown), a
1662
1662          loc_0_1662:
1662          ; CODE XREF: 0000:16A0|j
1662 21 88 63      ld      hl, #unk_0_6388
1665 34          inc      (hl)
1666 3E 01      ld      a, #1
1668 F7          rst      0x30                ; return if level bit not set
1669 21 0B 69      ld      hl, #soft_sprite_ram+0xB        ; sprite #2, x coord
166C 0E FC      ld      c, #0xFC ; '3'                ; -4
166E FF          rst      0x38                ; subtract 4 from x coord for 10 sprites
166F C9          ret
1670
1670          loc_0_1670:
1670          ; DATA XREF: 0000:1625|o
1670          ; wait for 8-bit countdown
1670 DF          rst      0x18
1671 21 32 39      ld      hl, #dk_throw_barrel_spr
1674 CD 4E 00      call     copy_sprites_2_11_data
1677 3E 20      ld      a, #0x20 ; '-'
1679 32 09 60      ld      (eight_bit_countdown), a
167C 21 88 63      ld      hl, #unk_0_6388
167F 34          inc      (hl)
1680 3E 04      ld      a, #4
1682 F7          rst      0x30                ; return if level bit not set
1683 21 0B 69      ld      hl, #soft_sprite_ram+0xB        ; sprite #2, x coord
1686 0E 04      ld      c, #4                ; +4
1688 FF          rst      0x38                ; add 4 to x coord for 10 sprites

```



```

1747 32 24 69      ld      (soft_sprite_ram+0x24), a
174A 3D            dec      a
174B 32 2C 69      ld      (soft_sprite_ram+0x2C), a
174E 21 21 6A      ld      hl, #soft_sprite_ram+0x121
1751 34            inc      hl
1752 21 88 63      ld      hl, #unk_0_6388
1755 34            inc      hl
1756 C9            ret
1757
1757
1757 loc_0_1757:      ; DATA XREF: 0000:162B|o
1757 CD 6F 30      ; 0000:163D|o
1757              call     animate_kong_climbing
175A CD 6C 17      call     sub_0_176C
175D 23            inc      hl
175E 13            inc      de
175F CD 83 17      call     sub_0_1783
1762 3E 40          ld      a, #0x40 ; '@'
1764 32 09 60      ld      (eight_bit_countdown), a
1767 21 88 63      ld      hl, #unk_0_6388
176A 34            inc      hl
176B C9            ret
176C
176C ; ██████████ S U B R O U T I N E ██████████
176C
176C
176C sub_0_176C:      ; CODE XREF: 0000:175A|p
176C 11 03 00      ld      de, #3
176F 21 2F 69      ld      hl, #soft_sprite_ram+0x2F
1772 06 0A          ld      b, #0xA
1774
1774 loc_0_1774:      ; CODE XREF: sub_0_176C+14|j
1774 A7            and      a
1775 7E            ld      a, (hl)
1776 ED 52          sbc      hl, de
1778 FE 19          cp      #0x19
177A D2 7F 17      jp      NC, loc_0_177F
177D 36 00          ld      (hl), #0
177F
177F loc_0_177F:      ; CODE XREF: sub_0_176C+E|j
177F 2B            dec      hl
1780 10 F2          djnz   loc_0_1774
1782 C9            ret
1782 ; End of function sub_0_176C
1782
1783
1783 ; ██████████ S U B R O U T I N E ██████████
1783
1783
1783 sub_0_1783:      ; CODE XREF: 0000:175F|p
1783 06 0A          ld      b, #0xA
1785
1785 loc_0_1785:      ; CODE XREF: sub_0_1783+8|j
1785 7E            ld      a, (hl)
1786 A7            and      a
1787 C2 26 00      jp      NZ, pop_hl_ret
178A 19            add      hl, de
178B 10 F8          djnz   loc_0_1785
178D C9            ret
178D ; End of function sub_0_1783
178D
178E
178E
178E loc_0_178E:      ; DATA XREF: 0000:162D|o
178E DF            ; 0000:163F|o
178E              rst      0x18
178E              ; wait for 8-bit countdown
178F 2A 2A 62      ld      hl, (seq_data)
1792 23            inc      hl
1793 7E            ld      a, (hl)
1794 FE 7F          cp      #0x7F ; ' '
1796 C2 9D 17      jp      NZ, loc_0_179D
1799 21 73 3A      ld      hl, #level_seq_2
179C 7E            ld      a, (hl)
179D              ; restart repeating levels?
179D              ; no, skip
179D              ; repeating levels
179D              ; get new level
179D
179D loc_0_179D:      ; CODE XREF: 0000:1796|j
179D 22 2A 62      ld      (seq_data), hl
17A0 32 27 62      ld      (level_type), a
17A3 11 00 05      ld      de, #0x500
17A6 CD 9F 30      call     queue_fg_vector_fn
17A9 AF            xor      a
17AA 32 88 63      ld      (unk_0_6388), a
17AD 21 09 60      ld      hl, #eight_bit_countdown
17B0 36 30          ld      (hl), #0x30 ; '0'
17B2 23            inc      hl
17B3 36 08          ld      (hl), #8
17B5 C9            ret
17B5 ; sequencer = how high screen
17B5
17B5
17B5 unk_0_17B6:      .db      0 ;
17B7
17B7 ;
17B7 CD 1C 01      call     stop_sound
17BA 21 8A 60      ld      hl, #unk_0_608A
17BD 36 0E          ld      (hl), #0xE
17BF 23            inc      hl
17C0 36 03          ld      (hl), #3
17C2 3E 10          ld      a, #0x10
17C4 11 20 00      ld      de, #0x20 ; ' '
17C7 21 23 76      ld      hl, #VRAM_start+0x223
17CA CD 14 05      call     display_3_tiles_HL
17CD 21 83 75      ld      hl, #VRAM_start+0x183
17D0 CD 14 05      call     display_3_tiles_HL
17D3 21 DA 76      ld      hl, #VRAM_start+0x2DA
17D6 CD 26 18      call     clear_14x5_HL
17D9 11 47 3A      ld      de, #draw_data_rivet_end1
17DC CD A7 0D      call     draw_level_background
17DF 21 D5 76      ld      hl, #VRAM_start+0x2D5
17E2 CD 26 18      call     clear_14x5_HL
17E5 11 4D 3A      ld      de, #draw_data_rivet_end2
17E8 CD A7 0D      call     draw_level_background
17EB 21 D0 76      ld      hl, #VRAM_start+0x2D0
17EE CD 26 18      call     clear_14x5_HL
17F1 11 53 3A      ld      de, #draw_data_rivet_end3
17F4 CD A7 0D      call     draw_level_background
17F7 21 CB 76      ld      hl, #VRAM_start+0x2CB
17FA CD 26 18      call     clear_14x5_HL
17FD 11 59 3A      ld      de, #draw_data_rivet_end4
1800 CD A7 0D      call     draw_level_background

```

```

1803 21 5C 38      ld      hl, #dk_normal_spr
1806 CD 4E 00      call    copy_sprites_2_l1_data
1809 21 08 69      ld      hl, #soft_sprite_ram+8          ; sprite #2, y coord
180C 0E 44         ld      c, #68
180E FF           rst      0x38              ; add 68 to y coord for 10 sprites
180F 21 05 69      ld      hl, #soft_sprite_ram+5
1812 36 13         ld      (hl), #0x13
1814 3E 20         ld      a, #0x20 ; ' '
1816 32 09 60      ld      (eight_bit_countdown), a
1819 3E 80         ld      a, #0x80 ; 'C'
181B 32 90 63      ld      (kong_thrash_tmr), a
181E 21 88 63      ld      hl, #unk_0_6388
1821 34           inc      (hl)
1822 22 C0 63      ld      (ptr_current_sequence), hl
1825 C9           ret

1826
1826 ; ██████████ S U B R O U T I N E ██████████
1826
1826 clear_14x5_HL:
1826 11 DB FF          ; CODE XREF: 0000:1322|p
1826                  ; 0000:1373|p ...
1829 0E 0E          ld      de, #0xFFDB
182B 3E 10          ld      c, #0xE
182D                  ld      a, #0x10          ; <space>
182D loc_0_182D:
182D 06 05          ld      b, #5          ; CODE XREF: clear_14x5_HL+F|j
182F
182F loc_0_182F:
182F 77             ld      (hl), a          ; CODE XREF: clear_14x5_HL+B|j
1830 23             inc      hl          ; display space
1831 10 FC          djnz    loc_0_182F    ; next row
1833 19             add      hl, de        ; loop 5 times
1834 0D             dec      c            ; next column
1835 C2 2D 18       jp      NZ, loc_0_182D ; loop through 14 columns
1838 C9           ret
1838 ; End of function clear_14x5_HL
1838
1839
1839 loc_0_1839:
1839 21 90 63       ld      hl, #kong_thrash_tmr          ; DATA XREF: 0000:164C|o
183C 34           inc      (hl)
183D CA 59 18       jp      Z, loc_0_1859
1840 7E           ld      a, (hl)
1841 E6 07          and      #7
1843 C0           ret      NZ
1844 11 CF 39       ld      de, #0x39CF
1847 CB 5E         bit      3, (hl)
1849 20 03         jr      NZ, loc_0_184E
184B 11 F7 39       ld      de, #0x39F7
184E
184E loc_0_184E:
184E EB           ex      de, hl          ; CODE XREF: 0000:1849|j
184F CD 4E 00      call    copy_sprites_2_l1_data
1852 21 08 69      ld      hl, #soft_sprite_ram+8          ; sprite #2, y coord
1855 0E 44         ld      c, #68
1857 FF           rst      0x38              ; add 68 to y coord for 10 sprites
1858 C9           ret
1859
1859
1859 loc_0_1859:
1859 21 5C 38       ld      hl, #dk_normal_spr          ; CODE XREF: 0000:183D|j
185C CD 4E 00      call    copy_sprites_2_l1_data
185F 21 08 69      ld      hl, #soft_sprite_ram+8          ; sprite #2, y coord
1862 0E 44         ld      c, #68
1864 FF           rst      0x38              ; add 68 to y coord for 10 sprites
1865 3E 20         ld      a, #0x20 ; ' '
1867 32 09 60      ld      (eight_bit_countdown), a
186A 21 88 63      ld      hl, #unk_0_6388
186D 34           inc      (hl)
186E C9           ret
186F
186F
186F loc_0_186F:
186F DF           rst      0x18              ; DATA XREF: 0000:164E|o
1870 21 1F 3A       ld      hl, #fk_falling_spr          ; wait for 8-bit countdown
1873 CD 4E 00      call    copy_sprites_2_l1_data
1876 3E 03         ld      a, #3
1878 32 84 60      ld      (digital_snd_tmr_kong_fall), a
187B 21 88 63      ld      hl, #unk_0_6388
187E 34           inc      (hl)
187F C9           ret
1880
1880
1880 loc_0_1880:
1880 21 0B 69       ld      hl, #soft_sprite_ram+0xB          ; DATA XREF: 0000:1650|o
1883 0E 01         ld      c, #1          ; sprite #2, x coord
1885 FF           rst      0x38              ; +1
1886 3A 1B 69       ld      a, (soft_sprite_ram+0x1B)          ; add 1 to x coord for 10 sprites
1889 FE D0         cp      #0xD0 ; '8'
188B C0           ret      NZ
188C 3E 20         ld      a, #0x20 ; ' '
188E 32 19 69      ld      (soft_sprite_ram+0x19), a
1891 21 24 6A      ld      hl, #soft_sprite_ram+0x124
1894 36 7F         ld      (hl), #0x7F ; ' '
1896 2C           inc      l
1897 36 39         ld      (hl), #0x39 ; '9'
1899 2C           inc      l
189A 36 01         ld      (hl), #1
189C 2C           inc      l
189D 36 D8         ld      (hl), #0xD8 ; 'I'
189F 21 C6 76      ld      hl, #VRAM_start+0x2C6
18A2 CD 26 18      call    clear_14x5_HL
18A5 11 5F 3A      ld      de, #draw_data_rivet_end5
18A8 CD A7 0D      call    draw_level_background
18AB 11 04 00      ld      de, #4
18AE 01 28 02      ld      bc, #0x228
18B1 21 03 69      ld      hl, #soft_sprite_ram+3          ; sprite #0, x coord
18B4 CD 3D 00      call    add_c_sprite_register_xB
18B7 3E 00         ld      a, #0
18B9 32 AF 62      ld      (byte_0_62AF), a
18BC 3E 03         ld      a, #3          ; tmr=3
18BE 32 82 60      ld      (digital_snd_tmr_thump), a
18C1 21 88 63      ld      hl, #unk_0_6388
18C4 34           inc      (hl)

```

```

18C5 C9                ; ----- ret -----
18C6
18C6
18C6
18C6 loc_0_18C6:                ; DATA XREF: 0000:1652|o
18C6 21 AF 62          ld      hl, #byte_0_62AF
18C9 35              dec      (hl)
18CA CA 3D 19        jp      Z, loc_0_193D
18CD 7E              ld      a, (hl)
18CE E6 07          and      #7
18D0 C0              ret      NZ
18D1 21 25 6A        ld      hl, #soft_sprite_ram+0x125
18D4 7E              ld      a, (hl)
18D5 EE 80          xor      #0x80 ; 'Ç'
18D7 77              ld      (hl), a
18D8 21 19 69        ld      hl, #soft_sprite_ram+0x19
18DB 46              ld      b, (hl)
18DC CB A8          res      5, b
18DE AF            xor      a
18DF CD 09 30        call   animate_mario_or_barrel_sprite
18E2 F6 20          or      #0x20 ; ' '
18E4 77              ld      (hl), a
18E5 21 AF 62        ld      hl, #byte_0_62AF
18E8 7E              ld      a, (hl)
18E9 FE E0          cp      #0xE0 ; 'Ô'
18EB C2 10 19        jp      NZ, loc_0_1910
18EE 3E 50          ld      a, #0x50 ; 'P'
18F0 32 4F 69        ld      (soft_sprite_ram+0x4F), a
18F3 3E 00          ld      a, #0
18F5 32 4D 69        ld      (soft_sprite_ram+0x4D), a
18F8 3E 9F          ld      a, #0x9F ; 'f'
18FA 32 4C 69        ld      (soft_sprite_ram+0x4C), a
18FD 3A 03 62        ld      a, (mario_y)
1900 FE 80          cp      #0x80 ; 'Ç'
1902 D2 0F 19        jp      NC, loc_0_190F
1905 3E 80          ld      a, #0x80 ; 'Ç'
1907 32 4D 69        ld      (soft_sprite_ram+0x4D), a
190A 3E 5F          ld      a, #0x5F ; '_'
190C 32 4C 69        ld      (soft_sprite_ram+0x4C), a
190F
190F loc_0_190F:                ; CODE XREF: 0000:1902|j
190F 7E              ld      a, (hl)
1910
1910 loc_0_1910:                ; CODE XREF: 0000:18EB|j
1910 FE C0          cp      #0xC0 ; 'L'
1912 C0              ret      NZ
1913 21 8A 60        ld      hl, #unk_0_608A
1916 36 0C          ld      (hl), #0xC
1918 3A 29 62        ld      a, (level)
191B 0F            rrca
191C 38 02          jr      C, loc_0_1920
191E 36 05          ld      (hl), #5
1920
1920 loc_0_1920:                ; CODE XREF: 0000:191C|j
1920 23              inc      hl
1921 36 03          ld      (hl), #3
1923 21 23 6A        ld      hl, #soft_sprite_ram+0x123
1926 36 40          ld      (hl), #0x40 ; '@'
1928 2B              dec      hl
1929 36 09          ld      (hl), #9
192B 2B              dec      hl
192C 36 76          ld      (hl), #0x76 ; 'v'
192E 2B              dec      hl
192F 36 8F          ld      (hl), #0x8F ; 'Ã'
1931 3A 03 62        ld      a, (mario_y)
1934 FE 80          cp      #0x80 ; 'Ç'
1936 D0              ret      NC
1937 3E 6F          ld      a, #0x6F ; 'o'
1939 32 20 6A        ld      (soft_sprite_ram+0x120), a
193C C9              ; ----- ret -----
193D
193D
193D loc_0_193D:                ; CODE XREF: 0000:18CA|j
193D 2A 2A 62        ld      hl, (seq_data)
1940 23              inc      hl
1941 7E              ld      a, (hl)
1942 FE 7F          cp      #0x7F ; ' '
1944 C2 4B 19        jp      NZ, loc_0_194B
1947 21 73 3A        ld      hl, #level_seq_2
194A 7E              ld      a, (hl)
194B
194B loc_0_194B:                ; CODE XREF: 0000:1944|j
194B 22 2A 62        ld      (seq_data), hl
194E 32 27 62        ld      (level_type), a
1951 21 29 62        ld      hl, #level
1954 34              inc      (hl)
1955 11 00 05        ld      de, #0x500
1958 CD 9F 30        call   queue_fg_vector_fn
195B AF            xor      a
195C 32 2E 62        ld      (height), a
195F 32 88 63        ld      (unk_0_6388), a
1962 21 09 60        ld      hl, #eight_bit_countdown
1965 36 E0          ld      (hl), #0xE0 ; 'Ô'
1967 23              inc      hl
1968 36 08          ld      (hl), #8
196A C9              ; ----- ret -----
196B
196B
196B cls_and_set_seq_for_current_play: ; DATA XREF: 0000:0730|o
196B CD 52 08        call   clear_tiles_and_sprites
196E 3A 0E 60        ld      a, (current_player_E)
1971 C6 12          add      a, #18
1973 32 0A 60        ld      (main_sequencer), a
1976 C9              ; ----- ret -----
1977
1977
1977 attract_mode_gameplay:        ; DATA XREF: 0000:074E|o
1977 CD EE 21        call   next_attract_action
197A
197A
197A gameplay:                ; DATA XREF: 0000:071A|o
197A CD BD 1D        call   check_and_handle_bonus
197D CD 8C 1E        call   sub_0_1E8C
1980 CD C3 1A        call   sub_0_1AC3
1983 CD 72 1F        call   sub_0_1F72
1986 CD 8F 2C        call   sub_0_2C8F
1989 CD 03 2C        call   sub_0_2C03
198C CD ED 30        call   sub_0_30ED
198C                                ; process fireballs?

```

```

198F CD 04 2E      call    sub_0_2E04      ; process springs
1992 CD EA 24      call    sub_0_24EA
1995 CD DB 2D      call    sub_0_2DDB
1998 CD D4 2E      call    sub_0_2ED4
199B CD 07 22      call    sub_0_2207
199E CD 33 1A      call    sub_0_1A33
19A1 CD 85 2A      call    sub_0_2A85
19A4 CD 46 1F      call    sub_0_1F46
19A7 CD FA 26      call    sub_0_26FA
19AA CD F2 25      call    sub_0_25F2
19AD CD DA 19      call    sub_0_19DA
19B0 CD FB 03      call    animate_kong_and_pauline
19B3 CD 08 28      call    sub_0_2808
19B6 CD 1D 28      call    sub_0_281D
19B9 CD 57 1E      call    sub_0_1E57
19BC CD 07 1A      call    sub_0_1A07
19BF CD CB 2F      call    sub_0_2FCB
19C2 00           nop
19C3 00           nop
19C4 00           nop
19C5 3A 00 62      ld      a, (mario_alive_flag)
19C8 A7           and      a
19C9 C0           ret      NZ      ; mario alive?
19CA CD 1C 01      call    stop_sound      ; yes, return
19CD 21 82 60      ld      hl, #digital_snd_tmr_thump
19D0 36 03      ld      (hl), #3      ; tmr=3
19D2
19D2      loc_0_19D2:      ; CODE XREF: 0000:1A30|j
19D2 21 0A 60      ld      hl, #main_sequencer
19D5 34           inc      (hl)
19D6 2B           dec      hl      ; next sequence
19D7 36 40      ld      (hl), #64      ; 8-bit countdown
19D9 C9           ret      ; set counter
19DA
19DA      ; ██████████ S U B R O U T I N E ██████████
19DA
19DA      sub_0_19DA:      ; CODE XREF: 0000:19AD|p
19DA 3A 03 62      ld      a, (mario_y)
19DD 06 03      ld      b, #3
19DF 21 0C 6A      ld      hl, #soft_sprite_ram+0x10C
19E2
19E2      loc_0_19E2:      ; CODE XREF: sub_0_19DA+10|j
19E2 BE           cp      (hl)
19E3 CA ED 19      jp      Z, loc_0_19ED
19E6 2C           inc      l
19E7 2C           inc      l
19E8 2C           inc      l
19E9 2C           inc      l
19EA 10 F6      djnz   loc_0_19E2
19EC C9           ret
19ED
19ED      ;
19ED      loc_0_19ED:      ; CODE XREF: sub_0_19DA+9|j
19ED 3A 05 62      ld      a, (mario_x)
19F0 2C           inc      l
19F1 2C           inc      l
19F2 2C           inc      l
19F3 BE           cp      (hl)
19F4 C0           ret      NZ
19F5 2D           dec      l
19F6 2D           dec      l
19F7 CB 5E      bit      3, (hl)
19F9 C0           ret      NZ
19FA 2D           dec      l
19FB 22 43 63      ld      (unk_0_6343), hl
19FE AF           xor      a
19FF 32 42 63      ld      (unk_0_6342), a
1A02 3C           inc      a
1A03 32 40 63      ld      (show_bonus_state), a
1A06 C9           ret
1A06      ; End of function sub_0_19DA
1A06
1A06      ; ██████████ S U B R O U T I N E ██████████
1A06
1A06      sub_0_1A07:      ; CODE XREF: 0000:19BC|p
1A06 3A 86 63      ld      a, (unk_0_6386)
1A0A EF           rst      0x28      ; go!
1A0A
1A0A      ;
1A0A      .dw locret_0_1A1E      ; Jump table
1A0D 15 1A      .dw loc_0_1A15
1A0F 1F 1A      .dw loc_0_1A1F
1A11 2A 1A      .dw loc_0_1A2A
1A13 00 00      .dw 0
1A15
1A15      loc_0_1A15:      ; DATA XREF: sub_0_1A07+6|o
1A15 AF           xor      a
1A16 32 87 63      ld      (unk_0_6387), a
1A19 3E 02      ld      a, #2
1A1B 32 86 63      ld      (unk_0_6386), a
1A1E
1A1E      locret_0_1A1E:      ; DATA XREF: sub_0_1A07+4|o
1A1E C9           ret
1A1E      ; End of function sub_0_1A07
1A1E
1A1E      ;
1A1E      loc_0_1A1F:      ; DATA XREF: sub_0_1A07+8|o
1A1E 21 87 63      ld      hl, #0x6387
1A22 35           dec      (hl)
1A23 C0           ret      NZ
1A24 3E 03      ld      a, #3
1A26 32 86 63      ld      (unk_0_6386), a
1A29 C9           ret
1A2A
1A2A      ;
1A2A      loc_0_1A2A:      ; DATA XREF: sub_0_1A07+A|o
1A2A 3A 16 62      ld      a, (mario_jumping)
1A2D A7           and      a
1A2E C0           ret      NZ
1A2F E1           pop      hl
1A30 C3 D2 19      jp      loc_0_19D2
1A33

```

```

1A33      ; SUBROUTINE
1A33
1A33
1A33      sub_0_1A33:                                ; CODE XREF: 0000:199E|p
1A33 3E 08      ld      a, #8
1A35 F7      rst      0x30                            ; return if level bit not set
1A36 3A 03 62      ld      a, (mario_y)
1A39 FE 4B      cp      #0x4B ; 'K'
1A3B CA 4B 1A      jp      Z, loc_0_1A4B
1A3E FE B3      cp      #0xB3 ; '|'
1A40 CA 4B 1A      jp      Z, loc_0_1A4B
1A43 3A 91 62      ld      a, (unk_0_6291)
1A46 3D      dec      a
1A47 CA 51 1A      jp      Z, loc_0_1A51
1A4A C9      ret
1A4B
1A4B
1A4B      loc_0_1A4B:                                ; CODE XREF: sub_0_1A33+8|j
1A4B      ; sub_0_1A33+D|j
1A4B      ld      a, #1
1A4D 32 91 62      ld      (unk_0_6291), a
1A50 C9      ret
1A51
1A51
1A51      loc_0_1A51:                                ; CODE XREF: sub_0_1A33+14|j
1A51 32 91 62      ld      (unk_0_6291), a
1A54 47      ld      b, a
1A55 3A 05 62      ld      a, (mario_x)
1A58 3D      dec      a
1A59 FE D0      cp      #0xD0 ; 'ð'
1A5B D0      ret      NC
1A5C 07      rlca
1A5D D2 62 1A      jp      NC, loc_0_1A62
1A60 CB D0      set      2, b
1A62
1A62      loc_0_1A62:                                ; CODE XREF: sub_0_1A33+2A|j
1A62 07      rlca
1A63 07      rlca
1A64 D2 69 1A      jp      NC, loc_0_1A69
1A67 CB C8      set      1, b
1A69
1A69      loc_0_1A69:                                ; CODE XREF: sub_0_1A33+31|j
1A69 E6 07      and      #7
1A6B FE 06      cp      #6
1A6D C2 72 1A      jp      NZ, loc_0_1A72
1A70 CB C8      set      1, b
1A72
1A72      loc_0_1A72:                                ; CODE XREF: sub_0_1A33+3A|j
1A72 3A 03 62      ld      a, (mario_y)
1A75 07      rlca
1A76 D2 7B 1A      jp      NC, loc_0_1A7B
1A79 CB C0      set      0, b
1A7B
1A7B      loc_0_1A7B:                                ; CODE XREF: sub_0_1A33+43|j
1A7B 21 92 62      ld      hl, #unk_0_6292
1A7E 78      ld      a, b
1A7F 85      add      a, 1
1A80 6F      ld      l, a
1A81 7E      ld      a, (hl)
1A82 A7      and      a
1A83 C8      ret      Z
1A84 36 00      ld      (hl), #0
1A86 21 90 62      ld      hl, #unk_0_6290
1A89 35      dec      (hl)
1A8A 78      ld      a, b
1A8B 01 05 00      ld      bc, #5
1A8E 1F      rra
1A8F DA BD 1A      jp      C, loc_0_1ABD
1A92 21 CB 02      ld      hl, #0x2CB
1A95
1A95      loc_0_1A95:                                ; CODE XREF: sub_0_1A33+8D|j
1A95 A7      and      a
1A96 CA 9E 1A      jp      Z, loc_0_1A9E
1A99
1A99      loc_0_1A99:                                ; CODE XREF: sub_0_1A33+68|j
1A99 09      add      hl, bc
1A9A 3D      dec      a
1A9B C2 99 1A      jp      NZ, loc_0_1A99
1A9E
1A9E      loc_0_1A9E:                                ; CODE XREF: sub_0_1A33+63|j
1A9E 01 00 74      ld      bc, #VRAM_start
1AA1 09      add      hl, bc
1AA2 3E 10      ld      a, #0x10
1AA4 77      ld      (hl), a
1AA5 2D      dec      l
1AA6 77      ld      (hl), a
1AA7 2C      inc      l
1AA8 2C      inc      l
1AA9 77      ld      (hl), a
1AAA 3E 01      ld      a, #1
1AAC 32 40 63      ld      (show_bonus_state), a
1AAF 32 42 63      ld      (unk_0_6342), a
1AB2 32 25 62      ld      (unk_0_6225), a
1AB5 3A 16 62      ld      a, (mario_jumping)
1AB8 A7      and      a
1AB9 CC 95 1D      call   Z, sub_0_1D95
1ABC C9      ret
1ABD
1ABD
1ABD      loc_0_1ABD:                                ; CODE XREF: sub_0_1A33+5C|j
1ABD 21 2B 01      ld      hl, #0x12B
1AC0 C3 95 1A      jp      loc_0_1A95
1AC0      ; End of function sub_0_1A33
1AC0
1AC3
1AC3
1AC3
1AC3      SUBROUTINE
1AC3
1AC3      sub_0_1AC3:                                ; CODE XREF: 0000:1980|p
1AC3 3A 16 62      ld      a, (mario_jumping)
1AC6 3D      dec      a
1AC7 CA B2 1B      jp      Z, loc_0_1BB2
1ACA 3A 1E 62      ld      a, (unk_0_621E)
1ACD A7      and      a
1ACE C2 55 1B      jp      NZ, loc_0_1B55
1AD1 3A 17 62      ld      a, (unk_0_6217)

```



```

1AD4 3D          dec     a
1AD5 CA E6 1A   jp      Z, loc_0_1AE6
1AD8 3A 15 62   ld      a, (mario_climbing)
1ADB 3D          dec     a
1ADC CA 38 1B   jp      Z, loc_0_1B38
1ADF 3A 10 60   ld      a, (controller_in)
1AE2 17          rla
1AE3 DA 6E 1B   jp      C, loc_0_1B6E
1AE6            ; jump pressed?
1AE6            ; yes, skip
1AE6            ; CODE XREF: sub_0_1AC3+12|j
loc_0_1AE6:
1AE6 CD 1F 24   call   sub_0_241F
1AE9 3A 10 60   ld      a, (controller_in)
1AEC 1D          dec     e
1AED CA F5 1A   jp      Z, loc_0_1AF5
1AF0 CB 47          bit     0, a
1AF2 C2 8F 1C   jp      NZ, mario_right
1AF5            ; right?
1AF5            ; yes, skip
1AF5            ; CODE XREF: sub_0_1AC3+2A|j
loc_0_1AF5:
1AF5 15          dec     d
1AF6 CA FE 1A   jp      Z, loc_0_1AFE
1AF9 CB 4F          bit     1, a
1AFB C2 AB 1C   jp      NZ, mario_left
1AFE            ; left?
1AFE            ; yes, skip
1AFE            ; CODE XREF: sub_0_1AC3+33|j
loc_0_1AFE:
1AFE 3A 17 62   ld      a, (unk_0_6217)
1B01 3D          dec     a
1B02 C8          ret     Z
1B03 3A 05 62   ld      a, (mario_x)
1B06 C6 08          add     a, #8
1B08 57          ld      d, a
1B09 3A 03 62   ld      a, (mario_y)
1B0C F6 03          or     #3
1B0E CB 97          res     2, a
1B10 01 15 00   ld      bc, #0x15
1B13 CD 6E 23   call   sub_0_236E
1B16 F5          push   af
1B17 21 07 62   ld      hl, #mario_flipy_tile
1B1A 7E          ld      a, (hl)
1B1B E6 80          and     #0x80 ; 'Ç'
1B1D F6 06          or     #6
1B1F 77          ld      (hl), a
1B20 21 1A 62   ld      hl, #unk_0_621A
1B23 3E 04          ld      a, #4
1B25 B9          cp     c
1B26 36 01          ld      (hl), #1
1B28 D2 2C 1B   jp      NC, loc_0_1B2C
1B2B 35          dec     (hl)
1B2C            ; set as broken ladder
1B2C            ; set as normal ladder
1B2C            ; CODE XREF: sub_0_1AC3+65|j
loc_0_1B2C:
1B2C F1          pop     af
1B2D A7          and     a
1B2E CA 4E 1B   jp      Z, loc_0_1B4E
1B31 7E          ld      a, (hl)
1B32 A7          and     a
1B33 C0          ret     NZ
1B34 2C          inc     l
1B35 72          ld      (hl), d
1B36 2C          inc     l
1B37 70          ld      (hl), b
1B38            ; set top coordinate of ladder
1B38            ; set bottom Y coordinate of ladder
1B38            ; CODE XREF: sub_0_1AC3+19|j
loc_0_1B38:
1B38 3A 10 60   ld      a, (controller_in)
1B3B CB 5F          bit     3, a
1B3D C2 F2 1C   jp      NZ, loc_0_1CF2
1B40 3A 15 62   ld      a, (mario_climbing)
1B43 A7          and     a
1B44 C8          ret     Z
1B45            ; yes, go
1B45            ; CODE XREF: sub_0_1AC3+8F|j
loc_0_1B45:
1B45 3A 10 60   ld      a, (controller_in)
1B48 CB 57          bit     2, a
1B4A C2 03 1D   jp      NZ, loc_0_1D03
1B4D C9          ret
1B4E            ; up?
1B4E            ; yes, go
1B4E            ; CODE XREF: sub_0_1AC3+6B|j
loc_0_1B4E:
1B4E 2C          inc     l
1B4F 70          ld      (hl), b
1B50 2C          inc     l
1B51 72          ld      (hl), d
1B52 C3 45 1B   jp      loc_0_1B45
1B55            ; set top Y corordinate of ladder
1B55            ; set bottom coordinate of ladder
1B55            ; CODE XREF: sub_0_1AC3+B|j
loc_0_1B55:
1B55 21 1E 62   ld      hl, #unk_0_621E
1B58 35          dec     (hl)
1B59 C0          ret     NZ
1B5A 3A 18 62   ld      a, (unk_0_6218)
1B5D 32 17 62   ld      (unk_0_6217), a
1B60 21 07 62   ld      hl, #mario_flipy_tile
1B63 7E          ld      a, (hl)
1B64 E6 80          and     #0x80 ; 'Ç'
1B66 77          ld      (hl), a
1B67 AF          xor     a
1B68 32 02 62   ld      (mario_animation_cell), a
1B6B C3 A6 1D   jp      update_mario_sprite_registers
1B6E            ; animation cell 0
1B6E            ; CODE XREF: sub_0_1AC3+20|j
loc_0_1B6E:
1B6E 3E 01          ld      a, #1
1B70 32 16 62   ld      (mario_jumping), a
1B73 21 10 62   ld      hl, #unk_0_6210
1B76 3A 10 60   ld      a, (controller_in)
1B79 01 80 00   ld      bc, #0x80 ; 'Ç'
1B7C 1F          rra
1B7D DA 8A 1B   jp      C, loc_0_1B8A
1B80 01 80 FF   ld      bc, #0xFF80
1B83 1F          rra
1B84 DA 8A 1B   jp      C, loc_0_1B8A
1B87 01 00 00   ld      bc, #0
1B8A            ; right?
1B8A            ; yes, skip
1B8A            ; left?
1B8A            ; yes, skip
1B8A            ; CODE XREF: sub_0_1AC3+BA|j
loc_0_1B8A:
1B8A AF          xor     a
1B8B 70          ld      (hl), b
1B8C 2C          inc     l

```

```

1B8D 71      ld      (hl), c
1B8E 2C      inc     l
1B8F 36 01   ld      (hl), #1
1B91 2C      inc     l
1B92 36 48   ld      (hl), #0x48 ; 'H'
1B94 2C      inc     l
1B95 77      ld      (hl), a
1B96 32 04 62 ld      (unk_0_6204), a
1B99 32 06 62 ld      (unk_0_6206), a
1B9C 3A 07 62 ld      a, (mario_flipy_tile)
1B9F E6 80   and     #0x80 ; 'Ç'
1BA1 F6 0E   or      #0xE                      ; mario jumping character
1BA3 32 07 62 ld      (mario_flipy_tile), a
1BA6 3A 05 62 ld      a, (mario_x)
1BA9 32 0E 62 ld      (unk_0_620E), a
1BAC 21 81 60 ld      hl, #digital_snd_tmr_jump
1BAF 36 03   ld      (hl), #3                      ; tmr=3
1BB1 C9      ret
1BB2
1BB2
1BB2      loc_0_1BB2:                      ; CODE XREF: sub_0_1AC3+41j
1BB2 DD 21 00 62 ld      ix, #mario_alive_flag
1BB6 3A 03 62 ld      a, (mario_y)
1BB9 DD 77 0B ld      0xB(ix), a                      ; store X position before a jump
1BBC 3A 05 62 ld      a, (mario_x)
1BBF DD 77 0C ld      0xC(ix), a                      ; store Y position before a jump
1BC2 CD 9C 23 call    sub_0_239C
1BC5 CD 1F 24 call    sub_0_241F
1BC8 15      dec     d
1BC9 C2 F2 1B jp      NZ, loc_0_1BF2
1BCC DD 36 10 00 ld      0x10(ix), #0
1BD0 DD 36 11 80 ld      0x11(ix), #0x80 ; 'Ç'
1BD4 DD CB 07 FE set     7, 7(ix)                      ; h-flip sprite
1BD8
1BD8      loc_0_1BD8:                      ; CODE XREF: sub_0_1AC3+13Fj
1BD8 3A 20 62 ld      a, (unk_0_6220)
1BDB 3D      dec     a
1BDC CA EC 1B jp      Z, loc_0_1BEC
1BDF CD 07 24 call    sub_0_2407
1BE2 DD 74 12 ld      0x12(ix), h
1BE5 DD 75 13 ld      0x13(ix), l
1BE8 DD 36 14 00 ld      0x14(ix), #0
1BEC
1BEC      loc_0_1BEC:                      ; CODE XREF: sub_0_1AC3+119j
1BEC CD 9C 23 call    sub_0_239C
1BEF C3 05 1C jp      loc_0_1C05
1BF2
1BF2      loc_0_1BF2:                      ; CODE XREF: sub_0_1AC3+106j
1BF2 1D      dec     e
1BF3 C2 05 1C jp      NZ, loc_0_1C05
1BF6 DD 36 10 FF ld      0x10(ix), #0xFF
1BFA DD 36 11 80 ld      0x11(ix), #0x80 ; 'Ç'
1BFE DD CB 07 BE res     7, 7(ix)                      ; un-hflip sprite
1C02 C3 D8 1B jp      loc_0_1BD8
1C05
1C05
1C05      loc_0_1C05:                      ; CODE XREF: sub_0_1AC3+12Cj
1C05 CD 1C 2B      ; sub_0_1AC3+130j
1C05      call    sub_0_2B1C
1C08 3D      dec     a                      ; are we jumping?
1C09 CA 3A 1C jp      Z, loc_0_1C3A
1C0C 3A 1F 62 ld      a, (unk_0_621F)
1C0F 3D      dec     a
1C10 CA 76 1C jp      Z, loc_0_1C76
1C13 3A 14 62 ld      a, (unk_0_6214)
1C16 D6 14   sub     #0x14
1C18 C2 33 1C jp      NZ, loc_0_1C33
1C1B 3E 01   ld      a, #1                      ; peak of the jump
1C1D 32 1F 62 ld      (unk_0_621F), a
1C20 CD 53 28 call    sub_0_2853                      ; check for bonus points?
1C23 A7      and     a                      ; any bonus points?
1C24 CA A6 1D jp      Z, update_mario_sprite_registers ; no, exit
1C27 32 42 63 ld      (unk_0_6342), a
1C2A 3E 01   ld      a, #1                      ; register bonus
1C2C 32 40 63 ld      (show_bonus_state), a
1C2F 32 25 62 ld      (unk_0_6225), a
1C32 00      nop
1C33
1C33      loc_0_1C33:                      ; CODE XREF: sub_0_1AC3+155j
1C33 3C      inc     a
1C34 CC 54 29 call    Z, sub_0_2954
1C37 C3 A6 1D jp      update_mario_sprite_registers
1C3A
1C3A      loc_0_1C3A:                      ; CODE XREF: sub_0_1AC3+146j
1C3A 05      dec     b
1C3B CA 4F 1C jp      Z, loc_0_1C4F
1C3E 3C      inc     a
1C3F 32 1F 62 ld      (unk_0_621F), a
1C42 AF      xor     a
1C43 21 10 62 ld      hl, #0x6210
1C46 06 05   ld      b, #5
1C48
1C48      loc_0_1C48:                      ; CODE XREF: sub_0_1AC3+187j
1C48 77      ld      (hl), a
1C49 2C      inc     l
1C4A 10 FC   djnz   loc_0_1C48
1C4C C3 A6 1D jp      update_mario_sprite_registers
1C4F
1C4F      loc_0_1C4F:                      ; CODE XREF: sub_0_1AC3+178j
1C4F 32 16 62 ld      (mario_jumping), a
1C52 3A 20 62 ld      a, (unk_0_6220)
1C55 EE 01   xor     #1
1C57 32 00 62 ld      (mario_alive_flag), a
1C5A 21 07 62 ld      hl, #mario_flipy_tile
1C5D 7E      ld      a, (hl)
1C5E E6 80   and     #0x80 ; 'Ç'
1C60 F6 0F   or      #0xF                      ; mario landing character
1C62 77      ld      (hl), a
1C63 3E 04   ld      a, #4
1C65 32 1E 62 ld      (unk_0_621E), a
1C68 AF      xor     a
1C69 32 1F 62 ld      (unk_0_621F), a
1C6C 3A 25 62 ld      a, (unk_0_6225)

```

```

1C6F 3D          dec      a
1C70 CC 95 1D   call     Z, sub_0_1D95
1C73 C3 A6 1D   jp      update_mario_sprite_registers
1C76
1C76
1C76          loc_0_1C76:
1C76 3A 05 62     ld      a, (mario_x)
1C79 21 0E 62     ld      hl, #unk_0_620E
1C7C D6 0F       sub      #0xF
1C7E BE         cp      (hl)
1C7F DA A6 1D     jp      C, update_mario_sprite_registers
1C82 3E 01       ld      a, #1
1C84 32 20 62     ld      (unk_0_6220), a
1C87 21 84 60     ld      hl, #0x6084
1C8A 36 03       ld      (hl), #3
1C8C C3 A6 1D     jp      update_mario_sprite_registers
1C8F
1C8F
1C8F          mario_right:
1C8F 06 01     ld      b, #1
1C91 3A 0F 62     ld      a, (mario_cell_animate_cntr)
1C94 A7         and     a
1C95 C2 D2 1C     jp      NZ, move_mario_left_right
1C98 3A 02 62     ld      a, (mario_animation_cell)
1C9B 47         ld      b, a
1C9C 3E 05     ld      a, #5
1C9E CD 09 30     call    animate_mario_or_barrel_sprite
1CA1 32 02 62     ld      (mario_animation_cell), a
1CA4 E6 03     and     #3
1CA6 F6 80     or      #0x80 ; 'Ç'
1CA8 C3 C2 1C     jp      update_mario_lr_sprite_data
1CAB
1CAB
1CAB          mario_left:
1CAB 06 FF     ld      b, #0xFF
1CAD 3A 0F 62     ld      a, (mario_cell_animate_cntr)
1CB0 A7         and     a
1CB1 C2 D2 1C     jp      NZ, move_mario_left_right
1CB4 3A 02 62     ld      a, (mario_animation_cell)
1CB7 47         ld      b, a
1CB8 3E 01     ld      a, #1
1CBA CD 09 30     call    animate_mario_or_barrel_sprite
1CBD 32 02 62     ld      (mario_animation_cell), a
1CC0 E6 03     and     #3
1CC2
1CC2          update_mario_lr_sprite_data:
1CC2 21 07 62     ld      hl, #mario_flipy_tile
1CC5 77         ld      (hl), a
1CC6 1F         rra
1CC7 DC 8F 1D     call    C, play_walking_sound
1CCA 3E 02     ld      a, #2
1CCC 32 0F 62     ld      (mario_cell_animate_cntr), a
1CCF C3 A6 1D     jp      update_mario_sprite_registers
1CD2
1CD2          move_mario_left_right:
1CD2 21 03 62     ld      hl, #mario_y
1CD5 7E         ld      a, (hl)
1CD6 80         add     a, b
1CD7 77         ld      (hl), a
1CD8 3A 27 62     ld      a, (level_type)
1CDB 3D         dec     a
1CDC C2 EB 1C     jp      NZ, loc_0_1CEB
1CDF 66         ld      h, (hl)
1CE0 3A 05 62     ld      a, (mario_x)
1CE3 6F         ld      l, a
1CE4 CD 33 23     call    adjust_height_on_girders
1CE7 7D         ld      a, l
1CE8 32 05 62     ld      (mario_x), a
1CEB
1CEB          loc_0_1CEB:
1CEB 21 0F 62     ld      hl, #mario_cell_animate_cntr
1CEE 35         dec     (hl)
1CEF C3 A6 1D     jp      update_mario_sprite_registers
1CF2
1CF2          loc_0_1CF2:
1CF2 3A 0F 62     ld      a, (mario_cell_animate_cntr)
1CF5 A7         and     a
1CF6 C2 8A 1D     jp      NZ, loc_0_1D8A
1CF9 3E 03     ld      a, #3
1CFB 32 0F 62     ld      (mario_cell_animate_cntr), a
1CFE 3E 02     ld      a, #2
1D00 C3 11 1D     jp      loc_0_1D11
1D03
1D03          loc_0_1D03:
1D03 3A 0F 62     ld      a, (mario_cell_animate_cntr)
1D06 A7         and     a
1D07 C2 76 1D     jp      NZ, loc_0_1D76
1D0A 3E 04     ld      a, #4
1D0C 32 0F 62     ld      (mario_cell_animate_cntr), a
1D0F 3E FE     ld      a, #0xFE ; '■'
1D11
1D11          loc_0_1D11:
1D11 21 05 62     ld      hl, #mario_x
1D14 86         add     a, (hl)
1D15 77         ld      (hl), a
1D16 47         ld      b, a
1D17 3A 22 62     ld      a, (unk_0_6222)
1D1A EE 01     xor     #1
1D1C 32 22 62     ld      (unk_0_6222), a
1D1F C2 51 1D     jp      NZ, loc_0_1D51
1D22 78         ld      a, b
1D23 C6 08     add     a, #8
1D25 21 1C 62     ld      hl, #unk_0_621C
1D28 BE         cp      (hl)
1D29 CA 67 1D     jp      Z, loc_0_1D67
1D2C 2D         dec     l
1D2D 96         sub     (hl)
1D2E CA 67 1D     jp      Z, loc_0_1D67
1D31 06 05     ld      b, #5
1D33 D6 08     sub     #8
1D35 CA 3F 1D     jp      Z, loc_0_1D3F
1D38 05         dec     b

```

```

1D39 D6 04      sub      #4
1D3B CA 3F 1D   jp       Z, loc_0_1D3F
1D3E 05        dec      b
1D3F
1D3F          loc_0_1D3F:                                ; CODE XREF: sub_0_1AC3+272|j
1D3F 3E 80      ; sub_0_1AC3+278|j
1D3F          ld       a, #0x80 ; 'Ç'
1D41 21 07 62   ld       hl, #mario_flipy_tile
1D44 A6         and      (hl)
1D45 EE 80      xor      #0x80 ; 'Ç'                ; hflip mario
1D47 B0         or       b
1D48 77         ld       (hl), a
1D49
1D49          loc_0_1D49:                                ; CODE XREF: sub_0_1AC3+2A1|j
1D49 3E 01      ld       a, #1                ; flag mario climbing a ladder
1D4B 32 15 62   ld       (mario_climbing), a
1D4E C3 A6 1D   jp       update_mario_sprite_registers
1D51
1D51          loc_0_1D51:                                ; CODE XREF: sub_0_1AC3+25C|j
1D51 2D         dec      l
1D52 2D         dec      l
1D53 7E         ld       a, (hl)
1D54 F6 03      or       #3
1D56 CB 97      res      2, a
1D58 77         ld       (hl), a
1D59 3A 24 62   ld       a, (unk_0_6224)
1D5C EE 01      xor      #1
1D5E 32 24 62   ld       (unk_0_6224), a
1D61 CC 8F 1D   call    Z, play_walking_sound
1D64 C3 49 1D   jp       loc_0_1D49
1D67
1D67          loc_0_1D67:                                ; CODE XREF: sub_0_1AC3+266|j
1D67 3E 06      ; sub_0_1AC3+26B|j
1D67          ld       a, #6                ; mario climbing character
1D69 32 07 62   ld       (mario_flipy_tile), a
1D6C AF         xor      a
1D6D 32 19 62   ld       (unk_0_6219), a
1D70 32 15 62   ld       (mario_climbing), a        ; flaf not climbing a ladder
1D73 C3 A6 1D   jp       update_mario_sprite_registers
1D76
1D76          loc_0_1D76:                                ; CODE XREF: sub_0_1AC3+244|j
1D76 3A 1A 62   ld       a, (unk_0_621A)
1D79 A7         and      a
1D7A CA 8A 1D   jp       Z, loc_0_1D8A
1D7D 32 19 62   ld       (unk_0_6219), a
1D80 3A 1C 62   ld       a, (unk_0_621C)
1D83 D6 13      sub      #0x13
1D85 21 05 62   ld       hl, #mario_x
1D88 BE         cp       (hl)
1D89 D0         ret      NC
1D8A
1D8A          loc_0_1D8A:                                ; CODE XREF: sub_0_1AC3+233|j
1D8A 21 0F 62   ld       hl, #mario_cell_animate_cntr ; sub_0_1AC3+2B7|j
1D8D 35         dec      (hl)
1D8E C9         ret
1D8E          ; End of function sub_0_1AC3
1D8F
1D8F          ; SUBROUTINE
1D8F
1D8F          play_walking_sound:                        ; CODE XREF: sub_0_1AC3+204|p
1D8F 3E 03      ld       a, #3                ; sub_0_1AC3+29E|p
1D91 32 80 60   ld       (digital_snd_tmr_walk), a    ; tmr=3
1D94 C9         ret
1D94          ; End of function play_walking_sound
1D95
1D95          ; SUBROUTINE
1D95
1D95          sub_0_1D95:                                ; CODE XREF: sub_0_1A33+86|p
1D95          ; sub_0_1AC3+1AD|p
1D95 32 25 62   ld       (unk_0_6225), a
1D98 3A 27 62   ld       a, (level_type)
1D9B 3D         dec      a
1D9C C8         ret      Z
1D9D 21 8A 60   ld       hl, #unk_0_608A
1DA0 36 0D      ld       (hl), #0xD
1DA2 2C         inc      l
1DA3 36 03      ld       (hl), #3
1DA5 C9         ret
1DA5          ; End of function sub_0_1D95
1DA6
1DA6          ;
1DA6          update_mario_sprite_registers:            ; CODE XREF: sub_0_1AC3+A8|j
1DA6 21 4C 69   ld       hl, #soft_sprite_ram+0x4C ; sub_0_1AC3+161|j ...
1DA9 3A 03 62   ld       a, (mario_y)                ; sprite #19
1DAC 77         ld       (hl), a
1DAD 3A 07 62   ld       a, (mario_flipy_tile)
1DB0 2C         inc      l
1DB1 77         ld       (hl), a
1DB2 3A 08 62   ld       a, (mario_flipx_colour)
1DB5 2C         inc      l
1DB6 77         ld       (hl), a
1DB7 3A 05 62   ld       a, (mario_x)
1DBA 2C         inc      l
1DBB 77         ld       (hl), a
1DBC C9         ret
1DBD
1DBD          ; SUBROUTINE
1DBD
1DBD          check_and_handle_bonus:                    ; CODE XREF: 0000:127C|p
1DBD          ; 0000:1641|p ...
1DBD 3A 40 63   ld       a, (show_bonus_state)
1DC0 EF         rst      0x28                ; go!
1DC0
1DC1 49 1E      .dw no_bonus

```

```

1DC3 C9 1D      .dw show_bonus
1DC5 4A 1E      .dw remove_bonus
1DC7 00        .db 0 ;
1DC8 00        .db 0 ;
1DC9           ;
1DC9           ;
1DC9           ;
1DC9           ; DATA XREF: check_and_handle_bonus+6[j]
1DC9           ; timer
1DC9 3E 40      ld      a, #0x40 ; '@'
1DCB 32 41 63   ld      (show_bonus_timer), a
1DCE 3E 02      ld      a, #2
1DD0 32 40 63   ld      (show_bonus_state), a
1DD3 3A 42 63   ld      a, (unk_0_6342)
1DD6 1F        rra
1DD7 DA 70 3E   jp      C, loc_0_3E70
1DDA 1F        rra
1ddb DA 00 1E   jp      C, award_300_pts
1DDE 1F        rra
1DDF DA F5 1D   jp      C, award_random_bonus
1DE2 21 85 60   ld      hl, #digital_snd_tmr_barrel_jump_priz
1DE5 36 03      ld      (hl), #3 ; tmr=3
1DE7 3A 29 62   ld      a, (level)
1DEA 3D        dec     a
1DEB CA 00 1E   jp      Z, award_300_pts
1DEE 3D        dec     a
1DEF CA 08 1E   jp      Z, award_500_pts
1DF2 C3 10 1E   jp      award_800_pts
1DF5           ;
1DF5           ;
1DF5           ; CODE XREF: check_and_handle_bonus+22[j]
1DF5 3A 18 60   ld      a, (random_no)
1DF8 1F        rra ; 50% chance for 500 pts
1DF9 DA 08 1E   jp      C, award_500_pts ; award 500 pts
1DFC 1F        rra ; 25% chance for 800 pts
1DFD DA 10 1E   jp      C, award_800_pts ; award 800 pts
1E00           ;
1E00           ; CODE XREF: check_and_handle_bonus+1E[j]
1E00 06 7D      ld      b, #0x7D ; '}'
1E02 11 03 00   ld      de, #3 ; '300' sprite tile
1E05 C3 15 1E   jp      award_points ; award 3 (300) points
1E08           ;
1E08           ;
1E08           ; CODE XREF: check_and_handle_bonus+32[j]
1E08 06 7E      ld      b, #0x7E ; '~'
1E0A 11 05 00   ld      de, #5 ; '500' sprite tile
1E0D C3 15 1E   jp      award_points ; award 5 (500) points
1E10           ;
1E10           ;
1E10           ; CODE XREF: check_and_handle_bonus+35[j]
1E10 06 7F      ld      b, #0x7F ; ' '
1E12 11 08 00   ld      de, #8 ; '800' sprite tile
1E15           ; add_bonus_and_update_high_score (800)
1E15           ;
1E15           ; CODE XREF: check_and_handle_bonus+48[j]
1E15 CD 9F 30   call     queue_fg_vector_fn ; check_and_handle_bonus+50[j]
1E18 2A 43 63   ld      hl, (unk_0_6343) ; schedule award points
1E1B 7E        ld      a, (hl) ; ptr x position
1E1C 36 00      ld      (hl), #0 ; prize x position
1E1E 2C        inc     l ; erase prize
1E1F 2C        inc     l ; go to y position
1E20 2C        inc     l
1E21 4E        ld      c, (hl) ; get y position
1E22 C3 36 1E   jp      loc_0_1E36 ; program award sprite
1E25           ;
1E25 11 01 00   ld      de, #1 ; add_bonus_and_update_high_score (100)
1E28           ;
1E28           ; CODE XREF: 0000:3E76[j]
1E28 CD 9F 30   call     queue_fg_vector_fn ; 0000:3E7E[j] ...
1E28           ; schedule award points
1E2B 3A 05 62   ld      a, (mario_x)
1E2E C6 14      add     a, #0x14
1E30 4F        ld      c, a
1E31 3A 03 62   ld      a, (mario_y)
1E34 00        nop
1E35 00        nop
1E36           ;
1E36           ; CODE XREF: check_and_handle_bonus+65[j]
1E36 21 30 6A   ld      hl, #soft_sprite_ram+0x130 ; add bonus points sprite to display
1E39 77        ld      (hl), a
1E3A 2C        inc     l
1E3B 70        ld      (hl), b
1E3C 2C        inc     l
1E3D 36 07      ld      (hl), #7
1E3F 2C        inc     l
1E40 71        ld      (hl), c
1E41 3E 05      ld      a, #5
1E43 F7        rst     0x30 ; return if level bit not set
1E44 21 85 60   ld      hl, #digital_snd_tmr_barrel_jump_priz
1E47 36 03      ld      (hl), #3 ; tmr=3
1E49           ;
1E49           ; DATA XREF: check_and_handle_bonus+4[j]
1E49 C9         ret
1E49           ; End of function check_and_handle_bonus
1E49           ;
1E4A           ;
1E4A           ;
1E4A           ;
1E4A           ; CODE XREF: check_and_handle_bonus+8[j]
1E4A 21 41 63   ld      hl, #show_bonus_timer
1E4D 35        dec     (hl)
1E4E C0        ret     NZ
1E4F AF        xor     a
1E50 32 30 6A   ld      (soft_sprite_ram+0x130), a
1E53 32 40 63   ld      (show_bonus_state), a
1E56 C9        ret
1E57           ;
1E57           ; SUBROUTINE
1E57           ;
1E57           ; CODE XREF: 0000:19B9[p]
1E57 sub_0_1E57:
1E57 3A 27 62   ld      a, (level_type)
1E5A CB 57      bit     2, a
1E5C C2 80 1E   jp      NZ, loc_0_1E80
1E5F 1F        rra
1E60 3A 05 62   ld      a, (mario_x)

```

```

1E63 DA 7A 1E      jp      C, loc_0_1E7A
1E66 FE 51      cp      #0x51 ; 'Q'
1E68 D0      ret      NC
1E69 3A 03 62      ld      a, (mario_y)
1E6C 17      rla
1E6D
1E6D      loc_0_1E6D:                                ; CODE XREF: sub_0_1E57+26↑j
1E6D 3E 00      ld      a, #0
1E6F DA 74 1E      jp      C, loc_0_1E74
1E72 3E 80      ld      a, #0x80 ; 'Ç'
1E74
1E74      loc_0_1E74:                                ; CODE XREF: sub_0_1E57+18↑j
1E74 32 4D 69      ld      (soft_sprite_ram+0x4D), a
1E77 C3 85 1E      jp      loc_0_1E85
1E7A
1E7A      loc_0_1E7A:                                ; CODE XREF: sub_0_1E57+C↑j
1E7A FE 31      cp      #0x31 ; '1'
1E7C D0      ret      NC
1E7D C3 6D 1E      jp      loc_0_1E6D
1E80
1E80      loc_0_1E80:                                ; CODE XREF: sub_0_1E57+5↑j
1E80 3A 90 62      ld      a, (unk_0_6290)
1E83 A7      and      a
1E84 C0      ret      NZ
1E85
1E85      loc_0_1E85:                                ; CODE XREF: sub_0_1E57+20↑j
1E85 3E 16      ld      a, #0x16
1E87 32 0A 60      ld      (main_sequencer), a
1E8A E1      pop      hl
1E8B C9      ret
1E8B      ; End of function sub_0_1E57
1E8B
1E8B      ; SUBROUTINE
1E8C
1E8C      sub_0_1E8C:                                ; CODE XREF: 0000:197D↑p
1E8C 3A 50 63      ld      a, (unk_0_6350)
1E8F A7      and      a
1E90 C8      ret      Z
1E91 CD 96 1E      call   sub_0_1E96
1E94 E1      pop      hl
1E95 C9      ret
1E95      ; End of function sub_0_1E8C
1E95
1E95      ; SUBROUTINE
1E96
1E96      sub_0_1E96:                                ; CODE XREF: sub_0_1E8C+5↑p
1E96 3A 45 63      ld      a, (unk_0_6345)
1E99 EF      rst      0x28 ; go!
1E99
1E9A A0 1E      .dw loc_0_1EA0
1E9C 09 1F      .dw loc_0_1F09
1E9E 23 1F      .dw loc_0_1F23
1E9E      ; Jump table
1E9E
1E9E      loc_0_1EA0:                                ; DATA XREF: sub_0_1E96+4↑o
1E9E 3A 52 63      ld      a, (unk_0_6352) ; hammer just hit something
1EA3 FE 65      cp      #0x65 ; 'e'
1EA5 21 B8 69      hl, #soft_sprite_ram+0xB8 ; process hammer hit effect (start)
1EA8 CA B4 1E      jp      Z, loc_0_1EB4
1EAB 21 D0 69      hl, #soft_sprite_ram+0xD0 ; fireball area in sprite ram
1EAE DA B4 1E      jp      C, loc_0_1EB4
1EB1 21 80 69      ld      hl, #soft_sprite_ram+0x80
1EB4
1EB4      loc_0_1EB4:                                ; CODE XREF: sub_0_1E96+12↑j
1EB4 DD 2A 51 63 ; sub_0_1E96+18↑j
1EB4
1EB8 16 00      ld      ix, (unk_0_6351)
1EBA 3A 53 63      ld      d, #0
1EBD 5F      ld      a, (unk_0_6353)
1EBE 01 04 00      ld      e, a
1EC1 3A 54 63      ld      bc, #4
1EC4 A7      ld      a, (unk_0_6354)
1EC5 CA CF 1E      and      a
1EC8      jp      Z, loc_0_1ECF
1EC8
1EC8      loc_0_1EC8:                                ; CODE XREF: sub_0_1E96+36↑j
1EC8 09      add      hl, bc
1EC9 DD 19      add      ix, de
1ECB 3D      dec      a
1ECC C2 C8 1E      jp      NZ, loc_0_1EC8
1ECF
1ECF      loc_0_1ECF:                                ; CODE XREF: sub_0_1E96+2F↑j
1ECF DD 36 00 00      ld      0(ix), #0
1ED3 DD 7E 15      ld      a, 0x15(ix)
1ED6 A7      and      a
1ED7 3E 02      ld      a, #2
1ED9 CA DE 1E      jp      Z, loc_0_1EDE
1EDC 3E 04      ld      a, #4
1EDE
1EDE      loc_0_1EDE:                                ; CODE XREF: sub_0_1E96+43↑j
1EDE 32 42 63      ld      (unk_0_6342), a
1EE1 01 2C 6A      ld      bc, #soft_sprite_ram+0x12C
1EE4 7E      ld      a, (hl)
1EE5 36 00      ld      (hl), #0
1EE7 02      ld      (bc), a ; flash sprite x coord
1EE8 0C      inc      c
1EE9 2C      inc      l
1EEA 3E 60      ld      a, #0x60 ; '' ; initial hit sprite character
1EEC 02      ld      (bc), a ; flash sprite character
1EED 0C      inc      c
1EEE 2C      inc      l
1EEF 3E 0C      ld      a, #0xC
1EF1 02      ld      (bc), a
1EF2 0C      inc      c
1EF3 2C      inc      l
1EF4 7E      ld      a, (hl)
1EF5 02      ld      (bc), a ; flash sprite y coord
1EF6 21 45 63      ld      hl, #unk_0_6345
1EF9 34      inc      (hl)
1EFA 2C      inc      l
1EFB 36 06      ld      (hl), #6

```

```

1EFD 2C          inc     1
1EFE 36 05      ld      (hl), #5
1F00 21 8A 60   ld      hl, #unk_0_608A
1F03 36 06      ld      (hl), #6
1F05 2C          inc     1
1F06 36 03      ld      (hl), #3
1F08 C9          ret
1F08          ; End of function sub_0_1E96
1F08
1F09
1F09
1F09
1F09 21 46 63    loc_0_1F09:          ; DATA XREF: sub_0_1E96+6↑o
                                ; process hammer hit effect (middle)
                                ld      hl, #unk_0_6346
                                dec     (hl)
                                ret     NZ
                                ld      (hl), #6
                                inc     1
                                dec     (hl)
                                jp      Z, loc_0_1F1D
                                ld      hl, #0x6A2D
                                ld      a, (hl)
                                xor     #1          ; animate hit flash
                                ld      (hl), a
                                ret
1F1D
1F1D
1F1D
1F1D 36 04      loc_0_1F1D:          ; CODE XREF: 0000:1F12↑j
                                ld      (hl), #4
                                dec     1
                                dec     1
                                inc     (hl)
                                ret
1F23
1F23
1F23 21 46 63    loc_0_1F23:          ; DATA XREF: sub_0_1E96+8↑o
                                ; process hammer hit effect (end)
                                ld      hl, #unk_0_6346
                                dec     (hl)
                                ret     NZ
                                ld      (hl), #0xC
                                inc     1
                                dec     (hl)
                                jp      Z, loc_0_1F34
                                ld      hl, #soft_sprite_ram+0x12D
                                inc     (hl)          ; animate hit flash
                                ret
1F34
1F34
1F34
1F34 2D          loc_0_1F34:          ; CODE XREF: 0000:1F2C↑j
                                dec     1
                                dec     1
                                xor     a, a
                                ld      (hl), a
                                ld      (unk_0_6350), a          ; stop effect process
                                inc     a
                                ld      (show_bonus_state), a
                                ld      hl, #soft_sprite_ram+0x12C
                                ld      (unk_0_6343), hl
                                ret
1F46
1F46
1F46
1F46
1F46
1F46 3A 21 62    sub_0_1F46:          ; CODE XREF: 0000:19A4↑p
                                ld      a, (unk_0_6221)
                                and     a
                                ret     Z
                                xor     a, a
                                ld      (unk_0_6204), a
                                ld      (unk_0_6206), a
                                ld      (unk_0_6221), a
                                ld      (unk_0_6210), a
                                ld      (unk_0_6211), a
                                ld      (unk_0_6212), a
                                ld      (unk_0_6213), a
                                ld      (unk_0_6214), a
                                inc     a
                                ld      (mario_jumping), a
                                ld      (unk_0_621F), a
                                ld      a, (mario_x)
                                ld      (unk_0_620E), a
                                ret
1F71
1F71          ; End of function sub_0_1F46
1F71
1F72
1F72
1F72
1F72
1F72 3A 27 62    sub_0_1F72:          ; CODE XREF: 0000:1983↑p
                                ld      a, (level_type)
                                dec     a
                                ret     NZ
                                ix, #unk_0_6700
                                ld      hl, #soft_sprite_ram+0x80
                                ld      de, #0x20 ; ' '
                                ld      b, #0xA
1F83
1F83
1F83 DD 7E 00    loc_0_1F83:          ; CODE XREF: sub_0_1F72+1E↑j
                                ld      a, 0(ix)
                                dec     a
                                jp      Z, loc_0_1F93
                                inc     1
                                inc     1
                                inc     1
1F8D
1F8D
1F8D
1F8D 2C          loc_0_1F8D:          ; CODE XREF: 0000:21CE↑j
                                inc     1
                                add     ix, de
                                djnz    loc_0_1F83
                                ret
1F93
1F93
1F93
1F93 DD 7E 01    loc_0_1F93:          ; CODE XREF: sub_0_1F72+15↑j
                                ld      a, 1(ix)
                                dec     a
                                jp      Z, loc_0_20EC
                                ld      a, 2(ix)

```

```

1F9D 1F          rra
1F9E DA AC 1F    jp      C, loc_0_1FAC
1FA1 1F          rra
1FA2 DA E5 1F    jp      C, loc_0_1FE5
1FA5 1F          rra
1FA6 DA EF 1F    jp      C, loc_0_1FEF
1FA9 C3 53 20    jp      loc_0_2053
1FAC
1FAC
1FAC
1FAC D9          exx
1FAD DD 34 05    inc     5(ix)
1FB0 DD 7E 17    ld      a, 0x17(ix)
1FB3 DD BE 05    cp      5(ix)
1FB6 C2 CE 1F    jp      NZ, loc_0_1FCE
1FB9 DD 7E 15    ld      a, 0x15(ix)
1FBC 07          rlca
1FBD 07          rlca
1FBE C6 15       add     a, #0x15
1FC0 DD 77 07    ld      7(ix), a
1FC3 DD 7E 02    ld      a, 2(ix)
1FC6 EE 07       xor     #7
1FC8 DD 77 02    ld      2(ix), a
1FCB C3 BA 21    jp      loc_0_21BA
1FCE
1FCE
1FCE
1FCE DD 7E 0F    loc_0_1FCE: ; CODE XREF: sub_0_1F72+2C|j
1FD1 3D          ld      a, 0xF(ix)
1FD2 C2 DF 1F    dec     a
1FD5 DD 7E 07    jp      NZ, loc_0_1FDF
1FD8 EE 01       ld      a, 7(ix)
1FDA DD 77 07    xor     #1
1FDD 3E 04       ld      7(ix), a
1FDF            ld      a, #4
1FDF
1FDF DD 77 0F    loc_0_1FDF: ; CODE XREF: sub_0_1F72+60|j
1FE2 C3 BA 21    jp      loc_0_21BA
1FE5
1FE5
1FE5 D9          loc_0_1FE5: ; CODE XREF: sub_0_1F72+30|j
1FE6 01 00 01    exx
1FE9 DD 34 03    ld      bc, #0x100
1FEC C3 F6 1F     inc     3(ix)
1FEF            jp      loc_0_1FF6
1FEF
1FEF
1FEF D9          loc_0_1FEF: ; CODE XREF: sub_0_1F72+34|j
1FF0 01 04 FF    exx
1FF3 DD 35 03    ld      bc, #0xFF04
1FF6            dec     3(ix)
1FF6
1FF6 DD 66 03    loc_0_1FF6: ; CODE XREF: sub_0_1F72+7A|j
1FF9 DD 6E 05    ld      h, 3(ix)
1FFC 7C          ld      l, 5(ix)
1FFD E6 07       ld      a, h
1FFF FE 03       and     #7
2001 CA 5F 21    cp      #3
2004 2D          jp      Z, loc_0_215F
2005 2D          dec     l
2006 2D          dec     l
2007 CD 33 23    dec     l
200A 2C          call adjust_height_on_girders
200B 2C          inc     l
200C 2C          inc     l
200D 7D          inc     l
200E DD 77 05    ld      a, l
2011 CD DE 23    ld      5(ix), a
2014 CD B4 24    call sub_0_23DE
2017 DD 7E 03    call sub_0_24B4
201A FE 1C       ld      a, 3(ix)
201C DA 2F 20    cp      #0x1C
201F FE E4       jp      C, loc_0_202F
2021 DA BA 21    cp      #0xE4 ; 'd'
2024 AF         jp      C, loc_0_21BA
2025 DD 77 10     xor     a
2028 DD 36 11 60 ld      0x10(ix), a
202C C3 38 20     ld      0x11(ix), #0x60 ; ''
202F            jp      loc_0_2038
202F
202F
202F AF          loc_0_202F: ; CODE XREF: sub_0_1F72+AA|j
2030 DD 36 10 FF xor     a
2034 DD 36 11 A0 ld      0x10(ix), #0xFF
2038            ld      0x11(ix), #0xA0 ; 'á'
2038
2038
2038 DD 36 12 FF    loc_0_2038: ; CODE XREF: sub_0_1F72+BA|j
203C DD 36 13 F0    ld      0x12(ix), #0xFF
2040 DD 77 14       ld      0x13(ix), #0xF0 ; '-'
2043 DD 77 0E       ld      0x14(ix), a
2046 DD 77 04       ld      0xE(ix), a
2049 DD 77 06       ld      4(ix), a
204C DD 36 02 08    ld      6(ix), a
2050 C3 BA 21       ld      2(ix), #8
2053            jp      loc_0_21BA
2053
2053
2053 D9          loc_0_2053: ; CODE XREF: sub_0_1F72+37|j
2054 CD 9C 23       exx
2057 CD 2F 2A       call sub_0_239C
205A A7            call sub_0_2A2F
205B C2 83 20       and     a
205E DD 7E 03       jp      NZ, loc_0_2083
2061 C6 08         ld      a, 3(ix)
2063 FE 10         add     a, #8
2065 DA 79 20       cp      #0x10
2068 CD B4 24       jp      C, loc_0_2079
206B DD 7E 10       call sub_0_24B4
206E E6 01         ld      a, 0x10(ix)
2070 07            and     #1
2071 07            rlca
2072 4F            rlca
2073 CD DE 23       ld      C, a
2076 C3 BA 21       call sub_0_23DE
                jp      loc_0_21BA

```



```

2079      ; -----
2079
2079      loc_0_2079:                                     ; CODE XREF: sub_0_1F72+F3↑j
2079 AF          xor     a                                     ;
207A DD 77 00      ld     0(ix), a
207D DD 77 03      ld     3(ix), a
2080 C3 BA 21      jp     loc_0_21BA
2083      ; -----
2083
2083      loc_0_2083:                                     ; CODE XREF: sub_0_1F72+E9↑j
2083 DD 34 0E      inc     0xE(ix)
2086 DD 7E 0E      ld     a, 0xE(ix)
2089 3D          dec     a
208A CA A2 20      jp     Z, loc_0_20A2
208D 3D          dec     a
208E CA C3 20      jp     Z, loc_0_20C3
2091 DD 7E 10      ld     a, 0x10(ix)
2094 3D          dec     a
2095 3E 04      ld     a, #4
2097 C2 9C 20      jp     NZ, loc_0_209C
209A 3E 02      ld     a, #2
209C
209C      loc_0_209C:                                     ; CODE XREF: sub_0_1F72+125↑j
209C DD 77 02      ld     2(ix), a
209F C3 BA 21      jp     loc_0_21BA
20A2      ; -----
20A2
20A2      loc_0_20A2:                                     ; CODE XREF: sub_0_1F72+118↑j
20A2 DD 7E 15      ld     a, 0x15(ix)
20A5 A7          and     a
20A6 C2 B5 20      jp     NZ, loc_0_20B5
20A9 21 05 62      ld     hl, #mario_x
20AC DD 7E 05      ld     a, 5(ix)
20AF D6 16      sub     #0x16                                     ; check har far mario has fallen when jumping
20B1 BE          cp     (hl)
20B2 D2 C3 20      jp     NC, loc_0_20C3
20B5
20B5      loc_0_20B5:                                     ; CODE XREF: sub_0_1F72+134↑j
20B5 DD 7E 10      ld     a, 0x10(ix)
20B8 A7          and     a
20B9 C2 E1 20      jp     NZ, loc_0_20E1
20BC DD 77 11      ld     0x11(ix), a
20BF DD 36 10 FF      ld     0x10(ix), #0xFF
20C3
20C3      loc_0_20C3:                                     ; CODE XREF: sub_0_1F72+11C↑j
20C3 CD 07 24      ; sub_0_1F72+140↑j ...
20C6 CB 3C      call    sub_0_2407
20C8 CB 1D      srl     h
20CA CB 3C      rr     l
20CC CB 1D      srl     h
20CE DD 74 12      rr     l
20D1 DD 75 13      ld     0x12(ix), h
20D4 AF          ld     0x13(ix), l
20D5 DD 77 14      xor     a
20D8 DD 77 04      ld     0x14(ix), a
20DB DD 77 06      ld     4(ix), a
20DE DD 77 06      ld     6(ix), a
20E1 C3 BA 21      jp     loc_0_21BA
20E1      ; -----
20E1
20E1      loc_0_20E1:                                     ; CODE XREF: sub_0_1F72+147↑j
20E1 DD 36 10 01      ld     0x10(ix), #1
20E5 DD 36 11 00      ld     0x11(ix), #0
20E9 C3 C3 20      jp     loc_0_20C3
20EC      ; -----
20EC
20EC      loc_0_20EC:                                     ; CODE XREF: sub_0_1F72+25↑j
20EC D9          exx
20ED CD 9C 23      call    sub_0_239C
20F0 7C          ld     a, h
20F1 D6 1A      sub     #0x1A
20F3 DD 46 19      ld     b, 0x19(ix)
20F6 B8          cp     b
20F7 DA 04 21      jp     C, loc_0_2104
20FA CD 2F 2A      call    sub_0_2A2F
20FD A7          and     a
20FE C2 18 21      jp     NZ, loc_0_2118
2101 CD B4 24      call    sub_0_24B4
2104
2104      loc_0_2104:                                     ; CODE XREF: sub_0_1F72+185↑j
2104 DD 7E 03      ld     a, 3(ix)
2107 C6 08      add     a, #8
2109 FE 10      cp     #0x10
210B D2 CE 1F      jp     NC, loc_0_1FCE
210E AF          xor     a
210F DD 77 00      ld     0(ix), a
2112 DD 77 03      ld     3(ix), a
2115 C3 BA 21      jp     loc_0_21BA
2118      ; -----
2118
2118      loc_0_2118:                                     ; CODE XREF: sub_0_1F72+18C↑j
2118 DD 7E 05      ld     a, 5(ix)
2118 FE E0      cp     #0xE0 ; '0'
211D DA 46 21      jp     C, loc_0_2146
2120 DD 7E 07      ld     a, 7(ix)
2123 E6 FC      and     #0xFC ; '3'
2125 F6 01      or     #1
2127 DD 77 07      ld     7(ix), a
212A AF          xor     a
212B DD 77 01      ld     1(ix), a
212E DD 77 02      ld     2(ix), a
2131 DD 36 10 FF      ld     0x10(ix), #0xFF
2135 DD 77 11      ld     0x11(ix), a
2138 DD 77 12      ld     0x12(ix), a
213B DD 36 13 B0      ld     0x13(ix), #0xB0 ; '0'
213F DD 36 0E 01      ld     0xE(ix), #1
2143 C3 53 21      jp     loc_0_2153
2146      ; -----
2146
2146      loc_0_2146:                                     ; CODE XREF: sub_0_1F72+1AB↑j
2146 CD 07 24      call    sub_0_2407
2149 CD CB 22      call    sub_0_22CB
214C DD 7E 05      ld     a, 5(ix)
214F DD 77 19      ld     0x19(ix), a
2152 AF          xor     a
2153

```

```

2153      loc_0_2153:      ld      0x14(ix), a      ; CODE XREF: sub_0_1F72+1D1↑j
2153 DD 77 14      ld      4(ix), a
2156 DD 77 04      ld      6(ix), a
2159 DD 77 06      jp      loc_0_21BA
215C C3 BA 21
215F
215F
215F      loc_0_215F:      ld      a, 1      ; CODE XREF: sub_0_1F72+8F↑j
215F 7D      add      a, #5
2160 C6 05      ld      d, a
2162 57      ld      a, h
2163 7C      ld      bc, #0x15
2164 01 15 00      call     sub_0_216D
2167 CD 6D 21      jp      loc_0_21BA
216A C3 BA 21
216A      ; End of function sub_0_1F72
216A
216D
216D      ; ██████████ S U B R O U T I N E ██████████
216D
216D      sub_0_216D:      call     sub_0_236E      ; CODE XREF: sub_0_1F72+1F5↑p
216D CD 6E 23      dec      a
2170 3D      ret      NZ
2171 C0      ld      a, b
2172 78      sub      #5
2173 D6 05      ld      0x17(ix), a
2175 DD 77 17      ld      a, (unk_0_6348)
2178 3A 48 63      and      a
217B A7      jp      Z, loc_0_21B2
217C CA B2 21      ld      a, (mario_x)
217F 3A 05 62      sub      #4
2182 D6 04      cp      d
2184 BA      ret      C
2185 D8      ld      a, (unk_0_6380)
2186 3A 80 63      rra
2189 1F      inc      a
218A 3C      ld      b, a
218B 47      ld      a, (random_no)
218C 3A 18 60      ld      c, a
218F 4F      and      #3
2190 E6 03      cp      b
2192 B8      ret      NC
2193 D0      ld      hl, #controller_in
2194 21 10 60      ld      a, (mario_y)
2197 3A 03 62      cp      e
219A BB      jp      Z, loc_0_21B2
219B CA B2 21      jp      NC, loc_0_21A9
219E D2 A9 21      bit      0, (hl)      ; right?
21A1 CB 46      jp      Z, loc_0_21AE      ; no, skip
21A3 CA AE 21      jp      loc_0_21B2
21A6 C3 B2 21
21A9
21A9      loc_0_21A9:      bit      1, (hl)      ; CODE XREF: sub_0_216D+31↑j
21A9 CB 4E      jp      NZ, loc_0_21B2      ; left?
21AB C2 B2 21      ; yes, skip
21AE
21AE      loc_0_21AE:      ld      a, c      ; CODE XREF: sub_0_216D+36↑j
21AE 79      and      #0x18
21AF E6 18      ret      NZ
21B1 C0
21B2
21B2      loc_0_21B2:      inc      7(ix)      ; CODE XREF: sub_0_216D+F↑j
21B2 DD 34 07      set      0, 2(ix)      ; sub_0_216D+2E↑j ...
21B5 DD CB 02 C6      ret      ; sprite tile #
21B9 C9      ; switch rolling barrel to going-down-ladder barrel
21B9
21B9      ; End of function sub_0_216D
21B9
21BA
21BA      ; ██████████ S U B R O U T I N E ██████████
21BA
21BA      loc_0_21BA:      exx      ; CODE XREF: sub_0_1F72+59↑j
21BA D9      ld      a, 3(ix)      ; sub_0_1F72+70↑j ...
21BA      ld      (hl), a      ; set sprite X
21BB DD 7E 03      inc      1
21BE 77      ld      a, 7(ix)      ; set sprite tile #
21BF 2C      ld      (hl), a
21C0 DD 7E 07      inc      1
21C3 77      ld      a, 8(ix)      ; set sprite vflip & palette
21C4 2C      ld      (hl), a
21C5 DD 7E 08      inc      1
21C8 77      ld      a, 5(ix)      ; set sprite Y
21C9 2C      ld      (hl), a
21CA DD 7E 05      ld      loc_0_1F8D
21CD 77      jp
21CE C3 8D 1F
21CE
21D1 80 FE      attract_mario_inputs: .dw 0xFE80      ; DATA XREF: next_attract_action|o
21D1      ; 1st byte is input, 2nd is timer
21D3 01 C0      .dw 0xC001
21D5 04 50      .dw 0x5004
21D7 02 10      .dw 0x1002
21D9 82 60      .dw 0x6082
21DB 02 10      .dw 0x1002
21DD 82 CA      .dw 0xCA82
21DF 01 10      .dw 0x1001
21E1 81 FF      .dw 0xFF81
21E3 02 38      .dw 0x3802
21E5 01 80      .dw 0x8001
21E7 02 FF      .dw 0xFF02
21E9 04 80      .dw 0x8004
21EB 04 60      .dw 0x6004
21ED 80      .db 0x80
21EE
21EE      ; ██████████ S U B R O U T I N E ██████████
21EE
21EE      next_attract_action:      ; CODE XREF: 0000:1977↑p
21EE 11 D1 21      ld      de, #attract_mario_inputs
21F1 21 CC 63      ld      hl, #attract_movement_entry
21F4 7E      ld      a, (hl)      ; get entry
21F5 07      rlca      ; convert to word
21F6 83      add      a, e      ; add to base
21F7 5F      ld      e, a      ; ptr to entry
21F8 1A      ld      a, (de)      ; 1st byte of entry

```

```

21F9 32 10 60      ld      (controller_in), a      ; store simulated inputs
21FC 2C           inc      l
21FD 7E           ld      a, (hl)           ; get movement timer
21FE 35           dec      (hl)           ; done?
21FF A7           and      a
2200 C0           ret      NZ           ; no, return
2201 1C           inc      e           ; ptr 2nd byte of entry
2202 1A           ld      a, (de)         ; get 2nd byte
2203 77           ld      (hl), a        ; store as timer
2204 2D           dec      l
2205 34           inc      (hl)         ; back to entry
2206 C9           ret                  ; next entry
2206             ; End of function next_attract_action
2207
2207             ; ██████████ SUBROUTINE ██████████
2207
2207 sub_0_2207:           ; CODE XREF: 0000:199B|p
2207 3E 02         ld      a, #2
2209 F7           rst      0x30           ; return if level bit not set
220A 3A 1A 60     ld      a, (gen_purpose_timer)
220D 1F           rra
220E 21 80 62     ld      hl, #unk_0_6280
2211 7E           ld      a, (hl)
2212 DA 19 22     jp      C, loc_0_2219
2215 21 88 62     ld      hl, #unk_0_6288
2218 7E           ld      a, (hl)
2219
2219 loc_0_2219:           ; CODE XREF: sub_0_2207+B|j
2219 E5           push     hl
221A EF           rst      0x28           ; go!
221B 27           daa
221C 22 59 22     ld      (loc_0_2259), hl
221F 99           sbc      a, c
2220 22 A2 22     ld      (loc_0_22A2), hl
2223 00           nop
2224 00           nop
2225 00           nop
2226 00           nop
2227 E1           pop      hl
2228 2C           inc      l
2229 35           dec      (hl)
222A C2 3A 22     jp      NZ, loc_0_223A
222D 2D           dec      l
222E 34           inc      (hl)
222F 2C           inc      l
2230 2C           inc      l
2231 CD 43 22     call    sub_0_2243
2234 3E 01         ld      a, #1
2236 32 1A 62     ld      (unk_0_621A), a
2239 C9           ret
223A
223A
223A loc_0_223A:           ; CODE XREF: sub_0_2207+23|j
223A 2C           inc      l
223B CD 43 22     call    sub_0_2243
223E AF           xor      a
223F 32 1A 62     ld      (unk_0_621A), a
2242 C9           ret
2242             ; End of function sub_0_2207
2242
2243             ; ██████████ SUBROUTINE ██████████
2243
2243 sub_0_2243:           ; CODE XREF: sub_0_2207+2A|p
2243 3A 05 62         ld      a, (mario_x)      ; sub_0_2207+34|p ...
2243 FE 7A         cp      #0x7A ; 'z'
2248 D2 57 22     jp      NC, loc_0_2257
224B 3A 16 62     ld      a, (mario_jumping)
224E A7           and      a
224F C2 57 22     jp      NZ, loc_0_2257
2252 3A 03 62     ld      a, (mario_y)
2255 BE           cp      (hl)
2256 C8           ret      Z
2257
2257 loc_0_2257:           ; CODE XREF: sub_0_2243+5|j
2257 E1           pop      hl      ; sub_0_2243+C|j
2258 C9           ret
2258             ; End of function sub_0_2243
2258
2259*
2259*
2259* loc_0_2259:           ; DATA XREF: sub_0_2207+15|w
2259* E1           pop      hl
225A 2C           inc      l
225B 2C           inc      l
225C 2C           inc      l
225D 2C           inc      l
225E 35           dec      (hl)
225F C0           ret      NZ
2260 3E 04         ld      a, #4
2262 77           ld      (hl), a
2263 2D           dec      l
2264 34           inc      (hl)
2265 CD BD 22     call    sub_0_22BD
2268 3E 78         ld      a, #0x78 ; 'x'
226A BE           cp      (hl)
226B C2 75 22     jp      NZ, loc_0_2275
226E 2D           dec      l
226F 2D           dec      l
2270 2D           dec      l
2271 34           inc      (hl)
2272 2C           inc      l
2273 2C           inc      l
2274 2C           inc      l
2275
2275 loc_0_2275:           ; CODE XREF: 0000:226B|j
2275 2D           dec      l
2276 CD 43 22     call    sub_0_2243
2279 3A 05 62     ld      a, (mario_x)
227C FE 68         cp      #0x68 ; 'h'
227E D2 8A 22     jp      NC, loc_0_228A

```

```

2281
2281 loc_0_2281: ; CODE XREF: 0000:228B↑j
2281 21 05 62 ld hl, #mario_x
2284 34 inc (hl)
2285 CD C0 3F call sub_0_3FC0
2288 34 inc (hl)
2289 C9 ret
;
228A
228A loc_0_228A: ; CODE XREF: 0000:227E↑j
228A 1F rra
228B DA 81 22 jp C, loc_0_2281
228E 1F rra
228F 3E 01 ld a, #1
2291 DA 95 22 jp C, loc_0_2295
2294 AF xor a
2295
2295 loc_0_2295: ; CODE XREF: 0000:2291↑j
2295 32 22 62 ld (unk_0_6222), a
2298 C9 ret
;
2299
2299 E1 pop hl
229A 3A 18 60 ld a, (random_no)
229D E6 3C and #0x3C ; '<'
229F C0 ret NZ
22A0 34 inc (hl)
22A1 C9 ret
;
22A2*
22A2*
22A2* loc_0_22A2: ; DATA XREF: sub_0_2207+19↑w
22A2*E1 pop hl
22A3 2C inc l
22A4 2C inc l
22A5 2C inc l
22A6 2C inc l
22A7 35 dec (hl)
22A8 C0 ret NZ
22A9 36 02 ld (hl), #2
22AB 2D dec l
22AC 35 dec (hl)
22AD CD BD 22 call sub_0_22BD
22B0 3E 68 ld a, #0x68 ; 'h'
22B2 BE cp (hl)
22B3 C0 ret NZ
22B4 AF xor a
22B5 06 80 ld b, #0x80 ; 'Ç'
22B7 2D dec l
22B8 2D dec l
22B9 70 ld (hl), b
22BA 2D dec l
22BB 77 ld (hl), a
22BC C9 ret
22BD
22BD ; SUBROUTINE
22BD
22BD sub_0_22BD: ; CODE XREF: 0000:2265↑p
22BD 7E ; 0000:22AD↑p
22BD ld a, (hl)
22BE CB 5D bit 3, l
22C0 11 4B 69 ld de, #soft_sprite_ram+0x4B
22C3 C2 C9 22 jp NZ, loc_0_22C9
22C6 11 47 69 ld de, #soft_sprite_ram+0x47
22C9
22C9 loc_0_22C9: ; CODE XREF: sub_0_22BD+6↑j
22C9 12 ld (de), a
22CA C9 ret
; End of function sub_0_22BD
22CA
22CA ; SUBROUTINE
22CB
22CB
22CB sub_0_22CB: ; CODE XREF: sub_0_1F72+1D7↑p
22CB 3A 48 63 ld a, (unk_0_6348)
22CE A7 and a
22CF CA E1 22 jp Z, loc_0_22E1
22D2 3A 80 63 ld a, (unk_0_6380)
22D5 3D dec a
22D6 EF rst 0x28 ; go!
22D7 F6 22 ;
22D9 F6 22 .dw loc_0_22F6 ; Jump table
22DB 03 23 .dw loc_0_22F6
22DD 03 23 .dw loc_0_2303
22DF 1A 23 .dw loc_0_2303
22E1
22E1
22E1 loc_0_22E1: ; CODE XREF: sub_0_22CB+4↑j
22E1 3A 29 62 ld a, (level)
22E4 47 ld b, a
22E5 05 dec b
22E6 3E 01 ld a, #1
22E8 CA F9 22 jp Z, loc_0_22F9
22EB 05 dec b
22EC 3E B1 ld a, #0xB1 ; 'Ë'
22EE CA F9 22 jp Z, loc_0_22F9
22F1 3E E9 ld a, #0xE9 ; 'Ù'
22F3 C3 F9 22 jp loc_0_22F9
22F6
22F6
22F6 loc_0_22F6: ; DATA XREF: sub_0_22CB+C↑o
22F6 3A 18 60 ld a, (random_no) ; sub_0_22CB+E↑o
22F9
22F9 loc_0_22F9: ; CODE XREF: sub_0_22CB+1D↑j
22F9 DD 77 11 ; sub_0_22CB+23↑j ...
22F9 ld 0x11(ix), a
22FC E6 01 and #1
22FE 3D dec a
22FF DD 77 10 ld 0x10(ix), a
2302 C9 ret
; End of function sub_0_22CB
2302
2302
2303
2303

```

```

2303      loc_0_2303:                                ; DATA XREF: sub_0_22CB+10↑o
2303 3A 18 60                                         ; sub_0_22CB+12↑o
2303      ld      a, (random_no)
2306 DD 77 11      ld      0x11(ix), a
2309 3A 03 62      ld      a, (mario_y)
230C DD BE 03      cp      3(ix)
230F 3E 01      ld      a, #1
2311 D2 16 23      jp      NC, loc_0_2316
2314 3D      dec      a
2315 3D      dec      a
2316
2316      loc_0_2316:                                ; CODE XREF: 0000:2311↑j
2316 DD 77 10      ld      0x10(ix), a
2319 C9      ret
231A
231A      loc_0_231A:                                ; DATA XREF: sub_0_22CB+14↑o
231A 3A 03 62      ld      a, (mario_y)
231D DD 96 03      sub     3(ix)
2320 0E FF      ld      c, #0xFF
2322 DA 26 23      jp      C, loc_0_2326
2325 0C      inc      c
2326
2326      loc_0_2326:                                ; CODE XREF: 0000:2322↑j
2326 07      rlca
2327 CB 11      rl      c
2329 07      rlca
232A CB 11      rl      c
232C DD 71 10      ld      0x10(ix), c
232F DD 77 11      ld      0x11(ix), a
2332 C9      ret
2333
2333      ; ██████████ SUBROUTINE ██████████
2333
2333      adjust_height_on_girders:                    ; CODE XREF: sub_0_1AC3+221↑p
2333 3E 0F      ; sub_0_1F72+95↑p ...
2333      ld      a, #0xF
2335 A4      and     h
2336 05      dec     b
2337 CA 42 23      jp      Z, loc_0_2342
233A FE 0F      cp      #0xF
233C D8      ret     C
233D 06 FF      ld      b, #0xFF
233F C3 47 23      jp      loc_0_2347
2342
2342      loc_0_2342:                                ; CODE XREF: adjust_height_on_girders+4↑j
2342 FE 01      cp      #1
2344 D0      ret     NC
2345 06 01      ld      b, #1
2347
2347      loc_0_2347:                                ; CODE XREF: adjust_height_on_girders+C↑j
2347 3E F0      ld      a, #0xF0 ; '-'
2349 BD      l      1
234A CA 60 23      jp      Z, loc_0_2360
234D 3E 4C      ld      a, #0x4C ; 'L'
234F BD      cp      1
2350 CA 66 23      jp      Z, loc_0_2366
2353 7D      ld      a, 1
2354 CB 6F      bit     5, a
2356 CA 5C 23      jp      Z, loc_0_235C
2359
2359      loc_0_2359:                                ; CODE XREF: adjust_height_on_girders+2F↑j
2359 90      sub     b
235A
235A      loc_0_235A:                                ; CODE XREF: adjust_height_on_girders+2A↑j
235A 6F      ld      1, a
235B C9      ; adjusted X
235C
235C      loc_0_235C:                                ; CODE XREF: adjust_height_on_girders+23↑j
235C 80      ; adjust_height_on_girders+38↑j
235C      add     a, b
235D C3 5A 23      jp      loc_0_235A
2360
2360      loc_0_2360:                                ; CODE XREF: adjust_height_on_girders+17↑j
2360 CB 7C      bit     7, h
2362 C2 59 23      jp      NZ, loc_0_2359
2365 C9      ;
2366
2366      loc_0_2366:                                ; CODE XREF: adjust_height_on_girders+1D↑j
2366 7C      ld      a, h
2367 FE 98      cp      #0x98 ; 'ÿ'
2369 D8      ret     C
236A 7D      ld      a, 1
236B C3 5C 23      jp      loc_0_235C
236B      ; End of function adjust_height_on_girders
236E
236E      ; ██████████ SUBROUTINE ██████████
236E
236E      sub_0_236E:                                ; CODE XREF: sub_0_1AC3+50↑p
236E 21 00 63      ; sub_0_216D↑p ...
236E      ld      hl, #unk_0_6300
2371
2371      loc_0_2371:                                ; CODE XREF: sub_0_236E+1E↑j
2371 ED B1      cpir
2373 C2 9A 23      jp      NZ, loc_0_239A
2376 E5      push    hl
2377 C5      push    bc
2378 01 14 00      ld      bc, #0x14
237B 09      add     hl, bc
237C 0C      inc     c
237D 5F      ld      e, a
237E 7A      ld      a, d
237F BE      cp      (hl)
2380 CA 8F 23      jp      Z, loc_0_238F
2383 09      add     hl, bc
2384 BE      cp      (hl)
2385 CA 95 23      jp      Z, loc_0_2395
2388 57      ld      d, a
2389 7B      ld      a, e

```

```

238A C1                pop    bc
238B E1                pop    hl
238C C3 71 23          jp      loc_0_2371
238F                    ;
238F                    loc_0_238F:
238F 09                add     hl, bc           ; CODE XREF: sub_0_236E+12↑j
2390 3E 01            ld      a, #1
2392 C3 98 23          jp      loc_0_2398
2395                    ;
2395                    loc_0_2395:
2395 AF                xor     a
2396 ED 42            sbc     hl, bc           ; CODE XREF: sub_0_236E+17↑j
2398                    loc_0_2398:
2398 C1                pop     bc           ; CODE XREF: sub_0_236E+24↑j
2399 46                ld      b, (hl)
239A                    loc_0_239A:
239A E1                pop     hl           ; CODE XREF: sub_0_236E+5↑j
239B C9                ret
239B                    ; End of function sub_0_236E
239C                    ;
239C                    ; SUBROUTINE
239C                    ;
239C                    sub_0_239C:
239C DD 7E 04          ld      a, 4(ix)           ; CODE XREF: sub_0_1AC3+FF↑p
239C                    ; sub_0_1AC3+129↑p ...
239F DD 86 11          add     a, 0x11(ix)
23A2 DD 77 04          ld      4(ix), a
23A5 DD 7E 03          ld      a, 3(ix)
23A8 DD 8E 10          adc     a, 0x10(ix)
23AB DD 77 03          ld      3(ix), a
23AE DD 7E 06          ld      a, 6(ix)
23B1 DD 96 13          sub     0x13(ix)
23B4 6F                ld      l, a
23B5 DD 7E 05          ld      a, 5(ix)
23B8 DD 9E 12          sbc     a, 0x12(ix)
23BB 67                ld      h, a
23BC DD 7E 14          ld      a, 0x14(ix)
23BF A7                and     a
23C0 17                rla
23C1 3C                inc     a
23C2 06 00            ld      b, #0
23C4 CB 10            rl      b
23C6 CB 27            sla     a
23C8 CB 10            rl      b
23CA CB 27            sla     a
23CC CB 10            rl      b
23CE CB 27            sla     a
23D0 CB 10            rl      b
23D2 4F                ld      c, a
23D3 09                add     hl, bc
23D4 DD 74 05          ld      5(ix), h
23D7 DD 75 06          ld      6(ix), l
23DA DD 34 14          inc     0x14(ix)
23DD C9                ret
23DD                    ; End of function sub_0_239C
23DE                    ;
23DE                    ; SUBROUTINE
23DE                    ;
23DE                    sub_0_23DE:
23DE DD 7E 0F          ld      a, 0xF(ix)           ; CODE XREF: sub_0_1F72+9F↑p
23DE                    ; sub_0_1F72+101↑p
23E1 3D                dec     a
23E2 C2 03 24          jp      NZ, loc_0_2403
23E5 AF                xor     a           ; animate rolling barrels
23E6 DD CB 07 26       sla     7(ix)           ; toggle H & V flips
23EA 17                rla
23EB DD CB 08 26       sla     8(ix)           ; toggle H & V flips
23EF 17                rla
23F0 47                ld      b, a
23F1 3E 03            ld      a, #3
23F3 B1                or      c
23F4 CD 09 30          call   animate_mario_or_barrel_sprite
23F7 1F                rra
23F8 DD CB 08 1E       rr      8(ix)           ; toggle H & V flips
23FC 1F                rra
23FD DD CB 07 1E       rr      7(ix)           ; toggle H & V flips
2401 3E 04            ld      a, #4
2403                    loc_0_2403:
2403 DD 77 0F          ld      0xF(ix), a           ; CODE XREF: sub_0_23DE+4↑j
2406 C9                ret
2406                    ; End of function sub_0_23DE
2407                    ;
2407                    ; SUBROUTINE
2407                    ;
2407                    sub_0_2407:
2407 DD 7E 14          ld      a, 0x14(ix)           ; CODE XREF: sub_0_1AC3+11C↑p
2407                    ; sub_0_1F72+151↑p ...
240A 07                rlca
240B 07                rlca
240C 07                rlca
240D 07                rlca
240E 4F                ld      c, a
240F E6 0F            and     #0xF
2411 67                ld      h, a
2412 79                ld      a, c
2413 E6 F0            and     #0xF0 ; '-'
2415 6F                ld      l, a
2416 DD 4E 13          ld      c, 0x13(ix)
2419 DD 46 12          ld      b, 0x12(ix)
241C ED 42            sbc     hl, bc
241E C9                ret
241E                    ; End of function sub_0_2407
241F                    ;
241F                    ; SUBROUTINE
241F                    ;

```

```

241F
241F
241F      sub_0_241F:
241F 11 00 01      ; CODE XREF: sub_0_1AC3+231p
241F      ld      de, #0x100
2422 3A 03 62      ld      a, (mario_y)
2425 FE 16      cp      #0x16
2427 D8      ret      C
2428 15      dec      d
2429 1C      inc      e
242A FE EA      cp      #0xEA ; 'Ë'
242C D0      ret      NC
242D 1D      dec      e
242E 3A 27 62      ld      a, (level_type)
2431 0F      rrrca
2432 D0      ret      NC
2433 3A 05 62      ld      a, (mario_x)
2436 FE 58      cp      #0x58 ; 'X'
2438 D0      ret      NC
2439 3A 03 62      ld      a, (mario_y)
243C FE 6C      cp      #0x6C ; 'l'
243E D0      ret      NC
243F 14      inc      d
2440 C9      ret
2440      ; End of function sub_0_241F
2440
2441
2441      ; ██████████ S U B R O U T I N E ██████████
2441
2441      sub_0_2441:
2441 21 0C 3F      ld      hl, #aNINTENDO+1
2444 3E 5E      ld      a, #0x5E ; '^'
2446 06 06      ld      b, #6
2448
2448      loc_0_2448:
2448 86      add      a, (hl)
2449 23      inc      hl
244A 10 FC      djnz   loc_0_2448
244C FD 21 10 63      ld      iy, #unk_0_6310
2450 A7      and      a
2451 CA 56 24      jp      Z, loc_0_2456
2454 FD 23      inc      iy
2456
2456      loc_0_2456:
2456 3A 27 62      ld      a, (level_type)
2459 3D      dec      a
245A 21 E4 3A      ld      hl, #barrel_level_tilemap_data
245D CA 71 24      jp      Z, loc_0_2471
2460 3D      dec      a
2461 21 5D 3B      ld      hl, #cement_pie_level_tilemap_data
2464 CA 71 24      jp      Z, loc_0_2471
2467 3D      dec      a
2468 21 E5 3B      ld      hl, #elevator_level_tilemap_data
246B CA 71 24      jp      Z, loc_0_2471
246E 21 8B 3C      ld      hl, #rivet_level_tilemap_data
2471
2471      loc_0_2471:
2471 DD 21 00 63      ld      ix, #unk_0_6300
2475 11 05 00      ld      de, #5
2478
2478      loc_0_2478:
2478 7E      ld      a, (hl)
2479 A7      and      a
247A CA 88 24      jp      Z, loc_0_2488
247D 3D      dec      a
247E CA 9E 24      jp      Z, loc_0_249E
2481 FE A9      cp      #0xA9 ; '@'
2483 C8      ret      Z
2484 19      add      hl, de
2485 C3 78 24      jp      loc_0_2478
2488
2488      ;
2488
2488      loc_0_2488:
2488 23      inc      hl
2489 7E      ld      a, (hl)
248A DD 77 00      ld      0(ix), a
248D 23      inc      hl
248E 7E      ld      a, (hl)
248F DD 77 15      ld      0x15(ix), a
2492 23      inc      hl
2493 23      inc      hl
2494 7E      ld      a, (hl)
2495 DD 77 2A      ld      0x2A(ix), a
2498 DD 23      inc      ix
249A 23      inc      hl
249B C3 78 24      jp      loc_0_2478
249E
249E      ;
249E
249E      loc_0_249E:
249E 23      inc      hl
249F 7E      ld      a, (hl)
24A0 FD 77 00      ld      0(iy), a
24A3 23      inc      hl
24A4 7E      ld      a, (hl)
24A5 FD 77 15      ld      0x15(iy), a
24A8 23      inc      hl
24A9 23      inc      hl
24AA 7E      ld      a, (hl)
24AB FD 77 2A      ld      0x2A(iy), a
24AE FD 23      inc      iy
24B0 23      inc      hl
24B1 C3 78 24      jp      loc_0_2478
24B1      ; End of function sub_0_2441
24B1
24B4
24B4      ; ██████████ S U B R O U T I N E ██████████
24B4
24B4
24B4      sub_0_24B4:
24B4 DD 7E 05      ld      a, 5(ix)
24B7 FE E8      cp      #0xE8 ; 'P'
24B9 D8      ret      C

```

```

24BA DD 7E 03      ld      a, 3(ix)
24BD FE 2A        cp      #0x2A ; '*'
24BF D0           ret      NC
24C0 FE 20        cp      #0x20 ; ' '
24C2 D8           ret      C
24C3 DD 7E 15      ld      a, 0x15(ix)
24C6 A7           and      a
24C7 CA D0 24      jp      Z, loc_0_24D0
24CA 3E 03         ld      a, #3
24CC 32 B9 62      ld      (unk_0_62B9), a
24CF AF           xor      a
24D0
24D0      loc_0_24D0:                                ; CODE XREF: sub_0_24B4+13|j
24D0 DD 77 00      ld      0(ix), a
24D3 DD 77 03      ld      3(ix), a
24D6 21 82 60      ld      hl, #digital_snd_tmr_thump
24D9 36 03         ld      (hl), #3                                ; tmr=3
24DB E1           pop      hl
24DC 3A 48 63      ld      a, (unk_0_6348)
24DF A7           and      a
24E0 C2 BA 21      jp      NZ, loc_0_21BA
24E3 3C           inc      a
24E4 32 48 63      ld      (unk_0_6348), a
24E7 C3 BA 21      jp      loc_0_21BA
24E7      ; End of function sub_0_24B4
24E7
24EA
24EA      ; ██████████ SUBROUTINE ██████████
24EA
24EA      sub_0_24EA:                                ; CODE XREF: 0000:1992|p
24EA 3E 02         ld      a, #2
24EC F7           rst      0x30                                ; return if level bit not set
24ED CD 23 25      call    sub_0_2523
24F0 CD 91 25      call    sub_0_2591
24F3 DD 21 A0 65   ld      ix, #unk_0_65A0
24F7 06 06         ld      b, #6                                ; 6 sprites to update
24F9 21 B8 69      ld      hl, #soft_sprite_ram+0xB8
24FC
24FC      loc_0_24FC:                                ; CODE XREF: sub_0_24EA+2F|j
24FC DD 7E 00      ld      a, 0(ix)
24FF A7           and      a
2500 CA 1C 25      jp      Z, loc_0_251C
2503 DD 7E 03      ld      a, 3(ix)                                ; sprite X
2506 77           ld      (hl), a
2507 2C           inc      l
2508 DD 7E 07      ld      a, 7(ix)                                ; sprite tile #
250B 77           ld      (hl), a
250C 2C           inc      l
250D DD 7E 08      ld      a, 8(ix)                                ; sprite v flip & palette
2510 77           ld      (hl), a
2511 2C           inc      l
2512 DD 7E 05      ld      a, 5(ix)                                ; sprite Y
2515 77           ld      (hl), a
2516 2C           inc      l
2517
2517      loc_0_2517:                                ; CODE XREF: sub_0_24EA+36|j
2517 DD 19         add      ix, de
2519 10 E1         djnz    loc_0_24FC
251B C9           ret
251C
251C      ; _____
251C      loc_0_251C:                                ; CODE XREF: sub_0_24EA+16|j
251C 7D           ld      a, 1
251D C6 04         add      a, #4
251F 6F           ld      l, a
2520 C3 17 25      jp      loc_0_2517
2520      ; End of function sub_0_24EA
2520
2523
2523      ; ██████████ SUBROUTINE ██████████
2523
2523      sub_0_2523:                                ; CODE XREF: sub_0_24EA+3|p
2523 21 9B 63      ld      hl, #unk_0_639B
2526 7E           ld      a, (hl)
2527 A7           and      a
2528 C2 8F 25      jp      NZ, loc_0_258F
252B 3A 9A 63      ld      a, (unk_0_639A)
252E A7           and      a
252F C8           ret      Z
2530 06 06         ld      b, #6
2532 11 10 00      ld      de, #0x10
2535 DD 21 A0 65   ld      ix, #unk_0_65A0
2539
2539      loc_0_2539:                                ; CODE XREF: sub_0_2523+1F|j
2539 DD CB 00 46     bit      0, 0(ix)
253D CA 45 25      jp      Z, loc_0_2545
2540 DD 19         add      ix, de
2542 10 F5         djnz    loc_0_2539
2544 C9           ret
2545
2545      ; _____
2545      loc_0_2545:                                ; CODE XREF: sub_0_2523+1A|j
2545 CD 57 00      call    rand
2548 FE 60        cp      #0x60 ; '0'
254A DD 36 05 7C   ld      5(ix), #0x7C ; '|'
254E DA 58 25      jp      C, loc_0_2558
2551 3A A3 62      ld      a, (unk_0_62A3)
2554 3D           dec      a
2555 C2 6E 25      jp      NZ, loc_0_256E
2558
2558      loc_0_2558:                                ; CODE XREF: sub_0_2523+2B|j
2558 DD 36 05 CC     ld      5(ix), #0xCC ; 'P'
255C 3A A6 62      ld      a, (unk_0_62A6)
255F 07           rlca
2560
2560      loc_0_2560:                                ; CODE XREF: sub_0_2523+50|j
2560 DD 36 03 07     ld      3(ix), #7
2564 D2 76 25      jp      NC, loc_0_2576
2567 DD 36 03 F8     ld      3(ix), #0xF8 ; 'o'
256B C3 76 25      jp      loc_0_2576
256E
256E      ; _____
256E      loc_0_256E:                                ; CODE XREF: sub_0_2523+32|j
256E CD 57 00      call    rand

```



```

2571 FE 68          cp      #0x68 ; 'h'
2573 C3 60 25      jp      loc_0_2560
2576
2576          ;
2576          loc_0_2576:
2576          ; CODE XREF: sub_0_2523+41↑j
2576          ; sub_0_2523+48↑j
2576 DD 36 00 01      ld      0(ix), #1
2576          ld      7(ix), #0x4B ; 'K'
2576          ld      9(ix), #8
2576          ld      0xA(ix), #3
2576          ld      a, #0x7C ; '|'
2576          ld      (unk_0_639B), a
2576          xor      a
2576          ld      (unk_0_639A), a
2576
2576          loc_0_258F:
2576          ; CODE XREF: sub_0_2523+5↑j
2576          dec      (hl)
2576          ret
2576          ; End of function sub_0_2523
2576
2576          ;
2576          ; SUBROUTINE
2576
2576          sub_0_2591:
2576          ; CODE XREF: sub_0_24EA+6↑p
2576          DD 21 A0 65      ld      ix, #unk_0_65A0
2576          11 10 00      ld      de, #0x10
2576          06 06          ld      b, #6
2576
2576          loc_0_259A:
2576          ; CODE XREF: sub_0_2591+2C↑j
2576          DD CB 00 46      bit      0, 0(ix)
2576          CA BB 25      jp      Z, loc_0_25BB
2576          DD 7E 03          ld      a, 3(ix)
2576          67          ld      h, a
2576          C6 07          add     a, #7
2576          FE 0E          cp      #0xE
2576          DA D6 25      jp      C, loc_0_25D6
2576          DD 7E 05      ld      a, 5(ix)
2576          FE 7C          cp      #0x7C ; '|'
2576          CA C0 25      jp      Z, loc_0_25C0
2576          3A A6 63      ld      a, (unk_0_63A6)
2576          84          add     a, h
2576          DD 77 03      ld      3(ix), a
2576
2576          loc_0_25BB:
2576          ; CODE XREF: sub_0_2591+D↑j
2576          ; sub_0_2591+42↑j ...
2576          DD 19          add     ix, de
2576          10 DB          djnz   loc_0_259A
2576          C9          ret
2576          ;
2576          loc_0_25C0:
2576          ; CODE XREF: sub_0_2591+20↑j
2576          7C          ld      a, h
2576          FE 80          cp      #0x80 ; 'Q'
2576          CA D6 25      jp      Z, loc_0_25D6
2576          3A A5 63      ld      a, (unk_0_63A5)
2576          D2 CF 25      jp      NC, loc_0_25CF
2576          3A A4 63      ld      a, (unk_0_63A4)
2576
2576          loc_0_25CF:
2576          ; CODE XREF: sub_0_2591+38↑j
2576          84          add     a, h
2576          DD 77 03      ld      3(ix), a
2576          C3 BB 25      jp      loc_0_25BB
2576          ;
2576          loc_0_25D6:
2576          ; CODE XREF: sub_0_2591+18↑j
2576          ; sub_0_2591+32↑j
2576          21 B8 69      ld      hl, #soft_sprite_ram+0xB8
2576          3E 06          ld      a, #6
2576          90          sub     b
2576
2576          loc_0_25DC:
2576          ; CODE XREF: sub_0_2591+53↑j
2576          CA E7 25      jp      Z, loc_0_25E7
2576          2C          inc     l
2576          2C          inc     l
2576          2C          inc     l
2576          2C          inc     l
2576          3D          dec     a
2576          C3 DC 25      jp      loc_0_25DC
2576          ;
2576          loc_0_25E7:
2576          ; CODE XREF: sub_0_2591+4B↑j
2576          AF          xor      a
2576          DD 77 00      ld      0(ix), a
2576          DD 77 03      ld      3(ix), a
2576          77          ld      (hl), a
2576          C3 BB 25      jp      loc_0_25BB
2576          ; End of function sub_0_2591
2576
2576          ;
2576          ; SUBROUTINE
2576
2576          sub_0_25F2:
2576          ; CODE XREF: 0000:19AA↑p
2576          3E 02          ld      a, #2
2576          F7          rst      0x30
2576          ; return if level bit not set
2576          CD 02 26      call   sub_0_2602
2576          CD 2F 26      call   sub_0_262F
2576          CD 79 26      call   sub_0_2679
2576          CD D3 2A      call   sub_0_2AD3
2576          C9          ret
2576          ; End of function sub_0_25F2
2576
2576          ;
2576          ; SUBROUTINE
2576
2576          sub_0_2602:
2576          ; CODE XREF: 0000:16D5↑p
2576          ; sub_0_25F2+3↑p
2576          0F          ld      a, (gen_purpose_timer)
2576          rrca
2576          DA 16 26      jp      C, loc_0_2616
2576          21 A0 62      ld      hl, #unk_0_62A0
2576          35          dec     (hl)
2576          C2 16 26      jp      NZ, loc_0_2616
2576          36 80          ld      (hl), #0x80 ; 'Q'

```

```

2612 2C          inc      1
2613 CD DE 26    call     sub_0_26DE
2616
2616          loc_0_2616:
2616          ; CODE XREF: sub_0_2602+41j
2616          ; sub_0_2602+B1j
2616
2619          ld         hl, #unk_0_62A1
2619          call        sub_0_26E9
261C 32 A3 63    ld         (unk_0_63A3), a
261F 3A 1A 60    ld         a, (gen_purpose_timer)
2622 E6 1F      and        #0x1F
2624 FE 01      cp         #1
2626 C0          ret        NZ
2627 11 E4 69    ld         de, #soft_sprite_ram+0xE4
262A EB          ex         de, hl
262B CD A6 26    call     sub_0_26A6
262E C9          ret
262E          ; End of function sub_0_2602
262E
262F          ; SUBROUTINE
262F
262F          sub_0_262F:
262F          ; CODE XREF: sub_0_25F2+61p
262F 21 A3 62    ld         hl, #unk_0_62A3
2632 3A 05 62    ld         a, (mario_x)
2635 FE C0      cp         #0xC0 ; 'L'
2637 DA 6F 26    jp         C, loc_0_266F
263A 3A 1A 60    ld         a, (gen_purpose_timer)
263D 0F          rrca
263E DA 4C 26    jp         C, loc_0_264C
2641 2D          dec        l
2642 35          dec        (hl)
2643 C2 4C 26    jp         NZ, loc_0_264C
2646 36 C0      ld         (hl), #0xC0 ; 'L'
2648 2C          inc        l
2649 CD DE 26    call     sub_0_26DE
264C
264C          loc_0_264C:
264C          ; CODE XREF: sub_0_262F+F1j
264C          ; sub_0_262F+141j ...
264C
264F          ld         hl, #unk_0_62A3
264F          call        sub_0_26E9
2652 32 A5 63    ld         (unk_0_63A5), a
2655 ED 44      neg
2657 32 A4 63    ld         (unk_0_63A4), a
265A 3A 1A 60    ld         a, (gen_purpose_timer)
265D E6 1F      and        #0x1F
265F C0          ret        NZ
2660 2D          dec        l
2661 11 EC 69    ld         de, #soft_sprite_ram+0xEC
2664 EB          ex         de, hl
2665 CD A6 26    call     sub_0_26A6
2668 E6 7F      and        #0x7F ; ' '
266A 21 ED 69    ld         hl, #soft_sprite_ram+0xED
266D 77          ld         (hl), a
266E C9          ret
266F
266F          loc_0_266F:
266F          ; CODE XREF: sub_0_262F+81j
266F CB 7E          bit         7, (hl)
2671 C2 4C 26    jp         NZ, loc_0_264C
2674 36 FF      ld         (hl), #0xFF
2676 C3 4C 26    jp         loc_0_264C
2676          ; End of function sub_0_262F
2676
2677          ; SUBROUTINE
2677
2677          sub_0_2679:
2677          ; CODE XREF: sub_0_25F2+91p
2677 3A 1A 60    ld         a, (gen_purpose_timer)
267C 0F          rrca
267D DA 8D 26    jp         C, loc_0_268D
2680 21 A5 62    ld         hl, #unk_0_62A5
2683 35          dec        (hl)
2684 C2 8D 26    jp         NZ, loc_0_268D
2687 36 FF      ld         (hl), #0xFF
2689 2C          inc        l
268A CD DE 26    call     sub_0_26DE
268D
268D          loc_0_268D:
268D          ; CODE XREF: sub_0_2679+41j
268D          ; sub_0_2679+B1j
268D
2690          ld         hl, #unk_0_62A6
2690          call        sub_0_26E9
2693 32 A6 63    ld         (unk_0_63A6), a
2696 3A 1A 60    ld         a, (gen_purpose_timer)
2699 E6 1F      and        #0x1F
269B FE 02      cp         #2
269D C0          ret        NZ
269E 11 F4 69    ld         de, #soft_sprite_ram+0xF4
26A1 EB          ex         de, hl
26A2 CD A6 26    call     sub_0_26A6
26A5 C9          ret
26A5          ; End of function sub_0_2679
26A5
26A6          ; SUBROUTINE
26A6
26A6          sub_0_26A6:
26A6          ; CODE XREF: sub_0_2602+291p
26A6          ; sub_0_262F+361p ...
26A6 2C          inc        l
26A7 1A          ld         a, (de)
26A8 17          rla
26A9 DA C5 26    jp         C, loc_0_26C5
26AC 7E          ld         a, (hl)
26AD 3C          inc        a
26AE FE 53      cp         #0x53 ; 'S'
26B0 C2 B5 26    jp         NZ, loc_0_26B5
26B3 3E 50      ld         a, #0x50 ; 'P'
26B5
26B5          loc_0_26B5:
26B5          ; CODE XREF: sub_0_26A6+A1j
26B5 77          ld         (hl), a
26B6 7D          ld         a, 1
26B7 C6 04      add         a, #4
26B9 6F          ld         l, a
26BA 7E          ld         a, (hl)

```

```

26BB 3D          dec      a
26BC FE CF      cp       #0xCF ; 'a'
26BE C2 C3 26   jp       NZ, loc_0_26C3
26C1 3E D2      ld       a, #0xD2 ; 'E'
26C3
26C3          loc_0_26C3:
26C3 77          ld       (hl), a                ; CODE XREF: sub_0_26A6+18j
26C4 C9          ret
26C5
26C5          loc_0_26C5:
26C5 7E          ld       a, (hl)                ; CODE XREF: sub_0_26A6+31j
26C6 3D          dec      a
26C7 FE 4F      cp       #0x4F ; 'O'
26C9 C2 CE 26   jp       NZ, loc_0_26CE
26CC 3E 52      ld       a, #0x52 ; 'R'
26CE
26CE          loc_0_26CE:
26CE 77          ld       (hl), a                ; CODE XREF: sub_0_26A6+231j
26CF 7D          ld       a, 1
26D0 C6 04      add      a, #4
26D2 6F          ld       l, a
26D3 7E          ld       a, (hl)
26D4 3C          inc      a
26D5 FE D3      cp       #0xD3 ; 'ë'
26D7 C2 DC 26   jp       NZ, loc_0_26DC
26DA 3E D0      ld       a, #0xD0 ; 'ð'
26DC
26DC          loc_0_26DC:
26DC 77          ld       (hl), a                ; CODE XREF: sub_0_26A6+311j
26DD C9          ret
26DD          ; End of function sub_0_26A6
26DE
26DE          ; ██████████ S U B R O U T I N E ██████████
26DE
26DE          sub_0_26DE:
26DE CB 7E          bit      7, (hl)                ; CODE XREF: sub_0_2602+111p
26DE          ; sub_0_262F+1A1p ...
26DE CA E6 26      jp       Z, loc_0_26E6
26E3 36 02      ld       (hl), #2
26E5 C9          ret
26E6
26E6          loc_0_26E6:
26E6 36 FE          ld       (hl), #0xFE ; '■'                ; CODE XREF: sub_0_26DE+21j
26E8 C9          ret
26E8          ; End of function sub_0_26DE
26E9
26E9          ; ██████████ S U B R O U T I N E ██████████
26E9
26E9          sub_0_26E9:
26E9          ; CODE XREF: sub_0_2602+171p
26E9          ; sub_0_262F+201p ...
26E9 3A 1A 60      ld       a, (gen_purpose_timer)
26EC E6 01      and      #1
26EE C8          ret       Z
26EF CB 7E          bit      7, (hl)
26F1 3E FF      ld       a, #0xFF
26F3 C2 F8 26   jp       NZ, loc_0_26F8
26F6 3E 01      ld       a, #1
26F8
26F8          loc_0_26F8:
26F8 77          ld       (hl), a                ; CODE XREF: sub_0_26E9+A1j
26F9 C9          ret
26F9          ; End of function sub_0_26E9
26FA
26FA          ; ██████████ S U B R O U T I N E ██████████
26FA
26FA          sub_0_26FA:
26FA          ; CODE XREF: 0000:19A71p
26FA 3E 04      ld       a, #4
26FC F7          rst       0x30                ; return if level bit not set
26FD 3A 05 62   ld       a, (mario_x)
2700 FE F0      cp       #0xF0 ; '-'
2702 D2 7F 27   jp       NC, mario_dies_on_elevator
2705 3A 29 62   ld       a, (level)
2708 3D          dec      a
2709 3A 1A 60      ld       a, (gen_purpose_timer)
270C C2 1A 27   jp       NZ, loc_0_271A
270F E6 03      and      #3
2711 FE 01      cp       #1
2713 CA 1E 27   jp       Z, loc_0_271E
2716 DA 22 27   jp       C, loc_0_2722
2719 C9          ret
271A
271A          loc_0_271A:
271A          ; CODE XREF: sub_0_26FA+121j
271A 0F          rrca
271B DA 22 27   jp       C, loc_0_2722
271E
271E          loc_0_271E:
271E CD 45 27   call      sub_0_2745                ; CODE XREF: sub_0_26FA+191j
2721 C9          ret
2722
2722          loc_0_2722:
2722          ; CODE XREF: sub_0_26FA+1C1j
2722          ; sub_0_26FA+211j
2722 CD 97 27   call      sub_0_2797
2725 CD DA 27   call      sub_0_27DA
2728 06 06      ld       b, #6
272A 11 00 00      ld       de, #0x10
272D 21 58 69      ld       hl, #soft_sprite_ram+0x58
2730 DD 21 00 66   ld       ix, #unk_0_6600
2734
2734          loc_0_2734:
2734          ; CODE XREF: sub_0_26FA+481j
2734          ; store coordinates
2734 DD 7E 03      ld       a, 3(ix)
2737 77          ld       (hl), a
2738 2C          inc      l
2739 2C          inc      l
273A 2C          inc      l
273B DD 7E 05      ld       a, 5(ix)
273E 77          ld       (hl), a

```

```

273F 2C          inc     1
2740 DD 19      add     ix, de
2742 10 F0      djnz    loc_0_2734
2744 C9          ret
2744          ; End of function sub_0_26FA
2744
2745          ; SUBROUTINE
2745
2745 sub_0_2745:
2745          ; CODE XREF: sub_0_26FA+24|p
2745 3A 98 63     ld      a, (mario_on_elevator)
2748 A7          and     a
2749 C8          ret     Z
2749          ; on elevator?
274A 3A 16 62     ld      a, (mario_jumping)
274D A7          and     a
274E C0          ret     NZ
274E          ; jumping?
274F 3A 03 62     ld      a, (mario_y)
2752 FE 2C      cp      #0x2C ; ','
2754 DA 66 27     jp      C, loc_0_2766
2754          ; not not elevator
2757 FE 43      cp      #0x43 ; 'C'
2759 DA 6F 27     jp      C, loc_0_276F
2759          ; on left elevator
275C FE 6C      cp      #0x6C ; 'l'
275E DA 66 27     jp      C, loc_0_2766
275E          ; not on elevator
2761 FE 83      cp      #0x83 ; 'a'
2763 DA 87 27     jp      C, loc_0_2787
2763          ; on right elevator
2766
2766 loc_0_2766:
2766          ; CODE XREF: sub_0_2745+F|j
2766 AF          ; sub_0_2745+19|j
2766          ; mark off elevator
2767 32 98 63     ld      (mario_on_elevator), a
276A 3C          inc     a
276B 32 21 62     ld      (unk_0_6221), a
276E C9          ret
276F
276F          ;
276F
276F loc_0_276F:
276F          ; CODE XREF: sub_0_2745+14|j
276F 3A 05 62     ld      a, (mario_x)
2772 FE 71      cp      #0x71 ; 'q'
2774 DA 7F 27     jp      C, mario_dies_on_elevator
2774          ; make mario die
2777 3D          dec     a
2777          ; on upwards moving elevator
2778 32 05 62     ld      (mario_x), a
277B 32 4F 69     ld      (soft_sprite_ram+0x4F), a
277E C9          ret
277F
277F          ;
277F
277F mario_dies_on_elevator:
277F          ; CODE XREF: sub_0_26FA+8|j
277F AF          ; sub_0_2745+2F|j ...
277F          ;
277F 32 00 62     ld      (mario_alive_flag), a
2783 32 98 63     ld      (mario_on_elevator), a
2786 C9          ret
2787
2787          ;
2787
2787 loc_0_2787:
2787          ; CODE XREF: sub_0_2745+1E|j
2787 3A 05 62     ld      a, (mario_x)
278A FE E8      cp      #0xE8 ; 'P'
278C D2 7F 27     jp      NC, mario_dies_on_elevator
278F 3C          inc     a
278F          ; on downwards moving elevator
2790 32 05 62     ld      (mario_x), a
2793 32 4F 69     ld      (soft_sprite_ram+0x4F), a
2796 C9          ret
2796          ; End of function sub_0_2745
2796
2797          ; SUBROUTINE
2797
2797
2797 sub_0_2797:
2797          ; CODE XREF: sub_0_26FA+28|p
2797 06 06      ld      b, #6
2799 11 10 00     ld      de, #0x10
2799          ; move elevators to the right side
279C DD 21 00 66 ld      ix, #unk_0_6600
27A0
27A0 loc_0_27A0:
27A0          ; CODE XREF: sub_0_2797+2D|j
27A0 DD CB 00 46 bit     0, 0(ix)
27A4 CA C2 27   jp      Z, loc_0_27C2
27A7 DD CB 0D 5E bit     3, 0xD(ix)
27AB CA C7 27   jp      Z, loc_0_27C7
27AE DD 7E 05   ld      a, 5(ix)
27B1 3D          dec     a
27B2 DD 77 05   ld      5(ix), a
27B5 FE 60      cp      #0x60 ; 'o'
27B7 C2 C2 27   jp      NZ, loc_0_27C2
27BA DD 36 03 77 ld      3(ix), #0x77 ; 'w'
27BE DD 36 0D 04 ld      0xD(ix), #4
27C2
27C2 loc_0_27C2:
27C2          ; CODE XREF: sub_0_2797+D|j
27C2 DD 19          ; sub_0_2797+20|j ...
27C2          ;
27C4 10 DA      djnz    loc_0_27A0
27C6 C9          ret
27C7
27C7          ;
27C7
27C7 loc_0_27C7:
27C7          ; CODE XREF: sub_0_2797+14|j
27C7 DD 7E 05     ld      a, 5(ix)
27CA 3C          inc     a
27CB DD 77 05     ld      5(ix), a
27CE FE F8      cp      #0xF8 ; 'o'
27D0 C2 C2 27     jp      NZ, loc_0_27C2
27D3 DD 36 00 00 ld      0(ix), #0
27D7 C3 C2 27     jp      loc_0_27C2
27D7          ; End of function sub_0_2797
27D7
27DA          ; SUBROUTINE
27DA
27DA
27DA sub_0_27DA:
27DA          ; CODE XREF: sub_0_26FA+2B|p
27DA 21 A7 62     ld      hl, #unk_0_62A7
27DD 7E          ; move elevators to the left side
27DE A7          ld      a, (hl)
27DF C2 06 28     jp      NZ, loc_0_2806
27E2 06 06      ld      b, #6
27E4 DD 21 00 66 ld      ix, #unk_0_6600
27E8
27E8 loc_0_27E8:
27E8          ; CODE XREF: sub_0_27DA+17|j

```

```

27E8 DD CB 00 46      bit      0, 0(ix)
27EC CA F4 27      jp      Z, loc_0_27F4
27EF DD 19      add      ix, de
27F1 10 F5      djnz     loc_0_27E8
27F3 C9      ret
27F4      ;
27F4      loc_0_27F4:
27F4 DD 36 00 01      ld      0(ix), #1
27F8 DD 36 03 37      ld      3(ix), #0x37 ; '7'
27FC DD 36 05 F8      ld      5(ix), #0xF8 ; 'o'
2800 DD 36 0D 08      ld      0xD(ix), #8
2804 36 34      ld      (hl), #0x34 ; '4'
2806
2806      loc_0_2806:
2806 35      dec      (hl)
2807 C9      ret
2807      ; End of function sub_0_27DA
2807
2808      ;
2808      ; SUBROUTINE
2808
2808      sub_0_2808:
2808 FD 21 00 62      ld      iy, #mario_alive_flag
280C 3A 05 62      ld      a, (mario_x)
280F 4F      ld      c, a
2810 21 07 04      ld      hl, #0x407
2813 CD 6F 28      call   sub_0_286F
2816 A7      and      a
2817 C8      ret      Z
2818 3D      dec      a
2819 32 00 62      ld      (mario_alive_flag), a
281C C9      ret
281C      ; End of function sub_0_2808
281C
281D      ;
281D      ; SUBROUTINE
281D
281D      sub_0_281D:
281D 06 02      ld      b, #2
281F 11 10 00      ld      de, #0x10
2822 FD 21 80 66      ld      iy, #unk_0_6680
2826      ; CODE XREF: sub_0_281D+12j
2826      loc_0_2826:
2826 FD CB 01 46      bit      0, 1(iy)
282A C2 32 28      jp      NZ, loc_0_2832
282D FD 19      add      iy, de
282F 10 F5      djnz     loc_0_2826
2831 C9      ret
2832      ;
2832      loc_0_2832:
2832 FD 4E 05      ld      c, 5(iy)
2835 FD 66 09      ld      h, 9(iy)
2838 FD 6E 0A      ld      l, 0xA(iy)
283B CD 6F 28      call   sub_0_286F
283E A7      and      a
283F C8      ret      Z
2840 32 50 63      ld      (unk_0_6350), a
2843 3A B9 63      ld      a, (unk_0_63B9)
2846 90      sub      b
2847 32 54 63      ld      (unk_0_6354), a
284A 7B      ld      a, e
284B 32 53 63      ld      (unk_0_6353), a
284E DD 22 51 63      ld      (unk_0_6351), ix
2852 C9      ret
2852      ; End of function sub_0_281D
2852
2853      ;
2853      ; SUBROUTINE
2853
2853      sub_0_2853:
2853 FD 21 00 62      ld      iy, #mario_alive_flag
2857 3A 05 62      ld      a, (mario_x)
285A      loc_0_285A:
285A C6 0C      add      a, #0xC
285C 4F      ld      c, a
285D 3A 10 60      ld      a, (controller_in)
2860 E6 03      and      #3
2862 21 08 05      ld      hl, #0x508
2865 CA 6B 28      jp      Z, loc_0_286B
2868 21 08 13      ld      hl, #0x1308
286B      loc_0_286B:
286B CD 88 3E      call   sub_0_3E88
286E C9      ret
286E      ; End of function sub_0_2853
286E
286F      ;
286F      ; SUBROUTINE
286F
286F      sub_0_286F:
286F 3A 27 62      ld      a, (level_type)
2872 E5      push     hl
2873 EF      rst      0x28
2873      ; go!
2873      ;
2874 00 00      .dw 0
2876 80 28      .dw 11_check_hammer_hit
2878 B0 28      .dw 12_check_hammer_hit
287A E0 28      .dw 13_check_hammer_hit
287C 01 29      .dw 14_check_hammer_hit
287E 00 00      .dw 0
2880      ;
2880      11_check_hammer_hit:
2880 E1      pop      hl
2881 06 0A      ld      b, #0xA
2883 78      ld      a, b
2884 32 B9 63      ld      (unk_0_63B9), a
2887 11 20 00      ld      de, #0x20 ; ' '

```

```

288A DD 21 00 67      ld      ix, #unk_0_6700
288E CD 13 29      call    sub_0_2913
2891 06 05      ld      b, #5
2893 78      ld      a, b
2894 32 B9 63      ld      (unk_0_63B9), a
2897 1E 20      ld      e, #0x20 ; ' '
2899 DD 21 00 64      ld      ix, #unk_0_6400      ; fireball character data
289D CD 13 29      call    sub_0_2913
28A0 06 01      ld      b, #1
28A2 78      ld      a, b
28A3 32 B9 63      ld      (unk_0_63B9), a
28A6 1E 00      ld      e, #0
28A8 DD 21 A0 66      ld      ix, #unk_0_66A0
28AC CD 13 29      call    sub_0_2913
28AF C9      ret
28AF      ; End of function sub_0_286F
28AF
28B0      ; -----
28B0
28B0      12_check_hammer_hit:      ; DATA XREF: sub_0_286F+9|o
28B0 E1      ; sub_0_3E88+9|o
28B0      pop      hl
28B1 06 05      ld      b, #5
28B3 78      ld      a, b
28B4 32 B9 63      ld      (unk_0_63B9), a
28B7 11 20 00      ld      de, #0x20 ; ' '
28BA DD 21 00 64      ld      ix, #unk_0_6400      ; fireball character data
28BE CD 13 29      call    sub_0_2913
28C1 06 06      ld      b, #6
28C3 78      ld      a, b
28C4 32 B9 63      ld      (unk_0_63B9), a
28C7 1E 10      ld      e, #0x10
28C9 DD 21 A0 65      ld      ix, #unk_0_65A0
28CD CD 13 29      call    sub_0_2913
28D0 06 01      ld      b, #1
28D2 78      ld      a, b
28D3 32 B9 63      ld      (unk_0_63B9), a
28D6 1E 00      ld      e, #0
28D8 DD 21 A0 66      ld      ix, #unk_0_66A0
28DC CD 13 29      call    sub_0_2913
28DF C9      ret
28E0      ; -----
28E0
28E0      13_check_hammer_hit:      ; DATA XREF: sub_0_286F+B|o
28E0 E1      ; sub_0_3E88+B|o
28E0      pop      hl
28E1 06 05      ld      b, #5
28E3 78      ld      a, b
28E4 32 B9 63      ld      (unk_0_63B9), a
28E7 11 20 00      ld      de, #0x20 ; ' '
28EA DD 21 00 64      ld      ix, #unk_0_6400      ; fireball character data
28EE CD 13 29      call    sub_0_2913
28F1 06 0A      ld      b, #0xA
28F3 78      ld      a, b
28F4 32 B9 63      ld      (unk_0_63B9), a
28F7 1E 10      ld      e, #0x10
28F9 DD 21 00 65      ld      ix, #unk_0_6500      ; check if hammer hits a spring
28FD CD 13 29      call    sub_0_2913
2900 C9      ret
2901      ; -----
2901
2901      14_check_hammer_hit:      ; DATA XREF: sub_0_286F+D|o
2901 E1      ; sub_0_3E88+D|o
2901      pop      hl
2902 06 07      ld      b, #7
2904 78      ld      a, b
2905 32 B9 63      ld      (unk_0_63B9), a
2908 11 20 00      ld      de, #0x20 ; ' '
290B DD 21 00 64      ld      ix, #unk_0_6400      ; fireball character data
290F CD 13 29      call    sub_0_2913
2912 C9      ret
2913
2913      ; [REDACTED] S U B R O U T I N E [REDACTED]
2913
2913      sub_0_2913:      ; CODE XREF: sub_0_286F+1F|p
2913 DD E5      ; sub_0_286F+2E|p ...
2913      push     ix
2915
2915      loc_0_2915:      ; CODE XREF: sub_0_2913+3B|j
2915 DD CB 00 46      bit      0, 0(ix)      ; check if hammer hits something else
2919 CA 4C 29      jp      Z, loc_0_294C
291C 79      ld      a, c
291D DD 96 05      sub     5(ix)
2920 D2 25 29      jp      NC, loc_0_2925
2923 ED 44      neg
2925
2925      loc_0_2925:      ; CODE XREF: sub_0_2913+D|j
2925 3C      inc     a
2926 95      sub     l
2927 DA 30 29      jp      C, loc_0_2930
292A DD 96 0A      sub     0xA(ix)
292D D2 4C 29      jp      NC, loc_0_294C
2930
2930      loc_0_2930:      ; CODE XREF: sub_0_2913+14|j
2930 FD 7E 03      ld      a, 3(iy)
2933 DD 96 03      sub     3(ix)
2936 D2 3B 29      jp      NC, loc_0_293B
2939 ED 44      neg
293B
293B      loc_0_293B:      ; CODE XREF: sub_0_2913+23|j
293B 94      sub     h
293C DA 45 29      jp      C, loc_0_2945
293F DD 96 09      sub     9(ix)
2942 D2 4C 29      jp      NC, loc_0_294C
2945
2945      loc_0_2945:      ; CODE XREF: sub_0_2913+29|j
2945 3E 01      ld      a, #1
2947 DD E1      pop     ix
2949 33      inc     sp
294A 33      inc     sp
294B C9      ret
294C      ; -----
294C
294C      loc_0_294C:      ; CODE XREF: sub_0_2913+6|j
294C DD 19      ; sub_0_2913+1A|j ...

```

```

294C      add     ix, de
294E 10 C5      djnz    loc_0_2915
2950 AF        xor     a
2951 DD E1      pop     ix
2953 C9        ret
2953      ; End of function sub_0_2913
2953
2954      ; SUBROUTINE
2954
2954      sub_0_2954:                                ; CODE XREF: sub_0_1AC3+171|p
2954 3E 0B      ld      a, #0xB
2956 F7      rst      0x30                          ; return if level bit not set
2957 CD 74 29   call    sub_0_2974
295A 32 18 62   ld      (unk_0_6218), a
295D 0F        rrca
295E 0F        rrca
295F 32 85 60   ld      (digital_snd_tmr_barrel_jump_priz), a
2962 78        ld      a, b
2963 A7        and     a
2964 C8        ret
2965 FE 01      cp      #1
2967 CA 6F 29   jp      Z, loc_0_296F
296A DD 36 01 01 ld      1(ix), #1
296E C9        ret
296F      ;
296F
296F      loc_0_296F:                                ; CODE XREF: sub_0_2954+13|j
296F DD 36 11 01 ld      0x11(ix), #1
2973 C9        ret
2973      ; End of function sub_0_2954
2973
2974      ; SUBROUTINE
2974
2974      sub_0_2974:                                ; CODE XREF: sub_0_2954+3|p
2974 FD 21 00 62 ld      iy, #mario_alive_flag
2978 3A 05 62   ld      a, (mario_x)
297B 4F        ld      c, a
297C 21 08 04   ld      hl, #0x408
297F 06 02      ld      b, #2
2981 11 10 00   ld      de, #0x10
2984 DD 21 80 66 ld      ix, #unk_0_6680      ; hammer character data
2988 CD 13 29   call    sub_0_2913
298B C9        ret
298B      ; End of function sub_0_2974
298B
298B      ; SUBROUTINE
298B
298C      sub_0_298C:                                ; CODE XREF: sub_0_3202+3C|p
298C 2A C8 63   ld      hl, (unk_0_63C8)
298F 7D        ld      a, 1
2990 C6 0E      add     a, #0xE
2992 6F        ld      l, a
2993 56        ld      d, (hl)
2994 2C        inc     l
2995 7E        ld      a, (hl)
2996 C6 0C      add     a, #0xC
2998 5F        ld      e, a
2999 EB        ex      de, hl
299A CD F0 2F   call    get_tilemap_addr_from_coords
299D 7E        ld      a, (hl)
299E FE B0      cp      #0xB0 ; '0'
29A0 DA AC 29   jp      C, loc_0_29AC
29A3 E6 0F      and     #0xF
29A5 FE 08      cp      #8
29A7 D2 AC 29   jp      NC, loc_0_29AC
29AA AF        xor     a
29AB C9        ret
29AC      ;
29AC
29AC      loc_0_29AC:                                ; CODE XREF: sub_0_298C+14|j
29AC 3E 01      ld      a, #1      ; sub_0_298C+1B|j
29AE C9        ret
29AE      ; End of function sub_0_298C
29AE
29AF      ; SUBROUTINE
29AF
29AF      sub_0_29AF:                                ; CODE XREF: sub_0_2B1C+7|p
29AF 3E 04      ld      a, #4
29B1 F7      rst      0x30                          ; return if level bit not set
29B2 FD 21 00 62 ld      iy, #mario_alive_flag
29B6 3A 05 62   ld      a, (mario_x)
29B9 4F        ld      c, a
29BA 21 08 04   ld      hl, #0x408
29BD CD 22 2A   call    sub_0_2A22
29C0 A7        and     a
29C1 CA 20 2A   jp      Z, loc_0_2A20
29C4 3E 06      ld      a, #6
29C6 90        sub     b
29C7
29C7      loc_0_29C7:                                ; CODE XREF: sub_0_29AF+1E|j
29C7 CA D0 29   jp      Z, loc_0_29D0
29CA DD 19      add     ix, de
29CC 3D        dec     a
29CD C3 C7 29   jp      loc_0_29C7
29D0      ;
29D0
29D0      loc_0_29D0:                                ; CODE XREF: sub_0_29AF+18|j
29D0 DD 7E 05   ld      a, 5(ix)
29D3 D6 04      sub     #4
29D5 57        ld      d, a
29D6 3A 0C 62   ld      a, (mario_y_before_jump)
29D9 C6 05      add     a, #5
29DB BA        cp      d
29DC D2 EE 29   jp      NC, loc_0_29EE      ; check if on or below elevator
29DF 7A        ld      a, d
29E0 D6 08      sub     #8
29E2 32 05 62   ld      (mario_x), a

```

```

29E5 3E 01          ld      a, #1                ; flag on elevator
29E7 47             ld      b, a
29E8 32 98 63       ld      (mario_on_elevator), a
29EB 33             inc     sp
29EC 33             inc     sp
29ED C9            ret
29EE
29EE
loc_0_29EE:
29EE 3A 0C 62       ld      a, (mario_y_before_jump)    ; CODE XREF: sub_0_29AF+2D↑j
29F1 D6 0E         sub     #0xE                ; collide with side of elevator
29F3 BA            cp      d
29F4 D2 1B 2A       jp      NC, loc_0_2A1B
29F7 3A 10 62       ld      a, (unk_0_6210)
29FA A7            and     a
29FB 3A 03 62       ld      a, (mario_y)
29FE CA 08 2A       jp      Z, loc_0_2A08
2A01 F6 07         or      #7
2A03 D6 04         sub     #4
2A05 C3 0E 2A       jp      loc_0_2A0E
2A08
2A08
2A08
2A08 D6 08         sub     #8                ; CODE XREF: sub_0_29AF+4F↑j
2A0A F6 07         or      #7
2A0C C6 04         add     a, #4
2A0E
loc_0_2A0E:
2A0E                ; CODE XREF: sub_0_29AF+56↑j
2A0E 32 03 62       ld      (mario_y), a
2A11 32 4C 69       ld      (soft_sprite_ram+0x4C), a
2A14 3E 01         ld      a, #1
2A16 06 00         ld      b, #0
2A18 33             inc     sp
2A19 33             inc     sp
2A1A C9            ret
2A1B
2A1B
loc_0_2A1B:
2A1B                ; CODE XREF: sub_0_29AF+45↑j
2A1B AF            xor     a
2A1C 32 00 62       ld      (mario_alive_flag), a
2A1F C9            ret
2A20
2A20
loc_0_2A20:
2A20                ; CODE XREF: sub_0_29AF+12↑j
2A20 47             ld      b, a
2A21 C9            ret
; End of function sub_0_29AF
2A21
2A21
2A22
2A22
; ██████████ S U B R O U T I N E ██████████
2A22
2A22
sub_0_2A22:
2A22                ; CODE XREF: sub_0_29AF+E↑p
2A22 06 06         ld      b, #6
2A24 11 10 00       ld      de, #0x10
2A27 DD 21 00 66    ld      ix, #unk_0_6600
2A2B CD 13 29       call   sub_0_2913
2A2E C9            ret
; End of function sub_0_2A22
2A2E
2A2E
2A2F
2A2F
; ██████████ S U B R O U T I N E ██████████
2A2F
2A2F
sub_0_2A2F:
2A2F                ; CODE XREF: sub_0_1F72+E5↑p
2A2F                ; sub_0_1F72+188↑p
2A2F DD 7E 03       ld      a, 3(ix)
2A32 67             ld      h, a
2A33 DD 7E 05       ld      a, 5(ix)
2A36 C6 04         add     a, #4
2A38 6F            ld      l, a
2A39 E5            push   hl
2A3A CD F0 2F       call   get_tilemap_addr_from_coords
2A3D D1            pop     de
2A3E 7E            ld      a, (hl)
2A3F FE B0         cp      #0xB0 ; 'L'
2A41 DA 7B 2A       jp      C, loc_0_2A7B
2A44 E6 0F         and     #0xF
2A46 FE 08         cp      #8
2A48 D2 7B 2A       jp      NC, loc_0_2A7B
2A4B 7E            ld      a, (hl)
2A4C FE C0         cp      #0xC0 ; 'I'
2A4E CA 7B 2A       jp      Z, loc_0_2A7B
2A51 DA 69 2A       jp      C, loc_0_2A69
2A54 FE D0         cp      #0xD0 ; '8'
2A56 DA 6E 2A       jp      C, loc_0_2A6E
2A59 FE E0         cp      #0xE0 ; '6'
2A5B DA 63 2A       jp      C, loc_0_2A63
2A5E FE F0         cp      #0xF0 ; '-'
2A60 DA 6E 2A       jp      C, loc_0_2A6E
2A63
loc_0_2A63:
2A63                ; CODE XREF: sub_0_2A2F+2C↑j
2A63 E6 0F         and     #0xF
2A65 3D            dec     a
2A66 C3 72 2A       jp      loc_0_2A72
2A69
2A69
loc_0_2A69:
2A69                ; CODE XREF: sub_0_2A2F+22↑j
2A69 3E FF         ld      a, #0xFF
2A6B C3 72 2A       jp      loc_0_2A72
2A6E
2A6E
loc_0_2A6E:
2A6E                ; CODE XREF: sub_0_2A2F+27↑j
2A6E                ; sub_0_2A2F+31↑j
2A6E E6 0F         and     #0xF
2A70 D6 09         sub     #9
2A72
loc_0_2A72:
2A72                ; CODE XREF: sub_0_2A2F+37↑j
2A72                ; sub_0_2A2F+3C↑j
2A72 4F            ld      c, a
2A73 7B            ld      a, e
2A74 E6 F8         and     #0xF8 ; 'o'
2A76 81            add     a, c
2A77 BB            cp      e
2A78 DA 7D 2A       jp      C, loc_0_2A7D
2A7B

```



```

2A7B      loc_0_2A7B:                                     ; CODE XREF: sub_0_2A2F+12↑j
2A7B AF      xor      a                                     ; sub_0_2A2F+19↑j ...
2A7C      ret
2A7C C9      ;
2A7D
2A7D
2A7D      loc_0_2A7D:                                     ; CODE XREF: sub_0_2A2F+49↑j
2A7D D6 04      sub      #4
2A7F DD 77 05      ld      5(ix), a
2A82 3E 01      ld      a, #1
2A84 C9      ret
2A84      ; End of function sub_0_2A2F
2A84
2A84
2A85      ; ██████████ S U B R O U T I N E ██████████
2A85
2A85      sub_0_2A85:                                     ; CODE XREF: 0000:19A1↑p
2A85 3A 15 62      ld      a, (mario_climbing)
2A88 A7      and      a                                     ; climbing?
2A89 C0      ret      NZ                                     ; yes, return
2A8A 3A 16 62      ld      a, (mario_jumping)
2A8D A7      and      a                                     ; jumping?
2A8E C0      ret      NZ                                     ; yes, return
2A8F 3A 98 63      ld      a, (mario_on_elevator)
2A92 FE 01      cp      #1                                     ; on elevator?
2A94 C8      ret      Z                                     ; yes, return
2A95 3A 03 62      ld      a, (mario_y)
2A98 D6 03      sub      #3
2A9A 67      ld      h, a
2A9B 3A 05 62      ld      a, (mario_x)
2A9E C6 0C      add      a, #0xC
2AA0 6F      ld      l, a
2AA1 E5      push     hl
2AA2 CD F0 2F      call    get_tilemap_addr_from_coords
2AA5 D1      pop      de
2AA6 7E      ld      a, (hl)
2AA7 FE B0      cp      #0xB0 ; '0'
2AA9 DA B4 2A      jp      C, loc_0_2AB4
2AAC E6 0F      and      #0xF
2AAE FE 08      cp      #8
2AB0 D2 B4 2A      jp      NC, loc_0_2AB4
2AB3 C9      ret
2AB4
2AB4
2AB4      loc_0_2AB4:                                     ; CODE XREF: sub_0_2A85+24↑j
2AB4 7A      ; sub_0_2A85+2B↑j ...
2AB4      ld      a, d
2AB5 E6 07      and      #7
2AB7 CA CD 2A      jp      Z, loc_0_2ACD
2ABA 01 20 00      ld      bc, #0x20 ; ' '
2ABD ED 42      sbc      hl, bc
2ABF 7E      ld      a, (hl)
2AC0 FE B0      cp      #0xB0 ; '0'
2AC2 DA CD 2A      jp      C, loc_0_2ACD
2AC5 E6 0F      and      #0xF
2AC7 FE 08      cp      #8
2AC9 D2 CD 2A      jp      NC, loc_0_2ACD
2ACC C9      ret
2ACD
2ACD
2ACD      loc_0_2ACD:                                     ; CODE XREF: sub_0_2A85+32↑j
2ACD 3E 01      ; sub_0_2A85+3D↑j ...
2ACD      ld      a, #1
2ACF 32 21 62      ld      (unk_0_6221), a
2AD2 C9      ret
2AD2      ; End of function sub_0_2A85
2AD2
2AD3      ; ██████████ S U B R O U T I N E ██████████
2AD3
2AD3      sub_0_2AD3:                                     ; CODE XREF: sub_0_25F2+C↑p
2AD3 3A 03 62      ld      a, (mario_y)
2AD6 47      ld      b, a
2AD7 3A 05 62      ld      a, (mario_x)
2ADA FE 50      cp      #0x50 ; 'P'
2ADC CA EA 2A      jp      Z, loc_0_2AEA
2ADF FE 78      cp      #0x78 ; 'x'
2AE1 CA F6 2A      jp      Z, loc_0_2AF6
2AE4 FE C8      cp      #0xC8 ; 'L'
2AE6 CA F0 2A      jp      Z, loc_0_2AF0
2AE9 C9      ret
2AEA
2AEA
2AEA      loc_0_2AEA:                                     ; CODE XREF: sub_0_2AD3+9↑j
2AEA 3A A3 63      ld      a, (unk_0_63A3)
2AED C3 02 2B      jp      loc_0_2B02
2AF0
2AF0
2AF0      loc_0_2AF0:                                     ; CODE XREF: sub_0_2AD3+13↑j
2AF0 3A A6 63      ld      a, (unk_0_63A6)
2AF3 C3 02 2B      jp      loc_0_2B02
2AF6
2AF6
2AF6      loc_0_2AF6:                                     ; CODE XREF: sub_0_2AD3+E↑j
2AF6      ld      a, b
2AF7 FE 80      cp      #0x80 ; 'Ç'
2AF9 3A A5 63      ld      a, (unk_0_63A5)
2AFC D2 02 2B      jp      NC, loc_0_2B02
2AFF 3A A4 63      ld      a, (unk_0_63A4)
2B02
2B02
2B02      loc_0_2B02:                                     ; CODE XREF: sub_0_2AD3+1A↑j
2B02 80      ; sub_0_2AD3+20↑j ...
2B02      add      a, b
2B03 32 03 62      ld      (mario_y), a
2B06 32 4C 69      ld      (soft_sprite_ram+0x4C), a
2B09 CD 1F 24      call    sub_0_241F
2B0C 21 03 62      ld      hl, #mario_y
2B0F 1D      dec      e
2B10 CA 18 2B      jp      Z, loc_0_2B18
2B13 15      dec      d
2B14 CA 1A 2B      jp      Z, loc_0_2B1A
2B17 C9      ret
2B18
2B18

```

```

2B18      loc_0_2B18:      dec      (hl)      ; CODE XREF: sub_0_2AD3+3D↑j
2B18 35
2B19 C9      ret
2B1A
2B1A
2B1A      loc_0_2B1A:      inc      (hl)      ; CODE XREF: sub_0_2AD3+41↑j
2B1A 34
2B1B C9      ret
2B1B      ; End of function sub_0_2AD3
2B1C
2B1C      ; ██████████ SUBROUTINE ██████████
2B1C
2B1C      sub_0_2B1C:      ld      ix, #mario_alive_flag      ; CODE XREF: sub_0_1AC3+142↑p
2B1C DD 21 00 62      call   sub_0_2B29
2B20 CD 29 2B      call   sub_0_29AF
2B23 CD AF 29      xor     a, a
2B26 AF      ld     b, a
2B27 47      ret
2B28 C9      ; End of function sub_0_2B1C
2B28
2B28      ; ██████████ SUBROUTINE ██████████
2B29
2B29      sub_0_2B29:      ld      a, (level_type)      ; CODE XREF: sub_0_2B1C+4↑p
2B29 3A 27 62      dec     a
2B2C 3D      jp     NZ, loc_0_2B53
2B2D C2 53 2B      ld     a, (mario_y)
2B30 3A 03 62      ld     h, a
2B33 67      ld     a, (mario_x)
2B34 3A 05 62      add    a, #7
2B37 C6 07      ld     l, a
2B39 6F      call   sub_0_2B9B
2B3A CD 9B 2B      and    a
2B3D A7      jp     Z, loc_0_2B51
2B3E CA 51 2B      ld     a, e
2B41 7B      sub    c
2B42 91      cp     #4
2B43 FE 04      jp     NC, loc_0_2B74
2B45 D2 74 2B      ld     a, c
2B48 79      sub    #7
2B49 D6 07      ld     (mario_x), a
2B4B 32 05 62      ld     a, #1
2B4E 3E 01      ld     b, a
2B50 47
2B51
2B51      loc_0_2B51:      pop     hl      ; CODE XREF: sub_0_2B29+15↑j
2B51 E1
2B52 C9      ret
2B53
2B53      loc_0_2B53:      ld      a, (mario_y)      ; CODE XREF: sub_0_2B29+4↑j
2B53 3A 03 62      sub    #3
2B56 D6 03      ld     h, a
2B58 67      ld     a, (mario_x)
2B59 3A 05 62      add    a, #7
2B5C C6 07      ld     l, a
2B5E 6F      call   sub_0_2B9B
2B5F CD 9B 2B      cp     #2
2B62 FE 02      jp     Z, loc_0_2B7A
2B64 CA 7A 2B      ld     a, d
2B67 7A      add    a, #7
2B68 C6 07      ld     h, a
2B6A 67      ld     l, e
2B6B 6B      call   sub_0_2B9B
2B6C CD 9B 2B      and    a
2B6F A7      ret     Z
2B70 C8      jp     loc_0_2B7A
2B71 C3 7A 2B
2B74
2B74      loc_0_2B74:      ld      a, #0      ; CODE XREF: sub_0_2B29+1C↑j
2B74      ld     b, #0
2B76 06 00      pop     hl
2B78 E1
2B79 C9      ret
2B7A
2B7A      loc_0_2B7A:      ld      a, (unk_0_6210)      ; CODE XREF: sub_0_2B29+3B↑j
2B7A 3A 10 62      and    a      ; sub_0_2B29+48↑j
2B7D A7      ld     a, (mario_y)
2B7E 3A 03 62      jp     Z, loc_0_2B8B
2B81 CA 8B 2B      or     #7
2B84 F6 07      sub    #4
2B86 D6 04      jp     loc_0_2B91
2B88 C3 91 2B
2B8B
2B8B      loc_0_2B8B:      sub     #8      ; CODE XREF: sub_0_2B29+58↑j
2B8B      or     #7
2B8D F6 07      add    a, #4
2B8F C6 04
2B91
2B91      loc_0_2B91:      ld      (mario_y), a      ; CODE XREF: sub_0_2B29+5F↑j
2B91 32 03 62      ld     (soft_sprite_ram+0x4C), a
2B94 32 4C 69      ld     a, #1
2B97 3E 01      pop     hl
2B99 E1
2B9A C9      ret
2B9A      ; End of function sub_0_2B29
2B9A
2B9A      ; ██████████ SUBROUTINE ██████████
2B9B
2B9B      sub_0_2B9B:      push    hl      ; CODE XREF: sub_0_2B29+11↑p
2B9B      call   get_tilemap_addr_from_coords      ; sub_0_2B29+36↑p ...
2B9B E5
2B9B      push    hl
2B9C CD F0 2F      call   de
2B9F D1      pop     a, (hl)
2BA0 7E      ld     #0xB0 ; 'B'
2BA1 FE B0      cp

```

```

2BA3 DA D9 2B      jp      C, loc_0_2BD9
2BA6 E6 0F      and     #0xF
2BA8 FE 08      cp      #8
2BAA D2 D9 2B      jp      NC, loc_0_2BD9
2BAD 7E      ld      a, (hl)
2BAE FE C0      cp      #0xC0 ; 'L'
2BB0 CA D9 2B      jp      Z, loc_0_2BD9
2BB3 DA DC 2B      jp      C, loc_0_2BDC
2BB6 FE D0      cp      #0xD0 ; 'd'
2BB8 DA CB 2B      jp      C, loc_0_2BCB
2BBB FE E0      cp      #0xE0 ; 'O'
2BBD DA C5 2B      jp      C, loc_0_2BC5
2BC0 FE F0      cp      #0xF0 ; '-'
2BC2 DA CB 2B      jp      C, loc_0_2BCB
2BC5
2BC5      loc_0_2BC5:                                ; CODE XREF: sub_0_2B9B+22↑j
2BC5 E6 0F      and     #0xF
2BC7 3D      dec     a
2BC8 C3 CF 2B      jp      loc_0_2BCF
2BCB
2BCB      ; -----
2BCB      loc_0_2BCB:                                ; CODE XREF: sub_0_2B9B+1D↑j
2BCB E6 0F      and     #0xF                                ; sub_0_2B9B+27↑j
2BCB      sub     #9
2BCD D6 09
2BCF      loc_0_2BCF:                                ; CODE XREF: sub_0_2B9B+2D↑j
2BCF 4F      ld      C, a
2BD0 7B      ld      a, e
2BD1 E6 F8      and     #0xF8 ; 'o'
2BD3 81      add     a, C
2BD4 4F      ld      C, a
2BD5 BB      cp      e
2BD6 DA E1 2B      jp      C, loc_0_2BE1
2BD9
2BD9      loc_0_2BD9:                                ; CODE XREF: sub_0_2B9B+8↑j
2BD9 AF      xor     a, a                                ; sub_0_2B9B+F↑j ...
2BD9      ld      b, a
2BDA 47
2BDB C9
2BDC
2BDC      ; -----
2BDC      loc_0_2BDC:                                ; CODE XREF: sub_0_2B9B+18↑j
2BDC 7B      ld      a, e
2BDD E6 F8      and     #0xF8 ; 'o'
2BDF 3D      dec     a
2BE0 4F      ld      C, a
2BE1
2BE1      loc_0_2BE1:                                ; CODE XREF: sub_0_2B9B+3B↑j
2BE1 3A 0C 62      ld      a, (mario_y_before_jump)
2BE4 DD 96 05      sub     5(ix)
2BE7 83      add     a, e
2BE8 B9      cp      C
2BE9 CA EF 2B      jp      Z, loc_0_2BEF
2BEC D2 F8 2B      jp      NC, loc_0_2BF8
2BEF
2BEF      loc_0_2BEF:                                ; CODE XREF: sub_0_2B9B+4E↑j
2BEF 79      ld      a, C
2BF0 D6 07      sub     #7
2BF2 32 05 62      ld      (mario_x), a
2BF5 C3 FD 2B      jp      loc_0_2BFD
2BF8
2BF8      ; -----
2BF8      loc_0_2BF8:                                ; CODE XREF: sub_0_2B9B+51↑j
2BF8 3E 02      ld      a, #2
2BFA 06 00      ld      b, #0
2BFC C9
2BFD
2BFD      ; -----
2BFD      loc_0_2BFD:                                ; CODE XREF: sub_0_2B9B+5A↑j
2BFD 3E 01      ld      a, #1
2BFF 47      ld      b, a
2C00 E1      pop     hl
2C01 E1      pop     hl
2C02 C9      ret
2C02
2C02      ; End of function sub_0_2B9B
2C03
2C03      ; ██████████ S U B R O U T I N E ██████████
2C03
2C03      sub_0_2C03:                                ; CODE XREF: 0000:1989↑p
2C03 3E 01      ld      a, #1
2C05 F7      rst     0x30                                ; return if level bit not set
2C06 D7      rst     0x10                                ; return if mario not alive
2C07 3A 93 63      ld      a, (barrel_deployment)
2C0A 0F      rrca
2C0B D8      ret     C
2C0C 3A B1 62      ld      a, (unk_0_62B1)
2C0F A7      and     a
2C10 C8      ret     Z
2C11 4F      ld      C, a
2C12 3A B0 62      ld      a, (bonus_timer_init_value)
2C15 D6 02      sub     #2
2C17 B9      cp      C
2C18 DA 7B 2C      jp      C, loc_0_2C7B
2C1B 3A 82 63      ld      a, (unk_0_6382)
2C1E CB 4F      bit     1, a
2C20 C2 86 2C      jp      NZ, loc_0_2C86
2C23 3A 80 63      ld      a, (unk_0_6380)
2C26 47      ld      b, a
2C27 3A 1A 60      ld      a, (gen_purpose_timer)
2C2A E6 1F      and     #0x1F
2C2C
2C2C      loc_0_2C2C:                                ; CODE XREF: sub_0_2C03+2D↑j
2C2C B8      cp      b
2C2D CA 33 2C      jp      Z, loc_0_2C33
2C30 10 FA      djnz   loc_0_2C2C
2C32 C9      ret
2C33
2C33      ; -----
2C33      loc_0_2C33:                                ; CODE XREF: sub_0_2C03+2A↑j
2C33 3A B0 62      ld      a, (bonus_timer_init_value)
2C36 CB 3F      srl     a
2C38 B9      cp      C
2C39 DA 41 2C      jp      C, loc_0_2C41

```

```

2C3C 3A 19 60      ld      a, (random_no+1)
2C3F 0F            rrca
2C40 D0            ret      NC
2C41
2C41             loc_0_2C41:                                ; CODE XREF: sub_0_2C03+36↑j
2C41 CD 57 00      call    rand
2C44 E6 0F          and     #0xF
2C46 C2 86 2C      jp      NZ, loc_0_2C86
2C49
2C49             loc_0_2C49:                                ; CODE XREF: sub_0_2C03+7B↑j
2C49 3E 01          ld      a, #1
2C4B
2C4B             loc_0_2C4B:                                ; CODE XREF: sub_0_2C03+80↑j
2C4B 32 82 63      ld      (unk_0_6382), a
2C4E 3C            inc     a
2C4F
2C4F             loc_0_2C4F:                                ; CODE XREF: sub_0_2C03+89↑j
2C4F 32 8F 63      ld      (unk_0_638F), a
2C52 3E 01          ld      a, #1
2C54 32 92 63      ld      (unk_0_6392), a
2C57 3A B2 62      ld      a, (unk_0_62B2)
2C5A B9            cp      c
2C5B C0            ret     NZ
2C5C D6 08          sub     #8
2C5E 32 B2 62      ld      (unk_0_62B2), a
2C61 11 20 00      ld      de, #0x20 ; ' '
2C64 21 00 64      ld      hl, #unk_0_6400
2C67 06 05          ld      b, #5
2C69             loc_0_2C69:                                ; CODE XREF: sub_0_2C03+6C↑j
2C69 7E            ld      a, (hl)
2C6A A7            and     a
2C6B CA 72 2C      jp      Z, loc_0_2C72
2C6E 19            add     hl, de
2C6F 10 F8          djnz   loc_0_2C69
2C71 C9            ret
2C72
2C72             ; -----
2C72             loc_0_2C72:                                ; CODE XREF: sub_0_2C03+68↑j
2C72 3A 82 63      ld      a, (unk_0_6382)
2C75 F6 80          or      #0x80 ; 'Ç'
2C77 32 82 63      ld      (unk_0_6382), a
2C7A C9            ret
2C7B
2C7B             loc_0_2C7B:                                ; CODE XREF: sub_0_2C03+15↑j
2C7B C6 02          add     a, #2
2C7D B9            cp      c
2C7E CA 49 2C      jp      Z, loc_0_2C49
2C81 3E 02          ld      a, #2
2C83 C3 4B 2C      jp      loc_0_2C4B
2C86
2C86             loc_0_2C86:                                ; CODE XREF: sub_0_2C03+1D↑j
2C86 AF            ; sub_0_2C03+43↑j
2C86             xor     a
2C87 32 82 63      ld      (unk_0_6382), a
2C8A 3E 03          ld      a, #3
2C8C C3 4F 2C      jp      loc_0_2C4F
2C8C             ; End of function sub_0_2C03
2C8F
2C8F             ; ██████████ SUBROUTINE ██████████
2C8F
2C8F             sub_0_2C8F:                                ; CODE XREF: 0000:1986↑p
2C8F 3E 01          ld      a, #1
2C91 F7            rst     0x30
2C92 D7            rst     0x10
2C93 3A 93 63      ld      a, (barrel_deployment)
2C96 0F            rrca
2C97 DA 15 2D      jp      C, loc_0_2D15
2C9A 3A 92 63      ld      a, (unk_0_6392)
2C9D 0F            rrca
2C9E D0            ret     NC
2C9F DD 21 00 67   ld      ix, #unk_0_6700
2CA3 11 20 00      ld      de, #0x20 ; ' '
2CA6 06 0A          ld      b, #0xA
2CA8
2CA8             loc_0_2CA8:                                ; CODE XREF: sub_0_2C8F+26↑j
2CA8 DD 7E 00      ld      a, 0(ix)
2CAB 0F            rrca
2CAC DA B3 2C      jp      C, loc_0_2CB3
2CAF 0F            rrca
2CB0 D2 B8 2C      jp      NC, loc_0_2CB8
2CB3
2CB3             loc_0_2CB3:                                ; CODE XREF: sub_0_2C8F+1D↑j
2CB3 DD 19          add     ix, de
2CB5 10 F1          djnz   loc_0_2CA8
2CB7 C9            ret
2CB8
2CB8             loc_0_2CB8:                                ; CODE XREF: sub_0_2C8F+21↑j
2CB8 DD 22 AA 62   ld      (unk_0_62AA), ix
2CBC DD 36 00 02   ld      0(ix), #2
2CC0 16 00          ld      d, #0
2CC2 3E 0A          ld      a, #0xA
2CC4 90            sub     b
2CC5 87            add     a, a
2CC6 87            add     a, a
2CC7 5F            ld      e, a
2CC8 21 80 69      ld      hl, #soft_sprite_ram+0x80
2CCB 19            add     hl, de
2CCC 22 AC 62      ld      (unk_0_62AC), hl
2CCF 3E 01          ld      a, #1
2CD1 32 93 63      ld      (barrel_deployment), a
2CD4 11 01 05      ld      de, #0x501
2CD7 CD 9F 30      call   queue_fg_vector_fn
2CDA 21 B1 62      ld      hl, #unk_0_62B1
2CDD 35            dec     (hl)
2CDE C2 E6 2C      jp      NZ, loc_0_2CE6
2CE1 3E 01          ld      a, #1
2CE3 32 86 63      ld      (unk_0_6386), a
2CE6
2CE6             loc_0_2CE6:                                ; CODE XREF: sub_0_2C8F+4F↑j
2CE6 7E            ld      a, (hl)

```

```

2CE7 FE 04      cp      #4
2CE9 D2 F6 2C   jp      NC, loc_0_2CF6
2CEC 21 A8 69   ld      hl, #soft_sprite_ram+0xA8
2CEF 87         add     a, a
2CF0 87         add     a, a
2CF1 5F         ld      e, a
2CF2 16 00      ld      d, #0
2CF4 19         add     hl, de
2CF5 72         ld      (hl), d
2CF6
2CF6          loc_0_2CF6:                                ; CODE XREF: sub_0_2C8F+5A|j
2CF6 DD 36 07 15 ld      7(ix), #0x15                                ; sideways barrel sprite tile
2CFA DD 36 08 0B ld      8(ix), #0xB
2CFE DD 36 15 00 ld      0x15(ix), #0
2D02 3A 82 63   ld      a, (unk_0_6382)
2D05 07         rlc     a
2D06 D2 15 2D   jp      NC, loc_0_2D15
2D09 DD 36 07 19 ld      7(ix), #0x19                                ; sideways blue barrel sprite tile
2D0D DD 36 08 0C ld      8(ix), #0xC                                ; set blue palette for barrel
2D11 DD 36 15 01 ld      0x15(ix), #1
2D15
2D15          loc_0_2D15:                                ; CODE XREF: sub_0_2C8F+8|j
2D15 21 AF 62   ld      hl, #byte_0_62AF                                ; sub_0_2C8F+77|j
2D18 35         dec     (hl)
2D19 C0         ret     NZ
2D1A 36 18      ld      (hl), #0x18
2D1C 3A 8F 63   ld      a, (unk_0_638F)
2D1F A7         and     a
2D20 CA 51 2D   jp      Z, loc_0_2D51
2D23 4F         ld      c, a
2D24 21 32 39   ld      hl, #dk_throw_barrel_spr
2D27 3A 82 63   ld      a, (unk_0_6382)
2D2A 0F         rrca
2D2B DA 2F 2D   jp      C, loc_0_2D2F
2D2E 0D         dec     c
2D2F
2D2F          loc_0_2D2F:                                ; CODE XREF: sub_0_2C8F+9C|j
2D2F 79         ld      a, c
2D30 87         add     a, a
2D31 87         add     a, a
2D32 87         add     a, a
2D33 4F         ld      c, a
2D34 87         add     a, a
2D35 87         add     a, a
2D36 81         add     a, c
2D37 5F         ld      e, a
2D38 16 00      ld      d, #0
2D3A 19         add     hl, de
2D3B CD 4E 00   call    copy_sprites_2_11_data
2D3E 21 8F 63   ld      hl, #unk_0_638F
2D41 35         dec     (hl)
2D42 C2 51 2D   jp      NZ, loc_0_2D51
2D45 3E 01      ld      a, #1
2D47 32 AF 62   ld      (byte_0_62AF), a
2D4A 3A 82 63   ld      a, (unk_0_6382)
2D4D 0F         rrca
2D4E DA 83 2D   jp      C, loc_0_2D83
2D51
2D51          loc_0_2D51:                                ; CODE XREF: sub_0_2C8F+91|j
2D51 2A A8 62   ld      hl, (unk_0_62A8)                                ; sub_0_2C8F+B3|j
2D54
2D54          loc_0_2D54:                                ; CODE XREF: sub_0_2C8F+FA|j
2D54 7E         ld      a, (hl)
2D55 DD 2A AA 62 ld      ix, (unk_0_62AA)
2D59 ED 5B AC 62 ld      de, (unk_0_62AC)
2D5D FE 7F      cp      #0x7F ; ' '
2D5F CA 8C 2D   jp      Z, loc_0_2D8C
2D62 4F         ld      c, a
2D63 E6 7F      and     #0x7F ; ' '
2D65 12         ld      (de), a                                ; sprite data X coord
2D66 DD 7E 07   ld      a, 7(ix)                                ; sprite tile #
2D69 CB 79      bit     7, c
2D6B CA 70 2D   jp      Z, loc_0_2D70
2D6E EE 03      xor     #3
2D70
2D70          loc_0_2D70:                                ; CODE XREF: sub_0_2C8F+DC|j
2D70 13         inc     de
2D71 12         ld      (de), a                                ; sprite tile # (barrel)
2D72 DD 77 07   ld      7(ix), a                                ; sprite tile #
2D75 DD 7E 08   ld      a, 8(ix)
2D78 13         inc     de
2D79 12         ld      (de), a
2D7A 23         inc     hl
2D7B 7E         ld      a, (hl)
2D7C 13         inc     de
2D7D 12         ld      (de), a
2D7E 23         inc     hl
2D7F 22 A8 62   ld      (unk_0_62A8), hl
2D82 C9         ret
2D83
2D83          loc_0_2D83:                                ; CODE XREF: sub_0_2C8F+BF|j
2D83 21 CC 39      ld      hl, #barrel_falling_data
2D86 22 A8 62   ld      (unk_0_62A8), hl
2D89 C3 54 2D   jp      loc_0_2D54
2D8C
2D8C          loc_0_2D8C:                                ; CODE XREF: sub_0_2C8F+D0|j
2D8C 21 C3 39      ld      hl, #barell_rolling_data
2D8F 22 A8 62   ld      (unk_0_62A8), hl
2D92 DD 36 01 01 ld      1(ix), #1
2D96 3A 82 63   ld      a, (unk_0_6382)
2D99 0F         rrca
2D9A DA A5 2D   jp      C, loc_0_2DA5
2D9D DD 36 01 00 ld      1(ix), #0
2DA1 DD 36 02 02 ld      2(ix), #2
2DA5
2DA5          loc_0_2DA5:                                ; CODE XREF: sub_0_2C8F+10B|j
2DA5 DD 36 00 01   ld      0(ix), #1
2DA9 DD 36 0F 01   ld      0xF(ix), #1
2DAD AF         xor     a
2DAE DD 77 10      ld      0x10(ix), a
2DB1 DD 77 11      ld      0x11(ix), a
2DB4 DD 77 12      ld      0x12(ix), a

```

```

2DB7 DD 77 13      ld      0x13(ix), a
2DBA DD 77 14      ld      0x14(ix), a
2DBD 32 93 63      ld      (barrel_deployment), a
2DC0 32 92 63      ld      (unk_0_6392), a
2DC3 1A           ld      a, (de)
2DC4 DD 77 03      ld      3(ix), a
2DC7 13           inc     de
2DC8 13           inc     de
2DC9 13           inc     de
2DCA 1A           ld      a, (de)
2DCB DD 77 05      ld      5(ix), a
2DCE 21 5C 38      ld      hl, #dk_normal_spr
2DD1 CD 4E 00      call    copy_sprites_2_11_data
2DD4 21 0B 69      ld      hl, #soft_sprite_ram+0xB
2DD7 0E FC         ld      c, #0xFC ; '3'
2DD9 FF           rst      0x38
2DDA C9           ret
; End of function sub_0_2C8F

2DDA
2DDB
2DDB ; ██████████ S U B R O U T I N E ██████████
2DDB
2DDB
2DDB sub_0_2DDB: ; CODE XREF: 0000:1995|p
2DDB 3E 0A         ld      a, #0xA
2DDD F7           rst      0x30
2DDE D7           rst      0x10
2DDF 3A 80 63      ld      a, (unk_0_6380)
2DE2 3C           inc     a
2DE3 A7           and     a
2DE4 1F           rra
2DE5 47           ld      b, a
2DE6 3A 27 62      ld      a, (level_type)
2DE9 FE 02         cp      #2
2DEB 20 01         jr      NZ, loc_0_2DEE
2DED 04           inc     b
2DEE
2DEE loc_0_2DEE: ; CODE XREF: sub_0_2DDB+10|j
2DEE 3E FE         ld      a, #0xFE ; '■'
2DF0 37           scf
2DF1
2DF1 loc_0_2DF1: ; CODE XREF: sub_0_2DDB+18|j
2DF1 1F           rra
2DF2 A7           and     a
2DF3 10 FC         djnz   loc_0_2DF1
2DF5 47           ld      b, a
2DF6 3A 1A 60      ld      a, (gen_purpose_timer)
2DF9 A0           and     b
2DFA C0           ret     NZ
2DFB 3E 01         ld      a, #1
2DFD 32 A0 63      ld      (unk_0_63A0), a
2E00 32 9A 63      ld      (unk_0_639A), a
2E03 C9           ret
; End of function sub_0_2DDB

2E03
2E04
2E04 ; ██████████ S U B R O U T I N E ██████████
2E04
2E04
2E04 sub_0_2E04: ; CODE XREF: 0000:198F|p
2E04 3E 04         ld      a, #4
2E06 F7           rst      0x30
2E07 D7           rst      0x10
2E08 DD 21 00 65    ld      ix, #unk_0_6500
2E0C FD 21 80 69    ld      iy, #soft_sprite_ram+0x80
2E10 06 0A         ld      b, #0xA
2E12
2E12 loc_0_2E12: ; CODE XREF: sub_0_2E04+7D|j
2E12 DD 7E 00         ld      a, 0(ix)
2E15 0F           rrca
2E16 D2 A7 2E       jp      NC, loc_0_2EA7
2E19 3A 1A 60      ld      a, (gen_purpose_timer)
2E1C E6 0F         and     #0xF
2E1E C2 29 2E       jp      NZ, loc_0_2E29
2E21 FD 7E 01      ld      a, 1(iy)
2E24 EE 07         xor     #7
2E26 FD 77 01      ld      1(iy), a
2E29
2E29 loc_0_2E29: ; CODE XREF: sub_0_2E04+1A|j
2E29 DD 7E 0D         ld      a, 0xD(ix)
2E2C FE 04         cp      #4
2E2E CA 84 2E       jp      Z, loc_0_2E84
2E31 DD 34 03       inc     3(ix)
2E34 DD 34 03       inc     3(ix)
2E37 DD 6E 0E       ld      l, 0xE(ix)
2E3A DD 66 0F       ld      h, 0xF(ix)
2E3D 7E           ld      a, (hl)
2E3E 4F           ld      c, a
2E3F FE 7F         cp      #0x7F ; ' '
2E41 CA 9C 2E       jp      Z, loc_0_2E9C
2E44 23           inc     hl
2E45 DD 86 05       add     a, 5(ix)
2E48 DD 77 05       ld      5(ix), a
2E4B
2E4B loc_0_2E4B: ; CODE XREF: sub_0_2E04+A0|j
2E4B DD 75 0E         ld      0xE(ix), 1
2E4E DD 74 0F       ld      0xF(ix), h
2E51 DD 7E 03       ld      a, 3(ix)
2E54 FE B7         cp      #0xB7 ; 'À'
2E56 DA 6C 2E       jp      C, loc_0_2E6C
2E59 79           ld      a, c
2E5A FE 7F         cp      #0x7F ; ' '
2E5C C2 6C 2E       jp      NZ, loc_0_2E6C
2E5F DD 36 0D 04    ld      0xD(ix), #4
2E63 AF           xor     a
2E64 32 83 60      ld      (digital_snd_tmr_coin_spring), a
2E67 3E 03         ld      a, #3
2E69 32 84 60      ld      (digital_snd_tmr_kong_fall), a
2E6C
2E6C loc_0_2E6C: ; CODE XREF: sub_0_2E04+52|j
2E6C DD 7E 03         ld      a, 3(ix)
2E6F FD 77 00         ld      0(iy), a
2E72 DD 7E 05         ld      a, 5(ix)
2E75 FD 77 03         ld      3(iy), a
2E78

```

```

2E78      loc_0_2E78:                                ; CODE XREF: sub_0_2E04+A7↑j
2E78 11 10 00                                         ; sub_0_2E04+CD↑j
2E78                                         ; 16 bytes/sprite
2E7B DD 19      ld     de, #0x10                     ; next spring data
2E7D 1E 04      ld     e, #4
2E7F FD 19      add    iy, de                         ; next sprite data
2E81 10 8F      djnz   loc_0_2E12
2E83 C9         ret
2E84
2E84      loc_0_2E84:                                ; CODE XREF: sub_0_2E04+2A↑j
2E84 3E 03      ld     a, #3
2E86 DD 86 05   add    a, 5(ix)
2E89 DD 77 05   ld     5(ix), a
2E8C FE F8      cp     #0xF8 ; 'o'
2E8E DA 6C 2E   jp     C, loc_0_2E6C
2E91 DD 36 03 00 ld     3(ix), #0
2E95 DD 36 00 00 ld     0(ix), #0
2E99 C3 6C 2E   jp     loc_0_2E6C
2E9C
2E9C      loc_0_2E9C:                                ; CODE XREF: sub_0_2E04+3D↑j
2E9C 21 AA 39   ld     hl, #bouncing_spring_data
2E9F 3E 03      ld     a, #3
2EA1 32 83 60   ld     (digital_snd_tmr_coin_spring), a ; tmr=3
2EA4 C3 4B 2E   jp     loc_0_2E4B
2EA7
2EA7      loc_0_2EA7:                                ; CODE XREF: sub_0_2E04+12↑j
2EA7 3A 96 63   ld     a, (unk_0_6396)
2EAA 0F      rrca
2EAB D2 78 2E   jp     NC, loc_0_2E78
2EAE AF      xor     a
2EAF 32 96 63   ld     (unk_0_6396), a
2EB2 DD 36 05 50 ld     5(ix), #0x50 ; 'P'
2EB6 DD 36 0D 01 ld     0xD(ix), #1
2EBA CD 57 00   call   rand
2EBD E6 0F      and    #0xF
2EBF C6 F8      add    a, #0xF8 ; 'o'
2EC1 DD 77 03   ld     3(ix), a
2EC4 DD 36 00 01 ld     0(ix), #1
2EC8 21 AA 39   ld     hl, #bouncing_spring_data
2ECB DD 75 0E   ld     0xE(ix), 1
2ECE DD 74 0F   ld     0xF(ix), h
2ED1 C3 78 2E   jp     loc_0_2E78 ; end of spring routine
2ED1 ; End of function sub_0_2E04
2ED1
2ED4      ; SUBROUTINE
2ED4
2ED4      sub_0_2ED4:                                ; CODE XREF: 0000:1998↑p
2ED4 3E 0B      ld     a, #0xB
2ED6 F7      rst     0x30 ; return if level bit not set
2ED7 D7      rst     0x10 ; return if mario not alive
2ED8 11 18 6A   ld     de, #soft_sprite_ram+0x118 ; hammers in sprite ram
2EDB DD 21 80 66 ld     ix, #unk_0_6680 ; hammer character data
2EDF DD 7E 01   ld     a, 1(ix)
2EE2 0F      rrca
2EE3 DA ED 2E   jp     C, loc_0_2EED
2EE6 11 1C 6A   ld     de, #soft_sprite_ram+0x11C
2EE9 DD 21 90 66 ld     ix, #unk_0_6690
2EED
2EED      loc_0_2EED:                                ; CODE XREF: sub_0_2ED4+F↑j
2EED DD 36 0E 00   ld     0xE(ix), #0
2EF1 DD 36 0F F0 ld     0xF(ix), #0xF0 ; '-'
2EF5 3A 17 62   ld     a, (unk_0_6217)
2EF8 0F      rrca
2EF9 D2 97 2F   jp     NC, loc_0_2F97
2EFC AF      xor     a
2EFD 32 18 62   ld     (unk_0_6218), a
2F00 21 89 60   ld     hl, #bg_music
2F03 36 04      ld     (hl), #4
2F05 DD 36 09 06 ld     9(ix), #6
2F09 DD 36 0A 03 ld     0xA(ix), #3
2F0D 06 1E      ld     b, #0x1E
2F0F 3A 07 62   ld     a, (mario_flipy_tile)
2F12 CB 27      sla     a
2F14 D2 1B 2F   jp     NC, loc_0_2F1B
2F17 F6 80      or     #0x80 ; 'Ç'
2F19 CB F8      set     7, b
2F1B
2F1B      loc_0_2F1B:                                ; CODE XREF: sub_0_2ED4+40↑j
2F1B F6 08      or     #8
2F1D 4F      ld     c, a
2F1E 3A 94 63   ld     a, (unk_0_6394)
2F21 CB 5F      bit     3, a
2F23 CA 43 2F   jp     Z, loc_0_2F43
2F26 CB C0      set     0, b
2F28 CB C1      set     0, c
2F2A DD 36 09 05 ld     9(ix), #5
2F2E DD 36 0A 06 ld     0xA(ix), #6
2F32 DD 36 0F 00 ld     0xF(ix), #0
2F36 DD 36 0E F0 ld     0xE(ix), #0xF0 ; '-'
2F3A CB 79      bit     7, c
2F3C CA 43 2F   jp     Z, loc_0_2F43
2F3F DD 36 0E 10   ld     0xE(ix), #0x10
2F43
2F43      loc_0_2F43:                                ; CODE XREF: sub_0_2ED4+4F↑j
2F43 79      ; sub_0_2ED4+68↑j
2F43
2F43      ld     a, c
2F44 32 4D 69   ld     (soft_sprite_ram+0x4D), a
2F47 0E 07      ld     c, #7
2F49 21 94 63   ld     hl, #unk_0_6394
2F4C 34      inc    (hl)
2F4D C2 B7 2F   jp     NZ, loc_0_2FB7
2F50 21 95 63   ld     hl, #unk_0_6395
2F53 34      inc    (hl)
2F54 7E      ld     a, (hl)
2F55 FE 02      cp     #2
2F57 C2 BE 2F   jp     NZ, loc_0_2FBE
2F5A AF      xor     a
2F5B 32 95 63   ld     (unk_0_6395), a
2F5E 32 17 62   ld     (unk_0_6217), a
2F61 DD 77 01   ld     1(ix), a
2F64 3A 03 62   ld     a, (mario_y)

```

```

2F67 ED 44          neg
2F69 DD 77 0E      ld      0xE(ix), a
2F6C 3A 07 62      ld      a, (mario_flipy_tile)
2F6F 32 4D 69      ld      (soft_sprite_ram+0x4D), a
2F72 DD 36 00 00   ld      0(ix), #0
2F76 3A 89 63      ld      a, (unk_0_6389)
2F79 32 89 60      ld      (bg_music), a
2F7C
2F7C          loc_0_2F7C:
2F7C EB
2F7C          ex      de, hl
2F7D 3A 03 62      ld      a, (mario_y)
2F80 DD 86 0E      add      a, 0xE(ix)
2F83 77          ld      (hl), a
2F84 DD 77 03      ld      3(ix), a
2F87 23          inc     hl
2F88 70          ld      (hl), b
2F89 23          inc     hl
2F8A 71          ld      (hl), c
2F8B 23          inc     hl
2F8C 3A 05 62      ld      a, (mario_x)
2F8F DD 86 0F      add      a, 0xF(ix)
2F92 77          ld      (hl), a
2F93 DD 77 05      ld      5(ix), a
2F96 C9          ret
;
2F97
2F97          loc_0_2F97:
2F97          ld      a, (unk_0_6218)
2F9A 0F          rrca
2F9B D0          ret     NC
2F9C DD 36 09 06   ld      9(ix), #6
2FA0 DD 36 0A 03   ld      0xA(ix), #3
2FA4 3A 07 62      ld      a, (mario_flipy_tile)
2FA7 07          rlca
2FA8 3E 3C          ld      a, #0x3C ; '<'
2FAA 1F          rra
2FAB 47          ld      b, a
2FAC 0E 07          ld      c, #7
2FAE 3A 89 60      ld      a, (bg_music)
2FB1 32 89 63      ld      (unk_0_6389), a
2FB4 C3 7C 2F      jp      loc_0_2F7C
;
2FB7
2FB7          loc_0_2FB7:
2FB7          ld      a, (unk_0_6395)
2FBA A7          and     a
2FBB CA 7C 2F      jp      Z, loc_0_2F7C
;
2FBE
2FBE          loc_0_2FBE:
2FBE          ld      a, (gen_purpose_timer)
2FC1 CB 5F          bit     3, a
2FC3 CA 7C 2F      jp      Z, loc_0_2F7C
2FC6 0E 01          ld      c, #1
2FC8 C3 7C 2F      jp      loc_0_2F7C
; End of function sub_0_2ED4
;
; SUBROUTINE
;
sub_0_2FCB:
; CODE XREF: 0000:19BF[p]
2FCB 3E 0E          ld      a, #0xE
2FCD F7          rst     0x30
; return if level bit not set
2FCE 21 B4 62      ld      hl, #unk_0_62B4
2FD1 35          dec     (hl)
2FD2 C0          ret     NZ
2FD3 3E 03          ld      a, #3
2FD5 32 B9 62      ld      (unk_0_62B9), a
2FD8 32 96 63      ld      (unk_0_6396), a
2FDB 11 01 05      ld      de, #0x501
2FDE CD 9F 30      call   queue_fg_vector_fn
2FE1 3A B3 62      ld      a, (unk_0_62B3)
2FE4 77          ld      (hl), a
2FE5 21 B1 62      ld      hl, #unk_0_62B1
2FE8 35          dec     (hl)
2FE9 C0          ret     NZ
2FEA 3E 01          ld      a, #1
2FEC 32 86 63      ld      (unk_0_6386), a
2FEF C9          ret
; End of function sub_0_2FCB
;
; SUBROUTINE
;
get_tilemap_addr_from_coords:
; CODE XREF: draw_level_background+10[p]
; draw_level_background+3D[p] ...
; Y pos in bits [7:3]
2FF0          ld      a, 1
2FF1 0F          rrca
2FF2 0F          rrca
2FF3 0F          rrca
2FF4 E6 1F          and     #0x1F
; shift to [4:0]
2FF6 6F          ld      l, a
; store as LSB of screen address
2FF7 7C          ld      a, h
2FF8 2F          cpl
; X pos in bits [7:3]
2FF9 E6 F8          and     #0xF8 ; 'o'
; mirror
2FFB 5F          ld      e, a
2FFC AF          xor     a
2FFD 67          ld      h, a
2FFE CB 13          rl      e
3000 17          rla
3001 CB 13          rl      e
3003 17          rla
; A=Xpos bits [7:6], E=[5:3]
3004 C6 74          add     a, #0x74 ; 't'
; add start of VRAM
3006 57          ld      d, a
; store
3007 19          add     hl, de
; HL = screen address
3008 C9          ret
; End of function get_tilemap_addr_from_coords
;
; SUBROUTINE
;
animate_mario_or_barrel_sprite:
; CODE XREF: 0000:18DF[p]
; sub_0_1AC3+1DB[p] ...
3009
3009 57

```



```

3009          ld      d, a          ; sprite type
300A 0F          rrca
300B DA 22 30    jp      C, loc_0_3022
300E 0E 93      ld      c, #0x93 ; '6'          ; sequence 3,0,1,2
3010 0F          rrca
3011 0F          rrca
3012 D2 17 30    jp      NC, loc_0_3017
3015 0E 6C      ld      c, #0x6C ; 'l'          ; sequence 0,3,2,1
3017
3017          loc_0_3017:          ; CODE XREF: animate_mario_or_barrel_sprite+9↑j
3017 07          rlca
3018 DA 31 30    jp      C, loc_0_3031
301B 79          ld      a, c
301C E6 F0      and     #0xF0 ; '-'
301E 4F          ld      c, a
301F C3 31 30    jp      loc_0_3031
3022
3022          loc_0_3022:          ; CODE XREF: animate_mario_or_barrel_sprite+2↑j
3022 0E B4      ld      c, #0xB4 ; '|'          ; sequence 0,1,3,2 (mario left)
3024 0F          rrca
3025 0F          rrca          ; moving sprite left?
3026 D2 2B 30    jp      NC, loc_0_302B
3029 0E 1E      ld      c, #0x1E          ; yes, skip
3029          ; sequence 2,3,1,0 (mario right)
302B
302B          loc_0_302B:          ; CODE XREF: animate_mario_or_barrel_sprite+1D↑j
302B CB 50      bit      2, b          ; animation cell #
302D CA 31 30    jp      Z, loc_0_3031          ; not special case (not 4)
3030 05          dec     b          ; 4->3
3031
3031          loc_0_3031:          ; CODE XREF: animate_mario_or_barrel_sprite+F↑j
3031 79          ; animate_mario_or_barrel_sprite+16↑j ...
3031          ld      a, c
3032 0F          rrca
3033 0F          rrca
3034 4F          ld      c, a
3035 E6 03      and     #3          ; get animation cell in sequence
3037 B8          cp      b          ; equals current cell?
3038 C2 31 30    jp      NZ, loc_0_3031          ; no, loop to find current cell
303B 79          ld      a, c
303C 0F          rrca
303D 0F          rrca
303E E6 03      and     #3          ; get next animation cell #
3040 FE 03      cp      #3          ; special case?
3042 C0          ret     NZ          ; no, return
3043 CB 92      res     2, d
3045 15          dec     d
3046 C0          ret     NZ
3047 3E 04      ld      a, #4          ; special case animation cell
3049 C9          ret
3049          ; End of function animate_mario_or_barrel_sprite
3049
304A          ; ██████████ SUBROUTINE ██████████
304A
304A          wipe_ladder_as_kong_climbs:          ; CODE XREF: display_1UP+9D↑p
304A          ; 0000:0B38↑p
304A          ; column offset
304A 11 E0 FF      ld      de, #0xFFE0
304D 3A 8E 63    ld      a, (byte_0_638E)
3050 4F          ld      c, a
3051 06 00      ld      b, #0
3053 21 00 76    ld      hl, #VRAM_start+0x200
3056 CD 64 30    call    copy_tile_from_next_column
3059 21 C0 75    ld      hl, #VRAM_start+0x1C0
305C CD 64 30    call    copy_tile_from_next_column
305F 21 8E 63    ld      hl, #byte_0_638E
3062 35          dec     (hl)
3063 C9          ret
3063          ; End of function wipe_ladder_as_kong_climbs
3063
3064          ; ██████████ SUBROUTINE ██████████
3064
3064          copy_tile_from_next_column:          ; CODE XREF: wipe_ladder_as_kong_climbs+C↑p
3064          ; wipe_ladder_as_kong_climbs+12↑p
3064 09          add     hl, bc
3065 7E          ld      a, (hl)
3066 19          add     hl, de
3067 77          ld      (hl), a
3068 C9          ret
3068          ; End of function copy_tile_from_next_column
3068
3069          ;
3069
3069          wait_and_inc_sequence:          ; DATA XREF: display_1UP+2D↑o
3069          ; display_1UP+31↑o ...
3069          ; wait for 8-bit countdown
3069 DF          rst     0x18
306A 2A C0 63    ld      hl, (ptr_current_sequence)
306D 34          inc     (hl)
306E C9          ret
306F
306F          ; ██████████ SUBROUTINE ██████████
306F
306F          animate_kong_climbing:          ; CODE XREF: display_1UP+95↑p
306F          ; 0000:1732↑p ...
306F 21 AF 62      ld      hl, #byte_0_62AF
3072 34          inc     (hl)
3073 7E          ld      a, (hl)
3074 E6 07      and     #7
3076 C0          ret     NZ
3077 21 0B 69    ld      hl, #soft_sprite_ram+0xB
307A 0E FC      ld      c, #0xFC ; '3'
307C FF          rst     0x38
307D 0E 81      ld      c, #0x81 ; 'ü'
307F 21 09 69    ld      hl, #soft_sprite_ram+9          ; sprite #2, flipy & code
3082 CD 96 30    call    flip_2_tiles
3085 21 1D 69    ld      hl, #soft_sprite_ram+0x1D          ; sprite #7, flipy & code
3088 CD 96 30    call    flip_2_tiles
308B CD 57 00    call    rand          ; Pauline kicking legs
308E E6 80      and     #0x80 ; 'Ç'
3090 21 2D 69    ld      hl, #soft_sprite_ram+0x2D          ; sprite #11, flipy & code (Pauline)
3093 AE          xor     (hl)
3094 77          ld      (hl), a

```

```

3095 C9                ret
3095                ; End of function animate_kong_climbing
3095
3096                ; ██████████ SUBROUTINE ██████████
3096
3096 flip_2_tiles:
3096 06 02                ; CODE XREF: animate_kong_climbing+13?p
3096                ; animate_kong_climbing+19?p
3096                ld        b, #2
3098
3098 loc_0_3098:
3098 79                ; CODE XREF: flip_2_tiles+6?j
3098                ld        a, c
3098 AE                xor     (hl)
3098 77                ld        (hl), a
3098 19                add     hl, de
3098 C 10 FA            djnz   loc_0_3098
3098 C9                ret
3098                ; End of function flip_2_tiles
3098
3098                ; ██████████ SUBROUTINE ██████████
3098
3098 queue_fg_vector_fn:
3098 E5                ; CODE XREF: check_coin_inserted+3B?p
3098                ; 0000:01F7?p ...
3098                push     hl
3098 21 C0 60            ld        hl, #fg_vector_fn_params
3098 3A B0 60            ld        a, (fg_fn_queue_tail)
3098 6F                ld        l, a
3098 CB 7E                bit     7, (hl)
3098 CA BB 30            jp      Z, loc_0_30BB
3098 72                ld        (hl), d
3098 2C                inc     l
3098 73                ld        (hl), e
3098 2C                inc     l
3098 7D                ld        a, l
3098 FE C0            cp      #0xC0 ; 'L'
3098 D2 B8 30            jp      NC, loc_0_30B8
3098 3E C0            ld        a, #0xC0 ; 'L'
3098
3098 loc_0_30B8:
3098 32 B0 60            ld        (fg_fn_queue_tail), a
3098
3098 loc_0_30BB:
3098 E1                pop      hl
3098 C9                ret
3098                ; End of function queue_fg_vector_fn
3098
3098                ; ██████████ SUBROUTINE ██████████
3098
3098 hide_object_sprites:
3098 21 50 69            ; CODE XREF: 0000:12A3?p
3098                ; 0000:1615?p
3098                ld        hl, #soft_sprite_ram+0x50
3098 06 02            ld        b, #2
3098 CD E4 30            call    zero_sprite_y_xB
3098 2E 80            ld        l, #0x80 ; 'C'
3098 06 0A            ld        b, #0xA
3098 CD E4 30            call    zero_sprite_y_xB
3098 2E B8            ld        l, #0xB8 ; '@'
3098 06 0B            ld        b, #0xB
3098 CD E4 30            call    zero_sprite_y_xB
3098 21 0C 6A            ld        hl, #soft_sprite_ram+0x10C
3098 06 05            ld        b, #5
3098 C3 E4 30            jp      zero_sprite_y_xB
3098                ; End of function hide_object_sprites
3098
3098                ; ██████████ SUBROUTINE ██████████
3098
3098 sub_0_30DB:
3098 21 4C 69            ; CODE XREF: 0000:12DF?p
3098                ; sprite #19 (Y)
3098                ld        hl, #soft_sprite_ram+0x4C
3098 36 00            ld        (hl), #0
3098 2E 58            ld        l, #0x58 ; 'X'
3098 06 06            ld        b, #6
3098                ; End of function sub_0_30DB
3098
3098                ; ██████████ SUBROUTINE ██████████
3098
3098 zero_sprite_y_xB:
3098 7D                ; CODE XREF: hide_object_sprites+5?p
3098                ; hide_object_sprites+C?p ...
3098                ld        a, l
3098
3098 loc_0_30E5:
3098                ; CODE XREF: zero_sprite_y_xB+6?j
3098 36 00            ld        (hl), #0
3098 C6 04            add     a, #4
3098 6F                ld        l, a
3098 A 10 F9            djnz   loc_0_30E5
3098 C9                ret
3098                ; End of function zero_sprite_y_xB
3098
3098                ; ██████████ SUBROUTINE ██████████
3098
3098 sub_0_30ED:
3098 CD FA 30            ; CODE XREF: 0000:198C?p
3098 CD 3C 31            call    sub_0_30FA
3098 CD B1 31            call    sub_0_31B1
3098 CD F3 34            call    sub_0_34F3
3098 C9                ret
3098                ; End of function sub_0_30ED
3098
3098                ; ██████████ SUBROUTINE ██████████
3098
3098 sub_0_30FA:
3098 3A 80 63            ; CODE XREF: sub_0_30ED?p
3098                ld        a, (unk_0_6380)
3098 FE 06            cp      #6
3098 38 02            jr      C, loc_0_3103

```

```

3101 3E 05          ld      a, #5
3103
3103          loc_0_3103:
3103 EF          rst      0x28          ; CODE XREF: sub_0_30FA+5↑j
3103          ;          ; go!
3104 10 31          .dw loc_0_3110          ; Jump table
3106 10 31          .dw loc_0_3110
3108 1B 31          .dw loc_0_311B
310A 26 31          .dw loc_0_3126
310C 26 31          .dw loc_0_3126
310E 31 31          .dw loc_0_3131
3110
3110
3110          loc_0_3110:
3110 3A 1A 60          ; DATA XREF: sub_0_30FA+A↑o
3110          ; sub_0_30FA+C↑o
3110          ld      a, (gen_purpose_timer)
3113 E6 01          and     #1
3115 FE 01          cp      #1
3117 C8          ret      Z
3118 33          inc     sp
3119 33          inc     sp
311A C9          ret
311B
311B
311B          loc_0_311B:
311B 3A 1A 60          ; DATA XREF: sub_0_30FA+E↑o
311E E6 07          ld      a, (gen_purpose_timer)
3120 FE 05          and     #7
3122 F8          cp      #5
3123 33          ret      M
3124 33          inc     sp
3125 C9          ret
3126
3126
3126          loc_0_3126:
3126 3A 1A 60          ; DATA XREF: sub_0_30FA+10↑o
3126          ; sub_0_30FA+12↑o
3129 E6 03          ld      a, (gen_purpose_timer)
312B FE 03          and     #3
312D F8          cp      #3
312E 33          ret      M
312F 33          inc     sp
3130 C9          ret
3131
3131
3131          loc_0_3131:
3131 3A 1A 60          ; DATA XREF: sub_0_30FA+14↑o
3134 E6 07          ld      a, (gen_purpose_timer)
3136 FE 07          and     #7
3138 F8          cp      #7
3139 33          ret      M
313A 33          inc     sp
313B C9          ret
313B          ; End of function sub_0_30FA
313B
313C          ; ██████████ S U B R O U T I N E ██████████
313C
313C          sub_0_313C:
313C DD 21 00 64          ; CODE XREF: sub_0_30ED+3↑p
3140 AF          ld      ix, #unk_0_6400          ; fireball character data
3141 32 A1 63          xor     a
3144 06 05          ld      (unk_0_63A1), a
3146 11 20 00          ld      b, #5
3149          ld      de, #0x20 ; ' '
3149
3149          loc_0_3149:
3149 DD 7E 00          ; CODE XREF: sub_0_313C+30↑j
314C FE 00          ld      a, 0(ix)
314E CA 7C 31          cp      #0
3151 3A A1 63          jp      Z, loc_0_317C
3154 3C          ld      a, (unk_0_63A1)
3155 32 A1 63          inc     a
3158 3E 01          ld      (unk_0_63A1), a
315A DD 77 08          ld      a, #1
315D 3A 17 62          ld      8(ix), a
3160 FE 01          ld      a, (unk_0_6217)
3162 C2 6A 31          cp      #1
3165 3E 00          jp      NZ, loc_0_316A
3167 DD 77 08          ld      a, #0
316A          ld      8(ix), a
316A
316A          loc_0_316A:
316A DD 19          ; CODE XREF: sub_0_313C+26↑j
316A          ; sub_0_313C+45↑j ...
316C 10 DB          add     ix, de
316E 21 A0 63          djnz   loc_0_3149
3171 36 00          ld      hl, #unk_0_63A0
3173 3A A1 63          ld      (hl), #0
3176 FE 00          ld      a, (unk_0_63A1)
3178 C0          cp      #0
3179 33          ret      NZ
317A 33          inc     sp
317B C9          inc     sp
317C          ret
317C
317C          loc_0_317C:
317C 3A A1 63          ; CODE XREF: sub_0_313C+12↑j
317F FE 05          ld      a, (unk_0_63A1)
3181 CA 6A 31          cp      #5
3184 3A 27 62          jp      Z, loc_0_316A
3187 FE 02          ld      a, (level_type)
3189 C2 95 31          cp      #2          ; cement level?
318C 3A A1 63          jp      NZ, loc_0_3195          ; no, continue
318F 4F          ld      a, (unk_0_63A1)          ; cement level timers
3190 3A 80 63          ld      c, a
3193 B9          ld      a, (unk_0_6380)
3194 C8          cp      c
3195          ret      Z
3195
3195          loc_0_3195:
3195 3A A0 63          ; CODE XREF: sub_0_313C+4D↑j
3198 FE 01          ; spawn a fireball
319A C2 6A 31          ld      a, (unk_0_63A0)
319D DD 77 00          cp      #1
31A0 DD 77 18          jp      NZ, loc_0_316A
31A3 AF          ld      0(ix), a
31A4 32 A0 63          ld      0x18(ix), a
31A4          xor     a
31A4          ld      (unk_0_63A0), a

```

```

31A7 3A A1 63      ld      a, (unk_0_63A1)
31AA 3C              inc      a
31AB 32 A1 63      ld      (unk_0_63A1), a
31AE C3 6A 31      jp      loc_0_316A
; End of function sub_0_313C
31AE
31AE
31B1
31B1 ; ██████████ S U B R O U T I N E ██████████
31B1
31B1 sub_0_31B1: ; CODE XREF: sub_0_30ED+6↑p
31B1 CD DD 31      call    sub_0_31DD
31B4 AF            xor      a
31B5 32 A2 63      ld      (unk_0_63A2), a
31B8 21 E0 63      ld      hl, #unk_0_63E0
31BB 22 C8 63      ld      (unk_0_63C8), hl
31BE
31BE loc_0_31BE: ; CODE XREF: sub_0_31B1+28↑j
31BE 2A C8 63      ld      hl, (unk_0_63C8)
31C1 01 20 00      ld      bc, #0x20 ; ' '
31C4 09            add     hl, bc
31C5 22 C8 63      ld      (unk_0_63C8), hl
31C8 7E            ld      a, (hl)
31C9 A7            and      a
31CA CA D0 31      jp      Z, loc_0_31D0
31CD CD 02 32      call    sub_0_3202
31D0
31D0 loc_0_31D0: ; CODE XREF: sub_0_31B1+19↑j
31D0 3A A2 63      ld      a, (unk_0_63A2)
31D3 3C              inc      a
31D4 32 A2 63      ld      (unk_0_63A2), a
31D7 FE 05          cp      #5
31D9 C2 BE 31      jp      NZ, loc_0_31BE
31DC C9              ret
; End of function sub_0_31B1
31DC
31DD
31DD ; ██████████ S U B R O U T I N E ██████████
31DD
31DD sub_0_31DD: ; CODE XREF: sub_0_31B1↑p
31DD 3A 80 63      ld      a, (unk_0_6380)
31E0 FE 03          cp      #3
31E2 F8              ret      M
31E3 CD F6 31      call    sub_0_31F6
31E6 FE 01          cp      #1
31E8 C0              ret      NZ
31E9 21 39 64      ld      hl, #unk_0_6439
31EC 3E 02          ld      a, #2
31EE 77            ld      (hl), a
31EF 21 79 64      ld      hl, #unk_0_6479
31F2 3E 02          ld      a, #2
31F4 77            ld      (hl), a
31F5 C9              ret
; End of function sub_0_31DD
31F5
31F6
31F6 ; ██████████ S U B R O U T I N E ██████████
31F6
31F6 sub_0_31F6: ; CODE XREF: sub_0_31DD+6↑p
31F6 3A 18 60      ld      a, (random_no)
31F9 E6 03          and      #3
31FB FE 01          cp      #1
31FD C0              ret      NZ
31FE 3A 1A 60      ld      a, (gen_purpose_timer)
3201 C9              ret
; End of function sub_0_31F6
3201
3202
3202 ; ██████████ S U B R O U T I N E ██████████
3202
3202 sub_0_3202: ; CODE XREF: sub_0_31B1+1C↑p
3202 DD 2A C8 63      ld      ix, (unk_0_63C8)
3206 DD 7E 18      ld      a, 0x18(ix)
3209 FE 01          cp      #1
320B CA 7A 32      jp      Z, loc_0_327A
320E DD 7E 0D      ld      a, 0xD(ix)
3211 FE 04          cp      #4
3213 F2 30 32      jp      P, loc_0_3230
3216 DD 7E 19      ld      a, 0x19(ix)
3219 FE 02          cp      #2
321B CA 7E 32      jp      Z, loc_0_327E
321E CD 0F 33      call    sub_0_330F
3221 3A 18 60      ld      a, (random_no)
3224 E6 03          and      #3
3226 C2 33 32      jp      NZ, loc_0_3233
3229
3229 loc_0_3229: ; CODE XREF: sub_0_3202+7F↑j
3229 DD 7E 0D      ld      a, 0xD(ix)
322C A7            and      a
322D CA 57 32      jp      Z, loc_0_3257
3230
3230 loc_0_3230: ; CODE XREF: sub_0_3202+11↑j
3230 CD 3D 33      call    sub_0_333D
3233
3233 loc_0_3233: ; CODE XREF: sub_0_3202+24↑j
3233 DD 7E 0D      ld      a, 0xD(ix)
3236 FE 04          cp      #4
3238 F2 91 32      jp      P, loc_0_3291
323B CD AD 33      call    sub_0_33AD
323E CD 8C 29      call    sub_0_298C
3241 FE 01          cp      #1
3243 CA 97 32      jp      Z, loc_0_3297
3246 DD 2A C8 63      ld      ix, (unk_0_63C8)
324A DD 7E 0E      ld      a, 0xE(ix)
324D FE 10          cp      #0x10
324F DA 8C 32      jp      C, loc_0_328C
3252 FE F0          cp      #0xF0 ; '-'
3254 D2 84 32      jp      NC, loc_0_3284
3257
3257 loc_0_3257: ; CODE XREF: sub_0_3202+2B↑j
3257 DD 7E 13      ld      a, 0x13(ix)
3257 ; sub_0_3202+87↑j ...
325A FE 00          cp      #0

```

```

325C C2 B9 32      jp      NZ, loc_0_32B9
325F 3E 11         ld      a, #0x11
3261
3261      loc_0_3261:
3261 DD 77 13      ld      0x13(ix), a ; CODE XREF: sub_0_3202+B8|j
3264 16 00      ld      d, #0
3266 5F         ld      e, a
3267 21 7A 3A    ld      hl, #fireball_bouncing_data
326A 19         add     hl, de
326B 7E         ld      a, (hl)
326C DD 46 0E    ld      b, 0xE(ix)
326F DD 70 03    ld      3(ix), b
3272 DD 4E 0F    ld      c, 0xF(ix)
3275 81         add     a, c
3276 DD 77 05    ld      5(ix), a
3279 C9         ret
327A
327A
327A      loc_0_327A:
327A CD BD 32      call     sub_0_32BD ; CODE XREF: sub_0_3202+9|j
327D C9         ret
327E
327E
327E      loc_0_327E:
327E CD D6 32      call     sub_0_32D6 ; CODE XREF: sub_0_3202+19|j
3281 C3 29 32    jp      loc_0_3229
3284
3284
3284      loc_0_3284:
3284 3E 02         ld      a, #2 ; CODE XREF: sub_0_3202+52|j
3286
3286      loc_0_3286:
3286 DD 77 0D      ld      0xD(ix), a ; CODE XREF: sub_0_3202+8C|j
3289 C3 57 32    jp      loc_0_3257
328C
328C      loc_0_328C:
328C 3E 01         ld      a, #1 ; CODE XREF: sub_0_3202+4D|j
328E C3 86 32    jp      loc_0_3286
3291
3291
3291      loc_0_3291:
3291 CD E7 33      call     sub_0_33E7 ; CODE XREF: sub_0_3202+36|j
3294 C3 57 32    jp      loc_0_3257
3297
3297
3297      loc_0_3297:
3297 DD 2A C8 63    ld      ix, (unk_0_63C8) ; CODE XREF: sub_0_3202+41|j
329B DD 7E 0D      ld      a, 0xD(ix)
329E FE 01      cp      #1
32A0 C2 B1 32    jp      NZ, loc_0_32B1
32A3 3E 02         ld      a, #2
32A5 DD 35 0E    dec     0xE(ix)
32A8
32A8      loc_0_32A8:
32A8 DD 77 0D      ld      0xD(ix), a ; CODE XREF: sub_0_3202+B4|j
32AB CD C3 33      call     sub_0_33C3
32AE C3 57 32    jp      loc_0_3257
32B1
32B1
32B1      loc_0_32B1:
32B1 3E 01         ld      a, #1 ; CODE XREF: sub_0_3202+9E|j
32B3 DD 34 0E      inc     0xE(ix)
32B6 C3 A8 32    jp      loc_0_32A8
32B9
32B9
32B9      loc_0_32B9:
32B9 3D         dec     a ; CODE XREF: sub_0_3202+5A|j
32BA C3 61 32    jp      loc_0_3261
32BA ; End of function sub_0_3202
32BD
32BD ; ██████████ S U B R O U T I N E ██████████
32BD
32BD
32BD      sub_0_32BD:
32BD 3A 27 62      ld      a, (level_type) ; CODE XREF: sub_0_3202+78|p
32C0 FE 01      cp      #1
32C2 CA CE 32    jp      Z, loc_0_32CE
32C5 FE 02      cp      #2
32C7 CA D2 32    jp      Z, loc_0_32D2
32CA CD B9 34      call     sub_0_34B9
32CD C9         ret
32CE
32CE
32CE      loc_0_32CE:
32CE CD 2C 34      call     sub_0_342C ; CODE XREF: sub_0_32BD+5|j
32D1 C9         ret
32D2
32D2
32D2      loc_0_32D2:
32D2 CD 78 34      call     sub_0_3478 ; CODE XREF: sub_0_32BD+A|j
32D5 C9         ret
32D5 ; End of function sub_0_32BD
32D6
32D6 ; ██████████ S U B R O U T I N E ██████████
32D6
32D6
32D6      sub_0_32D6:
32D6 DD 7E 1C      ld      a, 0x1C(ix) ; CODE XREF: sub_0_3202+7C|p
32D9 FE 00      cp      #0
32DB C2 FD 32    jp      NZ, loc_0_32FD
32DE DD 7E 1D      ld      a, 0x1D(ix)
32E1 FE 01      cp      #1
32E3 C2 0B 33    jp      NZ, loc_0_330B
32E6 DD 36 1D 00    ld      0x1D(ix), #0
32EA 3A 05 62    ld      a, (mario_x)
32ED DD 46 0F    ld      b, 0xF(ix)
32F0 90         sub     b
32F1 DA 03 33      jp      C, loc_0_3303
32F4 DD 36 1C FF    ld      0x1C(ix), #0xFF
32F8
32F8      loc_0_32F8:
32F8 DD 36 0D 00    ld      0xD(ix), #0 ; CODE XREF: sub_0_32D6+2A|j
32FC C9         ret

```

```

32FD ; -----
32FD
32FD loc_0_32FD: ; CODE XREF: sub_0_32D6+5↑j
32FD DD 35 1C      dec      0x1C(ix)
3300 C2 F8 32      jp       NZ, loc_0_32F8
3303
3303 loc_0_3303: ; CODE XREF: sub_0_32D6+1B↑j
3303 DD 36 19 00      ld       0x19(ix), #0
3307 DD 36 1C 00      ld       0x1C(ix), #0
330B
330B loc_0_330B: ; CODE XREF: sub_0_32D6+D↑j
330B CD 0F 33      call     sub_0_330F
330E C9              ret
330E ; End of function sub_0_32D6
330E
330F ; [REDACTED] S U B R O U T I N E [REDACTED]
330F
330F sub_0_330F: ; CODE XREF: sub_0_3202+1C↑p
330F DD 7E 16      ; sub_0_32D6+35↑p
330F          ld       a, 0x16(ix)
3312 FE 00          cp       #0
3314 C2 32 33      jp       NZ, loc_0_3332
3317 DD 36 16 2B      ld       0x16(ix), #0x2B ; '+'
331B DD 36 0D 00      ld       0xD(ix), #0
331F 3A 18 60      ld       a, (random_no)
3322 0F          rrca
3323 D2 32 33      jp       NC, loc_0_3332
3326 DD 7E 0D      ld       a, 0xD(ix)
3329 FE 01          cp       #1
332B CA 36 33      jp       Z, loc_0_3336
332E DD 36 0D 01      ld       0xD(ix), #1
3332
3332 loc_0_3332: ; CODE XREF: sub_0_330F+5↑j
3332 DD 35 16      ; sub_0_330F+14↑j ...
3332          dec      0x16(ix)
3335 C9              ret
3336 ; -----
3336
3336 loc_0_3336: ; CODE XREF: sub_0_330F+1C↑j
3336 DD 36 0D 02      ld       0xD(ix), #2
333A C3 32 33      jp       loc_0_3332
333A ; End of function sub_0_330F
333A
333D ; [REDACTED] S U B R O U T I N E [REDACTED]
333D
333D sub_0_333D: ; CODE XREF: sub_0_3202+2E↑p
333D DD 7E 0D      ld       a, 0xD(ix)
3340 FE 08          cp       #8
3342 CA 71 33      jp       Z, loc_0_3371
3345 FE 04          cp       #4
3347 CA 8A 33      jp       Z, loc_0_338A
334A CD A1 33      call     sub_0_33A1
334D DD 7E 0F      ld       a, 0xF(ix)
3350 C6 08          add      a, #8
3352 57          ld       d, a
3353 DD 7E 0E      ld       a, 0xE(ix)
3356 01 15 00      ld       bc, #0x15
3359 CD 6E 23      call     sub_0_236E
335C A7          and      a
335D CA 99 33      jp       Z, loc_0_3399
3360 DD 70 1F      ld       0x1F(ix), b
3363 3A 05 62      ld       a, (mario_x)
3366 47          ld       b, a
3367 DD 7E 0F      ld       a, 0xF(ix)
336A 90          sub      b, a
336B D0          ret     NC
336C DD 36 0D 04      ld       0xD(ix), #4
3370 C9              ret
3371 ; -----
3371
3371 loc_0_3371: ; CODE XREF: sub_0_333D+5↑j
3371 DD 7E 0F      ld       a, 0xF(ix)
3374 C6 08          add      a, #8
3376 DD 46 1F      ld       b, 0x1F(ix)
3379 B8          cp       b
337A C0          ret     NZ
337B DD 36 0D 00      ld       0xD(ix), #0
337F DD 7E 19      ld       a, 0x19(ix)
3382 FE 02          cp       #2
3384 C0          ret     NZ
3385 DD 36 1D 01      ld       0x1D(ix), #1
3389 C9              ret
338A ; -----
338A
338A loc_0_338A: ; CODE XREF: sub_0_333D+A↑j
338A DD 7E 0F      ld       a, 0xF(ix)
338D C6 08          add      a, #8
338F DD 46 1F      ld       b, 0x1F(ix)
3392 B8          cp       b
3393 C0          ret     NZ
3394 DD 36 0D 00      ld       0xD(ix), #0
3398 C9              ret
3399 ; -----
3399
3399 loc_0_3399: ; CODE XREF: sub_0_333D+20↑j
3399 DD 70 1F      ld       0x1F(ix), b
339C DD 36 0D 08      ld       0xD(ix), #8
33A0 C9              ret
33A0 ; End of function sub_0_333D
33A0
33A1 ; [REDACTED] S U B R O U T I N E [REDACTED]
33A1
33A1 sub_0_33A1: ; CODE XREF: sub_0_333D+D↑p
33A1          ld       a, #7
33A3 F7          rst      0x30 ; return if level bit not set
33A4 DD 7E 0F      ld       a, 0xF(ix)
33A7 FE 59          cp       #0x59 ; 'Y'
33A9 D0          ret     NC
33AA 33          inc     sp
33AB 33          inc     sp

```

```

33AC C9                ret
33AC                ; End of function sub_0_33A1
33AC
33AD                ; SUBROUTINE
33AD
33AD                ;
33AD
sub_0_33AD:          ; CODE XREF: sub_0_3202+39↑p
33AD DD 7E 0D          ld      a, 0xD(ix)
33B0 FE 01            cp      #1
33B2 CA D9 33          jp      Z, loc_0_33D9
33B5 DD 7E 07          ld      a, 7(ix)
33B8 E6 7F            and     #0x7F ; ' '
33BA DD 77 07          ld      7(ix), a
33BD DD 35 0E          dec     0xE(ix)
33C0
33C0                ; CODE XREF: 0000:33E4↑j
33C0 CD 09 34          call     sub_0_3409
33C0                ; End of function sub_0_33AD
33C0
33C3                ; SUBROUTINE
33C3
33C3                ;
33C3
sub_0_33C3:          ; CODE XREF: sub_0_3202+A9↑p
33C3 3A 27 62          ld      a, (level_type)
33C6 FE 01            cp      #1
33C8 C0              ret     NZ
33C9 DD 66 0E          ld      h, 0xE(ix)
33CC DD 6E 0F          ld      l, 0xF(ix)
33CF DD 46 0D          ld      b, 0xD(ix)
33D2 CD 33 23          call     adjust_height_on_girders
33D5 DD 75 0F          ld      0xF(ix), l
33D8 C9              ret
33D8                ; End of function sub_0_33C3
33D8
33D9                ;
33D9
33D9                ;
33D9
loc_0_33D9:          ; CODE XREF: sub_0_33AD+5↑j
33D9 DD 7E 07          ld      a, 7(ix)
33DC F6 80            or      #0x80 ; 'Ç'
33DE DD 77 07          ld      7(ix), a
33E1 DD 34 0E          inc     0xE(ix)
33E4 C3 C0 33          jp      loc_0_33C0
33E7
33E7                ; SUBROUTINE
33E7
33E7                ;
33E7
sub_0_33E7:          ; CODE XREF: sub_0_3202+8F↑p
33E7 CD 09 34          call     sub_0_3409
33EA DD 7E 0D          ld      a, 0xD(ix)
33ED FE 08            cp      #8
33EF C2 05 34          jp      NZ, loc_0_3405
33F2 DD 7E 14          ld      a, 0x14(ix)
33F5 A7              and     a
33F6 C2 01 34          jp      NZ, loc_0_3401
33F9 DD 36 14 02       ld      0x14(ix), #2
33FD DD 35 0F          dec     0xF(ix)
3400 C9              ret
3401
3401                ;
3401
loc_0_3401:          ; CODE XREF: sub_0_33E7+F↑j
3401 DD 35 14          dec     0x14(ix)
3404 C9              ret
3405
3405                ;
3405
loc_0_3405:          ; CODE XREF: sub_0_33E7+8↑j
3405 DD 34 0F          inc     0xF(ix)
3408 C9              ret
3408                ; End of function sub_0_33E7
3408
3409                ; SUBROUTINE
3409
3409                ;
3409
sub_0_3409:          ; CODE XREF: sub_0_33AD+13↑p
3409                ; sub_0_33E7↑p
3409 DD 7E 15          ld      a, 0x15(ix)
340C A7              and     a
340D C2 28 34          jp      NZ, loc_0_3428
3410 DD 36 15 02       ld      0x15(ix), #2
3414 DD 34 07          inc     7(ix)
3417 DD 7E 07          ld      a, 7(ix)
341A E6 0F            and     #0xF
341C FE 0F            cp      #0xF
341E C0              ret     NZ
341F DD 7E 07          ld      a, 7(ix)
3422 EE 02            xor     #2
3424 DD 77 07          ld      7(ix), a
3427 C9              ret
3428
3428                ;
3428
loc_0_3428:          ; CODE XREF: sub_0_3409+4↑j
3428 DD 35 15          dec     0x15(ix)
342B C9              ret
342B                ; End of function sub_0_3409
342B
342C                ; SUBROUTINE
342C
342C                ;
342C
sub_0_342C:          ; CODE XREF: sub_0_32BD+11↑p
342C DD 6E 1A          ld      l, 0x1A(ix)
342F DD 66 1B          ld      h, 0x1B(ix)
3432 AF              xor     a
3433 01 00 00          ld      bc, #0
3436 ED 4A            adc     hl, bc
3438 C2 42 34          jp      NZ, loc_0_3442
343B 21 8C 3A          ld      hl, #fireball_bounce_data
343E DD 36 03 26       ld      3(ix), #0x26 ; '&'
3442
3442                ; CODE XREF: sub_0_342C+C↑j
3442 DD 34 03          inc     3(ix)
3445
3445                ; CODE XREF: sub_0_3478+2D↑j
3445                ; sub_0_3478+3E↑j
3445 7E

```

```

3445                ld      a, (hl)
3446 FE AA          cp      #0xAA ; '-'
3448 CA 56 34      jp      Z, loc_0_3456
344B DD 77 05      ld      5(ix), a
344E 23          inc     hl
344F DD 75 1A      ld      0x1A(ix), l
3452 DD 74 1B      ld      0x1B(ix), h
3455 C9          ret
3456
3456
3456 loc_0_3456:                ; CODE XREF: sub_0_342C+1C↑j
3456 AF          xor      a
3457 DD 77 13      ld      0x13(ix), a
345A DD 77 18      ld      0x18(ix), a
345D DD 77 0D      ld      0xD(ix), a
3460 DD 77 1C      ld      0x1C(ix), a
3463 DD 7E 03      ld      a, 3(ix)
3466 DD 77 0E      ld      0xE(ix), a
3469 DD 7E 05      ld      a, 5(ix)
346C DD 77 0F      ld      0xF(ix), a
346F DD 36 1A 00   ld      0x1A(ix), #0
3473 DD 36 1B 00   ld      0x1B(ix), #0
3477 C9          ret
3477 ; End of function sub_0_342C
3478
3478 ; ██████████ SUBROUTINE ██████████
3478
3478 sub_0_3478:                ; CODE XREF: sub_0_32BD+15↑p
3478 DD 6E 1A      ld      l, 0x1A(ix)
347B DD 66 1B      ld      h, 0x1B(ix)
347E AF          xor      a
347F 01 00 00     ld      bc, #0
3482 ED 4A      adc     hl, bc
3484 C2 9A 34     jp      NZ, loc_0_349A
3487 21 AC 3A     ld      hl, #cement_fireball_data
348A 3A 03 62     ld      a, (mario_y)
348D CB 7F      bit     7, a
348F CA A8 34     jp      Z, loc_0_34A8
3492 DD 36 0D 01   ld      0xD(ix), #1
3496 DD 36 03 7E   ld      3(ix), #0x7E ; '~'
349A
349A loc_0_349A:                ; CODE XREF: sub_0_3478+C↑j
349A DD 7E 0D      ; sub_0_3478+38↑j
349A          ld      a, 0xD(ix)
349D FE 01      cp      #1
349F C2 B3 34     jp      NZ, loc_0_34B3
34A2 DD 34 03     inc     3(ix)
34A5 C3 45 34     jp      loc_0_3445
34A8
34A8
34A8 loc_0_34A8:                ; CODE XREF: sub_0_3478+17↑j
34A8 DD 36 0D 02   ld      0xD(ix), #2
34AC DD 36 03 80   ld      3(ix), #0x80 ; 'Q'
34B0 C3 9A 34     jp      loc_0_349A
34B3
34B3
34B3 loc_0_34B3:                ; CODE XREF: sub_0_3478+27↑j
34B3 DD 35 03      dec     3(ix)
34B6 C3 45 34     jp      loc_0_3445
34B6 ; End of function sub_0_3478
34B9
34B9 ; ██████████ SUBROUTINE ██████████
34B9
34B9 sub_0_34B9:                ; CODE XREF: sub_0_32BD+D↑p
34B9 3A 27 62      ld      a, (level_type)
34BC FE 03      cp      #3
34BE C8          ret     Z
34BF 3A 03 62     ld      a, (mario_y)
34C2 CB 7F      bit     7, a
34C4 C2 ED 34     jp      NZ, loc_0_34ED
34C7 21 C4 3A     ld      hl, #rivet_fireball_data
34CA
34CA loc_0_34CA:                ; CODE XREF: sub_0_34B9+37↑j
34CA 06 00      ld      b, #0
34CC 3A 19 60     ld      a, (random_no+1)
34CF E6 06      and     #6
34D1 4F      ld      c, a
34D2 09      add     hl, bc
34D3 7E      ld      a, (hl)
34D4 DD 77 03     ld      3(ix), a
34D7 DD 77 0E     ld      0xE(ix), a
34DA 23      inc     hl
34DB 7E      ld      a, (hl)
34DC DD 77 05     ld      5(ix), a
34DF DD 77 0F     ld      0xF(ix), a
34E2 AF      xor      a
34E3 DD 77 0D     ld      0xD(ix), a
34E6 DD 77 18     ld      0x18(ix), a
34E9 DD 77 1C     ld      0x1C(ix), a
34EC C9          ret
34ED
34ED
34ED loc_0_34ED:                ; CODE XREF: sub_0_34B9+B↑j
34ED 21 D4 3A     ld      hl, #rivet_fireball_start_points
34F0 C3 CA 34     jp      loc_0_34CA
34F0 ; End of function sub_0_34B9
34F3
34F3 ; ██████████ SUBROUTINE ██████████
34F3
34F3 sub_0_34F3:                ; CODE XREF: sub_0_30ED+9↑p
34F3 21 00 64      ld      hl, #unk_0_6400 ; fireball character data
34F6 11 D0 69     ld      de, #soft_sprite_ram+0xD0 ; fireballs in sprite ram
34F9 06 05     ld      b, #5 ; 5 fireballs (max)
34FB
34FB loc_0_34FB:                ; CODE XREF: sub_0_34F3+28↑j
34FB 7E      ld      a, (hl)
34FC A7      and     a
34FD CA 1E 35     jp      Z, loc_0_351E
3500 2C      inc     l
3501 2C      inc     l

```



```

3502 2C          inc     1
3503 7E          ld      a, (hl)
3504 12          ld      (de), a
3505 3E 04       ld      a, #4
3507 85          add     a, 1
3508 6F          ld      l, a
3509 1C          inc     e
350A 7E          ld      a, (hl)
350B 12          ld      (de), a
350C 2C          inc     1
350D 1C          inc     e
350E 7E          ld      a, (hl)
350F 12          ld      (de), a
3510 2D          dec     1
3511 2D          dec     1
3512 2D          dec     1
3513 1C          inc     e
3514 7E          ld      a, (hl)
3515 12          ld      (de), a
3516 13          inc     de
3517
3517          loc_0_3517:
3517 3E 1B       ld      a, #0x1B
3519 85          add     a, 1
351A 6F          ld      l, a
351B 10 DE      djnz    loc_0_34FB
351D C9          ret
;
351E
351E          loc_0_351E:
351E 3E 05       ld      a, #5
3520 85          add     a, 1
3521 6F          ld      l, a
3522 3E 04       ld      a, #4
3524 83          add     a, e
3525 5F          ld      e, a
3526 C3 17 35   jp      loc_0_3517
; End of function sub_0_34F3
;
3529 00 00 00    bonus_points_tbl: .db 0, 0, 0
352C 00 01 00    .db 0, 1, 0
352F 00 02 00    .db 0, 2, 0
3532 00 03 00    .db 0, 3, 0
3535 00 04 00    .db 0, 4, 0
3538 00 05 00    .db 0, 5, 0
353B 00 06 00    .db 0, 6, 0
353E 00 07 00    .db 0, 7, 0
3541 00 08 00    .db 0, 8, 0
3544 00 09 00    .db 0, 9, 0
3547 00 00 00    .db 0, 0, 0
354A 00 10 00    .db 0, 0x10, 0
354D 00 20 00    .db 0, 0x20, 0
3550 00 30 00    .db 0, 0x30, 0
3553 00 40 00    .db 0, 0x40, 0
3556 00 50 00    .db 0, 0x50, 0
3559 00 60 00    .db 0, 0x60, 0
355C 00 70 00    .db 0, 0x70, 0
355F 00 80 00    .db 0, 0x80, 0
3562 00 90 00    .db 0, 0x90, 0
3565 94 77      high_score_tbl: .dw VRAM_start+0x394
3567 01 23 24 10+ .db 1, 0x23, 0x24, 0x10, 0x10, 0, 0, 7, 6, 5, 0, 0x10
3567 10 00 00 07+ .db 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10
3567 06 05 00 10+ .db 0x10, 0x10, 0x10, 0x10, 0x3F, 0, 0x50, 0x76, 0
3585 F4 76      .dw VRAM_start+0x2F4
3587 96 77      .dw VRAM_start+0x396
3589 02 1E 14 10+ .db 2, 0x1E, 0x14, 0x10, 0x10, 0, 0, 6, 1, 0, 0, 0x10
3589 10 00 00 06+ .db 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10
3589 01 00 00 10+ .db 0x10, 0x10, 0x10, 0x10, 0x3F, 0, 0, 0x61, 0
35A7 F6 76      .dw VRAM_start+0x2F6
35A9 98 77      .dw VRAM_start+0x398
35AB 03 22 14 10+ .db 3, 0x22, 0x14, 0x10, 0x10, 0, 0, 5, 9, 5, 0, 0x10
35AB 10 00 00 05+ .db 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10
35AB 09 05 00 10+ .db 0x10, 0x10, 0x10, 0x10, 0x3F, 0, 0x50, 0x59, 0
35C9 F8 76      .dw VRAM_start+0x2F8
35CB 9A 77      .dw VRAM_start+0x39A
35CD 04 24 18 10+ .db 4, 0x24, 0x18, 0x10, 0x10, 0, 0, 5, 0, 5, 0, 0x10
35CD 10 00 00 05+ .db 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10
35CD 00 05 00 10+ .db 0x10, 0x10, 0x10, 0x10, 0x3F, 0, 0x50, 0x50, 0
35EB FA 76      .dw VRAM_start+0x2FA
35ED 9C 77      .dw VRAM_start+0x39C
35EF 05 24 18 10+ .db 5, 0x24, 0x18, 0x10, 0x10, 0, 0, 4, 3, 0, 0, 0x10
35EF 10 00 00 04+ .db 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10
35EF 03 00 00 10+ .db 0x10, 0x10, 0x10, 0x10, 0x3F, 0, 0, 0x43, 0
360D FC 76      .dw VRAM_start+0x2FC
360F 3B 5C 4B 5C+letter_coords: .db 0x3B, 0x5C, 0x4B, 0x5C, 0x5B, 0x5C, 0x6B, 0x5C, 0x7B
360F 5B 5C 6B 5C+ .db 0x5C, 0x8B, 0x5C, 0x9B, 0x5C, 0xAB, 0x5C, 0xBB, 0x5C
360F 7B 5C 8B 5C+ .db 0xCB, 0x5C, 0x3B, 0x6C, 0x4B, 0x6C, 0x5B, 0x6C, 0x6B
360F 9B 5C AB 5C+ .db 0x6C, 0x7B, 0x6C, 0x8B, 0x6C, 0x9B, 0x6C, 0xAB, 0x6C
360F BB 5C CB 5C+ .db 0xBB, 0x6C, 0xCB, 0x6C, 0x3B, 0x7C, 0x4B, 0x7C, 0x5B
360F 3C 6C 6B 6C+ .db 0x7C, 0x6B, 0x7C, 0x7B, 0x7C, 0x8B, 0x7C, 0x9B, 0x7C
360F 5B 6C 6B 6C+ .db 0xAB, 0x7C, 0xBB, 0x7C, 0xCB, 0x7C
360F 7B 6C 8B 6C+
364B 8B 36      message_table: .dw aGAME_OVER
364D 01 00      .dw 1
364F 98 36      .dw aPLAYER_I
3651 A5 36      .dw aPLAYER_II
3653 B2 36      .dw aHIGH_SCORE
3655 BF 36      .dw aCREDIT
3657 06 00      .dw 6
3659 CC 36      .dw aHOW_HIGH_CAN_YOU_GET
365B 08 00      .dw 8
365D E6 36      .dw aONLY_1_PLAYER_BUTTON
365F FD 36      .dw a1_OR_2_PLAYERS
3661 0B 00      .dw 0xB
3663 15 37      .dw aPUSH
3665 1C 37      .dw aNAME_REGISTRATION
3667 30 37      .dw aNAME
3669 38 37      .dw aDASHDASHDASH
366B 47 37      .dw aA_B_C_D_E_F_G_H_I_J
366D 5D 37      .dw aK_L_M_N_O_P_Q_R_S_T
366F 73 37      .dw aU_V_W_X_Y_Z_rub_end
3671 8B 37      .dw aREGI_TIME
3673 00 61      .dw high_score_tbl_ram
3675 22 61      .dw hs_tbl_2nd

```

```

3677 44 61      .dw hs_tbl_3rd
3679 66 61      .dw hs_tbl_4th
367B 88 61      .dw hs_tbl_5th
367D 9E 37      .dw aRANK_SCORE_NAME
367F B6 37      .dw aYOUR_NAME_WAS_REGISTERED
3681 D2 37      .dw aINSERT_COIN
3683 E1 37      .dw aPLAYER_COIN
3685 1D 00      .dw 0x1D
3687 00 3F      .dw aCOPYRIGHT_1981
3689 09 3F      .dw aNINTENDO_OF_AMERICA_INC
368B 96 76      aGAME_OVER:      .dw VRAM_start+0x296      ; DATA XREF: 0000:364B|o
368D 17 11 1D 15+      .db 0x17, 0x11, 0x1D, 0x15, 0x10, 0x10, 0x1F, 0x26, 0x15
368D 10 10 1F 26+      .db 0x22, 0x3F
3698 94 76      aPLAYER_I:      .dw VRAM_start+0x294      ; DATA XREF: 0000:364F|o
369A 20 1C 11 29+      .db 0x20, 0x1C, 0x11, 0x29, 0x15, 0x22, 0x10, 0x30, 0x32
369A 15 22 10 30+      .db 0x31, 0x3F
36A5 94 76      aPLAYER_II:      .dw VRAM_start+0x294      ; DATA XREF: 0000:3651|o
36A7 20 1C 11 29+      .db 0x20, 0x1C, 0x11, 0x29, 0x15, 0x22, 0x10, 0x30, 0x33
36A7 15 22 10 30+      .db 0x31, 0x3F
36B2 80 76      aHIGH_SCORE:      .dw VRAM_start+0x280      ; DATA XREF: 0000:3653|o
36B4 18 19 17 18+      .db 0x18, 0x19, 0x17, 0x18, 0x10, 0x23, 0x13, 0x1F, 0x22
36B4 10 23 13 1F+      .db 0x15, 0x3F
36BF 9F 75      aCREDIT:      .dw VRAM_start+0x19F      ; DATA XREF: 0000:3655|o
36C1 13 22 15 14+      .db 0x13, 0x22, 0x15, 0x14, 0x19, 0x24, 0x10, 0x10, 0x10
36C1 19 24 10 10+      .db 0x10, 0x3F
36CC 5E 77      aHOW_HIGH_CAN_YOU_GET:      .dw VRAM_start+0x35E      ; DATA XREF: 0000:3659|o
36CE 18 1F 27 10+      .db 0x18, 0x1F, 0x27, 0x10, 0x18, 0x19, 0x17, 0x18, 0x10
36CE 18 19 17 18+      .db 0x13, 0x11, 0x1E, 0x10, 0x29, 0x1F, 0x25, 0x10, 0x17
36CE 10 13 11 1E+      .db 0x15, 0x24, 0x10, 0xFB, 0x10, 0x3F
36E6 29 77      aONLY_1_PLAYER_BUTTON:      .dw VRAM_start+0x329      ; DATA XREF: 0000:365D|o
36E8 1F 1E 1C 29+      .db 0x1F, 0x1E, 0x1C, 0x29, 0x10, 1, 0x10, 0x20, 0x1C
36E8 10 01 10 20+      .db 0x11, 0x29, 0x15, 0x22, 0x10, 0x12, 0x25, 0x24, 0x24
36E8 1C 11 29 15+      .db 0x1F, 0x1E, 0x3F
36FD 29 77      aI_OR_2_PLAYERS:      .dw VRAM_start+0x329      ; DATA XREF: 0000:365F|o
36FF 01 10 1F 22+      .db 1, 0x10, 0x1F, 0x22, 0x10, 2, 0x10, 0x20, 0x1C, 0x11
36FF 10 02 10 20+      .db 0x29, 0x15, 0x22, 0x23, 0x10, 0x12, 0x25, 0x24, 0x24
36FF 1C 11 29 15+      .db 0x1F, 0x1E, 0x3F
3715 27 76      aPUSH:      .dw VRAM_start+0x227      ; DATA XREF: 0000:3663|o
3717 20 25 23 18+      .db 0x20, 0x25, 0x23, 0x18, 0x3F
371C 06 77      aNAME_REGISTRATION:      .dw VRAM_start+0x306      ; DATA XREF: 0000:3665|o
371E 1E 11 1D 15+      .db 0x1E, 0x11, 0x1D, 0x15, 0x10, 0x22, 0x15, 0x17, 0x19
371E 10 22 15 17+      .db 0x23, 0x24, 0x22, 0x11, 0x24, 0x19, 0x1F, 0x1E, 0x3F
3730 88 76      aNAME:      .dw VRAM_start+0x288      ; DATA XREF: 0000:3667|o
3732 1E 11 1D 15+      .db 0x1E, 0x11, 0x1D, 0x15, 0x2E, 0x3F
3738 E9 75      aDASHDASHDASH:      .dw VRAM_start+0x1E9      ; DATA XREF: 0000:3669|o
373A 2D 2D 2D 10+      .db 0x2D, 0x2D, 0x2D, 0x10, 0x10, 0x10, 0x10, 0x10, 0x10
373A 10 10 10 10+      .db 0x10, 0x10, 0x10, 0x10, 0x3F
3747 0B 77      aA_B_C_D_E_F_G_H_I_J:      .dw VRAM_start+0x30B      ; DATA XREF: 0000:366B|o
3749 11 10 12 10+      .db 0x11, 0x10, 0x12, 0x10, 0x13, 0x10, 0x14, 0x10, 0x15
3749 13 10 14 10+      .db 0x10, 0x16, 0x10, 0x17, 0x10, 0x18, 0x10, 0x19, 0x10
3749 15 10 16 10+      .db 0x1A, 0x3F
375D 0D 77      aK_L_M_N_O_P_Q_R_S_T:      .dw VRAM_start+0x30D      ; DATA XREF: 0000:366D|o
375F 1B 10 1C 10+      .db 0x1B, 0x10, 0x1C, 0x10, 0x1D, 0x10, 0x1E, 0x10, 0x1F
375F 1D 10 1E 10+      .db 0x10, 0x20, 0x10, 0x21, 0x10, 0x22, 0x10, 0x23, 0x10
375F 1F 10 20 10+      .db 0x24, 0x3F
3773 0F 77      aU_V_W_X_Y_Z_rub_end:      .dw VRAM_start+0x30F      ; DATA XREF: 0000:366F|o
3775 25 10 26 10+      .db 0x25, 0x10, 0x26, 0x10, 0x27, 0x10, 0x28, 0x10, 0x29
3775 27 10 28 10+      .db 0x10, 0x2A, 0x10, 0x2B, 0x10, 0x2C, 0x44, 0x45, 0x46
3775 29 10 2A 10+      .db 0x47, 0x48, 0x10, 0x3F
378B F2 76      aREGI_TIME:      .dw VRAM_start+0x2F2      ; DATA XREF: 0000:3671|o
378D 22 15 17 19+      .db 0x22, 0x15, 0x17, 0x19, 0x10, 0x24, 0x19, 0x1D, 0x15
378D 10 24 19 1D+      .db 0x10, 0x10, 0x30, 3, 0, 0x31, 0x10, 0x3F
379E 92 77      aRANK_SCORE_NAME:      .dw VRAM_start+0x392      ; DATA XREF: 0000:367D|o
37A0 2D 11 1E 1B+      .db 0x22, 0x11, 0x1E, 0x1B, 0x10, 0x10, 0x23, 0x13, 0x1F
37A0 10 10 23 13+      .db 0x22, 0x15, 0x10, 0x10, 0x1E, 0x11, 0x1D, 0x15, 0x10
37A0 1F 22 15 10+      .db 0x10, 0x10, 0x10, 0x3F
37B6 72 77      aYOUR_NAME_WAS_REGISTERED:      .dw VRAM_start+0x372      ; DATA XREF: 0000:367F|o
37B8 29 1F 25 22+      .db 0x29, 0x1F, 0x25, 0x22, 0x10, 0x1E, 0x11, 0x1D, 0x15
37B8 10 1E 11 1D+      .db 0x10, 0x27, 0x11, 0x23, 0x10, 0x22, 0x15, 0x17, 0x19
37B8 15 10 27 11+      .db 0x23, 0x24, 0x15, 0x22, 0x15, 0x14, 0x42, 0x3F
37D2 A7 76      aINSERT_COIN:      .dw VRAM_start+0x2A7      ; DATA XREF: 0000:3681|o
37D4 19 1E 23 15+      .db 0x19, 0x1E, 0x23, 0x15, 0x22, 0x24, 0x10, 0x13, 0x1F
37D4 22 24 10 13+      .db 0x19, 0x1E, 0x10, 0x3F
37E1 0A 77      aPLAYER_COIN:      .dw VRAM_start+0x30A      ; DATA XREF: 0000:3683|o
37E3 10 10 20 1C+      .db 0x10, 0x10, 0x20, 0x1C, 0x11, 0x29, 0x15, 0x22, 0x10
37E3 11 29 15 22+      .db 0x10, 0x10, 0x10, 0x13, 0x1F, 0x19, 0x1E, 0x3F
37F4 FC 76      .dw VRAM_start+0x2FC
37F6 49 4A 10 1E+a_NINTENDO:      .db 0x49, 0x4A, 0x10, 0x1E, 0x19, 0x1E, 0x24, 0x15, 0x1E
37F6 19 1E 24 15+      .db 0x14, 0x1F, 0x10, 0x10, 0x10, 0x10, 0x3F
3806 7C 75      .dw VRAM_start+0x17C
3808 01 09 08 01+      .db 1, 9, 8, 1, 0x3F
380D 02 97 38 68+draw_data_climb:      .db 2, 0x97, 0x38, 0x68, 0x38, 2, 0xDF, 0x54, 0x10, 0x54
380D 38 02 DF 54+      ; DATA XREF: display_LUP+3F|o
380D 10 54 02 EF+      .db 2, 0xEF, 0x6D, 0x20, 0x6D, 2, 0xDF, 0x8E, 0x10, 0x8E
380D 6D 20 6D 02+      .db 2, 0xEF, 0xAF, 0x20, 0xAF, 2, 0xDF, 0xD0, 0x10, 0xD0
380D DF 8E 10 8E+      .db 2, 0xEF, 0xF1, 0x10, 0xF1, 0, 0x53, 0x18, 0x53, 0x54
380D 02 EF AF 20+      .db 0, 0x63, 0x18, 0x63, 0x54, 0, 0x93, 0x38, 0x93, 0x54
380D AF 02 DF D0+      .db 0, 0x83, 0x54, 0x83, 0xF1, 0, 0x93, 0x54, 0x93, 0xF1
380D 10 D0 02 EF+      .db 0xAA
384A 8D 7D 8C      bonus_graphic_tiles:      .db 0x8D, 0x7D, 0x8C      ; DATA XREF: 0000:064D|o
384D 6F 00 7C      .db 0x6F, 0, 0x7C
3850 68 00 7C      .db 0x6E, 0, 0x7C
3853 6D 00 7C      .db 0x6D, 0, 0x7C
3856 6C 00 7C      .db 0x6C, 0, 0x7C
3859 8F 7F 8E      .db 0x8F, 0x7F, 0x8E
385C 47 27 08 50+dk_normal_spr:      .db 0x47, 0x27, 8, 0x50      ; DATA XREF: animate_kong_and_pauline+74|o
385C 2F A7 08 50+      ; display_LUP+CB|o ...
385C 3B 25 08 50+      .db 0x2F, 0xA7, 8, 0x50
385C 00 70 08 48+      .db 0x3B, 0x25, 8, 0x50
385C 3B 23 07 40+      .db 0, 0x70, 8, 0x48
385C 46 A9 08 44+      .db 0x3B, 0x23, 7, 0x40
385C 00 70 08 48+      .db 0x46, 0xA9, 8, 0x44
385C 30 29 08 44+      .db 0, 0x70, 8, 0x48
385C 00 70 08 48+      .db 0x30, 0x29, 8, 0x44
385C 00 70 0A 48      .db 0, 0x70, 8, 0x48
385C      .db 0, 0x70, 0xA, 0x48
3884 6F 10 09 23+pauline_spr:      .db 0x6F, 0x10, 9, 0x23
3884 6F 11 0A 33      .db 0x6F, 0x11, 0xA, 0x33
388C 50 34 08 3C dk_climbing_spr:      .db 0x50, 0x34, 8, 0x3C      ; DATA XREF: display_LUP+6D|o
388C      ; 0000:168B|o ...
3890 00 35 08 3C      .db 0, 0x35, 8, 0x3C
3894 53 32 08 40      .db 0x53, 0x32, 8, 0x40
3898 63 33 08 40      .db 0x63, 0x33, 8, 0x40
389C 00 70 08 48      .db 0, 0x70, 8, 0x48
38A0 53 36 08 50      .db 0x53, 0x36, 8, 0x50
38A4 63 37 08 50      .db 0x63, 0x37, 8, 0x50

```

```

38A8 6B 31 08 41      .db 0x6B, 0x31, 8, 0x41
38AC 00 70 08 48      .db 0, 0x70, 8, 0x48
38B0 6A 14 0A 48      .db 0x6A, 0x14, 0xA, 0x48
38B4 FD FD FD+dk_intro_jump_up_data: .db 0xFD, 0xFD, 0xFD, 0xFD, 0xFD, 0xFD, 0xFD, 0xFE, 0xFE
38B4 FD FD FE+        ; DATA XREF: display_lup+56|o
38B4 FE FE FE+        .db 0xFE, 0xFE, 0xFE, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0
38B4 FE FF FF FF+      .db 0, 1, 1, 1, 0x7F
38CB FF FF FF FF+dk_intro_jump_left_data: .db 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0, 0xFF, 0, 0, 1, 0
38CB FF 00 FF 00+      ; DATA XREF: display_lup+5C|o
38CB 00 01 00 01+      ; 0000:0B86|o
38CB 01 01 01 01+      .db 1, 1, 1, 1, 1, 0x7F
38DC 04 7F F0 10+draw_data_bend_girders_2: .db 4, 0x7F, 0xF0, 0x10, 0xF0, 2, 0xDF, 0xF2, 0x70, 0xF8
38DC F0 02 DF F2+      ; DATA XREF: 0000:0B91|o
38DC 70 F8 02 6F+      .db 2, 0x6F, 0xF8, 0x10, 0xF8, 0xAA, 4, 0xDF, 0xD0, 0x90
38DC F8 10 F8 AA+      .db 0xD0, 2, 0xDF, 0xDC, 0x20, 0xD1, 0xAA, 0xFF, 0xFF
38DC 04 DF D0 90+      .db 0xFF, 0xFF, 0xFF, 4, 0xDF, 0xA8, 0x20, 0xA8, 4, 0x5F
38DC D0 02 DF DC+      .db 0xB0, 0x20, 0xB0, 2, 0xDF, 0xB0, 0x20, 0xBB, 0xAA
38DC 20 D1 AA FF+      .db 4, 0xDF, 0x88, 0x30, 0x88, 4, 0xDF, 0x90, 0xB0, 0x90
38DC FF FF FF FF+      .db 2, 0xDF, 0x9A, 0x20, 0x8F, 0xAA, 4, 0xBF, 0x68, 0x20
38DC 04 DF A8 20+      .db 0x68, 4, 0x3F, 0x70, 0x20, 0x70, 2, 0xDF, 0x6E, 0x20
38DC A8 04 5F B0+      .db 0x79, 0xAA
392C 02 DF 58 A0+draw_data_bend_girders_1: .db 2, 0xDF, 0x58, 0xA0, 0x55, 0xAA ; DATA XREF: 0000:0B48|o
3932 00 70 08 44+dk_throw_barrel_spr: .db 0, 0x70, 8, 0x44, 0x2B, 0xAC, 8, 0x4C, 0x3B, 0xAE
3932 2B AC 08 4C+      ; DATA XREF: 0000:1671|o
3932 3B AE 08 4C+      ; sub_0_2C8F+95|o
3932 3B AF 08 3C+      .db 8, 0x4C, 0x3B, 0xAF, 8, 0x3C, 0x4B, 0xB0, 7, 0x3C
3932 4B B0 07 3C+      .db 0x4B, 0xAD, 8, 0x4C, 0, 0x70, 8, 0x44, 0, 0x70, 8
3932 4B AD 08 4C+      .db 0x44, 0, 0x70, 8, 0x44, 0, 0x70, 0xA, 0x44, 0x47, 0x27
3932 00 70 08 44+      .db 8, 0x4C, 0x2F, 0xA7, 8, 0x4C, 0x3B, 0x25, 8, 0x4C
3932 00 70 08 44+      .db 0, 0x70, 8, 0x44, 0x3B, 0x23, 7, 0x3C, 0x4B, 0x2A
3932 00 70 08 44+      .db 8, 0x3C, 0x4B, 0x2B, 8, 0x4C, 0x2B, 0xAA, 8, 0x3C
3932 00 70 0A 44+      .db 0x2B, 0xAB, 8, 0x4C, 0, 0x70, 0xA, 0x44, 0, 0x70, 8
3932 47 27 08 4C+      .db 0x44, 0x4B, 0x2C, 8, 0x4C, 0x3B, 0x2E, 8, 0x4C, 0x3B
3932 2F A7 08 4C+      .db 0x2F, 8, 0x3C, 0x2B, 0x30, 7, 0x3C, 0x2B, 0x2D, 8
3932 3B 25 08 4C+      .db 0x4C, 0, 0x70, 8, 0x44, 0, 0x70, 8, 0x44, 0, 0x70
3932 00 70 08 44+      .db 8, 0x44, 0, 0x70, 0xA, 0x44
39AA FD FD FD FE+ bouncing_spring_data: .db 0xFD, 0xFD, 0xFD, 0xFE, 0xFE, 0xFE, 0xFE, 0xFF, 0xFF
39AA FE FE FE FF+      ; DATA XREF: sub_0_2E04+98|o
39AA FF 00 FF 00+      ; sub_0_2E04+C4|o
39AA 00 01 00 01+      .db 0, 0xFF, 0, 0, 1, 0, 1, 1, 2, 2, 2, 3, 3, 3, 0x7F
39C3 1E BB 4C+ barrel_rolling_data: .db 0x1E, 0x4E, 0xBB, 0x4C, 0xD8, 0x4E, 0x59, 0x4E, 0x7F
39C3 D8 4E 59 4E+      ; DATA XREF: sub_0_2C8F+FD|o
39CC BB 4D 7F barrel_falling_data: .db 0xBB, 0x4D, 0x7F ; DATA XREF: sub_0_2C8F+F4|o
39CF 47 27 08 50 dk_thrash_right_spr: .db 0x47, 0x27, 8, 0x50 ; DATA XREF: animate_kong_and_pauline+43|o
39CF ; 0000:0816|o
39D3 2D 26 08 50      .db 0x2D, 0x26, 8, 0x50
39D7 3B 25 08 50      .db 0x3B, 0x25, 8, 0x50
39DB 00 70 08 48      .db 0, 0x70, 8, 0x48
39DF 3B 24 07 40      .db 0x3B, 0x24, 7, 0x40
39E3 4B 28 08 40      .db 0x4B, 0x28, 8, 0x40
39E7 00 70 08 48      .db 0, 0x70, 8, 0x48
39EB 30 29 08 44      .db 0x30, 0x29, 8, 0x44
39EF 00 70 08 48      .db 0, 0x70, 8, 0x48
39F3 00 70 0A 48      .db 0, 0x70, 0xA, 0x48
39F7 49 A6 08 50 dk_thrash_left_spr: .db 0x49, 0xA6, 8, 0x50 ; DATA XREF: animate_kong_and_pauline+4A|o
39FB 2F A7 08 50      .db 0x2F, 0xA7, 8, 0x50
39FF 3B 25 08 50      .db 0x3B, 0x25, 8, 0x50
3A03 00 70 08 48      .db 0, 0x70, 8, 0x48
3A07 3B 24 07 40      .db 0x3B, 0x24, 7, 0x40
3A0B 46 A9 08 44      .db 0x46, 0xA9, 8, 0x44
3A0F 00 70 08 48      .db 0, 0x70, 8, 0x48
3A13 2B A8 08 40      .db 0x2B, 0xA8, 8, 0x40
3A17 00 70 08 48      .db 0, 0x70, 8, 0x48
3A1B 00 70 0A 48      .db 0, 0x70, 0xA, 0x48
3A1F 73 A7 88 60 fk_falling_spr: .db 0x73, 0xA7, 0x88, 0x60 ; DATA XREF: 0000:1870|o
3A23 8B 27 88 60      .db 0x8B, 0x27, 0x88, 0x60
3A27 7F 25 88 60      .db 0x7F, 0x25, 0x88, 0x60
3A2B 00 70 88 68      .db 0, 0x70, 0x88, 0x68
3A2F 7F 24 87 70      .db 0x7F, 0x24, 0x87, 0x70
3A33 74 29 88 6C      .db 0x74, 0x29, 0x88, 0x6C
3A37 00 70 88 68      .db 0, 0x70, 0x88, 0x68
3A3B 8A A9 88 6C      .db 0x8A, 0xA9, 0x88, 0x6C
3A3F 00 70 88 68      .db 0, 0x70, 0x88, 0x68
3A43 00 70 8A 68      .db 0, 0x70, 0x8A, 0x68
3A47 05 AF F0 50+draw_data_rivet_end1: .db 5, 0xAF, 0xF0, 0x50, 0xF0, 0xAA ; DATA XREF: 0000:17D9|o
3A4D 05 AF E8 50+draw_data_rivet_end2: .db 5, 0xAF, 0xE8, 0x50, 0xE8, 0xAA ; DATA XREF: 0000:17E5|o
3A53 05 AF E0 50+draw_data_rivet_end3: .db 5, 0xAF, 0xE0, 0x50, 0xE0, 0xAA ; DATA XREF: 0000:17F1|o
3A59 05 AF D8 50+draw_data_rivet_end4: .db 5, 0xAF, 0xD8, 0x50, 0xD8, 0xAA ; DATA XREF: 0000:17FD|o
3A5F 05 B7 58 48+draw_data_rivet_end5: .db 5, 0xB7, 0x58, 0x48, 0x58, 0xAA ; DATA XREF: 0000:18A5|o
3A65 01 04 01 03+level_seq_1: .db 1, 4, 1, 3, 4, 1, 2, 3, 4, 1, 2, 1, 3, 4 ; DATA XREF: 0000:095F|o
3A73 01 02 01 03+level_seq_2: .db 1, 2, 1, 3, 1, 4, 0x7F ; DATA XREF: 0000:1799|o
3A73 01 04 7F ; 0000:1947|o
3A7A FF 00 FF FF+fireball_bouncing_data: .db 0xFF, 0, 0xFF, 0xFF, 0xFE, 0xFE, 0xFE, 0xFE, 0xFE
3A7A FE FE FE FE+      ; DATA XREF: sub_0_3202+65|o
3A7A FE FE FE FE+      .db 0xFE, 0xFE, 0xFE, 0xFE, 0xFE, 0xFE, 0xFF, 0xFF, 0
3A8C E8 E5 E3 E2+fireball_bounce_data: .db 0xE8, 0xE5, 0xE3, 0xE2, 0xE1, 0xE0, 0xDF, 0xDE, 0xDD
3A8C E1 E0 DF DE+      ; DATA XREF: sub_0_342C+F|o
3A8C DD DD DC DC+      .db 0xDD, 0xDC, 0xDC, 0xDC, 0xDC, 0xDC, 0xDC, 0xDD, 0xDD
3A8C DC DC DC DC+      .db 0xDE, 0xDF, 0xE0, 0xE1, 0xE2, 0xE3, 0xE4, 0xE5, 0xE7
3A8C DD DE DF+        .db 0xE9, 0xEB, 0xED, 0xF0, 0xAA
3AAC 80 7B 78 76+cement_fireball_data: .db 0x80, 0x7B, 0x78, 0x76, 0x74, 0x73, 0x72, 0x71, 0x70
3AAC 74 73 72 71+      ; DATA XREF: sub_0_3478+F|o
3AAC 70 70 6F 6F+      .db 0x70, 0x6F, 0x6F, 0x6F, 0x70, 0x70, 0x71, 0x72, 0x73
3AAC 6F 70 70 71+      .db 0x74, 0x75, 0x76, 0x77, 0x78, 0xAA
3AC4 EE F0 DB A0+rivet_fireball_data: .db 0xEE, 0xF0, 0xDB, 0xA0, 0xE6, 0xC8, 0xD6, 0x78, 0xEB
3AC4 E6 C8 D6 78+      ; DATA XREF: sub_0_34B9+E|o
3AC4 EB F0 DB A0+      .db 0xF0, 0xDB, 0xA0, 0xE6, 0xC8, 0xE6, 0xC8
3AD4 1B C8 23 A0+rivet_fireball_start_points: .db 0x1B, 0xC8, 0x23, 0xA0, 0x2B, 0x78, 0x12, 0xF0, 0x1B
3AD4 2B 78 12 F0+      ; DATA XREF: sub_0_34B9+34|o
3AD4 1B C8 23 A0+      .db 0xC8, 0x23, 0xA0, 0x12, 0xF0, 0x1B, 0xC8
3AE4 02 97 38 68+barrel_level_tilemap_data: .db 2, 0x97, 0x38, 0x68, 0x38, 2, 0x9F, 0x54, 0x10, 0x54
3AE4 38 02 9F 54+      ; DATA XREF: 0000:0CD4|o
3AE4 10 54 02 DF+      ; sub_0_2441+19|o
3AE4 58 A0 55 02+      .db 2, 0xDF, 0x58, 0xA0, 0x55, 2, 0xEF, 0x6D, 0x20, 0x79
3AE4 EF 6D 20 79+      .db 2, 0xDF, 0x9A, 0x10, 0x8E, 2, 0xEF, 0xAF, 0x20, 0xBB
3AE4 02 DF 9A 10+      .db 2, 0xDF, 0xDC, 0x10, 0xD0, 2, 0xFF, 0xF0, 0x80, 0xF7
3AE4 8E 02 EF AF+      .db 2, 0x7F, 0xF8, 0, 0xF8, 0, 0xCB, 0x57, 0xCB, 0x6F
3AE4 20 BB 02 DF+      .db 0, 0xCB, 0x99, 0xCB, 0xB1, 0, 0xCB, 0xDB, 0xCB, 0xF3
3AE4 DC 10 D0 02+      .db 0, 0x63, 0x18, 0x63, 0x54, 1, 0x63, 0xD5, 0x63, 0xF8
3AE4 FF F0 80 F7+      .db 0, 0x33, 0x78, 0x33, 0x90, 0, 0x33, 0xBA, 0x33, 0xD2
3AE4 02 7F F8 00+      .db 0, 0x53, 0x18, 0x53, 0x54, 1, 0x53, 0x92, 0x53, 0xB8
3AE4 F8 00 CB 57+      .db 0, 0x5B, 0x76, 0x5B, 0x92, 0, 0x73, 0xB6, 0x73, 0xD6
3AE4 CB 6F 00 CB+      .db 0, 0x83, 0x95, 0x83, 0xB5, 0, 0x93, 0x38, 0x93, 0x54
3AE4 99 CB B1 00+      .db 1, 0xBB, 0x70, 0xBB, 0x98, 1, 0x6B, 0x54, 0x6B, 0x75
3AE4 CB DB CB F3+      .db 0xAA
3B5D 06 8F 90 70+cement_pie_level_tilemap_data: .db 6, 0x8F, 0x90, 0x70, 0x90, 6, 0x8F, 0x98, 0x70, 0x98

```

```

3B5D 90 06 8F 98+ ; DATA XREF: 0000:0CDF|o
3B5D 70 98 06 8F+ ; sub_0_2441+20|o
3B5D A0 70 A0 00+ .db 6, 0x8F, 0xA0, 0x70, 0xA0, 0, 0x63, 0x18, 0x63, 0x58
3B5D 63 18 63 58+ .db 0, 0x63, 0x80, 0x63, 0xA8, 0, 0x63, 0xD0, 0x63, 0xF8
3B5D 00 63 80 63+ .db 0, 0x53, 0x18, 0x53, 0x58, 0, 0x53, 0xA8, 0x53, 0xD0
3B5D A8 00 63 D0+ .db 0, 0x9B, 0x80, 0x9B, 0xA8, 0, 0x9B, 0xD0, 0x9B, 0xF8
3B5D 63 F8 00 53+ .db 1, 0x23, 0x58, 0x23, 0x80, 1, 0xDB, 0x58, 0xDB, 0x80
3B5D 18 53 58 00+ .db 0, 0x2B, 0x80, 0x2B, 0xA8, 0, 0xD3, 0x80, 0xD3, 0xA8
3B5D 53 A8 53 D0+ .db 0, 0xA3, 0xA8, 0xA3, 0xD0, 0, 0x2B, 0xD0, 0x2B, 0xF8
3B5D 00 9B 80 9B+ .db 0, 0xD3, 0xD0, 0xD3, 0xF8, 0, 0x93, 0x38, 0x93, 0x58
3B5D A8 00 9B D0+ .db 2, 0x97, 0x38, 0x68, 0x38, 3, 0xEF, 0x58, 0x10, 0x58
3B5D 9B F8 01 23+ .db 3, 0xF7, 0x80, 0x88, 0x80, 3, 0x77, 0x80, 8, 0x80
3B5D 58 23 80 01+ .db 2, 0xA7, 0xA8, 0x50, 0xA8, 2, 0xE7, 0xA8, 0xB8, 0xA8
3B5D DB 58 DB 80+ .db 2, 0x3F, 0xA8, 0x18, 0xA8, 3, 0xEF, 0xD0, 0x10, 0xD0
3B5D 00 2B 80 2B+ .db 2, 0xEF, 0xF8, 0x10, 0xF8, 0xAA
3BE5 00 63 18 63+elevator_level_tilemap_data:.db 0, 0x63, 0x18, 0x63, 0x58, 0, 0x63, 0x88, 0x63, 0xD0
3BE5 58 00 63 88+ ; DATA XREF: 0000:0CFA|o
3BE5 63 D0 00 53+ ; sub_0_2441+27|o
3BE5 18 53 58 00+ .db 0, 0x53, 0x18, 0x53, 0x58, 0, 0x53, 0x88, 0x53, 0xD0
3BE5 53 88 53 D0+ .db 0, 0xE3, 0x68, 0xE3, 0x90, 0, 0xE3, 0xB8, 0xE3, 0xD0
3BE5 00 E3 68 E3+ .db 0, 0xCB, 0x90, 0xCB, 0xB0, 0, 0xB3, 0x58, 0xB3, 0x78
3BE5 90 00 E3 B8+ .db 0, 0x9B, 0x80, 0x9B, 0xA0, 0, 0x93, 0x38, 0x93, 0x58
3BE5 E3 D0 00 CB+ .db 0, 0x23, 0x88, 0x23, 0xC0, 0, 0x1B, 0xC0, 0x1B, 0xE8
3BE5 90 CB B0 00+ .db 2, 0x97, 0x38, 0x68, 0x38, 2, 0xB7, 0x58, 0x10, 0x58
3BE5 B3 58 B3 78+ .db 2, 0xEF, 0x68, 0xE0, 0x68, 2, 0xD7, 0x70, 0xC8, 0x70
3BE5 00 9B 80 9B+ .db 2, 0xBF, 0x78, 0xB0, 0x78, 2, 0xA7, 0x80, 0x90, 0x80
3BE5 A0 00 93 38+ .db 2, 0x67, 0x88, 0x48, 0x88, 2, 0x27, 0x88, 0x10, 0x88
3BE5 93 58 00 23+ .db 2, 0xEF, 0x90, 0xC8, 0x90, 2, 0xA7, 0xA0, 0x98, 0xA0
3BE5 88 23 C0 00+ .db 2, 0xBF, 0xA8, 0xB0, 0xA8, 2, 0xD7, 0xB0, 0xC8, 0xB0
3BE5 1B C0 1B E8+ .db 2, 0xEF, 0xB8, 0xE0, 0xB8, 2, 0x27, 0xC0, 0x10, 0xC0
3BE5 02 97 38 68+ .db 2, 0xEF, 0xD0, 0xD8, 0xD0, 2, 0x67, 0xD0, 0x50, 0xD0
3BE5 38 02 B7 58+ .db 2, 0xCF, 0xD8, 0xC0, 0xD8, 2, 0xB7, 0xE0, 0xA8, 0xE0
3BE5 10 58 02 EF+ .db 2, 0x9F, 0xE8, 0x88, 0xE8, 2, 0x27, 0xE8, 0x10, 0xE8
3BE5 68 E0 68 02+ .db 2, 0xEF, 0xF8, 0x10, 0xF8, 0xAA
3C8B 00 7B 80 7B+rivet_level_tilemap_data:.db 0, 0x7B, 0x80, 0x7B, 0xA8, 0, 0x7B, 0xD0, 0x7B, 0xF8
3C8B A8 00 7B D0+ ; DATA XREF: 0000:0CC3|o
3C8B 7B F8 00 33+ ; sub_0_2441+2D|o
3C8B 58 53 80 00+ .db 0, 0x33, 0x58, 0x33, 0x80, 0, 0x53, 0x58, 0x53, 0x80
3C8B 53 58 53 80+ .db 0, 0xAB, 0x58, 0xAB, 0x80, 0, 0xCB, 0x58, 0xCB, 0x80
3C8B 00 AB 58 AB+ .db 0, 0x2B, 0x80, 0x2B, 0xA8, 0, 0xD3, 0x80, 0xD3, 0xA8
3C8B 80 00 CB 58+ .db 0, 0x23, 0xA8, 0x23, 0xD0, 0, 0x5B, 0xA8, 0x5B, 0xD0
3C8B CB 80 00 2B+ .db 0, 0xA3, 0xA8, 0xA3, 0xD0, 0, 0xDB, 0xA8, 0xDB, 0xD0
3C8B 80 2B A8 00+ .db 0, 0x1B, 0xD0, 0x1B, 0xF8, 0, 0xE3, 0xD0, 0xE3, 0xF8
3C8B D3 80 D3 A8+ .db 5, 0xB7, 0x30, 0x48, 0x30, 5, 0xCF, 0x58, 0x30, 0x58
3C8B 00 23 A8 23+ .db 5, 0xD7, 0x80, 0x28, 0x80, 5, 0xDF, 0xA8, 0x20, 0xA8
3C8B D0 00 5B A8+ .db 5, 0xE7, 0xD0, 0x18, 0xD0, 5, 0xEF, 0xF8, 0x10, 0xF8
3C8B 5B D0 00 A3+ .db 0xAA
3CF0 10 82 85 8B how_high_strings:.db 0x10, 0x82, 0x85, 0x8B ; DATA XREF: 0000:0C50|o
3CF0 ; " 25m"
3CF4 10 85 80 8B .db 0x10, 0x85, 0x80, 0x8B ; " 50m"
3CF8 10 87 85 8B .db 0x10, 0x87, 0x85, 0x8B ; "100m"
3CFC 81 80 80 8B .db 0x81, 0x80, 0x80, 0x8B ; "125m"
3D00 81 82 85 8B .db 0x81, 0x82, 0x85, 0x8B ; "150m"
3D04 81 85 80 8B .db 0x81, 0x85, 0x80, 0x8B
3D08 05 title_screen:.db 5 ; DATA XREF: 0000:07F7|o
3D08 ; RLE-encoded "DONKEY KONG" title
3D09 88 77 .dw VRAM_start+0x388
3D0B 01 .db 1
3D0C 68 77 .dw VRAM_start+0x368
3D0E 01 .db 1
3D0F 6C 77 .dw VRAM_start+0x36C
3D11 03 .db 3
3D12 49 77 .dw VRAM_start+0x349
3D14 05 .db 5
3D15 08 77 .dw VRAM_start+0x308
3D17 01 .db 1
3D18 E8 76 .dw VRAM_start+0x2E8
3D1A 01 .db 1
3D1B EC 76 .dw VRAM_start+0x2EC
3D1D 05 .db 5
3D1E C8 76 .dw VRAM_start+0x2C8
3D20 05 .db 5
3D21 88 76 .dw VRAM_start+0x288
3D23 02 .db 2
3D24 69 76 .dw VRAM_start+0x269
3D26 02 .db 2
3D27 4A 76 .dw VRAM_start+0x24A
3D29 05 .db 5
3D2A 28 76 .dw VRAM_start+0x228
3D2C 05 .db 5
3D2D E8 75 .dw VRAM_start+0x1E8
3D2F 01 .db 1
3D30 CA 75 .dw VRAM_start+0x1CA
3D32 03 .db 3
3D33 A9 75 .dw VRAM_start+0x1A9
3D35 01 .db 1
3D36 88 75 .dw VRAM_start+0x188
3D38 01 .db 1
3D39 8C 75 .dw VRAM_start+0x18C
3D3B 05 .db 5
3D3C 48 75 .dw VRAM_start+0x148
3D3E 01 .db 1
3D3F 28 75 .dw VRAM_start+0x128
3D41 01 .db 1
3D42 2A 75 .dw VRAM_start+0x12A
3D44 01 .db 1
3D45 2C 75 .dw VRAM_start+0x12C
3D47 01 .db 1
3D48 08 75 .dw VRAM_start+0x108
3D4A 01 .db 1
3D4B 0A 75 .dw VRAM_start+0x10A
3D4D 01 .db 1
3D4E 0C 75 .dw VRAM_start+0x10C
3D50 03 .db 3
3D51 C8 74 .dw VRAM_start+0xC8
3D53 03 .db 3
3D54 AA 74 .dw VRAM_start+0xAA
3D56 03 .db 3
3D57 88 74 .dw VRAM_start+0x88
3D59 05 .db 5
3D5A 2F 77 .dw VRAM_start+0x32F
3D5C 05 .db 5
3D5D 0F 77 .dw VRAM_start+0x30F
3D5F 02 .db 2
3D60 F0 76 .dw VRAM_start+0x2F0
3D62 02 .db 2

```

```

3D63 CF 76 .dw VRAM_start+0x2CF
3D65 02 .db 2
3D66 D2 76 .dw VRAM_start+0x2D2
3D68 05 .db 5
3D69 8F 76 .dw VRAM_start+0x28F
3D6B 05 .db 5
3D6C 6F 76 .dw VRAM_start+0x26F
3D6E 01 .db 1
3D6F 4F 76 .dw VRAM_start+0x24F
3D71 01 .db 1
3D72 53 76 .dw VRAM_start+0x253
3D74 05 .db 5
3D75 2F 76 .dw VRAM_start+0x22F
3D77 05 .db 5
3D78 EF 75 .dw VRAM_start+0x1EF
3D7A 02 .db 2
3D7B D0 75 .dw VRAM_start+0x1D0
3D7D 02 .db 2
3D7E B1 75 .dw VRAM_start+0x1B1
3D80 05 .db 5
3D81 8F 75 .dw VRAM_start+0x18F
3D83 03 .db 3
3D84 50 75 .dw VRAM_start+0x150
3D86 05 .db 5
3D87 2F 75 .dw VRAM_start+0x12F
3D89 01 .db 1
3D8A 0F 75 .dw VRAM_start+0x10F
3D8C 01 .db 1
3D8D 13 75 .dw VRAM_start+0x113
3D8F 01 .db 1
3D90 EF 74 .dw VRAM_start+0xEF
3D92 01 .db 1
3D93 F1 74 .dw VRAM_start+0xF1
3D95 01 .db 1
3D96 F3 74 .dw VRAM_start+0xF3
3D98 02 .db 2
3D99 D1 74 .dw VRAM_start+0xD1
3D9B 00 .db 0
3D9C 00 00 23 68+level_init_data: .db 0, 0, 0x23, 0x68, 1, 0x11, 0, 0, 0, 0x10, 0xDB, 0x68
3D9C 01 11 00 00+ ; DATA XREF: 0000:0F6F|o
3D9C 00 10 DB 68+ .db 1, 0x40, 0, 0, 8, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0
3D9C 01 40 00 00+ .db 0, 0, 0, 0, 0x80, 1, 0xC0, 0xFF, 1, 0xFF, 0xFF, 0x34
3D9C 08 01 01 01+ .db 0xC3, 0x39, 0, 0x67, 0x80, 0x69, 0x1A, 1, 0, 0, 0
3D9C 01 01 01 01+ .db 0, 0, 0, 0, 0, 4, 0, 0x10, 0, 0, 0, 0, 0
3DDC 1E 18 0B 4B+top_barrel_spr: .db 0x1E, 0x18, 0xB, 0x4B, 0x14, 0x18, 0xB, 0x4B, 0x1E
3DDC 14 18 0B 4B+ ; DATA XREF: 0000:0FD7|o
3DDC 1E 18 0B 3B+ .db 0x18, 0xB, 0x3B, 0x14, 0x18, 0xB, 0x3B
3DEC 3D 01 03 02 fireball_spr: .db 0x3D, 1, 3, 2 ; DATA XREF: 0000:0FE2|o
3DEC ; 0000:101F|o ...
3DF0 4D 01 04 01 rivet_fireball_spr: .db 0x4D, 1, 4, 1 ; DATA XREF: 0000:1131|o
3DF4 27 70 01 E0+girders_fireball_spr: .db 0x27, 0x70, 1, 0xE0, 0, 0 ; DATA XREF: 0000:0FEF|o
3DFA 7F 40 01 78+cement_fireball_spr: .db 0x7F, 0x40, 1, 0x78, 2, 0 ; DATA XREF: 0000:1049|o
3E00 27 49 0C F0 girder_oil_barrel_spr: .db 0x27, 0x49, 0xC, 0xF0 ; DATA XREF: 0000:0FF5|o
3E04 7F 49 0C 88 cement_oil_barrel_spr: .db 0x7F, 0x49, 0xC, 0x88 ; DATA XREF: 0000:104F|o
3E08 1E 07 03 09 hammer_pickup_spr: .db 0x1E, 7, 3, 9 ; DATA XREF: init_hammer_sprites+9|o
3E0C 24 64 BB C0 girder_hammer_locs: .db 0x24, 0x64, 0xBB, 0xC0 ; DATA XREF: 0000:1000|o
3E10 23 8D 7B B4 cement_hammer_locs: .db 0x23, 0x8D, 0x7B, 0xB4 ; DATA XREF: 0000:1070|o
3E14 1B 8C 7C 64 rivet_hammer_locs: .db 0x1B, 0x8C, 0x7C, 0x64 ; DATA XREF: 0000:113D|o
3E18 4B 0E 04 02 cement_pie_spr: .db 0x4B, 0xE, 4, 2 ; DATA XREF: 0000:102E|o
3E1C 23 46 03 68+cement_ladder_spr: .db 0x23, 0x46, 3, 0x68, 0xDB, 0x46, 3, 0x68 ; DATA XREF: 0000:105A|o
3E24 17 50 00 5C+cement_conveyor_spr: .db 0x17, 0x50, 0, 0x5C, 0xE7, 0xD0, 0, 0x5C, 0x8C, 0x50
3E24 E7 D0 00 5C+ ; DATA XREF: 0000:1065|o
3E24 8C 50 00 84+ .db 0, 0x84, 0x73, 0xD0, 0, 0x84, 0x17, 0x50, 0, 0xD4
3E24 73 D0 00 84+ .db 0xE7, 0xD0, 0, 0xD4
3E3C 53 73 0A A0+cement_obj_spr: .db 0x53, 0x73, 0xA, 0xA0, 0x8B, 0x74, 0xA, 0xF0, 0xDB
3E3C 8B 74 0A F0+ ; DATA XREF: 0000:1076|o
3E3C DB 75 0A A0 .db 0x75, 0xA, 0xA0
3E48 5B 73 0A C8+elevator_obj_spr: .db 0x5B, 0x73, 0xA, 0xC8, 0xE3, 0x74, 0xA, 0x60, 0x1B
3E48 E3 74 0A 60+ ; DATA XREF: 0000:10DE|o
3E48 1B 75 0A 80 .db 0x75, 0xA, 0x80
3E54 DB 73 0A C8+rivet_obj_spr: .db 0xDB, 0x73, 0xA, 0xC8, 0x93, 0x74, 0xA, 0xF0, 0x33
3E54 93 74 0A F0+ ; DATA XREF: 0000:1143|o
3E54 33 75 0A 50 .db 0x75, 0xA, 0x50
3E60 44 03 08 04 elevator_spr: .db 0x44, 3, 8, 4 ; DATA XREF: 0000:10C3|o
3E64 37 F4 37 C0+elevator_spr_locs: .db 0x37, 0xF4, 0x37, 0xC0, 0x37, 0x8C, 0x77, 0x70, 0x77
3E64 37 8C 77 70+ ; DATA XREF: 0000:10B7|o
3E64 77 A4 77 D8 .db 0xA4, 0x77, 0xD8
3E70 ;
3E70
3E70 loc_0_3E70: ; CODE XREF: check_and_handle_bonus+1A|j
3E70 11 01 00 ld de, #1
3E73
3E73 loc_0_3E73:
3E73 06 7B ld b, #0x7B ; '{'
3E75 1F rra
3E76 D2 28 1E jp NC, loc_0_1E28
3E79 1E 03 ld e, #3
3E7B 06 7D ld b, #0x7D ; '}'
3E7D 1F rra
3E7E D2 28 1E jp NC, loc_0_1E28
3E81 1E 05 ld e, #5
3E83 06 7F ld b, #0x7F ; ''
3E85 C3 28 1E jp loc_0_1E28
3E88
3E88 ; SUBROUTINE
3E88
3E88
3E88 sub_0_3E88: ; CODE XREF: sub_0_2853+18|p
3E88 3A 27 62 ld a, (level_type)
3E8B E5 push hl
3E8C EF rst 0x28 ; go!
3E8C ;
3E8D 00 00 .dw 0 ; Jump table
3E8F 99 3E .dw loc_0_3E99
3E91 B0 28 .dw l2_check_hammer_hit
3E93 E0 28 .dw l3_check_hammer_hit
3E95 01 29 .dw l4_check_hammer_hit
3E97 00 00 .dw 0
3E99 ;
3E99
3E99 loc_0_3E99: ; DATA XREF: sub_0_3E88+7|o
3E99 E1 pop hl
3E9A AF xor a
3E9B 32 60 60 ld (unk_0_6060), a
3E9E 06 0A ld b, #0xA
3EA0 11 20 00 ld de, #0x20 ; ''

```

```

3EA3 DD 21 00 67      ld      ix, #unk_0_6700
3EA7 CD C3 3E         call   sub_0_3EC3
3EAA 06 05           ld      b, #5
3EAC DD 21 00 64      ld      ix, #unk_0_6400      ; fireball character data
3EB0 CD C3 3E         call   sub_0_3EC3
3EB3 3A 60 60        ld      a, (unk_0_6060)
3EB6 A7             and     a
3EB7 C8             ret     Z
3EB8 FE 01          cp      #1
3EBA C8             ret     Z
3EBB FE 03          cp      #3
3EBD 3E 03          ld      a, #3
3EBF D8             ret     C
3EC0 3E 07          ld      a, #7
3EC2 C9             ret
3EC2             ; End of function sub_0_3E88
3EC2
3EC3
3EC3             ;
3EC3             SUBROUTINE
3EC3
3EC3             sub_0_3EC3:
3EC3             ; CODE XREF: sub_0_3E88+1F↑p
3EC3             ; sub_0_3E88+28↑p ...
3EC3 DD CB 00 46      bit      0, 0(ix)
3EC7 CA FA 3E        jp      Z, loc_0_3EFA
3ECA 79             ld      a, c
3ECB DD 96 05        sub     5(ix)
3ECE D2 D3 3E        jp      NC, loc_0_3ED3
3ED1 ED 44          neg
3ED3
3ED3             loc_0_3ED3:
3ED3             ; CODE XREF: sub_0_3EC3+B↑j
3ED3 3C             inc     a
3ED4 95             sub     1
3ED5 DA DE 3E        jp      C, loc_0_3EDE
3ED8 DD 96 0A        sub     0xA(ix)
3EDB D2 FA 3E        jp      NC, loc_0_3EFA
3EDE
3EDE             loc_0_3EDE:
3EDE             ; CODE XREF: sub_0_3EC3+12↑j
3EDE FD 7E 03        ld      a, 3(iy)
3EE1 DD 96 03        sub     3(ix)
3EE4 D2 E9 3E        jp      NC, loc_0_3EE9
3EE7 ED 44          neg
3EE9
3EE9             loc_0_3EE9:
3EE9             ; CODE XREF: sub_0_3EC3+21↑j
3EE9 94             sub     h
3EEA DA F3 3E        jp      C, loc_0_3EF3
3EED DD 96 09        sub     9(ix)
3EF0 D2 FA 3E        jp      NC, loc_0_3EFA
3EF3
3EF3             loc_0_3EF3:
3EF3             ; CODE XREF: sub_0_3EC3+27↑j
3EF3 3A 60 60        ld      a, (unk_0_6060)
3EF6 3C             inc     a
3EF7 32 60 60        ld      (unk_0_6060), a
3EFA
3EFA             loc_0_3EFA:
3EFA             ; CODE XREF: sub_0_3EC3+4↑j
3EFA             ; sub_0_3EC3+18↑j ...
3EFA DD 19          add     ix, de
3EFC 10 C5          djnz   sub_0_3EC3
3EFE C9             ret
3EFE             ; End of function sub_0_3EC3
3EFE
3EFE             ;
3EFF 00             .db     0 ;
3F00 5C 76          aCOPYRIGHT_1981: .dw VRAM_start+0x25C      ; DATA XREF: 0000:3687↑o
3F02 49 4A 01 09+   .db     0x49, 0x4A, 1, 9, 8, 1, 0x3F
3F09 7D 77          aNINTENDO_OF_AMERICA_INC: .dw VRAM_start+0x37D      ; DATA XREF: 0000:3689↑o
3F0B 1E 19 1E 24+   aNINTENDO: .db     0x1E, 0x19, 0x1E, 0x24, 0x15, 0x1E, 0x14, 0x1F, 0x10      ; DATA XREF: sub_0_2441↑o
3F0B 15 1E 14 1F+   .db     0x1F, 0x16, 0x10, 0x11, 0x1D, 0x15, 0x22, 0x19, 0x13
3F0B 10 1F 16 10+   .db     0x11, 0x10, 0x19, 0x1E, 0x13, 0x2B, 0x3F
3F0B 11 1D 15 22+   .db     0x11, 0x10, 0x19, 0x1E, 0x13, 0x2B, 0x3F
3F24
3F24             ;
3F24             SUBROUTINE
3F24
3F24             display_tm:
3F24             ; CODE XREF: 0000:081C↑p
3F24 21 AF 74        ld      hl, #VRAM_start+0xAF
3F27 11 E0 FF        ld      de, #0xFFE0
3F2A 36 9F          ld      (hl), #0x9F ; 'f'
3F2C 19            add     hl, de
3F2D 36 9E          ld      (hl), #0x9E ; 'x'
3F2F C9             ret
3F2F             ; End of function display_tm
3F2F
3F2F             ;
3F30 50 52 4F 47+   aProgramWeWouldTeachYou_Tel_tokyo: .ascii 'PROGRAM,WE WOULD TEACH YOU.*****TEL.TOKYO-JAPAN 044(244)'
3F30 52 41 4D 2C+   .ascii '2151 EXTENSION 304 SYSTEM DESIGN IKEGAMI CO. LIM.'
3FA0
3FA0             ;
3FA0             init_level_data_tmrs_spr:
3FA0             ; CODE XREF: 0000:0CD1↑j
3FA0 CD A6 3F        call   fix_retractable_ladders
3FA3 C3 5F 0D        jp      init_level_data_tmrs_spr_cont
3FA6
3FA6             ;
3FA6             SUBROUTINE
3FA6
3FA6             fix_retractable_ladders:
3FA6             ; CODE XREF: 0000:3FA0↑p
3FA6             ; ladders for cement pie level
3FA6             ; return if level bit not set
3FA8 3E 02          ld      a, #2
3FA8 F7             rst     0x30
3FA9 06 02          ld      b, #2
3FAB 21 6C 77        ld      hl, #VRAM_start+0x36C
3FAE
3FAE             loc_0_3FAE:
3FAE             ; CODE XREF: fix_retractable_ladders+11↑j
3FAE 36 10          ld      (hl), #0x10
3FB0 23            inc     hl
3FB1 23            inc     hl
3FB2 36 C0          ld      (hl), #0xC0 ; 'L'
3FB4 21 8C 74        ld      hl, #VRAM_start+0x8C
3FB7 10 F5          djnz   loc_0_3FAE
3FB9 C9             ret
3FB9             ; End of function fix_retractable_ladders
3FB9
3FB9             ;
3FB9             ;
3FBA 00 00 00 00+   .db     0, 0, 0, 0, 0, 0
3FC0
3FC0             ;
3FC0             SUBROUTINE
3FC0

```

```

3FC0
3FC0      sub_0_3FC0:
3FC0 21 4D 69      ld      hl, #soft_sprite_ram+0x4D      ; CODE XREF: 0000:2285|p
3FC3 36 03          ld      (hl), #3
3FC5 2C            inc      l
3FC6 2C            inc      l
3FC7 C9            ret
3FC7      ; End of function sub_0_3FC0
3FC7
3FC7      ;
3FC8 00 00 41 7F+   .db 0, 0, 0x41, 0x7F, 0x7F, 0x41, 0, 0, 0, 0x7F, 0x7F
3FC8 7F 41 00 00+   .db 0x18, 0x3C, 0x76, 0x63, 0x41, 0, 0, 0x7F, 0x7F, 0x49
3FC8 00 7F 7F 18+   .db 0x49, 0x49, 0x41, 0, 0x1C, 0x3E, 0x63, 0x41, 0x49
3FC8 3C 76 63 41+   .db 0x79, 0x79, 0, 0x7C, 0x7E, 0x13, 0x11, 0x13, 0x7E
3FC8 00 00 7F 7F+   .db 0x7C, 0, 0x7F, 0x7F, 0xE, 0x1C, 0xE, 0x7F, 0x7F, 0
3FC8 49 49 49 41+   .db 0, 0x41, 0x7F, 0x7F, 0x41, 0, 0
3FC8 00 1C 3E 63+; end of 'ROM'
3FC8 41 49 79 79+
6000      ;
6000
6000      ; Segment type: Regular
6000      ; segment 'RAM'
6000      .org 0x6000
6000 ??      RAM_start: .ds 1      ; DATA XREF: 0000:0268|o
6001 ??      no_of_credits: .ds 1      ; DATA XREF: display_credits+5|o
6001      ; 0000:073F|r ...
6002 ??      .ds 1
6003 ??      coin_state: .ds 1      ; DATA XREF: check_coin_inserted+5|o
6004 ??      .ds 1
6005 ??      nmi_sequencer: .ds 1      ; DATA XREF: 0000:00C6|r
6005      ; check_coin_inserted+12|r ...
6006 ??      .ds 1
6007 ??      attract_mode_flag: .ds 1
6008 ??      sixteen_bit_countdown_msb: .ds 1      ; DATA XREF: return_NOT_16bit_timeout|o
6009*??      eight_bit_countdown: .ds 1      ; DATA XREF: return_NOT_8bit_timeout|o
6009*      ; 0000:078E|o ...
600A ??      main_sequencer: .ds 1      ; DATA XREF: 0000:01EE|w
600A      ; 0000:06FE|r ...
600B ??      .ds 1
600C ??      .ds 1
600D ??      current_player_D: .ds 1
600E ??      current_player_E: .ds 1
600F ??      two_players: .ds 1
6010 ??      controller_in: .ds 1      ; DATA XREF: 0000:00AC|w
6010      ; 0000:1502|r ...
6011 ??      last_raw_in: .ds 1
6012 ??      .ds 1
6013 ??      .ds 1
6014 ??      .ds 1
6015 ??      .ds 1
6016 ??      .ds 1
6017 ??      .ds 1
6018 ?? ??      random_no: .ds 2      ; DATA XREF: rand|r
6018      ; rand+B|w ...
601A ??      gen_purpose_timer: .ds 1      ; DATA XREF: rand+3|o
601A      ; 0000:00B5|o ...
601B ??      .ds 1
601C ??      .ds 1
601D ??      .ds 1
601E ??      .ds 1
601F ??      .ds 1
6020 ??      lives_per_game: .ds 1      ; DATA XREF: read_dips_and_high_score_tbl+4|o
6020      ; 0000:0922|r ...
6021 ??      bonus_setting: .ds 1      ; DATA XREF: check_and_award_bonus+1E|o
6021      ; 7/10/15/20K
6022 ?? ?? ?? ?? coinage: .ds 4      ; DATA XREF: check_coin_inserted+27|o
6026 ??      upright: .ds 1      ; DATA XREF: 0000:0087|r
6026      ; 0000:099F|r ...
6027 ??      .ds 1
6028 ??      .ds 1
6029 ??      .ds 1
602A ??      .ds 1
602B ??      .ds 1
602C ??      .ds 1
602D ??      .ds 1
602E ??      .ds 1
602F ??      .ds 1
6030 ??      unk_0_6030: .ds 1      ; DATA XREF: 0000:1499|o
6030      ; 0000:14FC|o
6031*??      byte_0_6031: .ds 1      ; DATA XREF: 0000:1591|r
6031*      ; 0000:159A|w ...
6032*??      byte_0_6032: .ds 1      ; DATA XREF: 0000:158A|o
6032*      ; 0000:15B2|w
6033 ??      regi_second_cntr: .ds 1
6034 ??      regi_vblank_cntr: .ds 1      ; DATA XREF: 0000:14DC|o
6035 ??      regi_current_char: .ds 1
6036*?? ??      regi_entry_cursor_loc: .ds 2      ; DATA XREF: 0000:14B0|w
6036*      ; 0000:1553|r ...
6038 ?? ??      regi_ptr_hs_entry_flag: .ds 2      ; DATA XREF: 0000:14C9|w
6038      ; 0000:15A0|r ...
603A ?? ??      regi_ptr_hs_entry_name: .ds 2      ; DATA XREF: 0000:14D0|w
603A      ; 0000:15D8|r
603C ??      .ds 1
603D ??      .ds 1
603E ??      .ds 1
603F ??      .ds 1
6040 ??      p1_ingame_data: .ds 1      ; DATA XREF: 0000:093E|o
6040      ; 0000:09AB|o ...
6040      ; game init data copied here
6041 ??      .ds 1
6042 ?? ??      .ds 2      ; ptr sequence data
6044 ??      .ds 1
6045 ??      .ds 1
6046 ??      .ds 1
6047 ??      .ds 1
6048 ??      p2_ingame_data: .ds 1      ; DATA XREF: 0000:0909|o
6048      ; 0000:091F|o ...
6049 ??      .ds 1
604A ??      .ds 1
604B ??      .ds 1
604C ??      .ds 1
604D ??      .ds 1
604E ??      .ds 1
604F ??      .ds 1
6050 ??      .ds 1
6051 ??      .ds 1

```

```

6052 ??                .ds 1
6053 ??                .ds 1
6054 ??                .ds 1
6055 ??                .ds 1
6056 ??                .ds 1
6057 ??                .ds 1
6058 ??                .ds 1
6059 ??                .ds 1
605A ??                .ds 1
605B ??                .ds 1
605C ??                .ds 1
605D ??                .ds 1
605E ??                .ds 1
605F ??                .ds 1
6060 ??    unk_0_6060: .ds 1
6061 ??                .ds 1
6062 ??                .ds 1
6063 ??                .ds 1
6064 ??                .ds 1
6065 ??                .ds 1
6066 ??                .ds 1
6067 ??                .ds 1
6068 ??                .ds 1
6069 ??                .ds 1
606A ??                .ds 1
606B ??                .ds 1
606C ??                .ds 1
606D ??                .ds 1
606E ??                .ds 1
606F ??                .ds 1
6070 ??                .ds 1
6071 ??                .ds 1
6072 ??                .ds 1
6073 ??                .ds 1
6074 ??                .ds 1
6075 ??                .ds 1
6076 ??                .ds 1
6077 ??                .ds 1
6078 ??                .ds 1
6079 ??                .ds 1
607A ??                .ds 1
607B ??                .ds 1
607C ??                .ds 1
607D ??                .ds 1
607E ??                .ds 1
607F ??                .ds 1
6080 ??    digital_snd_tmr_walk:.ds 1    ; DATA XREF: update_sounds!o
6080                                ; stop_sound+6!o ...
6081 ??    digital_snd_tmr_jump:.ds 1    ; DATA XREF: sub_0_1AC3+E9!o
6082 ??    digital_snd_tmr_thump:.ds 1    ; DATA XREF: animate_kong_and_pauline+52!w
6082                                ; 0000:0B45!w ...
6083 ??    digital_snd_tmr_coin_spring:.ds 1
6084 ??    digital_snd_tmr_kong_fall:.ds 1
6085 ??    digital_snd_tmr_barrel_jump_priz:.ds 1    ; DATA XREF: check_and_handle_bonus+25!o
6086 ??    digital_snd_tmr_6:.ds 1    ; check_and_handle_bonus+87!o ...
6087 ??    digital_snd_tmr_7:.ds 1
6088 ??    music_something:.ds 1    ; DATA XREF: update_sounds+2E!o
6088                                ; 0000:12A8!w
6089 ??    bg_music:      .ds 1    ; DATA XREF: 0000:067A!w
6089                                ; 0000:0CC0!w ...
608A ??    unk_0_608A:    .ds 1    ; DATA XREF: display_1UP+88!o
608A                                ; 0000:0BB3!o ...
608B ??    unk_0_608B:    .ds 1    ; DATA XREF: update_sounds+1A!o
608C ??                .ds 1
608D ??                .ds 1
608E ??                .ds 1
608F ??                .ds 1
6090 ??                .ds 1
6091 ??                .ds 1
6092 ??                .ds 1
6093 ??                .ds 1
6094 ??                .ds 1
6095 ??                .ds 1
6096 ??                .ds 1
6097 ??                .ds 1
6098 ??                .ds 1
6099 ??                .ds 1
609A ??                .ds 1
609B ??                .ds 1
609C ??                .ds 1
609D ??                .ds 1
609E ??                .ds 1
609F ??                .ds 1
60A0 ??                .ds 1
60A1 ??                .ds 1
60A2 ??                .ds 1
60A3 ??                .ds 1
60A4 ??                .ds 1
60A5 ??                .ds 1
60A6 ??                .ds 1
60A7 ??                .ds 1
60A8 ??                .ds 1
60A9 ??                .ds 1
60AA ??                .ds 1
60AB ??                .ds 1
60AC ??                .ds 1
60AD ??                .ds 1
60AE ??                .ds 1
60AF ??                .ds 1
60B0 ??    fg_fn_queue_tail:.ds 1
60B1 ??    fg_fn_queue_head:.ds 1
60B2 ?? ?? ??    p1_score: .ds 3    ; DATA XREF: 0000:01C9!o
60B2                                ; current_player_score_DE!o ...
60B5 ?? ?? ??    p2_score: .ds 3    ; DATA XREF: current_player_score_DE+8!o
60B5                                ; zero_score_or_high_score+D!o ...
60B8 ?? ?? ??    high_score: .ds 3    ; DATA XREF: add_bonus_and_update_high_score+37!o
60B8                                ; zero_score_or_high_score+15!o ...
60BB ??                .ds 1
60BC ??                .ds 1
60BD ??                .ds 1
60BE ??                .ds 1
60BF ??                .ds 1
60C0 ?? ?? ?? ??+fg_vector_fn_params:.ds 0x40    ; DATA XREF: 0000:0291!o
60C0 ?? ?? ?? ??+                                ; queue_fg_vector_fn+1!o
6100 ?? ?? ?? ??+high_score_tbl_ram:.ds 0x22    ; DATA XREF: read_dips_and_high_score_tbl+56!o

```



```

6100 ?? ?? ?? ??+ ; 0000:3673|o
6100 ?? ?? ?? ??+ ; 1st
6122 ?? ?? ?? ??+hs_tbl_2nd: .ds 0x22 ; DATA XREF: 0000:3675|o
6122 ?? ?? ?? ??+ ; 2nd
6144 ?? ?? ?? ??+hs_tbl_3rd: .ds 0x22 ; DATA XREF: 0000:3677|o
6144 ?? ?? ?? ??+ ; 3rd
6166 ?? ?? ?? ??+hs_tbl_4th: .ds 0x22 ; DATA XREF: 0000:3679|o
6166 ?? ?? ?? ??+ ; 4th
6188 ?? ?? ?? ??+hs_tbl_5th: .ds 0x22 ; DATA XREF: 0000:367B|o
6188 ?? ?? ?? ??+ ; 5th
61AA ?? .ds 1
61AB ?? .ds 1
61AC ?? .ds 1
61AD ?? .ds 1
61AE ?? .ds 1
61AF ?? .ds 1
61B0 ?? .ds 1
61B1 ?? unk_0_61B1: .ds 1 ; DATA XREF: sub_0_13CA+D|o
61B2 ?? .ds 1
61B3 ?? .ds 1
61B4 ?? .ds 1
61B5 ?? .ds 1
61B6 ?? .ds 1
61B7 ?? .ds 1
61B8 ?? .ds 1
61B9 ?? .ds 1
61BA ?? .ds 1
61BB ?? .ds 1
61BC ?? .ds 1
61BD ?? .ds 1
61BE ?? .ds 1
61BF ?? .ds 1
61C0 ?? .ds 1
61C1 ?? .ds 1
61C2 ?? .ds 1
61C3 ?? .ds 1
61C4 ?? .ds 1
61C5 ?? .ds 1
61C6 ?? unk_0_61C6: .ds 1 ; DATA XREF: sub_0_13CA|o
61C7 ?? unk_0_61C7: .ds 1 ; DATA XREF: sub_0_13CA+2F|o
61C8 ?? .ds 1
61C9 ?? .ds 1
61CA ?? .ds 1
61CB ?? .ds 1
61CC ?? .ds 1
61CD ?? .ds 1
61CE ?? .ds 1
61CF ?? .ds 1
61D0 ?? .ds 1
61D1 ?? .ds 1
61D2 ?? .ds 1
61D3 ?? .ds 1
61D4 ?? .ds 1
61D5 ?? .ds 1
61D6 ?? .ds 1
61D7 ?? .ds 1
61D8 ?? .ds 1
61D9 ?? .ds 1
61DA ?? .ds 1
61DB ?? .ds 1
61DC ?? .ds 1
61DD ?? .ds 1
61DE ?? .ds 1
61DF ?? .ds 1
61E0 ?? .ds 1
61E1 ?? .ds 1
61E2 ?? .ds 1
61E3 ?? .ds 1
61E4 ?? .ds 1
61E5 ?? .ds 1
61E6 ?? .ds 1
61E7 ?? .ds 1
61E8 ?? .ds 1
61E9 ?? .ds 1
61EA ?? .ds 1
61EB ?? .ds 1
61EC ?? .ds 1
61ED ?? .ds 1
61EE ?? .ds 1
61EF ?? .ds 1
61F0 ?? .ds 1
61F1 ?? .ds 1
61F2 ?? .ds 1
61F3 ?? .ds 1
61F4 ?? .ds 1
61F5 ?? .ds 1
61F6 ?? .ds 1
61F7 ?? .ds 1
61F8 ?? .ds 1
61F9 ?? .ds 1
61FA ?? .ds 1
61FB ?? .ds 1
61FC ?? .ds 1
61FD ?? .ds 1
61FE ?? .ds 1
61FF ?? .ds 1
6200 ?? mario_alive_flag: .ds 1 ; DATA XREF: return_if_mario_not_alive|r
6200 ; 0000:0BE3|r ...
6201 ?? .ds 1
6202 ?? mario_animation_cell: .ds 1
6203 ?? mario_y: .ds 1 ; DATA XREF: animate_kong_and_pauline+D6|r
6203 ; animate_kong_and_pauline+10E|r ...
6204 ?? unk_0_6204: .ds 1
6205 ?? mario_x: .ds 1 ; DATA XREF: sub_0_19DA+13|r
6205 ; sub_0_1A33+22|r ...
6206 ?? unk_0_6206: .ds 1
6207 ?? mario_flipx_tile: .ds 1 ; DATA XREF: sub_0_1AC3+54|o
6207 ; sub_0_1AC3+9D|o ...
6208 ?? mario_flipx_colour: .ds 1
6209 ?? unk_0_6209: .ds 1 ; DATA XREF: 0000:0FA5|o
6209 ; init to 4
620A ?? .ds 1 ; init to 8
620B ?? mario_x_before_jump: .ds 1
620C ?? mario_y_before_jump: .ds 1
620D ?? .ds 1
620E ?? unk_0_620E: .ds 1 ; DATA XREF: sub_0_1AC3+E6|w

```

```

620E      ; sub_0_1AC3+1B6[o] ...
620F ??      mario_cell_animate_cntr:.ds 1      ; DATA XREF: sub_0_1AC3+1CE[r]
620F      ; sub_0_1AC3+1EA[r] ...
620F      ; mario_???
6210 ??      unk_0_6210:      .ds 1      ; DATA XREF: sub_0_1AC3+B0[o]
6210      ; sub_0_1F46+F[w] ...
6211 ??      unk_0_6211:      .ds 1
6212 ??      unk_0_6212:      .ds 1
6213 ??      unk_0_6213:      .ds 1
6214 ??      unk_0_6214:      .ds 1
6215 ??      mario_climbing: .ds 1
6216 ??      mario_jumping:  .ds 1
6217 ??      unk_0_6217:      .ds 1
6218 ??      unk_0_6218:      .ds 1
6219 ??      unk_0_6219:      .ds 1
621A ??      unk_0_621A:      .ds 1      ; DATA XREF: sub_0_1AC3+5D[o]
621A      ; sub_0_1AC3+2B3[r] ...
621B ??      .ds 1
621C ??      unk_0_621C:      .ds 1      ; DATA XREF: sub_0_1AC3+262[o]
621C      ; sub_0_1AC3+2BD[r]
621D ??      .ds 1
621E ??      unk_0_621E:      .ds 1      ; DATA XREF: sub_0_1AC3+7[r]
621E      ; sub_0_1AC3+92[o] ...
621F ??      unk_0_621F:      .ds 1
6220 ??      unk_0_6220:      .ds 1
6221 ??      unk_0_6221:      .ds 1
6222 ??      unk_0_6222:      .ds 1
6223 ??      .ds 1
6224 ??      unk_0_6224:      .ds 1
6225 ??      unk_0_6225:      .ds 1
6226 ??      .ds 1
6227 ??      level_type:      .ds 1      ; DATA XREF: sub_0_30+14[o]
6227      ; 0000:01EA[w] ...
6228 ??      lives_left:      .ds 1      ; DATA XREF: 0000:01D9[w]
6228      ; check_and_award_bonus+28[o] ...
6229 ??      level:          .ds 1      ; DATA XREF: 0000:01D6[w]
6229      ; difficulty_timer_tick+15[r] ...
6229      ; keeps incrementing
622A ?? ??    seq_data:      .ds 2
622C ??      seen_intro:     .ds 1      ; DATA XREF: display_1UP+1B[o]
622C      ; 0000:12F6[w] ...
622D ??      awarded_bonus_life:.ds 1
622E ??      height:        .ds 1      ; DATA XREF: 0000:0C05[r]
622E      ; 0000:0C0E[w] ...
622F ??      last_seq_lsb:   .ds 1
6230 ??      .ds 1
6231 ??      .ds 1
6232 ??      .ds 1
6233 ??      .ds 1
6234 ??      .ds 1
6235 ??      .ds 1
6236 ??      .ds 1
6237 ??      .ds 1
6238 ??      .ds 1
6239 ??      .ds 1
623A ??      .ds 1
623B ??      .ds 1
623C ??      .ds 1
623D ??      .ds 1
623E ??      .ds 1
623F ??      .ds 1
6240 ??      .ds 1
6241 ??      .ds 1
6242 ??      .ds 1
6243 ??      .ds 1
6244 ??      .ds 1
6245 ??      .ds 1
6246 ??      .ds 1
6247 ??      .ds 1
6248 ??      .ds 1
6249 ??      .ds 1
624A ??      .ds 1
624B ??      .ds 1
624C ??      .ds 1
624D ??      .ds 1
624E ??      .ds 1
624F ??      .ds 1
6250 ??      .ds 1
6251 ??      .ds 1
6252 ??      .ds 1
6253 ??      .ds 1
6254 ??      .ds 1
6255 ??      .ds 1
6256 ??      .ds 1
6257 ??      .ds 1
6258 ??      .ds 1
6259 ??      .ds 1
625A ??      .ds 1
625B ??      .ds 1
625C ??      .ds 1
625D ??      .ds 1
625E ??      .ds 1
625F ??      .ds 1
6260 ??      .ds 1
6261 ??      .ds 1
6262 ??      .ds 1
6263 ??      .ds 1
6264 ??      .ds 1
6265 ??      .ds 1
6266 ??      .ds 1
6267 ??      .ds 1
6268 ??      .ds 1
6269 ??      .ds 1
626A ??      .ds 1
626B ??      .ds 1
626C ??      .ds 1
626D ??      .ds 1
626E ??      .ds 1
626F ??      .ds 1
6270 ??      .ds 1
6271 ??      .ds 1
6272 ??      .ds 1
6273 ??      .ds 1
6274 ??      .ds 1
6275 ??      .ds 1
6276 ??      .ds 1

```

```

6277 ?? .ds 1
6278 ?? .ds 1
6279 ?? .ds 1
627A ?? .ds 1
627B ?? .ds 1
627C ?? .ds 1
627D ?? .ds 1
627E ?? .ds 1
627F ?? .ds 1
6280 ?? unk_0_6280: .ds 1 ; DATA XREF: 0000:0F64!o
6280 ; 0000:0F72!o ...
6281 ?? .ds 1
6282 ?? .ds 1
6283 ?? .ds 1
6284 ?? .ds 1
6285 ?? .ds 1
6286 ?? .ds 1
6287 ?? .ds 1
6288 ?? unk_0_6288: .ds 1 ; DATA XREF: sub_0_2207+E!o
6289 ?? .ds 1
628A ?? .ds 1
628B ?? .ds 1
628C ?? .ds 1
628D ?? .ds 1
628E ?? .ds 1
628F ?? .ds 1
6290 ?? unk_0_6290: .ds 1 ; DATA XREF: sub_0_1A33+53!o
6290 ; sub_0_1E57+29!r
6291 ?? unk_0_6291: .ds 1
6292 ?? unk_0_6292: .ds 1 ; DATA XREF: sub_0_1A33+48!o
6293 ?? .ds 1
6294 ?? .ds 1
6295 ?? .ds 1
6296 ?? .ds 1
6297 ?? .ds 1
6298 ?? .ds 1
6299 ?? .ds 1
629A ?? .ds 1
629B ?? .ds 1
629C ?? .ds 1
629D ?? .ds 1
629E ?? .ds 1
629F ?? .ds 1
62A0 ?? unk_0_62A0: .ds 1 ; DATA XREF: 0000:16BC!w
62A0 ; 0000:16D2!w ...
62A1 ?? unk_0_62A1: .ds 1 ; DATA XREF: sub_0_2602+14!o
62A2 ?? .ds 1
62A3 ?? unk_0_62A3: .ds 1 ; DATA XREF: sub_0_2523+2E!r
62A3 ; sub_0_262F!o ...
62A4 ?? .ds 1
62A5 ?? unk_0_62A5: .ds 1 ; DATA XREF: sub_0_2679+7!o
62A6 ?? unk_0_62A6: .ds 1 ; DATA XREF: sub_0_2523+39!r
62A6 ; sub_0_2679+14!o
62A7 ?? unk_0_62A7: .ds 1 ; DATA XREF: sub_0_27DA!o
62A8 ?? unk_0_62A8: .ds 1
62A9 ?? .ds 1
62AA ?? unk_0_62AA: .ds 1
62AB ?? .ds 1
62AC ?? unk_0_62AC: .ds 1
62AD ?? .ds 1
62AE ?? .ds 1
62AF*?? byte_0_62AF: .ds 1 ; DATA XREF: display_1UP+53!w
62AF* ; display_1UP+98!r ...
62B0 ?? bonus_timer_init_value: .ds 1 ; DATA XREF: 0000:063A!r
62B0 ; 0000:0F8E!o ...
62B0 ; level timer #1
62B1 ?? unk_0_62B1: .ds 1 ; DATA XREF: sub_0_2C03+9!r
62B1 ; sub_0_2C8F+4B!o ...
62B1 ; level timer #2
62B2 ?? unk_0_62B2: .ds 1 ; level timer #3
62B3 ?? unk_0_62B3: .ds 1 ; level timer #4
62B4 ?? unk_0_62B4: .ds 1 ; DATA XREF: sub_0_2FCB+3!o
62B4 ; level timer #5
62B5 ?? .ds 1
62B6 ?? .ds 1
62B7 ?? .ds 1
62B8 ?? unk_0_62B8: .ds 1 ; DATA XREF: sub_0_3A2+9!o
62B9 ?? unk_0_62B9: .ds 1
62BA ?? unk_0_62BA: .ds 1 ; DATA XREF: sub_0_3A2+2F!o
62BA ; sub_0_3A2+3E!w
62BB ?? .ds 1
62BC ?? .ds 1
62BD ?? .ds 1
62BE ?? .ds 1
62BF ?? .ds 1
62C0 ?? .ds 1
62C1 ?? .ds 1
62C2 ?? .ds 1
62C3 ?? .ds 1
62C4 ?? .ds 1
62C5 ?? .ds 1
62C6 ?? .ds 1
62C7 ?? .ds 1
62C8 ?? .ds 1
62C9 ?? .ds 1
62CA ?? .ds 1
62CB ?? .ds 1
62CC ?? .ds 1
62CD ?? .ds 1
62CE ?? .ds 1
62CF ?? .ds 1
62D0 ?? .ds 1
62D1 ?? .ds 1
62D2 ?? .ds 1
62D3 ?? .ds 1
62D4 ?? .ds 1
62D5 ?? .ds 1
62D6 ?? .ds 1
62D7 ?? .ds 1
62D8 ?? .ds 1
62D9 ?? .ds 1
62DA ?? .ds 1
62DB ?? .ds 1
62DC ?? .ds 1
62DD ?? .ds 1
62DE ?? .ds 1

```

```
62DF ?? .ds 1
62E0 ?? .ds 1
62E1 ?? .ds 1
62E2 ?? .ds 1
62E3 ?? .ds 1
62E4 ?? .ds 1
62E5 ?? .ds 1
62E6 ?? .ds 1
62E7 ?? .ds 1
62E8 ?? .ds 1
62E9 ?? .ds 1
62EA ?? .ds 1
62EB ?? .ds 1
62EC ?? .ds 1
62ED ?? .ds 1
62EE ?? .ds 1
62EF ?? .ds 1
62F0 ?? .ds 1
62F1 ?? .ds 1
62F2 ?? .ds 1
62F3 ?? .ds 1
62F4 ?? .ds 1
62F5 ?? .ds 1
62F6 ?? .ds 1
62F7 ?? .ds 1
62F8 ?? .ds 1
62F9 ?? .ds 1
62FA ?? .ds 1
62FB ?? .ds 1
62FC ?? .ds 1
62FD ?? .ds 1
62FE ?? .ds 1
62FF ?? .ds 1
6300 ?? unk_0_6300: .ds 1 ; DATA XREF: sub_0_236E|o
6300 ; sub_0_2441+30|o
6301 ?? .ds 1
6302 ?? .ds 1
6303 ?? .ds 1
6304 ?? .ds 1
6305 ?? .ds 1
6306 ?? .ds 1
6307 ?? .ds 1
6308 ?? .ds 1
6309 ?? .ds 1
630A ?? .ds 1
630B ?? .ds 1
630C ?? .ds 1
630D ?? .ds 1
630E ?? .ds 1
630F ?? .ds 1
6310 ?? unk_0_6310: .ds 1 ; DATA XREF: sub_0_2441+B|o
6311 ?? .ds 1
6312 ?? .ds 1
6313 ?? .ds 1
6314 ?? .ds 1
6315 ?? .ds 1
6316 ?? .ds 1
6317 ?? .ds 1
6318 ?? .ds 1
6319 ?? .ds 1
631A ?? .ds 1
631B ?? .ds 1
631C ?? .ds 1
631D ?? .ds 1
631E ?? .ds 1
631F ?? .ds 1
6320 ?? .ds 1
6321 ?? .ds 1
6322 ?? .ds 1
6323 ?? .ds 1
6324 ?? .ds 1
6325 ?? .ds 1
6326 ?? .ds 1
6327 ?? .ds 1
6328 ?? .ds 1
6329 ?? .ds 1
632A ?? .ds 1
632B ?? .ds 1
632C ?? .ds 1
632D ?? .ds 1
632E ?? .ds 1
632F ?? .ds 1
6330 ?? .ds 1
6331 ?? .ds 1
6332 ?? .ds 1
6333 ?? .ds 1
6334 ?? .ds 1
6335 ?? .ds 1
6336 ?? .ds 1
6337 ?? .ds 1
6338 ?? .ds 1
6339 ?? .ds 1
633A ?? .ds 1
633B ?? .ds 1
633C ?? .ds 1
633D ?? .ds 1
633E ?? .ds 1
633F ?? .ds 1
6340 ?? show_bonus_state: .ds 1
6341 ?? show_bonus_timer: .ds 1 ; DATA XREF: check_and_handle_bonus+E|w
6341 ; 0000:1E4A|o
6342 ?? unk_0_6342: .ds 1
6343 ?? unk_0_6343: .ds 1
6344 ?? .ds 1
6345 ?? unk_0_6345: .ds 1 ; DATA XREF: sub_0_1E96|r
6345 ; sub_0_1E96+60|o
6346 ?? unk_0_6346: .ds 1 ; DATA XREF: 0000:1F09|o
6346 ; 0000:1F23|o
6347 ?? .ds 1
6348 ?? unk_0_6348: .ds 1
6349 ?? .ds 1
634A ?? .ds 1
634B ?? .ds 1
634C ?? .ds 1
634D ?? .ds 1
634E ?? .ds 1
```

```

634F ??          unk_0_6350:      .ds 1
6350 ??          unk_0_6351:      .ds 1
6351 ??          unk_0_6352:      .ds 1
6352 ??          unk_0_6353:      .ds 1
6353 ??          unk_0_6354:      .ds 1
6354 ??          unk_0_6354:      .ds 1
6355 ??          .ds 1
6356 ??          .ds 1
6357 ??          .ds 1
6358 ??          .ds 1
6359 ??          .ds 1
635A ??          .ds 1
635B ??          .ds 1
635C ??          .ds 1
635D ??          .ds 1
635E ??          .ds 1
635F ??          .ds 1
6360 ??          .ds 1
6361 ??          .ds 1
6362 ??          .ds 1
6363 ??          .ds 1
6364 ??          .ds 1
6365 ??          .ds 1
6366 ??          .ds 1
6367 ??          .ds 1
6368 ??          .ds 1
6369 ??          .ds 1
636A ??          .ds 1
636B ??          .ds 1
636C ??          .ds 1
636D ??          .ds 1
636E ??          .ds 1
636F ??          .ds 1
6370 ??          .ds 1
6371 ??          .ds 1
6372 ??          .ds 1
6373 ??          .ds 1
6374 ??          .ds 1
6375 ??          .ds 1
6376 ??          .ds 1
6377 ??          .ds 1
6378 ??          .ds 1
6379 ??          .ds 1
637A ??          .ds 1
637B ??          .ds 1
637C ??          .ds 1
637D ??          .ds 1
637E ??          .ds 1
637F ??          .ds 1
6380 ??          unk_0_6380:      .ds 1
6381 ??          unk_0_6381:      .ds 1
6382 ??          unk_0_6382:      .ds 1
6383 ??          unk_0_6383:      .ds 1
6384 ??          unk_0_6384:      .ds 1
6385 ??          intro_sequencer: .ds 1
6385          ; DATA XREF: difficulty_timer_tick+7[o]
6386 ??          unk_0_6386:      .ds 1
6387 ??          unk_0_6387:      .ds 1
6388 ??          unk_0_6388:      .ds 1
6388          ; DATA XREF: 0000:161F[r]
6389 ??          unk_0_6389:      .ds 1
638A ??          title_flash_tmr_1: .ds 1
638A          ; DATA XREF: 0000:1633[r] ...
638B ??          title_flash_tmr_2: .ds 1
638C ??          bonus_timer:      .ds 1
638D ??          next_girder_to_deform: .ds 1
638D          ; DATA XREF: 0000:0B58[w]
638E*??          byte_0_638E:      .ds 1
638E*          ; 0000:0B94[r] ...
638F ??          unk_0_638F:      .ds 1
638F          ; DATA XREF: display_1UP+81[w]
6390 ??          kong_thrash_tmr: .ds 1
6390          ; 0000:0B3B[r] ...
6391 ??          kong_thrash_flag: .ds 1
6392 ??          unk_0_6392:      .ds 1
6393 ??          barrel_deployment: .ds 1
6394 ??          unk_0_6394:      .ds 1
6394          ; DATA XREF: sub_0_2C03+4C[w]
6395 ??          unk_0_6395:      .ds 1
6395          ; sub_0_2C8F+8D[r] ...
6396 ??          unk_0_6396:      .ds 1
6397 ??          .ds 1
6398 ??          mario_on_elevator: .ds 1
6399 ??          .ds 1
639A ??          unk_0_639A:      .ds 1
639B ??          unk_0_639B:      .ds 1
639B          ; DATA XREF: animate_kong_and_pauline+2B[o]
639C ??          .ds 1
639D ??          mario_death_state: .ds 1
639D          ; animate_kong_and_pauline+8B[r] ...
639E ??          death_spin_counter: .ds 1
639E          ; DATA XREF: sub_0_2523[o]
639F ??          .ds 1
63A0 ??          unk_0_63A0:      .ds 1
63A0          ; sub_0_2ED4+75[o]
63A1 ??          unk_0_63A1:      .ds 1
63A2 ??          unk_0_63A2:      .ds 1
63A3 ??          unk_0_63A3:      .ds 1
63A4 ??          unk_0_63A4:      .ds 1
63A5 ??          unk_0_63A5:      .ds 1
63A6 ??          unk_0_63A6:      .ds 1
63A7 ??          height_counter: .ds 1
63A7          ; DATA XREF: sub_0_2ED4+7C[o]
63A8 ??          disp_loc_for_height_string: .ds 2
63A8          ; sub_0_2ED4+87[w] ...
63AA ??          .ds 1
63AB*??          segment_addr_1: .ds 2
63AB*          ; DATA XREF: sub_0_2ED4+4A[r]
63AD*??          segment_addr_2: .ds 2
63AD*          ; sub_0_2ED4+75[o]
63AF*??          start_tile_index: .ds 1
63AF*          ; DATA XREF: sub_0_2ED4+7C[o]
63B0*??          end_tile_index: .ds 1
63B0*          ; sub_0_2ED4+87[w] ...
63B1*??          dY:                .ds 1
63B1*          ; DATA XREF: sub_0_2523+65[w]
63B2*??          dX:                .ds 1
63B2*          ; DATA XREF: sub_0_3A2+39[w]
63B2*          ; 0000:0768[w] ...
63B2*          ; DATA XREF: draw_level_background+14[w]
63B2*          ; draw_level_background+5E[r] ...
63B2*          ; DATA XREF: draw_level_background+41[w]
63B2*          ; draw_level_background+88[r]
63B2*          ; DATA XREF: draw_level_background+20[w]
63B2*          ; draw_level_background+52[r] ...
63B2*          ; DATA XREF: draw_level_background+39[w]
63B2*          ; draw_level_background+83[r] ...
63B2*          ; DATA XREF: draw_level_background+2C[w]
63B2*          ; draw_level_background+D5[r] ...
63B2*          ; DATA XREF: draw_level_background+33[w]

```

```

63B2*      segment_type: .ds 1      ; draw_level_background+4C|r ...
63B3*??    ; DATA XREF: draw_level_background+1|w
63B3*      ; draw_level_background+44|r ...
63B4*??    tile_byte_1: .ds 1      ; DATA XREF: draw_level_background+1A|w
63B5*??    current_tile_in_segment: .ds 1 ; DATA XREF: draw_level_background+B5|w
63B5*      ; draw_level_background+BB|r ...
63B6 ??    .ds 1
63B7 ??    unk_0_63B7: .ds 1
63B8 ??    bonus_timer_expired: .ds 1
63B8      ; DATA XREF: 0000:0635|r
63B8      ; 0000:06AC|o
63B9 ??    unk_0_63B9: .ds 1
63BA ??    .ds 1
63BB ??    .ds 1
63BC ??    .ds 1
63BD ??    .ds 1
63BE ??    .ds 1
63BF ??    .ds 1
63C0*?? ?? ptr_current_sequence: .ds 2      ; DATA XREF: display_LUP+AF|w
63C0*      ; 0000:0B64|w ...
63C2*?? ?? ptr_current_jump_up_data: .ds 2    ; DATA XREF: display_LUP+59|w
63C2*      ; display_LUP+B8|r ...
63C4*?? ?? ptr_current_jump_left_data: .ds 2  ; DATA XREF: display_LUP+5F|w
63C4*      ; 0000:0B6D|r ...
63C6 ??    .ds 1
63C7 ??    .ds 1
63C8 ??    unk_0_63C8: .ds 1
63C9 ??    .ds 1
63CA ??    .ds 1
63CB ??    .ds 1
63CC ??    attract_movement_entry: .ds 1      ; DATA XREF: next_attract_action+3|o
63CD ??    .ds 1
63CE ??    .ds 1
63CF ??    .ds 1
63D0 ??    .ds 1
63D1 ??    .ds 1
63D2 ??    .ds 1
63D3 ??    .ds 1
63D4 ??    .ds 1
63D5 ??    .ds 1
63D6 ??    .ds 1
63D7 ??    .ds 1
63D8 ??    .ds 1
63D9 ??    .ds 1
63DA ??    .ds 1
63DB ??    .ds 1
63DC ??    .ds 1
63DD ??    .ds 1
63DE ??    .ds 1
63DF ??    .ds 1
63E0 ??    unk_0_63E0: .ds 1      ; DATA XREF: sub_0_31B1+7|o
63E1 ??    .ds 1
63E2 ??    .ds 1
63E3 ??    .ds 1
63E4 ??    .ds 1
63E5 ??    .ds 1
63E6 ??    .ds 1
63E7 ??    .ds 1
63E8 ??    .ds 1
63E9 ??    .ds 1
63EA ??    .ds 1
63EB ??    .ds 1
63EC ??    .ds 1
63ED ??    .ds 1
63EE ??    .ds 1
63EF ??    .ds 1
63F0 ??    .ds 1
63F1 ??    .ds 1
63F2 ??    .ds 1
63F3 ??    .ds 1
63F4 ??    .ds 1
63F5 ??    .ds 1
63F6 ??    .ds 1
63F7 ??    .ds 1
63F8 ??    .ds 1
63F9 ??    .ds 1
63FA ??    .ds 1
63FB ??    .ds 1
63FC ??    .ds 1
63FD ??    .ds 1
63FE ??    .ds 1
63FF ??    .ds 1
6400 ??    unk_0_6400: .ds 1      ; DATA XREF: 0000:10E9|o
6400      ; sub_0_286F+2A|o ...
6401 ??    .ds 1
6402 ??    .ds 1
6403 ??    .ds 1
6404 ??    .ds 1
6405 ??    .ds 1
6406 ??    .ds 1
6407 ??    unk_0_6407: .ds 1      ; DATA XREF: 0000:0FE5|o
6407      ; 0000:1022|o ...
6408 ??    .ds 1
6409 ??    .ds 1
640A ??    .ds 1
640B ??    .ds 1
640C ??    .ds 1
640D ??    .ds 1
640E ??    .ds 1
640F ??    .ds 1
6410 ??    .ds 1
6411 ??    .ds 1
6412 ??    .ds 1
6413 ??    .ds 1
6414 ??    .ds 1
6415 ??    .ds 1
6416 ??    .ds 1
6417 ??    .ds 1
6418 ??    .ds 1
6419 ??    .ds 1
641A ??    .ds 1
641B ??    .ds 1
641C ??    .ds 1
641D ??    .ds 1
641E ??    .ds 1
641F ??    .ds 1
6420 ??    .ds 1

```

6421	??	.ds	1	
6422	??	.ds	1	
6423	??	.ds	1	
6424	??	.ds	1	
6425	??	.ds	1	
6426	??	.ds	1	
6427	??	.ds	1	
6428	??	.ds	1	
6429	??	.ds	1	
642A	??	.ds	1	
642B	??	.ds	1	
642C	??	.ds	1	
642D	??	.ds	1	
642E	??	.ds	1	
642F	??	.ds	1	
6430	??	.ds	1	
6431	??	.ds	1	
6432	??	.ds	1	
6433	??	.ds	1	
6434	??	.ds	1	
6435	??	.ds	1	
6436	??	.ds	1	
6437	??	.ds	1	
6438	??	.ds	1	
6439	unk_0_6439:	.ds	1	; DATA XREF: sub_0_31DD+C!o
643A	??	.ds	1	
643B	??	.ds	1	
643C	??	.ds	1	
643D	??	.ds	1	
643E	??	.ds	1	
643F	??	.ds	1	
6440	??	.ds	1	
6441	??	.ds	1	
6442	??	.ds	1	
6443	??	.ds	1	
6444	??	.ds	1	
6445	??	.ds	1	
6446	??	.ds	1	
6447	??	.ds	1	
6448	??	.ds	1	
6449	??	.ds	1	
644A	??	.ds	1	
644B	??	.ds	1	
644C	??	.ds	1	
644D	??	.ds	1	
644E	??	.ds	1	
644F	??	.ds	1	
6450	??	.ds	1	
6451	??	.ds	1	
6452	??	.ds	1	
6453	??	.ds	1	
6454	??	.ds	1	
6455	??	.ds	1	
6456	??	.ds	1	
6457	??	.ds	1	
6458	??	.ds	1	
6459	??	.ds	1	
645A	??	.ds	1	
645B	??	.ds	1	
645C	??	.ds	1	
645D	??	.ds	1	
645E	??	.ds	1	
645F	??	.ds	1	
6460	??	.ds	1	
6461	??	.ds	1	
6462	??	.ds	1	
6463	??	.ds	1	
6464	??	.ds	1	
6465	??	.ds	1	
6466	??	.ds	1	
6467	??	.ds	1	
6468	??	.ds	1	
6469	??	.ds	1	
646A	??	.ds	1	
646B	??	.ds	1	
646C	??	.ds	1	
646D	??	.ds	1	
646E	??	.ds	1	
646F	??	.ds	1	
6470	??	.ds	1	
6471	??	.ds	1	
6472	??	.ds	1	
6473	??	.ds	1	
6474	??	.ds	1	
6475	??	.ds	1	
6476	??	.ds	1	
6477	??	.ds	1	
6478	??	.ds	1	
6479	unk_0_6479:	.ds	1	; DATA XREF: sub_0_31DD+12!o
647A	??	.ds	1	
647B	??	.ds	1	
647C	??	.ds	1	
647D	??	.ds	1	
647E	??	.ds	1	
647F	??	.ds	1	
6480	??	.ds	1	
6481	??	.ds	1	
6482	??	.ds	1	
6483	??	.ds	1	
6484	??	.ds	1	
6485	??	.ds	1	
6486	??	.ds	1	
6487	??	.ds	1	
6488	??	.ds	1	
6489	??	.ds	1	
648A	??	.ds	1	
648B	??	.ds	1	
648C	??	.ds	1	
648D	??	.ds	1	
648E	??	.ds	1	
648F	??	.ds	1	
6490	??	.ds	1	
6491	??	.ds	1	
6492	??	.ds	1	
6493	??	.ds	1	
6494	??	.ds	1	

```

6495 ?? .ds 1
6496 ?? .ds 1
6497 ?? .ds 1
6498 ?? .ds 1
6499 ?? .ds 1
649A ?? .ds 1
649B ?? .ds 1
649C ?? .ds 1
649D ?? .ds 1
649E ?? .ds 1
649F ?? .ds 1
64A0 ?? unk_0_64A0: .ds 1 ; DATA XREF: 0000:1166|o
64A1 ?? .ds 1
64A2 ?? .ds 1
64A3 ?? unk_0_64A3: .ds 1 ; DATA XREF: 0000:1151|o
64A4 ?? .ds 1
64A5 ?? .ds 1
64A6 ?? .ds 1
64A7 ?? unk_0_64A7: .ds 1 ; DATA XREF: 0000:115D|o
64A8 ?? .ds 1
64A9 ?? .ds 1
64AA ?? .ds 1
64AB ?? .ds 1
64AC ?? .ds 1
64AD ?? .ds 1
64AE ?? .ds 1
64AF ?? .ds 1
64B0 ?? .ds 1
64B1 ?? .ds 1
64B2 ?? .ds 1
64B3 ?? .ds 1
64B4 ?? .ds 1
64B5 ?? .ds 1
64B6 ?? .ds 1
64B7 ?? .ds 1
64B8 ?? .ds 1
64B9 ?? .ds 1
64BA ?? .ds 1
64BB ?? .ds 1
64BC ?? .ds 1
64BD ?? .ds 1
64BE ?? .ds 1
64BF ?? .ds 1
64C0 ?? .ds 1
64C1 ?? .ds 1
64C2 ?? .ds 1
64C3 ?? .ds 1
64C4 ?? .ds 1
64C5 ?? .ds 1
64C6 ?? .ds 1
64C7 ?? .ds 1
64C8 ?? .ds 1
64C9 ?? .ds 1
64CA ?? .ds 1
64CB ?? .ds 1
64CC ?? .ds 1
64CD ?? .ds 1
64CE ?? .ds 1
64CF ?? .ds 1
64D0 ?? .ds 1
64D1 ?? .ds 1
64D2 ?? .ds 1
64D3 ?? .ds 1
64D4 ?? .ds 1
64D5 ?? .ds 1
64D6 ?? .ds 1
64D7 ?? .ds 1
64D8 ?? .ds 1
64D9 ?? .ds 1
64DA ?? .ds 1
64DB ?? .ds 1
64DC ?? .ds 1
64DD ?? .ds 1
64DE ?? .ds 1
64DF ?? .ds 1
64E0 ?? .ds 1
64E1 ?? .ds 1
64E2 ?? .ds 1
64E3 ?? .ds 1
64E4 ?? .ds 1
64E5 ?? .ds 1
64E6 ?? .ds 1
64E7 ?? .ds 1
64E8 ?? .ds 1
64E9 ?? .ds 1
64EA ?? .ds 1
64EB ?? .ds 1
64EC ?? .ds 1
64ED ?? .ds 1
64EE ?? .ds 1
64EF ?? .ds 1
64F0 ?? .ds 1
64F1 ?? .ds 1
64F2 ?? .ds 1
64F3 ?? .ds 1
64F4 ?? .ds 1
64F5 ?? .ds 1
64F6 ?? .ds 1
64F7 ?? .ds 1
64F8 ?? .ds 1
64F9 ?? .ds 1
64FA ?? .ds 1
64FB ?? .ds 1
64FC ?? .ds 1
64FD ?? .ds 1
64FE ?? .ds 1
64FF ?? .ds 1
6500 ?? unk_0_6500: .ds 1 ; DATA XREF: init_spring_sprites+C|o
6500 ; 0000:28F9|o ...
6501 ?? .ds 1
6502 ?? .ds 1
6503 ?? .ds 1
6504 ?? .ds 1
6505 ?? .ds 1
6506 ?? .ds 1
6507 ?? unk_0_6507: .ds 1 ; DATA XREF: init_spring_sprites+3|o

```



6508	??	.ds	1
6509	??	.ds	1
650A	??	.ds	1
650B	??	.ds	1
650C	??	.ds	1
650D	??	.ds	1
650E	??	.ds	1
650F	??	.ds	1
6510	??	.ds	1
6511	??	.ds	1
6512	??	.ds	1
6513	??	.ds	1
6514	??	.ds	1
6515	??	.ds	1
6516	??	.ds	1
6517	??	.ds	1
6518	??	.ds	1
6519	??	.ds	1
651A	??	.ds	1
651B	??	.ds	1
651C	??	.ds	1
651D	??	.ds	1
651E	??	.ds	1
651F	??	.ds	1
6520	??	.ds	1
6521	??	.ds	1
6522	??	.ds	1
6523	??	.ds	1
6524	??	.ds	1
6525	??	.ds	1
6526	??	.ds	1
6527	??	.ds	1
6528	??	.ds	1
6529	??	.ds	1
652A	??	.ds	1
652B	??	.ds	1
652C	??	.ds	1
652D	??	.ds	1
652E	??	.ds	1
652F	??	.ds	1
6530	??	.ds	1
6531	??	.ds	1
6532	??	.ds	1
6533	??	.ds	1
6534	??	.ds	1
6535	??	.ds	1
6536	??	.ds	1
6537	??	.ds	1
6538	??	.ds	1
6539	??	.ds	1
653A	??	.ds	1
653B	??	.ds	1
653C	??	.ds	1
653D	??	.ds	1
653E	??	.ds	1
653F	??	.ds	1
6540	??	.ds	1
6541	??	.ds	1
6542	??	.ds	1
6543	??	.ds	1
6544	??	.ds	1
6545	??	.ds	1
6546	??	.ds	1
6547	??	.ds	1
6548	??	.ds	1
6549	??	.ds	1
654A	??	.ds	1
654B	??	.ds	1
654C	??	.ds	1
654D	??	.ds	1
654E	??	.ds	1
654F	??	.ds	1
6550	??	.ds	1
6551	??	.ds	1
6552	??	.ds	1
6553	??	.ds	1
6554	??	.ds	1
6555	??	.ds	1
6556	??	.ds	1
6557	??	.ds	1
6558	??	.ds	1
6559	??	.ds	1
655A	??	.ds	1
655B	??	.ds	1
655C	??	.ds	1
655D	??	.ds	1
655E	??	.ds	1
655F	??	.ds	1
6560	??	.ds	1
6561	??	.ds	1
6562	??	.ds	1
6563	??	.ds	1
6564	??	.ds	1
6565	??	.ds	1
6566	??	.ds	1
6567	??	.ds	1
6568	??	.ds	1
6569	??	.ds	1
656A	??	.ds	1
656B	??	.ds	1
656C	??	.ds	1
656D	??	.ds	1
656E	??	.ds	1
656F	??	.ds	1
6570	??	.ds	1
6571	??	.ds	1
6572	??	.ds	1
6573	??	.ds	1
6574	??	.ds	1
6575	??	.ds	1
6576	??	.ds	1
6577	??	.ds	1
6578	??	.ds	1
6579	??	.ds	1
657A	??	.ds	1
657B	??	.ds	1

```
657C ?? .ds 1
657D ?? .ds 1
657E ?? .ds 1
657F ?? .ds 1
6580 ?? .ds 1
6581 ?? .ds 1
6582 ?? .ds 1
6583 ?? .ds 1
6584 ?? .ds 1
6585 ?? .ds 1
6586 ?? .ds 1
6587 ?? .ds 1
6588 ?? .ds 1
6589 ?? .ds 1
658A ?? .ds 1
658B ?? .ds 1
658C ?? .ds 1
658D ?? .ds 1
658E ?? .ds 1
658F ?? .ds 1
6590 ?? .ds 1
6591 ?? .ds 1
6592 ?? .ds 1
6593 ?? .ds 1
6594 ?? .ds 1
6595 ?? .ds 1
6596 ?? .ds 1
6597 ?? .ds 1
6598 ?? .ds 1
6599 ?? .ds 1
659A ?? .ds 1
659B ?? .ds 1
659C ?? .ds 1
659D ?? .ds 1
659E ?? .ds 1
659F ?? .ds 1
65A0 ?? unk_0_65A0: .ds 1 ; DATA XREF: 0000:103A|o
65A0 ; sub_0_24EA+9|o ...
65A1 ?? .ds 1
65A2 ?? .ds 1
65A3 ?? .ds 1
65A4 ?? .ds 1
65A5 ?? .ds 1
65A6 ?? .ds 1
65A7 ?? unk_0_65A7: .ds 1 ; DATA XREF: 0000:1031|o
65A8 ?? .ds 1
65A9 ?? .ds 1
65AA ?? .ds 1
65AB ?? .ds 1
65AC ?? .ds 1
65AD ?? .ds 1
65AE ?? .ds 1
65AF ?? .ds 1
65B0 ?? .ds 1
65B1 ?? .ds 1
65B2 ?? .ds 1
65B3 ?? .ds 1
65B4 ?? .ds 1
65B5 ?? .ds 1
65B6 ?? .ds 1
65B7 ?? .ds 1
65B8 ?? .ds 1
65B9 ?? .ds 1
65BA ?? .ds 1
65BB ?? .ds 1
65BC ?? .ds 1
65BD ?? .ds 1
65BE ?? .ds 1
65BF ?? .ds 1
65C0 ?? .ds 1
65C1 ?? .ds 1
65C2 ?? .ds 1
65C3 ?? .ds 1
65C4 ?? .ds 1
65C5 ?? .ds 1
65C6 ?? .ds 1
65C7 ?? .ds 1
65C8 ?? .ds 1
65C9 ?? .ds 1
65CA ?? .ds 1
65CB ?? .ds 1
65CC ?? .ds 1
65CD ?? .ds 1
65CE ?? .ds 1
65CF ?? .ds 1
65D0 ?? .ds 1
65D1 ?? .ds 1
65D2 ?? .ds 1
65D3 ?? .ds 1
65D4 ?? .ds 1
65D5 ?? .ds 1
65D6 ?? .ds 1
65D7 ?? .ds 1
65D8 ?? .ds 1
65D9 ?? .ds 1
65DA ?? .ds 1
65DB ?? .ds 1
65DC ?? .ds 1
65DD ?? .ds 1
65DE ?? .ds 1
65DF ?? .ds 1
65E0 ?? .ds 1
65E1 ?? .ds 1
65E2 ?? .ds 1
65E3 ?? .ds 1
65E4 ?? .ds 1
65E5 ?? .ds 1
65E6 ?? .ds 1
65E7 ?? .ds 1
65E8 ?? .ds 1
65E9 ?? .ds 1
65EA ?? .ds 1
65EB ?? .ds 1
65EC ?? .ds 1
65ED ?? .ds 1
65EE ?? .ds 1
```

```
65EF ?? .ds 1
65F0 ?? .ds 1
65F1 ?? .ds 1
65F2 ?? .ds 1
65F3 ?? .ds 1
65F4 ?? .ds 1
65F5 ?? .ds 1
65F6 ?? .ds 1
65F7 ?? .ds 1
65F8 ?? .ds 1
65F9 ?? .ds 1
65FA ?? .ds 1
65FB ?? .ds 1
65FC ?? .ds 1
65FD ?? .ds 1
65FE ?? .ds 1
65FF ?? .ds 1
6600 ?? unk_0_6600: .ds 1 ; DATA XREF: 0000:1096|o
; 0000:10CF|o ...
6601 ?? .ds 1
6602 ?? .ds 1
6603 ?? unk_0_6603: .ds 1 ; DATA XREF: 0000:10BA|o
6604 ?? .ds 1
6605 ?? .ds 1
6606 ?? .ds 1
6607 ?? unk_0_6607: .ds 1 ; DATA XREF: 0000:10C6|o
6608 ?? .ds 1
6609 ?? .ds 1
660A ?? .ds 1
660B ?? .ds 1
660C ?? .ds 1
660D ?? unk_0_660D: .ds 1 ; DATA XREF: 0000:10AA|o
660E ?? .ds 1
660F ?? .ds 1
6610 ?? .ds 1
6611 ?? .ds 1
6612 ?? .ds 1
6613 ?? .ds 1
6614 ?? .ds 1
6615 ?? .ds 1
6616 ?? .ds 1
6617 ?? .ds 1
6618 ?? .ds 1
6619 ?? .ds 1
661A ?? .ds 1
661B ?? .ds 1
661C ?? .ds 1
661D ?? .ds 1
661E ?? .ds 1
661F ?? .ds 1
6620 ?? .ds 1
6621 ?? .ds 1
6622 ?? .ds 1
6623 ?? .ds 1
6624 ?? .ds 1
6625 ?? .ds 1
6626 ?? .ds 1
6627 ?? .ds 1
6628 ?? .ds 1
6629 ?? .ds 1
662A ?? .ds 1
662B ?? .ds 1
662C ?? .ds 1
662D ?? .ds 1
662E ?? .ds 1
662F ?? .ds 1
6630 ?? .ds 1
6631 ?? .ds 1
6632 ?? .ds 1
6633 ?? .ds 1
6634 ?? .ds 1
6635 ?? .ds 1
6636 ?? .ds 1
6637 ?? .ds 1
6638 ?? .ds 1
6639 ?? .ds 1
663A ?? .ds 1
663B ?? .ds 1
663C ?? .ds 1
663D ?? .ds 1
663E ?? .ds 1
663F ?? .ds 1
6640 ?? .ds 1
6641 ?? .ds 1
6642 ?? .ds 1
6643 ?? .ds 1
6644 ?? .ds 1
6645 ?? .ds 1
6646 ?? .ds 1
6647 ?? .ds 1
6648 ?? .ds 1
6649 ?? .ds 1
664A ?? .ds 1
664B ?? .ds 1
664C ?? .ds 1
664D ?? .ds 1
664E ?? .ds 1
664F ?? .ds 1
6650 ?? .ds 1
6651 ?? .ds 1
6652 ?? .ds 1
6653 ?? .ds 1
6654 ?? .ds 1
6655 ?? .ds 1
6656 ?? .ds 1
6657 ?? .ds 1
6658 ?? .ds 1
6659 ?? .ds 1
665A ?? .ds 1
665B ?? .ds 1
665C ?? .ds 1
665D ?? .ds 1
665E ?? .ds 1
665F ?? .ds 1
6660 ?? .ds 1
6661 ?? .ds 1
```

```

6662 ?? .ds 1
6663 ?? .ds 1
6664 ?? .ds 1
6665 ?? .ds 1
6666 ?? .ds 1
6667 ?? .ds 1
6668 ?? .ds 1
6669 ?? .ds 1
666A ?? .ds 1
666B ?? .ds 1
666C ?? .ds 1
666D ?? .ds 1
666E ?? .ds 1
666F ?? .ds 1
6670 ?? .ds 1
6671 ?? .ds 1
6672 ?? .ds 1
6673 ?? .ds 1
6674 ?? .ds 1
6675 ?? .ds 1
6676 ?? .ds 1
6677 ?? .ds 1
6678 ?? .ds 1
6679 ?? .ds 1
667A ?? .ds 1
667B ?? .ds 1
667C ?? .ds 1
667D ?? .ds 1
667E ?? .ds 1
667F ?? .ds 1
6680 ?? unk_0_6680: .ds 1 ; DATA XREF: init_hammer_sprites+15!o
; sub_0_281D+5!o ...
6681 ?? .ds 1
6682 ?? .ds 1
6683 ?? unk_0_6683: .ds 1 ; DATA XREF: init_hammer_sprites!o
6684 ?? .ds 1
6685 ?? .ds 1
6686 ?? .ds 1
6687 ?? unk_0_6687: .ds 1 ; DATA XREF: init_hammer_sprites+C!o
6688 ?? .ds 1
6689 ?? .ds 1
668A ?? .ds 1
668B ?? .ds 1
668C ?? .ds 1
668D ?? .ds 1
668E ?? .ds 1
668F ?? .ds 1
6690 ?? unk_0_6690: .ds 1 ; DATA XREF: sub_0_2ED4+15!o
6691 ?? .ds 1
6692 ?? .ds 1
6693 ?? .ds 1
6694 ?? .ds 1
6695 ?? .ds 1
6696 ?? .ds 1
6697 ?? .ds 1
6698 ?? .ds 1
6699 ?? .ds 1
669A ?? .ds 1
669B ?? .ds 1
669C ?? .ds 1
669D ?? .ds 1
669E ?? .ds 1
669F ?? .ds 1
66A0 ?? unk_0_66A0: .ds 1 ; DATA XREF: sub_0_3A2+1A!o
; init_fireball_sprite!o ...
66A1 ?? .ds 1
66A2 ?? .ds 1
66A3 ?? .ds 1
66A4 ?? .ds 1
66A5 ?? .ds 1
66A6 ?? .ds 1
66A7 ?? .ds 1
66A8 ?? .ds 1
66A9 ?? .ds 1
66AA ?? .ds 1
66AB ?? .ds 1
66AC ?? .ds 1
66AD ?? .ds 1
66AE ?? .ds 1
66AF ?? .ds 1
66B0 ?? .ds 1
66B1 ?? .ds 1
66B2 ?? .ds 1
66B3 ?? .ds 1
66B4 ?? .ds 1
66B5 ?? .ds 1
66B6 ?? .ds 1
66B7 ?? .ds 1
66B8 ?? .ds 1
66B9 ?? .ds 1
66BA ?? .ds 1
66BB ?? .ds 1
66BC ?? .ds 1
66BD ?? .ds 1
66BE ?? .ds 1
66BF ?? .ds 1
66C0 ?? .ds 1
66C1 ?? .ds 1
66C2 ?? .ds 1
66C3 ?? .ds 1
66C4 ?? .ds 1
66C5 ?? .ds 1
66C6 ?? .ds 1
66C7 ?? .ds 1
66C8 ?? .ds 1
66C9 ?? .ds 1
66CA ?? .ds 1
66CB ?? .ds 1
66CC ?? .ds 1
66CD ?? .ds 1
66CE ?? .ds 1
66CF ?? .ds 1
66D0 ?? .ds 1
66D1 ?? .ds 1
66D2 ?? .ds 1
66D3 ?? .ds 1

```

66D4 ??	.ds 1	
66D5 ??	.ds 1	
66D6 ??	.ds 1	
66D7 ??	.ds 1	
66D8 ??	.ds 1	
66D9 ??	.ds 1	
66DA ??	.ds 1	
66DB ??	.ds 1	
66DC ??	.ds 1	
66DD ??	.ds 1	
66DE ??	.ds 1	
66DF ??	.ds 1	
66E0 ??	.ds 1	
66E1 ??	.ds 1	
66E2 ??	.ds 1	
66E3 ??	.ds 1	
66E4 ??	.ds 1	
66E5 ??	.ds 1	
66E6 ??	.ds 1	
66E7 ??	.ds 1	
66E8 ??	.ds 1	
66E9 ??	.ds 1	
66EA ??	.ds 1	
66EB ??	.ds 1	
66EC ??	.ds 1	
66ED ??	.ds 1	
66EE ??	.ds 1	
66EF ??	.ds 1	
66F0 ??	.ds 1	
66F1 ??	.ds 1	
66F2 ??	.ds 1	
66F3 ??	.ds 1	
66F4 ??	.ds 1	
66F5 ??	.ds 1	
66F6 ??	.ds 1	
66F7 ??	.ds 1	
66F8 ??	.ds 1	
66F9 ??	.ds 1	
66FA ??	.ds 1	
66FB ??	.ds 1	
66FC ??	.ds 1	
66FD ??	.ds 1	
66FE ??	.ds 1	
66FF ??	.ds 1	
6700 ??	.ds 1	unk_0_6700: ; DATA XREF: sub_0_1F72+5 o ; sub_0_286F+1B o ...
6701 ??	.ds 1	
6702 ??	.ds 1	
6703 ??	.ds 1	
6704 ??	.ds 1	
6705 ??	.ds 1	
6706 ??	.ds 1	
6707 ??	.ds 1	unk_0_6707: ; DATA XREF: 0000:1009 o
6708 ??	.ds 1	
6709 ??	.ds 1	
670A ??	.ds 1	
670B ??	.ds 1	
670C ??	.ds 1	
670D ??	.ds 1	
670E ??	.ds 1	
670F ??	.ds 1	
6710 ??	.ds 1	
6711 ??	.ds 1	
6712 ??	.ds 1	
6713 ??	.ds 1	
6714 ??	.ds 1	
6715 ??	.ds 1	
6716 ??	.ds 1	
6717 ??	.ds 1	
6718 ??	.ds 1	
6719 ??	.ds 1	
671A ??	.ds 1	
671B ??	.ds 1	
671C ??	.ds 1	
671D ??	.ds 1	
671E ??	.ds 1	
671F ??	.ds 1	
6720 ??	.ds 1	
6721 ??	.ds 1	
6722 ??	.ds 1	
6723 ??	.ds 1	
6724 ??	.ds 1	
6725 ??	.ds 1	
6726 ??	.ds 1	
6727 ??	.ds 1	
6728 ??	.ds 1	
6729 ??	.ds 1	
672A ??	.ds 1	
672B ??	.ds 1	
672C ??	.ds 1	
672D ??	.ds 1	
672E ??	.ds 1	
672F ??	.ds 1	
6730 ??	.ds 1	
6731 ??	.ds 1	
6732 ??	.ds 1	
6733 ??	.ds 1	
6734 ??	.ds 1	
6735 ??	.ds 1	
6736 ??	.ds 1	
6737 ??	.ds 1	
6738 ??	.ds 1	
6739 ??	.ds 1	
673A ??	.ds 1	
673B ??	.ds 1	
673C ??	.ds 1	
673D ??	.ds 1	
673E ??	.ds 1	
673F ??	.ds 1	
6740 ??	.ds 1	
6741 ??	.ds 1	
6742 ??	.ds 1	
6743 ??	.ds 1	
6744 ??	.ds 1	
6745 ??	.ds 1	
6746 ??	.ds 1	

6747	??	.ds	1
6748	??	.ds	1
6749	??	.ds	1
674A	??	.ds	1
674B	??	.ds	1
674C	??	.ds	1
674D	??	.ds	1
674E	??	.ds	1
674F	??	.ds	1
6750	??	.ds	1
6751	??	.ds	1
6752	??	.ds	1
6753	??	.ds	1
6754	??	.ds	1
6755	??	.ds	1
6756	??	.ds	1
6757	??	.ds	1
6758	??	.ds	1
6759	??	.ds	1
675A	??	.ds	1
675B	??	.ds	1
675C	??	.ds	1
675D	??	.ds	1
675E	??	.ds	1
675F	??	.ds	1
6760	??	.ds	1
6761	??	.ds	1
6762	??	.ds	1
6763	??	.ds	1
6764	??	.ds	1
6765	??	.ds	1
6766	??	.ds	1
6767	??	.ds	1
6768	??	.ds	1
6769	??	.ds	1
676A	??	.ds	1
676B	??	.ds	1
676C	??	.ds	1
676D	??	.ds	1
676E	??	.ds	1
676F	??	.ds	1
6770	??	.ds	1
6771	??	.ds	1
6772	??	.ds	1
6773	??	.ds	1
6774	??	.ds	1
6775	??	.ds	1
6776	??	.ds	1
6777	??	.ds	1
6778	??	.ds	1
6779	??	.ds	1
677A	??	.ds	1
677B	??	.ds	1
677C	??	.ds	1
677D	??	.ds	1
677E	??	.ds	1
677F	??	.ds	1
6780	??	.ds	1
6781	??	.ds	1
6782	??	.ds	1
6783	??	.ds	1
6784	??	.ds	1
6785	??	.ds	1
6786	??	.ds	1
6787	??	.ds	1
6788	??	.ds	1
6789	??	.ds	1
678A	??	.ds	1
678B	??	.ds	1
678C	??	.ds	1
678D	??	.ds	1
678E	??	.ds	1
678F	??	.ds	1
6790	??	.ds	1
6791	??	.ds	1
6792	??	.ds	1
6793	??	.ds	1
6794	??	.ds	1
6795	??	.ds	1
6796	??	.ds	1
6797	??	.ds	1
6798	??	.ds	1
6799	??	.ds	1
679A	??	.ds	1
679B	??	.ds	1
679C	??	.ds	1
679D	??	.ds	1
679E	??	.ds	1
679F	??	.ds	1
67A0	??	.ds	1
67A1	??	.ds	1
67A2	??	.ds	1
67A3	??	.ds	1
67A4	??	.ds	1
67A5	??	.ds	1
67A6	??	.ds	1
67A7	??	.ds	1
67A8	??	.ds	1
67A9	??	.ds	1
67AA	??	.ds	1
67AB	??	.ds	1
67AC	??	.ds	1
67AD	??	.ds	1
67AE	??	.ds	1
67AF	??	.ds	1
67B0	??	.ds	1
67B1	??	.ds	1
67B2	??	.ds	1
67B3	??	.ds	1
67B4	??	.ds	1
67B5	??	.ds	1
67B6	??	.ds	1
67B7	??	.ds	1
67B8	??	.ds	1
67B9	??	.ds	1
67BA	??	.ds	1

67BB	??	.ds	1
67BC	??	.ds	1
67BD	??	.ds	1
67BE	??	.ds	1
67BF	??	.ds	1
67C0	??	.ds	1
67C1	??	.ds	1
67C2	??	.ds	1
67C3	??	.ds	1
67C4	??	.ds	1
67C5	??	.ds	1
67C6	??	.ds	1
67C7	??	.ds	1
67C8	??	.ds	1
67C9	??	.ds	1
67CA	??	.ds	1
67CB	??	.ds	1
67CC	??	.ds	1
67CD	??	.ds	1
67CE	??	.ds	1
67CF	??	.ds	1
67D0	??	.ds	1
67D1	??	.ds	1
67D2	??	.ds	1
67D3	??	.ds	1
67D4	??	.ds	1
67D5	??	.ds	1
67D6	??	.ds	1
67D7	??	.ds	1
67D8	??	.ds	1
67D9	??	.ds	1
67DA	??	.ds	1
67DB	??	.ds	1
67DC	??	.ds	1
67DD	??	.ds	1
67DE	??	.ds	1
67DF	??	.ds	1
67E0	??	.ds	1
67E1	??	.ds	1
67E2	??	.ds	1
67E3	??	.ds	1
67E4	??	.ds	1
67E5	??	.ds	1
67E6	??	.ds	1
67E7	??	.ds	1
67E8	??	.ds	1
67E9	??	.ds	1
67EA	??	.ds	1
67EB	??	.ds	1
67EC	??	.ds	1
67ED	??	.ds	1
67EE	??	.ds	1
67EF	??	.ds	1
67F0	??	.ds	1
67F1	??	.ds	1
67F2	??	.ds	1
67F3	??	.ds	1
67F4	??	.ds	1
67F5	??	.ds	1
67F6	??	.ds	1
67F7	??	.ds	1
67F8	??	.ds	1
67F9	??	.ds	1
67FA	??	.ds	1
67FB	??	.ds	1
67FC	??	.ds	1
67FD	??	.ds	1
67FE	??	.ds	1
67FF	??	.ds	1
6800	??	.ds	1
6801	??	.ds	1
6802	??	.ds	1
6803	??	.ds	1
6804	??	.ds	1
6805	??	.ds	1
6806	??	.ds	1
6807	unk_0_6807:	.ds	1 ; DATA XREF: 0000:1012 o
6808	??	.ds	1
6809	??	.ds	1
680A	??	.ds	1
680B	??	.ds	1
680C	??	.ds	1
680D	??	.ds	1
680E	??	.ds	1
680F	??	.ds	1
6810	??	.ds	1
6811	??	.ds	1
6812	??	.ds	1
6813	??	.ds	1
6814	??	.ds	1
6815	??	.ds	1
6816	??	.ds	1
6817	??	.ds	1
6818	??	.ds	1
6819	??	.ds	1
681A	??	.ds	1
681B	??	.ds	1
681C	??	.ds	1
681D	??	.ds	1
681E	??	.ds	1
681F	??	.ds	1
6820	??	.ds	1
6821	??	.ds	1
6822	??	.ds	1
6823	??	.ds	1
6824	??	.ds	1
6825	??	.ds	1
6826	??	.ds	1
6827	??	.ds	1
6828	??	.ds	1
6829	??	.ds	1
682A	??	.ds	1
682B	??	.ds	1
682C	??	.ds	1
682D	??	.ds	1
682E	??	.ds	1

682F ??	.ds 1
6830 ??	.ds 1
6831 ??	.ds 1
6832 ??	.ds 1
6833 ??	.ds 1
6834 ??	.ds 1
6835 ??	.ds 1
6836 ??	.ds 1
6837 ??	.ds 1
6838 ??	.ds 1
6839 ??	.ds 1
683A ??	.ds 1
683B ??	.ds 1
683C ??	.ds 1
683D ??	.ds 1
683E ??	.ds 1
683F ??	.ds 1
6840 ??	.ds 1
6841 ??	.ds 1
6842 ??	.ds 1
6843 ??	.ds 1
6844 ??	.ds 1
6845 ??	.ds 1
6846 ??	.ds 1
6847 ??	.ds 1
6848 ??	.ds 1
6849 ??	.ds 1
684A ??	.ds 1
684B ??	.ds 1
684C ??	.ds 1
684D ??	.ds 1
684E ??	.ds 1
684F ??	.ds 1
6850 ??	.ds 1
6851 ??	.ds 1
6852 ??	.ds 1
6853 ??	.ds 1
6854 ??	.ds 1
6855 ??	.ds 1
6856 ??	.ds 1
6857 ??	.ds 1
6858 ??	.ds 1
6859 ??	.ds 1
685A ??	.ds 1
685B ??	.ds 1
685C ??	.ds 1
685D ??	.ds 1
685E ??	.ds 1
685F ??	.ds 1
6860 ??	.ds 1
6861 ??	.ds 1
6862 ??	.ds 1
6863 ??	.ds 1
6864 ??	.ds 1
6865 ??	.ds 1
6866 ??	.ds 1
6867 ??	.ds 1
6868 ??	.ds 1
6869 ??	.ds 1
686A ??	.ds 1
686B ??	.ds 1
686C ??	.ds 1
686D ??	.ds 1
686E ??	.ds 1
686F ??	.ds 1
6870 ??	.ds 1
6871 ??	.ds 1
6872 ??	.ds 1
6873 ??	.ds 1
6874 ??	.ds 1
6875 ??	.ds 1
6876 ??	.ds 1
6877 ??	.ds 1
6878 ??	.ds 1
6879 ??	.ds 1
687A ??	.ds 1
687B ??	.ds 1
687C ??	.ds 1
687D ??	.ds 1
687E ??	.ds 1
687F ??	.ds 1
6880 ??	.ds 1
6881 ??	.ds 1
6882 ??	.ds 1
6883 ??	.ds 1
6884 ??	.ds 1
6885 ??	.ds 1
6886 ??	.ds 1
6887 ??	.ds 1
6888 ??	.ds 1
6889 ??	.ds 1
688A ??	.ds 1
688B ??	.ds 1
688C ??	.ds 1
688D ??	.ds 1
688E ??	.ds 1
688F ??	.ds 1
6890 ??	.ds 1
6891 ??	.ds 1
6892 ??	.ds 1
6893 ??	.ds 1
6894 ??	.ds 1
6895 ??	.ds 1
6896 ??	.ds 1
6897 ??	.ds 1
6898 ??	.ds 1
6899 ??	.ds 1
689A ??	.ds 1
689B ??	.ds 1
689C ??	.ds 1
689D ??	.ds 1
689E ??	.ds 1
689F ??	.ds 1
68A0 ??	.ds 1
68A1 ??	.ds 1
68A2 ??	.ds 1



```

68A3 ?? .ds 1
68A4 ?? .ds 1
68A5 ?? .ds 1
68A6 ?? .ds 1
68A7 ?? .ds 1
68A8 ?? .ds 1
68A9 ?? .ds 1
68AA ?? .ds 1
68AB ?? .ds 1
68AC ?? .ds 1
68AD ?? .ds 1
68AE ?? .ds 1
68AF ?? .ds 1
68B0 ?? .ds 1
68B1 ?? .ds 1
68B2 ?? .ds 1
68B3 ?? .ds 1
68B4 ?? .ds 1
68B5 ?? .ds 1
68B6 ?? .ds 1
68B7 ?? .ds 1
68B8 ?? .ds 1
68B9 ?? .ds 1
68BA ?? .ds 1
68BB ?? .ds 1
68BC ?? .ds 1
68BD ?? .ds 1
68BE ?? .ds 1
68BF ?? .ds 1
68C0 ?? .ds 1
68C1 ?? .ds 1
68C2 ?? .ds 1
68C3 ?? .ds 1
68C4 ?? .ds 1
68C5 ?? .ds 1
68C6 ?? .ds 1
68C7 ?? .ds 1
68C8 ?? .ds 1
68C9 ?? .ds 1
68CA ?? .ds 1
68CB ?? .ds 1
68CC ?? .ds 1
68CD ?? .ds 1
68CE ?? .ds 1
68CF ?? .ds 1
68D0 ?? .ds 1
68D1 ?? .ds 1
68D2 ?? .ds 1
68D3 ?? .ds 1
68D4 ?? .ds 1
68D5 ?? .ds 1
68D6 ?? .ds 1
68D7 ?? .ds 1
68D8 ?? .ds 1
68D9 ?? .ds 1
68DA ?? .ds 1
68DB ?? .ds 1
68DC ?? .ds 1
68DD ?? .ds 1
68DE ?? .ds 1
68DF ?? .ds 1
68E0 ?? .ds 1
68E1 ?? .ds 1
68E2 ?? .ds 1
68E3 ?? .ds 1
68E4 ?? .ds 1
68E5 ?? .ds 1
68E6 ?? .ds 1
68E7 ?? .ds 1
68E8 ?? .ds 1
68E9 ?? .ds 1
68EA ?? .ds 1
68EB ?? .ds 1
68EC ?? .ds 1
68ED ?? .ds 1
68EE ?? .ds 1
68EF ?? .ds 1
68F0 ?? .ds 1
68F1 ?? .ds 1
68F2 ?? .ds 1
68F3 ?? .ds 1
68F4 ?? .ds 1
68F5 ?? .ds 1
68F6 ?? .ds 1
68F7 ?? .ds 1
68F8 ?? .ds 1
68F9 ?? .ds 1
68FA ?? .ds 1
68FB ?? .ds 1
68FC ?? .ds 1
68FD ?? .ds 1
68FE ?? .ds 1
68FF ?? .ds 1
6900 ?? ?? ?? ??+soft_sprite_ram: .ds 0x180
6900 ?? ?? ?? ??+
6900 ?? ?? ?? ??+
6900 ?? ?? ?? ??+
6900 ?? ?? ?? ??+
6900 ?? ?? ?? ??+
6A80 ?? .ds 1
6A81 ?? .ds 1
6A82 ?? .ds 1
6A83 ?? .ds 1
6A84 ?? .ds 1
6A85 ?? .ds 1
6A86 ?? .ds 1
6A87 ?? .ds 1
6A88 ?? .ds 1
6A89 ?? .ds 1
6A8A ?? .ds 1
6A8B ?? .ds 1
6A8C ?? .ds 1
6A8D ?? .ds 1
6A8E ?? .ds 1
6A8F ?? .ds 1
6A90 ?? .ds 1
; DATA XREF: 0000:0139|o
; clear_tiles_and_sprites+11|o ...
; 0- 1 = pauline
; 2-11 = kong
; 12-
; 19 = mario

```

6A91 ??	.ds 1
6A92 ??	.ds 1
6A93 ??	.ds 1
6A94 ??	.ds 1
6A95 ??	.ds 1
6A96 ??	.ds 1
6A97 ??	.ds 1
6A98 ??	.ds 1
6A99 ??	.ds 1
6A9A ??	.ds 1
6A9B ??	.ds 1
6A9C ??	.ds 1
6A9D ??	.ds 1
6A9E ??	.ds 1
6A9F ??	.ds 1
6AA0 ??	.ds 1
6AA1 ??	.ds 1
6AA2 ??	.ds 1
6AA3 ??	.ds 1
6AA4 ??	.ds 1
6AA5 ??	.ds 1
6AA6 ??	.ds 1
6AA7 ??	.ds 1
6AA8 ??	.ds 1
6AA9 ??	.ds 1
6AAA ??	.ds 1
6AAB ??	.ds 1
6AAC ??	.ds 1
6AAD ??	.ds 1
6AAE ??	.ds 1
6AAF ??	.ds 1
6AB0 ??	.ds 1
6AB1 ??	.ds 1
6AB2 ??	.ds 1
6AB3 ??	.ds 1
6AB4 ??	.ds 1
6AB5 ??	.ds 1
6AB6 ??	.ds 1
6AB7 ??	.ds 1
6AB8 ??	.ds 1
6AB9 ??	.ds 1
6ABA ??	.ds 1
6ABB ??	.ds 1
6ABC ??	.ds 1
6ABD ??	.ds 1
6ABE ??	.ds 1
6ABF ??	.ds 1
6AC0 ??	.ds 1
6AC1 ??	.ds 1
6AC2 ??	.ds 1
6AC3 ??	.ds 1
6AC4 ??	.ds 1
6AC5 ??	.ds 1
6AC6 ??	.ds 1
6AC7 ??	.ds 1
6AC8 ??	.ds 1
6AC9 ??	.ds 1
6ACA ??	.ds 1
6ACB ??	.ds 1
6ACC ??	.ds 1
6ACD ??	.ds 1
6ACE ??	.ds 1
6ACF ??	.ds 1
6AD0 ??	.ds 1
6AD1 ??	.ds 1
6AD2 ??	.ds 1
6AD3 ??	.ds 1
6AD4 ??	.ds 1
6AD5 ??	.ds 1
6AD6 ??	.ds 1
6AD7 ??	.ds 1
6AD8 ??	.ds 1
6AD9 ??	.ds 1
6ADA ??	.ds 1
6ADB ??	.ds 1
6ADC ??	.ds 1
6ADD ??	.ds 1
6ADE ??	.ds 1
6ADF ??	.ds 1
6AE0 ??	.ds 1
6AE1 ??	.ds 1
6AE2 ??	.ds 1
6AE3 ??	.ds 1
6AE4 ??	.ds 1
6AE5 ??	.ds 1
6AE6 ??	.ds 1
6AE7 ??	.ds 1
6AE8 ??	.ds 1
6AE9 ??	.ds 1
6AEA ??	.ds 1
6AEB ??	.ds 1
6AEC ??	.ds 1
6AED ??	.ds 1
6AEE ??	.ds 1
6AEF ??	.ds 1
6AF0 ??	.ds 1
6AF1 ??	.ds 1
6AF2 ??	.ds 1
6AF3 ??	.ds 1
6AF4 ??	.ds 1
6AF5 ??	.ds 1
6AF6 ??	.ds 1
6AF7 ??	.ds 1
6AF8 ??	.ds 1
6AF9 ??	.ds 1
6AFA ??	.ds 1
6AFB ??	.ds 1
6AFC ??	.ds 1
6AFD ??	.ds 1
6AFE ??	.ds 1
6AFF ??	.ds 1
6B00 ??	.ds 1
6B01 ??	.ds 1
6B02 ??	.ds 1
6B03 ??	.ds 1
6B04 ??	.ds 1

6B05 ??	.ds 1
6B06 ??	.ds 1
6B07 ??	.ds 1
6B08 ??	.ds 1
6B09 ??	.ds 1
6B0A ??	.ds 1
6B0B ??	.ds 1
6B0C ??	.ds 1
6B0D ??	.ds 1
6B0E ??	.ds 1
6B0F ??	.ds 1
6B10 ??	.ds 1
6B11 ??	.ds 1
6B12 ??	.ds 1
6B13 ??	.ds 1
6B14 ??	.ds 1
6B15 ??	.ds 1
6B16 ??	.ds 1
6B17 ??	.ds 1
6B18 ??	.ds 1
6B19 ??	.ds 1
6B1A ??	.ds 1
6B1B ??	.ds 1
6B1C ??	.ds 1
6B1D ??	.ds 1
6B1E ??	.ds 1
6B1F ??	.ds 1
6B20 ??	.ds 1
6B21 ??	.ds 1
6B22 ??	.ds 1
6B23 ??	.ds 1
6B24 ??	.ds 1
6B25 ??	.ds 1
6B26 ??	.ds 1
6B27 ??	.ds 1
6B28 ??	.ds 1
6B29 ??	.ds 1
6B2A ??	.ds 1
6B2B ??	.ds 1
6B2C ??	.ds 1
6B2D ??	.ds 1
6B2E ??	.ds 1
6B2F ??	.ds 1
6B30 ??	.ds 1
6B31 ??	.ds 1
6B32 ??	.ds 1
6B33 ??	.ds 1
6B34 ??	.ds 1
6B35 ??	.ds 1
6B36 ??	.ds 1
6B37 ??	.ds 1
6B38 ??	.ds 1
6B39 ??	.ds 1
6B3A ??	.ds 1
6B3B ??	.ds 1
6B3C ??	.ds 1
6B3D ??	.ds 1
6B3E ??	.ds 1
6B3F ??	.ds 1
6B40 ??	.ds 1
6B41 ??	.ds 1
6B42 ??	.ds 1
6B43 ??	.ds 1
6B44 ??	.ds 1
6B45 ??	.ds 1
6B46 ??	.ds 1
6B47 ??	.ds 1
6B48 ??	.ds 1
6B49 ??	.ds 1
6B4A ??	.ds 1
6B4B ??	.ds 1
6B4C ??	.ds 1
6B4D ??	.ds 1
6B4E ??	.ds 1
6B4F ??	.ds 1
6B50 ??	.ds 1
6B51 ??	.ds 1
6B52 ??	.ds 1
6B53 ??	.ds 1
6B54 ??	.ds 1
6B55 ??	.ds 1
6B56 ??	.ds 1
6B57 ??	.ds 1
6B58 ??	.ds 1
6B59 ??	.ds 1
6B5A ??	.ds 1
6B5B ??	.ds 1
6B5C ??	.ds 1
6B5D ??	.ds 1
6B5E ??	.ds 1
6B5F ??	.ds 1
6B60 ??	.ds 1
6B61 ??	.ds 1
6B62 ??	.ds 1
6B63 ??	.ds 1
6B64 ??	.ds 1
6B65 ??	.ds 1
6B66 ??	.ds 1
6B67 ??	.ds 1
6B68 ??	.ds 1
6B69 ??	.ds 1
6B6A ??	.ds 1
6B6B ??	.ds 1
6B6C ??	.ds 1
6B6D ??	.ds 1
6B6E ??	.ds 1
6B6F ??	.ds 1
6B70 ??	.ds 1
6B71 ??	.ds 1
6B72 ??	.ds 1
6B73 ??	.ds 1
6B74 ??	.ds 1
6B75 ??	.ds 1
6B76 ??	.ds 1
6B77 ??	.ds 1
6B78 ??	.ds 1

6B79 ??	.ds 1
6B7A ??	.ds 1
6B7B ??	.ds 1
6B7C ??	.ds 1
6B7D ??	.ds 1
6B7E ??	.ds 1
6B7F ??	.ds 1
6B80 ??	.ds 1
6B81 ??	.ds 1
6B82 ??	.ds 1
6B83 ??	.ds 1
6B84 ??	.ds 1
6B85 ??	.ds 1
6B86 ??	.ds 1
6B87 ??	.ds 1
6B88 ??	.ds 1
6B89 ??	.ds 1
6B8A ??	.ds 1
6B8B ??	.ds 1
6B8C ??	.ds 1
6B8D ??	.ds 1
6B8E ??	.ds 1
6B8F ??	.ds 1
6B90 ??	.ds 1
6B91 ??	.ds 1
6B92 ??	.ds 1
6B93 ??	.ds 1
6B94 ??	.ds 1
6B95 ??	.ds 1
6B96 ??	.ds 1
6B97 ??	.ds 1
6B98 ??	.ds 1
6B99 ??	.ds 1
6B9A ??	.ds 1
6B9B ??	.ds 1
6B9C ??	.ds 1
6B9D ??	.ds 1
6B9E ??	.ds 1
6B9F ??	.ds 1
6BA0 ??	.ds 1
6BA1 ??	.ds 1
6BA2 ??	.ds 1
6BA3 ??	.ds 1
6BA4 ??	.ds 1
6BA5 ??	.ds 1
6BA6 ??	.ds 1
6BA7 ??	.ds 1
6BA8 ??	.ds 1
6BA9 ??	.ds 1
6BAA ??	.ds 1
6BAB ??	.ds 1
6BAC ??	.ds 1
6BAD ??	.ds 1
6BAE ??	.ds 1
6BAF ??	.ds 1
6BB0 ??	.ds 1
6BB1 ??	.ds 1
6BB2 ??	.ds 1
6BB3 ??	.ds 1
6BB4 ??	.ds 1
6BB5 ??	.ds 1
6BB6 ??	.ds 1
6BB7 ??	.ds 1
6BB8 ??	.ds 1
6BB9 ??	.ds 1
6BBA ??	.ds 1
6BBB ??	.ds 1
6BBC ??	.ds 1
6BBD ??	.ds 1
6BBE ??	.ds 1
6BBF ??	.ds 1
6BC0 ??	.ds 1
6BC1 ??	.ds 1
6BC2 ??	.ds 1
6BC3 ??	.ds 1
6BC4 ??	.ds 1
6BC5 ??	.ds 1
6BC6 ??	.ds 1
6BC7 ??	.ds 1
6BC8 ??	.ds 1
6BC9 ??	.ds 1
6BCA ??	.ds 1
6BCB ??	.ds 1
6BCC ??	.ds 1
6BCD ??	.ds 1
6BCE ??	.ds 1
6BCF ??	.ds 1
6BD0 ??	.ds 1
6BD1 ??	.ds 1
6BD2 ??	.ds 1
6BD3 ??	.ds 1
6BD4 ??	.ds 1
6BD5 ??	.ds 1
6BD6 ??	.ds 1
6BD7 ??	.ds 1
6BD8 ??	.ds 1
6BD9 ??	.ds 1
6BDA ??	.ds 1
6BDB ??	.ds 1
6BDC ??	.ds 1
6BDD ??	.ds 1
6BDE ??	.ds 1
6BDF ??	.ds 1
6BE0 ??	.ds 1
6BE1 ??	.ds 1
6BE2 ??	.ds 1
6BE3 ??	.ds 1
6BE4 ??	.ds 1
6BE5 ??	.ds 1
6BE6 ??	.ds 1
6BE7 ??	.ds 1
6BE8 ??	.ds 1
6BE9 ??	.ds 1
6BEA ??	.ds 1
6BEB ??	.ds 1
6BEC ??	.ds 1

```

6BED ??                .ds 1
6BEE ??                .ds 1
6BEF ??                .ds 1
6BF0 ??                .ds 1
6BF1 ??                .ds 1
6BF2 ??                .ds 1
6BF3 ??                .ds 1
6BF4 ??                .ds 1
6BF5 ??                .ds 1
6BF6 ??                .ds 1
6BF7 ??                .ds 1
6BF8 ??                .ds 1
6BF9 ??                .ds 1
6BFA ??                .ds 1
6BFB ??                .ds 1
6BFC ??                .ds 1
6BFD ??                .ds 1
6BFE ??                .ds 1
6BFF ??                .ds 1
6BFF                ; end of 'RAM'
6BFF                ; =====
7000                ;
7000                ; Segment type: Regular
7000                ; segment 'SPRAM'
7000                .org 0x7000
7000 ?? ?? ?? ??+SPRAM_start: .ds 0x400                ; DATA XREF: 0000:013D|o
7000 ?? ?? ?? ??+                ; 0000:0276|o
7000 ?? ?? ?? ??+; end of 'SPRAM'                ; 2 banks of 128 sprites
7000 ?? ?? ?? ??+                ; - only 16 displayed per scanline
7000 ?? ?? ?? ??+                ; @0 7:0=y
7000 ?? ?? ?? ??+                ; @1 7=flipy,6:0=code
7000 ?? ?? ?? ??+                ; @2 7=flipx,3:0=colour
7000 ?? ?? ?? ??+                ; @3 7:0=x
7400                ; =====
7400                ; Segment type: Regular
7400                ; segment 'VRAM'
7400                .org 0x7400
7400 ?? ?? ?? ??+VRAM_start: .ds 0x400                ; DATA XREF: 0000:0285|o
7400 ?? ?? ?? ??+                ; clear_tiles_and_sprites|o ...
7400 ?? ?? ?? ??+; end of 'VRAM'
7400 ?? ?? ?? ??+
7800                ; =====
7800                ; Segment type: Regular
7800                ; segment 'I8257'
7800                .org 0x7800
7800 ?? ?? ?? ??+i8257_io: .ds 0x10
7800 ?? ?? ?? ??+; end of 'I8257'
7800 ?? ?? ?? ??+
7C00                ; =====
7C00                ; Segment type: Regular
7C00                ; segment 'IN0'
7C00                .org 0x7C00
7C00 ?? in0: .ds 1
7C00                ; end of 'IN0'
7C80                ; =====
7C80                ; Segment type: Regular
7C80                ; segment 'IN1'
7C80                .org 0x7C80
7C80 ?? in1: .ds 1
7C80                ; end of 'IN1'
7D00                ; =====
7D00                ; Segment type: Regular
7D00                ; segment 'IO'
7D00                .org 0x7D00
7D00 ?? in2_snd_latch: .ds 1                ; DATA XREF: 0000:0072|r
7D00                ; update_sounds+3|o ...
7D01 ??                .ds 1
7D02 ??                .ds 1
7D03 ??                .ds 1
7D04 ??                .ds 1
7D05 ??                .ds 1
7D06 ??                .ds 1
7D07 ??                .ds 1
7D08 ?? ?? ?? ??+                .ds 0x78
7D80 ?? dsw_audio_irq: .ds 1
7D81 ??                .ds 1
7D82 ?? flipscreen: .ds 1                ; DATA XREF: 0000:01E4|w
7D82                ; 0000:02AF|w ...
7D83 ?? spritebank: .ds 1
7D84 ?? nmi_mask: .ds 1
7D85 ?? p8257_drq: .ds 1
7D86*?? palette_bank: .ds 2                ; DATA XREF: 0000:02A8|w
7D86*                ; 0000:0779|o ...
7D88 ?? ?? ?? ??+                .ds 0x78
7D88 ?? ?? ?? ??+; end of 'IO'
7D88 ?? ?? ?? ??+
7D88 ?? ?? ?? ??+; end of file

```