

# !!SECTION UNDER CONSTRUCTION!!

Instruction Name	Length	Example / Info	History
{00} Nop	01	<pre>00++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x00  // 0x00 } Nop; This bytecode is used for alignment of 1-byte opcodes and ending Elself blocks.</pre>	08-02-2024 Newly Added
{01} Evt_end	02	<pre>01 00++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x01  // 0x01     UCHAR zAlign;        // Always Zero (Alignment byte) } Evt_end; This bytecode ends the current Main/Sub script.</pre>	08-02-2024 Newly Added
{02} Evt_next	01	<pre>02++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x02  // 0x02 } Evt_next; This bytecode moves to the next event in the sequence.</pre>	08-02-2024 Newly Added
{03} Evt_chain	02	<pre>03 ID++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x03  // 0x03     UCHAR NextEventId;   // Event ID to chain to } Evt_chain; This bytecode chains the current event to the specified next event ID, allowing the script to continue execution from the linked event.</pre>	08-02-2024 Newly Added
{04} Evt_exec	04	<pre>04 ID ?? ??++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x04  // 0x04     UCHAR data1;         // Typically FF     UCHAR GoSub;         // Opcode for GoSub 0x18     UCHAR ScdId;         // Sub Script ID to Jump to } Evt_exec; This bytecode executes the specified event with given parameters.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{05} Evt_kill	02	<p><b>05 ID++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x05 // 0x05     UCHAR EventId; // Event ID to terminate } Evt_kill; This bytecode terminates the specified event.</pre>	08-02-2024 Newly Added
{06} Ifel_ck	04	<p><b>06 00 SI ZE++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x06 // 0x06     UCHAR zAlign; // Always Zero (Alignment byte)     USHORT Size; // Size of the block to check } Ifel_ck; This bytecode checks a condition and branches accordingly.</pre>	08-02-2024 Newly Added
{07} Else_ck	04	<p><b>07 00 SI ZE++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x07 // 0x07     UCHAR zAlign; // Always Zero (Alignment byte)     USHORT Size; // Size of the block to check } Else_ck; This bytecode specifies the size of the block to check if the corresponding Ifel_ck condition is met.</pre>	08-02-2024 Newly Added
{08} Endif	02	<p><b>08 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x08 // 0x08     UCHAR zAlign; // Always Zero (Alignment byte) } Endif; This bytecode marks the end of an If/Else/Else block.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{09} Sleep	04	<pre>09 ?? ?? 00++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x09  // 0x09     UCHAR Sleeping;      // Opcode for Sleeping 0x0A     USHORT Count;        // Timer / Sleep Duration } Sleep; This bytecode pauses script execution for the specified duration.</pre>	08-02-2024 Newly Added
{0A} Sleeping	03	<pre>0A ?? 00++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x0A  // 0x0A     USHORT Count;        // Timer / Sleep Duration } Sleeping; This bytecode pauses script execution for the specified duration.</pre>	08-02-2024 Newly Added
{0B} Wsleep	01	<pre>0B++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x0B  // 0x0B } Wsleep; This bytecode used before 0C will wait until the current XA sound has finished playing before proceeding.</pre>	08-02-2024 Newly Added
{0C} Wsleeping	01	<pre>0C++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x0C  // 0x0C } Wsleeping; This bytecode used after 0B will wait until the current XA sound has finished playing before proceeding.</pre>	08-02-2024 Newly Added
{0D} For	06	<pre>0D 00 ?? ?? ?? ??++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x0D  // 0x0D     UCHAR zAlign;        // Always Zero (Alignment byte)     USHORT Size;         // Size of the block to check     USHORT Count;        // Amount of times block is looped } For; This bytecode begins a for-loop with the specified start and end values.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{0E} Next	02	<p><b>0E 00++</b></p> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x0E  // 0x0E     UCHAR zAlign;        // Always Zero (Alignment byte) } Next; This bytecode marks the end of a for-loop.</pre>	08-02-2024 Newly Added
{0F} While	04	<p><b>0F 00 ?? ??++</b></p> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x0F  // 0x0F     UCHAR zAlign;        // Always Zero (Alignment byte)     SHORT Size;          // Size of the block to check } While; This bytecode begins a while-loop that continues as long as the specified condition is true.</pre>	08-02-2024 Newly Added
{10} Ewhile	02	<p><b>10 ID++</b></p> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x10  // 0x10     UCHAR LoopId;        // ID of the while-loop to end } Ewhile; This bytecode ends the specified while-loop.</pre>	08-02-2024 Newly Added
{11} Do	04	<p><b>11 00 ?? ??++</b></p> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x11  // 0x11     UCHAR zAlign;        // Always Zero (Alignment byte)     SHORT Size;          // Size of the block to check } Do; This bytecode begins a do-while loop that executes the loop body once before checking the condition.</pre>	08-02-2024 Newly Added
{12} Edwhile	02	<p><b>12 ID++</b></p> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x12  // 0x12     UCHAR LoopId;        // ID of the do-while loop to end } Edwhile; This bytecode ends the specified do-while loop.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{13} Switch	04	<p><b>13 ID SI ZE++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x13 // 0x13     UCHAR SwitchId; // ID of the switch variable     USHORT Size; // Size of the block to check } Switch; This bytecode begins a switch-case block with the specified switch variable and default size.</pre>	08-02-2024 Newly Added
{14} Case	06	<p><b>14 ID ?? ?? ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x14 // 0x14     UCHAR zAlign; // Always Zero (Alignment byte)     USHORT Size; // Size of the block to check     USHORT CaseValue; // Value to compare with the switch variable } Case; This bytecode defines a case within a switch-case block.</pre>	08-02-2024 Newly Added
{15} Default	02	<p><b>15 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x15 // 0x15     UCHAR zAlign; // Always Zero (Alignment byte) } Default; This bytecode marks the default case in a switch- case block.</pre>	08-02-2024 Newly Added
{16} Eswitch	02	<p><b>16 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x16 // 0x16     UCHAR zAlign; // Always Zero (Alignment byte) } Eswitch; This bytecode ends the switch-case block.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{17} Goto	04	<p><b>17 ?? ?? 00 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x17 // 0x17     UCHAR Ifel_ctr; // Always 0xFF (0x01 on r304-sub05, only)     UCHAR Loop_ctr; // Always 0xFF (0x00 on r500-sub04 and sub07, only)     UCHAR zAlign; // Always 0x00     SHORT Offset; // Relative Pointer, always references same script } Goto; This bytecode jumps to the specified offset within the script.</pre>	08-02-2024 Newly Added
{18} Gosub	02	<p><b>18 ID++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x18 // 0x18     UCHAR SubroutineId; // ID of the subroutine to call } Gosub; This bytecode calls the specified subroutine.</pre>	08-02-2024 Newly Added
{19} Return	02	<p><b>19 ID++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x19 // 0x19     UCHAR SubroutineId; // ID of the subroutine to return from } Return; This bytecode returns from the specified subroutine.</pre>	08-02-2024 Newly Added
{1A} Break	02	<p><b>1A ID++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x1A // 0x1A     UCHAR LoopId; // ID of the loop to break from } Break; This bytecode breaks out of the specified loop.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{1B} For2	06	<p><b>1B 00 ?? ?? 00 ??++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x1B // 0x1B     UCHAR zAlign0; // Always Zero (Alignment byte)     SHORT StartValue; // Start value of the loop counter     UCHAR zAlign1; // Always Zero (Alignment byte)     UCHAR EndValue; // End value of the loop counter } For2;</pre> <p>This bytecode begins a for-loop with the specified start and end values.</p>	08-02-2024 Newly Added
{1C} Break_point	01	<p><b>1C++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x1C // 0x1C } Break_point;</pre> <p>This bytecode sets a breakpoint for debugging purposes.</p>	08-02-2024 Newly Added
{1D} Work_copy	04	<p><b>1D ?? ?? ??++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x1D // 0x1D     UCHAR Source; // Source index     UCHAR Destination; // Destination index     UCHAR Typecast; // Typecast operation } Work_copy;</pre> <p>This bytecode copies a value from the source index to the destination index with an optional typecast.</p>	08-02-2024 Newly Added
{1E} Nop1E	01	<p><b>1E++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x1E // 0x1E } Nop1E;</pre> <p>This bytecode is used for alignment of 1-byte opcodes and ending Elself blocks.</p>	08-02-2024 Newly Added
{1F} Nop1F	01	<p><b>1F++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x1F // 0x1F } Nop1F;</pre> <p>This bytecode is used for alignment of 1-byte opcodes and ending Elself blocks.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{20} Nop	01	<p><b>20++</b></p> <pre>typedef struct { // Ptr //     Description     UCHAR Opcode; // 0x20 //     0x20 } Nop; This bytecode is used for alignment of 1-byte opcodes and ending Elself blocks.</pre>	08-02-2024 Newly Added
{21} Ck	04	<p><b>21 ?? ?? ??++</b></p> <pre>typedef struct { // Ptr //     Description     UCHAR Opcode; // 0x21 //     0x21     UCHAR Flag; // System flag to check     UCHAR Id; // Bit ID to check     UCHAR OnOff; // On/Off state to check } Ck; This bytecode checks the specified system flag and bit ID for the given On/Off state.</pre>	08-02-2024 Newly Added
{22} Set	04	<p><b>22 ?? ?? ??++</b></p> <pre>typedef struct { // Ptr //     Description     UCHAR Opcode; // 0x22 //     0x22     UCHAR Flag; // System flag to set     UCHAR Id; // Bit ID to set     UCHAR OnOff; // On/Off state to set } Set; This bytecode sets the specified system flag and bit ID to the given On/Off state.</pre>	08-02-2024 Newly Added
{23} Cmp	06	<p><b>23 ?? ?? ?? ?? ??++</b></p> <pre>typedef struct { // Ptr //     Description     UCHAR Opcode; // 0x23 //     0x23     UCHAR Flag; // System flag to compare     UCHAR Operator; // Comparison operator     USHORT Value; // Value to compare against } Cmp; This bytecode compares the specified system flag with the given value using the provided comparison operator.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{24} Save	04	<pre>24 ID ?? ??++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x24   // 0x24     UCHAR Destination;  // Destination index     SHORT Source;       // Source value } Save; This bytecode saves the specified source value to the destination index.</pre>	08-02-2024 Newly Added
{25} Copy	03	<pre>25 ?? ??++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x25   // 0x25     UCHAR Destination;  // Destination index     UCHAR Source;       // Source index } Copy; This bytecode copies the value from the source index to the destination index.</pre>	08-02-2024 Newly Added
{26} Calc	06	<pre>26 ?? ?? ?? ?? ??++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x26   // 0x26     UCHAR zAlign;       // Always Zero (Alignment byte)     UCHAR Operator;     // Arithmetic operation to perform     UCHAR Flag;         // Memory Location to apply math to     SHORT Value;        // Amount used in operation } Calc; This bytecode performs the specified arithmetic operation on the operands and stores the result.</pre>	08-02-2024 Newly Added
{27} Calc2	04	<pre>27 ?? ?? ??++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x27   // 0x27     UCHAR Operator;     // Arithmetic operation to perform     UCHAR Flag;         // Memory Location to apply math to     UCHAR Value;        // Amount used in operation } Calc2; This bytecode performs the specified arithmetic operation on the operand and stores the result.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{28} Sce_rnd	01	<p><b>28++</b>                      typedef struct { // Ptr //                      Description                      UCHAR Opcode; // 0x28 //                      0x28                      } Sce_rnd;                      This bytecode generates a random value.</p>	08-02-2024 Newly Added
{29} Cut_chg	02	<p><b>29 ID++</b>                      typedef struct { // Ptr //                      Description                      UCHAR Opcode; // 0x29 //                      0x29                      UCHAR CutId; // ID of the camera                      to change to                      } Cut_chg;                      This bytecode changes the current camera to the                      specified camera ID.</p>	08-02-2024 Newly Added
{2A} Cut_old	01	<p><b>2A++</b>                      typedef struct { // Ptr //                      Description                      UCHAR Opcode; // 0x2A //                      0x2A                      } Cut_old;                      This bytecode reverts to the previous camera.</p>	08-02-2024 Newly Added
{2B} Message_on	06	<p><b>2B 00 ID ?? ?? ??++</b>                      typedef struct { // Ptr //                      Description                      UCHAR Opcode; // 0x2B //                      0x2B                      UCHAR Type; // Sub/Main?                      UCHAR MessageId; // ID of the                      message to display                      UCHAR zAlign; // Always Zero                      (Alignment byte)                      USHORT DisplayTime; // Time to                      display the message                      } Message_on;                      This bytecode displays the specified message.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{2C} Aot_set	20	<pre> 2C ID ?? ?? FL ?? XX XX ZZ ZZ WW WW DD DD ?? ?? ?? ?? ?? ?? 00 00++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x2C     CHAR Aot;            // Aot     UCHAR SCE;           // Id /* tagSCE_AOT     UCHAR SAT;           // Type     UCHAR nFloor;        // nFloor     UCHAR Super;         // Super     SHORT X;              // Position     SHORT Z;              // Position     USHORT W;             // Size     USHORT D;             // Size */ tagSCE_AOT     USHORT Data0;        // Sce_Message                         // byte0   ???                         // byte1   MSG Id                         //                         // Sce_Flg_chg                         // Flag Type                         // For example, if(Ck(ROOM, 0x01, OFF))                         // This variable would be 0x0005                         //                         // Sce_Event                         // Always 0x00FF                         //     USHORT Data1;        // Sce_Message                         // byte0   ???                         // byte1   ???                         //                         // Sce_Flg_chg                         // Flag Id                         // For example, if(Ck(ROOM, 0x01, OFF))                         // This variable would be 0x0001                         //                         // Sce_Event                         // byte0 Script Id Init                         // byte1 Script Id Complete                         //     USHORT Data2;        // Sce_Message                         // Always 0xFFFF                         // } Aot_set;              // 0x2C // 0x00 bytes // Aot_set This bytecode sets the properties of the specified AOT.                 </pre>	<p>08-02-2024 Newly Added</p>

Instruction Name	Length	Example / Info	History
{2D} Obj_model_set	38	<pre> 2D ID ?? 00 00++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x2D  // 0x2D     UCHAR MD1;           // MD1 File Id     UCHAR ModelId;       // ID of the object model     UCHAR Ccol_old;      // CC_WORK structure     UCHAR Ccol_no;       // CC_WORK structure     UCHAR Ctex_old;      // CC_WORK structure     UCHAR nFloor;        // Floor     UCHAR Super;         //     USHORT Type;         // Global-&gt;Ob_model[].Type     USHORT BeFlag;       // Global-&gt;Ob_model[].Be_flg     SHORT Attribute;     // Global-&gt;Ob_model[].Attribute     SHORT PosX;          // X position (2 bytes)     SHORT PosY;          // Y position (2 bytes)     SHORT PosZ;          // Z position (2 bytes)     SHORT DirX;          // X direction (2 bytes)     SHORT DirY;          // Y direction (2 bytes)     SHORT DirZ;          // Z direction (2 bytes)     SHORT AtariOffsetX;  // For Moveable Object     SHORT AtariOffsetY;  // For Moveable Object     SHORT AtariOffsetZ;  // For Moveable Object     SHORT AtariSizeX;    // For Moveable Object     SHORT AtariSizeY;    // For Moveable Object     SHORT AtariSizeZ;    // For Moveable Object } Obj_model_set; This bytecode sets the properties of the specified object model.                 </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{2E} Work_set	03	<p><b>2E ?? ??++</b></p> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x2E  // 0x2E     UCHAR Type;          // Type of Work Set to Select     UCHAR Entity;        // ID of Entity to select } Work_set; This bytecode sets the properties of the specified work (task).</pre>	08-02-2024 Newly Added
{2F} Speed_set	04	<p><b>2F ID ?? 00++</b></p> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x2F  // 0x2F     UCHAR SpeedId;       // ID of the speed setting     USHORT SpeedValue;   // Value of the speed setting } Speed_set; This bytecode sets the specified speed setting.</pre>	08-02-2024 Newly Added
{30} Add_speed	01	<p><b>30++</b></p> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x30  // 0x30 } Add_speed; This bytecode increments the speed setting.</pre>	08-02-2024 Newly Added
{31} Add_aspeed	01	<p><b>31++</b></p> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x31  // 0x31 } Add_aspeed; This bytecode increments the angular speed setting.</pre>	08-02-2024 Newly Added
{32} Pos_set	08	<p><b>32 00 XX XX YY YY ZZ ZZ++</b></p> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x32  // 0x32     UCHAR zAlign;        // Always Zero (Alignment byte)     SHORT PosX;          // X position (2 bytes)     SHORT PosY;          // Y position (2 bytes)     SHORT PosZ;          // Z position (2 bytes) } Pos_set; This bytecode sets the position in 3D space.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{33} Dir_set	08	<p><b>33 00 RX RX RY RY RZ RZ++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x33 // 0x33     UCHAR zAlign; // Always Zero (Alignment byte)     SHORT DirX; // X direction (2 bytes)     SHORT DirY; // Y direction (2 bytes)     SHORT DirZ; // Z direction (2 bytes) } Dir_set; This bytecode sets the direction in 3D space.</pre>	08-02-2024 Newly Added
{34} Member_set	04	<p><b>34 ID ?? ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x34 // 0x34     UCHAR Destination; // Set written from     SHORT Source; // Memory to be written to } Member_set; This bytecode sets the properties of the specified member.</pre>	08-02-2024 Newly Added
{35} Member_set2	03	<p><b>35 ID ??++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x34 // 0x34     UCHAR Destination; // Memory to be written to     UCHAR Source; // Set written from } Member_set2; This bytecode sets a single property of the specified member.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{36} Se_on	12	<pre> 36 ID ?? ?? ?? ?? XX XX YY YY ZZ ZZ++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x36   // 0x36     UCHAR SeId;          // ID of the sound effect     UCHAR Volume;       // Volume of the sound effect     SHORT data0;        // Sound Reverberation, Work Aot/Obj No     SHORT PosX;         // X position (2 bytes)     SHORT PosY;         // Y position (2 bytes)     SHORT PosZ;         // Z position (2 bytes) } Se_on; This bytecode plays the specified sound effect with the given properties.                     </pre>	08-02-2024 Newly Added
{37} Sca_id_set	04	<pre> 37 ID ?? ?? 00++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x37   // 0x37     UCHAR ScaId;         // ID of the Boundary     USHORT Id;           // New Collision Height } Sca_id_set; This bytecode turns a boundary on or off.                     </pre>	08-02-2024 Newly Added
{38} Flr_set	03	<pre> 38 ID ??++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x38   // 0x38     UCHAR FlrId;         // ID of the floor     UCHAR Flag; } Flr_set; This bytecode sets the height of the specified floor.                     </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{39} Dir_ck	08	<p><b>39 RX RX RY RY RZ RZ 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x39 // 0x39     UCHAR zAlign; // Always Zero (Alignment byte)     USHORT DirX; // X direction to check (2 bytes)     USHORT DirY; // Y direction to check (2 bytes)     USHORT DirZ; // Z direction to check (2 bytes) } Dir_ck; This bytecode checks the direction in 3D space.</pre>	08-02-2024 Newly Added
{3A} Sce_espr_on	16	<p><b>3A 00 ?? ?? ?? ?? ?? ?? XX XX YY YY ZZ ZZ DY DY++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x3A     UCHAR zAlign;     USHORT data0;     USHORT data1;     USHORT data2;     SHORT PosX; // X position (2 bytes)     SHORT PosY; // Y position (2 bytes)     SHORT PosZ; // Z position (2 bytes)     SHORT DirY; // Y direction to check (2 bytes) } Sce_espr_on; This bytecode activates the specified effect sprite with the given properties.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{3B} Door_aot_set	32	<pre> 3B ID ?? typedef struct {          // Ptr          // Description     UCHAR Opcode;        // 0x3B      // 0x3B     UCHAR AotId;         // 0x01      // Index ID     UCHAR SCE;          // 0x02      // Dummy     UCHAR SAT;          // 0x03      // Dummy     UCHAR nFloor;       // 0x04      // nFloor     UCHAR Super;        // 0x05      // Attach Function     SHORT X;            // 0x06      // X Position     SHORT Z;            // 0x08      // Z Position     USHORT W;           // 0x0A      // X/Z Width     USHORT D;           // 0x0C      // Y Width     SHORT Next_pos_x;   // 0x0E      // X coordinates in next room     SHORT Next_pos_y;   // 0x10      // Y coordinates in next room     SHORT Next_pos_z;   // 0x12      // Z coordinates in next room     SHORT Next_cdir_y;  // 0x14      // Rotation in next room     UCHAR Next_stage;   // 0x16      // Next room Stage     UCHAR Next_room;    // 0x17      // Next room Number     UCHAR Next_cut;     // 0x18      // Next room amera angle     UCHAR Next_nfloor;  // 0x19      // Next room Floor     UCHAR Dtex_type;    // 0x1A      // Door Type     UCHAR Door_type;    // 0x1B      // D02 Animation Set     UCHAR Knock_type;   // 0x1C      // Dummy     UCHAR Key_id;       // 0x1D      // Lock Flag     UCHAR Key_type;     // 0x1E      // Key Required     UCHAR Free;         // 0x1F      // Dummy } Door_aot_set;                 </pre> <p>This bytecode sets the properties of the specified door, including position, rotation, and lock status.</p>	<p>08-02-2024 Newly Added</p>

Instruction Name	Length	Example / Info	History
{3C} Cut_auto	02	<p><b>3C ID++</b>                      typedef struct { // Ptr //                      Description                      UCHAR Opcode; // 0x3C //                      0x3C                      UCHAR OnOff; // Set whether                      the camera changes automatically when                      hitting switch zone or not.                      } Cut_auto;                      This bytecode starts the specified cutscene                      automatically.</p>	08-02-2024 Newly Added
{3D} Member_copy	03	<p><b>3D ID ID++</b>                      typedef struct { // Ptr //                      Description                      UCHAR Opcode; // 0x3D                      // 0x3D                      CHAR DestinationId; // ID of                      the destination member                      CHAR SourceMemberId; // ID of the                      source member                      } Member_copy;                      This bytecode copies the properties from the source                      member to the destination member.</p>	08-02-2024 Newly Added
{3E} Member_cmp	06	<p><b>3E ?? ?? ?? ?? ??++</b>                      typedef struct { // Ptr //                      Description                      UCHAR Opcode; // 0x3E //                      0x3E                      UCHAR zAlign;                      UCHAR Flag; // System flag to                      compare                      UCHAR Operator; // Comparison                      operator                      SHORT Value; // Value to                      compare against                      } Member_cmp;                      This bytecode compares the specified property of                      the member with the given value using the specified                      comparison type.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{3F} Plc_motion	04	<p><b>3F ID ?? ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x3F // 0x3F     UCHAR MotionId; // ID of the motion to play     UCHAR Speed; // Speed of the motion     UCHAR Loop; // Loop flag (0 = no loop, 1 = loop)     UCHAR zAlign; // Always Zero (Alignment byte) } Plc_motion; This bytecode sets the specified motion to play at the given speed with the optional loop flag.</pre>	08-02-2024 Newly Added
{40} Plc_dest	08	<p><b>40 00 ?? ?? XX XX ZZ ZZ++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x40 // 0x40     UCHAR zAlign; // Always Zero (Alignment byte)     UCHAR Animation; // EDD/EMR Id     UCHAR Bit; // Room_flg     SHORT X;     SHORT Z; // Destination } Plc_dest; This bytecode sets the destination position in 3D space.</pre>	08-02-2024 Newly Added
{41} Plc_neck	10	<p><b>41 ID XX XX YY YY ZZ ZZ ?? ??++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x41 // 0x41     UCHAR NeckId; // ID of the neck motion     SHORT PosX; // X position of the neck motion (2 bytes)     SHORT PosY; // Y position of the neck motion (2 bytes)     SHORT PosZ; // Z position of the neck motion (2 bytes)     UCHAR SpeedX;     UCHAR SpeedZ; } Plc_neck; This bytecode sets the specified neck motion with the given position and rotation properties.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{42} Plc_ret	01	<p><b>42++</b>                      typedef struct { // Ptr //                      Description                      UCHAR Opcode; // 0x42 //                      0x42                      } Plc_ret;                      This bytecode returns control from the current motion or behavior.</p>	08-02-2024 Newly Added
{43} Plc_flg	04	<p><b>43 ?? ?? 00++</b>                      typedef struct { // Ptr //                      Description                      UCHAR Opcode; // 0x43 //                      0x43                      UCHAR Type; //                      USHORT Flag; //                      } Plc_flg;                      This bytecode sets the specified flag to the given value.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{44} Sce_em_set	22	<pre> 44 ??++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x44    // 0x44     UCHAR Nop;           // 0x01    // 0x00     CHAR EmId;           // SCE Index ID of the enemy or entity     UCHAR EMD;           // EMD File to load     UCHAR Animation      //     UCHAR AI Related     //     UCHAR nFloor;        // nFloor     UCHAR Soundbank      //     UCHAR Texture        // Texture selector for EMIF     UCHAR Flag           // Enemy Flag     USHORT PosX;         // X position (2 bytes)     USHORT PosY;         // Y position (2 bytes)     USHORT PosZ;         // Z position (2 bytes)     UCHAR RotationX;     // X rotation     UCHAR Speed;         // Movement speed     UCHAR Animation #    //     UCHAR Anim Block     //     UCHAR Anim Execution//     UCHAR zAlign;        // Always Zero (Alignment bytes) } Sce_em_set; This bytecode sets the specified enemy or entity with the given position, rotation, speed, and health properties.                     </pre>	08-02-2024 Newly Added
{45} Col_chg_set	05	<pre> 45 ?? ?? ?? ?? ??++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x45    // 0x45     UCHAR R;             // Red color value     UCHAR G;             // Green color value     UCHAR B;             // Blue color value     UCHAR Alpha;         // Alpha transparency value } Col_chg_set; This bytecode sets the specified color change properties.                     </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{46} Aot_reset	10	<pre>46 ID 00 00 00 00 00 00 00 00 00++ typedef struct { // Ptr // Description     UCHAR Opcode; // 0x46 // 0x46     UCHAR AotId; // ID of the AOT to reset     UCHAR SCE;     UCHAR SAT; // Scenario Atari     SHORT Data0;     SHORT Data1;     SHORT Data2; } Aot_reset; This bytecode resets the specified AOT to its default state.</pre>	08-02-2024 Newly Added
{47} Aot_on	02	<pre>47 ID++ typedef struct { // Ptr // Description     UCHAR Opcode; // 0x47 // 0x47     CHAR AotId; // ID of the AOT to activate } Aot_on; This bytecode activates the specified AOT.</pre>	08-02-2024 Newly Added
{48} Super_set	16	<pre>48 ID XX XX YY YY ZZ ZZ RX RY RZ ?? ?? ?? ?? 00 00++ typedef struct { // Ptr // Description     UCHAR Opcode; // 0x48 // 0x48     UCHAR SuperId; // ID of the super effect     USHORT PosX; // X position (2 bytes)     USHORT PosY; // Y position (2 bytes)     USHORT PosZ; // Z position (2 bytes)     UCHAR ScaleX; // X scale     UCHAR ScaleY; // Y scale     UCHAR Rotation; // Rotation value     UCHAR Alpha; // Alpha transparency value     ULONG Duration; // Duration of the effect (4 bytes)     UCHAR zAlign[2]; // Always Zero (Alignment bytes) } Super_set; This bytecode sets the specified super effect with the given properties.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{49} Super_reset	08	<p><b>49 ID 00 00 00 00 00 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x49 // 0x49     UCHAR SuperId; // ID of the super effect to reset     UCHAR zAlign[7]; // Always Zero (Alignment bytes) } Super_reset; This bytecode resets the specified super effect to its default state.</pre>	08-02-2024 Newly Added
{4A} Plc_gun	02	<p><b>4A ID++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x4A // 0x4A     UCHAR GunId; // ID of the gun to equip } Plc_gun; This bytecode equips the specified gun.</pre>	08-02-2024 Newly Added
{4B} Cut_replace	03	<p><b>4B ?? ID++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x4B // 0x4B     UCHAR OldCutId; // ID of the cutscene to replace     UCHAR NewCutId; // ID of the new cutscene } Cut_replace; This bytecode replaces the specified cutscene with a new cutscene.</pre>	08-02-2024 Newly Added
{4C} Sce_espr_kill	05	<p><b>4C ID XX XX YY YY ZZ ZZ++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x4C // 0x4C     UCHAR EsprId; // ID of the effect sprite to kill     USHORT PosX; // X position of the effect sprite (2 bytes)     USHORT PosY; // Y position of the effect sprite (2 bytes)     USHORT PosZ; // Z position of the effect sprite (2 bytes) } Sce_espr_kill; This bytecode kills the specified effect sprite at the given position.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{4D} Door_model_set	22	<pre> 4D ID XX XX YY YY ZZ ZZ RX RY RZ ?? ?? ?? ?? ?? ?? 00 00++ typedef struct { // Ptr // Description     UCHAR Opcode; // 0x4D // 0x4D     UCHAR ModelId; // ID of the door model     USHORT PosX; // X position of the door model (2 bytes)     USHORT PosY; // Y position of the door model (2 bytes)     USHORT PosZ; // Z position of the door model (2 bytes)     UCHAR RotationX; // X rotation of the door model     UCHAR RotationY; // Y rotation of the door model     UCHAR RotationZ; // Z rotation of the door model     UCHAR Speed; // Movement speed     UCHAR Health; // Health value     UCHAR zAlign[8]; // Always Zero (Alignment bytes) } Door_model_set; This bytecode sets the specified door model with the given properties.                     </pre>	08-02-2024 Newly Added
{4E} Trg_model_set	10	<pre> 4E ID XX XX YY YY ZZ ZZ ?? ?? ?? 00++ typedef struct { // Ptr // Description     UCHAR Opcode; // 0x4E // 0x4E     UCHAR TrgId; // ID of the target model     USHORT PosX; // X position of the target model (2 bytes)     USHORT PosY; // Y position of the target model (2 bytes)     USHORT PosZ; // Z position of the target model (2 bytes)     UCHAR Scale; // Scale value     UCHAR Alpha; // Alpha transparency value     UCHAR zAlign[2]; // Always Zero (Alignment bytes) } Trg_model_set; This bytecode sets the specified target model with the given properties.                     </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{4F} Plc_gun_equip	02	<b>4F ID++</b> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x4F  // 0x4F     UCHAR GunId;         // ID of the gun to equip } Plc_gun_equip; This bytecode equips the specified gun.</pre>	08-02-2024 Newly Added
{50} Pos_reset	08	<b>50 XX XX YY YY ZZ ZZ 00++</b> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x50  // 0x50     USHORT PosX;         // X position to reset to (2 bytes)     USHORT PosY;         // Y position to reset to (2 bytes)     USHORT PosZ;         // Z position to reset to (2 bytes)     UCHAR zAlign;        // Always Zero (Alignment byte) } Pos_reset; This bytecode resets the position to the specified coordinates.</pre>	08-02-2024 Newly Added
{51} Member_hide	02	<b>51 ID++</b> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x51  // 0x51     UCHAR MemberId;     // ID of the member to hide } Member_hide; This bytecode hides the specified member.</pre>	08-02-2024 Newly Added
{52} Member_show	02	<b>52 ID++</b> <pre>typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x52  // 0x52     UCHAR MemberId;     // ID of the member to show } Member_show; This bytecode shows the specified member.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{53} Sce_work_set	05	<p><b>53 ?? ?? ?? ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x53 // 0x53     UCHAR WorkId; // ID of the work     UCHAR Parameter1; // Parameter 1 for the work     UCHAR Parameter2; // Parameter 2 for the work     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_work_set; This bytecode sets the specified work with the given parameters.</pre>	08-02-2024 Newly Added
{54} Sce_trg_chk	08	<p><b>54 ID XX XX YY YY ZZ ZZ 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x54 // 0x54     UCHAR TrgId; // ID of the target     USHORT PosX; // X position of the target (2 bytes)     USHORT PosY; // Y position of the target (2 bytes)     USHORT PosZ; // Z position of the target (2 bytes)     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_trg_chk; This bytecode checks the specified target at the given position.</pre>	08-02-2024 Newly Added
{55} Sce_trg_on	04	<p><b>55 ID ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x55 // 0x55     UCHAR TrgId; // ID of the target     UCHAR State; // State to set the target to     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_trg_on; This bytecode sets the state of the specified target.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{56} Sce_trg_off	02	<b>56 ID++</b> typedef struct { // Ptr // Description UCHAR Opcode; // 0x56 // 0x56 UCHAR TrgId; // ID of the target } Sce_trg_off; This bytecode turns off the specified target.	08-02-2024 Newly Added
{57} Sce_trg_reset	02	<b>57 ID++</b> typedef struct { // Ptr // Description UCHAR Opcode; // 0x57 // 0x57 UCHAR TrgId; // ID of the target } Sce_trg_reset; This bytecode resets the specified target.	08-02-2024 Newly Added
{58} Sce_esp_off	02	<b>58 ID++</b> typedef struct { // Ptr // Description UCHAR Opcode; // 0x58 // 0x58 UCHAR EsprId; // ID of the effect sprite } Sce_esp_off; This bytecode turns off the specified effect sprite.	08-02-2024 Newly Added
{59} Trg_speed_set	04	<b>59 ID ?? 00++</b> typedef struct { // Ptr // Description UCHAR Opcode; // 0x59 // 0x59 UCHAR TrgId; // ID of the target UCHAR Speed; // Speed to set UCHAR zAlign; // Always Zero (Alignment byte) } Trg_speed_set; This bytecode sets the speed of the specified target.	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{5A} Sce_espr_copy	05	<p><b>5A ID XX XX YY YY ZZ ZZ++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x5A // 0x5A     UCHAR EsprId; // ID of the effect sprite to copy     USHORT PosX; // X position of the effect sprite (2 bytes)     USHORT PosY; // Y position of the effect sprite (2 bytes)     USHORT PosZ; // Z position of the effect sprite (2 bytes) } Sce_espr_copy; This bytecode copies the specified effect sprite at the given position.</pre>	08-02-2024 Newly Added
{5B} Em_door_set	24	<p><b>5B ID XX XX YY YY ZZ ZZ RX RY RZ ?? ?? ?? ?? ?? ??</b> <b>?? ?? ?? ?? ?? ?? 00 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x5B // 0x5B     UCHAR EmId; // ID of the enemy or entity     USHORT PosX; // X position of the enemy or entity (2 bytes)     USHORT PosY; // Y position of the enemy or entity (2 bytes)     USHORT PosZ; // Z position of the enemy or entity (2 bytes)     UCHAR RotationX; // X rotation     UCHAR RotationY; // Y rotation     UCHAR RotationZ; // Z rotation     UCHAR Speed; // Movement speed     UCHAR Health; // Health value     UCHAR zAlign[10]; // Always Zero (Alignment bytes) } Em_door_set; This bytecode sets the specified enemy or entity with the given properties.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{5C} Sce_item_set	12	<pre> 5C ID ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? 00++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x5C  // 0x5C     UCHAR ItemId;        // ID of the item     UCHAR Quantity;      // Quantity of the item     UCHAR Property1;     // Property 1 of the item     UCHAR Property2;     // Property 2 of the item     UCHAR Property3;     // Property 3 of the item     UCHAR Property4;     // Property 4 of the item     UCHAR Property5;     // Property 5 of the item     UCHAR Property6;     // Property 6 of the item     UCHAR Property7;     // Property 7 of the item     UCHAR Property8;     // Property 8 of the item     UCHAR zAlign;        // Always Zero (Alignment byte) } Sce_item_set; This bytecode sets the specified item with the given properties.                     </pre>	08-02-2024 Newly Added
{5D} Sce_item_reset	04	<pre> 5D ID ?? ??++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x5D  // 0x5D     UCHAR ItemId;        // ID of the item     UCHAR Quantity;      // Quantity of the item     UCHAR zAlign;        // Always Zero (Alignment byte) } Sce_item_reset; This bytecode resets the specified item with the given quantity.                     </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{5E} Sce_work_chk	04	<p><b>5E ID ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x5E // 0x5E     UCHAR WorkId; // ID of the work     UCHAR State; // State to check     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_work_chk; This bytecode checks the state of the specified work.</pre>	08-02-2024 Newly Added
{5F} Sce_work_on	04	<p><b>5F ID ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x5F // 0x5F     UCHAR WorkId; // ID of the work     UCHAR State; // State to set the work to     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_work_on; This bytecode sets the state of the specified work.</pre>	08-02-2024 Newly Added
{60} Sce_work_off	02	<p><b>60 ID++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x60 // 0x60     UCHAR WorkId; // ID of the work } Sce_work_off; This bytecode turns off the specified work.</pre>	08-02-2024 Newly Added
{61} Sce_work_reset	02	<p><b>61 ID++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x61 // 0x61     UCHAR WorkId; // ID of the work } Sce_work_reset; This bytecode resets the specified work.</pre>	08-02-2024 Newly Added
{62} Sce_item_off	02	<p><b>62 ID++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x62 // 0x62     UCHAR ItemId; // ID of the item } Sce_item_off; This bytecode turns off the specified item.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{63} Sce_item_on	04	<p><b>63 ID ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x63 // 0x63     UCHAR ItemId; // ID of the item     UCHAR State; // State to set the item to     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_item_on; This bytecode sets the state of the specified item.</pre>	08-02-2024 Newly Added
{64} Sce_esp_reset	02	<p><b>64 ID++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x64 // 0x64     UCHAR EsprId; // ID of the effect sprite } Sce_esp_reset; This bytecode resets the specified effect sprite.</pre>	08-02-2024 Newly Added
{65} Sce_eff_set	12	<p><b>65 ID XX XX YY YY ZZ ZZ RX RY RZ ?? ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x65 // 0x65     UCHAR EffId; // ID of the effect     USHORT PosX; // X position of the effect (2 bytes)     USHORT PosY; // Y position of the effect (2 bytes)     USHORT PosZ; // Z position of the effect (2 bytes)     UCHAR RotationX; // X rotation of the effect     UCHAR RotationY; // Y rotation of the effect     UCHAR RotationZ; // Z rotation of the effect     UCHAR Scale; // Scale value     UCHAR Alpha; // Alpha transparency value     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_eff_set; This bytecode sets the specified effect with the given properties.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{66} Sce_eff_reset	02	<p><b>66 ID++</b>                      typedef struct { // Ptr //                      Description                      UCHAR Opcode; // 0x66 //                      0x66                      UCHAR EffId; // ID of the                      effect                      } Sce_eff_reset;                      This bytecode resets the specified effect.</p>	08-02-2024 Newly Added
{67} Sce_trg_copy	05	<p><b>67 ID XX XX YY YY ZZ ZZ++</b>                      typedef struct { // Ptr //                      Description                      UCHAR Opcode; // 0x67 //                      0x67                      UCHAR TrgId; // ID of the                      target to copy                      USHORT PosX; // X position of                      the target (2 bytes)                      USHORT PosY; // Y position of                      the target (2 bytes)                      USHORT PosZ; // Z position of                      the target (2 bytes)                      } Sce_trg_copy;                      This bytecode copies the specified target at the                      given position.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{68} Door_aot_set_4p	40	<p>68 ID XX XX YY YY ZZ ZZ RX RY RZ ?? ?? ?? ?? ?? ??                      ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? 00                      00++</p> <pre>                     typedef struct { // Ptr //                     Description                         UCHAR Opcode; // 0x68 //                     0x68                         UCHAR DoorId; // ID of the                     door                         USHORT PosX; // X position of                     the door (2 bytes)                         USHORT PosY; // Y position of                     the door (2 bytes)                         USHORT PosZ; // Z position of                     the door (2 bytes)                         UCHAR RotationX; // X rotation of                     the door                         UCHAR RotationY; // Y rotation of                     the door                         UCHAR RotationZ; // Z rotation of                     the door                         UCHAR ScaleX; // X scale of                     the door                         UCHAR ScaleY; // Y scale of                     the door                         UCHAR zAlign[28]; // Always Zero                     (Alignment bytes)                     } Door_aot_set_4p;                     </pre> <p>This bytecode sets the properties of the specified door with position, rotation, and scale values.</p>	08-02-2024 Newly Added
{69} Trg_init	03	<p>69 ID ??++</p> <pre>                     typedef struct { // Ptr //                     Description                         UCHAR Opcode; // 0x69 //                     0x69                         UCHAR TrgId; // ID of the                     target to initialize                         UCHAR State; // Initial state                     of the target                     } Trg_init;                     </pre> <p>This bytecode initializes the specified target with the given state.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{6A} Dir_set2	06	<p><b>6A RX RX RY RY RZ RZ++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x6A // 0x6A     USHORT DirX; // X direction (2 bytes)     USHORT DirY; // Y direction (2 bytes)     USHORT DirZ; // Z direction (2 bytes) } Dir_set2; This bytecode sets the direction in 3D space.</pre>	08-02-2024 Newly Added
{6B} Col_set2	05	<p><b>6B ?? ?? ?? ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x6B // 0x6B     UCHAR R; // Red color value     UCHAR G; // Green color value     UCHAR B; // Blue color value     UCHAR Alpha; // Alpha transparency value } Col_set2; This bytecode sets the specified color values.</pre>	08-02-2024 Newly Added
{6C} Sce_eff_on	12	<p><b>6C ID XX XX YY YY ZZ ZZ RX RY RZ ?? ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x6C // 0x6C     UCHAR EffId; // ID of the effect     USHORT PosX; // X position of the effect (2 bytes)     USHORT PosY; // Y position of the effect (2 bytes)     USHORT PosZ; // Z position of the effect (2 bytes)     UCHAR RotationX; // X rotation of the effect     UCHAR RotationY; // Y rotation of the effect     UCHAR RotationZ; // Z rotation of the effect     UCHAR Scale; // Scale value     UCHAR Alpha; // Alpha transparency value     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_eff_on; This bytecode activates the specified effect with the given properties.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{6D} Eff_aot_set	40	<p>6D ID XX YY ZZ ZZ RX RY RZ ?? ?? ?? ?? ?? ??                      ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? 00                      00++</p> <pre>                     typedef struct {          // Ptr          //                     Description                         UCHAR Opcode;          // 0x6D          //                     0x6D                         UCHAR EffId;           // ID of the                     effect                         USHORT PosX;          // X position of                     the effect (2 bytes)                         USHORT PosY;          // Y position of                     the effect (2 bytes)                         USHORT PosZ;          // Z position of                     the effect (2 bytes)                         UCHAR RotationX;       // X rotation of                     the effect                         UCHAR RotationY;       // Y rotation of                     the effect                         UCHAR RotationZ;       // Z rotation of                     the effect                         UCHAR ScaleX;          // X scale of                     the effect                         UCHAR ScaleY;          // Y scale of                     the effect                         UCHAR zAlign[28];      // Always Zero                     (Alignment bytes)                     } Eff_aot_set;                 </pre> <p>This bytecode sets the specified effect with the given properties.</p>	08-02-2024 Newly Added
{6E} Col_set3	05	<p>6E ?? ?? ?? ?? 00++</p> <pre>                     typedef struct {          // Ptr          //                     Description                         UCHAR Opcode;          // 0x6E          //                     0x6E                         UCHAR R;              // Red color value                         UCHAR G;              // Green color value                         UCHAR B;              // Blue color value                         UCHAR Alpha;          // Alpha                     transparency value                     } Col_set3;                 </pre> <p>This bytecode sets the specified color values.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{6F} Item_aot_set	40	<pre> 6F ID XX XX YY YY ZZ ZZ RX RY RZ ?? 00 00++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x6F    // 0x6F     UCHAR ItemId;        // ID of the item     USHORT PosX;         // X position of the item (2 bytes)     USHORT PosY;         // Y position of the item (2 bytes)     USHORT PosZ;         // Z position of the item (2 bytes)     UCHAR RotationX;     // X rotation of the item     UCHAR RotationY;     // Y rotation of the item     UCHAR RotationZ;     // Z rotation of the item     UCHAR ScaleX;        // X scale of the item     UCHAR ScaleY;        // Y scale of the item     UCHAR zAlign[28];    // Always Zero (Alignment bytes) } Item_aot_set; This bytecode sets the specified item with the given properties.                     </pre>	08-02-2024 Newly Added
{70} Member_speed_set	04	<pre> 70 ID ?? 00++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x70    // 0x70     UCHAR MemberId;      // ID of the member     UCHAR Speed;         // Speed to set     UCHAR zAlign;        // Always Zero (Alignment byte) } Member_speed_set; This bytecode sets the speed of the specified member.                     </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{71} Sce_item_reset2	04	<pre>71 ID ?? ??++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x71  // 0x71     UCHAR ItemId;        // ID of the item     UCHAR Quantity;      // Quantity of the item     UCHAR zAlign;        // Always Zero (Alignment byte) } Sce_item_reset2; This bytecode resets the specified item with the given quantity.</pre>	08-02-2024 Newly Added
{72} Member_scale_set	04	<pre>72 ID ?? 00++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x72  // 0x72     UCHAR MemberId;      // ID of the member     UCHAR Scale;         // Scale to set     UCHAR zAlign;        // Always Zero (Alignment byte) } Member_scale_set; This bytecode sets the scale of the specified member.</pre>	08-02-2024 Newly Added
{73} Pos_copy	06	<pre>73 XX XX YY YY ZZ ZZ++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x73  // 0x73     USHORT PosX;         // X position to copy (2 bytes)     USHORT PosY;         // Y position to copy (2 bytes)     USHORT PosZ;         // Z position to copy (2 bytes) } Pos_copy; This bytecode copies the specified position.</pre>	08-02-2024 Newly Added
{74} Sce_item_copy	04	<pre>74 ID ?? ??++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x74  // 0x74     UCHAR ItemId;        // ID of the item     UCHAR Quantity;      // Quantity of the item     UCHAR zAlign;        // Always Zero (Alignment byte) } Sce_item_copy; This bytecode copies the specified item with the given quantity.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{75} Item_ck	04	<p><b>75 ID ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x75 // 0x75     UCHAR ItemId; // ID of the item     UCHAR State; // State to check     UCHAR zAlign; // Always Zero (Alignment byte) } Item_ck; This bytecode checks the state of the specified item.</pre>	08-02-2024 Newly Added
{76} Member_dir_set	08	<p><b>76 RX RX RY RY RZ RZ 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x76 // 0x76     USHORT DirX; // X direction of the member (2 bytes)     USHORT DirY; // Y direction of the member (2 bytes)     USHORT DirZ; // Z direction of the member (2 bytes)     UCHAR zAlign; // Always Zero (Alignment byte) } Member_dir_set; This bytecode sets the direction of the specified member.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{77} Aot_pos_set	40	<pre> 77 ID XX YY ZZ ZZ RX RY RZ ?? 00 00++ typedef struct { // Ptr // Description     UCHAR Opcode; // 0x77 // 0x77     UCHAR AotId; // ID of the AOT     USHORT PosX; // X position of the AOT (2 bytes)     USHORT PosY; // Y position of the AOT (2 bytes)     USHORT PosZ; // Z position of the AOT (2 bytes)     UCHAR RotationX; // X rotation of the AOT     UCHAR RotationY; // Y rotation of the AOT     UCHAR RotationZ; // Z rotation of the AOT     UCHAR ScaleX; // X scale of the AOT     UCHAR ScaleY; // Y scale of the AOT     UCHAR zAlign[28]; // Always Zero (Alignment bytes) } Aot_pos_set; This bytecode sets the specified AOT with the given position, rotation, and scale properties.                     </pre>	08-02-2024 Newly Added
{78} Pos_set2	08	<pre> 78 XX XX YY YY ZZ ZZ 00++ typedef struct { // Ptr // Description     UCHAR Opcode; // 0x78 // 0x78     USHORT PosX; // X position (2 bytes)     USHORT PosY; // Y position (2 bytes)     USHORT PosZ; // Z position (2 bytes)     UCHAR zAlign; // Always Zero (Alignment byte) } Pos_set2; This bytecode sets the position in 3D space.                     </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{79} Pos_cmp	06	<p><b>79 XX XX YY YY ZZ ZZ++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x79 // 0x79     USHORT PosX; // X position to compare (2 bytes)     USHORT PosY; // Y position to compare (2 bytes)     USHORT PosZ; // Z position to compare (2 bytes) } Pos_cmp; This bytecode compares the specified position.</pre>	08-02-2024 Newly Added
{7A} Sce_work_copy	03	<p><b>7A ID ??++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x7A // 0x7A     UCHAR WorkId; // ID of the work     UCHAR State; // State to copy } Sce_work_copy; This bytecode copies the specified work with the given state.</pre>	08-02-2024 Newly Added
{7B} Sce_item_set2	12	<p><b>7B ID ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x7B // 0x7B     UCHAR ItemId; // ID of the item     UCHAR Quantity; // Quantity of the item     UCHAR Property1; // Property 1 of the item     UCHAR Property2; // Property 2 of the item     UCHAR Property3; // Property 3 of the item     UCHAR Property4; // Property 4 of the item     UCHAR Property5; // Property 5 of the item     UCHAR Property6; // Property 6 of the item     UCHAR Property7; // Property 7 of the item     UCHAR Property8; // Property 8 of the item     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_item_set2; This bytecode sets the specified item with the given properties.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{7C} Sce_eff_on2	12	<pre> 7C ID XX XX YY YY ZZ ZZ RX RY RZ ?? ?? 00++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x7C  // 0x7C     UCHAR EffId;         // ID of the effect     USHORT PosX;         // X position of the effect (2 bytes)     USHORT PosY;         // Y position of the effect (2 bytes)     USHORT PosZ;         // Z position of the effect (2 bytes)     UCHAR RotationX;     // X rotation of the effect     UCHAR RotationY;     // Y rotation of the effect     UCHAR RotationZ;     // Z rotation of the effect     UCHAR Scale;         // Scale value     UCHAR Alpha;         // Alpha transparency value     UCHAR zAlign;        // Always Zero (Alignment byte) } Sce_eff_on2;                 </pre> <p>This bytecode activates the specified effect with the given properties.</p>	08-02-2024 Newly Added
{7D} Trg_speed_reset	02	<pre> 7D ID++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x7D  // 0x7D     UCHAR TrgId;         // ID of the target } Trg_speed_reset;                 </pre> <p>This bytecode resets the speed of the specified target.</p>	08-02-2024 Newly Added
{7E} Member_color_set	05	<pre> 7E ID ?? ?? ?? ?? ??++ typedef struct {          // Ptr    // Description     UCHAR Opcode;        // 0x7E  // 0x7E     UCHAR MemberId;      // ID of the member     UCHAR R;              // Red color value     UCHAR G;              // Green color value     UCHAR B;              // Blue color value     UCHAR Alpha;         // Alpha transparency value } Member_color_set;                 </pre> <p>This bytecode sets the color values of the specified member.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{7F} Sce_item_color_set	05	<pre> 7F ID ?? ?? ?? ??++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x7F  // 0x7F     UCHAR ItemId;        // ID of the item     UCHAR R;             // Red color value     UCHAR G;             // Green color value     UCHAR B;             // Blue color value     UCHAR Alpha;         // Alpha transparency value } Sce_item_color_set; This bytecode sets the color values of the specified item.</pre>	08-02-2024 Newly Added
{80} Sce_trg_color_set	05	<pre> 80 ID ?? ?? ?? ??++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x80  // 0x80     UCHAR TrgId;         // ID of the target     UCHAR R;             // Red color value     UCHAR G;             // Green color value     UCHAR B;             // Blue color value     UCHAR Alpha;         // Alpha transparency value } Sce_trg_color_set; This bytecode sets the color values of the specified target.</pre>	08-02-2024 Newly Added
{81} Sce_work_color_set	05	<pre> 81 ID ?? ?? ?? ??++ typedef struct {          // Ptr  // Description     UCHAR Opcode;        // 0x81  // 0x81     UCHAR WorkId;        // ID of the work     UCHAR R;             // Red color value     UCHAR G;             // Green color value     UCHAR B;             // Blue color value     UCHAR Alpha;         // Alpha transparency value } Sce_work_color_set; This bytecode sets the color values of the specified work.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{82} Eff_copy	12	<p>82 ID XX XX YY YY ZZ ZZ RX RY RZ ?? ?? 00++</p> <pre> typedef struct { // Ptr // Description     UCHAR Opcode; // 0x82 // 0x82     UCHAR EffId; // ID of the effect to copy     USHORT PosX; // X position of the effect (2 bytes)     USHORT PosY; // Y position of the effect (2 bytes)     USHORT PosZ; // Z position of the effect (2 bytes)     UCHAR RotationX; // X rotation of the effect     UCHAR RotationY; // Y rotation of the effect     UCHAR RotationZ; // Z rotation of the effect     UCHAR Scale; // Scale value     UCHAR Alpha; // Alpha transparency value     UCHAR zAlign; // Always Zero (Alignment byte) } Eff_copy; This bytecode copies the specified effect with the given properties.</pre>	08-02-2024 Newly Added
{83} Sce_work_reset2	02	<p>83 ID++</p> <pre> typedef struct { // Ptr // Description     UCHAR Opcode; // 0x83 // 0x83     UCHAR WorkId; // ID of the work } Sce_work_reset2; This bytecode resets the specified work.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{84} Sce_trg_set	10	<p>84 ID XX XX YY YY ZZ ZZ ?? ?? ?? 00++</p> <pre> typedef struct { // Ptr // Description     UCHAR Opcode; // 0x84 // 0x84     UCHAR TrgId; // ID of the target     USHORT PosX; // X position of the target (2 bytes)     USHORT PosY; // Y position of the target (2 bytes)     USHORT PosZ; // Z position of the target (2 bytes)     UCHAR Scale; // Scale value     UCHAR Alpha; // Alpha transparency value     UCHAR zAlign[2]; // Always Zero (Alignment bytes) } Sce_trg_set; This bytecode sets the specified target with the given properties.</pre>	08-02-2024 Newly Added
{85} Sce_eff_ck	12	<p>85 ID XX XX YY YY ZZ ZZ RX RY RZ ?? ?? 00++</p> <pre> typedef struct { // Ptr // Description     UCHAR Opcode; // 0x85 // 0x85     UCHAR EffId; // ID of the effect     USHORT PosX; // X position of the effect (2 bytes)     USHORT PosY; // Y position of the effect (2 bytes)     USHORT PosZ; // Z position of the effect (2 bytes)     UCHAR RotationX; // X rotation of the effect     UCHAR RotationY; // Y rotation of the effect     UCHAR RotationZ; // Z rotation of the effect     UCHAR Scale; // Scale value     UCHAR Alpha; // Alpha transparency value     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_eff_ck; This bytecode checks the specified effect with the given properties.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{86} Sce_eff_kill	05	<p><b>86 ID XX YY ZZ ZZ++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x86 // 0x86     UCHAR EffId; // ID of the effect to kill     USHORT PosX; // X position of the effect (2 bytes)     USHORT PosY; // Y position of the effect (2 bytes)     USHORT PosZ; // Z position of the effect (2 bytes) } Sce_eff_kill; This bytecode kills the specified effect at the given position.</pre>	08-02-2024 Newly Added
{87} Sce_eff_reset	02	<p><b>87 ID++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x87 // 0x87     UCHAR EffId; // ID of the effect } Sce_eff_reset; This bytecode resets the specified effect.</pre>	08-02-2024 Newly Added
{88} Sce_work_set2	05	<p><b>88 ID ?? ?? ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x88 // 0x88     UCHAR WorkId; // ID of the work     UCHAR Parameter1; // Parameter 1 for the work     UCHAR Parameter2; // Parameter 2 for the work     UCHAR Parameter3; // Parameter 3 for the work     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_work_set2; This bytecode sets the specified work with the given parameters.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{89} Sce_work_ck2	04	<p><b>89 ID ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x89 // 0x89     UCHAR WorkId; // ID of the work     UCHAR State; // State to check     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_work_ck2; This bytecode checks the state of the specified work.</pre>	08-02-2024 Newly Added
{8A} Sce_eff_copy	12	<p><b>8A ID XX XX YY YY ZZ ZZ RX RY RZ ?? ?? 00++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x8A // 0x8A     UCHAR EffId; // ID of the effect to copy     USHORT PosX; // X position of the effect (2 bytes)     USHORT PosY; // Y position of the effect (2 bytes)     USHORT PosZ; // Z position of the effect (2 bytes)     UCHAR RotationX; // X rotation of the effect     UCHAR RotationY; // Y rotation of the effect     UCHAR RotationZ; // Z rotation of the effect     UCHAR Scale; // Scale value     UCHAR Alpha; // Alpha transparency value     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_eff_copy; This bytecode copies the specified effect with the given properties.</pre>	08-02-2024 Newly Added
{8B} Sce_work_reset3	02	<p><b>8B ID++</b></p> <pre>typedef struct { // Ptr // Description     UCHAR Opcode; // 0x8B // 0x8B     UCHAR WorkId; // ID of the work } Sce_work_reset3; This bytecode resets the specified work.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{8C} Sce_eff_set2	12	<p>8C ID XX XX YY YY ZZ ZZ RX RY RZ ?? ?? 00++</p> <pre> typedef struct { // Ptr // Description     UCHAR Opcode; // 0x8C // 0x8C     UCHAR EffId; // ID of the effect     USHORT PosX; // X position of the effect (2 bytes)     USHORT PosY; // Y position of the effect (2 bytes)     USHORT PosZ; // Z position of the effect (2 bytes)     UCHAR RotationX; // X rotation of the effect     UCHAR RotationY; // Y rotation of the effect     UCHAR RotationZ; // Z rotation of the effect     UCHAR Scale; // Scale value     UCHAR Alpha; // Alpha transparency value     UCHAR zAlign; // Always Zero (Alignment byte) } Sce_eff_set2;                 </pre> <p>This bytecode sets the specified effect with the given properties.</p>	08-02-2024 Newly Added
{8D} Sce_work_cmp	06	<p>8D ID ?? ?? ?? 00++</p> <pre> typedef struct { // Ptr // Description     UCHAR Opcode; // 0x8D // 0x8D     UCHAR WorkId; // ID of the work     UCHAR Property; // Property to compare     USHORT Value; // Value to compare against     UCHAR ComparisonType; // Type of comparison (e.g., equal, not equal) } Sce_work_cmp;                 </pre> <p>This bytecode compares the specified property of the work with the given value using the specified comparison type.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{8E} Sce_eff_set3	12	<pre> 8E ID XX XX YY YY ZZ ZZ RX RY RZ ?? ?? 00++ typedef struct {          // Ptr    // Description     UCHAR Opcode;         // 0x8E  // 0x8E     UCHAR EffId;          // ID of the effect     USHORT PosX;          // X position of the effect (2 bytes)     USHORT PosY;          // Y position of the effect (2 bytes)     USHORT PosZ;          // Z position of the effect (2 bytes)     UCHAR RotationX;     // X rotation of the effect     UCHAR RotationY;     // Y rotation of the effect     UCHAR RotationZ;     // Z rotation of the effect     UCHAR Scale;         // Scale value     UCHAR Alpha;         // Alpha transparency value     UCHAR zAlign;        // Always Zero (Alignment byte) } Sce_eff_set3;                 </pre> <p>This bytecode sets the specified effect with the given properties.</p>	08-02-2024 Newly Added

From: <https://classicremodification.com/> - **Classic RE Modification**

Permanent link: [https://classicremodification.com/doku.php?id=re2\\_opcodes&rev=1722671618](https://classicremodification.com/doku.php?id=re2_opcodes&rev=1722671618)

Last update: **2024/08/03 00:53**

