

Instruction Name	Length	Example / Info	History
{00} Nop	01	00++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x00 } Nop; This bytecode is used for alignment of 1 byte opcodes and ending Elself blocks.	08-02-2024 Newly Added
{01} Evt_end	02	01 00++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x01 UCHAR zAlign; // Always Zero } Evt_end; This bytecode ends the current Main/Sub script.	08-02-2024 Newly Added
{02} Evt_next	01	02++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x02 } Evt_next; This bytecode moves to the next event.	08-02-2024 Newly Added
{03} Evt_chain	02	03 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x03 UCHAR Data1; // Data } Evt_chain; This bytecode chains the next event.	08-02-2024 Newly Added
{04} Evt_exec	04	04 ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x04 UCHAR Data1; // Data UCHAR Data2; // Data UCHAR Data3; // Data } Evt_exec; This bytecode executes the event.	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{05} Evt_kill	02	<p>05 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x05 UCHAR Data1; // Data } Evt_kill; This bytecode kills the event.</p>	08-02-2024 Newly Added
{06} Ifel_ck	04	<p>06 ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x06 UCHAR zAlign; // Alignment byte USHORT data2; // Data } Ifel_ck; This bytecode checks the condition of an If-Else block.</p>	08-02-2024 Newly Added
{07} Else_ck	04	<p>07 ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x07 UCHAR zAlign; // Alignment byte USHORT data2; // Data } Else_ck; This bytecode checks the condition of an Else block.</p>	08-02-2024 Newly Added
{08} Endif	02	<p>08 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x08 UCHAR zAlign; // Alignment byte } Endif; This bytecode ends an If-Else block.</p>	08-02-2024 Newly Added
{09} Sleep	04	<p>09 ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x09 UCHAR Data1; // Data USHORT data2; // Data } Sleep; This bytecode pauses the event for a specified duration.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{0A} Sleeping	03	<p>0A ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x0A USHORT data2; // Data } Sleeping; This bytecode sets the sleeping state.</p>	08-02-2024 Newly Added
{0B} Wsleep	01	<p>0B++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x0B } Wsleep; This bytecode sets the wake sleep state.</p>	08-02-2024 Newly Added
{0C} Wsleeping	01	<p>0C++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x0C } Wsleeping; This bytecode sets the wake sleeping state.</p>	08-02-2024 Newly Added
{0D} For	06	<p>0D ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x0D UCHAR zAlign; // Alignment byte SHORT data2; // Data USHORT data4; // Data } For; This bytecode starts a For loop.</p>	08-02-2024 Newly Added
{0E} Next	02	<p>0E ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x0E UCHAR zAlign; // Alignment byte } Next; This bytecode ends a For loop.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{0F} While	04	<p>0F ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x0F UCHAR zAlign; // Alignment byte SHORT data1; // Data } While; This bytecode starts a While loop.</p>	08-02-2024 Newly Added
{10} Ewhile	02	<p>10 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x10 UCHAR data1; // Data } Ewhile; This bytecode ends a While loop.</p>	08-02-2024 Newly Added
{11} Do	04	<p>11 ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x11 UCHAR zAlign; // Alignment byte SHORT data2; // Data } Do; This bytecode starts a Do loop.</p>	08-02-2024 Newly Added
{12} Edwhile	02	<p>12 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x12 UCHAR data1; // Data } Edwhile; This bytecode ends a Do-While loop.</p>	08-02-2024 Newly Added
{13} Switch	04	<p>13 ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x13 UCHAR data1; // Data USHORT data2; // Data } Switch; This bytecode starts a Switch statement.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{14} Case	06	<pre>14 ?? ?? ?? ?? ?? ++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x14 UCHAR data1; // Data USHORT data2; // Data } Case; This bytecode defines a Case in a Switch statement.</pre>	08-02-2024 Newly Added
{15} Default	02	<pre>15 ?? ++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x15 UCHAR zAlign; // Alignment byte } Default; This bytecode defines the Default case in a Switch statement.</pre>	08-02-2024 Newly Added
{16} Eswitch	02	<pre>16 ?? ++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x16 UCHAR zAlign; // Alignment byte } Eswitch; This bytecode ends a Switch statement.</pre>	08-02-2024 Newly Added
{17} Goto	06	<pre>17 ?? ?? ?? ?? ?? ++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x17 UCHAR data1; // Data UCHAR data2; // Data UCHAR zAlign; // Alignment byte SHORT data4; // Data } Goto; This bytecode performs a Goto operation.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{18} Gosub	02	<p>18 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x18 UCHAR data1; // Data } Gosub; This bytecode performs a Gosub operation.</p>	08-02-2024 Newly Added
{19} Return	02	<p>19 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x19 UCHAR data[3]; // Data } Return; This bytecode returns from a subroutine.</p>	08-02-2024 Newly Added
{1A} Break	02	<p>1A ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x1A CHAR data1; // Data } Break; This bytecode breaks out of a loop.</p>	08-02-2024 Newly Added
{1B} For2	06	<p>1B ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x1B UCHAR zAlign0; // Alignment byte SHORT data2; // Data UCHAR zAlign1; // Alignment byte UCHAR data5; // Data } For2; This bytecode starts a secondary For loop.</p>	08-02-2024 Newly Added
{1C} Break_point	01	<p>1C++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x1C } Break_point; This bytecode sets a breakpoint.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{1D} Work_copy	04	<p>1D ?? ?? ??++</p> <pre>typedef struct { // Ptr // Description UCHAR Opcode; // 0x1D UCHAR Source; // Source UCHAR Destination; // Destination UCHAR Typecast; // Typecast } Work_copy; This bytecode copies work data.</pre>	08-02-2024 Newly Added
{1E} Nop1E	01	<p>1E++</p> <pre>typedef struct { // Ptr // Description UCHAR Opcode; // 0x1E } Nop1E; This bytecode is used for alignment of 1 byte opcodes and ending Elself blocks.</pre>	08-02-2024 Newly Added
{1F} Nop1F	01	<p>1F++</p> <pre>typedef struct { // Ptr // Description UCHAR Opcode; // 0x1F } Nop1F; This bytecode is used for alignment of 1 byte opcodes and ending Elself blocks.</pre>	08-02-2024 Newly Added
{20} Nop	01	<p>20++</p> <pre>typedef struct { // Ptr // Description UCHAR Opcode; // 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>21 ?? ?? ??++</p> <pre>typedef struct { // Ptr // Description UCHAR Opcode; // 0x21 UCHAR Flag; // System_flg, etc UCHAR Id; // Bit UCHAR OnOff; // On/Off } Ck; This bytecode checks a flag.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{22} Set	04	<p>22 ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x22 UCHAR Flag; // System_flg, etc UCHAR Id; // Bit UCHAR OnOff; // On/Off } Set; This bytecode sets a flag.</p>	08-02-2024 Newly Added
{23} Cmp	06	<p>23 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x23 UCHAR Flag; // Flag UCHAR Operator; // Operator SHORT Value; // Value } Cmp; This bytecode compares values.</p>	08-02-2024 Newly Added
{24} Save	04	<p>24 ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x24 UCHAR Destination; // Destination SHORT Source; // Source } Save; This bytecode saves data.</p>	08-02-2024 Newly Added
{25} Copy	03	<p>25 ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x25 UCHAR Source; // Source operand UCHAR Destination; // Destination operand } Copy; This bytecode copies a value from source to destination.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{26} Calc	06	<p>26 ?? ?? ?? ?? ?? ??++</p> <pre>typedef struct { // Ptr // Description UCHAR Opcode; // 0x26 UCHAR Operation; // Operation type UCHAR Operand1; // First operand UCHAR Operand2; // Second operand UCHAR Result; // Result operand UCHAR zAlign; // Alignment byte } Calc; This bytecode performs a calculation.</pre>	08-02-2024 Newly Added
{27} Calc2	04	<p>27 ?? ?? ?? ??++</p> <pre>typedef struct { // Ptr // Description UCHAR Opcode; // 0x27 UCHAR Operation; // Operation type UCHAR Operand; // Operand UCHAR Result; // Result operand } Calc2; This bytecode performs a secondary calculation.</pre>	08-02-2024 Newly Added
{28} Sce_rnd	01	<p>28++</p> <pre>typedef struct { // Ptr // Description UCHAR Opcode; // 0x28 } Sce_rnd; This bytecode generates a random number.</pre>	08-02-2024 Newly Added
{29} Cut_chg	02	<p>29 ??++</p> <pre>typedef struct { // Ptr // Description UCHAR Opcode; // 0x29 UCHAR CutsceneId; // Cutscene ID } Cut_chg; This bytecode changes the cutscene.</pre>	08-02-2024 Newly Added
{2A} Cut_old	01	<p>2A++</p> <pre>typedef struct { // Ptr // Description UCHAR Opcode; // 0x2A } Cut_old; This bytecode reverts to the old cutscene.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{2B} Message_on	06	<pre> 2B ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x2B UCHAR MessageId; // Message ID UCHAR PositionX[2]; // X position UCHAR PositionY[2]; // Y position UCHAR Duration; // Display duration } Message_on; This bytecode displays a message on the screen.</pre>	08-02-2024 Newly Added
{2C} Aot_set	20	<pre> 2C ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x2C UCHAR AotType; // AOT Type UCHAR Id; // ID UCHAR SCE; // SCE UCHAR Floor; // Floor UCHAR Super; // Super UCHAR X[2]; // X position UCHAR Z[2]; // Z position UCHAR W; // Width UCHAR D; // Depth UCHAR Flag[2]; // Flags UCHAR Path[2]; // Path UCHAR Direction; // Direction UCHAR Param[6]; // Parameters } Aot_set; This bytecode sets an AOT (Area of Trigger).</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{2D} Obj_model_set	38	<pre> 2D ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x2D UCHAR ObjId; // Object ID UCHAR ModelId; // Model ID UCHAR MotionId; // Motion ID UCHAR Unknown1; // Unknown UCHAR Unknown2; // Unknown UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR R[2]; // Rotation UCHAR Unknown3[26]; // Unknown } Obj_model_set; This bytecode sets an object model. </pre>	08-02-2024 Newly Added
{2E} Work_set	03	<pre> 2E ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x2E UCHAR WorkNo; // Work number UCHAR Value; // Value } Work_set; This bytecode sets a work value. </pre>	08-02-2024 Newly Added
{2F} Speed_set	04	<pre> 2F ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x2F UCHAR SpeedX; // Speed X UCHAR SpeedY; // Speed Y UCHAR SpeedZ; // Speed Z } Speed_set; This bytecode sets the speed of an object. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{30} Add_speed	01	<p>30++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x30 } Add_speed; This bytecode adds to the speed of an object.</p>	08-02-2024 Newly Added
{31} Add_aspeed	01	<p>31++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x31 } Add_aspeed; This bytecode adds to the angular speed of an object.</p>	08-02-2024 Newly Added
{32} Pos_set	08	<p>32 ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x32 UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR R[2]; // Rotation } Pos_set; This bytecode sets the position of an object.</p>	08-02-2024 Newly Added
{33} Dir_set	08	<p>33 ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x33 UCHAR DirX[2]; // Direction X UCHAR DirY[2]; // Direction Y UCHAR DirZ[2]; // Direction Z UCHAR Speed; // Speed } Dir_set; This bytecode sets the direction of an object.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{34} Member_set	04	<pre> 34 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x34 UCHAR MemberNo; // Member number UCHAR Value[2]; // Value UCHAR zAlign; // Alignment byte } Member_set; This bytecode sets a member value. </pre>	08-02-2024 Newly Added
{35} Member_set2	03	<pre> 35 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x35 UCHAR MemberNo; // Member number UCHAR Value; // Value } Member_set2; This bytecode sets a secondary member value. </pre>	08-02-2024 Newly Added
{36} Se_on	12	<pre> 36 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x36 UCHAR SeId; // Sound effect ID UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Volume; // Volume UCHAR Pan; // Pan UCHAR Pitch; // Pitch UCHAR zAlign[2]; // Alignment bytes } Se_on; This bytecode plays a sound effect. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{37} Sca_id_set	04	<pre> 37 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x37 UCHAR ScaleId; // Scale ID UCHAR X[2]; // X scale UCHAR Y[2]; // Y scale UCHAR Z[2]; // Z scale } Sca_id_set; This bytecode sets the scale of an object. </pre>	08-02-2024 Newly Added
{38} Flr_set	03	<pre> 38 ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x38 UCHAR FloorNo; // Floor number UCHAR Height; // Height UCHAR zAlign; // Alignment byte } Flr_set; This bytecode sets the floor height. </pre>	08-02-2024 Newly Added
{39} Dir_ck	08	<pre> 39 ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x39 UCHAR DirX[2]; // Direction X UCHAR DirY[2]; // Direction Y UCHAR DirZ[2]; // Direction Z UCHAR CheckType; // Check type UCHAR zAlign[2]; // Alignment bytes } Dir_ck; This bytecode checks the direction of an object. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{3A} Sce_espr_on	16	<pre> 3A ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x3A UCHAR EspType; // ESPR Type UCHAR Id; // ID UCHAR SCE; // SCE UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR zAlign[4]; // Alignment bytes } Sce_espr_on; This bytecode activates an ESPR. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{3B} Door_aot_set	32	<pre> 3B ?? typedef struct { // Ptr // Description UCHAR Opcode; // 0x3B UCHAR DoorId; // Door ID UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR KeyItem; // Key item required UCHAR NextStage[2]; // Next stage UCHAR Transition; // Transition effect UCHAR zAlign[17]; // Alignment bytes } Door_aot_set; This bytecode sets a door AOT (Area of Trigger). </pre>	08-02-2024 Newly Added
{3C} Cut_auto	02	<pre> 3C ???+ typedef struct { // Ptr // Description UCHAR Opcode; // 0x3C UCHAR CutsceneId; // Cutscene ID } Cut_auto; This bytecode automatically triggers a cutscene. </pre>	08-02-2024 Newly Added
{3D} Member_copy	03	<pre> 3D ?? ?? ??+ typedef struct { // Ptr // Description UCHAR Opcode; // 0x3D UCHAR Source; // Source member UCHAR Destination; // Destination member } Member_copy; This bytecode copies a member value. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{3E} Member_cmp	06	<pre> 3E ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x3E UCHAR MemberNo; // Member number UCHAR Operator; // Comparison operator UCHAR Value[2]; // Value UCHAR Result; // Comparison result } Member_cmp; This bytecode compares a member value. </pre>	08-02-2024 Newly Added
{3F} Plc_motion	04	<pre> 3F ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x3F UCHAR MotionId; // Motion ID UCHAR Speed; // Speed UCHAR zAlign[2]; // Alignment bytes } Plc_motion; This bytecode sets the motion of an object. </pre>	08-02-2024 Newly Added
{40} Plc_dest	08	<pre> 40 ?? ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x40 UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Speed; // Speed UCHAR zAlign[2]; // Alignment bytes } Plc_dest; This bytecode sets the destination of an object. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{41} Plc_neck	10	<pre> 41 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x41 UCHAR NeckId; // Neck ID UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Rotation; // Rotation UCHAR zAlign[4]; // Alignment bytes } Plc_neck; This bytecode sets the neck parameters of an object.</pre>	08-02-2024 Newly Added
{42} Plc_ret	01	<pre> 42++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x42 } Plc_ret; This bytecode returns the PLC state.</pre>	08-02-2024 Newly Added
{43} Plc_flg	04	<pre> 43 ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x43 UCHAR Flag; // Flag UCHAR Value; // Value UCHAR zAlign[2]; // Alignment bytes } Plc_flg; This bytecode sets a PLC flag.</pre>	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 UCHAR EmType; // Enemy type UCHAR Id; // ID UCHAR SCE; // SCE UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR Life; // Life UCHAR Speed; // Speed UCHAR zAlign[9]; // Alignment bytes } Sce_em_set; This bytecode sets an enemy. </pre>	08-02-2024 Newly Added
{45} Col_chg_set	05	<pre> 45 ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x45 UCHAR ColorR; // Red component UCHAR ColorG; // Green component UCHAR ColorB; // Blue component UCHAR zAlign; // Alignment byte } Col_chg_set; This bytecode changes the color settings. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{46} Aot_reset	10	<pre> 46 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x46 UCHAR AotType; // AOT Type UCHAR Id; // ID UCHAR SCE; // SCE UCHAR Floor; // Floor UCHAR Super; // Super UCHAR X[2]; // X position UCHAR Z[2]; // Z position UCHAR W; // Width UCHAR D; // Depth UCHAR Flag[2]; // Flags UCHAR Path[2]; // Path UCHAR Direction; // Direction UCHAR zAlign[3]; // Alignment bytes } Aot_reset; This bytecode resets an AOT (Area of Trigger). </pre>	08-02-2024 Newly Added
{47} Aot_on	02	<pre> 47 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x47 UCHAR AotId; // AOT ID } Aot_on; This bytecode activates an AOT. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{48} Super_set	16	<pre> 48 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x48 UCHAR SuperType; // Super type UCHAR Id; // ID UCHAR SCE; // SCE UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR zAlign[2]; // Alignment bytes } Super_set; This bytecode sets a super parameter. </pre>	08-02-2024 Newly Added
{49} Super_reset	08	<pre> 49 ?? ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x49 UCHAR SuperId; // Super ID UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR zAlign[2]; // Alignment bytes } Super_reset; This bytecode resets a super parameter. </pre>	08-02-2024 Newly Added
{4A} Plc_gun	02	<pre> 4A ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x4A UCHAR GunId; // Gun ID } Plc_gun; This bytecode sets the gun parameters. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{4B} Cut_replace	03	<pre> 4B ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x4B UCHAR OldCutId; // Old cutscene ID UCHAR NewCutId; // New cutscene ID UCHAR zAlign; // Alignment byte } Cut_replace; This bytecode replaces a cutscene. </pre>	08-02-2024 Newly Added
{4C} Sce_espr_kill	05	<pre> 4C ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x4C UCHAR EspType; // ESPR Type UCHAR Id; // ID UCHAR SCE; // SCE UCHAR zAlign; // Alignment byte } Sce_espr_kill; This bytecode kills an ESPR. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{4D} Door_model_set	22	<pre> 4D ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x4D UCHAR DoorId; // Door ID UCHAR ModelId; // Model ID UCHAR MotionId; // Motion ID UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR zAlign[10]; // Alignment bytes } Door_model_set; This bytecode sets a door model. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{4E} Item_aot_set	22	<pre> 4E ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x4E UCHAR ItemId; // Item ID UCHAR AotType; // AOT Type UCHAR Id; // ID UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR zAlign[10]; // Alignment bytes } Item_aot_set; This bytecode sets an item AOT. </pre>	08-02-2024 Newly Added
{4F} Sce_key_ck	04	<pre> 4F ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x4F UCHAR KeyId; // Key ID UCHAR Flag; // Flag UCHAR zAlign; // Alignment byte } Sce_key_ck; This bytecode checks for a key item. </pre>	08-02-2024 Newly Added
{50} Sce_trg_ck	04	<pre> 50 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x50 UCHAR TriggerId; // Trigger ID UCHAR Flag; // Flag UCHAR zAlign; // Alignment byte } Sce_trg_ck; This bytecode checks for a trigger. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{51} Sce_bgm_control	06	<p>51 ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x51 UCHAR BgmId; // BGM ID UCHAR Volume; // Volume UCHAR Pan; // Pan UCHAR Pitch; // Pitch UCHAR zAlign; // Alignment byte } Sce_bgm_control; This bytecode controls the background music.</p>	08-02-2024 Newly Added
{52} Sce_espr_control	06	<p>52 ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x52 UCHAR EspId; // ESPR ID UCHAR Command; // Command UCHAR Param1; // Parameter 1 UCHAR Param2; // Parameter 2 UCHAR zAlign; // Alignment byte } Sce_espr_control; This bytecode controls an ESPR.</p>	08-02-2024 Newly Added
{53} Sce_fade_set	06	<p>53 ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x53 UCHAR FadeType; // Fade type UCHAR Duration; // Duration UCHAR ColorR; // Red component UCHAR ColorG; // Green component UCHAR ColorB; // Blue component } Sce_fade_set; This bytecode sets a fade effect.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{54} Sce_espr3d_on	22	<pre>54 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x54 UCHAR EspType; // ESPR Type UCHAR Id; // ID UCHAR SCE; // SCE UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR zAlign[10]; // Alignment bytes } Sce_espr3d_on; This bytecode activates a 3D ESPR.</pre>	08-02-2024 Newly Added
{55} Member_calc	06	<pre>55 ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x55 UCHAR MemberNo; // Member number UCHAR Operation; // Operation type UCHAR Operand1; // First operand UCHAR Operand2; // Second operand UCHAR Result; // Result operand } Member_calc; This bytecode performs a calculation on a member value.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{56} Member_calc2	04	<pre>56 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x56 UCHAR MemberNo; // Member number UCHAR Operation; // Operation type UCHAR Result; // Result operand } Member_calc2; This bytecode performs a secondary calculation on a member value.</pre>	08-02-2024 Newly Added
{57} Sce_bgmtbl_set	08	<pre>57 ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x57 UCHAR BgmId; // BGM ID UCHAR Volume; // Volume UCHAR Pan; // Pan UCHAR Pitch; // Pitch UCHAR zAlign[3]; // Alignment bytes } Sce_bgmtbl_set; This bytecode sets the BGM table.</pre>	08-02-2024 Newly Added
{58} Plc_rot	04	<pre>58 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x58 UCHAR RotationX; // Rotation X UCHAR RotationY; // Rotation Y UCHAR RotationZ; // Rotation Z } Plc_rot; This bytecode sets the rotation of an object.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{59} Xa_on	04	<pre>59 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x59 UCHAR XaId; // XA ID UCHAR Volume; // Volume UCHAR Pan; // Pan UCHAR zAlign; // Alignment byte } Xa_on; This bytecode activates an XA sound.</pre>	08-02-2024 Newly Added
{5A} Weapon_chg	02	<pre>5A ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x5A UCHAR WeaponId; // Weapon ID } Weapon_chg; This bytecode changes the weapon.</pre>	08-02-2024 Newly Added
{5B} Plc_cnt	02	<pre>5B ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x5B UCHAR Count; // Count } Plc_cnt; This bytecode sets the PLC count.</pre>	08-02-2024 Newly Added
{5C} Sce_shake_on	03	<pre>5C ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x5C UCHAR ShakeType; // Shake type UCHAR Duration; // Duration UCHAR zAlign; // Alignment byte } Sce_shake_on; This bytecode activates a screen shake effect.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{5D} Mizu_div_set	02	<p>5D ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x5D UCHAR Division; // Division } Mizu_div_set; This bytecode sets the water division.</p>	08-02-2024 Newly Added
{5E} Keep_Item_ck	02	<p>5E ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x5E UCHAR ItemId; // Item ID } Keep_Item_ck; This bytecode checks for a kept item.</p>	08-02-2024 Newly Added
{5F} Xa_vol	02	<p>5F ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x5F UCHAR Volume; // Volume } Xa_vol; This bytecode sets the XA volume.</p>	08-02-2024 Newly Added
{60} Kage_set	14	<p>60 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x60 UCHAR KageType; // Shadow type UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR zAlign[2]; // Alignment bytes } Kage_set; This bytecode sets a shadow.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{61} Cut_be_set	04	<pre>61 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x61 UCHAR CutsceneId; // Cutscene ID UCHAR Flag; // Flag UCHAR zAlign[2]; // Alignment bytes } Cut_be_set; This bytecode sets a cutscene BE.</pre>	08-02-2024 Newly Added
{62} Sce_Item_lost	02	<pre>62 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x62 UCHAR ItemId; // Item ID } Sce_Item_lost; This bytecode sets an item as lost.</pre>	08-02-2024 Newly Added
{63} Plc_gun_eff	01	<pre>63++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x63 } Plc_gun_eff; This bytecode sets the gun effect.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{64} Sce_espr_on2	16	<pre> 64 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x64 UCHAR EspType; // ESPR Type UCHAR Id; // ID UCHAR SCE; // SCE UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR zAlign[4]; // Alignment bytes } Sce_espr_on2; This bytecode activates a secondary ESPR. </pre>	08-02-2024 Newly Added
{65} Sce_espr_kill2	02	<pre> 65 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x65 UCHAR EspId; // ESPR ID } Sce_espr_kill2; This bytecode kills a secondary ESPR. </pre>	08-02-2024 Newly Added
{66} Plc_stop	01	<pre> 66++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x66 } Plc_stop; This bytecode stops the PLC. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{67} Aot_set_4p	28	<pre> 67 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x67 UCHAR AotType; // AOT Type UCHAR Id; // ID UCHAR SCE; // SCE UCHAR Floor; // Floor UCHAR Super; // Super UCHAR X[2]; // X position UCHAR Z[2]; // Z position UCHAR W; // Width UCHAR D; // Depth UCHAR Flag[2]; // Flags UCHAR Path[2]; // Path UCHAR Direction; // Direction UCHAR Param[10]; // Parameters } Aot_set_4p; This bytecode sets an AOT (Area of Trigger) with 4 parameters. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{68} Door_aot_set_4p	40	<pre> 68 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x68 UCHAR DoorId; // Door ID UCHAR AotType; // AOT Type UCHAR Id; // ID UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR KeyItem; // Key item required UCHAR NextStage[2]; // Next stage UCHAR Transition; // Transition effect UCHAR Param[12]; // Parameters } Door_aot_set_4p; This bytecode sets a door AOT (Area of Trigger) with 4 parameters. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{69} Item_aot_set_4p	30	<pre> 69 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x69 UCHAR ItemId; // Item ID UCHAR AotType; // AOT Type UCHAR Id; // ID UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR Param[14]; // Parameters } Item_aot_set_4p; This bytecode sets an item AOT with 4 parameters.</pre>	08-02-2024 Newly Added
{6A} Light_pos_set	06	<pre> 6A ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x6A UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position } Light_pos_set; This bytecode sets the position of a light.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{6B} Light_kido_set	04	<pre> 6B ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x6B UCHAR Intensity; // Light intensity UCHAR ColorR; // Red component UCHAR ColorG; // Green component UCHAR ColorB; // Blue component } Light_kido_set; This bytecode sets the light intensity and color. </pre>	08-02-2024 Newly Added
{6C} Rbj_reset	01	<pre> 6C++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x6C } Rbj_reset; This bytecode resets the RBJ. </pre>	08-02-2024 Newly Added
{6D} Sce_scr_move	04	<pre> 6D ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x6D UCHAR ScreenX[2]; // Screen X position UCHAR ScreenY[2]; // Screen Y position } Sce_scr_move; This bytecode moves the screen position. </pre>	08-02-2024 Newly Added
{6E} Parts_set	06	<pre> 6E ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x6E UCHAR PartId; // Part ID UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position } Parts_set; This bytecode sets the parts. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{73} Mirror_set	08	<pre> 73 ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x73 UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Reflection; // Reflection type } Mirror_set; This bytecode sets a mirror. </pre>	08-02-2024 Newly Added
{74} Sce_fade_adjust	04	<pre> 74 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x74 UCHAR FadeType; // Fade type UCHAR Duration; // Duration UCHAR ColorR; // Red component UCHAR ColorG; // Green component UCHAR ColorB; // Blue component } Sce_fade_adjust; This bytecode adjusts the fade effect. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{75} Sce_espr3d_on2	22	<pre> 75 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x75 UCHAR EspType; // ESPR Type UCHAR Id; // ID UCHAR SCE; // SCE UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR zAlign[10]; // Alignment bytes } Sce_espr3d_on2; This bytecode activates a secondary 3D ESPR. </pre>	08-02-2024 Newly Added
{76} Sce_Item_get	03	<pre> 76 ??? ?++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x76 UCHAR ItemId; // Item ID UCHAR Flag; // Flag UCHAR zAlign; // Alignment byte } Sce_Item_get; This bytecode gets an item. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{77} Sce_line_start	04	<pre> 77 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x77 UCHAR LineId; // Line ID UCHAR StartX[2]; // Start X position UCHAR StartY[2]; // Start Y position UCHAR StartZ[2]; // Start Z position } Sce_line_start; This bytecode starts a line. </pre>	08-02-2024 Newly Added
{78} Sce_line_main	06	<pre> 78 ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x78 UCHAR LineId; // Line ID UCHAR MainX[2]; // Main X position UCHAR MainY[2]; // Main Y position UCHAR MainZ[2]; // Main Z position } Sce_line_main; This bytecode sets the main line. </pre>	08-02-2024 Newly Added
{79} Sce_line_end	01	<pre> 79++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x79 } Sce_line_end; This bytecode ends a line. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{7A} Sce_parts_bomb	16	<pre> 7A ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x7A UCHAR PartId; // Part ID UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Radius; // Radius UCHAR Power; // Power UCHAR zAlign[8]; // Alignment bytes } Sce_parts_bomb; This bytecode sets a parts bomb. </pre>	08-02-2024 Newly Added
{7B} Sce_parts_down	16	<pre> 7B ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x7B UCHAR PartId; // Part ID UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR zAlign[6]; // Alignment bytes } Sce_parts_down; This bytecode sets a parts down. </pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{7C} Light_color_set	06	<pre>7C ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x7C UCHAR ColorR; // Red component UCHAR ColorG; // Green component UCHAR ColorB; // Blue component UCHAR zAlign[3]; // Alignment bytes } Light_color_set; This bytecode sets the color of a light.</pre>	08-02-2024 Newly Added
{7D} Light_pos_set2	06	<pre>7D ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x7D UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position } Light_pos_set2; This bytecode sets the position of a light (secondary).</pre>	08-02-2024 Newly Added
{7E} Light_kido_set2	06	<pre>7E ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x7E UCHAR Intensity; // Light intensity UCHAR ColorR; // Red component UCHAR ColorG; // Green component UCHAR ColorB; // Blue component UCHAR zAlign; // Alignment byte } Light_kido_set2; This bytecode sets the light intensity and color (secondary).</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{7F} Light_color_set2	06	<pre>7F ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x7F UCHAR ColorR; // Red component UCHAR ColorG; // Green component UCHAR ColorB; // Blue component UCHAR zAlign[3]; // Alignment bytes } Light_color_set2; This bytecode sets the color of a light (secondary).</pre>	08-02-2024 Newly Added
{80} Se_vol	02	<pre>80 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x80 UCHAR Volume; // Volume } Se_vol; This bytecode sets the volume of a sound effect.</pre>	08-02-2024 Newly Added
{81} Sce_Item_cmp	03	<pre>81 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x81 UCHAR ItemId; // Item ID UCHAR Flag; // Flag UCHAR zAlign; // Alignment byte } Sce_Item_cmp; This bytecode compares an item.</pre>	08-02-2024 Newly Added
{82} Sce_espr_task	03	<pre>82 ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x82 UCHAR EspId; // ESPR ID UCHAR TaskId; // Task ID UCHAR zAlign; // Alignment byte } Sce_espr_task; This bytecode assigns a task to an ESPR.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{83} Plc_heal	01	83++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x83 } Plc_heal; This bytecode heals a PLC.	08-02-2024 Newly Added
{84} St_map_hint	02	84 ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x84 UCHAR HintId; // Hint ID } St_map_hint; This bytecode sets a map hint.	08-02-2024 Newly Added
{85} Sce_em_pos_ck	06	85 ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x85 UCHAR EmType; // Enemy type UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position } Sce_em_pos_ck; This bytecode checks the position of an enemy.	08-02-2024 Newly Added
{86} Poison_ck	01	86++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x86 } Poison_ck; This bytecode checks for poison status.	08-02-2024 Newly Added
{87} Poison_clr	01	87++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x87 } Poison_clr; This bytecode clears the poison status.	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{88} Sce_Item_lost2	03	<p>88 ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x88 UCHAR ItemId; // Item ID UCHAR Flag; // Flag UCHAR zAlign; // Alignment byte } Sce_Item_lost2; This bytecode sets an item as lost (secondary).</p>	08-02-2024 Newly Added
{89} Evt_next2	01	<p>89++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x89 } Evt_next2; This bytecode moves to the next event (secondary).</p>	08-02-2024 Newly Added
{8A} Vib_set0	06	<p>8A ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x8A UCHAR VibrationId; // Vibration ID UCHAR Intensity; // Intensity UCHAR Duration; // Duration UCHAR Frequency; // Frequency UCHAR zAlign[2]; // Alignment bytes } Vib_set0; This bytecode sets vibration parameters.</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{8B} Vib_set1	06	<pre> 8B ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x8B UCHAR VibrationId; // Vibration ID UCHAR Intensity; // Intensity UCHAR Duration; // Duration UCHAR Frequency; // Frequency UCHAR zAlign[2]; // Alignment bytes } Vib_set1; This bytecode sets secondary vibration parameters.</pre>	08-02-2024 Newly Added
{8C} Vib_fade_set	08	<pre> 8C ?? ?? ?? ?? ?? ?? ?? ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x8C UCHAR FadeId; // Fade ID UCHAR StartIntensity; // Start intensity UCHAR EndIntensity; // End intensity UCHAR Duration; // Duration UCHAR Frequency; // Frequency UCHAR zAlign[2]; // Alignment bytes } Vib_fade_set; This bytecode sets the fade effect for vibration.</pre>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{8D} Item_aot_set2	24	<pre> 8D ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x8D UCHAR ItemId; // Item ID UCHAR AotType; // AOT Type UCHAR Id; // ID UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR Param[12]; // Parameters } Item_aot_set2; </pre> <p>This bytecode sets an item AOT (secondary).</p>	08-02-2024 Newly Added

Instruction Name	Length	Example / Info	History
{8E} Sce_em_set2	24	<pre> 8E ??++ typedef struct { // Ptr // Description UCHAR Opcode; // 0x8E UCHAR EmType; // Enemy type UCHAR Id; // ID UCHAR SCE; // SCE UCHAR X[2]; // X position UCHAR Y[2]; // Y position UCHAR Z[2]; // Z position UCHAR Width; // Width UCHAR Height; // Height UCHAR Depth; // Depth UCHAR Life; // Life UCHAR Speed; // Speed UCHAR zAlign[9]; // Alignment bytes } Sce_em_set2; This bytecode sets a secondary enemy. </pre>	08-02-2024 Newly Added

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

Permanent link: https://classicmodification.com/doku.php?id=re2_opcodes&rev=1722655695

Last update: **2024/08/02 20:28**

