{00} Nop	01	00++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x00
		} Nop;
		This bytecode is used for alignment of 1-byte opcodes and ending ElseIf blocks.
			08-02-2024 Newly Added

{01} Evt_end	02	01 00++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x01
		UCHAR zAlign; // Always Zero (Alignment byte)
		} 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 in the sequence.
			08-02-2024 Newly Added

{03} Evt_chain	02	03 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 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.
			08-02-2024 Newly Added

{04} Evt_exec	04	04 ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x04
		UCHAR EventId; // Event ID to execute
		UCHAR Parameter1; // Parameter 1 for the event
		UCHAR Parameter2; // Parameter 2 for the event
		} Evt_exec;
		This bytecode executes the specified event with given parameters.
			08-02-2024 Newly Added

{05} Evt_kill	02	05 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x05
		UCHAR EventId; // Event ID to terminate
		} Evt_kill;
		This bytecode terminates the specified event.
			08-02-2024 Newly Added

{06} Ifel_ck	04	06 00 ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x06
		UCHAR zAlign; // Always Zero (Alignment byte)
		USHORT Condition; // Condition to check
		} Ifel_ck;
		This bytecode checks a condition and branches accordingly.
			08-02-2024 Newly Added

{07} Else_ck	04	07 00 ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x07
		UCHAR zAlign; // Always Zero (Alignment byte)
		USHORT Offset; // Offset to jump if condition is met
		} Else_ck;
		This bytecode specifies the offset to jump to if the corresponding Ifel_ck condition is met.
			08-02-2024 Newly Added

{08} Endif	02	08 00++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x08
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Endif;
		This bytecode marks the end of an If/ElseIf/Else block.
			08-02-2024 Newly Added

{09} Sleep	04	09 ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x09
		UCHAR DurationLow; // Low byte of sleep duration
		UCHAR DurationHigh; // High byte of sleep duration
		USHORT zAlign; // Always Zero (Alignment bytes)
		} Sleep;
		This bytecode pauses script execution for the specified duration.
			08-02-2024 Newly Added

{0A} Sleeping	03	0A ?? 00++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x0A
		USHORT Duration; // Duration of sleep
		} Sleeping;
		This bytecode pauses script execution for the specified duration.
			08-02-2024 Newly Added

{0B} Wsleep	01	0B++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x0B
		} Wsleep;
		This bytecode causes the script to wait indefinitely.
			08-02-2024 Newly Added

{0C} Wsleeping	01	0C++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x0C
		} Wsleeping;
		This bytecode causes the script to wait indefinitely.
			08-02-2024 Newly Added

{0D} For	06	0D 00 ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x0D
		UCHAR zAlign; // Always Zero (Alignment byte)
		SHORT StartValue; // Start value of the loop counter
		USHORT EndValue; // End value of the loop counter
		} For;
		This bytecode begins a for-loop with the specified start and end values.
			08-02-2024 Newly Added

{0E} Next	02	0E 00++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x0E
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Next;
		This bytecode marks the end of a for-loop.
			08-02-2024 Newly Added

{0F} While	04	0F 00 ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x0F
		UCHAR zAlign; // Always Zero (Alignment byte)
		SHORT Condition; // Condition to check
		} While;
		This bytecode begins a while-loop that continues as long as the specified condition is true.
			08-02-2024 Newly Added

{10} Ewhile	02	10 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x10
		UCHAR LoopId; // ID of the while-loop to end
		} Ewhile;
		This bytecode ends the specified while-loop.
			08-02-2024 Newly Added

{11} Do	04	11 00 ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x11
		UCHAR zAlign; // Always Zero (Alignment byte)
		SHORT Condition; // Condition to check
		} Do;
		This bytecode begins a do-while loop that executes the loop body once before checking the condition.
			08-02-2024 Newly Added

{12} Edwhile	02	12 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x12
		UCHAR LoopId; // ID of the do-while loop to end
		} Edwhile;
		This bytecode ends the specified do-while loop.
			08-02-2024 Newly Added

{13} Switch	04	13 ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x13
		UCHAR SwitchId; // ID of the switch variable
		USHORT DefaultOffset; // Default offset to jump to if no case matches
		} Switch;
		This bytecode begins a switch-case block with the specified switch variable and default offset.
			08-02-2024 Newly Added

{14} Case	06	14 ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x14
		UCHAR CaseValue; // Value to compare with the switch variable
		USHORT Offset; // Offset to jump to if the case matches
		} Case;
		This bytecode defines a case within a switch-case block.
			08-02-2024 Newly Added

{15} Default	02	15 00++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x15
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Default;
		This bytecode marks the default case in a switch-case block.
			08-02-2024 Newly Added

{16} Eswitch	02	16 00++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x16
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Eswitch;
		This bytecode ends the switch-case block.
			08-02-2024 Newly Added

{17} Goto	06	17 ?? ?? ?? 00 00++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x17
		UCHAR OffsetLow; // Low byte of the offset to jump to
		UCHAR OffsetHigh; // High byte of the offset to jump to
		USHORT zAlign; // Always Zero (Alignment bytes)
		} Goto;
		This bytecode jumps to the specified offset within the script.
			08-02-2024 Newly Added

{18} Gosub	02	18 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x18
		UCHAR SubroutineId; // ID of the subroutine to call
		} Gosub;
		This bytecode calls the specified subroutine.
			08-02-2024 Newly Added

{19} Return	02	19 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x19
		UCHAR SubroutineId; // ID of the subroutine to return from
		} Return;
		This bytecode returns from the specified subroutine.
			08-02-2024 Newly Added

{1A} Break	02	1A ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x1A
		UCHAR LoopId; // ID of the loop to break from
		} Break;
		This bytecode breaks out of the specified loop.
			08-02-2024 Newly Added

{1B} For2	06	1B 00 ?? ?? 00 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 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;
		This bytecode begins a for-loop with the specified start and end values.
			08-02-2024 Newly Added

{1C} Break_point	01	1C++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x1C
		} Break_point;
		This bytecode sets a breakpoint for debugging purposes.
			08-02-2024 Newly Added

{1D} Work_copy	04	1D ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x1D
		UCHAR Source; // Source index
		UCHAR Destination; // Destination index
		UCHAR Typecast; // Typecast operation
		} Work_copy;
		This bytecode copies a value from the source index to the destination index with an optional typecast.
			08-02-2024 Newly Added

{1E} Nop1E	01	1E++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x1E
		} Nop;
		This bytecode is used for alignment of 1-byte opcodes and ending ElseIf blocks.
			08-02-2024 Newly Added

{1F} Nop1F	01	1F++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x1F
		} Nop;
		This bytecode is used for alignment of 1-byte opcodes and ending ElseIf blocks.
			08-02-2024 Newly Added

{20} Nop	01	20++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x20
		} Nop;
		This bytecode is used for alignment of 1-byte opcodes and ending ElseIf blocks.
			08-02-2024 Newly Added

{21} Ck	04	21 ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 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.
			08-02-2024 Newly Added

{22} Set	04	22 ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 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.
			08-02-2024 Newly Added

{23} Cmp	06	23 ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 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.
			08-02-2024 Newly Added

{24} Save	04	24 ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x24
		UCHAR Destination; // Destination index
		SHORT Source; // Source value
		} Save;
		This bytecode saves the specified source value to the destination index.
			08-02-2024 Newly Added

{25} Copy	03	25 ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x25
		UCHAR Source; // Source index
		UCHAR Destination; // Destination index
		} Copy;
		This bytecode copies the value from the source index to the destination index.
			08-02-2024 Newly Added

{26} Calc	06	26 ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x26
		UCHAR Operation; // Arithmetic operation to perform
		UCHAR Operand1; // First operand index
		UCHAR Operand2; // Second operand index
		UCHAR Result; // Result index
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Calc;
		This bytecode performs the specified arithmetic operation on the operands and stores the result.
			08-02-2024 Newly Added

{27} Calc2	04	27 ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x27
		UCHAR Operation; // Arithmetic operation to perform
		UCHAR Operand; // Operand index
		UCHAR Result; // Result index
		} Calc2;
		This bytecode performs the specified arithmetic operation on the operand and stores the result.
			08-02-2024 Newly Added

{28} Sce_rnd	01	28++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x28
		} Sce_rnd;
		This bytecode generates a random value.
			08-02-2024 Newly Added

{29} Cut_chg	02	29 ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x29
		  UCHAR CutsceneId; // ID of the cutscene to change to
		} Cut_chg;
		This bytecode changes the current cutscene to the specified cutscene ID.
			08-02-2024 Newly Added

{2A} Cut_old	01	2A++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x2A
		} Cut_old;
		This bytecode reverts to the previous cutscene.
			08-02-2024 Newly Added

{2B} Message_on	06	2B ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x2B
		  UCHAR MessageId; // ID of the message to display
		  UCHAR DisplayTime; // Time to display the message
		  UCHAR PositionX; // X position of the message
		  UCHAR PositionY; // Y position of the message
		  UCHAR zAlign; // Always Zero (Alignment byte)
		} Message_on;
		This bytecode displays the specified message at the given position for the specified duration.
			08-02-2024 Newly Added

{2C} Aot_set	20	2C ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x2C
		  UCHAR AotId; // ID of the AOT (Animation Object)
		  UCHAR AotType; // Type of the AOT
		  UCHAR Data1[17]; // Data specific to the AOT
		  UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Aot_set;
		This bytecode sets the properties of the specified AOT.
			08-02-2024 Newly Added

{2D} Obj_model_set	38	2D ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x2D
		  UCHAR ModelId; // ID of the object model
		  UCHAR Data1[35]; // Data specific to the object model
		  UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Obj_model_set;
		This bytecode sets the properties of the specified object model.
			08-02-2024 Newly Added

{2E} Work_set	03	2E ?? ??++
		typedef struct { // Ptr //
		Description
		UCHAR Opcode; // 0x2E
		UCHAR WorkId; // ID of the work (task)
		UCHAR Data1; // Data specific to the work
		} Work_set;
		This bytecode sets the properties of the specified work (task).
			08-02-2024 Newly Added

{2F} Speed_set	04	2F ?? ?? ??++
		typedef struct { // Ptr //
		Description
		UCHAR Opcode; // 0x2F
		UCHAR SpeedId; // ID of the speed setting
		UCHAR SpeedValue; // Value of the speed setting
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Speed_set;
		This bytecode sets the specified speed setting.
			08-02-2024 Newly Added

{30} Add_speed	01	30++
		typedef struct { // Ptr //
		Description
		UCHAR Opcode; // 0x30
		} Add_speed;
		This bytecode increments the speed setting.
			08-02-2024 Newly Added

{31} Add_aspeed	01	31++
		typedef struct { // Ptr //
		Description
		UCHAR Opcode; // 0x31
		} Add_aspeed;
		This bytecode increments the angular speed setting.
			08-02-2024 Newly Added

{32} Pos_set	08	32 ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr //
		Description
		UCHAR Opcode; // 0x32
		UCHAR PosX[2]; // X position (2 bytes)
		UCHAR PosY[2]; // Y position (2 bytes)
		UCHAR PosZ[2]; // Z position (2 bytes)
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Pos_set;
		This bytecode sets the position in 3D space.
			08-02-2024 Newly Added

{33} Dir_set	08	33 ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x33
		UCHAR DirX[2]; // X direction (2 bytes)
		UCHAR DirY[2]; // Y direction (2 bytes)
		UCHAR DirZ[2]; // Z direction (2 bytes)
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Dir_set;
		This bytecode sets the direction in 3D space.
			08-02-2024 Newly Added

{34} Member_set	04	34 ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x34
		UCHAR MemberId; // ID of the member
		UCHAR Property1; // Property 1 of the member
		UCHAR Property2; // Property 2 of the member
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Member_set;
		This bytecode sets the properties of the specified member.
			08-02-2024 Newly Added

{35} Member_set2	03	35 ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x35
		UCHAR MemberId; // ID of the member
		UCHAR Property; // Property of the member
		} Member_set2;
		This bytecode sets a single property of the specified member.
			08-02-2024 Newly Added

{36} Se_on	12	36 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x36
		UCHAR SeId; // ID of the sound effect
		UCHAR Volume; // Volume of the sound effect
		UCHAR Pitch; // Pitch of the sound effect
		UCHAR Pan; // Pan of the sound effect
		UCHAR Delay[8]; // Delay before playing the sound effect (8 bytes)
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Se_on;
		This bytecode plays the specified sound effect with the given properties.
			08-02-2024 Newly Added

{37} Sca_id_set	04	37 ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x37
		UCHAR ScaId; // ID of the scale
		UCHAR ScaleX; // X scale value
		UCHAR ScaleY; // Y scale value
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Sca_id_set;
		This bytecode sets the scale of the specified object.
			08-02-2024 Newly Added

{38} Flr_set	03	38 ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x38
		UCHAR FlrId; // ID of the floor
		UCHAR Height; // Height of the floor
		} Flr_set;
		This bytecode sets the height of the specified floor.
			08-02-2024 Newly Added

{39} Dir_ck	08	39 ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x39
		UCHAR DirX[2]; // X direction to check (2 bytes)
		UCHAR DirY[2]; // Y direction to check (2 bytes)
		UCHAR DirZ[2]; // Z direction to check (2 bytes)
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Dir_ck;
		This bytecode checks the direction in 3D space.
			08-02-2024 Newly Added

{3A} Sce_espr_on	16	3A ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x3A
		UCHAR EsprId; // ID of the ESPR (effect sprite)
		UCHAR PosX[2]; // X position of the effect sprite (2 bytes)
		UCHAR PosY[2]; // Y position of the effect sprite (2 bytes)
		UCHAR PosZ[2]; // Z position of the effect sprite (2 bytes)
		UCHAR ScaleX; // X scale of the effect sprite
		UCHAR ScaleY; // Y scale of the effect sprite
		UCHAR Rotation; // Rotation of the effect sprite
		UCHAR Alpha; // Alpha transparency of the effect sprite
		UCHAR Duration[4]; // Duration of the effect sprite (4 bytes)
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Sce_espr_on;
		This bytecode activates the specified effect sprite with the given properties.
			08-02-2024 Newly Added

{3B} Door_aot_set	32	3B ??++
		`typedef struct {          // Ptr // Description`
		`    UCHAR Opcode;                  // 0x3B`
		`    UCHAR DoorId;                  // ID of the door`
		`    UCHAR PosX[2];                 // X position of the door (2 bytes)`
		`    UCHAR PosY[2];                 // Y position of the door (2 bytes)`
		`    UCHAR PosZ[2];                 // Z position of the door (2 bytes)`
		`    UCHAR Rotation;                // Rotation of the door`
		`    UCHAR LockStatus;              // Lock status of the door`
		`    UCHAR KeyItemId;               // ID of the key item required to unlock the door`
		`    UCHAR zAlign[23];              // Always Zero (Alignment bytes)`
		`} Door_aot_set;`
		This bytecode sets the properties of the specified door, including position, rotation, and lock status.
			08-02-2024 Newly Added

{3C} Cut_auto	02	3C ??++
		`typedef struct {          // Ptr // Description`
		`    UCHAR Opcode;                  // 0x3C`
		`    UCHAR CutsceneId;              // ID of the cutscene to automatically start`
		`} Cut_auto;`
		This bytecode starts the specified cutscene automatically.
			08-02-2024 Newly Added

{3D} Member_copy	03	3D ?? ??++
		`typedef struct {          // Ptr // Description`
		`    UCHAR Opcode;                  // 0x3D`
		`    UCHAR SourceMemberId;          // ID of the source member`
		`    UCHAR DestinationMemberId;     // ID of the destination member`
		`} Member_copy;`
		This bytecode copies the properties from the source member to the destination member.
			08-02-2024 Newly Added

{3E} Member_cmp	06	3E ?? ?? ?? ?? ??++
		`typedef struct {          // Ptr // Description`
		`    UCHAR Opcode;                  // 0x3E`
		`    UCHAR MemberId;                // ID of the member`
		`    UCHAR Property;                // Property to compare`
		`    USHORT Value;                  // Value to compare against`
		`    UCHAR ComparisonType;          // Type of comparison (e.g., equal, not equal)`
		`} Member_cmp;`
		This bytecode compares the specified property of the member with the given value using the specified comparison type.
			08-02-2024 Newly Added

{3F} Plc_motion	04	3F ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x3F
		UCHAR MotionId; // ID of the motion to play
		UCHAR Speed; // Speed of the motion
		UCHAR Loop; // Loop flag (0 = no loop, 1 = loop)
		} Plc_motion;
		This bytecode sets the specified motion to play at the given speed with the optional loop flag.
			08-02-2024 Newly Added

{40} Plc_dest	08	40 ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x40
		UCHAR DestX[2]; // X destination (2 bytes)
		UCHAR DestY[2]; // Y destination (2 bytes)
		UCHAR DestZ[2]; // Z destination (2 bytes)
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Plc_dest;
		This bytecode sets the destination position in 3D space.
			08-02-2024 Newly Added

{41} Plc_neck	10	41 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x41
		UCHAR NeckId; // ID of the neck motion
		UCHAR PosX[2]; // X position of the neck motion (2 bytes)
		UCHAR PosY[2]; // Y position of the neck motion (2 bytes)
		UCHAR PosZ[2]; // Z position of the neck motion (2 bytes)
		UCHAR RotationX; // X rotation of the neck motion
		UCHAR RotationY; // Y rotation of the neck motion
		UCHAR RotationZ; // Z rotation of the neck motion
		UCHAR zAlign[4]; // Always Zero (Alignment bytes)
		} Plc_neck;
		This bytecode sets the specified neck motion with the given position and rotation properties.
			08-02-2024 Newly Added

{42} Plc_ret	01	42++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x42
		} Plc_ret;
		This bytecode returns control from the current motion or behavior.
			08-02-2024 Newly Added

{43} Plc_flg	04	43 ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x43
		UCHAR Flag; // Flag to set
		UCHAR Value; // Value to set the flag to
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Plc_flg;
		This bytecode sets the specified flag to the given value.
			08-02-2024 Newly Added

{44} Sce_em_set	22	44 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x44
		UCHAR EmId; // ID of the enemy or entity
		UCHAR PosX[2]; // X position (2 bytes)
		UCHAR PosY[2]; // Y position (2 bytes)
		UCHAR PosZ[2]; // Z position (2 bytes)
		UCHAR RotationX; // X rotation
		UCHAR RotationY; // Y rotation
		UCHAR RotationZ; // Z rotation
		UCHAR Speed; // Movement speed
		UCHAR Health; // Health value
		UCHAR zAlign[8]; // Always Zero (Alignment bytes)
		} Sce_em_set;
		This bytecode sets the specified enemy or entity with the given position, rotation, speed, and health properties.
			08-02-2024 Newly Added

{45} Col_chg_set	05	45 ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 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.
			08-02-2024 Newly Added

{46} Aot_reset	10	46 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x46
		UCHAR AotId; // ID of the AOT to reset
		UCHAR zAlign[11]; // Always Zero (Alignment bytes)
		} Aot_reset;
		This bytecode resets the specified AOT to its default state.
			08-02-2024 Newly Added

{47} Aot_on	02	47 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x47
		UCHAR AotId; // ID of the AOT to activate
		} Aot_on;
		This bytecode activates the specified AOT.
			08-02-2024 Newly Added

{48} Super_set	16	48 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x48
		UCHAR SuperId; // ID of the super effect
		UCHAR PosX[2]; // X position (2 bytes)
		UCHAR PosY[2]; // Y position (2 bytes)
		UCHAR PosZ[2]; // Z position (2 bytes)
		UCHAR ScaleX; // X scale
		UCHAR ScaleY; // Y scale
		UCHAR Rotation; // Rotation value
		UCHAR Alpha; // Alpha transparency value
		UCHAR Duration[4]; // 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.
			08-02-2024 Newly Added

{49} Super_reset	08	49 ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 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.
			08-02-2024 Newly Added

{4A} Plc_gun	02	4A ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x4A
		UCHAR GunId; // ID of the gun to equip
		} Plc_gun;
		This bytecode equips the specified gun.
			08-02-2024 Newly Added

{4B} Cut_replace	03	4B ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 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.
			08-02-2024 Newly Added

{4C} Sce_espr_kill	05	4C ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x4C
		UCHAR EsprId; // ID of the effect sprite to kill
		UCHAR PosX[2]; // X position of the effect sprite (2 bytes)
		UCHAR PosY[2]; // Y position of the effect sprite (2 bytes)
		UCHAR PosZ[2]; // Z position of the effect sprite (2 bytes)
		} Sce_espr_kill;
		This bytecode kills the specified effect sprite at the given position.
			08-02-2024 Newly Added

{4D} Door_model_set	22	4D ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x4D
		UCHAR DoorId; // ID of the door model
		UCHAR PosX[2]; // X position of the door model (2 bytes)
		UCHAR PosY[2]; // Y position of the door model (2 bytes)
		UCHAR PosZ[2]; // 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 ScaleX; // X scale of the door model
		UCHAR ScaleY; // Y scale of the door model
		UCHAR zAlign[12]; // Always Zero (Alignment bytes)
		} Door_model_set;
		This bytecode sets the properties of the specified door model with position, rotation, and scale values.
			08-02-2024 Newly Added

{4E} Item_aot_set	22	4E ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x4E
		UCHAR ItemId; // ID of the item
		UCHAR PosX[2]; // X position of the item (2 bytes)
		UCHAR PosY[2]; // Y position of the item (2 bytes)
		UCHAR PosZ[2]; // 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[12]; // Always Zero (Alignment bytes)
		} Item_aot_set;
		This bytecode sets the properties of the specified item with position, rotation, and scale values.
			08-02-2024 Newly Added

{4F} Sce_key_ck	04	4F ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x4F
		UCHAR KeyId; // ID of the key to check
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		USHORT Result; // Result of the key check
		} Sce_key_ck;
		This bytecode checks if the specified key is present and returns the result.
			08-02-2024 Newly Added

{50} Sce_trg_ck	04	50 ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x50
		UCHAR TriggerId; // ID of the trigger to check
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		USHORT Result; // Result of the trigger check
		} Sce_trg_ck;
		This bytecode checks if the specified trigger is activated and returns the result.
			08-02-2024 Newly Added

{51} Sce_bgm_control	06	51 ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x51
		  UCHAR BgmId; // ID of the background music track
		  UCHAR Volume; // Volume level
		  UCHAR Loop; // Loop flag (0 = no loop, 1 = loop)
		  UCHAR FadeIn; // Fade-in duration
		  UCHAR FadeOut; // Fade-out duration
		} Sce_bgm_control;
		This bytecode controls the playback of the specified background music track with volume, loop, fade-in, and fade-out settings.
			08-02-2024 Newly Added

{52} Sce_espr_control	06	52 ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x52
		  UCHAR EsprId; // ID of the effect sprite
		  UCHAR Action; // Action to perform (e.g., start, stop)
		  UCHAR PosX[2]; // X position of the effect sprite (2 bytes)
		  UCHAR PosY[2]; // Y position of the effect sprite (2 bytes)
		  UCHAR PosZ[2]; // Z position of the effect sprite (2 bytes)
		} Sce_espr_control;
		This bytecode controls the specified effect sprite with the given action and position settings.
			08-02-2024 Newly Added

{53} Sce_fade_set	06	53 ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x53
		  UCHAR FadeId; // ID of the fade effect
		  UCHAR StartIntensity; // Start intensity of the fade effect
		  UCHAR EndIntensity; // End intensity of the fade effect
		  UCHAR Duration; // Duration of the fade effect
		  UCHAR Color; // Color of the fade effect
		} Sce_fade_set;
		This bytecode sets the properties of the specified fade effect with start intensity, end intensity, duration, and color values.
			08-02-2024 Newly Added

{54} Sce_espr3d_on	22	54 ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x54
		  UCHAR Espr3dId; // ID of the 3D effect sprite
		  UCHAR PosX[2]; // X position of the 3D effect sprite (2 bytes)
		  UCHAR PosY[2]; // Y position of the 3D effect sprite (2 bytes)
		  UCHAR PosZ[2]; // Z position of the 3D effect sprite (2 bytes)
		  UCHAR RotationX; // X rotation of the 3D effect sprite
		  UCHAR RotationY; // Y rotation of the 3D effect sprite
		  UCHAR RotationZ; // Z rotation of the 3D effect sprite
		  UCHAR ScaleX; // X scale of the 3D effect sprite
		  UCHAR ScaleY; // Y scale of the 3D effect sprite
		  UCHAR zAlign[12]; // Always Zero (Alignment bytes)
		} Sce_espr3d_on;
		This bytecode activates the specified 3D effect sprite with the given position, rotation, and scale values.
			08-02-2024 Newly Added

{55} Member_calc	06	55 ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x55
		  UCHAR Operation; // Arithmetic operation to perform
		  UCHAR Operand1; // First operand index
		  UCHAR Operand2; // Second operand index
		  UCHAR Result; // Result index
		  UCHAR zAlign; // Always Zero (Alignment byte)
		} Member_calc;
		This bytecode performs the specified arithmetic operation on the operands and stores the result in a member.
			08-02-2024 Newly Added

{56} Member_calc2	04	56 ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x56
		  UCHAR Operation; // Arithmetic operation to perform
		  UCHAR Operand; // Operand index
		  UCHAR Result; // Result index
		  UCHAR zAlign; // Always Zero (Alignment byte)
		} Member_calc2;
		This bytecode performs the specified arithmetic operation on the operand and stores the result in a member.
			08-02-2024 Newly Added

{57} Sce_bgmtbl_set	08	57 ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x57
		UCHAR BgmTblId; // ID of the background music table
		UCHAR TrackId[2]; // ID of the music track (2 bytes)
		UCHAR Volume; // Volume level
		UCHAR Loop; // Loop flag (0 = no loop, 1 = loop)
		UCHAR FadeIn; // Fade-in duration
		UCHAR FadeOut; // Fade-out duration
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Sce_bgmtbl_set;
		This bytecode sets the properties of the specified background music table with track ID, volume, loop, fade-in, and fade-out settings.
			08-02-2024 Newly Added

{58} Plc_rot	04	58 ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x58
		UCHAR RotationX; // X rotation value
		UCHAR RotationY; // Y rotation value
		UCHAR RotationZ; // Z rotation value
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Plc_rot;
		This bytecode sets the rotation values in 3D space.
			08-02-2024 Newly Added

{59} Xa_on	04	59 ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x59
		UCHAR XaId; // ID of the XA audio stream
		UCHAR Volume; // Volume level
		UCHAR Loop; // Loop flag (0 = no loop, 1 = loop)
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Xa_on;
		This bytecode plays the specified XA audio stream with volume and loop settings.
			08-02-2024 Newly Added

{5A} Weapon_chg	02	5A ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x5A
		UCHAR WeaponId; // ID of the weapon to change to
		} Weapon_chg;
		This bytecode changes the player's weapon to the specified weapon ID.
			08-02-2024 Newly Added

{5B} Plc_cnt	02	5B ??++
		typedef struct { // Ptr //
		Description
		UCHAR Opcode; // 0x5B
		UCHAR CounterId; // ID of the counter to increment
		} Plc_cnt;
		This bytecode increments the specified counter.
			08-02-2024 Newly Added

{5C} Sce_shake_on	03	5C ?? ?? ??++
		typedef struct { // Ptr //
		Description
		UCHAR Opcode; // 0x5C
		UCHAR Intensity; // Intensity of the shake effect
		UCHAR Duration; // Duration of the shake effect
		UCHAR Frequency; // Frequency of the shake effect
		} Sce_shake_on;
		This bytecode activates the screen shake effect with the specified intensity, duration, and frequency.
			08-02-2024 Newly Added

{5D} Mizu_div_set	02	5D ??++
		typedef struct { // Ptr //
		Description
		UCHAR Opcode; // 0x5D
		UCHAR Division; // Division factor for water effects
		} Mizu_div_set;
		This bytecode sets the division factor for water effects.
			08-02-2024 Newly Added

{5E} Keep_Item_ck	02	5E ??++
		typedef struct { // Ptr //
		Description
		UCHAR Opcode; // 0x5E
		UCHAR ItemId; // ID of the item to check
		} Keep_Item_ck;
		This bytecode checks if the player has the specified item and returns the result.
			08-02-2024 Newly Added

{5F} Xa_vol	02	5F ??++
		typedef struct { // Ptr //
		Description
		UCHAR Opcode; // 0x5F
		UCHAR Volume; // Volume level for XA audio stream
		} Xa_vol;
		This bytecode sets the volume level for the XA audio stream.
			08-02-2024 Newly Added

{60} Kage_set	14	60 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x60
		UCHAR KageId; // ID of the shadow effect
		UCHAR PosX[2]; // X position of the shadow effect (2 bytes)
		UCHAR PosY[2]; // Y position of the shadow effect (2 bytes)
		UCHAR PosZ[2]; // Z position of the shadow effect (2 bytes)
		UCHAR ScaleX; // X scale of the shadow effect
		UCHAR ScaleY; // Y scale of the shadow effect
		UCHAR Rotation; // Rotation of the shadow effect
		UCHAR Alpha; // Alpha transparency of the shadow effect
		UCHAR Duration[4]; // Duration of the shadow effect (4 bytes)
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Kage_set;
		This bytecode sets the properties of the specified shadow effect with position, scale, rotation, alpha, and duration values.
			08-02-2024 Newly Added

{61} Cut_be_set	04	61 ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x61
		UCHAR CutsceneId; // ID of the cutscene to set
		UCHAR EventId; // ID of the event associated with the cutscene
		UCHAR TriggerId; // ID of the trigger associated with the cutscene
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Cut_be_set;
		This bytecode sets the properties of the specified cutscene with event and trigger associations.
			08-02-2024 Newly Added

{62} Sce_Item_lost	02	62 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x62
		UCHAR ItemId; // ID of the item to remove
		} Sce_Item_lost;
		This bytecode removes the specified item from the player's inventory.
			08-02-2024 Newly Added

{63} Plc_gun_eff	01	63++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x63
		} Plc_gun_eff;
		This bytecode triggers the gun effect for the current weapon.
			08-02-2024 Newly Added

{64} Sce_espr_on2	16	64 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x64
		UCHAR EsprId; // ID of the effect sprite
		UCHAR PosX[2]; // X position of the effect sprite (2 bytes)
		UCHAR PosY[2]; // Y position of the effect sprite (2 bytes)
		UCHAR PosZ[2]; // Z position of the effect sprite (2 bytes)
		UCHAR ScaleX; // X scale of the effect sprite
		UCHAR ScaleY; // Y scale of the effect sprite
		UCHAR Rotation; // Rotation of the effect sprite
		UCHAR Alpha; // Alpha transparency of the effect sprite
		UCHAR Duration[4]; // Duration of the effect sprite (4 bytes)
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Sce_espr_on2;
		This bytecode activates the specified effect sprite with position, scale, rotation, alpha, and duration values.
			08-02-2024 Newly Added

{65} Sce_espr_kill2	02	65 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x65
		UCHAR EsprId; // ID of the effect sprite to kill
		} Sce_espr_kill2;
		This bytecode kills the specified effect sprite.
			08-02-2024 Newly Added

{66} Plc_stop	01	66++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x66
		} Plc_stop;
		This bytecode stops the current action or motion.
			08-02-2024 Newly Added

{67} Aot_set_4p	28	67 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x67
		UCHAR AotId; // ID of the AOT (Animation Object)
		UCHAR PosX[2]; // X position of the AOT (2 bytes)
		UCHAR PosY[2]; // Y position of the AOT (2 bytes)
		UCHAR PosZ[2]; // 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[16]; // Always Zero (Alignment bytes)
		} Aot_set_4p;
		This bytecode sets the properties of the specified AOT (Animation Object) with position, rotation, and scale values.
			08-02-2024 Newly Added

{68} Door_aot_set_4p	40	68 ??++
		`typedef struct {          // Ptr // Description`
		`    UCHAR Opcode;                // 0x68`
		`    UCHAR DoorId;                // ID of the door`
		`    UCHAR PosX[2];               // X position of the door (2 bytes)`
		`    UCHAR PosY[2];               // Y position of the door (2 bytes)`
		`    UCHAR PosZ[2];               // 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;`
		This bytecode sets the properties of the specified door with position, rotation, and scale values.
			08-02-2024 Newly Added

{69} Item_aot_set_4p	30	69 ??++
		`typedef struct {          // Ptr // Description`
		`    UCHAR Opcode;                // 0x69`
		`    UCHAR ItemId;                // ID of the item`
		`    UCHAR PosX[2];               // X position of the item (2 bytes)`
		`    UCHAR PosY[2];               // Y position of the item (2 bytes)`
		`    UCHAR PosZ[2];               // 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[18];            // Always Zero (Alignment bytes)`
		`} Item_aot_set_4p;`
		This bytecode sets the properties of the specified item with position, rotation, and scale values.
			08-02-2024 Newly Added

{6A} Light_pos_set	06	6A ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x6A
		UCHAR PosX[2]; // X position of the light (2 bytes)
		UCHAR PosY[2]; // Y position of the light (2 bytes)
		UCHAR PosZ[2]; // Z position of the light (2 bytes)
		} Light_pos_set;
		This bytecode sets the position of the specified light in 3D space.

{6B} Light_kido_set	04	6B ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x6B
		UCHAR Intensity; // Intensity of the light
		UCHAR Range; // Range of the light
		UCHAR Color; // Color of the light
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Light_kido_set;
		This bytecode sets the intensity, range, and color of the specified light.

{6C} Rbj_reset	01	6C++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x6C
		} Rbj_reset;
		This bytecode resets the RBJ (Resident Biohazard Jump) system to its default state.

{6D} Sce_scr_move	04	6D ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x6D
		UCHAR PosX[2]; // X position of the screen (2 bytes)
		UCHAR PosY[2]; // Y position of the screen (2 bytes)
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Sce_scr_move;
		This bytecode moves the screen to the specified position.

{6E} Parts_set	06	6E ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x6E
		UCHAR PartId; // ID of the part to set
		UCHAR PosX[2]; // X position of the part (2 bytes)
		UCHAR PosY[2]; // Y position of the part (2 bytes)
		UCHAR PosZ[2]; // Z position of the part (2 bytes)
		} Parts_set;
		This bytecode sets the properties of the specified part with position values.

{6F} Movie_on	02	6F ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x6F
		UCHAR MovieId; // ID of the movie to play
		} Movie_on;
		This bytecode plays the specified movie.
			08-02-2024 Newly Added

{70} Splc_ret	01	70++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x70
		} Splc_ret;
		This bytecode returns from the current script location.
			08-02-2024 Newly Added

{71} Splc_sce	01	71++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x71
		} Splc_sce;
		This bytecode sets the specified scene.
			08-02-2024 Newly Added

{72} Super_on	16	72 ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x72
		UCHAR SuperId; // ID of the super effect
		UCHAR PosX[2]; // X position of the super effect (2 bytes)
		UCHAR PosY[2]; // Y position of the super effect (2 bytes)
		UCHAR PosZ[2]; // Z position of the super effect (2 bytes)
		UCHAR ScaleX; // X scale of the super effect
		UCHAR ScaleY; // Y scale of the super effect
		UCHAR Rotation; // Rotation of the super effect
		UCHAR Alpha; // Alpha transparency of the super effect
		UCHAR Duration[4]; // Duration of the super effect (4 bytes)
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Super_on;
		This bytecode activates the specified super effect with position, scale, rotation, alpha, and duration values.
			08-02-2024 Newly Added

{73} Mirror_set	08	73 ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x73
		UCHAR PosX[2]; // X position of the mirror (2 bytes)
		UCHAR PosY[2]; // Y position of the mirror (2 bytes)
		UCHAR PosZ[2]; // Z position of the mirror (2 bytes)
		UCHAR ScaleX; // X scale of the mirror
		UCHAR ScaleY; // Y scale of the mirror
		UCHAR Rotation; // Rotation of the mirror
		} Mirror_set;
		This bytecode sets the properties of the specified mirror with position, scale, and rotation values.
			08-02-2024 Newly Added

{74} Sce_fade_adjust	04	74 ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x74
		UCHAR FadeId; // ID of the fade effect
		UCHAR Adjustment; // Adjustment value for the fade effect
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Sce_fade_adjust;
		This bytecode adjusts the properties of the specified fade effect.
			08-02-2024 Newly Added

{75} Sce_espr3d_on2	22	75 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x75
		UCHAR Espr3dId; // ID of the 3D effect sprite
		UCHAR PosX[2]; // X position of the 3D effect sprite (2 bytes)
		UCHAR PosY[2]; // Y position of the 3D effect sprite (2 bytes)
		UCHAR PosZ[2]; // Z position of the 3D effect sprite (2 bytes)
		UCHAR RotationX; // X rotation of the 3D effect sprite
		UCHAR RotationY; // Y rotation of the 3D effect sprite
		UCHAR RotationZ; // Z rotation of the 3D effect sprite
		UCHAR ScaleX; // X scale of the 3D effect sprite
		UCHAR ScaleY; // Y scale of the 3D effect sprite
		UCHAR zAlign[12]; // Always Zero (Alignment bytes)
		} Sce_espr3d_on2;
		This bytecode activates the specified 3D effect sprite with position, rotation, and scale values.
			08-02-2024 Newly Added

{76} Sce_Item_get	03	76 ?? ?? ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x76
		  UCHAR ItemId; // ID of the item to get
		  UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Sce_Item_get;
		This bytecode adds the specified item to the player's inventory.
			08-02-2024 Newly Added

{77} Sce_line_start	04	77 ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x77
		  UCHAR LineId; // ID of the line to start
		  UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		  UCHAR Duration; // Duration of the line
		} Sce_line_start;
		This bytecode starts the specified line with the given duration.
			08-02-2024 Newly Added

{78} Sce_line_main	06	78 ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x78
		  UCHAR LineId; // ID of the line
		  UCHAR PosX[2]; // X position of the line (2 bytes)
		  UCHAR PosY[2]; // Y position of the line (2 bytes)
		  UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Sce_line_main;
		This bytecode sets the main properties of the specified line with position values.
			08-02-2024 Newly Added

{79} Sce_line_end	01	79++
		typedef struct { // Ptr // Description
		  UCHAR Opcode; // 0x79
		} Sce_line_end;
		This bytecode ends the specified line.
			08-02-2024 Newly Added

{7A} Sce_parts_bomb	16	7A ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x7A
		UCHAR BombId; // ID of the bomb
		UCHAR PosX[2]; // X position of the bomb (2 bytes)
		UCHAR PosY[2]; // Y position of the bomb (2 bytes)
		UCHAR PosZ[2]; // Z position of the bomb (2 bytes)
		UCHAR ScaleX; // X scale of the bomb
		UCHAR ScaleY; // Y scale of the bomb
		UCHAR Rotation; // Rotation of the bomb
		UCHAR Alpha; // Alpha transparency of the bomb
		UCHAR Duration[4]; // Duration of the bomb (4 bytes)
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Sce_parts_bomb;
		This bytecode sets the properties of the specified bomb with position, scale, rotation, alpha, and duration values.
			08-02-2024 Newly Added

{7B} Sce_parts_down	16	7B ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x7B
		UCHAR PartId; // ID of the part to set down
		UCHAR PosX[2]; // X position of the part (2 bytes)
		UCHAR PosY[2]; // Y position of the part (2 bytes)
		UCHAR PosZ[2]; // Z position of the part (2 bytes)
		UCHAR ScaleX; // X scale of the part
		UCHAR ScaleY; // Y scale of the part
		UCHAR Rotation; // Rotation of the part
		UCHAR Alpha; // Alpha transparency of the part
		UCHAR Duration[4]; // Duration of the part (4 bytes)
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Sce_parts_down;
		This bytecode sets the properties of the specified part with position, scale, rotation, alpha, and duration values.
			08-02-2024 Newly Added

{7C} Light_color_set	06	7C ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x7C
		UCHAR R; // Red color value
		UCHAR G; // Green color value
		UCHAR B; // Blue color value
		UCHAR Intensity; // Intensity of the light
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Light_color_set;
		This bytecode sets the color and intensity of the specified light.
			08-02-2024 Newly Added

{7D} Light_pos_set2	06	7D ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x7D
		UCHAR PosX[2]; // X position of the light (2 bytes)
		UCHAR PosY[2]; // Y position of the light (2 bytes)
		UCHAR PosZ[2]; // Z position of the light (2 bytes)
		} Light_pos_set2;
		This bytecode sets the position of the specified light in 3D space.
			08-02-2024 Newly Added

{7E} Light_kido_set2	06	7E ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x7E
		UCHAR Intensity; // Intensity of the light
		UCHAR Range; // Range of the light
		UCHAR Color; // Color of the light
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Light_kido_set2;
		This bytecode sets the intensity, range, and color of the specified light.
			08-02-2024 Newly Added

{7F} Light_color_set2	06	7F ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x7F
		UCHAR R; // Red color value
		UCHAR G; // Green color value
		UCHAR B; // Blue color value
		UCHAR Intensity; // Intensity of the light
		UCHAR zAlign; // Always Zero (Alignment byte)
		} Light_color_set2;
		This bytecode sets the color and intensity of the specified light.
			08-02-2024 Newly Added

{80} Se_vol	02	80 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x80
		UCHAR Volume; // Volume level for the sound effect
		} Se_vol;
		This bytecode sets the volume level for the specified sound effect.
			08-02-2024 Newly Added

{81} Sce_Item_cmp	03	81 ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x81
		UCHAR ItemId; // ID of the item to compare
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Sce_Item_cmp;
		This bytecode compares the specified item with the player's inventory.
			08-02-2024 Newly Added

{82} Sce_espr_task	03	82 ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x82
		UCHAR TaskId; // ID of the ESPR (effect sprite) task
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Sce_espr_task;
		This bytecode activates the specified ESPR (effect sprite) task.
			08-02-2024 Newly Added

{83} Plc_heal	01	83++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x83
		} Plc_heal;
		This bytecode heals the player.
			08-02-2024 Newly Added

{84} St_map_hint	02	84 ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x84
		UCHAR HintId; // ID of the map hint
		} St_map_hint;
		This bytecode displays the specified map hint.
			08-02-2024 Newly Added

{85} Sce_em_pos_ck	06	85 ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x85
		UCHAR EmId; // ID of the enemy or entity
		UCHAR PosX[2]; // X position to check (2 bytes)
		UCHAR PosY[2]; // Y position to check (2 bytes)
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Sce_em_pos_ck;
		This bytecode checks the position of the specified enemy or entity.
			08-02-2024 Newly Added

{86} Poison_ck	01	86++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x86
		} Poison_ck;
		This bytecode checks if the player is poisoned.
			08-02-2024 Newly Added

{87} Poison_clr	01	87++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x87
		} Poison_clr;
		This bytecode clears the player's poison status.
			08-02-2024 Newly Added

{88} Sce_Item_lost2	03	88 ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x88
		UCHAR ItemId; // ID of the item to remove
		UCHAR zAlign[2]; // Always Zero (Alignment bytes)
		} Sce_Item_lost2;
		This bytecode removes the specified item from the player's inventory.
			08-02-2024 Newly Added

{89} Evt_next2	01	89++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x89
		} Evt_next2;
		This bytecode moves to the next event in the sequence.
			08-02-2024 Newly Added

{8A} Vib_set0	06	8A ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x8A
		UCHAR Intensity; // Intensity of the vibration
		UCHAR Duration; // Duration of the vibration
		UCHAR Frequency; // Frequency of the vibration
		UCHAR zAlign[3]; // Always Zero (Alignment bytes)
		} Vib_set0;
		This bytecode sets the properties of the vibration effect with intensity, duration, and frequency values.
			08-02-2024 Newly Added

{8B} Vib_set1	06	8B ?? ?? ?? ?? ?? ??++
		typedef struct { // Ptr // Description
		UCHAR Opcode; // 0x8B
		UCHAR Intensity; // Intensity of the vibration
		UCHAR Duration; // Duration of the vibration
		UCHAR Frequency; // Frequency of the vibration
		UCHAR zAlign[3]; // Always Zero (Alignment bytes)
		} Vib_set1;
		This bytecode sets the properties of the vibration effect with intensity, duration, and frequency values.
			08-02-2024 Newly Added

{8C} Vib_fade_set	08	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.
			08-02-2024 Newly Added

{8D} Item_aot_set2	24	8D ??++
		typedef struct { // Ptr //
		Description
		UCHAR Opcode; // 0x8D
		UCHAR ItemId; // ID of the item
		UCHAR PosX[2]; // X position of the item (2 bytes)
		UCHAR PosY[2]; // Y position of the item (2 bytes)
		UCHAR PosZ[2]; // 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[12]; // Always Zero (Alignment bytes)
		} Item_aot_set2;
		This bytecode sets the properties of the specified item with position, rotation, and scale values.
			08-02-2024 Newly Added

{8E} Sce_em_set2	24	8E ??++
		`typedef struct {          // Ptr // Description`
		`    UCHAR Opcode;                // 0x8E`
		`    UCHAR EmId;                  // ID of the enemy or entity`
		`    UCHAR PosX[2];               // X position (2 bytes)`
		`    UCHAR PosY[2];               // Y position (2 bytes)`
		`    UCHAR PosZ[2];               // Z position (2 bytes)`
		`    UCHAR RotationX;             // X rotation`
		`    UCHAR RotationY;             // Y rotation`
		`    UCHAR RotationZ;             // Z rotation`
		`    UCHAR ScaleX;                // X scale`
		`    UCHAR ScaleY;                // Y scale`
		`    UCHAR Health;                // Health value`
		`    UCHAR zAlign[12];            // Always Zero (Alignment bytes)`
		`} Sce_em_set2;`
		This bytecode sets the specified enemy or entity with the given position, rotation, scale, and health properties.
			08-02-2024 Newly Added