OpenFlight®
Scene Description
Database Specification

Version 14.2.4, Revision A
January 1996

550 S. Winchester Blvd., Suite 500, San Jose, CA 95128 Phone (408) 261-4100 Fax (408) 261-4101

OpenFlight Scene Description Database Specification, v14.2.4, Revision A (1/96)

Printed in the U.S.A.

P/N MG-100-01-004

Table of Contents

Chapter 1 OpenFlight® Scene Description 1

Document Conventions 1
Concepts Supported in OpenFlight 1
Database Hierarchy 1
Database Files 3
Instancing 3
Replication 4
Bounding Volumes 4
Header Record 4
Group Record 7
Level-of-Detail Record 8
Degree-of-Freedom Record 9
Object Record 11
Polygon Record 12
Vertex Table 14
Control Records 18
Comment and ID Records 19
Key Table Records 19
Linkage Records 21
Color Table 25
Material Table 25
Transformations 28
Geometry 31
Bounding Volumes 32
Binary Separating Plane (BSP) Record 33
Replication and Instancing 34
Texture Pattern File Reference 35
Eyepoint and Trackplane Positions 36
Sound Palette Records 37
Sound Bead Record 38
Sound Files 39
Light Source Palette Records 40
Light Source Bead Records 41
Road Segment Records 41
Path Beads 42
Zone Files 42
Line Style Records 44
Clipping Regions Records 44
Font and Text Records 45
Switch Records 46

Chapter 2 Texture Files 47

Texture Pattern Files 47
Texture Attribute Files 47

Chapter 3 OpenFlight Opcodes 53

1 OpenFlight® Scene Description

This document describes the concepts and file formats of the binary OpenFlight® Scene Description data format created and maintained by MultiGen Inc. Databases in this format can be created and edited using any software program produced by MultiGen Inc.

Document Conventions

Lines and paragraphs that contain a discussion of material new to the current release of the software are marked with a revision bar, such as the one to the left.

Concepts Supported in OpenFlight

The OpenFlight database format is designed to support both simple and relatively sophisticated real-time software applications. The full implementation of OpenFlight supports variable levels of detail, degrees of freedom, sound, instancing (both within a file and to external files), replication, animation sequences, bounding boxes for real-time culling, shadows, advanced scene lighting features, lights and light strings, transparency, texture mapping, material properties, and several other features.

A simple real-time software package that interprets an OpenFlight database can implement a subset of the database specification and use databases that contain that subset. Such an application would scan for the color table, polygons, and vertices and ignore the groups, objects, and other more sophisticated features described here.

Database Hierarchy

The OpenFlight database hierarchy allows the visual database to be organized in logical groupings and is designed to facilitate real-time functions such as level-of-detail switching and instancing. Each OpenFlight database is organized in a tree structure. Each node (or bead) of the tree can point down and/or across.

Header: There is one header record per file. It is always the first record in the file and represents the top of the database hierarchy and tree structure. The header always points down to a group.

Group: A group bead is used to organize a logical subset of a database. Group beads can be manipulated (translated, rotated, scaled, etc.) as a single entity. Groups can point down and across to other groups, level-of-detail (LOD) beads, or objects.

Level of detail: An level-of-detail (LOD) bead is similar to a group, but serves as a switch to turn the display of everything below it on or off based on range (the switch in/switch out distance and center location).

Degree of freedom: A degree-of-freedom (DOF) bead is similar to a group with several transformations. It is used to specify the articulation of parts in the database and set limits on the motion of those parts.

Sound: A sound bead is similar to a group, but serves as a location for a sound emitter. The emitter position is the sound offset transformed by the transformations above it in the tree (if any).

Light source: A light source bead is similar to a group, but also stores an index into the light source palette, as well as the position and direction of a light source bead. The light source position and direction are transformed by the transformations above it in the tree (if any).

Switch: A switch bead is a more general case of an LOD bead. It allows the selection of multiple children by invoking a selector mask.

Clip region: A clip bead is used to disable the drawing of geometry below this node from outside the planes specified in this bead.

Text: A text bead is used to render text in a string with a specified font, without injecting the actual polygons into the database as faces.

Object: An object bead contains a logical collection of polygons. An object can point across to another object, group, or LOD, and down to a polygon.

Polygon: A polygon bead contains a set of vertices that describe a closed polygon in a counterclockwise direction. Polygons have color, texture, materials, transparency, etc., associated with them.

Nested polygon: A nested polygon (or subface) is a bead describing a face that lies within, and is drawn on top of, another “super” polygon. Nested faces can themselves be nested. This construct is used to determine z buffer priority.

Vertex: A vertex contains a double precision coordinate x, y, and z. Some vertices also contain vertex normals and texture mapping information. Vertex records are stored in a vertex table near the beginning of the file, and are accessed through relative offset pointers after the Polygon record.

Database Files

When MultiGen writes a database to disk, it converts the tree structure to a linear stream of records. The first part of each record is a header that specifies the record opcode (e.g., its type), record length, and, in some cases, an 8-byte ASCII ID. A record containing the push opcode (or “push record”) is used to represent each down pointer. A record containing a pop opcode (or “pop record”) returns to the previous level of hierarchy.

The ID field in the record itself contains only the first seven characters of the name. If a database construct has an ID longer than eight characters, an additional record is written to the file immediately following that database node containing the full name.

If the opcode of a record is neither push nor pop, a sibling pointer is implied. Thus, a record with a polygon opcode is followed by a push record, then the vertex information describing the polygon, then a pop record. This, in turn, is followed by the polygon record for the next polygon in the same object, or by a pop record to return to object level.

OpenFlight database files have the extension .flt by convention.

Instancing

Instancing is the ability to describe a group or object one time, then display it one or more times with various transformations. OpenFlight supports instancing of objects and groups with operations such as rotate, translate, scale, and put.

In OpenFlight, a group or object definition that can be instanced is called an instance definition. An instance definition contains a record with an instance defi-

nition opcode, followed by an ID and a stand-alone database tree. An instance is invoked from a group by following the group record with a record containing a transformation matrix, and then records for each translate, rotate, and scale operation (these are for MultiGen's use and can be ignored by the real-time program), followed by an instance reference opcode and an instance ID. Instance definitions can themselves contain instance definitions and references.

OpenFlight also allows entire database files to be instanced. This is known as external referencing.

Replication

Replication is the ability to repeat the drawing of a group or object several times, applying a transformation each time. For example, a string of lights can be drawn by replicating a single light several times with a translation. In OpenFlight, replication is accomplished by following the group by one or more transformation opcode records and a replication opcode record.

Bounding Volumes

Bounding volume records can be used by the real-time software to determine if a particular group is in view. The (optional) bounding volume opcode records are placed immediately after the group record and include the extents created by instancing and replication. A bounding volume can be either a box, a sphere, or a cylinder. Each group bead can have only one bounding volume.

Header Record

The header record is at the beginning of the database file.

Latitude and longitude values are stored in the database header if it was created using the MultiGen Terrain Option.

Positive latitudes reference the northern hemisphere and negative longitudes reference the western hemisphere.

Delta x and y values are used to “place” the database when several separate databases, each of which has a local origin of zero, are used to represent an area.

Header Record Format

Data type	Length (bytes)	Description
Int	2	Header Opcode 1
Int	2	Length of the record
Char	8	ID field (Not currently used)
Int	4	Format revision level
Int	4	This database revision level
Char	32	Date and time of last revision
Int	2	Next group ID number
Int	2	Next LOD ID number
Int	2	Next object ID number
Int	2	Next polygon ID number
Int	2	Unit multiplier/divisor, always equal to 1
Int	1	Vertex coordinate units 0 = Meters 1 = Kilometers 4 = Feet 5 = Inches 8 = Nautical miles
Int	1	if TRUE set texwhite on new polygons
Boolean	4	Flags (bits, from left to right) 0 = Save vertex normals 1-31 Spare
Int	4	Not Used
Int	4	Not Used
Int	4	Not Used
Int	4	Not Used
Int	4	Not Used
Int	4	Not Used
Int	4	Projection Type 0 = Flat Earth 1 = Trapezoidal 2 = Round Earth 3 = Lambert 4 = UTM 5 = Geodetic 6 = Geocentric
Int	4	Not Used
Int	4	Not Used
Int	4	Not Used
Int	4	Not Used

Int	4	Not Used
Int	4	Not Used
Int	4	Not Used
Int	2	Next degree of freedom ID number
Int	2	Vertex Storage Type 1 = Double Precision Float
Int	4	Database Origin 100 = OpenFlight 200 = DIG I/DIG II 300 = Evans and Sutherland CT5A/ CT6 400 = PSP DIG 600 = General Electric CIV/CV / PT2000 700 = Evans and Sutherland GDF
Double	8	Southwest Database Coordinate x
Double	8	Southwest Database Coordinate y
Double	8	Delta x to Place Database
Double	8	Delta y to Place Database
Int	2	Next Sound Bead Id
Int	2	Next Path Bead ID
Int	4*2	Reserved for MultiGen
Int	2	Next Clipping Region Bead ID
Int	2	Next Text Bead ID
Int	2	Next BSP ID
Int	2	Next Switch Bead ID
Int	4	Reserved
Double	8	South West Corner Latitude
Double	8	South West Corner Longitude
Double	8	North East Corner Latitude
Double	8	North East Corner Longitude
Double	8	Origin Latitude
Double	8	Origin Longitude
Double	8	Lambert Upper Latitude
Double	8	Lambert Lower Latitude
Int	2	Next Light Source ID Number
Int	2	Reserved
Int	4	Reserved

Group Record

Group flags are available to the real-time software as follows: The animation flags specify that the beads directly below the group are an animation sequence, each bead being one frame of the sequence. The special effects IDs are normally zero, but can be set to support an application program's interpretation of the data. The group's relative priority specifies a fixed ordering of the object relative to the other groups at this level. Since MultiGen sorts based on this field before saving the database, it can be ignored by the real-time software.

The Layer field is used to assign drawing priorities to groups independent of their locations in the hierarchy.

Group Record Format

Data type	Length (bytes)	Description
Int	2	Group Opcode 2
Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	2	Group relative priority
Int	2	Spare for fullword alignment
Boolean	4	Flags (bits, from left to right) 0 = Reserved 1 = Forward animation 2 = Cycling animation 3 = Bounding box follows 4 = Freeze bounding box 5 = Default parent 6-31 Spare
Int	2	Special effects ID 1 - defined by real time
Int	2	Special effects ID 2 - defined by real time
Int	2	Significance Flags
Int	1	Layer Number
Int	1	Reserved
Int	4	Reserved

Level-of-Detail Record

The slant range distance is calculated by the real-time software by using the distance from the eyepoint to the LOD center found in the bead; this center takes instancing and replication into account. When the Use previous slant range flag is set, it means the slant range is the same as the previous LOD at the same level. This can be used to save the real-time software the calculation of redundant slant ranges when determining if a LOD should be displayed.

The Transition range specifies the distance from the eyepoint over which morphing takes place. Morph vertices are first displayed at the Switch in distance. The LOD's actual vertices are displayed when the eyepoint reaches the value computed by Switch in distance minus Transition range.

Level-of-Detail Record Format

Data type	Length (bytes)	Description
Int	2	Level of Detail Opcode 73
Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Spare
Double	8	Switch in distance
Double	8	Switch out distance
Int	2	Special effects ID 1 - defined by real time
Int	2	Special effects ID 2 - defined by real time
Boolean	4	Flags (bits, from left to right) 0 = Use previous slant range 1 = SPT flag: set to 0 for replacement LOD, 1 for additive LOD 2 = Freeze center (don't recalculate) 3-31 Spare
Double	8	Center coordinate x of LOD block
Double	8	Center coordinate y of LOD block
Double	8	Center coordinate z of LOD block
Double	8	Transition Range for Morphing

Degree-of-Freedom Record

The fields of the DOF record combine to specify a local coordinate system and the range allowed for translation, rotation, and scale with respect to that coordinate system.

The DOF record can be viewed as a list of applied transformations consisting of the following elements:

[PTTTRRRSSSP]

It is important to understand the order in which these transformations are applied to the geometry. A pre-multiplication is assumed by MultiGen, so the transformation linked list must be read backwards to describe its effect on the geometry contained below the DOF. Here, a DOF is interpreted as a Put followed by three Scales, three Rotates, three Translates, and a final Put.

Taking the transformations in reverse order, they represent:

1. A Put (3 point to 3 point transformation). This Put brings the local coordinate system to the world origin, with its x-axis aligned along the world x-axis and the local y-axis in the world x-y plane. Testing against the DOF's constraints is performed in this standard position and then the final Put repositions the local coordinate system in its original position. The first Put is therefore the inverse of the last.
2. Scale in x.
3. Scale in y.
4. Scale in z.
5. Rotation about z (yaw).
6. Rotation about y (roll).
7. Rotation about x (pitch).
8. Translation in x.
9. Translation in y.

10. Translation in z.

11. A final Put. This Put moves the DOF local coordinate system back to its original position in the scene. (See 1 above).

The DOF record specifies the minimum, maximum, and current values for each transformation. Only the current value affects the actual transformation applied to the geometry. The increment value is included to allow the setting of discrete allowable values within the range of legal values represented by the DOF.

Degree-of-Freedom Record Format

Data type	Length (bytes)	Description
Int	2	Degree of Freedom Opcode 14
Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Double	8	Origin of the DOF's local coordinate system; x coordinate
Double	8	Origin of the DOF's local coordinate system; y coordinate
Double	8	Origin of the DOF's local coordinate system; z coordinate
Double	8	Point on the x-axis of the DOF's local coordinate system; x coordinate
Double	8	Point on the x-axis of the DOF's local coordinate system; y coordinate
Double	8	Point on the x-axis of the DOF's local coordinate system; z coordinate
Double	8	Point in xy plane of the DOF's local coordinate system; x coordinate
Double	8	Point in xy plane of the DOF's local coordinate system; y coordinate
Double	8	Point in xy plane of the DOF's local coordinate system; z coordinate
Double	8	Minimum z value with respect to the local coordinate system
Double	8	Maximum z value with respect to the local coordinate system
Double	8	Current z value with respect to the local coordinate system

Double	8	Increment in z
Double	8	Minimum y value with respect to the local coordinate system
Double	8	Maximum y value with respect to the local coordinate system
Double	8	Current y value with respect to the local coordinate system
Double	8	Increment in y
Double	8	Minimum x value with respect to the local coordinate system
Double	8	Maximum x value with respect to the local coordinate system
Double	8	Current x value with respect to the local coordinate system
Double	8	Increment in x
Double	8	Minimum pitch (rotation about the x-axis)
Double	8	Maximum pitch
Double	8	Current pitch
Double	8	Increment in pitch
Double	8	Minimum roll (rotation about the y-axis)
Double	8	Maximum roll
Double	8	Current roll
Double	8	Increment in roll
Double	8	Minimum yaw (rotation about the z-axis)
Double	8	Maximum yaw
Double	8	Current yaw
Double	8	Increment in yaw
Double	8	Minimum z scale (about local origin)
Double	8	Maximum z scale (about local origin)
Double	8	Current z scale (about local origin)
Double	8	Increment for scale in z
Double	8	Minimum y scale (about local origin)
Double	8	Maximum y scale (about local origin)
Double	8	Current y scale (about local origin)
Double	8	Increment for scale in y
Double	8	Minimum x scale (about local origin)
Double	8	Maximum x scale (about local origin)
Double	8	Current x scale (about local origin)
Double	8	Increment for scale in x

Object Record

The time of day object flags can be used to inhibit display of certain objects depending on the current time of day. The illumination flag, when set, means the object is self illuminating and is not subject to normal lighting effects. The shadow flag is used to indicate the object represents the shadow of the rest of the group. When used as part of a moving model (e.g., an aircraft), the real-time software can

apply appropriate distortions to create a realistic shadow on the terrain or runway. The object's relative priority specifies a fixed ordering of the object relative to the others in its group. Since MultiGen sorts on relative priority, it can be ignored by the real-time software.

Object Record Format

Data type	Length (bytes)	Description
Int	2	Object Opcode 4
Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Boolean	4	Flags (bits from to right) 0 = Don't display in daylight 1 = Don't display at dusk 2 = Don't display at night 3 = Don't illuminate 4 = Flat shaded 5 = Group's shadow object 6-31 Spare
Int	2	Object relative priority
Int	2	Transparency factor = 0 for solid = 0xffff for totally clear
Int	2	Special effects ID 1 - defined by real time
Int	2	Special effects ID 2 - defined by real time
Int	2	Significance
Int	2	Spare

Polygon Record

If a polygon contains a non-negative material code, its apparent color is a combination of the face color and the material color, as described in the Material Record section below.

If a polygon contains a non-negative material with an alpha component and the transparency field is set, the total transparency is the product of the material alpha and the face transparency.

If a polygon is a unidirectional or bidirectional light, the polygon record is followed by a vector record (Vector Opcode 50) that contains the unit vector of the direction of the primary color. For bidirectional lights, the secondary color is in the opposite direction (180 degrees opposed).

Polygon Record Format

Data type	Length (bytes)	Description
Int	2	Polygon Opcode 5
Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	IR Color Code
Int	2	Polygon relative priority
Int	1	How to draw the polygon = 0 Draw solid backfaced = 1 Draw solid no backface = 2 Draw wireframe and not closed = 3 Draw closed wireframe = 4 Surround with wireframe in alternate color = 8 Omni-directional light = 9 Unidirectional light = 10 Bidirectional light
Int	1	Texwhite = if TRUE, draw textured polygon white (see note 1 below)
Unsigned Int	2	Primary color/intensity code
Unsigned Int	2	Secondary color code, if any
Int	1	Not used
Int	1	Set template transparency = 0 None = 1 Fixed = 3 Axis type rotate = 5 Point rotate
Int	2	Detail texture pattern no. -1 if none
Int	2	Texture pattern no. -1 if none
Int	2	Material code [0-63]. -1 if none
Int	2	Surface material code (for DFAD)
Int	2	Feature ID (for DFAD)
Int	4	IR Material codes

Int	2	Transparency = 0 for solid = 0xffff for totally clear
Int	1	Influences LOD Generation
Int	1	Linestyle Index
Bool	4	Flags (bits from to right) 0 = Terrain 1 = No Color 2 = No Alt Color 3 = RGB Mode 4-31 Spare
Int	1	Lightmode = 0 for none = 1 for color = 2 for light = 3 for both
Int	1	Reserved
Bool	6	Reserved
Int	4	Packed Color Primary (A, B, G, R)
Int	4	Packed Color Secondary (A, B, G, R)

Vertex Table

Double precision vertex records are stored in a vertex pool for the entire database. This pool is located near the beginning of the OpenFlight file, ahead of all the polygon records.

The vertex table header record signifies the start of the vertex table. It contains a one word entry specifying the total length of the vertex table, which is equal to the length of the header record plus the length of the following vertex records. The individual vertex records follow this header, each starting with its opcode. The length field in the vertex table header record makes it possible to skip over the vertex records until the data is actually needed.

Vertices may be shared, and are accessed through the vertex list record that follows each polygon record. The length of each vertex list record is determined by the number of vertices in the polygon; for each vertex, there is a one word field pointing to its vertex record in the vertex table. Since this offset includes the length of the vertex header record, the value of the first pointer is 8.

There are actually two types of vertex list records. The first type contains only the list of vertices used by the face. The second type contains both the pointers to the vertices of the face and pointers to the morph vertices. In this case, the pointers to the actual vertices of the face alternate with the morph vertex pointers.

Vertex Table Header Record Format

Data type	Length (bytes)	Description
Int	2	Shared Vertex Table Opcode 67
Int	2	Length of the record
Int	4	Length of this record plus length of the vertex pool

Followed immediately by:

Shared Vertex Record Format

Data type	Length (bytes)	Description
Int	2	Vertex Coordinate Opcode 68
Int	2	Length of the record
Unsigned Int	2	Vertex color
Bool	2	Flags (bits, from left to right) 0 = Hard edge flag 1 = Don't touch normal when shading 2 = True if no Vertex Color 3 = True if RGB colored 4-15 Spare
Double	8	x coordinate
Double	8	y coordinate
Double	8	z coordinate
Int	4	Packed color (A, B, G, R)
Int	4	Reserved

Shared Vertex Record with Normal Format

Data type	Length (bytes)	Description
Int	2	Vertex with Normal Opcode 69
Int	2	Length of the record
Unsigned Int	2	Vertex Color
Bool	2	Flags (bits, from left to right) 0 = Hard edge flag 1 = Don't touch normal when shading 2 = True if no vertex color 3 = True if RGB colored 4-15 Spare
Double	8	x coordinate
Double	8	y coordinate
Double	8	z coordinate
Float	12	Vertex normal
Int	4	Packed color (A, B, G, R)

Shared Vertex Record with Texture Format

Data type	Length (bytes)	Description
Int	2	Vertex with UV Opcode 71
Int	2	Length of the record
Unsigned Int	2	Vertex color
Bool	2	Flags (bits, from left to right) 0 = Hard edge flag 1 = Don't touch normal when shading 2 = True if no vertex color 3 = True if RGB colored 4-15 Spare
Double	8	x coordinate
Double	8	y coordinate
Double	8	z coordinate
Float	8	Texture(u,v)
Int	4	Packed color (A, B, G, R)
Int	4	Reserved

Shared Vertex Record with Normal and Texture Format

Data type	Length (bytes)	Description
Int	2	Vertex with Normal and UV Opcode 70
Int	2	Length of the record
Unsigned Int	2	Vertex color
Bool	2	Flags (bits, from left to right) 0 = Hard edge flag 1 = Don't touch normal when shading 2 = True if no vertex color 3 = True if RGB colored 4-15 Spare
Double	8	x coordinate
Double	8	y coordinate
Double	8	z coordinate
Float	12	Vertex normal
Float	8	Texture(u,v)
Int	4	Packed color (A, B, G, R)

Vertex List (Single Vertex) Format

Data type	Length (bytes)	Description
Int	2	Vertex List Opcode 72
Int	2	Length of the record
Int	4	Byte offset to this vertex record in vertex table; Number of vertices in this list is determined by: (Length of this record - 4) / 4

Vertex List (Morphing Vertex) Format

Data type	Length (bytes)	Description
Int	2	Morphing Vertex List Opcode 89
Int	2	Length of the record
Int	4	Byte offset to this vertex record in vertex table- the actual vertex of the face;

Int	4	Byte offset to the morph vertex record in the vertex table Number of vertices in this list is determined by: (Length of this record - 8) / 8

Control Records

Push Level Control Record Format

Data type	Length (bytes)	Description
Int	2	Push Level Opcode 10
Int	2	Length of the record = 4

Pop Level Control Record Format

Data type	Length (bytes)	Description
Int	2	Pop Level Opcode 11
Int	2	Length of the record = 4

Push Subface Control Record Format

Data type	Length (bytes)	Description
Int	2	Push Subface Opcode 19
Int	2	Length of the record = 4

Pop Subface Control Record Format

Data type	Length (bytes)	Description
Int	2	Pop Subface Opcode 20
Int	2	Length of the record = 4

Comment and ID Records

Comment records contain text that can follow any database element.

Comment Record Format

Data type	Length (bytes)	Description
Int	2	Text Comment Opcode 31
Int	2	Length of the record
Char	(variable)	Text description of database

Long ID records follow named database records where those IDs exceed 8 characters.

Long ID Record Format

Data type	Length (bytes)	Description
Int	2	Long Identifier Opcode 33
Int	2	Length of the record
Char	(variable)	The ID of the database element
	8.10	Key table records

Key Table Records

Key table records are used for storing variable length data records and their identifiers. The linkage editor, as well as the sound palette, use key table records. The first key table record contains the key table header as well as all the keys. This is immediately followed by one or more key table data records.

For an example of the use of Key table records, refer to the discussion of Sound below.

Key Table Header Format

Data type	Length (bytes)	Description
Int	2	X (Opcode of record using key table for storage)
Int	2	Length of the record
Int	4	Sub-Type = 1
Int	4	Max number of entries
Int	4	Number of entries
Int	4	Total length of packed data
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved

Key Records Format

Data type	Length (bytes)	Description
Int	4	Key value
Int	4	Data type
Int	4	Offset from start of packed data The offset is calculated from the start of the packed data in the data record. The length of the header information for all data records is ignored.

Key Table Data Format

Data type	Length (bytes)	Description
Int	2	X (Opcode of record using key table for storage)
Int	2	Length of the record
Int	4	Sub-Type = 2
Int	4	Data length
Char	data length	Packed Data Data is always long word aligned, with unused bytes being set to NULL.

Linkage Records

Database linkages use key table records. Linkage data consists of two different constructs: nodes and arcs. Nodes usually contain data pertaining to database entities such as degrees of freedom (DOFs). In addition, the nodes may represent modeling driver functions and code beads. The arcs contain information on how all the nodes are connected to each other. The key value is used to represent a node, an arc, or a node name, if the node represents a database entity. Names are stored as null-terminated ASCII strings.

Linkage Header Format

Data type	Length (bytes)	Description
Int	2	Linkage Record Opcode 90
Int	2	Length of the record
Int	4	Sub-Type = 1 (indicating this is a header, rather than data record)
Int	4	Max number of nodes, arcs, and entity references
Int	4	Number of nodes, arcs, and entity references
Int	4	Total length of data
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved

Immediately followed by a series of key subrecords, as below.

Key Subrecords Format

Data type	Length (bytes)	Description
Int	4	Identifier
Int	4	Data type 0x12120001 = Node data 0x12120002 = Arc data 0x12120004 = Database entity name
Int	4	Offset from start of packed data field in linkage data record

Key Subrecords repeat for all types (nodes, arcs, and entity references).

Linkage Data Record Format

Data type	Length (bytes)	Description
Int	2	Linkage Record Opcode 90
Int	2	Length of the record
Int	4	Sub-Type = 2 (indicating this is a data, rather than header, record)
Int	4	Data length
Char	data length	Packed data (in the format of Node data subrecords and Arc data subrecords, and Entity Name subrecords, as described below)

General Node Data Subrecord Format

Data type	Length (bytes)	Description
Int	4	Identifier
Int	4	Reserved
Int	4	Node type 0x12120003 = Header node 0x12120005 = Database entity node
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Sinks
Int	4	Sources
Int	4	Next node identifier
Int	4	Previous node identifier
Int	4	Arc source identifier
Int	4	Arc sink identifier

Operator Node Data Subrecord Format

Data type	Length (bytes)	Description
Int	4	Identifier
Int	4	Reserved
Int	4	Node type 0x12130001 = Plus Operator node 0x12130002 = Minus Operator node 0x12130003 = Times Operator node 0x12130004 = Divide Operator node 0x12130005 = Equal Operator node 0x12130006 = Not Equal Operator node 0x12130007 = Greater Than Operator node 0x12130008 = Less Than Operator node
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Sinks
Int	4	Sources
Int	4	Next node identifier
Int	4	Previous node identifier
Int	4	Arc source identifier
Int	4	Arc sink identifier
Float	4	Current value
Float	4	Operand value

Driver Node Data Subrecord Format

Data type	Length (bytes)	Description
Int	4	Identifier
Int	4	Reserved
Int	4	Node type 0x12140001 = Ramp Driver node 0x12140002 = Step Driver node 0x12140003 = Sine Wave Driver node 0x12140004 = Variable Driver node 0x12140005 = External File Driver node

		0x12140006 = Push Button Driver node 0x12140007 = Toggle Button Driver node
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved
Int	4	Sinks
Int	4	Sources
Int	4	Next node identifier
Int	4	Previous node identifier
Int	4	Arc source identifier
Int	4	Arc sink identifier
Float	4	Current value
Float	4	Min Amplitude
Float	4	Max Amplitude
Float	4	Wave offset
Float	4	Min time
Float	4	Max Time
Float	4	Time Steps
Float	4	Time interval
Int	4	Input trigger type
Int	4	Output trigger type
Int	4	Real-time flag

Arc Data Subrecord Format

Data type	Length (bytes)	Description
Int	4	Identifier
Int	4	Reserved
Int	4	Data type = 0x12120002
Int	4	Reserved
Int	4	Reserved
Int	4	Priority
Int	4	Source parameter
Int	4	Sink parameter
Int	4	Reserved
Int	4	Next source identifier
Int	4	Next sink identifier
Int	4	Node source identifier
Int	4	Node sink identifier

Database Entity Name Subrecord Format

Data type	Length (bytes)	Description
char	variable	Null terminated ASCII string

Color Table

The color record must follow the header record and precede the first push.

All color entries are in 'CPACK' format (alpha, blue, green, red 8-bits each). The color table consists of 512 ramped colors of 128 intensities each.

Color Table Record Format

Data type	Length (bytes)	Description
Int	2	Color Table Opcode 32
Int	2	Length of the record
Char	128	Reserved
Int	4	Brightest RGB of color 0, intensity 127
Int	4	Brightest RGB of color 1, intensity 127
etc.	...	
Int	4	Brightest RGB of color 511

Material Table

The material table contains descriptions of 64 material types. The material table is not written with the database unless a face has been assigned a non-negative material code. The appearance of a face in MultiGen is a combination of the face color and the material code. The material record must follow the header record and precede the first push. The face color is factored into the material properties as follows:

Ambient:

The displayed material's ambient component is the product of the ambient component of the material and the face color:

Displayed ambient (red) = Material ambient (red)* face color (red)

Displayed ambient (green) = Material ambient (green)* face color (green)

Displayed ambient (blue) = Material ambient (blue)* face color (blue)

For example, suppose the material has an ambient component of {1.0,.5,.5} and the face color is {100, 100, 100}. The displayed material has as its ambient color {100, 50, 50}.

Diffuse:

As with the ambient component, the diffuse component is the product of the diffuse component of the material and the face color:

Displayed diffuse (red) = Material diffuse (red)* face color (red)

Displayed diffuse (green) = Material diffuse (green)* face color (green)

Displayed diffuse (blue) = Material diffuse (blue)* face color (blue)

Specular:

Unlike ambient and diffuse components, the displayed specular component is taken directly from the material:

Displayed specular (red) = Material specular (red)

Displayed specular (green) = Material specular (green)

Displayed specular (blue) = Material specular (blue)

Emissive:

The displayed emissive component is taken directly from the material:

Displayed emissive (red) = Material emissive (red)

Displayed emissive (green) = Material emissive (green)

Displayed emissive (blue) = Material emissive (blue)

Shininess:

MultiGen drawing takes the shininess directly from the material. Specular highlights are tighter, with higher shininess values.

Alpha:

An alpha of 1.0 is fully opaque, while 0.0 is fully transparent. When drawing polygons (faces), MultiGen combines the transparency value of the polygon record with the alpha value of the material record.

The final alpha applied to a polygon as it is drawn by MultiGen is a floating point number between 0.0 (transparent) and 1.0 (opaque), and is computed as follows:

Final alpha = material alpha * (1.0 - (polygon transparency / 0xffff)).

Material Table Format

Data type	Length (bytes)	Description
Int	2	Material Table Opcode 66
Int	2	Length of the record
Float	4	Ambient red component of material 0.*
Float	4	Ambient green component of material 0.*
Float	4	Ambient blue component of material 0.*
Float	4	Diffuse red component of material 0*.
Float	4	Diffuse green component of material 0*.
Float	4	Diffuse blue component of material 0.*
Float	4	Specular red component of material 0.*
Float	4	Specular green component of material 0.*
Float	4	Specular blue component of material 0.*
Float	4	Emissive red component of material 0.*
Float	4	Emissive green component of material 0.*
Float	4	Emissive blue component of material 0.*
Float	4	Shininess. (Single precision float in the range [0.0-128.0])

Float	4	Alpha. (Single precision float in [0.0-1.0], where 1.0 is opaque)
Bool	4	Flags 0 = Materials used 1-31 Spare
Char	12	Material name
Int	4*28	Spares for material 0
Float	4	Ambient red component of material 1.*
etc.		

*Single precision floating point values, [0.0, 1.0]

Transformations

Transformation Matrix Format

Data type	Length (bytes)	Description
Int	2	Transformation Matrix Opcode 49
Int	2	Length of the record
Float	16*4	4x4 Single Precision Matrix

Note: Opcodes 40-48 are single-precision representations of Opcodes 76-82 and are no longer used. Opcodes 76-82 follow the transformation matrix and specify the individual transformations that make up the matrix.

Rotate About Edge Format

Data type	Length (bytes)	Description
Int	2	Rotate About Edge Transform Opcode 76
Int	2	Length of the record
Double	8	x, first point on edge
Double	8	y, first point on edge
Double	8	z, first point on edge
Double	8	x, second point on edge
Double	8	y, second point on edge
Double	8	z, second point on edge
Int	4	Angle by which to rotate

Scale Format

Data type	Length (bytes)	Description
Int	2	Scale Transform Opcode 77
Int	2	Length of the record
Double	8	Center x
Double	8	Center y
Double	8	Center z
Int	4	Scale factor

Translate Format

Data type	Length (bytes)	Description
Int	2	Translate Transform Opcode 78
Int	2	Length of the record
Double	8	x, reference FROM point
Double	8	y, reference FROM point
Double	8	z, reference FROM point
Double	8	Delta x to translate bead by
Double	8	Delta y to translate bead by
Double	8	Delta z to translate bead by

Scale with Independent XYZ Format

Data type	Length (bytes)	Description
Int	2	Scale with Independent XYZ Scale Opcode 79
Int	2	Length of the record
Double	8	Center x
Double	8	Center y
Double	8	Center z
Int	4	x scale factor
Int	4	y scale factor
Int	4	z scale factor

Rotate About Point Format

Data type	Length (bytes)	Description
Int	2	Rotate About Point Transform Opcode 80
Int	2	Length of the record
Double	8	x, center of rotation
Double	8	y, center of rotation
Double	8	z, center of rotation
Int	4	i, axis of rotation
Int	4	j, axis of rotation
Int	4	k, axis of rotation
Int	4	Angle by which to rotate

Rotate and/or Scale Format

Data type	Length (bytes)	Description
Int	2	Rotate and/or Scale Transform Opcode 81
Int	2	Length of the record
Double	8	x, center of scale
Double	8	y, center of scale
Double	8	z, center of scale
Double	8	x, reference point
Double	8	y, reference point
Double	8	z, reference point
Double	8	x, TO point
Double	8	y, TO point
Double	8	z, TO point
Int	4	Overall scale factor
Int	4	Scale factor in direction of axis
Int	4	Angle by which to rotate

Put Format

Data type	Length (bytes)	Description
Int	2	Put Transform Opcode 82

Int	2	Length of the record
Double	8	x, FROM origin
Double	8	y, FROM origin
Double	8	z, FROM origin
Double	8	x, FROM align
Double	8	y, FROM align
Double	8	z, FROM align
Double	8	x, FROM track
Double	8	y, FROM track
Double	8	z, FROM track
Double	8	x, TO align
Double	8	y, TO align
Double	8	z, TO align
Double	8	x, TO track
Double	8	y, TO track
Double	8	z, TO track

General Matrix Format

Data type	Length (bytes)	Description
Int	2	General Matrix Transform Opcode 94
Int	2	Length of the record
Float	16*4	4x4 Single Precision Matrix

Geometry

Vector Format

Data type	Length (bytes)	Description
Int	2	Vector Opcode 50
Int	2	Length of the record
Float	4	i component, 32 bit float
Float	4	j component
Float	4	k component

Bounding Volumes

Bounding Format

Data type	Length (bytes)	Description
Int	2	Bounding Box Opcode 74
Int	2	Length of the record
Int	4	Reserved
Double	8	x coordinate of lowest corner
Double	8	y coordinate of lowest corner
Double	8	z coordinate of lowest corner
Double	8	x coordinate of highest corner
Double	8	y coordinate of highest corner
Double	8	z coordinate of highest corner

Bounding Sphere Format

Data type	Length (bytes)	Description
Int	2	Bounding Sphere Opcode 105
Int	2	Length of the record
Int	4	Reserved
Double	8	Radius of the sphere

Bounding Cylinder Format

Data type	Length (bytes)	Description
Int	2	Bounding Cylinder Opcode 106
Int	2	Length of the record
Int	4	Reserved
Double	8	Radius of the cylinder base
Double	8	Height of the cylinder

Bounding Volume Center Format

Data type	Length (bytes)	Description
Int	2	Bounding Volume Center Opcode 108
Int	2	Length of the record
Int	4	Reserved
Double	8	x coordinate of center
Double	8	y coordinate of center
Double	8	z coordinate of center

Bounding Volume Orientation

Data type	Length (bytes)	Description
Int	2	Bounding Volume Orientation Opcode 109
Int	2	Length of the record
Int	4	Reserved
Double	8	Yaw angle
Double	8	Pitch angle
Double	8	Roll angle

Binary Separating Plane (BSP) Record

Binary separating planes (BSPs) allow you to model 3D databases with Z buffer turned off. This record contains an equation ax+by+cz+d=0 that describes the separating plane.

Binary Separating Plane Record Format

Data type	Length (bytes)	Description
Int	2	Binary Separating Plane (BSP) Opcode 55
Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Double	8	First plane equation coefficient (a)
Double	8	Second plane equation coefficient (b)

Double	8	Third plane equation coefficient (c)
Double	8	Fourth plane equation coefficient (d)

Replication and Instancing

Replicate Record

Data type	Length (bytes)	Description
Int	2	Replicate Opcode 60
Int	2	Length of the record
Int	2	Number of replications
Int	2	Spare for fullword alignment

Instance Reference Record

Data type	Length (bytes)	Description
Int	2	Local Instance Opcode 61 (Rev. 3 code = 16)
Int	2	Length of the record
Int	2	Spare
Int	2	Instance definition number

Instance Definition Record

Data type	Length (bytes)	Description
Int	2	Local Instance Library Opcode 62 (Rev. 3 code = 17)
Int	2	Length of the record
Int	2	Spare
Int	2	Instance definition number

External Reference Record

Data type	Length (bytes)	Description
Int	2	External Reference Opcode 63
Int	2	Length of the record
Char	200	199 char ASCII Path; 0 terminates
Int	1	Reserved
Int	1	Reserved
Int	2	Padding
Bool	4	Flags (bits from left to right) 0 = Color Palette Override 1 = Material Palette Override 2 = Texture Palette Override 3 = Line Palette Override 4 = Sound Palette Override 5-31 Spare

Texture Pattern File Reference

There is one record for each texture pattern referenced in the database. These records must follow the header record and precede the first push.

Texture Pattern File Reference Format

Data type	Length (bytes)	Description
Int	2	Texture Reference Record Opcode 64
Int	2	Length of the record
Char	200	Filename of texture pattern
Int	4	Pattern index
Int	4	x location in texture palette
Int	4	y location in texture palette

Add 1 to the pattern index and the polygon pattern reference number on Silicon Graphics, Inc. (SGI) machines because the texture pattern IDs start at 1.

A palette and pattern system can be used to reference the texture patterns. A texture palette is made up of 256 patterns, currently 512 texels on a side. The pattern

index for the first palette is 0 - 255, for the second palette 256 - 511, etc. Note that if less than 256 patterns exist on a palette, several pattern indices are unused. The x and y palette locations are used to store offset locations in the palette for display.

Eyepoint and Trackplane Positions

Eyepoint Position Format

Eyepoint/ Trackplane	Data type	Length (bytes)	Description
	Int	2	Eyepoint & Trackplane Position Opcode 83
	Int	2	Length of the record
	Int	4	Reserved
Last Eyepoint 0	Double	3*8	x, y, z of rotation center
	Float	3*4	Yaw, pitch, and roll angles
	Float	16*4	4x4 Single Prec. Rotation Matrix
	Float	4	Field of View
	Float	4	Scale
	Float	2*4	Near and Far clipping plane
	Float	16*4	4x4 Single Prec. Fly Through Matrix
	Float	3*4	x, y, z of eyepoint in database
	Float	4	Yaw of Fly Through
	Float	4	Pitch of Fly Through
	Float	3*4	i, j, k Vector for eyepoint direction
	Int	4	Flag (True if no Fly Through)
	Int	4	Flag (True if ortho drawing mode)
	Int	4	Flag (True if this is a valid eyepoint)
	Int	4	Image offset x
	Int	4	Image offset y
	Int	4	Image zoom
	Int	9*4	Reserved
Eyepoint 1	Same as Last Eyepoint		
Eyepoint 2	Same as Last Eyepoint		
Eyepoint 3	Same as Last Eyepoint		
Eyepoint 4	Same as Last Eyepoint		
Eyepoint 5	Same as Last Eyepoint		
Eyepoint 6	Same as Last Eyepoint		
Eyepoint 7	Same as Last Eyepoint		

Eyepoint 8	Same as Last Eyepoint		
Eyepoint 9	Same as Last Eyepoint		
Track Plane 0	Int	4	Active Flag
	Int	4	Spare
	Double	8*3	Trackplane Origin Coordinate
	Double	8*3	Trackplane Alignment Coordinate
	Double	8*3	Trackplane Plane Coordinate
	Int	4	Grid State Flag
	Int	4	Grid Under Flag
	Float	4	Grid Angle for Radial Grid
	Int	4	Reserved
	Double	8	Grid Spacing in X
	Double	8	Grid Spacing in Y
	Int	4	Snap to Grid Flag
	Double	8	Grid Size
	Int	4	Grid Spacing Direction
	Int	4	Mask for Grid Quadrants
	Int	4	Reserved
Track Plane 1	Same as Last Trackplane		
...	Same as Last Trackplane		
through	Same as Last Trackplane		
...	Same as Last Trackplane		
Track Plane 9	Same as Last Trackplane		

Sound Palette Records

The sound palette uses key table records to store the sound index and filename. The index is the Key value, and the file name is the Data record, formatted as a null-terminated ASCII string. The Sound palette header record indicates the number of sounds that have been associated with the database.

Sound Palette Header Record Format

Data type	Length (bytes)	Description
Int	2	Sound Palette Opcode 93
Int	2	Length of the record
Int	4	Sub-Type = 1 (indicating this is header rather than palette record)
Int	4	Max number of sounds
Int	4	Actual number of sounds in palette

Int	4	Total length of sound filenames.
Int	4	Reserved
Int	4	Reserved
Int	4	Reserved

Followed by a series of Sound key subrecords:

Sound Key Subrecord Format

Data type	Length (bytes)	Description
Int	4	Sound Index
Int	4	Reserved
Int	4	Data record offset from start of Packed filenames (in Sound palette data record)

Key records repeat for number of sounds.

Sound Palette Data Record Format

Data type	Length (bytes)	Description
Int	2	Sound Palette Opcode 93
Int	2	Length of the record
Int	4	Sub-Type = 2 (indicating this is palette rather than sound record)
Int	4	Filenames' length
Char	data length	Packed filenames

Sound Bead Record

Amplitude and pitch blend at the node level are relative to the amplitude in the waveform file. Priority determines which sounds are played when more emitters populate a scene than the sound system can play simultaneously. Falloff defines how amplitude falls off when approaching the edge of the sound lobe with maximum amplitude at the center of the lobe. Width defines the half angle of the sound lobe. Doppler, absorption, and delay, when TRUE, enable the modeling of doppler,

atmospheric absorption, and propagation delay in the sound environment. Direction sets the type of sound lobe -- omnidirectional = 0, bidirectional = 1, and unidirectional = 2. Active indicates a sound is to be activated when read in.

Sound Record Format

Data type	Length (bytes)	Description
Int	2	Sound Bead Opcode 91
Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Spare
Int	4	Index into sound palette
Int	4	Reserved
Double	8	x coordinate of offset from local origin
Double	8	y coordinate of offset from local origin
Double	8	z coordinate of offset from local origin
Int	4	Spare
Float	4	i component of sound direction wrt local coordinate axes
Float	4	j component of direction wrt local coordinate axes
Float	4	k component of direction wrt local coordinate axes
Float	4	amplitude of sound
Float	4	pitch bend of sound
Float	4	priority of sound
Float	4	falloff of sound
Float	4	width of sound lobe
Boolean	4	Flags (bits, from left to right) 0 = doppler 1 = atmospheric absorption 2 = delay 3-4 = direction: 0 = omnidirectional, 1 = bidirectional 2 = unidirectional 5 = active 6-31 Spare

Sound Files

Sound file formats will be addressed in a future revision of this specification.

Light Source Palette Records

Each of these records represents a new entry in the light source palette. Entries may be referenced by light source beads using the palette index. Lights can be flagged as modeling lights, which are used to illuminate a scene without having to be stored as part of the hierarchy. A modeling light is always positioned at the eye; its direction is stored in the palette. A light referenced by a bead obtains its position and direction from the bead. In this case, the palette yaw and pitch components are ignored.

Light Source Palette Element Record Format

Data type	Length (bytes)	Description
Int	2	Light Source Palette Opcode 102
Int	2	Length of the record
Int	4	Palette index
Int	2*4	Reserved
Char	20	Light source name
Int	4	Reserved
Float	4*4	Ambient RGBA (alpha component is currently unused)
Float	4*4	Diffuse RGBA (alpha component is currently unused)
Float	4*4	Specular RGBA (alpha component is currently unused)
Int	4	Light type 0 = INFINITE, 1 = LOCAL 2 = SPOT
Int	10*4	Reserved
Float	4	Spot exponential dropoff term
Float	4	Spot cutoff angle (in degrees)
Float	4	Yaw
Float	4	Pitch
Float	4	Constant attenuation coefficient
Float	4	Linear attenuation coefficient
Float	4	Quadratic attenuation coefficient
Boolean	4	Modeling Light (TRUE/FALSE)
Int	19*4	Spare

Light Source Bead Records

Light source beads are similar to other bead types. They are comprised of position and rotation data (overriding any information stored in the light palette), an index into the light palette, and information on how the light behaves within the hierarchy. The enabled flag indicates whether the bead is turned on (displayed). The global flag specifies whether the light shines only on its children (for example, the cabin light in a car) or has global influence.

Light Source Bead Record Format

Data type	Length (bytes)	Description
Int	2	Light Source Record Opcode 101
Int	2	Length of the record
Char	8	7 char ASCII ID; 0 terminates
Int	4	Reserved
Int	4	Index into light palette
Int	4	Reserved
Char	4	Flags (bits, from left to right) 0 = enabled 1 = global 2 = reserved 3 = export 4 = reserved 5-31 Spare
Int	4	Reserved
Double	3*8	XYZ coordinates
Float	4	Yaw
Float	4	Pitch

Road Segment Records

Road Record Opcode 87 stores the parameters used to generate a road segment. This opcode is for MultiGen use only and should be ignored by the real-time software.

Path Beads

The faces grouped under a road path bead are paths for the lanes of traffic.

Path Bead Record Format

Data type	Length (bytes)	Description
Int	2	Road Path Bead Opcode 92
Int	2	Length of record
Char	8	ASCII ID
Int	4	Reserved
Char	120	Pathname
Double	8	Speed limit
Int	4	No passing flag
Char	484	Spare

Path beads may also be written to an ASCII file for easy access by the real-time software. The keywords in the file are:

SPEED

NO PASSING

X Y Z I J K

.

.

XYZ is the position of the path and IJK is the normal at that location. For example:

SPEED = 50.00

NO PASSING = FALSE

23.23 34.45 78.54 0.885 0.123 0.222

34.45 44.34 99.99 0.456 0.345 0.324

Zone Files

Zone files are gridded posts files containing elevation and attribute data for the road.

Zone File Format

Data type	Length (bytes)	Description
Int	4	Version (road tools format revision)
Int	4	Spare
Double	8	Lower left corner x, y, z
Double	8	
Double	8	
Double	8	Upper right corner x, y, z
Double	8	
Double	8	
Double	8	Grid interval (spacing between data points)
Int	4	Number of data points in x
Int	4	Number of data points in y
Float	4	Low z elevation data point
Float	4	High z elevation data point
Char	440	Spare

Followed immediately by a series of (number of spaces in x + 1) * (number of spaces in y + 1) elevation data points. Data begins at the lower left corner. Succeeding values go from bottom to top, then in columns from left to right.

Elevation Data Point Format

Data type	Length (bytes)	Description
Float	4	Z elevation value

Followed immediately by a series of (number of spaces in x + 1) * (number of spaces in y + 1) surface types corresponding to each of the elevation data points above.

Surface Type Format

Data type	Length (bytes)	Description
Char	1	Road surface type (user defined)

Line Style Records

Line style records define the outline displayed around polygons in wireframe or wireframe over solid mode. The Pattern field defines a mask to control the display of segments of the line. For example, if all the bits of the mask are set, the line is drawn as a solid line. If every other bit is on, the line is displayed as a dashed line. The Line Width field controls the width of the line in pixels. Line style 0 is the default. Polygons are assigned line styles in the Line Style field of the polygon record. One of these records appears for each line style defined in the OpenFlight file.

Line Style Record

Data type	Length (bytes)	Description
Int	2	Line Style Record Opcode 97
Int	2	Length of record
Int	2	Line Style Index
Int	2	Pattern Mask
Int	4	Line Width

Clipping Regions Records

Clipping region records define those regions in 3D space outside which no drawing may occur. Clipping regions only clip the geometry below the clip bead in the hierarchy. The coordinates create a four-sided polygon that defines the clip region in space. The planes are formed along the edges of the four-sided polygon normal to the polygon and the fifth plane clips the back side of the polygon.

Clipping Region Records

Data type	Length (bytes)	Description
Int	2	Clipping Quadrilateral Bead Opcode 98
Int	2	Length of record
Char	8	ASCII ID
Int	2	Unused
Char	5	Flags for Enabling the Individual Clip Planes (char 1 is the plane on edge defined by the 1st two

		coordinates etc.; char 5 enables the plane that clips the half space behind the polygon)
Char	1	Unused
Double	4*3	Four Coordinates Defining the Clip Region (x0, y0, z0, x1, y1, z1, x2...)
Double	5*4	Five Plane Equation Coefficients (ax + by +cz +d) (a0, a1, a2, a3, a4, b0, b1, b2, b3, b4, c0, c1, c2, c3, c4, d0, d1, d2, d3, d4)

Font and Text Records

Text records in OpenFlight are used to render a string of data using a specified font. The record specifies the visual characteristics of the text, in addition to formatting information. The actual string for the text is stored in the comment record immediately following. The format of the text record is:

Text Record

Data type	Length (bytes)	Description
Int	2	Text Bead Opcode 95
Int	2	Length of record
Char	8	ASCII ID
Int	4	Spare
Int	4	Type -1 = Static 0 = Text String 1 = Float
Int	4	Draw Type 0 = Solid 1 = Wireframe Closed 2 = Wireframe Unclosed 3 = Surround with Wireframe in Alt Color
Int	4	Justification 0 = Left Justify 1 = Right Justify 2 = Center Justify
Double	8	Floating Point Value
Int	4	Integer Value
Int	5*4	Used by Formatter Routines
Int	4	Flags

		Bit 0 = Boxable (Unused) Bits 1-31 Spare
Int	4	Color
Int	4	Color 2 (Unused)
Int	4	Material
Int	4	Spare
Int	4	Maximum Number of Lines (Unused)
Int	4	Maximum Number of Characters
Int	4	Current Length of Text (Unused)
Int	4	Next Line Number Available (Unused)
Int	4	Line Number At Top of Display (Unused)
Int	2*4	Low/High Values for Integers
Double	8*2	Low/High Values for Doubles
Double	3*8	Lower left rectangle around Text
Double	3*8	Upper right rectangle around Text
Char	120	Fontname
Int	4	Draw Vertical
Int	4	Draw with Italic Slant Factor
Int	4	Draw with Underline
Int	4	Linestyle

Switch Records

A switch bead is a set of masks that controls the display of its children. The mask may inhibit the display of some, none, or all of the switch bead children. The index of the current selected mask is the Index of Current Mask field.

Switch Record Format

Data type	Length (bytes)	Description
Int	2	Switch Bead Opcode 96
Int	2	Length of record
Char	8	ASCII ID
Int	4	Index of Current Mask
Int	4	Number of Words Required for Each Mask (Number of Children / 32 + Number of Children% 32)
Int	4	Number of Masks
etc.	variable	First Mask (Length = Number of Words Required for Each Mask)

2 Texture Files

Texture Pattern Files

OpenFlight does not have its own texture pattern format, but rather uses existing texture formats and references patterns by filename. (See "Texture Pattern File Reference," page 35.) File formats currently supported include:

- AT & T image 8 format (8-bit color lookup)
- AT & T image 8 template format
- SGI intensity modulation
- SGI intensity modulation with alpha
- SGI RGB
- SGI RGB with alpha

The format of the file can be determined by the file name extension, the magic numbers within the file, or the texture attribute file, as described below.

Texture Attribute Files

A corresponding attribute file is created for each texture pattern, with the name of the attribute file the same as the texture file, followed by the extension .attr. These attribute files are used by MultiGen, and may not be necessary for the real-time software using the database. They are in the following format:

Texture Attribute File Format

Data type	Length (bytes)	Description
Int	4	Number of texels in u direction
Int	4	Number of texels in v direction
Int	4	Real world size u direction
Int	4	Real world size v direction
Int	4	x component of up vector
Int	4	y component of up vector
Int	4	File format type

		-1 Not used 0 AT & T image 8 pattern 1 AT & T image 8 template 2 SGI intensity modulation 3 SGI intensity w/ alpha 4 SGI RGB 5 SGI RGB w/ alpha
Int	4	Minification filter type: 0 - TX_POINT 1 - TX_BILINEAR 2 - TX_MIPMAP (Obsolete) 3 - TX_MIPMAP_POINT 4 - TX_MIPMAP_LINEAR 5 - TX_MIPMAP_BILINEAR 6 - TX_MIPMAP_TRILINEAR 7 - None 8 - TX_BICUBIC 9 - TX_BILINEAR_GEQUAL 10- TX_BILINEAR_LEQUAL 11 - TX_BICUBIC_GEQUAL 12 - TX_BICUBIC_LEQUAL
Int	4	Magnification filter type: 0 - TX_POINT 1 - TX_BILINEAR 2 - None 3 - TX_BICUBIC 4 - TX_SHARPEN 5 - TX_ADD_DETAIL 6 - TX_MODULATE_DETAIL 7 - TX_BILINEAR_GEQUAL 8 - TX_BILINEAR_LEQUAL 9 - TX_BICUBIC_GEQUAL 10 - TX_BICUBIC_LEQUAL
Int	4	Repetition type: 0 - TX_REPEAT 1 - TX_CLAMP 2 - (Obsolete)
Int	4	Repetition type in u direction (See Above)
Int	4	Repetition type in v direction (See Above)
Int	4	Modify flag (for internal use)
Int	4	x Pivot point for rotating textures
Int	4	y Pivot point for rotating textures

Int	4	Environment type: 0 - TV_MODULATE 1 - TV_BLEND 2 - TV_DECAL 3 - TV_COLOR
Int	4	TRUE if intensity pattern to be loaded in alpha with white in color.
Int	8*4	8 words of spare.
Double	8	Real world size u for Floating point databases.
Double	8	Real world size v for Floating point databases.
Int	4	Code for origin of imported texture.
Int	4	Kernel version number.
Int	4	Internal Format type: 0 - default 1 - TX_I_12A_4 2 - TX_IA_8 3 - TX_RGB_5 4 - TX_RGBA_4 5 - TX_IA_12 6 - TX_RGBA_8 7 - TX_RGBA_12 8 - TX_I_16 (shadow mode only) 9 - TX_RGB_12
Int	4	External Format type: 0 - default 1 - TX_PACK_8 2 - TX_PACK_16
Int	4	Boolean TRUE if using following 8 floats for MIPMAP kernel.
Float	8*4	8 Floats for kernel of separable symmetric filter.
Int	4	Boolean if TRUE send:
Float	4	LOD0 for TX_CONTROL_POINT
Float	4	SCALE0 for TX_CONTROL_POINT
Float	4	LOD1 for TX_CONTROL_POINT
Float	4	SCALE1 for TX_CONTROL_POINT
Float	4	LOD2 for TX_CONTROL_POINT
Float	4	SCALE2 for TX_CONTROL_POINT
Float	4	LOD3 for TX_CONTROL_POINT
Float	4	SCALE3 for TX_CONTROL_POINT
Float	4	LOD4 for TX_CONTROL_POINT
Float	4	SCALE4 for TX_CONTROL_POINT
Float	4	LOD5 for TX_CONTROL_POINT
Float	4	SCALE5 for TX_CONTROL_POINT
Float	4	LOD6 for TX_CONTROL_POINT

Float	4	SCALE6 for TX_CONTROL_POINT
Float	4	LOD7 for TX_CONTROL_POINT
Float	4	SCALE7 for TX_CONTROL_POINT
Float	4	clamp
Int	4	magfilteralpha: 0 = TX_POINT 1 = TX_BILINEAR 2 = None 3 = TX_BICUBIC 4 = TX_SHARPEN 5 = TX_ADD_DETAIL 6 = TX_MODULATE_DETAIL 7 = TX_BILINEAR_GEQUAL 8 = TX_BILINEAR_LEQUAL 9 = TX_BICUBIC_GEQUAL 10 = TX_BIBICUBIC_LEQUAL
Int	4	magfiltercolor: 0 = TX_POINT 1 = TX_BILINEAR 2 = None 3 = TX_BICUBIC 4 = TX_SHARPEN 5 = TX_ADD_DETAIL 6 = TX_MODULATE_DETAIL 7 = TX_BILINEAR_GEQUAL 8 = TX_BILINEAR_LEQUAL 9 = TX_BICUBIC_GEQUAL 10 = TX_BIBICUBIC_LEQUAL
Float	22*4	spare
Int	4	Boolean TRUE if using next 5 integers for Detail Texture
Int	4	J argument for TX_DETAIL.
Int	4	K argument for TX_DETAIL
Int	4	M argument for TX_DETAIL
Int	4	N argument for TX_DETAIL
Int	4	Scramble argument for TX_DETAIL
Int	4	Boolean TRUE if using next for floats for TX_TILE.
Float	4	Lower left u value for TX_TILE
Float	4	Lower left v value for TX_TILE
Float	4	Upper right u value for TX_TILE
Float	4	Upper right v value for TX_TILE

Int	160*4	spare
Char	512*1	Comments.
Int	13*4	Reserved

The attribute file determines how to parse the texture pattern file and set the texture hardware and software environment for a specific pattern.

3 OpenFlight Opcodes

Valid Opcodes

Opcode	Function
1	Header Opcode
2	Group Opcode
4	Object Opcode
5	Polygon Opcode
10	Push Level Opcode
11	Pop Level Opcode
14	Degree of Freedom Opcode
19	Push Subface Opcode
20	Pop Subface Opcode
31	Text Comment Opcode
32	Color Table Opcode
33	Long Identifier Opcode
49	Transformation Matrix Opcode
50	Vector Opcode
55	Binary Separating Plane (BSP) Opcode
60	Replicate Opcode
61	Local Instance Opcode
62	Local Instance Library Opcode
63	External Reference Opcode
64	Texture Reference Record Opcode
66	Material Table Opcode
67	Shared Vertex Table Opcode
68	Vertex Coordinate Opcode
69	Vertex with Normal Opcode
70	Vertex with Normal and UV Opcode
71	Vertex with UV Opcode
72	Vertex List Opcode
73	Level Of Detail Opcode
74	Bounding Box Opcode
76	Rotate about Edge Transform Opcode
77	Scale Transform Opcode
78	Translate Transform Opcode
79	Scale with Independent XYZ Scale Opcode
80	Rotate about Point Transform Opcode
81	Rotate and or Scale Transform Opcode

82	Put Transform Opcode
83	Eyepoint & Trackplane Position Opcode
87	Road Record Opcode
89	Morphing Vertex List Opcode
90	Linkage Record Opcode
91	Sound Bead Opcode
92	Road Path Bead Opcode
93	Sound Palette Opcode
94	General Matrix Transform Opcode
95	Text Bead Opcode
96	Switch Bead Opcode
97	Line Style Record Opcode
98	Clipping Quadrilateral Bead Opcode
101	Light Source Record Opcode
102	Light Source Palette Opcode
103	Delaunay Header Opcode (Internal Use Only)
104	Delaunay Points Opcode (Internal Use Only)
105	Bounding Sphere Opcode
106	Bounding Cylinder Opcode
107	Reserved
108	Bounding Volume Center Opcode
109	Bounding Volume Orientation Opcode
110	Reserved

Obsolete Opcodes

Opcode	Function
3	Level of Detail
6	Vertex with ID
7	Short Vertex
8	Vertex with Color
9	Vertex with Color and Normal
12	Translate
13	Degree of Freedom
16	Local Instance Record
17	Local Instance Bead
40	Translate
41	Rotate about Point Transform
42	Rotate about Edge Transform
43	Scale Transform
44	Translate Transform
45	Scale Transform with Independent XYZ
46	Rotate about Point Transform
47	Rotate/Scale to Point Transform
48	Put Transform
51	Bounding Box

Index

A

Alpha **27**
Ambient **26**
Arc data subrecord format **24**

B

Binary separating plane (BSP) record **33**
Binary separating plane record format **33**
Bounding formats **32**
Bounding volume opcode record **4**

C

Clip region **2**
Clipping region records **44**
Color table **25**
Color table record format **25**
Comment record format **19**
Control record formats **18**

D

Database entity name subrecord format **25**
Database files, on disk **3**
Database hierarchy **1**
Degree of freedom **2**
Degree-of-freedom record format **10**
Diffuse **26**
Driver node data subrecord format **23**

E

Elevation data point format **43**
Emissive **26**
External reference record **35**
External referencing **4**
Eyepoint position format **36**

G

General matrix format **31**
General node data subrecord format **22**
Geometry record **31**
Group **2**
Group record format **7**

H

Header **2**
Header record format **5**

I

Instance definition **3**
Instance definition record **34**
Instance reference record **34**
Instancing **3, 34**

K

Key records format **20**
Key subrecords format **21**
Key table data format **20**
Key table header format **20**

L

Level of detail **2**
Level-of-detail record format **8**
Light source **2**
Light source bead record format **41**
Light source bead records **41**
Light source palette element record format **40**
Light source palette records **40**
Line style record **44**
Line style records **44**
Linkage data record format **22**
Linkage header format **21**
Linkage records **21**
Long ID record format **19**

M

Material table **25**
Material table format **27**

N

Nested polygon **3**

O

Object **2**
Object record format **12**
Obsolete OpenFlight opcodes **55**
Operator node data subrecord format **23**

P

Path bead record format **42**
Polygon **2**
Polygon record format **13**
Put format **30**

R

Replicate record **34**
Replication **4, 34**
Road segment records **41**
Rotate about edge format **28**
Rotate about point format **30**
Rotate and/or scale format **30**

S

Scale format **29**
Scale with independent XYZ format **29**
Shared vertex record format **15**
Shared vertex record with normal and texture format **17**
Shared vertex record with normal format **16**
Shared vertex record with texture format **16**
Shininess **27**
Sound **2**
Sound bead records **38**
Sound files **39**
Sound key subrecord format **38**
Sound palette data record format **38**
Sound palette header record format **37**
Sound palette records **37**
Sound record format **39**
Specular **26**
Surface type format **43**
Switch **2**
Switch record format **46**

T

Text **2**
Text record **45**
Text records **45**
Texture attribute file format **47**
Texture attribute files **47**
Texture pattern file reference format **35**
Texture pattern files **47**
Trackplane position format **36**
Transformation matrix format **28**
Transformations **28**
Translate format **29**

V

Vector **31**
Vector format **31**
Vertex **3**
Vertex list formats **17**
Vertex table header record format **15**

Z

Zone file format **43**
Zone files **42**