Cimarron Systems

Elements of the H.264 Video/AAC Audio MP4 Movie

Application Note: AN101

April 28, 2014

Cimarron Systems, LLC

Table of Contents

List of Figures

Document**:** AN-140428-1
Document Status: Released
Revision: 1.0

Construction of the MP4 Movie Container

The present application note describes the core elements of a canonically constructed MPEG-4 movie using H.264 video and AAC-LC audio encapsulated in a media container commonly termed a MP4 movie. The MP4 movie elements described here were generated by the RTP A/V Streaming Client/Server Applications—available as part of the Cimarron Systems Digital Media Software Development Kits (DMSDK)—and, while the MP4 movie container complies with ISO/IEC 14496-12 *Information technology — Coding of audio-visual objects Part 12: ISO base media file format* and the *QuickTime Movie File Format Specification*, Apple, August, 2010, the discussion here focuses on a subset of the container elements defined in both specifications.

DMSDK Development Environment

Figure 1 shows a context diagram of the typical development environment in which the DMSDK operates, including: the TM320DM36x DVEVM running a number of the DMSDK components, a Ubuntu Linux Host computer, an audio/video source, and a video display/audio output device.

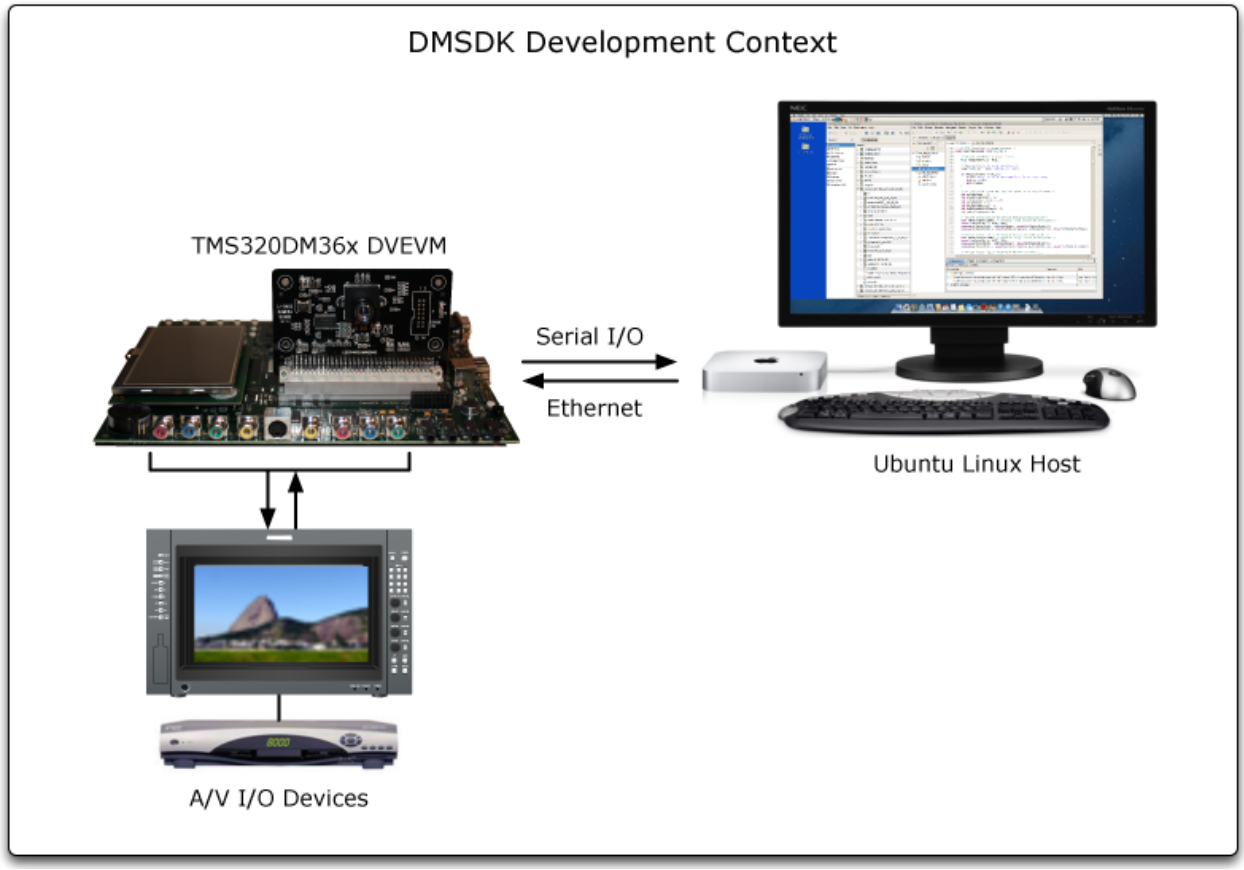

Figure 1: DMSDK Development Context Diagram.

MP4 Atom Descriptions

As illustrated in Figure 2, the MP4 movie container file format has a very specific hierarchy of elements called Atoms (in MPEG-4 terminology, they are called Boxes). Using Apple's naming convention, each of these movie container elements are described briefly below (Color Code for Atom Description Figures: blue represents the 'atom size'; green represents the 'atom content'; yellow represents the 'NAL Unit Size'; and watermelon represents the NAL Unit Byte Sequence).

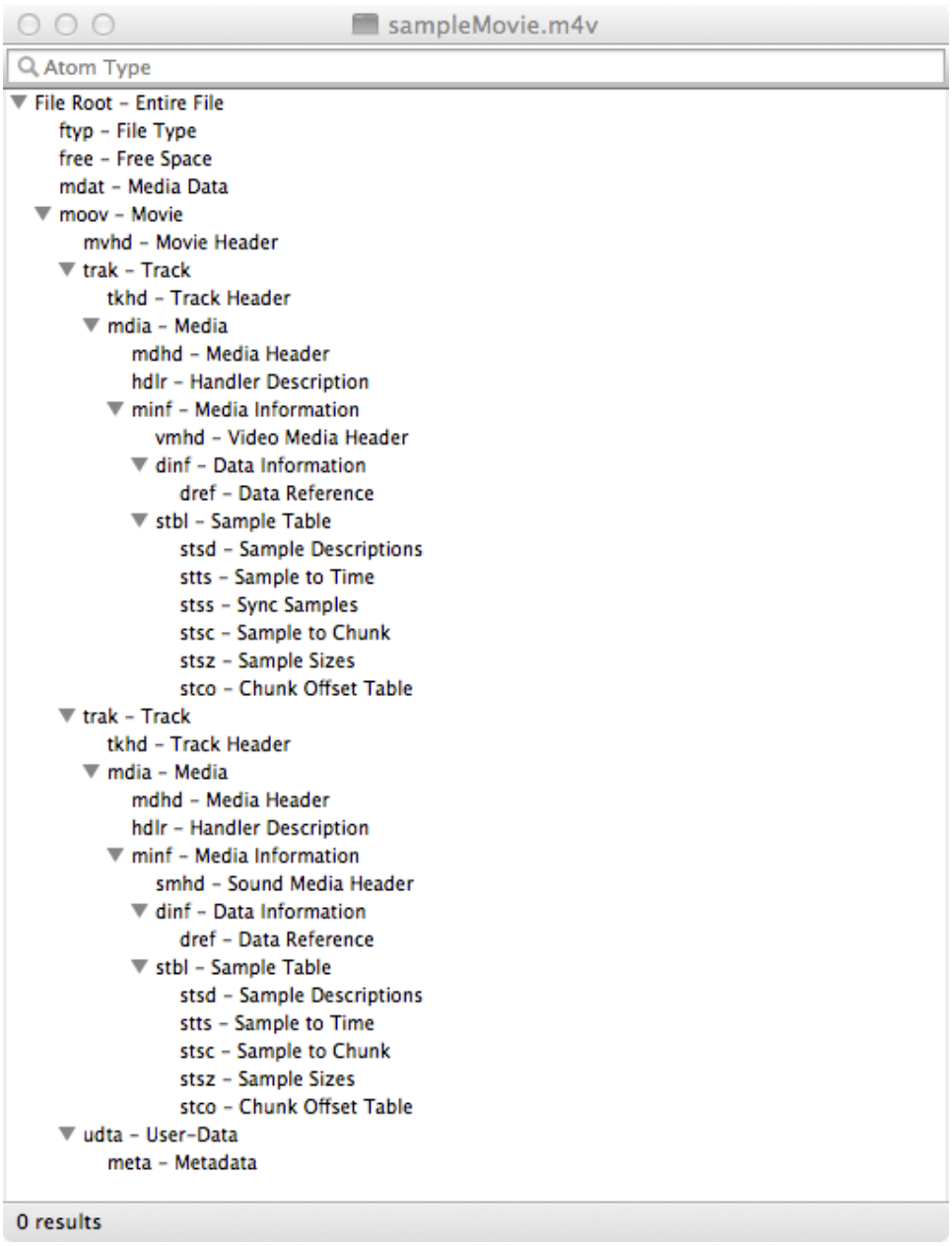

Figure 2: MP4 Movie Container Overview.

File Type, Free, and Media Data Atoms

Figure 3 illustrates the general structure of the MP4 movie container file structure, specifically the: 'ftyp' atom; free' atom; 'mdat' A/V media data block; and two full (plus one partial) H.264 NAL Units .

- **ftyp** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this File Type atom; set here to '0x0000001C'.
  - Type—A 32-bit unsigned integer that identifies the atom type, represented as a four-character code; this field must be set to 'ftyp'.
  - Major Brand—A 32-bit unsigned integer that identifies the movie file type, represented as a four-character code; is set to 'm4v ' (note the trailing ASCII space character).

- Minor Version— A 32-bit unsigned integer that identifies the movie file type Minor Version, represented as a four-byte number represented in binary coded decimal (BCD) form; is set to '0x00000200'.
- Compatible Brands— A series of unsigned 32-bit integers listing compatible file formats; set here to 'isom', 'iso2', and 'avc1'.

▪ **free** contains the following fields:

- Size—A 32-bit unsigned integer that specifies the number of bytes in this Free atom; set here to '0x00000008'.
- Type—A 32-bit unsigned integer that identifies the atom type, represented as a four-character code; this field must be set to 'free'.
- Free Space—The number of bytes of Free Space; a place holder not allocated here.

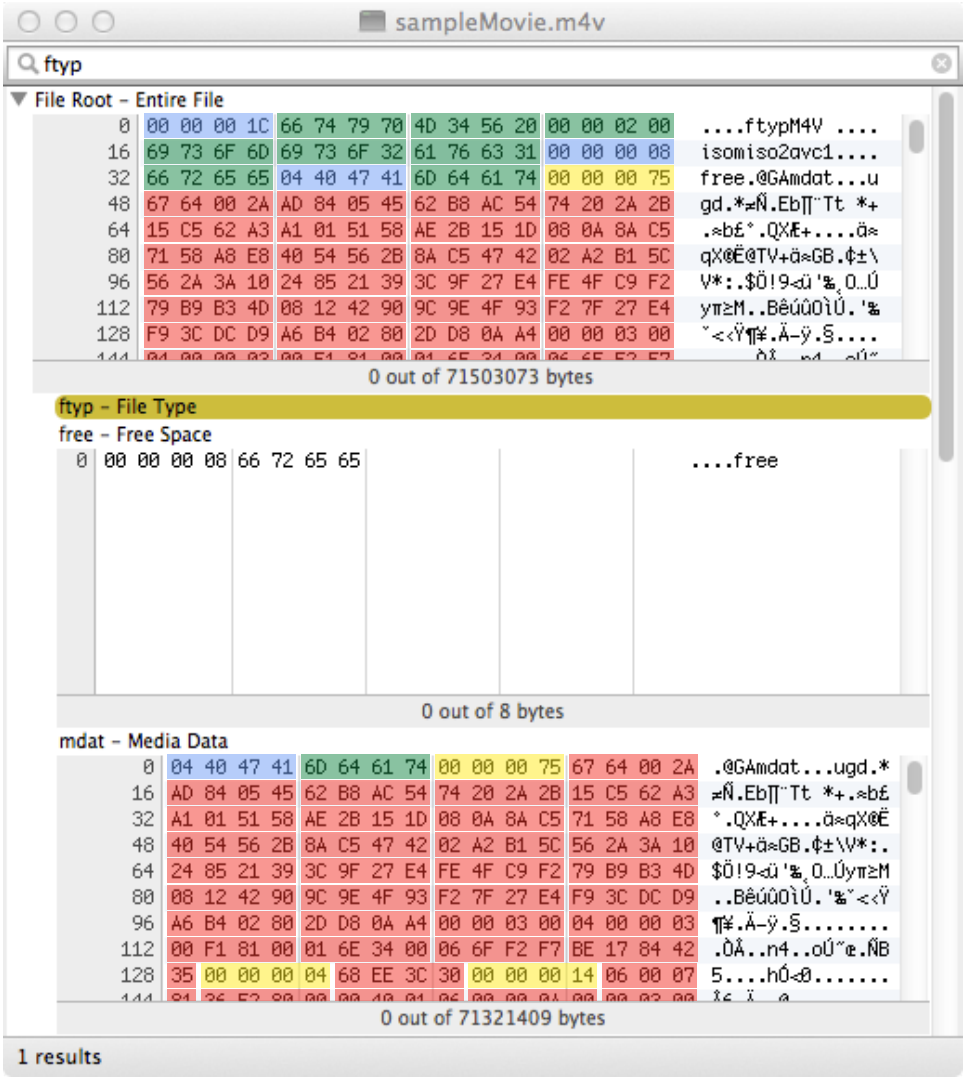

Figure 3: MP4 Movie Container 'ftyp', 'free', and 'mdat' Structures.

▪ **mdat** contains the following fields:

- Size—A 32-bit unsigned integer that specifies the number of bytes in this Media Data portion of the MP4 movie file; set here to '0x04404741'.

- Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'mdat'. Note: 'mdat' is technically not an atom rather it is the actual A/V Media Data that other atoms may reference via a byte offset into the data block.

Movie and Movie Header Atoms

Figure 4 shows the structure of the 'moov' and 'mvhd' atoms:

- **moov** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Movie atom; set here to '0x0002C57C'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'moov'.

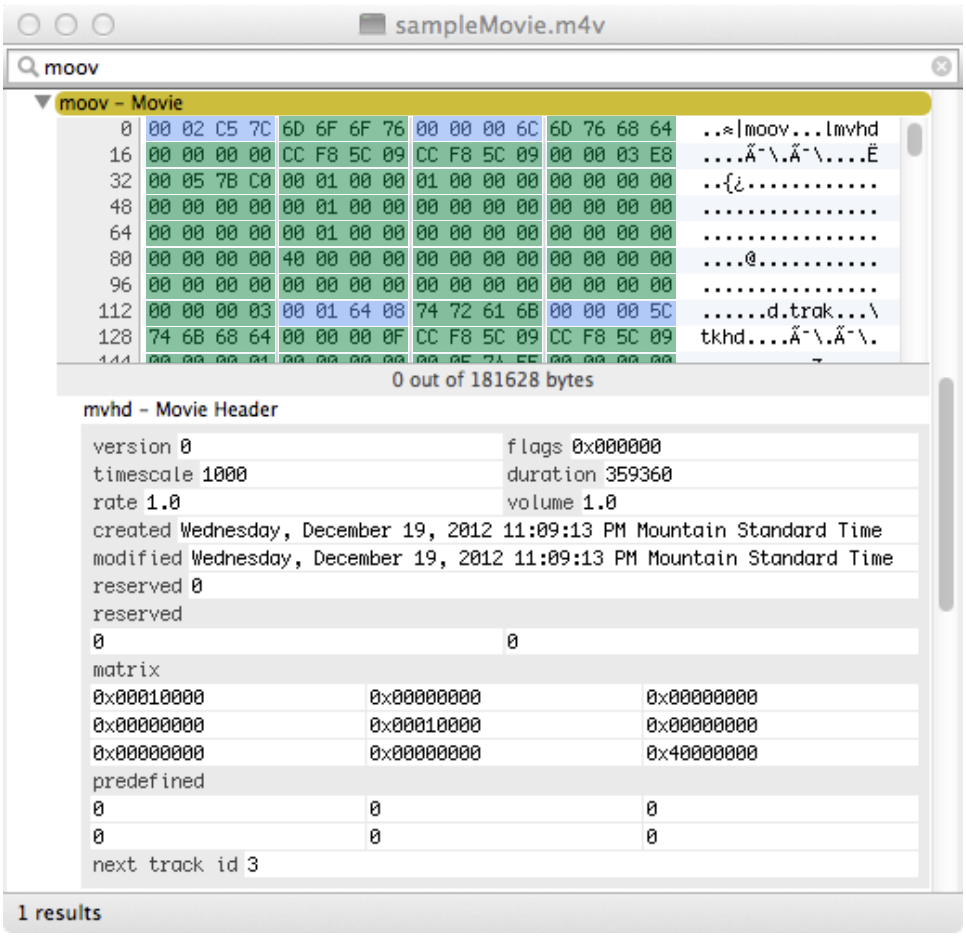

Figure 4: MP4 Movie Container 'moov' and 'mvhd' Structures.

- **mvhd** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Movie Header atom; set here to '0x0000006C'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'mvhd'.
  - Version—A 1-byte specification of the version of this Movie Header atom; set here to '0x00'.
  - Flags—Three bytes of space for future movie header flags; set here to '0x000000'.
  - Creation Time—A 32-bit integer that specifies the calendar date and time (in seconds since

midnight, January 1, 1904) when the movie atom was created in coordinated universal time (UTC); set here to '0xCCF85C09'.

- Modification Time—A 32-bit integer that specifies the calendar date and time (in seconds since midnight, January 1, 1904) when the movie atom was created in coordinated universal time (UTC); set here to '0xCCF85C09'.
- Time Scale—A time value that indicates the time scale for this movie—that is, the number of time units that pass per second in its time coordinate system; set here to '0x000003E8'.
- Duration—A time value that indicates the duration of the movie in time scale units, derived from the movie's tracks, corresponding to the duration of the longest track in the movie; set here to '0x00057BC0'.
- Preferred Rate— A 32-bit fixed-point number that specifies the rate at which to play this movie (a value of 1.0 indicates normal rate); set here to '0x00010000'.
- Preferred Volume—A 16-bit fixed-point number that specifies how loud to play this movie's sound (a value of 1.0 indicates full volume); set here to '0x0100'.
- Reserved—Ten reserved bytes set to zero.
- Matrix—A transformation matrix that defines how to map points from one coordinate space into another coordinate space (please reference to the *QuickTime File Format Specification* for details).
- Predefines—Media Header predefines; set to zero (please refer to the *QuickTime File Format Specification* for details).
- Next Track ID—The number of the Next Track ID; set here to **'3'**.

Track and Track Header Atoms

Figure 5 shows the structure of the 'trak' and 'tkhd' atoms:

- **trak** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this first Movie Track atom; set here to '0x00016408'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'trak'.
- **tkhd** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this first Movie Track Header atom; set here to '0x0000005C'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'tkhd'.
  - Version—A 1-byte specification of the version of this first Movie Track Header atom; set here to **'0x00'**.
  - Flags—Three bytes specification of the version of this first Movie Track Header atom; set here to 0x00000F (please refer to the *QuickTime File Format Specification* for details).
  - Creation Time—A 32-bit integer that specifies the calendar date and time (in seconds since midnight, January 1, 1904) when the movie atom was created in coordinated universal time (UTC); set here to '0xCCF85C09'.
  - Modification Time—A 32-bit integer that specifies the calendar date and time (in seconds since midnight, January 1, 1904) when the movie atom was created in coordinated universal time (UTC); set here to '0xCCF85C09'.
  - Track ID—A 32-bit integer that uniquely identifies the track; the value 0 cannot be used.
  - Reserved—A 32-bit integer that is reserved; this field is set to '0x00000000'.
  - Duration—A time value that indicates the duration of this track (in the movie's time coordinate system). Note that this property is derived from the track's edits: the value of this field is equal to the sum of the durations of all of the track's edits and that if there is no edit list, the duration is the

sum of the sample durations, converted into the movie timescale; set here to '0x00057AFE'.

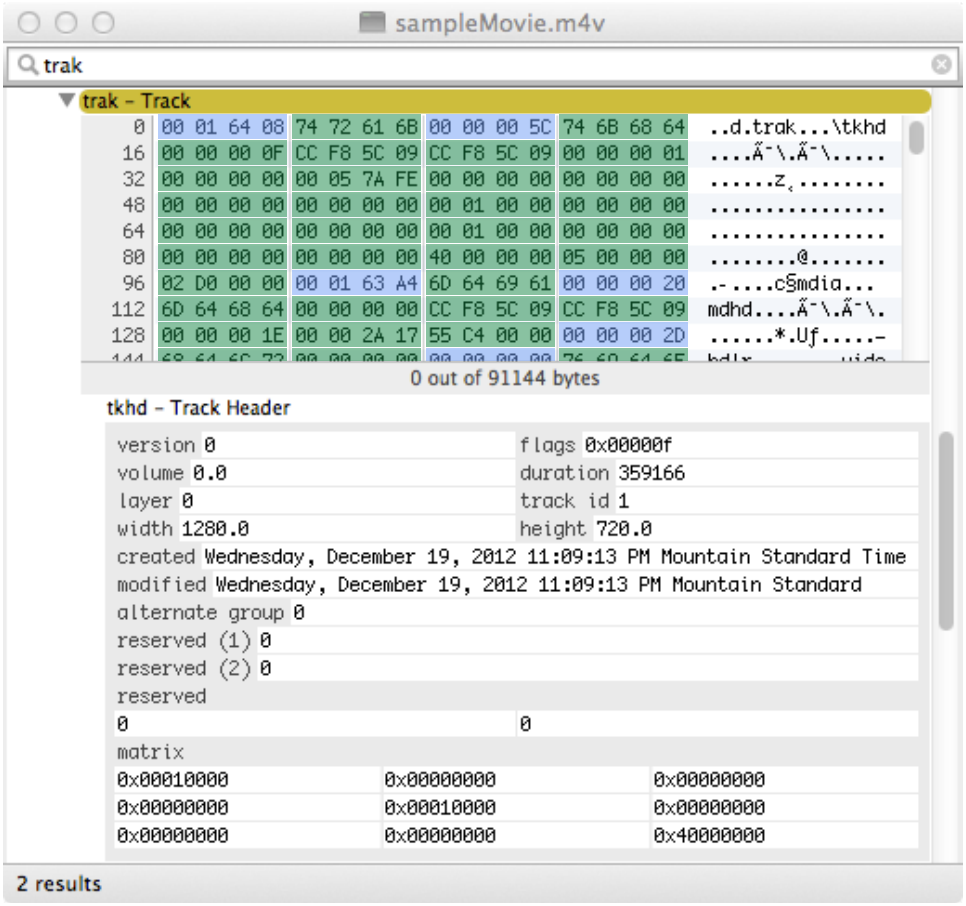

Figure 5: MP4 Movie Container 'trak' and 'tkhd' Structures.

- Reserved—An 8-byte value that is reserved; this field is set to 0.
- Layer—A 16-bit integer that indicates this track's spatial priority in its movie (the QuickTime Movie Toolbox uses this value to determine how tracks overlay one another). Tracks with lower layer values are displayed in front of tracks with higher layer values.
- Alternative Group— A 16-bit integer that specifies a collection of movie tracks that contain alternate data for one another; set here to '0x0000'.
- Volume—A 16-bit fixed-point value that indicates how loudly this track's audio is to be played; a value of 1.0 indicates normal volume.
- Reserved—A 16-bit integer that is reserved; this field is set to '0x0000'.
- Matrix Structure—The matrix structure associated with this track (please refer to the *QuickTime File Format Specification* for details).
- Track Width—A 32-bit fixed-point number that specifies the width of this track in pixels; set here to '0x05000000'.
- Track Height—A 32-bit fixed-point number that specifies the height of this track in pixels; set here to '0x02D00000'.

Movie Media, Movie Media Header, and Media Handler Reference Atoms

Figure 6 shows the structure of the 'mdia', 'mdhd', and 'hdlr' atoms:

- **mdia** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this first Movie Media atom; set here to '0x000163A4'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'mdia'.

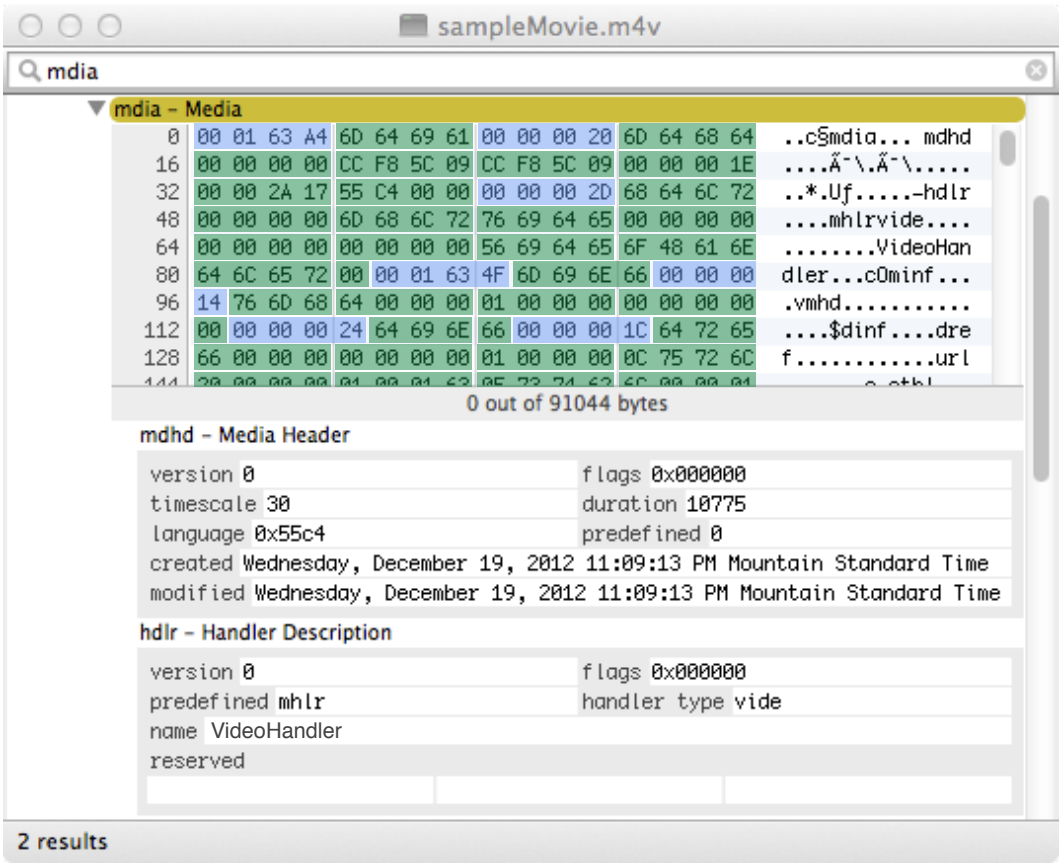

Figure 6: MP4 Movie Container 'mdia', 'mdhd', and 'hdlr' Structures.

- **mdhd** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this first Movie Media Header atom; set here to '0x00000020'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'mdhd'.
  - Version—A 1-byte specification of the version of this first Movie Track Header atom; set here to '0x00'.
  - Flags—Three bytes specification of the version of this first Movie Media Header atom; set here to 0x000000 (please refer to the *QuickTime File Format Specification* for details).
  - Creation Time—A 32-bit integer that specifies the calendar date and time (in seconds since midnight, January 1, 1904) when the movie atom was created in coordinated universal time (UTC); set here to '0xCCF85C09'.
  - Modification Time—A 32-bit integer that specifies the calendar date and time (in seconds since

midnight, January 1, 1904) when the movie atom was created in coordinated universal time (UTC); set here to '0xCCF85C09'.

- Time Scale—A time value that indicates the time scale for this media—that is, the number of time units that pass per second in its time coordinate system; set here to '0x0000001E' which represents 30 fps.
- Duration—Duration of the media in Time Scale units; set here to '0x00002A17' which represents 10,775 frames (or 323,250 seconds).
- Language— A 16-bit integer that specifies the language code for this media; set here to '0x55C4'.
- Predefined—Set here to '0x0000'.

- **hdlr** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this first Movie Media Handler Reference atom; set here to '0x0000002D'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'hdlr'.
  - Version—A 1-byte specification of the version of this first Movie Track Header atom; set here to '0x00'.
  - Flags—Three bytes specification of the version of this first Movie Media Header atom; set here to '0x000000' (please refer to the *QuickTime File Format Specification* for details).
  - Component Type—A four-character code that identifies the type of the handler (normally only two values are valid for this field: 'mhlr' for media handlers and 'dhlr' for data handlers); set here to '0x6D686C72'.
  - Component Subtype—A four-character code that identifies the type of the media handler or data handler. For media handlers, this field defines the type of data—for example, 'vide' for video data or 'soun' for sound data.; set here to '0x76696465'.
  - Component Name— A (counted) string that specifies the name of the component; set here to 'VideoHandler'.

Media Information, Media Information Header, Media Data Information, and Media Data Reference Atoms

Figure 7 shows the structure of the 'minf', 'vmhd', 'dinf', and 'dref' atoms:

- **minf** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Video Media Information atom; set here to '0x0001634F'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'minf'.
- **vmhd** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Video Media Information Header atom; set here to '0x00000014'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'vmhd'.
  - Version—A 1-byte specification of the version of this Video Media Information Header; set here to '0x00'.
  - Flags—A 3-byte space for video media information flags; set here to '0x000001'.
  - Graphics Mode—A 16-bit integer that specifies the transfer mode; set here to '0x0000'.
  - Opcolor—Three 16-bit values that specify the red, green, and blue colors for the transfer mode operation indicated in the graphics mode field; all set here to '0x0000'.
- **dinf** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Video Media Data

Information atom; set here to '0x00000024'.

- Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'dinf'.

- **dref** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Media Data Reference atom; set here to '0x0000001C'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'dref'.
  - Version—A 1-byte specification of the version of this Data Reference; set here to '0x00'.

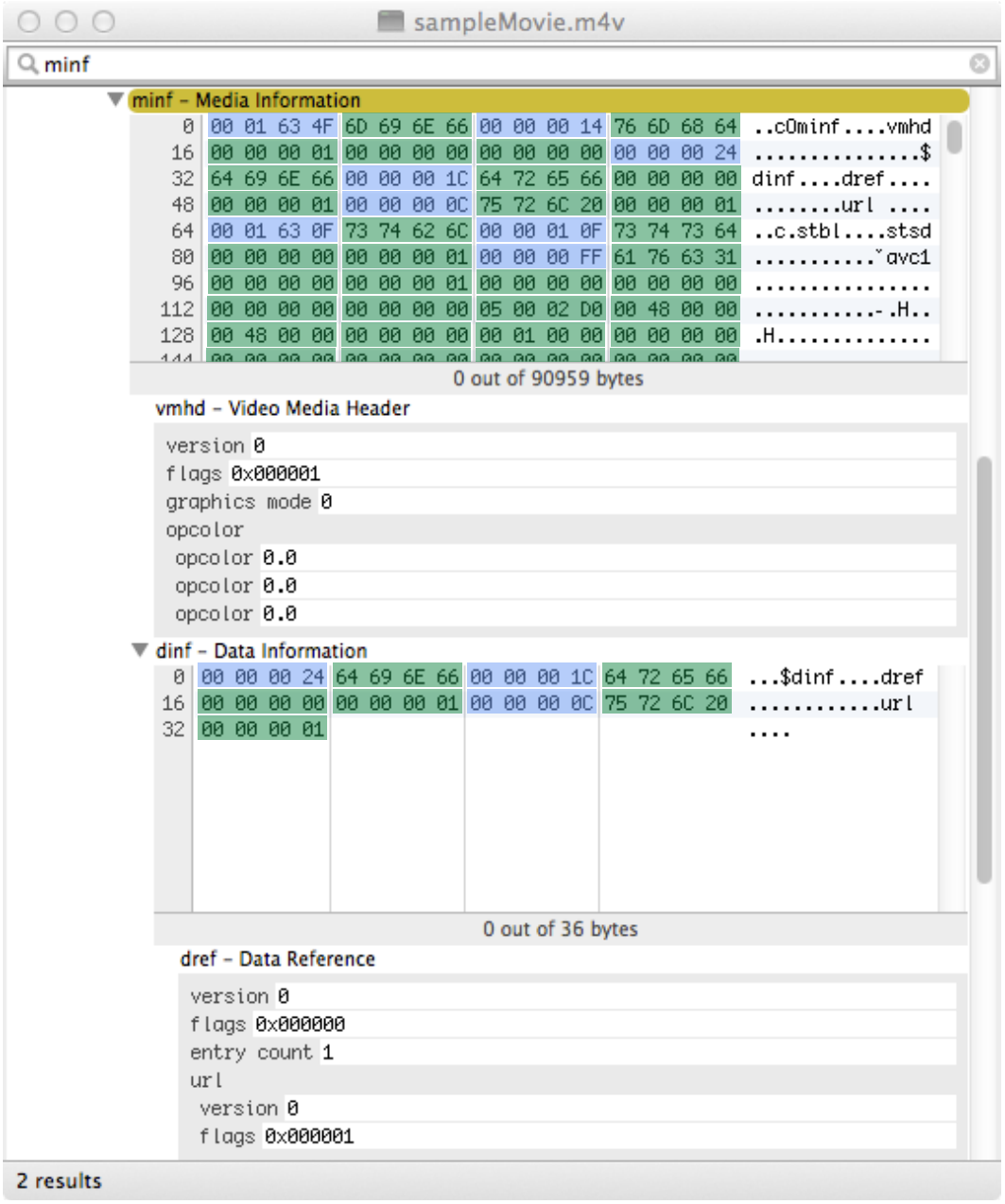

Figure 7: MP4 Movie Container 'minf', 'vmhd', 'dinf', and 'dref' Structures.

- Number of Entries—A 32-bit integer containing the count of data references that follow; set here to '0x000001'.
- Data References— An array of data references: Each data reference is formatted like an atom and contains the following data elements.
- **url** referenced data item:
- Size—A 32-bit unsigned integer that specifies the number of bytes in this Data atom; set here to '0x0000000C'.
- Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field is set to 'url '.
- Version—A 1-byte specification of the version of this Data atom; set here to '0x00'.
- Flags—A 3-byte space for Data flags; set here to '0x000001'.

Sample Table and Sample Description Atoms

Figure 8 shows the structure of the 'stbl' and 'stsd' atoms:

- **stbl** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Sample Table atom; set here to '0x0001630F'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'stbl'.
- **stsd** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Sample Description atom; set here to '0x0000010F'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'stsd'.
  - Version—A 1-byte specification of the version of this Sample Description; set here to '0x00'.
  - Flags—A 3-byte space for Data Reference flags; set here to '0x000000'.
  - Number of Entries—A 32-bit integer containing the count of data references that follow; set here to '0x000001'.
  - Sample Description Size—A 32-bit integer indicating the number of bytes in the sample description; set here to '0x0000FF'.
  - Data Format/Type— A 32-bit value indicating the format of the stored data: this depends on the media type, but is usually either the compression format or the media type; set here to 'avc1'.
  - Reserved—Six bytes that must be set to '0x00000000'.
  - Data Reference Index— A 16-bit integer that contains the index of the data reference to use to retrieve data associated with samples that use this sample description (data references are stored in data reference atoms); set here to '0x0001'.
  - Predefines—Values set here to 0.
  - Reserved—Values set here to 0.
  - Media Width—Video Width is 1280 pixels.
  - Media Height—Video Height is 720 pixels.
  - Horizontal Resolution—Horizontal Video Resolution is 72 pixels/inch.
  - Vertical Resolution—Vertical Video Resolution is 72 pixels/inch.
  - Reserved—Values set here to 0.
  - Frame Count— A 16-bit integer that indicates how many frames of compressed data are stored in each sample; set here to 1.

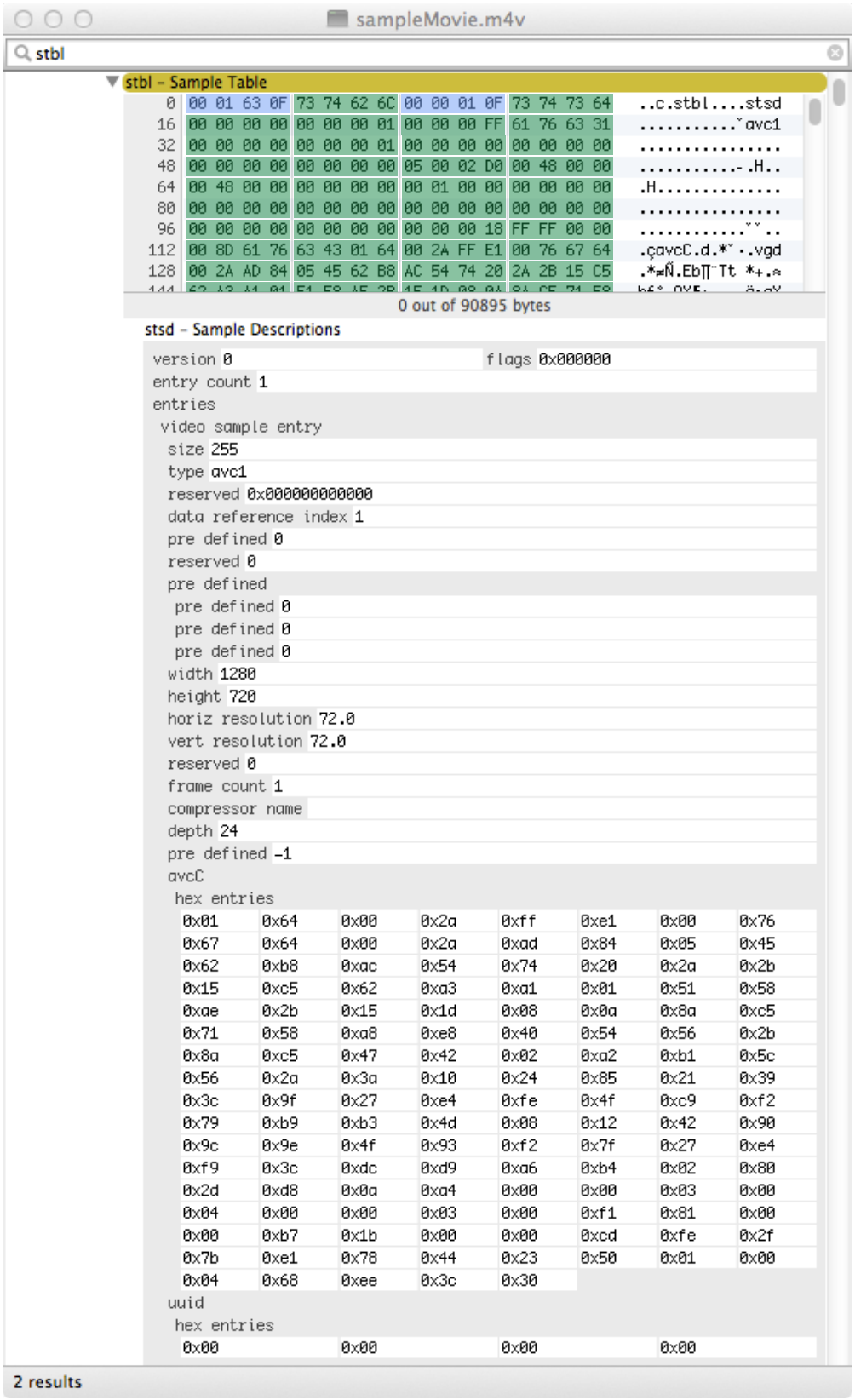

Figure 8: MP4 Movie Container 'stbl' and 'stsd' Structures.

Sample-to-Time Table and Sync Sample Atoms

Figure 9 shows the structure of the 'stts' and 'stss' atoms:

- **stts** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Sample-to Time Table atom; set here to '0x00000018'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'stts'.
  - Version—A 1-byte specification of the version of this Sample-to-Time Table; set here to '0x00'.
  - Flags—A 3-byte space for Sample-to-Time Table flags; set here to '0x000000'.
  - Number of Entries—A 32-bit integer containing the count of the Sample-to-Time Table entries that follow; set here to '1078'.
  - Sample Duration—A 32-bit integer that specifies the duration of each sample; set here to '1'.
  - Sample Count—A 32-bit integer that specifies the number of consecutive samples that have the same duration; set here to '10775'.

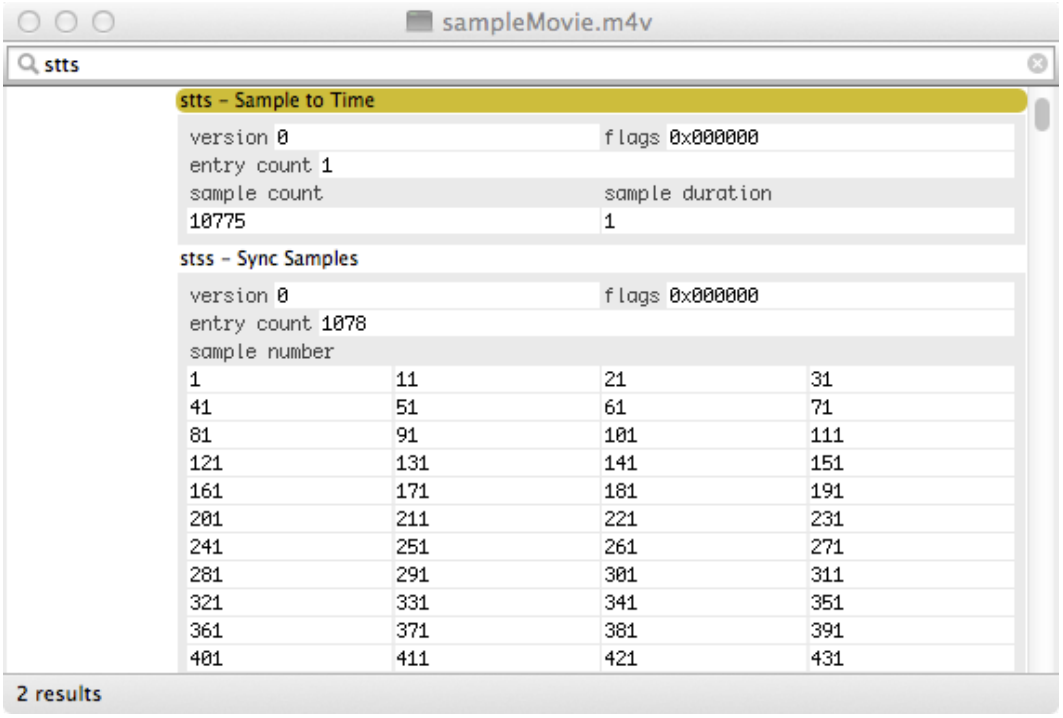

Figure 9: MP4 Movie Container 'stts' and 'stss' Structures.

- **stss** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Sync Sample atom; set here to '0x00000018'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'stts'.
  - Version—A 1-byte specification of the version of this Sync Sample atom; set here to '0x00'.
  - Flags—A 3-byte space for Sync Sample atom flags; set here to '0x000000'.
  - Number of Entries—A 32-bit integer containing the count of the Sync Sample Table entries that follow; set here to '1078'.

- Sample Duration—A 32-bit integer that specifies the duration of each sample; set here to '1'.

Sample-to-Chunk and Sample Sizes Atoms

Figure 10 shows the structure of the 'stsc' and 'stsz' atoms:

- **stsc** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Sample-to-Chuck atom; set here to '0x0000001C'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'stsc'.
  - Version—A 1-byte specification of the version of this Sample-to-Chunk atom; set here to '0x00'.
  - Flags—A 3-byte space for Sample-to-Chunk atom flags; set here to '0x000000'.
  - Number of Entries—A 32-bit integer containing the count of the Time-to-Chunk Table entries that follow; set here to '1078'.
  - Sample-to-Chunk Table—The table that maps samples to chunks. Each sample-to-chunk atom contains such a table, which identifies the chunk for each sample in a media. Each entry in the table contains a first chunk field, a samples per chunk field, and a sample description ID field. From this information, you can ascertain where samples reside in the media data.

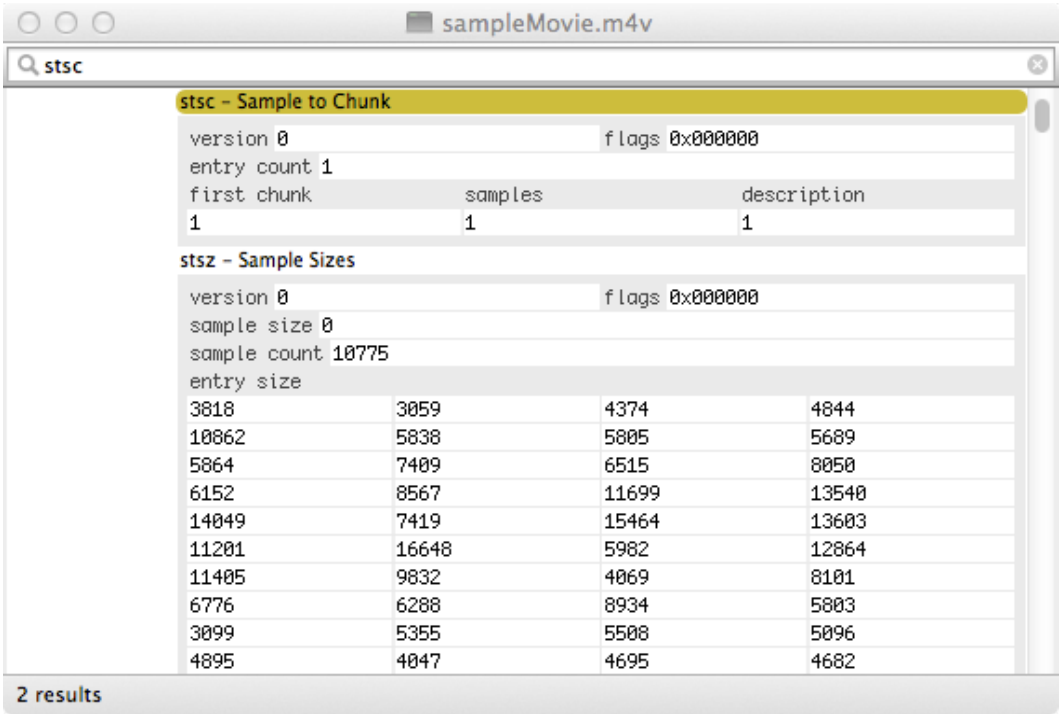

Figure 10: MP4 Movie Container 'stsc' and 'stsz' Structures.

- **stsz** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Sample Size atom; set here to '0x0000A870'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'stsz'.
  - Version—A 1-byte specification of the version of this Sample Sizes atom; set here to '0x00'.
  - Flags—A 3-byte space for Sample Sizes atom flags; set here to '0x000000'.

- Sample Size— A 32-bit integer specifying the sample size: if all the samples are the same size, this field contains that size value. If this field is set to 0, then the samples have different sizes, and those sizes are stored in the sample size table.
- Number of Entries—A 32-bit integer containing the count of the Time-to-Chunk Table entries that follow; set here to '10775'.
- Sample-to-Chunk Table—The table that maps samples to chunks. Each Sample-to-Chunk atom contains such a table, which identifies the chunk for each sample in a media. Each entry in the table contains a first chunk field, a samples per chunk field, and a sample description ID field. From this information, you can ascertain where samples reside in the media data.

Chunk Offset Atom

Figure 11 shows the structure of the 'stco' atom:

- **stco** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this Sample Offset Table atom; set here to '0x0000A86C'.
  - Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'stsc'.
  - Version—A 1-byte specification of the version of this Sample Offset Table atom; set here to '0x00'.
  - Flags—A 3-byte space for Sample Offset Table atom flags; set here to '0x000000'.
  - Number of Entries—A 32-bit integer containing the count of the Sample Offset Table entries that follow; set here to '10775'.

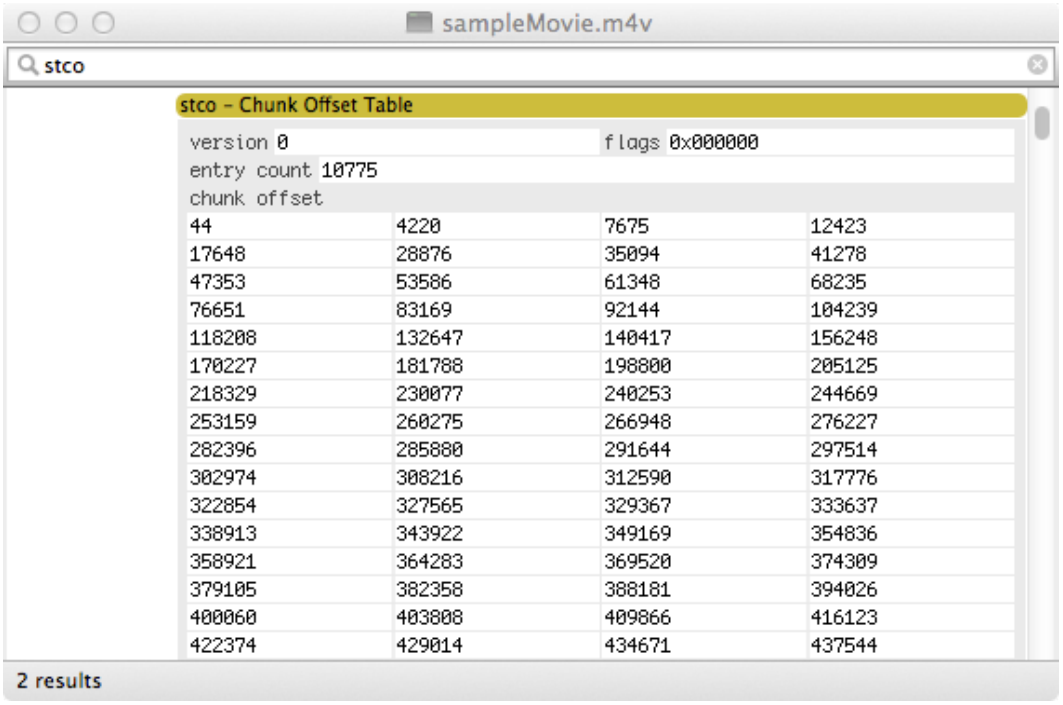

Figure 11: MP4 Movie Container 'stco' Structure.

User Data Atom

Figure 12 shows the structure of the 'udta' atom:

- **udta** contains the following fields:
  - Size—A 32-bit unsigned integer that specifies the number of bytes in this User Data atom; set here

to '0x00000060'.

- Type—A 32-bit unsigned integer that identifies the type, represented as a four-character code; this field must be set to 'udta'.
- Version—A 1-byte specification of the version of this Sample Offset Table atom; set here to '0x00'.
- Flags—A 3-byte space for Sample Offset Table atom flags; set here to '0x000000'.
- User Data List—A user data list that is formatted as a series of atoms. Each data element in the user data list contains size and type information along with its data. For historical reasons, the data list is optionally terminated by a 32-bit integer set to 0. If you are writing a program to read user data atoms, you should allow for the terminating 0. However, if you are writing a program to create user data atoms, you can safely leave out the trailing 0.

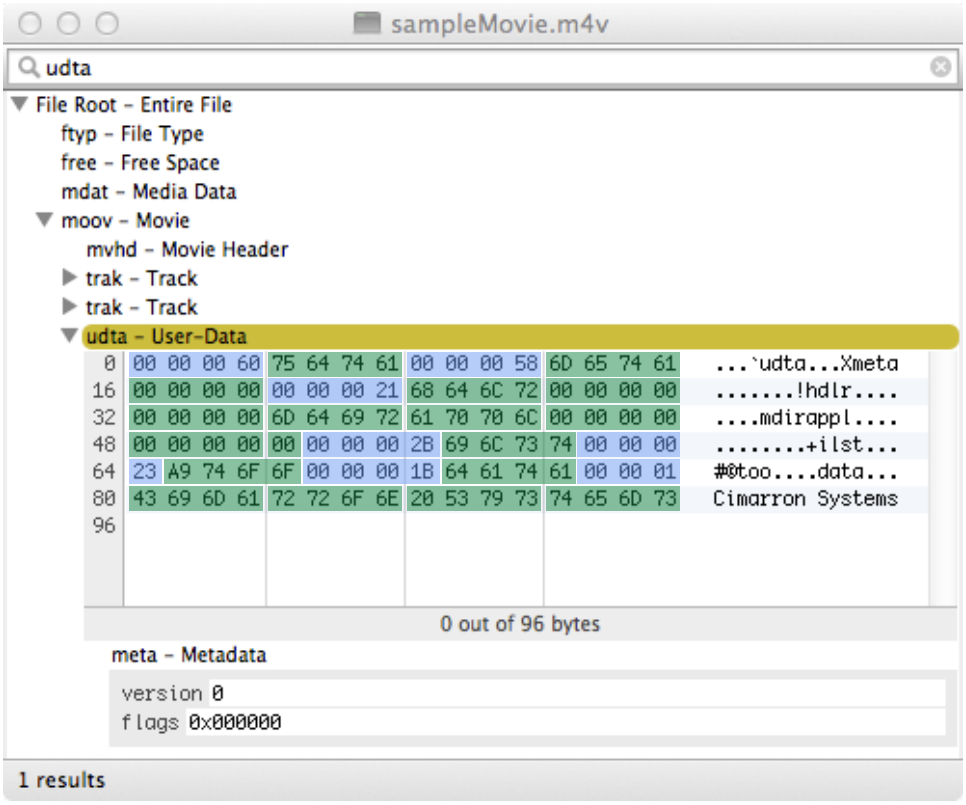

Figure 12: MP4 Movie Container 'udta' Structure.

In this Application Note 101, we have presented an overview of the elements of a canonically constructed MP4 Movie Container encapsulate H.264 Video and AAC-LC Audio. Although we have not presented the MP4 atoms used to encapsulate the AAC-LC Audio, the hierarchy and structure of these atoms are very similar to the video atoms.

Additional Resources

In addition to those already described, listed below are a number of resources that may be helpful:

1. QuickTime File Format
2. M4V File Format
3. MPEG-4 Part 14

For more information regarding this and other Cimarron Systems products, please contact us using the

contact information below.

Contact Information:

Cimarron Systems, LLC
Evergreen, Colorado
telephone: (303) 674-9207
email: info@cimarronsystems.com
www.cimarronsystems.com

Revision History:

Date	Version	Notes
10/15/2012	version 1.0	Initial version.
4/28/2014	version 2.0	Minor editorial updates.

