HD Radio™ Air Interface Design Description - Program Service Data

Rev. C

March 31, 2005

Doc. No. SY_IDD_1028s

TRADEMARKS

iBiquity Digital Corporation
8865 Stanford Boulevard
Suite 202
Columbia, MD 21045
410–872–1530 (Phone)
410–872–1531 (FAX)
info@ibiquity.com

Table of Contents

Contents

List of Figures

List of Tables

1 Scope

1.1 System Overview

The iBiquity Digital Corporation HD Radio™ system is designed to permit a smooth evolution from current analog amplitude modulation (AM) and frequency modulation (FM) radio to a fully digital in-band on-channel (IBOC) system. This system delivers digital audio and data services to mobile, portable, and fixed receivers from terrestrial transmitters in the existing medium frequency (MF) and very high frequency (VHF) radio bands. Broadcasters may continue to transmit analog AM and FM simultaneously with the new, higher-quality and more robust digital signals, allowing themselves and their listeners to convert from analog to digital radio while maintaining their current frequency allocations.

1.2 Document Overview

This document provides a description of the Program Service Data capability. Program Service Data refers to both Main Program Service Data (MPSD) and Supplemental Program Service Data (SPSD). A detailed description of ID3v2 message encoding is explained in the Appendices. Appendix A contains the ID3 Standard Reference specification. Appendix A-1 contains the ID3 standard as implemented in the HD Radio system and the relevant sections of the specifications.

2 Referenced Documents

[1] iBiquity Digital Corporation, “HD Radio™ Air Interface Design Description – Program Service Data Transport,” Doc. No. SY_IDD_1085s, Revision C.

[2] iBiquity Digital Corporation, “HD Radio™ Air Interface Design Description – Audio Transport,” Doc. No. SY_IDD_1017s, Revision E.

[3] Martin Nilsson, "ID3v2.3.0 Informal standard," URL: http://www.id3.org.

3 Abbreviations and Acronyms

3.1 Abbreviations and Acronyms

AAS	Advanced Application Services
AM	Amplitude Modulation
API	Application Programming Interface
CODEC	Coder Decoder
COMM	Comment (ID3 Frame ID)
COMR	Commercial (ID3 Frame ID)
FM	Frequency Modulation
IBOC	In-Band On-Channel
ID	Identification
MF	Medium Frequency
MPSA	Main Program Service Audio
MPS	Main Program Service
MPSD	Main Program Service Data
PSD	Program Service Data
REFID	Reference Identifier
HD RLS	Radio Link Subsystem
SIS	Station Information Service
SPS	Supplementary Program Service
SPSA	Supplementary Program Service Audio
SPSD	Supplementary Program Service Data
TALB	Album (ID3 Frame ID)
TCON	Genre (ID3 Frame ID)
TIT2	Title (ID3 Frame ID)
TPE1	Artist (ID3 Frame ID)
UFID	Reference Identifier (ID3 Frame ID)
URL	Uniform Resource Locator
VHF	Very High Frequency

3.2 Presentation Conventions

Unless otherwise noted, the following conventions apply to this document:

- All vectors are indexed starting with 0.
- The element of a vector with the lowest index is considered to be first.
- In drawings and tables, the leftmost bit is considered to occur first in time.
- Bit 0 of a byte or word is considered the least significant bit.
- When presenting the dimensions of a matrix, the number of rows is given first (e.g., an n x m matrix has n rows and m columns).
- In timing diagrams, earliest time is on the left.
- Binary numbers are presented with the most significant bit having the highest index.
- In representations of binary numbers, the least significant bit is on the right.

4 Overview

The Main Program Service allows the transmission of existing analog radio-programming in both analog and digital formats. MPS includes the Main Program Service Audio (MPSA) and the Main Program Service Data (MPSD). Similarly, the Supplemental Program Service Data includes the Supplemental Program Service Audio (SPSA) and the Supplemental Program Service Data (SPSD). Both MPSD and SPSD are in general referred to as Program Service Data (PSD). PSD provides additional information about the audio. The processing of the Supplemental Program Service Data (SPS Data) is exactly the same as the MPS Data.

The main program audio and data are synchronized at the broadcast studio. That is, the PSD is transmitted so that radio receivers can acquire it at the same time the audio program that it describes is being heard by radio listeners.

4.1 Program Service Data Flow

Figure 4-1 illustrates the PSD flow through the HD Radio Broadcast System. The PSD is sent through the Service Interface as ID3 tags which are transmitted as data packets. The PSD is processed by the HD RLS mechanism within the PSD Transport [1] for framing and encapsulation as byte streams and then inserted into the Audio streams by the Audio Transport as PSD PDUs (MPSD/SPSD PDUs). The Audio Transport [2] then multiplexes the PSD PDUs with the audio streams to generate MPS/SPS PDUs.

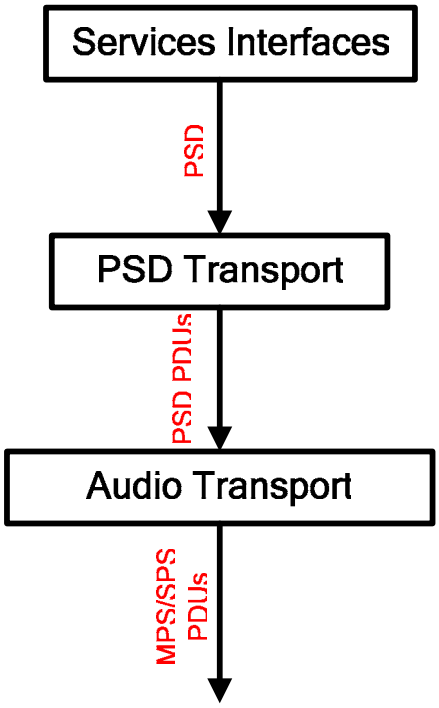

Figure 4-1 PSD Flow

5 Program Service Data Description

Program Service Data is transmitted along with the program audio. PSD is intended to describe or complement the audio program heard by the radio listener.

The following subsections provide:

- An introduction to the basic PSD content
- A description of broadcast PSD processing
- The format of PSD messages

5.1 PSD Content

PSD consists of a general set of categories that describe the various programming content, such as a song, talk show, advertisement, or announcement. For example, the *Title* field can be used to describe the name of a song, topic of a talk show, advertisement, or announcement.

The PSD fields include the following:

- Title
- Artist
- Album
- Genre
- Comment
- Commercial
- Reference Identifiers

A detailed description of the PSD structure is discussed in Subsection 5.3.

5.2 PSD Transmit Processing

Program Service Data can originate from a studio automation system or any other computing resource where program audio originates. Regardless of the source, the processing and interface to facilitate broadcast of PSD is consistent. Program Service Data providers input the desired content (e.g., artist, title, etc.) and transfer the resulting PSD message to the Service Interface. The PSD data packets then undergoes framing and encapsulation by the HD RLS subsystem as byte streams within the PSD Transport before being sent as PSD PDUs to the Audio transport where it is multiplexed with the audio streams.

Some key requirements for Program Service Data are:

- Program audio and associated data shall be transmitted synchronously, so receivers can acquire correlated audio and data at the same time.
- PSD messages shall not exceed 1024 bytes.

Some key considerations for Program Service Data are:

- PSD messages are continuously transmitted with the most recent message transmitted repeatedly.
- PSD providers send a new PSD message when the PSD content has changed.

5.3 PSD Message Format

Program Service Data is formatted using a subset of the standard called ID3v2 (see reference [3]). Historically, ID3 has been used to allow textual information, such as artist, title, and genre information to

co-exist within MPEG-3 (MP3) audio files. The HD Radio system uses ID3 to deliver Program Service Data along with real-time broadcast audio.

HD Radio system implements a specific subset of the ID3v2 parameters. The ID3v2 general structure is as follows:

- The complete ID3 message is called an *ID3 tag*.
- ID3 tags contain one or more content types referred to as *frames.* Frames contain individual pieces of information (e.g., song artist, title, etc.). Each *frame* has a four-character identifier. For example, the commercial frame is identified as *COMR.*
- Within *frames*, sub-elements called *fields* can exist. Fields further categorize the information within a frame. For example, the commercial frame has a field to specify *sale price.*

Figure 5-1 shows the general ID3 message structure.

PSD utilizes only the identified subset of the defined ID3v2 standard. Broadcasters providing PSD must, at a minimum, transmit the *Title* and *Artist* information. The Artist field is used to transmit the name of the artist of the current song. It is limited to 128 characters. Typical message lengths are less than 30 characters. ***Table 5-1*** gives a description of ID3 frames used for PSD. For a detailed description of ID3v2 message encoding used for PSD, see Appendix A-1.

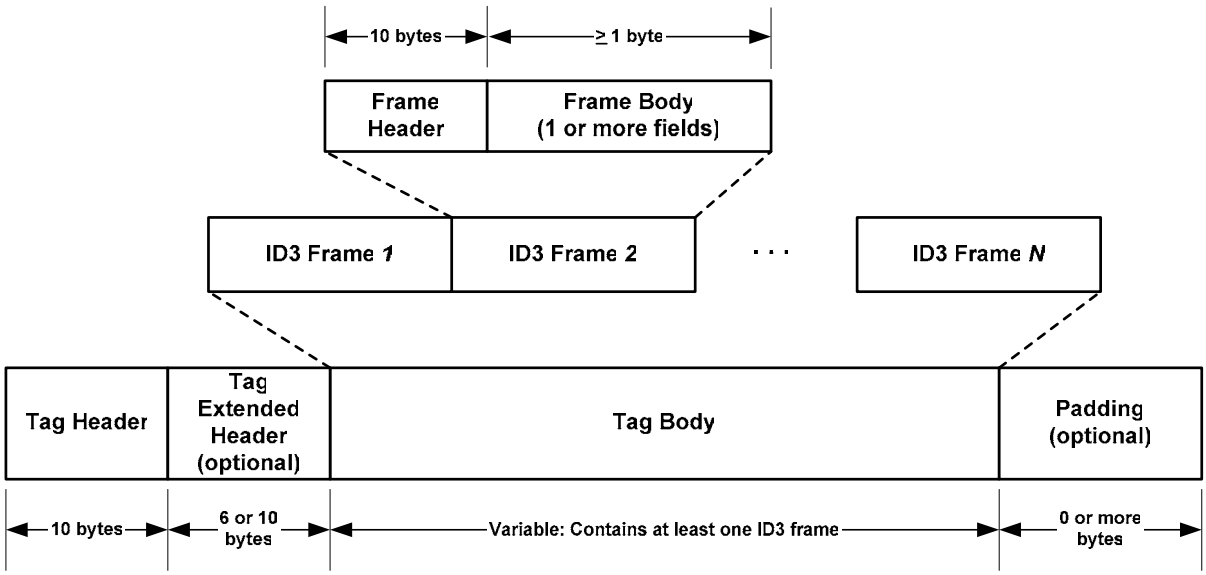

Figure 5-1 ID3 Message Structure

Appendix A contains the ID3 Standard Reference specification. Appendix A-1 contains the ID3 standard as implemented in the HD Radio system and the relevant sections of the specifications.

Table 5-1 ID3 Frames Supported by PSD

	PSD	ID3	ID3	Description	Type		
	Attribute	Frame ID	Field		Music	Talk	Announcement
1.	Title	TIT2	Info	One-line Title Name	Song title	Talk Topic	Announcement or Advertisement Title
2.	Artist	TPE1	Info	Performer, Originator, Author, Sponsor	Artist Name	Show Host	Author/Sponsor
3.	Album	TALB	Info	Content Source	Album Name	Show Name	Sponsor Name
4.	Genre	TCON	Info	Categorization of content. This is an enumerated field of predefined types.	e.g., (8) Jazz, (17) Rock, (32) Classical	(101) Speech	(101) Speech
5.	Comment	COMM	short description field	One-line Title for Comment Description	Comment Title	Comment Title	Comment Title
			Content field	Comment Description. Detailed explanation, user callback information or further information.	web site, contact or other info	Talk Show call-in number, or other show info	Announcement or Advertising statement, Point-of-sale or more info
6.	Commercial	COMR	Price	Price of merchandise	The commercial frame facilitates sale of products and services. (Note, the binary picture is not required.)		
			Valid Until	Expiration data for transaction			
			Contact URL	URL identifier used to contact the seller. Can be used to initiate purchase transaction via an external return channel, such as a cellular phone network.			
			Received as	Method in which merchandise is received (e.g., over the internet)			
			Name of seller	Text identifying seller			
			Description	Textual description of advertisement			
			Picture	Picture of advertised item			
			Seller logo	Binary graphic of seller logo			
7.	Reference Identifier	UFID	Owner Identifier	This ID is used to: 1. Broadcast a single MPS Data message as one or more separate MPS Data messages. 2. Provide additional MPS Data identification and cross-referencing for future enhancements.	If Owner Identifier is set to PADLINK, then the identifier contains a unique message identifier, which allows a single PSD message to be broken into one or more PSD messages for broadcast transmission. Each individual PSD message is a complete decodable message, but may only contain a subset of the PSD content. For example, the title, artist and album may be transmitted in one PSD message and the commercial information may be transmitted separately. Both messages contain a common unique identifier, which is the mechanism for correlating the complete PSD message. Collectively the two messages represent a unified PSD message. The combined messages should be treated as a whole, when performing receiver-side processing. The identifier field contains a unique number ranging from 0 to 65,535. This number is unique only to the Main Program Service/SPS instance of a given station.		
			Identifier	Unique Identifier			

HD Radio™ Air Interface Design Description - Program Service Data Appendix A – ID3 Standard Reference

Informal standard
Document: id3v2.3.0.txt
M. Nilsson
3rd February 1999

ID3 tag version 2.3.0

Status of this document

This document is an informal standard and replaces the ID3v2.2.0 standard [ID3v2]. The informal standard is released so that implementors could have a set standard before a formal standard is set. The formal standard will use another version or revision number if not identical to what is described in this document. The contents in this document may change for clarifications but never for added or altered functionallity.

Distribution of this document is unlimited.

Abstract

This document describes the ID3v2.3.0, which is a more developed version of the ID3v2 informal standard [ID3v2] (version 2.2.0), evolved from the ID3 tagging system. The ID3v2 offers a flexible way of storing information about an audio file within itself to determine its origin and contents. The information may be technical information, such as equalisation curves, as well as related meta information, such as title, performer, copyright etc.

1. Table of contents

2. Conventions in this document

In the examples, text within "" is a text string exactly as it appears in a file. Numbers preceded with $ are hexadecimal and numbers preceded with % are binary. $xx is used to indicate a byte with unknown content. %x is used to indicate a bit with unknown content. The most significant bit (MSB) of a byte is called 'bit 7' and the least significant bit (LSB) is called 'bit 0'.

A tag is the whole tag described in this document. A frame is a block of information in the tag. The tag consists of a header, frames and optional padding. A field is a piece of information; one value, a string etc. A numeric string is a string that consists of the characters 0-9 only.

3. ID3v2 overview

The two biggest design goals were to be able to implement ID3v2 without disturbing old software too much and that ID3v2 should be as flexible and expandable as possible.

The first criterion is met by the simple fact that the MPEG [MPEG] decoding software uses a syncsignal, embedded in the audiostream, to 'lock on to' the audio. Since the ID3v2 tag doesn't contain a valid syncsignal, no software will attempt to play the tag. If, for any reason, coincidence make a syncsignal appear within the tag it will be taken care of by the 'unsynchronisation scheme' described in section 5.

The second criterion has made a more noticeable impact on the design

of the ID3v2 tag. It is constructed as a container for several information blocks, called frames, whose format need not be known to the software that encounters them. At the start of every frame there is an identifier that explains the frames' format and content, and a size descriptor that allows software to skip unknown frames.

If a total revision of the ID3v2 tag should be needed, there is a version number and a size descriptor in the ID3v2 header.

The ID3 tag described in this document is mainly targeted at files encoded with MPEG-1/2 layer I, MPEG-1/2 layer II, MPEG-1/2 layer III and MPEG-2.5, but may work with other types of encoded audio.

The bitorder in ID3v2 is most significant bit first (MSB). The byteorder in multibyte numbers is most significant byte first (e.g. $12345678 would be encoded $12 34 56 78).

It is permitted to include padding after all the final frame (at the end of the ID3 tag), making the size of all the frames together smaller than the size given in the head of the tag. A possible purpose of this padding is to allow for adding a few additional frames or enlarge existing frames within the tag without having to rewrite the entire file. The value of the padding bytes must be $00.

3.1. ID3v2 header

The ID3v2 tag header, which should be the first information in the file, is 10 bytes as follows:

```
ID3v2/file identifier      "ID3"
ID3v2 version              $03 00
ID3v2 flags                %abc00000
ID3v2 size             4 * %0xxxxxxx
```

The first three bytes of the tag are always "ID3" to indicate that this is an ID3v2 tag, directly followed by the two version bytes. The first byte of ID3v2 version is it's major version, while the second byte is its revision number. In this case this is ID3v2.3.0. All revisions are backwards compatible while major versions are not. If software with ID3v2.2.0 and below support should encounter version three or higher it should simply ignore the whole tag. Version and revision will never be $FF.

The version is followed by one the ID3v2 flags field, of which currently only three flags are used.

a - Unsynchronisation

Bit 7 in the 'ID3v2 flags' indicates whether or not unsynchronisation is used (see section 5 for details); a set bit indicates usage.

b - Extended header

The second bit (bit 6) indicates whether or not the header is followed by an extended header. The extended header is described in section 3.2.

c - Experimental indicator

The third bit (bit 5) should be used as an 'experimental indicator'. This flag should always be set when the tag is in an experimental stage.

All the other flags should be cleared. If one of these undefined flags are set that might mean that the tag is not readable for a parser that does not know the flags function.

The ID3v2 tag size is encoded with four bytes where the most significant bit (bit 7) is set to zero in every byte, making a total of 28 bits. The zeroed bits are ignored, so a 257 bytes long tag is represented as $00 00 02 01.

The ID3v2 tag size is the size of the complete tag after unsychronisation, including padding, excluding the header but not excluding the extended header (total tag size - 10). Only 28 bits (representing up to 256MB) are used in the size description to avoid the introducuction of 'false syncsignals'.

An ID3v2 tag can be detected with the following pattern:
$49 44 33 yy yy xx zz zz zz zz
Where yy is less than $FF, xx is the 'flags' byte and zz is less than $80.

3.2. ID3v2 extended header

The extended header contains information that is not vital to the correct parsing of the tag information, hence the extended header is optional.

Extended header size	$xx xx xx xx
Extended Flags	$xx xx
Size of padding	$xx xx xx xx

Where the 'Extended header size', currently 6 or 10 bytes, excludes itself. The 'Size of padding' is simply the total tag size excluding the frames and the headers, in other words the padding. The extended header is considered separate from the header proper, and as such is subject to unsynchronisation.

The extended flags are a secondary flag set which describes further attributes of the tag. These attributes are currently defined as follows

%x0000000 00000000

x - CRC data present

If this flag is set four bytes of CRC-32 data is appended to the extended header. The CRC should be calculated before unsynchronisation on the data between the extended header and the padding, i.e. the frames and only the frames.

Total frame CRC	$xx xx xx xx

3.3. ID3v2 frame overview

As the tag consists of a tag header and a tag body with one or more frames, all the frames consists of a frame header followed by one or more fields containing the actual information. The layout of the frame header:

Frame ID	$xx xx xx xx	(four characters)
Size	$xx xx xx xx	
Flags	$xx xx	

The frame ID made out of the characters capital A-Z and 0-9. Identifiers beginning with "X", "Y" and "Z" are for experimental use and free for everyone to use, without the need to set the experimental bit in the tag header. Have in mind that someone else might have used the same identifier as you. All other identifiers are either used or reserved for future use.

The frame ID is followed by a size descriptor, making a total header size of ten bytes in every frame. The size is calculated as frame size excluding frame header (frame size - 10).

In the frame header the size descriptor is followed by two flags bytes. These flags are described in section 3.3.1.

There is no fixed order of the frames' appearance in the tag, although it is desired that the frames are arranged in order of significance concerning the recognition of the file. An example of such order: UFID, TIT2, MCDI, TRCK ...

A tag must contain at least one frame. A frame must be at least 1 byte big, excluding the header.

If nothing else is said a string is represented as ISO-8859-1 [ISO-8859-1] characters in the range $20 - $FF. Such strings are represented as <text string>, or <full text string> if newlines are allowed, in the frame descriptions. All Unicode strings [UNICODE] use 16-bit unicode 2.0 (ISO/IEC 10646-1:1993, UCS-2). Unicode strings must begin with the Unicode BOM ($FF FE or $FE FF) to identify the byte order.

All numeric strings and URLs [URL] are always encoded as ISO-8859-1. Terminated strings are terminated with $00 if encoded with ISO-8859-1 and $00 00 if encoded as unicode. If nothing else is said newline character is forbidden. In ISO-8859-1 a new line is represented, when allowed, with $0A only. Frames that allow different types of text encoding have a text encoding description byte directly after the frame size. If ISO-8859-1 is used this byte should be $00, if Unicode is used it should be $01. Strings dependent on encoding is

represented as <text string according to encoding>, or <full text string according to encoding> if newlines are allowed. Any empty Unicode strings which are NULL-terminated may have the Unicode BOM followed by a Unicode NULL ($FF FE 00 00 or $FE FF 00 00).

The three byte language field is used to describe the language of the frame's content, according to ISO-639-2 [ISO-639-2].

All URLs [URL] may be relative, e.g. "picture.png", "../doc.txt".

If a frame is longer than it should be, e.g. having more fields than specified in this document, that indicates that additions to the frame have been made in a later version of the ID3v2 standard. This is reflected by the revision number in the header of the tag.

3.3.1. Frame header flags

In the frame header the size descriptor is followed by two flags bytes. All unused flags must be cleared. The first byte is for 'status messages' and the second byte is for encoding purposes. If an unknown flag is set in the first byte the frame may not be changed without the bit cleared. If an unknown flag is set in the second byte it is likely to not be readable. The flags field is defined as follows.

```
%abc00000 %ijk00000
```

a - Tag alter preservation

This flag tells the software what to do with this frame if it is unknown and the tag is altered in any way. This applies to all kinds of alterations, including adding more padding and reordering the frames.

0 Frame should be preserved.
1 Frame should be discarded.

b - File alter preservation

This flag tells the software what to do with this frame if it is unknown and the file, excluding the tag, is altered. This does not apply when the audio is completely replaced with other audio data.

0 Frame should be preserved.
1 Frame should be discarded.

c - Read only

This flag, if set, tells the software that the contents of this frame is intended to be read only. Changing the contents might break something, e.g. a signature. If the contents are changed, without knowledge in why the frame was flagged read only and without taking the proper means to compensate, e.g. recalculating

the signature, the bit should be cleared.

i - Compression

This flag indicates whether or not the frame is compressed.

0 Frame is not compressed.
1 Frame is compressed using zlib [zlib] with 4 bytes for 'decompressed size' appended to the frame header.

j - Encryption

This flag indicates wether or not the frame is enrypted. If set one byte indicating with which method it was encrypted will be appended to the frame header. See section 4.26. for more information about encryption method registration.

0 Frame is not encrypted.
1 Frame is encrypted.

k - Grouping identity

This flag indicates whether or not this frame belongs in a group with other frames. If set a group identifier byte is added to the frame header. Every frame with the same group identifier belongs to the same group.

0 Frame does not contain group information
1 Frame contains group information

Some flags indicates that the frame header is extended with additional information. This information will be added to the frame header in the same order as the flags indicating the additions. I.e. the four bytes of decompressed size will preceed the encryption method byte. These additions to the frame header, while not included in the frame header size but are included in the 'frame size' field, are not subject to encryption or compression.

3.3.2. Default flags

The default settings for the frames described in this document can be divided into the following classes. The flags may be set differently if found more suitable by the software.

1. Discarded if tag is altered, discarded if file is altered.

   None.

2. Discarded if tag is altered, preserved if file is altered.

   None.

3. Preserved if tag is altered, discarded if file is altered.

   AENC, ETCO, EQUA, MLLT, POSS, SYLT, SYTC, RVAD, TENC, TLEN, TSIZ

4. Preserved if tag is altered, preserved if file is altered.

   The rest of the frames.

4. Declared ID3v2 frames

The following frames are declared in this draft.

4.21 AENC Audio encryption
4.15 APIC Attached picture

4.11 COMM Comments
4.25 COMR Commercial frame

4.26 ENCR Encryption method registration
4.13 EQUA Equalization
4.6 ETCO Event timing codes

4.16 GEOB General encapsulated object
4.27 GRID Group identification registration

4.4 IPLS Involved people list

4.21 LINK Linked information

4.5 MCDI Music CD identifier
4.7 MLLT MPEG location lookup table

4.24 OWNE Ownership frame

4.28. PRIV Private frame
4.17 PCNT Play counter
4.18 POPM Popularimeter
4.22 POSS Position synchronisation frame

4.19 RBUF Recommended buffer size
4.12 RVAD Relative volume adjustment
4.14 RVRB Reverb

4.10 SYLT Synchronized lyric/text
4.8 SYTC Synchronized tempo codes

4.2.1 TALB Album/Movie/Show title
4.2.1 TBPM BPM (beats per minute)
4.2.1 TCOM Composer
4.2.1 TCON Content type
4.2.1 TCOP Copyright message
4.2.1 TDAT Date
4.2.1 TDLY Playlist delay
4.2.1 TENC Encoded by
4.2.1 TEXT Lyricist/Text writer
4.2.1 TFLT File type

4.2.1 TIME Time
4.2.1 TIT1 Content group description
4.2.1 TIT2 Title/songname/content description
4.2.1 TIT3 Subtitle/Description refinement
4.2.1 TKEY Initial key
4.2.1 TLAN Language(s)
4.2.1 TLEN Length
4.2.1 TMED Media type
4.2.1 TOAL Original album/movie/show title
4.2.1 TOFN Original filename
4.2.1 TOLY Original lyricist(s)/text writer(s)
4.2.1 TOPE Original artist(s)/performer(s)
4.2.1 TORY Original release year
4.2.1 TOWN File owner/licensee
4.2.1 TPE1 Lead performer(s)/Soloist(s)
4.2.1 TPE2 Band/orchestra/accompaniment
4.2.1 TPE3 Conductor/performer refinement
4.2.1 TPE4 Interpreted, remixed, or otherwise modified by
4.2.1 TPOS Part of a set
4.2.1 TPUB Publisher
4.2.1 TRCK Track number/Position in set
4.2.1 TRDA Recording dates
4.2.1 TRSN Internet radio station name
4.2.1 TRSO Internet radio station owner
4.2.1 TSIZ Size
4.2.1 TSRC ISRC (international standard recording code)
4.2.1 TSSE Software/Hardware and settings used for encoding
4.2.1 TYER Year
4.2.2 TXXX User defined text information frame

4.1 UFID Unique file identifier
4.23 USER Terms of use
4.9 USLT Unsychronized lyric/text transcription

4.3.1 WCOM Commercial information
4.3.1 WCOP Copyright/Legal information
4.3.1 WOAF Official audio file webpage
4.3.1 WOAR Official artist/performer webpage
4.3.1 WOAS Official audio source webpage
4.3.1 WORS Official internet radio station homepage
4.3.1 WPAY Payment
4.3.1 WPUB Publishers official webpage
4.3.2 WXXX User defined URL link frame

4.1. Unique file identifier

This frame's purpose is to be able to identify the audio file in a database that may contain more information relevant to the content. Since standardisation of such a database is beyond this document, all frames begin with a null-terminated string with a URL [URL] containing an email address, or a link to a location where an email address can be found, that belongs to the organisation responsible for this specific database implementation. Questions regarding the database should be sent to the indicated email address. The URL should not be used for the actual database queries. The string "http://www.id3.org/dummy/ufid.html" should be used for tests.

Software that isn't told otherwise may safely remove such frames. The 'Owner identifier' must be non-empty (more than just a termination). The 'Owner identifier' is then followed by the actual identifier, which may be up to 64 bytes. There may be more than one "UFID" frame in a tag, but only one with the same 'Owner identifier'.

```
<Header for 'Unique file identifier', ID: "UFID">
Owner identifier        <text string> $00
Identifier              <up to 64 bytes binary data>
```

4.2. Text information frames

The text information frames are the most important frames, containing information like artist, album and more. There may only be one text information frame of its kind in an tag. If the textstring is followed by a termination ($00 (00)) all the following information should be ignored and not be displayed. All text frame identifiers begin with "T". Only text frame identifiers begin with "T", with the exception of the "TXXX" frame. All the text information frames have the following format:

```
<Header for 'Text information frame', ID: "T000" - "TZZZ",
excluding "TXXX" described in 4.2.2.>
Text encoding                $xx
Information                  <text string according to encoding>
```

4.2.1. Text information frames - details

TALB
The 'Album/Movie/Show title' frame is intended for the title of the recording(/source of sound) which the audio in the file is taken from.

TBPM
The 'BPM' frame contains the number of beats per minute in the mainpart of the audio. The BPM is an integer and represented as a numerical string.

TCOM
The 'Composer(s)' frame is intended for the name of the composer(s). They are seperated with the "/" character.

TCON
The 'Content type', which previously was stored as a one byte numeric value only, is now a numeric string. You may use one or several of the types as ID3v1.1 did or, since the category list would be impossible to maintain with accurate and up to date categories, define your own.

References to the ID3v1 genres can be made by, as first byte, enter "(" followed by a number from the genres list (appendix A.) and ended with a ")" character. This is optionally followed by a refinement, e.g. "(21)" or "(4)Eurodisco". Several references can be made in the same frame, e.g. "(51)(39)". If the refinement should begin with a "(" character it should be replaced with "((", e.g. "((I

can figure out any genre)" or "(55)((I think...)". The following new content types is defined in ID3v2 and is implemented in the same way as the numerig content types, e.g. "(RX)".

```
  RX  Remix
  CR  Cover
```

TCOP
The 'Copyright message' frame, which must begin with a year and a space character (making five characters), is intended for the copyright holder of the original sound, not the audio file itself. The absence of this frame means only that the copyright information is unavailable or has been removed, and must not be interpreted to mean that the sound is public domain. Every time this field is displayed the field must be preceded with "Copyright " (C) " ", where (C) is one character showing a C in a circle.

TDAT
The 'Date' frame is a numeric string in the DDMM format containing the date for the recording. This field is always four characters long.

TDLY
The 'Playlist delay' defines the numbers of milliseconds of silence between every song in a playlist. The player should use the "ETC" frame, if present, to skip initial silence and silence at the end of the audio to match the 'Playlist delay' time. The time is represented as a numeric string.

TENC
The 'Encoded by' frame contains the name of the person or organisation that encoded the audio file. This field may contain a copyright message, if the audio file also is copyrighted by the encoder.

TEXT
The 'Lyricist(s)/Text writer(s)' frame is intended for the writer(s) of the text or lyrics in the recording. They are seperated with the "/" character.

TFLT
The 'File type' frame indicates which type of audio this tag defines. The following type and refinements are defined:

```
  MPG    MPEG Audio
    /1     MPEG 1/2 layer I
    /2     MPEG 1/2 layer II
    /3     MPEG 1/2 layer III
    /2.5   MPEG 2.5
    /AAC   Advanced audio compression
  VQF    Transform-domain Weighted Interleave Vector Quantization
  PCM    Pulse Code Modulated audio
```

but other types may be used, not for these types though. This is used in a similar way to the predefined types in the "TMED" frame, but without parentheses. If this frame is not present audio type is assumed to be "MPG".

TIME
The 'Time' frame is a numeric string in the HHMM format containing the time for the recording. This field is always four characters long.

TIT1
The 'Content group description' frame is used if the sound belongs to a larger category of sounds/music. For example, classical music is often sorted in different musical sections (e.g. "Piano Concerto", "Weather - Hurricane").

TIT2
The 'Title/Songname/Content description' frame is the actual name of the piece (e.g. "Adagio", "Hurricane Donna").

TIT3
The 'Subtitle/Description refinement' frame is used for information directly related to the contents title (e.g. "Op. 16" or "Performed live at Wembley").

TKEY
The 'Initial key' frame contains the musical key in which the sound starts. It is represented as a string with a maximum length of three characters. The ground keys are represented with "A","B","C","D","E", "F" and "G" and halfkeys represented with "b" and "#". Minor is represented as "m". Example "Cbm". Off key is represented with an "o" only.

TLAN
The 'Language(s)' frame should contain the languages of the text or lyrics spoken or sung in the audio. The language is represented with three characters according to ISO-639-2. If more than one language is used in the text their language codes should follow according to their usage.

TLEN
The 'Length' frame contains the length of the audiofile in milliseconds, represented as a numeric string.

TMED
The 'Media type' frame describes from which media the sound originated. This may be a text string or a reference to the predefined media types found in the list below. References are made within "(" and ")" and are optionally followed by a text refinement, e.g. "(MC) with four channels". If a text refinement should begin with a "(" character it should be replaced with "((" in the same way as in the "TCO" frame. Predefined refinements is appended after the media type, e.g. "(CD/A)" or "(VID/PAL/VHS)".

DIG Other digital media
/A Analog transfer from media

ANA Other analog media
/WAC Wax cylinder
/8CA 8-track tape cassette

```
CD     CD
  /A    Analog transfer from media
  /DD   DDD
  /AD   ADD
  /AA   AAD

LD     Laserdisc
  /A     Analog transfer from media

TT     Turntable records
  /33    33.33 rpm
  /45    45 rpm
  /71    71.29 rpm
  /76    76.59 rpm
  /78    78.26 rpm
  /80    80 rpm

MD     MiniDisc
  /A    Analog transfer from media

DAT    DAT
  /A    Analog transfer from media
  /1    standard, 48 kHz/16 bits, linear
  /2    mode 2, 32 kHz/16 bits, linear
  /3    mode 3, 32 kHz/12 bits, nonlinear, low speed
  /4    mode 4, 32 kHz/12 bits, 4 channels
  /5    mode 5, 44.1 kHz/16 bits, linear
  /6    mode 6, 44.1 kHz/16 bits, 'wide track' play

DCC    DCC
  /A    Analog transfer from media

DVD    DVD
  /A    Analog transfer from media

TV     Television
  /PAL    PAL
  /NTSC   NTSC
  /SECAM  SECAM

VID    Video
  /PAL    PAL
  /NTSC   NTSC
  /SECAM  SECAM
  /VHS    VHS
  /SVHS   S-VHS
  /BETA   BETAMAX

RAD    Radio
  /FM   FM
  /AM   AM
  /LW   LW
  /MW   MW

TEL    Telephone
  /I    ISDN
```

```
  MC     MC (normal cassette)
    /4    4.75 cm/s (normal speed for a two sided cassette)
    /9    9.5 cm/s
    /I    Type I cassette (ferric/normal)
    /II   Type II cassette (chrome)
    /III  Type III cassette (ferric chrome)
    /IV   Type IV cassette (metal)

  REE    Reel
    /9    9.5 cm/s
    /19   19 cm/s
    /38   38 cm/s
    /76   76 cm/s
    /I    Type I cassette (ferric/normal)
    /II   Type II cassette (chrome)
    /III  Type III cassette (ferric chrome)
    /IV   Type IV cassette (metal)
```

TOAL
The 'Original album/movie/show title' frame is intended for the title of the original recording (or source of sound), if for example the music in the file should be a cover of a previously released song.

TOFN
The 'Original filename' frame contains the preferred filename for the file, since some media doesn't allow the desired length of the filename. The filename is case sensitive and includes its suffix.

TOLY
The 'Original lyricist(s)/text writer(s)' frame is intended for the text writer(s) of the original recording, if for example the music in the file should be a cover of a previously released song. The text writers are seperated with the "/" character.

TOPE
The 'Original artist(s)/performer(s)' frame is intended for the performer(s) of the original recording, if for example the music in the file should be a cover of a previously released song. The performers are seperated with the "/" character.

TORY
The 'Original release year' frame is intended for the year when the original recording, if for example the music in the file should be a cover of a previously released song, was released. The field is formatted as in the "TYER" frame.

TOWN
The 'File owner/licensee' frame contains the name of the owner or licensee of the file and it's contents.

TPE1
The 'Lead artist(s)/Lead performer(s)/Soloist(s)/Performing group' is used for the main artist(s). They are seperated with the "/" character.

TPE2
The 'Band/Orchestra/Accompaniment' frame is used for additional

information about the performers in the recording.

TPE3
The 'Conductor' frame is used for the name of the conductor.

TPE4
The 'Interpreted, remixed, or otherwise modified by' frame contains more information about the people behind a remix and similar interpretations of another existing piece.

TPOS
The 'Part of a set' frame is a numeric string that describes which part of a set the audio came from. This frame is used if the source described in the "TALB" frame is divided into several mediums, e.g. a double CD. The value may be extended with a "/" character and a numeric string containing the total number of parts in the set. E.g. "1/2".

TPUB
The 'Publisher' frame simply contains the name of the label or publisher.

TRCK
The 'Track number/Position in set' frame is a numeric string containing the order number of the audio-file on its original recording. This may be extended with a "/" character and a numeric string containing the total numer of tracks/elements on the original recording. E.g. "4/9".

TRDA
The 'Recording dates' frame is a intended to be used as complement to the "TYER", "TDAT" and "TIME" frames. E.g. "4th-7th June, 12th June" in combination with the "TYER" frame.

TRSN
The 'Internet radio station name' frame contains the name of the internet radio station from which the audio is streamed.

TRSO
The 'Internet radio station owner' frame contains the name of the owner of the internet radio station from which the audio is streamed.

TSIZ
The 'Size' frame contains the size of the audiofile in bytes, excluding the ID3v2 tag, represented as a numeric string.

TSRC
The 'ISRC' frame should contain the International Standard Recording Code [ISRC] (12 characters).

TSSE
The 'Software/Hardware and settings used for encoding' frame includes the used audio encoder and its settings when the file was encoded. Hardware refers to hardware encoders, not the computer on which a program was run.

TYER
The 'Year' frame is a numeric string with a year of the recording. This frames is always four characters long (until the year 10000).

4.2.2. User defined text information frame

This frame is intended for one-string text information concerning the audiofile in a similar way to the other "T"-frames. The frame body consists of a description of the string, represented as a terminated string, followed by the actual string. There may be more than one "TXXX" frame in each tag, but only one with the same description.

```
<Header for 'User defined text information frame', ID: "TXXX">
Text encoding     $xx
Description       <text string according to encoding> $00 (00)
Value             <text string according to encoding>
```

4.3. URL link frames

With these frames dynamic data such as webpages with touring information, price information or plain ordinary news can be added to the tag. There may only be one URL [URL] link frame of its kind in an tag, except when stated otherwise in the frame description. If the textstring is followed by a termination ($00 (00)) all the following information should be ignored and not be displayed. All URL link frame identifiers begins with "W". Only URL link frame identifiers begins with "W". All URL link frames have the following format:

```
<Header for 'URL link frame', ID: "W000" - "WZZZ", excluding "WXXX"
described in 4.3.2.>
URL              <text string>
```

4.3.1. URL link frames - details

WCOM
The 'Commercial information' frame is a URL pointing at a webpage with information such as where the album can be bought. There may be more than one "WCOM" frame in a tag, but not with the same content.

WCOP
The 'Copyright/Legal information' frame is a URL pointing at a webpage where the terms of use and ownership of the file is described.

WOAF
The 'Official audio file webpage' frame is a URL pointing at a file specific webpage.

WOAR
The 'Official artist/performer webpage' frame is a URL pointing at the artists official webpage. There may be more than one "WOAR" frame in a tag if the audio contains more than one performer, but not with the same content.

WOAS
The 'Official audio source webpage' frame is a URL pointing at the official webpage for the source of the audio file, e.g. a movie.

WORS
The 'Official internet radio station homepage' contains a URL pointing at the homepage of the internet radio station.

WPAY
The 'Payment' frame is a URL pointing at a webpage that will handle the process of paying for this file.

WPUB
The 'Publishers official webpage' frame is a URL pointing at the official wepage for the publisher.

4.3.2. User defined URL link frame

This frame is intended for URL [URL] links concerning the audiofile in a similar way to the other "W"-frames. The frame body consists of a description of the string, represented as a terminated string, followed by the actual URL. The URL is always encoded with ISO-8859-1 [ISO-8859-1]. There may be more than one "WXXX" frame in each tag, but only one with the same description.

```
<Header for 'User defined URL link frame', ID: "WXXX">
Text encoding     $xx
Description       <text string according to encoding> $00 (00)
URL               <text string>
```

4.4. Involved people list

Since there might be a lot of people contributing to an audio file in various ways, such as musicians and technicians, the 'Text information frames' are often insufficient to list everyone involved in a project. The 'Involved people list' is a frame containing the names of those involved, and how they were involved. The body simply contains a terminated string with the involvement directly followed by a terminated string with the involvee followed by a new involvement and so on. There may only be one "IPLS" frame in each tag.

```
<Header for 'Involved people list', ID: "IPLS">
Text encoding          $xx
People list strings    <text strings according to encoding>
```

4.5. Music CD identifier

This frame is intended for music that comes from a CD, so that the CD can be identified in databases such as the CDDB [CDDB]. The frame consists of a binary dump of the Table Of Contents, TOC, from the CD, which is a header of 4 bytes and then 8 bytes/track on the CD plus 8 bytes for the 'lead out' making a maximum of 804 bytes. The offset to the beginning of every track on the CD should be described with a

four bytes absolute CD-frame address per track, and not with absolute time. This frame requires a present and valid "TRCK" frame, even if the CD's only got one track. There may only be one "MCDI" frame in each tag.

```
<Header for 'Music CD identifier', ID: "MCDI">
CD TOC                <binary data>
```

4.6. Event timing codes

This frame allows synchronisation with key events in a song or sound. The header is:

```
<Header for 'Event timing codes', ID: "ETCO">
Time stamp format    $xx
```

Where time stamp format is:

```
$01  Absolute time, 32 bit sized, using MPEG [MPEG] frames as unit
$02  Absolute time, 32 bit sized, using milliseconds as unit
```

Abolute time means that every stamp contains the time from the beginning of the file.

Followed by a list of key events in the following format:

```
Type of event   $xx
Time stamp      $xx (xx ...)
```

The 'Time stamp' is set to zero if directly at the beginning of the sound or after the previous event. All events should be sorted in chronological order. The type of event is as follows:

```
$00  padding (has no meaning)
$01  end of initial silence
$02  intro start
$03  mainpart start
$04  outro start
$05  outro end
$06  verse start
$07  refrain start
$08  interlude start
$09  theme start
$0A  variation start
$0B  key change
$0C  time change
$0D  momentary unwanted noise (Snap, Crackle & Pop)
$0E  sustained noise
$0F  sustained noise end
$10  intro end
$11  mainpart end
$12  verse end
$13  refrain end
$14  theme end

$15-$DF  reserved for future use
```

```
$E0-$EF  not predefined sync 0-F

$F0-$FC  reserved for future use

$FD  audio end (start of silence)
$FE  audio file ends
$FF  one more byte of events follows (all the following bytes with
     the value $FF have the same function)
```

Terminating the start events such as "intro start" is not required. The 'Not predefined sync's ($E0-EF) are for user events. You might want to synchronise your music to something, like setting of an explosion on-stage, turning on your screensaver etc.

There may only be one "ETCO" frame in each tag.

4.7. MPEG location lookup table

To increase performance and accuracy of jumps within a MPEG [MPEG] audio file, frames with timecodes in different locations in the file might be useful. The ID3v2 frame includes references that the software can use to calculate positions in the file. After the frame header is a descriptor of how much the 'frame counter' should increase for every reference. If this value is two then the first reference points out the second frame, the 2nd reference the 4th frame, the 3rd reference the 6th frame etc. In a similar way the 'bytes between reference' and 'milliseconds between reference' points out bytes and milliseconds respectively.

Each reference consists of two parts; a certain number of bits, as defined in 'bits for bytes deviation', that describes the difference between what is said in 'bytes between reference' and the reality and a certain number of bits, as defined in 'bits for milliseconds deviation', that describes the difference between what is said in 'milliseconds between reference' and the reality. The number of bits in every reference, i.e. 'bits for bytes deviation'+'bits for milliseconds deviation', must be a multiple of four. There may only be one "MLLT" frame in each tag.

```
<Header for 'Location lookup table', ID: "MLLT">
MPEG frames between reference  $xx xx
Bytes between reference        $xx xx xx
Milliseconds between reference $xx xx xx
Bits for bytes deviation       $xx
Bits for milliseconds dev.     $xx
```

Then for every reference the following data is included;

```
Deviation in bytes         %xxx....
Deviation in milliseconds  %xxx....
```

4.8. Synchronised tempo codes

For a more accurate description of the tempo of a musical piece this

frame might be used. After the header follows one byte describing which time stamp format should be used. Then follows one or more tempo codes. Each tempo code consists of one tempo part and one time part. The tempo is in BPM described with one or two bytes. If the first byte has the value $FF, one more byte follows, which is added to the first giving a range from 2 - 510 BPM, since $00 and $01 is reserved. $00 is used to describe a beat-free time period, which is not the same as a music-free time period. $01 is used to indicate one single beat-stroke followed by a beat-free period.

The tempo descriptor is followed by a time stamp. Every time the tempo in the music changes, a tempo descriptor may indicate this for the player. All tempo descriptors should be sorted in chronological order. The first beat-stroke in a time-period is at the same time as the beat description occurs. There may only be one "SYTC" frame in each tag.

```
<Header for 'Synchronised tempo codes', ID: "SYTC">
Time stamp format   $xx
Tempo data          <binary data>
```

Where time stamp format is:

```
$01  Absolute time, 32 bit sized, using MPEG [MPEG] frames as unit
$02  Absolute time, 32 bit sized, using milliseconds as unit
```

Abolute time means that every stamp contains the time from the beginning of the file.

4.9. Unsychronised lyrics/text transcription

This frame contains the lyrics of the song or a text transcription of other vocal activities. The head includes an encoding descriptor and a content descriptor. The body consists of the actual text. The 'Content descriptor' is a terminated string. If no descriptor is entered, 'Content descriptor' is $00 (00) only. Newline characters are allowed in the text. There may be more than one 'Unsynchronised lyrics/text transcription' frame in each tag, but only one with the same language and content descriptor.

```
<Header for 'Unsynchronised lyrics/text transcription', ID: "USLT">
Text encoding        $xx
Language             $xx xx xx
Content descriptor   <text string according to encoding> $00 (00)
Lyrics/text          <full text string according to encoding>
```

4.10. Synchronised lyrics/text

This is another way of incorporating the words, said or sung lyrics, in the audio file as text, this time, however, in sync with the audio. It might also be used to describing events e.g. occurring on a stage or on the screen in sync with the audio. The header includes a content descriptor, represented with as terminated textstring. If no descriptor is entered, 'Content descriptor' is $00 (00) only.

```
  <Header for 'Synchronised lyrics/text', ID: "SYLT">
  Text encoding        $xx
  Language             $xx xx xx
  Time stamp format    $xx
  Content type         $xx
  Content descriptor   <text string according to encoding> $00 (00)
```

```
Encoding:   $00  ISO-8859-1 [ISO-8859-1] character set is used => $00
                 is sync identifier.
            $01  Unicode [UNICODE] character set is used => $00 00 is
                 sync identifier.
```

```
Content type:   $00 is other
                $01 is lyrics
                $02 is text transcription
                $03 is movement/part name (e.g. "Adagio")
                $04 is events (e.g. "Don Quijote enters the stage")
                $05 is chord (e.g. "Bb F Fsus")
                $06 is trivia/'pop up' information
```

Time stamp format is:

```
  $01  Absolute time, 32 bit sized, using MPEG [MPEG] frames as unit
  $02  Absolute time, 32 bit sized, using milliseconds as unit
```

Abolute time means that every stamp contains the time from the beginning of the file.

The text that follows the frame header differs from that of the unsynchronised lyrics/text transcription in one major way. Each syllable (or whatever size of text is considered to be convenient by the encoder) is a null terminated string followed by a time stamp denoting where in the sound file it belongs. Each sync thus has the following structure:

```
  Terminated text to be synced (typically a syllable)
  Sync identifier (terminator to above string)   $00 (00)
  Time stamp                                     $xx (xx ...)
```

The 'time stamp' is set to zero or the whole sync is omitted if located directly at the beginning of the sound. All time stamps should be sorted in chronological order. The sync can be considered as a validator of the subsequent string.

Newline ($0A) characters are allowed in all "SYLT" frames and should be used after every entry (name, event etc.) in a frame with the content type $03 - $04.

A few considerations regarding whitespace characters: Whitespace separating words should mark the beginning of a new word, thus occurring in front of the first syllable of a new word. This is also valid for new line characters. A syllable followed by a comma should not be broken apart with a sync (both the syllable and the comma should be before the sync).

An example: The "USLT" passage

"Strangers in the night" $0A "Exchanging glances"

would be "SYLT" encoded as:

"Strang" $00 xx xx "ers" $00 xx xx " in" $00 xx xx " the" $00 xx xx " night" $00 xx xx 0A "Ex" $00 xx xx "chang" $00 xx xx "ing" $00 xx xx "glan" $00 xx xx "ces" $00 xx xx

There may be more than one "SYLT" frame in each tag, but only one with the same language and content descriptor.

4.11. Comments

This frame is indended for any kind of full text information that does not fit in any other frame. It consists of a frame header followed by encoding, language and content descriptors and is ended with the actual comment as a text string. Newline characters are allowed in the comment text string. There may be more than one comment frame in each tag, but only one with the same language and content descriptor.

```
<Header for 'Comment', ID: "COMM">
Text encoding          $xx
Language               $xx xx xx
Short content descrip. <text string according to encoding> $00 (00)
The actual text        <full text string according to encoding>
```

4.12. Relative volume adjustment

This is a more subjective function than the previous ones. It allows the user to say how much he wants to increase/decrease the volume on each channel while the file is played. The purpose is to be able to align all files to a reference volume, so that you don't have to change the volume constantly. This frame may also be used to balance adjust the audio. If the volume peak levels are known then this could be described with the 'Peak volume right' and 'Peak volume left' field. If Peakvolume is not known these fields could be left zeroed or, if no other data follows, be completely omitted. There may only be one "RVAD" frame in each tag.

```
<Header for 'Relative volume adjustment', ID: "RVAD">
Increment/decrement           %00xxxxxx
Bits used for volume descr.   $xx
Relative volume change, right $xx xx (xx ...)
Relative volume change, left  $xx xx (xx ...)
Peak volume right             $xx xx (xx ...)
Peak volume left              $xx xx (xx ...)
```

In the increment/decrement field bit 0 is used to indicate the right channel and bit 1 is used to indicate the left channel. 1 is increment and 0 is decrement.

The 'bits used for volume description' field is normally $10 (16 bits) for MPEG 2 layer I, II and III [MPEG] and MPEG 2.5. This value

may not be $00. The volume is always represented with whole bytes, padded in the beginning (highest bits) when 'bits used for volume description' is not a multiple of eight.

This datablock is then optionally followed by a volume definition for the left and right back channels. If this information is appended to the frame the first two channels will be treated as front channels. In the increment/decrement field bit 2 is used to indicate the right back channel and bit 3 for the left back channel.

```
Relative volume change, right back $xx xx (xx ...)
Relative volume change, left back  $xx xx (xx ...)
Peak volume right back             $xx xx (xx ...)
Peak volume left back              $xx xx (xx ...)
```

If the center channel adjustment is present the following is appended to the existing frame, after the left and right back channels. The center channel is represented by bit 4 in the increase/decrease field.

```
Relative volume change, center  $xx xx (xx ...)
Peak volume center              $xx xx (xx ...)
```

If the bass channel adjustment is present the following is appended to the existing frame, after the center channel. The bass channel is represented by bit 5 in the increase/decrease field.

```
Relative volume change, bass  $xx xx (xx ...)
Peak volume bass              $xx xx (xx ...)
```

4.13. Equalisation

This is another subjective, alignment frame. It allows the user to predefine an equalisation curve within the audio file. There may only be one "EQUA" frame in each tag.

```
<Header of 'Equalisation', ID: "EQUA">
Adjustment bits    $xx
```

The 'adjustment bits' field defines the number of bits used for representation of the adjustment. This is normally $10 (16 bits) for MPEG 2 layer I, II and III [MPEG] and MPEG 2.5. This value may not be $00.

This is followed by 2 bytes + ('adjustment bits' rounded up to the nearest byte) for every equalisation band in the following format, giving a frequency range of 0 - 32767Hz:

```
Increment/decrement   %x (MSB of the Frequency)
Frequency             (lower 15 bits)
Adjustment            $xx (xx ...)
```

The increment/decrement bit is 1 for increment and 0 for decrement. The equalisation bands should be ordered increasingly with reference to frequency. All frequencies don't have to be declared. The equalisation curve in the reading software should be interpolated

between the values in this frame. Three equal adjustments for three subsequent frequencies. A frequency should only be described once in the frame.

4.14. Reverb

Yet another subjective one. You may here adjust echoes of different kinds. Reverb left/right is the delay between every bounce in ms. Reverb bounces left/right is the number of bounces that should be made. $FF equals an infinite number of bounces. Feedback is the amount of volume that should be returned to the next echo bounce. $00 is 0%, $FF is 100%. If this value were $7F, there would be 50% volume reduction on the first bounce, 50% of that on the second and so on. Left to left means the sound from the left bounce to be played in the left speaker, while left to right means sound from the left bounce to be played in the right speaker.

'Premix left to right' is the amount of left sound to be mixed in the right before any reverb is applied, where $00 id 0% and $FF is 100%. 'Premix right to left' does the same thing, but right to left. Setting both premix to $FF would result in a mono output (if the reverb is applied symmetric). There may only be one "RVRB" frame in each tag.

```
  <Header for 'Reverb', ID: "RVRB">
  Reverb left (ms)                $xx xx
  Reverb right (ms)               $xx xx
  Reverb bounces, left            $xx
  Reverb bounces, right           $xx
  Reverb feedback, left to left   $xx
  Reverb feedback, left to right  $xx
  Reverb feedback, right to right $xx
  Reverb feedback, right to left  $xx
  Premix left to right            $xx
  Premix right to left            $xx
```

4.15. Attached picture

This frame contains a picture directly related to the audio file. Image format is the MIME type and subtype [MIME] for the image. In the event that the MIME media type name is omitted, "image/" will be implied. The "image/png" [PNG] or "image/jpeg" [JFIF] picture format should be used when interoperability is wanted. Description is a short description of the picture, represented as a terminated textstring. The description has a maximum length of 64 characters, but may be empty. There may be several pictures attached to one file, each in their individual "APIC" frame, but only one with the same content descriptor. There may only be one picture with the picture type declared as picture type $01 and $02 respectively. There is the possibility to put only a link to the image file by using the 'MIME type' "-->" and having a complete URL [URL] instead of picture data. The use of linked files should however be used sparingly since there is the risk of separation of files.

```
  <Header for 'Attached picture', ID: "APIC">
```

```
  Text encoding      $xx
  MIME type          <text string> $00
  Picture type       $xx
  Description        <text string according to encoding> $00 (00)
  Picture data       <binary data>
```

```
Picture type:  $00  Other
               $01  32x32 pixels 'file icon' (PNG only)
               $02  Other file icon
               $03  Cover (front)
               $04  Cover (back)
               $05  Leaflet page
               $06  Media (e.g. lable side of CD)
               $07  Lead artist/lead performer/soloist
               $08  Artist/performer
               $09  Conductor
               $0A  Band/Orchestra
               $0B  Composer
               $0C  Lyricist/text writer
               $0D  Recording Location
               $0E  During recording
               $0F  During performance
               $10  Movie/video screen capture
               $11  A bright coloured fish
               $12  Illustration
               $13  Band/artist logotype
               $14  Publisher/Studio logotype
```

4.16. General encapsulated object

In this frame any type of file can be encapsulated. After the header, 'Frame size' and 'Encoding' follows 'MIME type' [MIME] represented as as a terminated string encoded with ISO 8859-1 [ISO-8859-1]. The filename is case sensitive and is encoded as 'Encoding'. Then follows a content description as terminated string, encoded as 'Encoding'. The last thing in the frame is the actual object. The first two strings may be omitted, leaving only their terminations. MIME type is always an ISO-8859-1 text string. There may be more than one "GEOB" frame in each tag, but only one with the same content descriptor.

```
  <Header for 'General encapsulated object', ID: "GEOB">
  Text encoding          $xx
  MIME type              <text string> $00
  Filename               <text string according to encoding> $00 (00)
  Content description    <text string according to encóding> $00 (00)
  Encapsulated object    <binary data>
```

4.17. Play counter

This is simply a counter of the number of times a file has been played. The value is increased by one every time the file begins to play. There may only be one "PCNT" frame in each tag. When the counter reaches all one's, one byte is inserted in front of the counter thus making the counter eight bits bigger. The counter must

be at least 32-bits long to begin with.

```
<Header for 'Play counter', ID: "PCNT">
Counter        $xx xx xx xx (xx ...)
```

4.18. Popularimeter

The purpose of this frame is to specify how good an audio file is. Many interesting applications could be found to this frame such as a playlist that features better audiofiles more often than others or it could be used to profile a person's taste and find other 'good' files by comparing people's profiles. The frame is very simple. It contains the email address to the user, one rating byte and a four byte play counter, intended to be increased with one for every time the file is played. The email is a terminated string. The rating is 1-255 where 1 is worst and 255 is best. 0 is unknown. If no personal counter is wanted it may be omitted. When the counter reaches all one's, one byte is inserted in front of the counter thus making the counter eight bits bigger in the same away as the play counter ("PCNT"). There may be more than one "POPM" frame in each tag, but only one with the same email address.

```
<Header for 'Popularimeter', ID: "POPM">
Email to user   <text string> $00
Rating          $xx
Counter         $xx xx xx xx (xx ...)
```

4.19. Recommended buffer size

Sometimes the server from which a audio file is streamed is aware of transmission or coding problems resulting in interruptions in the audio stream. In these cases, the size of the buffer can be recommended by the server using this frame. If the 'embedded info flag' is true (1) then this indicates that an ID3 tag with the maximum size described in 'Buffer size' may occur in the audiostream. In such case the tag should reside between two MPEG [MPEG] frames, if the audio is MPEG encoded. If the position of the next tag is known, 'offset to next tag' may be used. The offset is calculated from the end of tag in which this frame resides to the first byte of the header in the next. This field may be omitted. Embedded tags are generally not recommended since this could render unpredictable behaviour from present software/hardware.

For applications like streaming audio it might be an idea to embed tags into the audio stream though. If the clients connects to individual connections like HTTP and there is a possibility to begin every transmission with a tag, then this tag should include a 'recommended buffer size' frame. If the client is connected to a arbitrary point in the stream, such as radio or multicast, then the 'recommended buffer size' frame should be included in every tag. Every tag that is picked up after the initial/first tag is to be considered as an update of the previous one. E.g. if there is a "TIT2" frame in the first received tag and one in the second tag, then the first should be 'replaced' with the second.

The 'Buffer size' should be kept to a minimum. There may only be one "RBUF" frame in each tag.

```
<Header for 'Recommended buffer size', ID: "RBUF">
Buffer size               $xx xx xx
Embedded info flag        %0000000x
Offset to next tag        $xx xx xx xx
```

4.20. Audio encryption

This frame indicates if the actual audio stream is encrypted, and by whom. Since standardisation of such encrypion scheme is beyond this document, all "AENC" frames begin with a terminated string with a URL containing an email address, or a link to a location where an email address can be found, that belongs to the organisation responsible for this specific encrypted audio file. Questions regarding the encrypted audio should be sent to the email address specified. If a $00 is found directly after the 'Frame size' and the audiofile indeed is encrypted, the whole file may be considered useless.

After the 'Owner identifier', a pointer to an unencrypted part of the audio can be specified. The 'Preview start' and 'Preview length' is described in frames. If no part is unencrypted, these fields should be left zeroed. After the 'preview length' field follows optionally a datablock required for decryption of the audio. There may be more than one "AENC" frames in a tag, but only one with the same 'Owner identifier'.

```
<Header for 'Audio encryption', ID: "AENC">
Owner identifier   <text string> $00
Preview start      $xx xx
Preview length     $xx xx
Encryption info    <binary data>
```

4.21. Linked information

To keep space waste as low as possible this frame may be used to link information from another ID3v2 tag that might reside in another audio file or alone in a binary file. It is recommended that this method is only used when the files are stored on a CD-ROM or other circumstances when the risk of file seperation is low. The frame contains a frame identifier, which is the frame that should be linked into this tag, a URL [URL] field, where a reference to the file where the frame is given, and additional ID data, if needed. Data should be retrieved from the first tag found in the file to which this link points. There may be more than one "LINK" frame in a tag, but only one with the same contents. A linked frame is to be considered as part of the tag and has the same restrictions as if it was a physical part of the tag (i.e. only one "RVRB" frame allowed, whether it's linked or not).

```
<Header for 'Linked information', ID: "LINK">
Frame identifier        $xx xx xx
URL                     <text string> $00
```

```
  ID and additional data  <text string(s)>
```

Frames that may be linked and need no additional data are "IPLS", "MCID", "ETCO", "MLLT", "SYTC", "RVAD", "EQUA", "RVRB", "RBUF", the text information frames and the URL link frames.

The "TXXX", "APIC", "GEOB" and "AENC" frames may be linked with the content descriptor as additional ID data.

The "COMM", "SYLT" and "USLT" frames may be linked with three bytes of language descriptor directly followed by a content descriptor as additional ID data.

4.22. Position synchronisation frame

This frame delivers information to the listener of how far into the audio stream he picked up; in effect, it states the time offset of the first frame in the stream. The frame layout is:

```
  <Head for 'Position synchronisation', ID: "POSS">
  Time stamp format         $xx
  Position                  $xx (xx ...)
```

Where time stamp format is:

```
  $01  Absolute time, 32 bit sized, using MPEG frames as unit
  $02  Absolute time, 32 bit sized, using milliseconds as unit
```

and position is where in the audio the listener starts to receive, i.e. the beginning of the next frame. If this frame is used in the beginning of a file the value is always 0. There may only be one "POSS" frame in each tag.

4.23. Terms of use frame

This frame contains a brief description of the terms of use and ownership of the file. More detailed information concerning the legal terms might be available through the "WCOP" frame. Newlines are allowed in the text. There may only be one "USER" frame in a tag.

```
  <Header for 'Terms of use frame', ID: "USER">
  Text encoding        $xx
  Language             $xx xx xx
  The actual text      <text string according to encoding>
```

4.24. Ownership frame

The ownership frame might be used as a reminder of a made transaction or, if signed, as proof. Note that the "USER" and "TOWN" frames are good to use in conjunction with this one. The frame begins, after the frame ID, size and encoding fields, with a 'price payed' field. The first three characters of this field contains the currency used for the transaction, encoded according to ISO 4217 [ISO-4217] alphabetic currency code. Concatenated to this is the actual price payed, as a

numerical string using "." as the decimal separator. Next is an 8 character date string (YYYYMMDD) followed by a string with the name of the seller as the last field in the frame. There may only be one "OWNE" frame in a tag.

```
<Header for 'Ownership frame', ID: "OWNE">
Text encoding     $xx
Price payed       <text string> $00
Date of purch.    <text string>
Seller            <text string according to encoding>
```

4.25. Commercial frame

This frame enables several competing offers in the same tag by bundling all needed information. That makes this frame rather complex but it's an easier solution than if one tries to achieve the same result with several frames. The frame begins, after the frame ID, size and encoding fields, with a price string field. A price is constructed by one three character currency code, encoded according to ISO 4217 [ISO-4217] alphabetic currency code, followed by a numerical value where "." is used as decimal seperator. In the price string several prices may be concatenated, seperated by a "/" character, but there may only be one currency of each type.

The price string is followed by an 8 character date string in the format YYYYMMDD, describing for how long the price is valid. After that is a contact URL, with which the user can contact the seller, followed by a one byte 'received as' field. It describes how the audio is delivered when bought according to the following list:

```
$00  Other
$01  Standard CD album with other songs
$02  Compressed audio on CD
$03  File over the Internet
$04  Stream over the Internet
$05  As note sheets
$06  As note sheets in a book with other sheets
$07  Music on other media
$08  Non-musical merchandise
```

Next follows a terminated string with the name of the seller followed by a terminated string with a short description of the product. The last thing is the ability to include a company logotype. The first of them is the 'Picture MIME type' field containing information about which picture format is used. In the event that the MIME media type name is omitted, "image/" will be implied. Currently only "image/png" and "image/jpeg" are allowed. This format string is followed by the binary picture data. This two last fields may be omitted if no picture is to attach.

```
<Header for 'Commercial frame', ID: "COMR">
Text encoding      $xx
Price string       <text string> $00
Valid until        <text string>
Contact URL        <text string> $00
Received as        $xx
```

```
Name of seller     <text string according to encoding> $00 (00)
Description        <text string according to encoding> $00 (00)
Picture MIME type  <string> $00
Seller logo        <binary data>
```

4.26. Encryption method registration

To identify with which method a frame has been encrypted the encryption method must be registered in the tag with this frame. The 'Owner identifier' is a null-terminated string with a URL [URL] containing an email address, or a link to a location where an email address can be found, that belongs to the organisation responsible for this specific encryption method. Questions regarding the encryption method should be sent to the indicated email address. The 'Method symbol' contains a value that is associated with this method throughout the whole tag. Values below $80 are reserved. The 'Method symbol' may optionally be followed by encryption specific data. There may be several "ENCR" frames in a tag but only one containing the same symbol and only one containing the same owner identifier. The method must be used somewhere in the tag. See section 3.3.1, flag j for more information.

```
<Header for 'Encryption method registration', ID: "ENCR">
Owner identifier    <text string> $00
Method symbol       $xx
Encryption data     <binary data>
```

4.27. Group identification registration

This frame enables grouping of otherwise unrelated frames. This can be used when some frames are to be signed. To identify which frames belongs to a set of frames a group identifier must be registered in the tag with this frame. The 'Owner identifier' is a null-terminated string with a URL [URL] containing an email address, or a link to a location where an email address can be found, that belongs to the organisation responsible for this grouping. Questions regarding the grouping should be sent to the indicated email address. The 'Group symbol' contains a value that associates the frame with this group throughout the whole tag. Values below $80 are reserved. The 'Group symbol' may optionally be followed by some group specific data, e.g. a digital signature. There may be several "GRID" frames in a tag but only one containing the same symbol and only one containing the same owner identifier. The group symbol must be used somewhere in the tag. See section 3.3.1, flag j for more information.

```
<Header for 'Group ID registration', ID: "GRID">
Owner identifier      <text string> $00
Group symbol          $xx
  Group dependent data  <binary data>
```

4.28. Private frame

This frame is used to contain information from a software producer that its program uses and does not fit into the other frames. The

frame consists of an 'Owner identifier' string and the binary data. The 'Owner identifier' is a null-terminated string with a URL [URL] containing an email address, or a link to a location where an email address can be found, that belongs to the organisation responsible for the frame. Questions regarding the frame should be sent to the indicated email address. The tag may contain more than one "PRIV" frame but only with different contents. It is recommended to keep the number of "PRIV" frames as low as possible.

```
<Header for 'Private frame', ID: "PRIV">
Owner identifier      <text string> $00
  The private data      <binary data>
```

5. The 'unsynchronisation scheme'

The only purpose of the 'unsynchronisation scheme' is to make the ID3v2 tag as compatible as possible with existing software. There is no use in 'unsynchronising' tags if the file is only to be processed by new software. Unsynchronisation may only be made with MPEG 2 layer I, II and III and MPEG 2.5 files.

Whenever a false synchronisation is found within the tag, one zeroed byte is inserted after the first false synchronisation byte. The format of a correct sync that should be altered by ID3 encoders is as follows:

```
%11111111 111xxxxx
```

And should be replaced with:

```
%11111111 00000000 111xxxxx
```

This has the side effect that all $FF 00 combinations have to be altered, so they won't be affected by the decoding process. Therefore all the $FF 00 combinations have to be replaced with the $FF 00 00 combination during the unsynchronisation.

To indicate usage of the unsynchronisation, the first bit in 'ID3 flags' should be set. This bit should only be set if the tag contains a, now corrected, false synchronisation. The bit should only be clear if the tag does not contain any false synchronisations.

Do bear in mind, that if a compression scheme is used by the encoder, the unsynchronisation scheme should be applied *afterwards*. When decoding a compressed, 'unsynchronised' file, the 'unsynchronisation scheme' should be parsed first, decompression afterwards.

If the last byte in the tag is $FF, and there is a need to eliminate false synchronisations in the tag, at least one byte of padding should be added.

6. Copyright

7. References

[CDDB] Compact Disc Data Base

<url:http://www.cddb.com>

[ID3v2] Martin Nilsson, "ID3v2 informal standard".

<url:http://www.id3.org/id3v2-00.txt>

[ISO-639-2] ISO/FDIS 639-2.
Codes for the representation of names of languages, Part 2: Alpha-3 code. Technical committee / subcommittee: TC 37 / SC 2

[ISO-4217] ISO 4217:1995.
Codes for the representation of currencies and funds.
Technical committee / subcommittee: TC 68

[ISO-8859-1] ISO/IEC DIS 8859-1.
8-bit single-byte coded graphic character sets, Part 1: Latin alphabet No. 1. Technical committee / subcommittee: JTC 1 / SC 2

[ISRC] ISO 3901:1986
International Standard Recording Code (ISRC).
Technical committee / subcommittee: TC 46 / SC 9

[JFIF] JPEG File Interchange Format, version 1.02

<url:http://www.w3.org/Graphics/JPEG/jfif.txt>

[MIME] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.

<url:ftp://ftp.isi.edu/in-notes/rfc2045.txt>

[MPEG] ISO/IEC 11172-3:1993.
Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s, Part 3: Audio.

Technical committee / subcommittee: JTC 1 / SC 29
and
ISO/IEC 13818-3:1995
Generic coding of moving pictures and associated audio information,
Part 3: Audio.
Technical committee / subcommittee: JTC 1 / SC 29
and
ISO/IEC DIS 13818-3
Generic coding of moving pictures and associated audio information,
Part 3: Audio (Revision of ISO/IEC 13818-3:1995)

[PNG] Portable Network Graphics, version 1.0

<url:http://www.w3.org/TR/REC-png-multi.html>

[UNICODE] ISO/IEC 10646-1:1993.
Universal Multiple-Octet Coded Character Set (UCS), Part 1:
Architecture and Basic Multilingual Plane.
Technical committee / subcommittee: JTC 1 / SC 2

<url:http://www.unicode.org>

[URL] T. Berners-Lee, L. Masinter & M. McCahill, "Uniform Resource
Locators (URL).", RFC 1738, December 1994.

<url:ftp://ftp.isi.edu/in-notes/rfc1738.txt>

[ZLIB] P. Deutsch, Aladdin Enterprises & J-L. Gailly, "ZLIB
Compressed
Data Format Specification version 3.3", RFC 1950, May 1996.

<url:ftp://ftp.isi.edu/in-notes/rfc1950.txt>

8. Appendix

A. Appendix A - Genre List from ID3v1

The following genres is defined in ID3v1

0.Blues
1.Classic Rock
2.Country
3.Dance
4.Disco
5.Funk
6.Grunge
7.Hip-Hop
8.Jazz
9.Metal
10.New Age
11.Oldies
12.Other
13.Pop
14.R&B

15.Rap
16.Reggae
17.Rock
18.Techno
19.Industrial
20.Alternative
21.Ska
22.Death Metal
23.Pranks
24.Soundtrack
25.Euro-Techno
26.Ambient
27.Trip-Hop
28.Vocal
29.Jazz+Funk
30.Fusion
31.Trance
32.Classical
33.Instrumental
34.Acid
35.House
36.Game
37.Sound Clip
38.Gospel
39.Noise
40.AlternRock
41.Bass
42.Soul
43.Punk
44.Space
45.Meditative
46.Instrumental Pop
47.Instrumental Rock
48.Ethnic
49.Gothic
50.Darkwave
51.Techno-Industrial
52.Electronic
53.Pop-Folk
54.Eurodance
55.Dream
56.Southern Rock
57.Comedy
58.Cult
59.Gangsta
60.Top 40
61.Christian Rap
62.Pop/Funk
63.Jungle
64.Native American
65.Cabaret
66.New Wave
67.Psychadelic
68.Rave
69.Showtunes
70.Trailer
71.Lo-Fi

72.Tribal
73.Acid Punk
74.Acid Jazz
75.Polka
76.Retro
77.Musical
78.Rock & Roll
79.Hard Rock

The following genres are Winamp extensions

80.Folk
81.Folk-Rock
82.National Folk
83.Swing
84.Fast Fusion
85.Bebob
86.Latin
87.Revival
88.Celtic
89.Bluegrass
90.Avantgarde
91.Gothic Rock
92.Progressive Rock
93.Psychedelic Rock
94.Symphonic Rock
95.Slow Rock
96.Big Band
97.Chorus
98.Easy Listening
99.Acoustic
100.Humour
101.Speech
102.Chanson
103.Opera
104.Chamber Music
105.Sonata
106.Symphony
107.Booty Bass
108.Primus
109.Porn Groove
110.Satire
111.Slow Jam
112.Club
113.Tango
114.Samba
115.Folklore
116.Ballad
117.Power Ballad
118.Rhythmic Soul
119.Freestyle
120.Duet
121.Punk Rock
122.Drum Solo
123.Acapella
124.Euro-House
125.Dance Hall

9. Author's Address

Written by

Martin Nilsson
Rydsvägen 246 C. 30
S-584 34 Linköping
Sweden

Email: nilsson@id3.org

Edited by

Dirk Mahoney
57 Pechey Street
Chermside Q
Australia 4032

Email: dirk@id3.org

Johan Sundström
Alsättersgatan 5 A. 34
S-584 35 Linköping
Sweden

Email: johan@id3.org

HD Radio™ Air Interface Design Description - Program Service Data Appendix A-1 – ID3 Standard Reference for HD Radio Program Service Data

ID3 Standard Reference for HD Radio Program Service Data

Informal standard M. Nilsson

Document: id3v2.3.0.txt 3rd February 1999

ID3 tag version 2.3.0

Status of this document

This document is an informal standard and replaces the ID3v2.2.0 standard [ID3v2]. The informal standard is released so that implementors could have a set standard before a formal standard is set. The formal standard will use another version or revision number if not identical to what is described in this document. The contents in this document may change for clarifications but never for added or altered functionallity.

Distribution of this document is unlimited.

Abstract

This document describes the ID3v2.3.0, which is a more developed version of the ID3v2 informal standard [ID3v2] (version 2.2.0), evolved from the ID3 tagging system. The ID3v2 offers a flexible way of storing information about an audio file within itself to determine its origin and contents. The information may be technical information, such as equalisation curves, as well as related meta information, such as title, performer, copyright etc.

2. Conventions in this document

In the examples, text within "" is a text string exactly as it appears in a file. Numbers preceded with $ are hexadecimal and numbers preceded with % are binary. $xx is used to indicate a byte with unknown content. %x is used to indicate a bit with unknown content. The most significant bit (MSB) of a byte is called 'bit 7' and the least significant bit (LSB) is called 'bit 0'.

A tag is the whole tag described in this document. A frame is a block of information in the tag. The tag consists of a header, frames and optional padding. A field is a piece of information; one value, a string etc. A numeric string is a string that consists of the characters 0-9 only.

3. ID3v2 overview

The two biggest design goals were to be able to implement ID3v2 without disturbing old software too much and that ID3v2 should be as flexible and expandable as possible.

The first criterion is met by the simple fact that the MPEG [MPEG] decoding software uses a syncsignal, embedded in the audiostream, to 'lock on to' the audio. Since the ID3v2 tag doesn't contain a valid syncsignal, no software will attempt to play the tag. If, for any reason, coincidence make a syncsignal appear within the tag it will be taken care of by the 'unsynchronisation scheme' described in section 5.

The second criterion has made a more noticeable impact on the design of the ID3v2 tag. It is constructed as a container for several information blocks, called frames, whose format need not be known to the software that encounters them. At the start of every frame there is an identifier that explains the frames' format and content, and a size descriptor that allows software to skip unknown frames.

If a total revision of the ID3v2 tag should be needed, there is a version number and a size descriptor in the ID3v2 header.

The ID3 tag described in this document is mainly targeted at files encoded with MPEG-1/2 layer I, MPEG-1/2 layer II, MPEG-1/2 layer III and MPEG-2.5, but may work with other types of encoded audio.

The bitorder in ID3v2 is most significant bit first (MSB). The byteorder in multibyte numbers is most significant byte first (e.g. $12345678 would be encoded $12 34 56 78).

It is permitted to include padding after all the final frame (at the end of the ID3 tag), making the size of all the frames together smaller than the size given in the head of the tag. A possible purpose of this padding is to allow for adding a few additional frames or enlarge existing frames within the tag without having to rewrite the entire file. The value of the padding bytes must be $00.

3.1. ID3v2 header

The ID3v2 tag header, which should be the first information in the file, is 10 bytes as follows:

ID3v2/file identifier	"ID3"
ID3v2 version	$03 00
ID3v2 flags	%abc00000
ID3v2 size	4 * %0xxxxxxx

The first three bytes of the tag are always "ID3" to indicate that this is an ID3v2 tag, directly followed by the two version bytes. The first byte of ID3v2 version is it's major version, while the second byte is its revision number. In this case this is ID3v2.3.0. All revisions are backwards compatible while major versions are not. If software with ID3v2.2.0 and below support should encounter version three or higher it should simply ignore the whole tag. Version and revision will never be $FF.

The version is followed by one the ID3v2 flags field, of which currently only three flags are used.

a - Unsynchronisation

Bit 7 in the 'ID3v2 flags' indicates whether or not unsynchronisation is used (see section 5 for details); a set bit indicates usage.

b - Extended header

The second bit (bit 6) indicates whether or not the header is followed by an extended header. The extended header is described in section 3.2.

c - Experimental indicator

The third bit (bit 5) should be used as an 'experimental indicator'. This flag should always be set when the tag is in an experimental stage.

All the other flags should be cleared. If one of these undefined flags are set that might mean that the tag is not readable for a parser that does not know the flags function.

The ID3v2 tag size is encoded with four bytes where the most significant bit (bit 7) is set to zero in every byte, making a total of 28 bits. The zeroed bits are ignored, so a 257 bytes long tag is represented as $00 00 02 01.

The ID3v2 tag size is the size of the complete tag after unsychronisation, including padding, excluding the header but not excluding the extended header (total tag size - 10). Only 28 bits (representing up to 256MB) are used in the size description to avoid the introducuction of 'false syncsignals'.

An ID3v2 tag can be detected with the following pattern:

$49 44 33 yy yy xx zz zz zz zz

Where yy is less than $FF, xx is the 'flags' byte and zz is less than $80.

3.2. ID3v2 extended header

The extended header contains information that is not vital to the correct parsing of the tag information, hence the extended header is optional.

Extended header size $xx xx xx xx

Extended Flags $xx xx

Size of padding $xx xx xx xx

Where the 'Extended header size', currently 6 or 10 bytes, excludes itself. The 'Size of padding' is simply the total tag size excluding the frames and the headers, in other words the padding. The extended header is considered separate from the header proper, and as such is subject to unsynchronisation.

The extended flags are a secondary flag set which describes further attributes of the tag. These attributes are currently defined as follows

%x0000000 00000000

x - CRC data present

If this flag is set four bytes of CRC-32 data is appended to the extended header. The CRC should be calculated before unsynchronisation on the data between the extended header and the padding, i.e. the frames and only the frames.

Total frame CRC $xx xx xx xx

3.3. ID3v2 frame overview

As the tag consists of a tag header and a tag body with one or more frames, all the frames consists of a frame header followed by one or more fields containing the actual information. The layout of the frame header:

Frame ID $xx xx xx xx (four characters)

Size $xx xx xx xx

Flags $xx xx

The frame ID made out of the characters capital A-Z and 0-9. Identifiers beginning with "X", "Y" and "Z" are for experimental use and free for everyone to use, without the need to set the experimental bit in the tag header. Have in mind that someone else might have used the same identifier as you. All other identifiers are either used or reserved for future use.

The frame ID is followed by a size descriptor, making a total header size of ten bytes in every frame. The size is calculated as frame size excluding frame header (frame size - 10).

In the frame header the size descriptor is followed by two flags bytes. These flags are described in section 3.3.1.

There is no fixed order of the frames' appearance in the tag, although it is desired that the frames are arranged in order of significance concerning the recognition of the file. An example of such order: UFID, TIT2, MCDI, TRCK ...

A tag must contain at least one frame. A frame must be at least 1 byte big, excluding the header.

If nothing else is said a string is represented as ISO-8859-1 [ISO-8859-1] characters in the range $20 - $FF. Such strings are represented as <text string>, or <full text string> if newlines are allowed, in the frame descriptions. All Unicode strings [UNICODE] use 16-bit unicode 2.0 (ISO/IEC 10646-1:1993, UCS-2). Unicode strings must begin with the Unicode BOM ($FF FE or $FE FF) to identify the byte order.

All numeric strings and URLs [URL] are always encoded as ISO-8859-1. Terminated strings are terminated with $00 if encoded with ISO-8859-1 and $00 00 if encoded as unicode. If nothing else is said newline character is forbidden. In ISO-8859-1 a new line is represented, when allowed, with $0A only. Frames that allow different types of text encoding have a text encoding description byte directly after the frame size. If ISO-8859-1 is used this byte should be $00, if Unicode is used it should be $01. Strings dependent on encoding is represented as <text string according to encoding>, or <full text string according to encoding> if newlines are allowed. Any empty Unicode strings which are NULL-terminated may have the Unicode BOM followed by a Unicode NULL ($FF FE 00 00 or $FE FF 00 00).

The three byte language field is used to describe the language of the frame's content, according to ISO-639-2 [ISO-639-2].

All URLs [URL] may be relative, e.g. "picture.png", "../doc.txt".

If a frame is longer than it should be, e.g. having more fields than specified in this document, that indicates that additions to the frame have been made in a later version of the ID3v2 standard. This is reflected by the revision number in the header of the tag.

3.3.1. Frame header flags

In the frame header the size descriptor is followed by two flags bytes. All unused flags must be cleared. The first byte is for 'status messages' and the second byte is for encoding purposes. If an unknown flag is set in the first byte the frame may not be changed without the bit cleared. If an unknown flag is set in the second byte it is likely to not be readable. The flags field is defined as follows.

%abc00000 %ijk00000

a - Tag alter preservation

This flag tells the software what to do with this frame if it is unknown and the tag is altered in any way. This applies to all kinds of alterations, including adding more padding and reordering the frames.

0 Frame should be preserved.

1 Frame should be discarded.

b - File alter preservation

This flag tells the software what to do with this frame if it is unknown and the file, excluding the tag, is altered. This does not apply when the audio is completely replaced with other audio data.

0 Frame should be preserved.

1 Frame should be discarded.

c - Reserved

This bit is reserved. It may be set or cleared by external subsystems and should therefore be ignored.

i - Compression

This flag indicates whether or not the frame is compressed.

0 Frame is not compressed.

1 Frame is compressed using zlib [zlib] with 4 bytes for 'decompressed size' appended to the frame header.

j - Encryption

This flag indicates wether or not the frame is enrypted. If set one byte indicating with which method it was encrypted will be appended to the frame header. See section 4.26. for more information about encryption method registration.

0 Frame is not encrypted.

1 Frame is encrypted.

k - Grouping identity

This flag indicates whether or not this frame belongs in a group with other frames. If set a group identifier byte is added to the frame header. Every frame with the same group identifier belongs to the same group.

0 Frame does not contain group information

1 Frame contains group information

Some flags indicates that the frame header is extended with additional information. This information will be added to the frame header in the same order as the flags indicating the additions. I.e. the four bytes of decompressed size will preceed the encryption method byte. These additions to the frame header, while

not included in the frame header size but are included in the 'frame size' field, are not subject to encryption or compression.

3.3.2. Default flags

The default settings for the frames described in this document can be divided into the following classes. The flags may be set differently if found more suitable by the software.

1. Discarded if tag is altered, discarded if file is altered.

   None.

2. Discarded if tag is altered, preserved if file is altered.

   None.

3. Preserved if tag is altered, discarded if file is altered.

   AENC, ETCO, EQUA, MLLT, POSS, SYLT, SYTC, RVAD, TENC, TLEN, TSIZ

4. Preserved if tag is altered, preserved if file is altered.

   The rest of the frames.

4. This section has been omitted.

4.1. Unique file identifier

This frame's purpose is to be able to identify the audio file in a database that may contain more information relevant to the content. Since standardisation of such a database is beyond this document, all frames begin with a null-terminated string with a URL [URL] containing an email address, or a link to a location where an email address can be found, that belongs to the organisation responsible for this specific database implementation. Questions regarding the database should be sent to the indicated email address. The URL should not be used for the actual database queries. The string "http://www.id3.org/dummy/ufid.html" should be used for tests. Software that isn't told otherwise may safely remove such frames. The 'Owner identifier' must be non-empty (more than just a termination). The 'Owner identifier' is then followed by the actual identifier, which may be up to 64 bytes. There may be more than one "UFID" frame in a tag, but only one with the same 'Owner identifier'.

<Header for 'Unique file identifier', ID: "UFID">

Owner identifier <text string> $00

Identifier <up to 64 bytes binary data>

4.2. Text information frames

The text information frames are the most important frames, containing information like artist, album and more. There may only be one text information frame of its kind in an tag. If the textstring is followed by a termination ($00 (00)) all the following information should be ignored and not be displayed. All text frame identifiers begin with "T". Only text frame identifiers begin with "T", with the exception of the "TXXX" frame. All the text information frames have the following format:

<Header for 'Text information frame', ID: "T000" - "TZZZ",

excluding "TXXX" described in 4.2.2.>

Text encoding $xx

Information <text string according to encoding>

4.2.1. Text information frames - details

This section only includes frames supported in MPS.

TALB

The 'Album/Movie/Show title' frame is intended for the title of the recording(/source of sound) which the audio in the file is taken from.

TCON

The 'Content type', which previously was stored as a one byte numeric value only, is now a numeric string. You may use one or several of the types as ID3v1.1 did or, since the category list would be impossible to maintain with accurate and up to date categories, define your own.

References to the ID3v1 genres can be made by, as first byte, enter "(" followed by a number from the genres list (appendix A.) and ended with a ")" character. This is optionally followed by a refinement, e.g. "(21)" or "(4)Eurodisco". Several references can be made in the same frame, e.g. "(51)(39)". If the refinement should begin with a "(" character it should be replaced with "((", e.g. "((I can figure out any genre)" or "(55)((I think...)". The following new content types is defined in ID3v2 and is implemented in the same way as the numerig content types, e.g. "(RX)".

RX Remix

CR Cover

TIT2

The 'Title/Songname/Content description' frame is the actual name of the piece (e.g. "Adagio", "Hurricane Donna").

TPE1

The 'Lead artist(s)/Lead performer(s)/Soloist(s)/Performing group' is

used for the main artist(s). They are seperated with the "/"

character.

4.2.2. This section has been omitted.

4.3 thru 4.10 omitted

4.11. Comments

This frame is indended for any kind of full text information that does not fit in any other frame. It consists of a frame header followed by encoding, language and content descriptors and is ended with the actual comment as a text string. Newline characters are allowed in the comment text string. There may be more than one comment frame in each tag, but only one with the same language and content descriptor.

<Header for 'Comment', ID: "COMM">

Text encoding $xx

Language $xx xx xx

Short content descrip. <text string according to encoding> $00 (00)

The actual text <full text string according to encoding>

4.12 thru 4.24 omitted

4.25. Commercial frame

This frame enables several competing offers in the same tag by bundling all needed information. That makes this frame rather complex but it's an easier solution than if one tries to achieve the same result with several frames. The frame begins, after the frame ID, size and encoding fields, with a price string field. A price is constructed by one three character currency code, encoded according to ISO 4217 [ISO-4217] alphabetic currency code, followed by a numerical value where "." is used as decimal seperator. In the price string several prices may be concatenated, seperated by a "/" character, but there may only be one currency of each type.

The price string is followed by an 8 character date string in the format YYYYMMDD, describing for how long the price is valid. After that is a contact URL, with which the user can contact the seller, followed by a one byte 'received as' field. It describes how the audio is delivered when bought according to the following list:

$00 Other

$01 Standard CD album with other songs

$02 Compressed audio on CD

$03 File over the Internet

$04 Stream over the Internet

$05 As note sheets

$06 As note sheets in a book with other sheets

$07 Music on other media

$08 Non-musical merchandise

Next follows a terminated string with the name of the seller followed by a terminated string with a short description of the product. The last thing is the ability to include a company logotype. The first of them is the 'Picture MIME type' field containing information about which picture format is used. In the event that the MIME media type name is omitted, "image/" will be implied. Currently only "image/png" and "image/jpeg" are allowed. This format string is followed by the binary picture data. This two last fields may be omitted if no picture is to attach.

<Header for 'Commercial frame', ID: "COMR">

Text encoding $xx

Price string <text string> $00

Valid until <text string>

Contact URL <text string> $00

Received as $xx

Name of seller <text string according to encoding> $00 (00)

Description <text string according to encoding> $00 (00)

Picture MIME type <string> $00

Seller logo <binary data>

4.26. Encryption method registration

To identify with which method a frame has been encrypted the encryption method must be registered in the tag with this frame. The 'Owner identifier' is a null-terminated string with a URL [URL] containing an email address, or a link to a location where an email address can be found, that belongs to the organisation responsible for this specific encryption method. Questions regarding the encryption method should be sent to the indicated email address. The 'Method symbol' contains a value that is associated with this method throughout the whole tag. Values below $80 are reserved. The 'Method symbol' may optionally be followed by encryption specific data. There may be several "ENCR" frames in a tag but only one containing the same symbol and only one containing the same owner identifier. The method must be used somewhere in the tag. See section 3.3.1, flag j for more information.

<Header for 'Encryption method registration', ID: "ENCR">

Owner identifier <text string> $00

Method symbol $xx

Encryption data <binary data>

4.27 thru 4.28 omitted

5. The 'unsynchronisation scheme'

The only purpose of the 'unsynchronisation scheme' is to make the ID3v2 tag as compatible as possible with existing software. There is no use in 'unsynchronising' tags if the file is only to be processed by new software. Unsynchronisation may only be made with MPEG 2 layer I, II and III and MPEG 2.5 files.

Whenever a false synchronisation is found within the tag, one zeroed byte is inserted after the first false synchronisation byte. The format of a correct sync that should be altered by ID3 encoders is as follows:

%11111111 111xxxxx

And should be replaced with:

%11111111 00000000 111xxxxx

This has the side effect that all $FF 00 combinations have to be altered, so they won't be affected by the decoding process. Therefore all the $FF 00 combinations have to be replaced with the $FF 00 00 combination during the unsynchronisation.

To indicate usage of the unsynchronisation, the first bit in 'ID3 flags' should be set. This bit should only be set if the tag contains a, now corrected, false synchronisation. The bit should only be clear if the tag does not contain any false synchronisations.

Do bear in mind, that if a compression scheme is used by the encoder, the unsynchronisation scheme should be applied *afterwards*. When decoding a compressed, 'unsynchronised' file, the unsynchronisation scheme' should be parsed first, decompression afterwards.

If the last byte in the tag is $FF, and there is a need to eliminate false synchronisations in the tag, at least one byte of padding should be added.

6. Copyright

7. References

[CDDB] Compact Disc Data Base
<url:http://www.cddb.com>

[ID3v2] Martin Nilsson, "ID3v2 informal standard".
<url:http://www.id3.org/id3v2-00.txt>

[ISO-639-2] ISO/FDIS 639-2.
Codes for the representation of names of languages, Part 2: Alpha-3 code. Technical committee / subcommittee: TC 37 / SC 2

[ISO-4217] ISO 4217:1995.
Codes for the representation of currencies and funds. Technical committee / subcommittee: TC 68

[ISO-8859-1] ISO/IEC DIS 8859-1.
8-bit single-byte coded graphic character sets, Part 1: Latin alphabet No. 1. Technical committee / subcommittee: JTC 1 / SC 2

[ISRC] ISO 3901:1986
International Standard Recording Code (ISRC). Technical committee / subcommittee: TC 46 / SC 9

[JFIF] JPEG File Interchange Format, version 1.02
<url:http://www.w3.org/Graphics/JPEG/jfif.txt>

[MIME] Freed, N. and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", RFC 2045, November 1996.
<url:ftp://ftp.isi.edu/in-notes/rfc2045.txt>

[MPEG] ISO/IEC 11172-3:1993. Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s, Part 3: Audio. Technical committee / subcommittee: JTC 1 / SC 29
and
ISO/IEC 13818-3:1995 Generic coding of moving pictures and associated audio information, Part 3: Audio. Technical committee / subcommittee: JTC 1 / SC 29
and
ISO/IEC DIS 13818-3 Generic coding of moving pictures and associated audio information, Part 3: Audio (Revision of ISO/IEC 13818-3:1995)

[PNG] Portable Network Graphics, version 1.0
<url:http://www.w3.org/TR/REC-png-multi.html>

[UNICODE] ISO/IEC 10646-1:1993. Universal Multiple-Octet Coded Character Set (UCS), Part 1: Architecture and Basic Multilingual Plane. Technical committee / subcommittee: JTC 1 / SC 2
<url:http://www.unicode.org>

[URL] T. Berners-Lee, L. Masinter & M. McCahill, "Uniform Resource Locators (URL).", RFC 1738, December 1994.
<url:ftp://ftp.isi.edu/in-notes/rfc1738.txt>

[ZLIB] P. Deutsch, Aladdin Enterprises & J-L. Gailly, "ZLIB Compressed Data Format Specification version 3.3", RFC 1950, May 1996.
<url:ftp://ftp.isi.edu/in-notes/rfc1950.txt>

8. Genre List from ID3v1

The following genres is defined in ID3v1		The following genres are Winamp extensions
0.Blues	46.Instrumental Pop	80.Folk
1.Classic Rock	47.Instrumental Rock	81.Folk-Rock
2.Country	48.Ethnic	82.National Folk
3.Dance	49.Gothic	83.Swing
4.Disco	50.Darkwave	84.Fast Fusion
5.Funk	51.Techno-Industrial	85.Bebob
6.Grunge	52.Electronic	86.Latin
7.Hip-Hop	53.Pop-Folk	87.Revival
8.Jazz	54.Eurodance	88.Celtic
9.Metal	55.Dream	89.Bluegrass
10.New Age	56.Southern Rock	90.Avantgarde
11.Oldies	57.Comedy	91.Gothic Rock
12.Other	58.Cult	92.Progressive Rock
13.Pop	59.Gangsta	93.Psychedelic Rock
14.R&B	60.Top 40	94.Symphonic Rock
15.Rap	61.Christian Rap	95.Slow Rock
16.Reggae	62.Pop/Funk	96.Big Band
17.Rock	63.Jungle	97.Chorus
18.Techno	64.Native American	98.Easy Listening
19.Industrial	65.Cabaret	99.Acoustic
20.Alternative	66.New Wave	100.Humour
21.Ska	67.Psychadelic	101.Speech
22.Death Metal	68.Rave	102.Chanson
23.Pranks	69.Showtunes	103.Opera
24.Soundtrack	70.Trailer	104.Chamber Music
25.Euro-Techno	71.Lo-Fi	105.Sonata
26.Ambient	72.Tribal	106.Symphony
27.Trip-Hop	73.Acid Punk	107.Booty Bass
28.Vocal	74.Acid Jazz	108.Primus
29.Jazz+Funk	75.Polka	109.Porn Groove
30.Fusion	76.Retro	110.Satire
31.Trance	77.Musical	111.Slow Jam
32.Classical	78.Rock & Roll	112.Club
33.Instrumental	79.Hard Rock	113.Tango
34.Acid		114.Samba
35.House		115.Folklore
36.Game		116.Ballad
37.Sound Clip		117.Power Ballad
38.Gospel		118.Rhythmic Soul
39.Noise		119.Freestyle
40.AlternRock		120.Duet
41.Bass		121.Punk Rock
42.Soul		122.Drum Solo
43.Punk		123.Acapella
44.Space		124.Euro-House
45.Meditative		125.Dance Hall

- **9. This section has been omitted**