



Recommendation ITU-R BS.2088-0
(10/2015)

**Long-form file format for the
international exchange of audio
programme materials with metadata**

BS Series
Broadcasting service (sound)



International
Telecommunication
Union
Documents
Document Sharing Hub

Foreword

The role of the Radiocommunication Sector is to ensure the rational, equitable, efficient and economical use of the radio-frequency spectrum by all radiocommunication services, including satellite services, and carry out studies without limit of frequency range on the basis of which Recommendations are adopted.

The regulatory and policy functions of the Radiocommunication Sector are performed by World and Regional Radiocommunication Conferences and Radiocommunication Assemblies supported by Study Groups.

Policy on Intellectual Property Right (IPR)

ITU-R policy on IPR is described in the Common Patent Policy for ITU-T/ITU-R/ISO/IEC referenced in Annex 1 of Resolution ITU-R 1. Forms to be used for the submission of patent statements and licensing declarations by patent holders are available from <http://www.itu.int/ITU-R/go/patents/en> where the Guidelines for Implementation of the Common Patent Policy for ITU-T/ITU-R/ISO/IEC and the ITU-R patent information database can also be found.

Series of ITU-R Recommendations

(Also available online at <http://www.itu.int/publ/R-REC/en>)

Series	Title
BO	Satellite delivery
BR	Recording for production, archival and play-out; film for television
BS	Broadcasting service (sound)
BT	Broadcasting service (television)
F	Fixed service
M	Mobile, radiodetermination, amateur and related satellite services
P	Radiowave propagation
RA	Radio astronomy
RS	Remote sensing systems
S	Fixed-satellite service
SA	Space applications and meteorology
SF	Frequency sharing and coordination between fixed-satellite and fixed service systems
SM	Spectrum management
SNG	Satellite news gathering
TF	Time signals and frequency standards emissions
V	Vocabulary and related subjects

Note: This ITU-R Recommendation was approved in English under the procedure detailed in Resolution ITU-R 1.

Electronic Publication
Geneva, 2015

© ITU 2015

All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without written permission of ITU.

RECOMMENDATION ITU-R BS.2088-0

Long-form file format for the international exchange of audio programme materials with metadata

(2015)

Keywords

File, file format, metadata, wave, BW64, exchange, audio programme, WAV, BWF, RIFF, RF64, wave-file, Immersive

Scope

This Recommendation contains the specification of the BW64 (Broadcast Wave 64Bit) audio file format including the new chunks <ds64>, <axml> and <chna> which enable the file to carry large multichannel files and metadata including the Audio Definition Model (ADM) specified in Recommendation ITU-R BS.2076.

The ITU Radiocommunication Assembly,

considering

- a) that storage media based on Information Technology, including data disks and tapes, have penetrated all areas of audio production for radio broadcasting, namely non-linear editing, on-air play-out and archives;
- b) that this technology offers significant advantages in terms of operating flexibility, production flow and station automation and it is therefore attractive for the up-grading of existing studios and the design of new studio installations;
- c) that the adoption of a single file format for signal interchange would greatly simplify the interoperability of individual equipment and remote studios, it would facilitate the desirable integration of editing, on-air play-out and archiving;
- d) that a minimum set of broadcast related information must be included in the file to document the metadata related to the audio signal;
- e) that, to ensure the compatibility between applications with different complexity, a minimum set of functions, common to all the applications able to handle the recommended file format must be agreed;
- f) that Recommendation ITU-R BS.646 defines the digital audio format used in audio production for radio and television broadcasting;
- g) that the compatibility with currently available commercial file formats could minimize the industry efforts required to implement this format in the equipment;
- h) that a standard format for the coding history information and other related metadata would simplify the use of the information after programme exchange;
- i) that the quality of an audio signal is influenced by signal processing experienced by the signal, particularly by the use of non-linear coding and decoding during bit-rate reduction processes;
- j) that future audio systems will require metadata associated with the audio to be carried in the file;
- k) that future audio systems will use a variety of multichannel configurations including channel, object and scene-based audio such as specified in Recommendation ITU-R BS.2051;

l) that Recommendation ITU-R BS.1352 has limitations with respect to file size and its ability to carry additional metadata;

m) that multichannel audio files could potentially be larger than 4 Gbytes in size,

recommends

1 that, for the exchange of audio programmes, the audio signal parameters, sampling frequency (part 1), bit depth (part 4 and 5) and pre-emphasis (part 6) should be set in agreement with the relevant parts of Recommendation ITU-R BS.646;

2 that the file format specified in Annex 1 should be used for the interchange of audio programmes in the following use-cases:

- in WAVE-file based environments, where WAVE-file based broadcast applications wish to upgrade to handle immersive content, while maintaining forward compatibility;
- in file-based workflows where a mixed library of legacy WAVE-file based content and immersive content will exist;
- in file-based workflows, where a single package data plus metadata wrapper is preferred;

Annex 1 (normative)

Specification of the BW64 File Format

1 Introduction

The BW64 format is based on the WAVE audio file format (described in Annex 2), which is a type of file specified in the Resource Interchange File Format (RIFF). WAVE files specifically contain audio data. The basic building block of the RIFF file format, called a chunk, contains a group of tightly related pieces of information. It consists of a chunk identifier, an integer value representing the length in bytes and the information carried. A RIFF file is made up of a collection of chunks. This BW64 format uses the core elements of the format as described in EBU Tech 3306.

The BWF file format, Recommendation ITU-R BS.1352, has a number of limitations, most notably:

- Maximum file size of less than 4Gbytes.
- No support for advanced multichannel audio.
- Inadequate support for technical metadata.

The BW64 format described in this Recommendation aims to overcome these limitations, and maintain as much compatibility as possible with the Recommendation ITU-R BS.1352 format with many of the core elements shared.

There is an increasing demand on the transfer of metadata, especially the transfer of Audio Definition Model (ADM) metadata according to Recommendation ITU-R BS.2076. This Recommendation includes a definition of the <axml> chunk for storing and transferring metadata as XML.

The primary purpose of the <chna> chunk described in this Recommendation is to provide the references from each track in a BW64 file to the IDs in the ADM metadata defined in Recommendation ITU-R BS.2076.

Apart from the primary purpose of linking each track in the file with its associated ADM metadata, the <chna> chunk also allows faster access to ADM IDs without having to gain access the XML metadata (if the IDs are within a range of values predefined for standard ADM configurations). As the <chna> chunk can be fixed in size, and is placed before the <data> and <axml> chunks, it is easier to access, generate or modify its contents on the fly.

Data types throughout this document are used in accordance with Annex 3.

2 BW64 format description

2.1 Contents of a BW64 format file

A BW64 format file should start with the mandatory “WAVE” header and at least the following chunks:

```
<WAVE-form> ->
    BW64 ( 'WAVE'
        <ds64-ck>          // ds64 chunk for 64-bit addressing
        <fmt-ck>           // Format of the audio signal: PCM/non-PCM
        <chna-ck>          // chna chunk for ADM look-up
        <axml-ck>          // axml chunk for carrying XML metadata
        <wave-data>)      // sound data
```

NOTE 1 – Additional chunks may be present in the file. Some of these may be outside the scope of this Recommendation. Applications may or may not interpret or make use of these chunks, so the integrity of the data contained in such unknown chunks cannot be guaranteed. However, compliant applications shall pass on unknown chunks transparently.

NOTE 2 – It would be permissible to place the <axml> chunk after the <data> chunk, as the XML metadata will likely to be of an unknown length and a known starting position of the audio samples in the file might be more practical.

2.2 Existing chunks defined as part of the RIFF/WAVE standard

The RIFF/WAVE standard uses a number of chunks that are already defined. These are:

- <RIFF>
- <fmt>
- <data>

These chunks are described in § 2.4.1.

The RIFF/WAVE is a subset of the ITU-R BS.1352 format. Recommendation ITU-R BS.1352 contains these additional chunks:

- <bext>
- <ubxt>

These chunks will not be included in the BW64 format, which provides a more flexible solution carrying broadcast metadata.

2.3 New Chunks and Structs in the BW64 format

The new chunks introduced for BW64 are:

- <BW64>

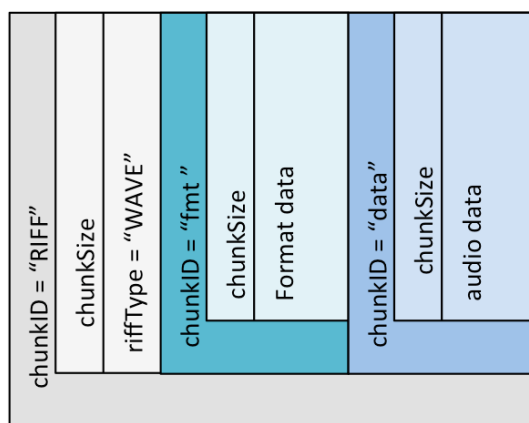
- <ds64>
- <JUNK>
- <axml>
- <chna>

These chunks are described in §§ 3 to 6.

2.4 Using the <ds64> chunk to enable the use of files greater than 4 Gbyte in size

The reason for the 4 Gbyte barrier is the 32-bit addressing in RIFF/WAVE and BWF. With 32 bits a maximum of 4294967296 bytes = 4 Gbyte can be addressed. To solve this issue, 64-bit addressing is needed. The structure of a basic conventional RIFF/WAVE file is shown in Fig. 1, where the chunkSize fields are 32-bit numbers representing the sizes of their chunks.

FIGURE 1
Basic RIFF/WAVE file structure



Just changing the size of every field in a BWF to 64-bit would produce a file that is not compatible with the standard RIFF/WAVE format – an obvious but important observation.

The approach adopted is to define a new 64-bit based RIFF called BW64 that is identical to the original RIFF/WAVE format, except for the following changes:

- The ID 'BW64' is used instead of 'RIFF' in the first four bytes of the file
- A mandatory <ds64> (data size 64) chunk is added, which has to be the first chunk after the "BW64 chunk".

The 'ds64' chunk has two mandatory 64-bit integer values, which replace two 32-bit fields of the RIFF/WAVE format:

- bw64Size (replaces the size field of the <RIFF> chunk)
- dataSize (replaces the size field of the <data> chunk)

For all two 32-bit fields of the RIFF/WAVE format the following rule applies:

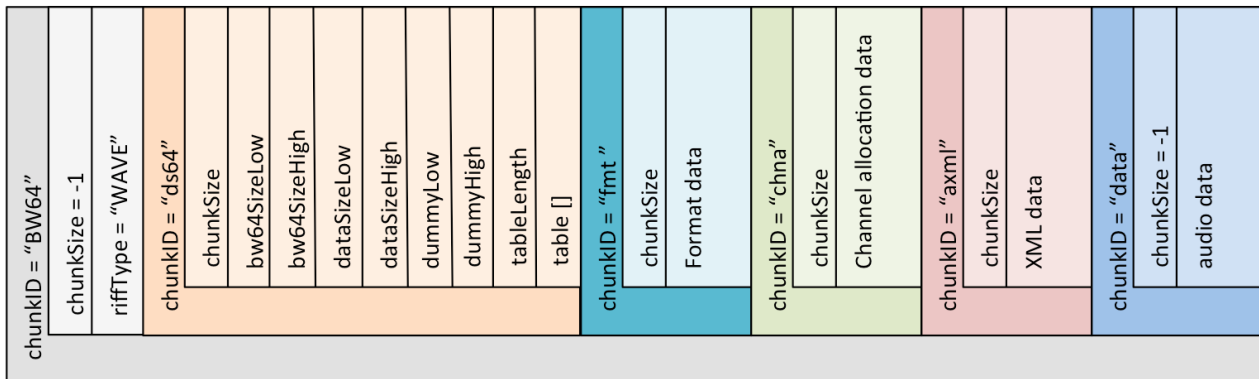
If the 32-bit value in the field is not "-1" (= FFFFFFFF hex) then this 32-bit value is used.

If the 32-bit value in the field is "-1" the 64-bit value in the 'ds64' chunk is used instead.

- One optional array of structs (see Annex A) with additional 64-bit chunk sizes is possible

The complete structure of the BW64 file format is illustrated in Fig. 2, where the chunkSize values for the <BW64> and <data> chunks are set to -1, to allow them to use 64-bit size values from the <ds64> chunk.

FIGURE 2
BW64 file structure



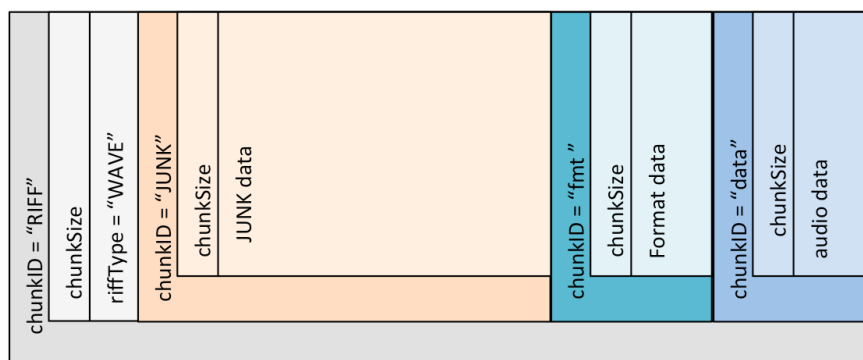
2.5 Achieving compatibility between RIFF/WAVE and BW64

In spite of higher sampling frequencies and multi-channel audio, some production audio files will inevitably be smaller than 4 Gbyte and they should therefore stay in the short-form RIFF/WAVE format (as described in Annex 2). The problem arises that a recording application cannot know in advance whether the recorded audio it is compiling will exceed 4 Gbyte or not at end of recording (i.e. whether it needs to use BW64 or not).

The solution is to enable the recording application to switch from RIFF/WAVE to BW64 on the fly at the 4 Gbyte size limit, while the recording is still going on.

This is achieved by reserving additional space in the RIFF/WAVE by inserting a <JUNK> chunk that is of the same size as a <ds64> chunk. This reserved space has no meaning for short-form WAVE, but will become the <ds64> chunk, if a transition to BW64 is necessary. The diagram in Fig. 3 shows the <JUNK> placeholder chunk placed before the <fmt> chunk.

FIGURE 3
File structure with JUNK chunk



At the beginning of a recording, a BW64-aware application will create a standard RIFF/WAVE with a 'JUNK' chunk as the first chunk. While recording, it will check the RIFF and data sizes. If they exceed 4 Gbyte, the application will:

- Replace the chunkID <JUNK> with <ds64> chunk. (This transforms the previous <JUNK> chunk into a <ds64> chunk).
- Insert the RIFF size, 'data' chunk size and sample count in the <ds64> chunk
- Set RIFF size, 'data' chunk size and sample count in the 32 bit fields to -1 = FFFFFFFF hex
- Replaces the ID 'RIFF' with 'BW64' in the first four bytes of the file
- Continue with the recording.

2.6 Existing Chunks and Structs in the RIFF/WAVE format

The chunks that exist in the RIF/WAVE format as shown below:

```
struct RiffChunk          // declare RiffChunk structure
{
    CHAR    chunkId[4];    // 'RIFF'
    DWORD   chunkSize;     // 4 byte size of the traditional RIFF/WAVE file
    CHAR    riffType[4];   // 'WAVE'
};

struct FormatChunk        // declare FormatChunk structure
{
    CHAR    chunkId[4];    // 'fmt '
    DWORD   chunkSize;     // 4 byte size of the 'fmt ' chunk
    WORD    formatTag;     // WAVE_FORMAT_PCM = 0x0001, etc.
    WORD    channelCount;  // 1 = mono, 2 = stereo, etc.
    DWORD   sampleRate;    // 32000, 44100, 48000, etc.
    DWORD   bytesPerSecond; // only important for compressed formats
    WORD    blockAlignment; // container size (in bytes) of one set of samples
    WORD    bitsPerSample;  // valid bits per sample 16, 20 or 24
    WORD    cbSize;        // extra information (after cbSize) to store
                        // should be set to zero as extraData is not used
    CHAR    extraData[22]; // extra data of WAVE_FORMAT_EXTENSIBLE when necessary,
                        // should not be used as cbSize will be zero.
};

struct DataChunk          // declare DataChunk structure
{
    CHAR    chunkId[4];    // 'data'
    DWORD   chunkSize;     // 4 byte size of the 'data' chunk
    CHAR    waveData[ ];   // audio samples
};
```

The empty array brackets indicate a variable number of elements can be used (including zero).

2.6.1 Elements of the <RIFF> chunk

The <RIFF> chunk is the top level for the file.