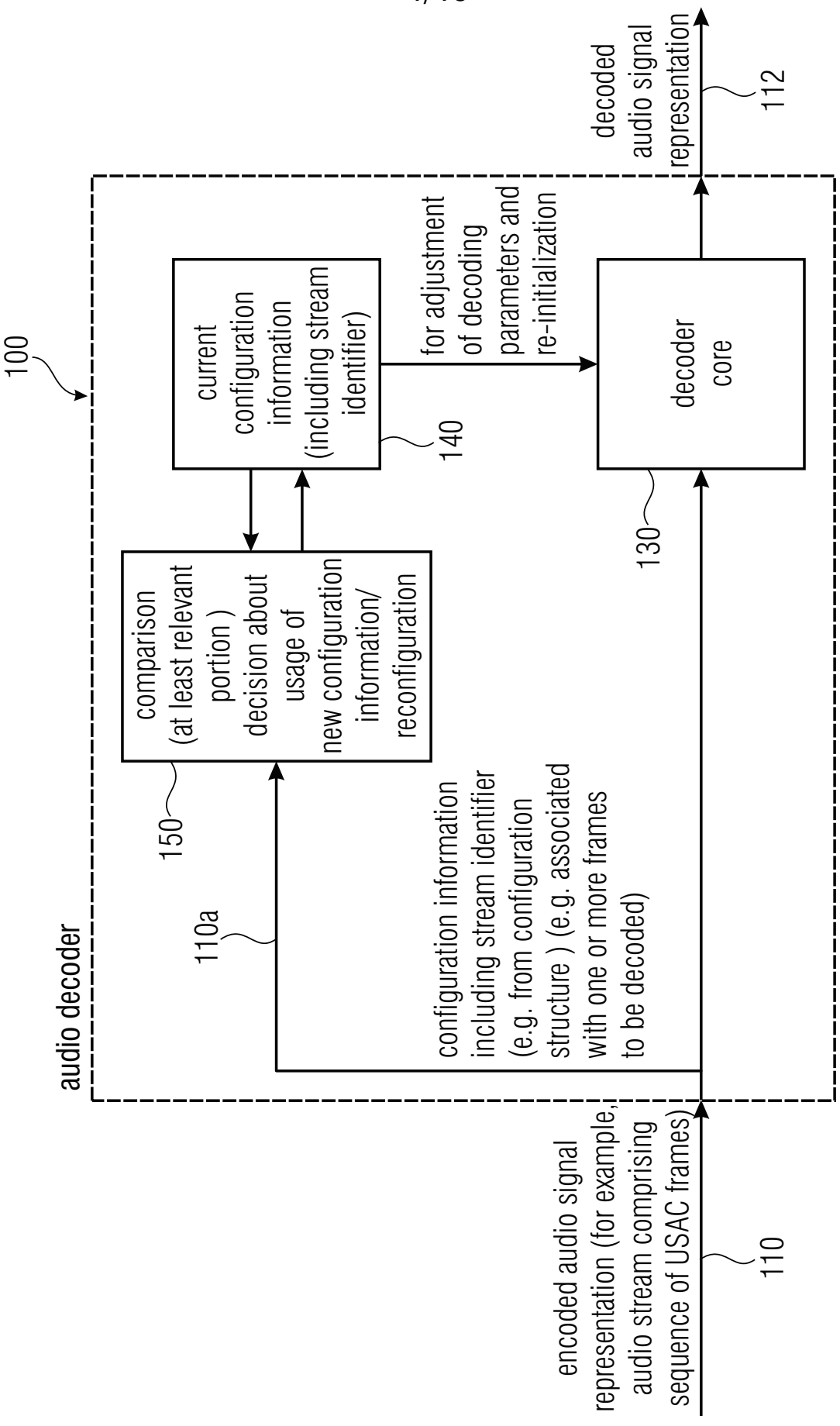

100
audio decoder
encoded audio signal representation (for example, audio stream comprising sequence of USAC frames)
110
decoded audio signal representation
112
current configuration information (including stream identifier)
140
for adjustment of decoding parameters and re-initialization
decoder core
130
comparison (at least relevant portion) decision about usage of new configuration information/ reconfiguration
150
110a
configuration information including stream identifier (e.g. from configuration structure) (e.g. associated with one or more frames to be decoded)

Fig. 1

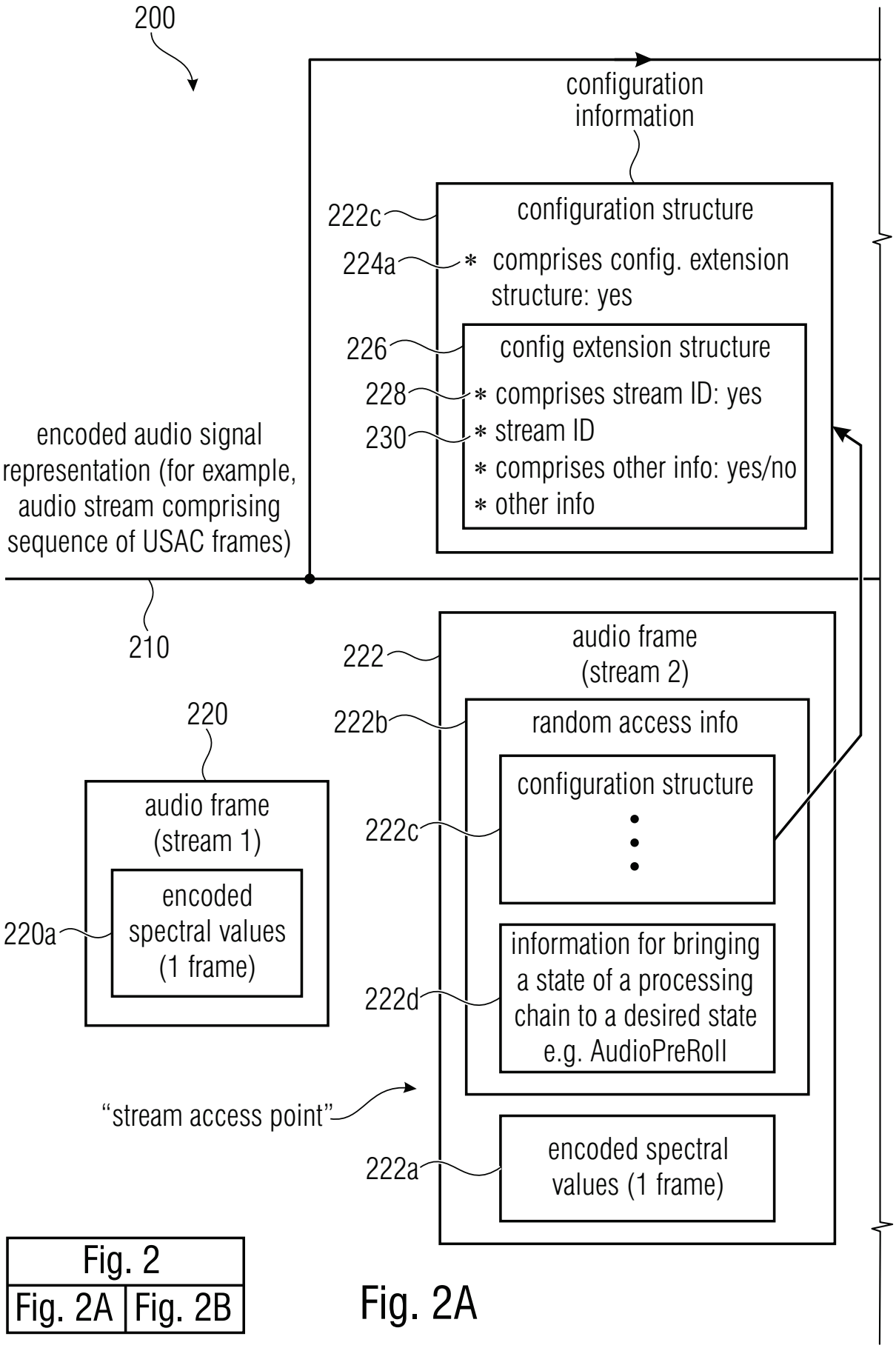


Fig. 2A

Fig. 2	
Fig. 2A	Fig. 2B

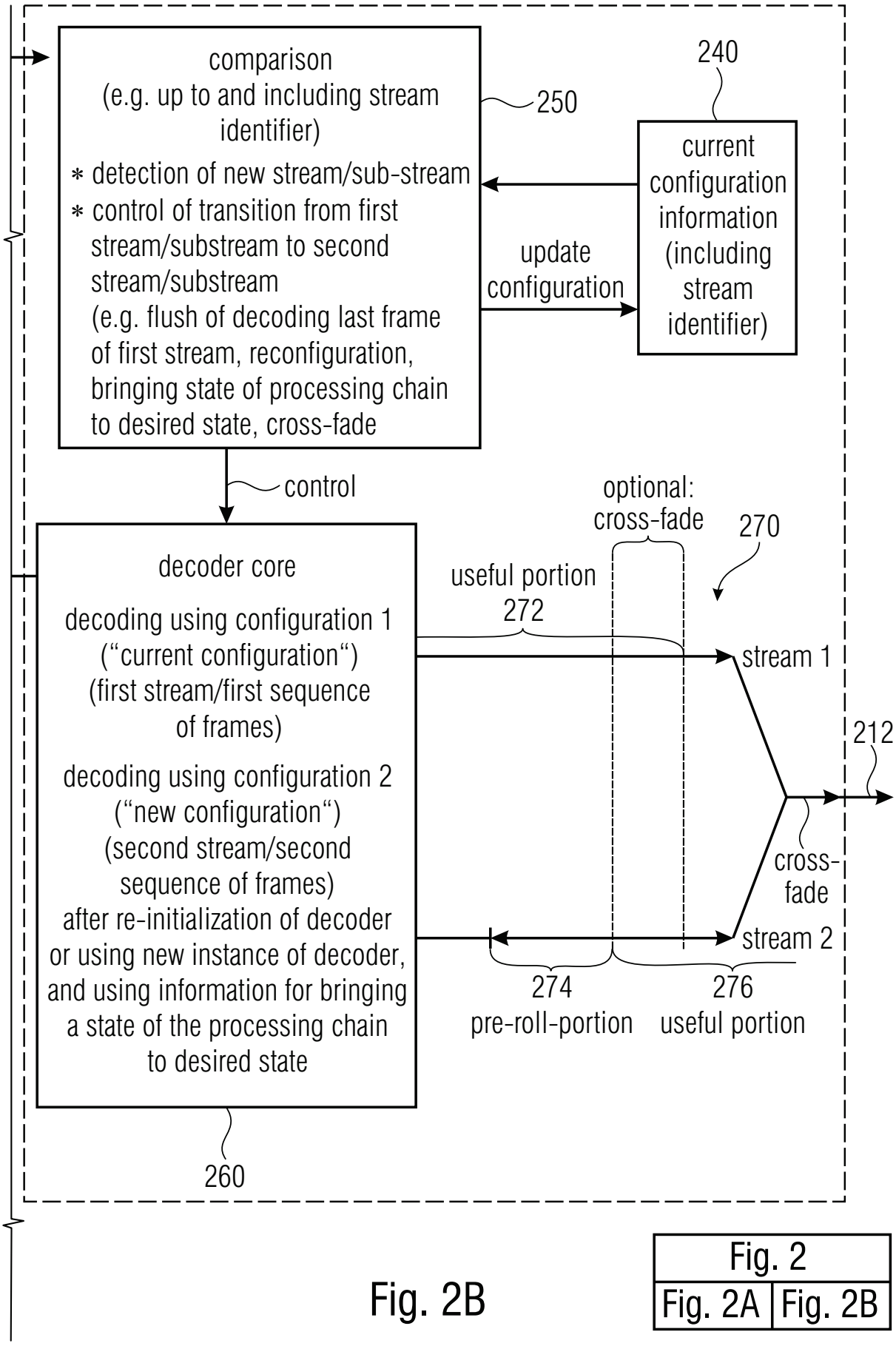


Fig. 2B

Fig. 2	
Fig. 2A	Fig. 2B

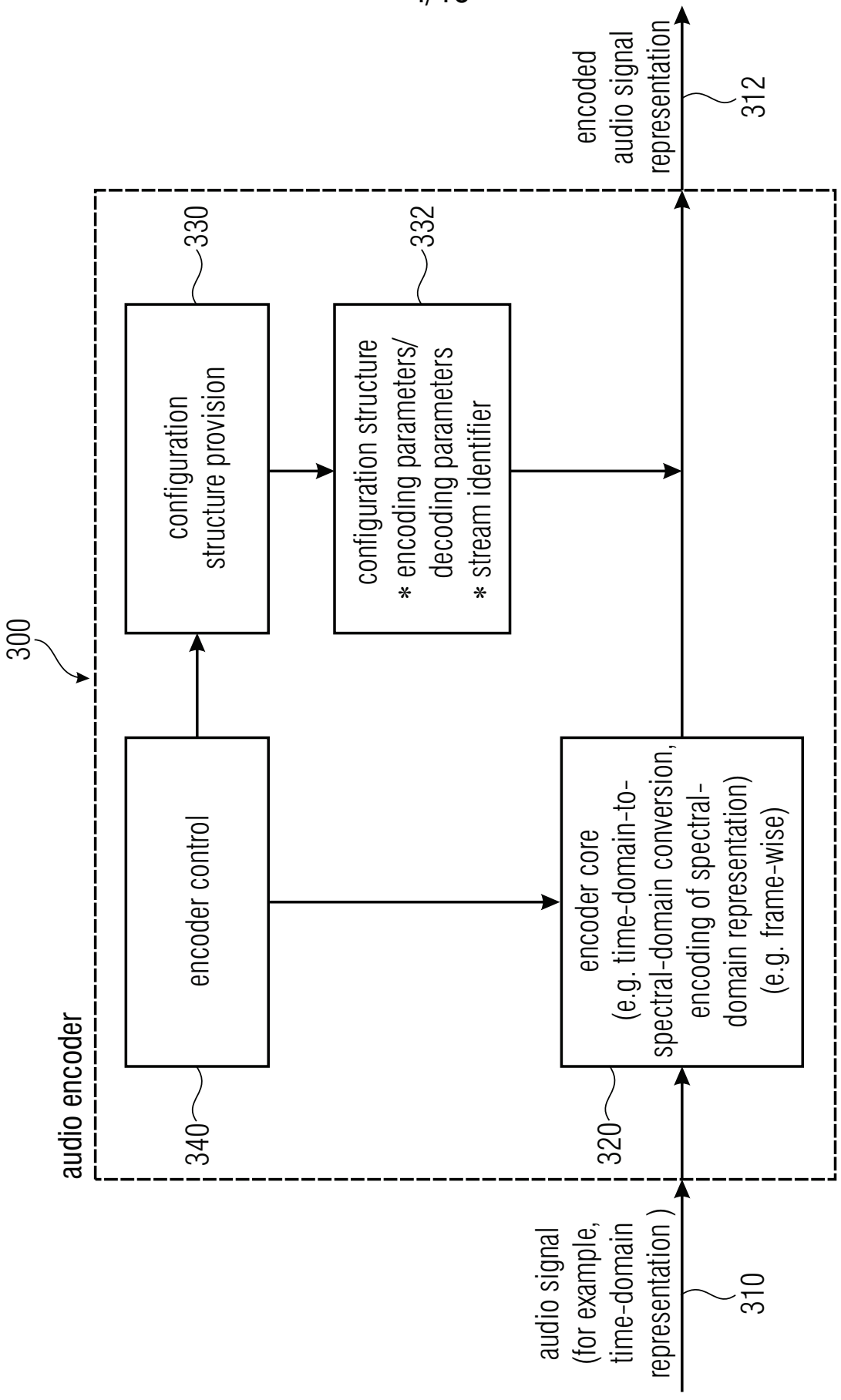

audio encoder
300
encoder control
340
configuration structure provision
330
configuration structure
* encoding parameters/ decoding parameters
* stream identifier
332
encoder core (e.g. time-domain-to-spectral-domain conversion, encoding of spectral-domain representation) (e.g. frame-wise)
320
audio signal (for example, time-domain representation)
310
encoded audio signal representation
312

Fig. 3

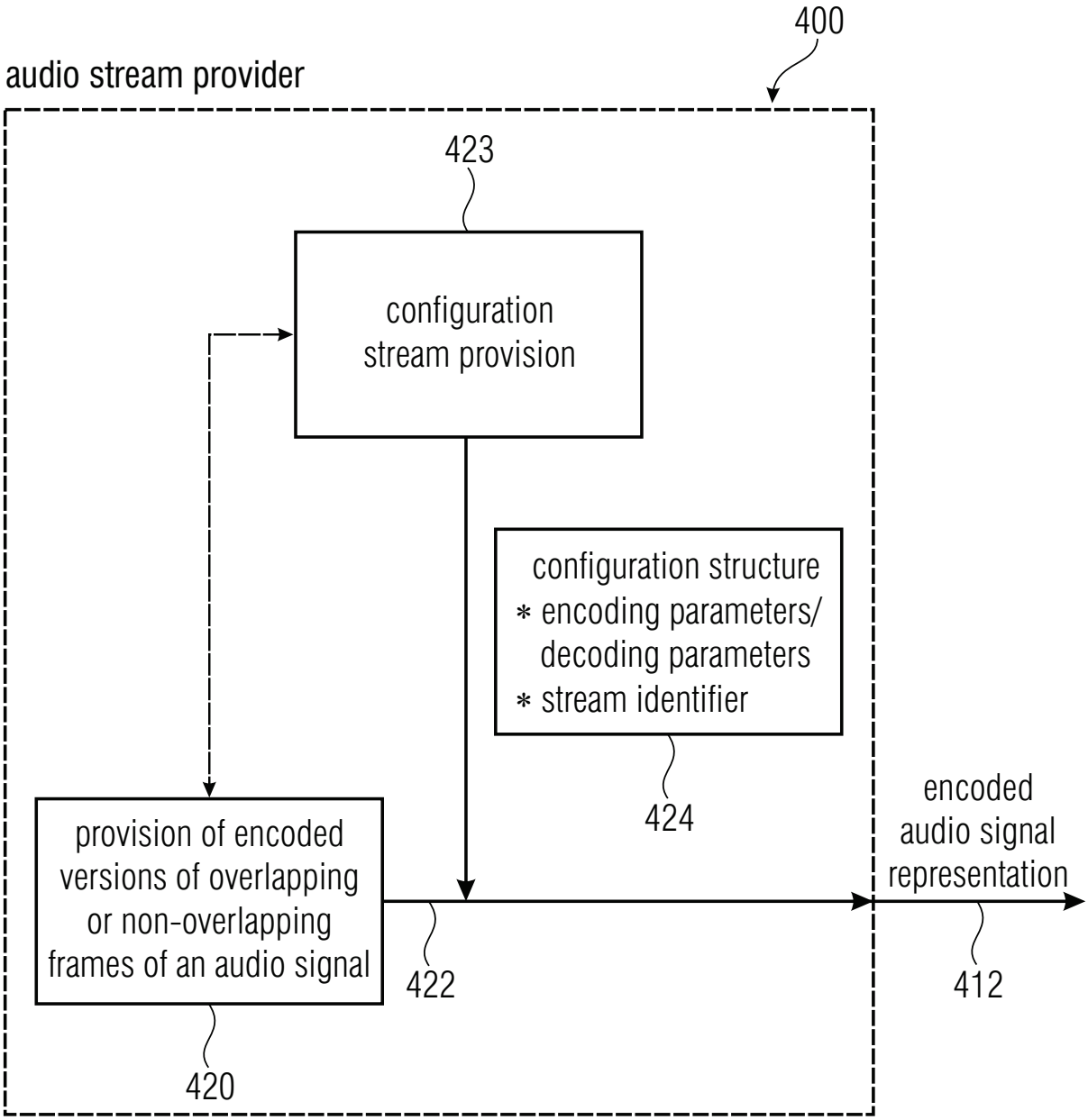

400
audio stream provider
423
configuration
stream provision
configuration structure
* encoding parameters/
decoding parameters
* stream identifier
424
provision of encoded
versions of overlapping
or non-overlapping
frames of an audio signal
422
420
encoded
audio signal
representation
412

Fig. 4

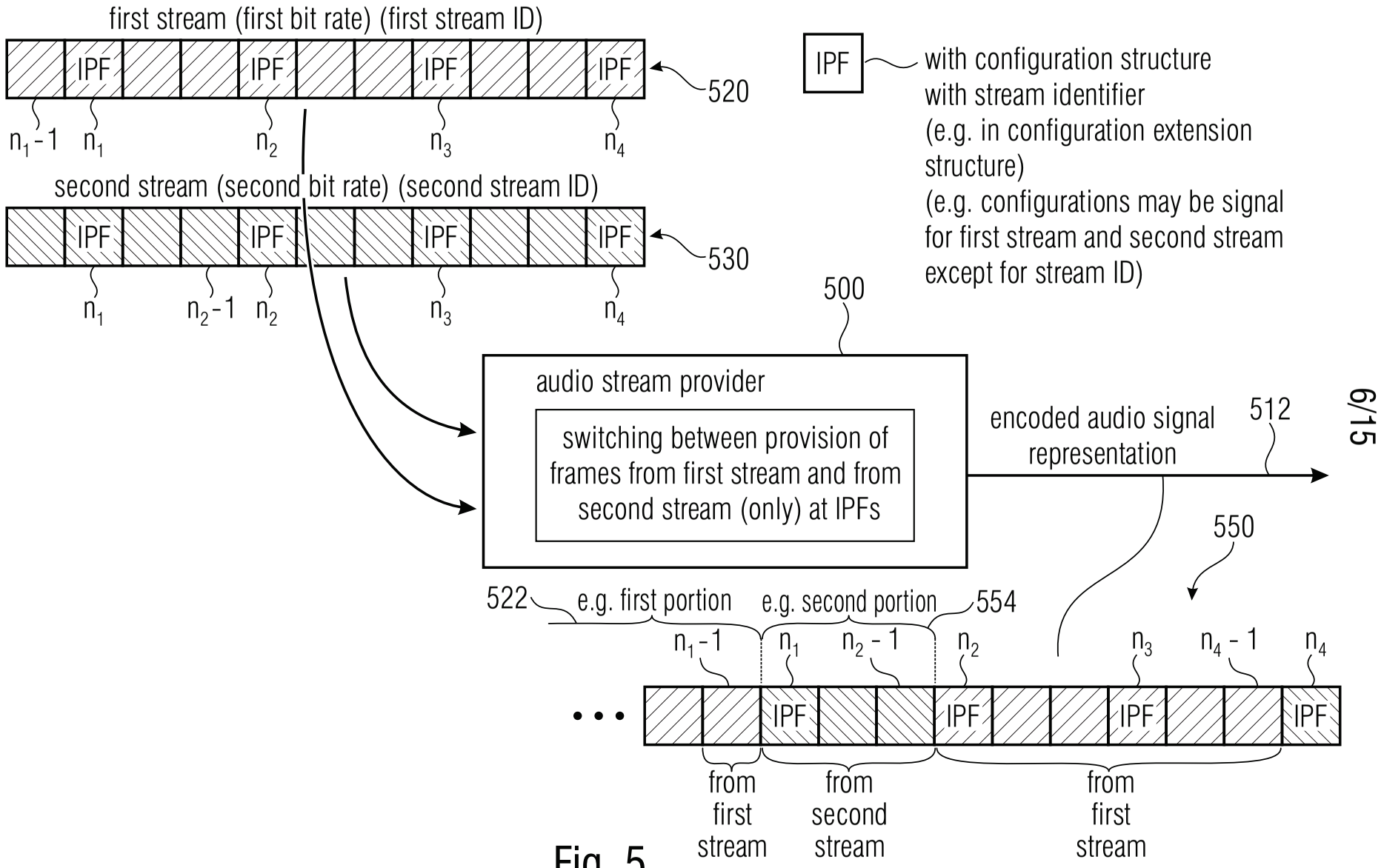

first stream (first bit rate) (first stream ID)
IPF
IPF
IPF
IPF
520
$n_1 - 1$
n_1
n_2
n_3
n_4
second stream (second bit rate) (second stream ID)
IPF
IPF
IPF
IPF
530
n_1
$n_2 - 1$
n_2
n_3
n_4
IPF
with configuration structure
with stream identifier
(e.g. in configuration extension structure)
(e.g. configurations may be signal for first stream and second stream except for stream ID)
500
audio stream provider
switching between provision of frames from first stream and from second stream (only) at IPFs
encoded audio signal representation
512
550
522
e.g. first portion
e.g. second portion
554
$n_1 - 1$
n_1
$n_2 - 1$
n_2
n_3
$n_4 - 1$
n_4
IPF
IPF
IPF
IPF
from first stream
from second stream
from first stream

Fig. 5

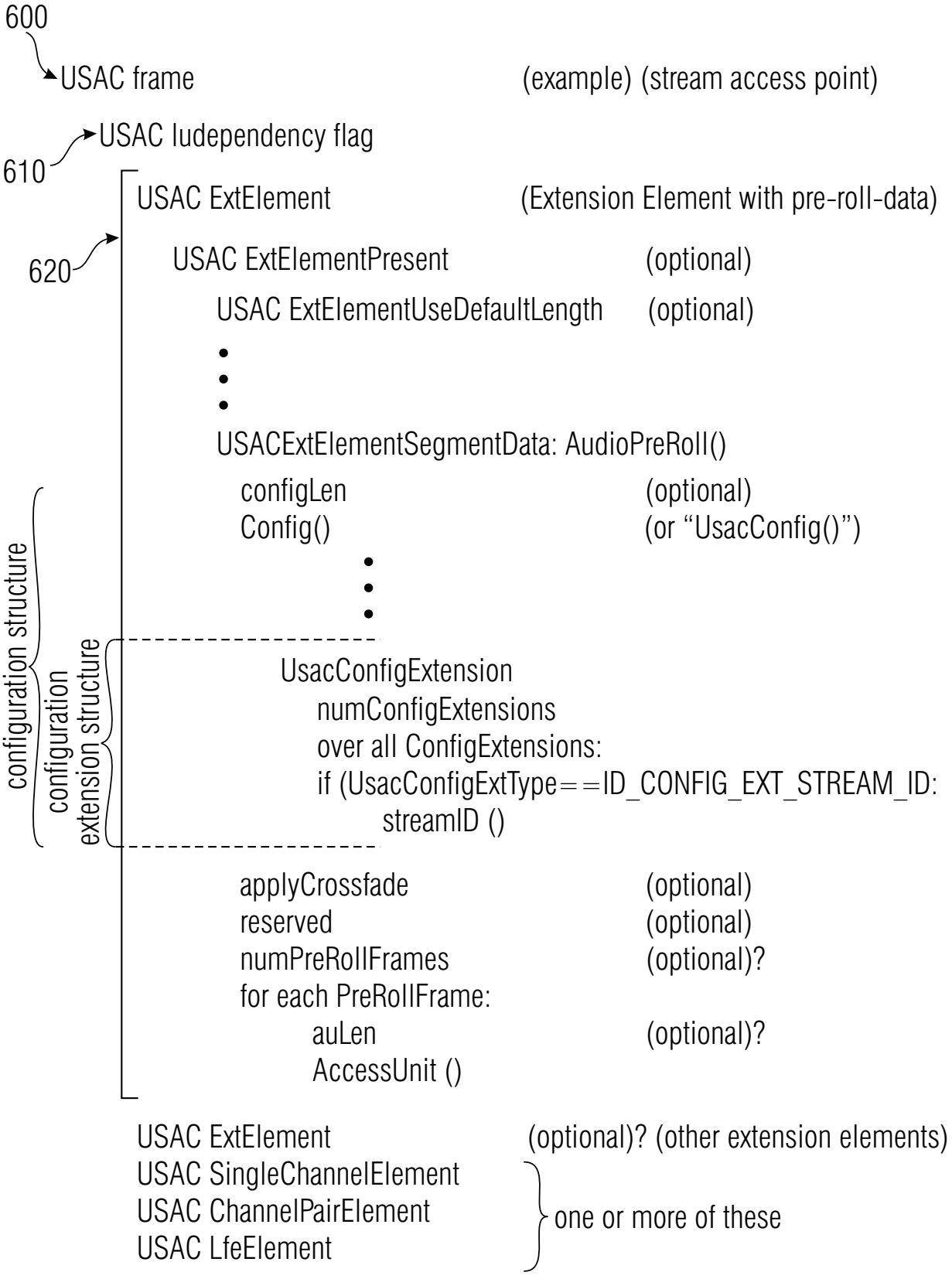

600
USAC frame (example) (stream access point)
USAC Iudependency flag
610
USAC ExtElement (Extension Element with pre-roll-data)
620
USAC ExtElementPresent (optional)
USAC ExtElementUseDefaultLength (optional)
USACExtElementSegmentData: AudioPreRoll()
configLen (optional)
Config() (or "UsacConfig()")
configuration structure
configuration extension structure
UsacConfigExtension
numConfigExtensions
over all ConfigExtensions:
if (UsacConfigExtType==ID_CONFIG_EXT_STREAM_ID:
streamID ()
applyCrossfade (optional)
reserved (optional)
numPreRollFrames (optional)?
for each PreRollFrame:
auLen (optional)?
AccessUnit ()
USAC ExtElement (optional)? (other extension elements)
USAC SingleChannelElement
USAC ChannelPairElement
USAC LfeElement
one or more of these

Fig. 6

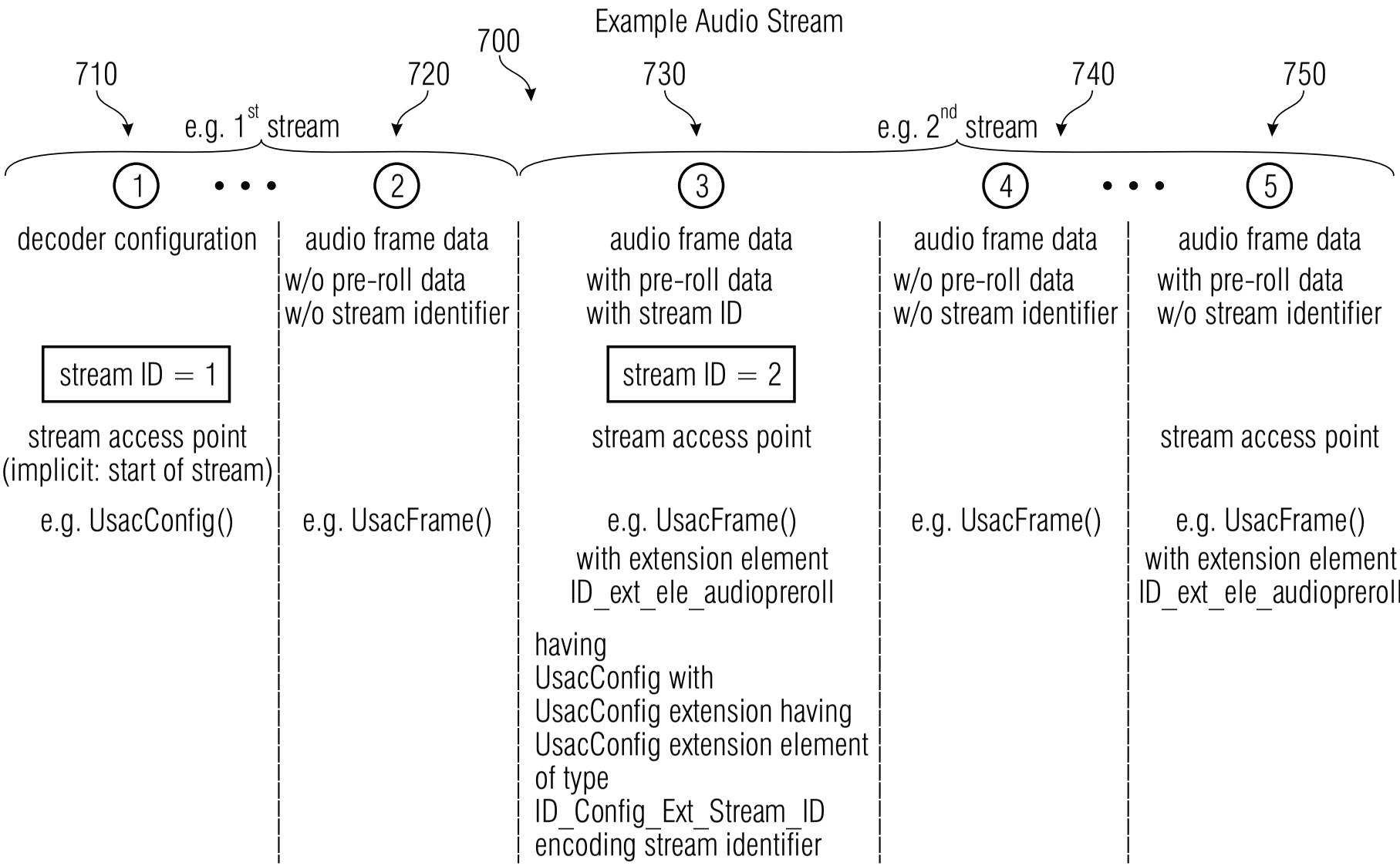

Example Audio Stream
700
710
720
730
740
750
e.g. 1st stream
e.g. 2nd stream
1
2
3
4
5
decoder configuration
audio frame data
w/o pre-roll data
w/o stream identifier
audio frame data
with pre-roll data
with stream ID
audio frame data
w/o pre-roll data
w/o stream identifier
audio frame data
with pre-roll data
w/o stream identifier
stream ID = 1
stream ID = 2
stream access point
(implicit: start of stream)
stream access point
stream access point
e.g. UsacConfig()
e.g. UsacFrame()
e.g. UsacFrame()
with extension element
ID_ext_ele_audiopreroll
having
UsacConfig with
UsacConfig extension having
UsacConfig extension element
of type
ID_Config_Ext_Stream_ID
encoding stream identifier
e.g. UsacFrame()
e.g. UsacFrame()
with extension element
ID_ext_ele_audiopreroll

Fig. 7

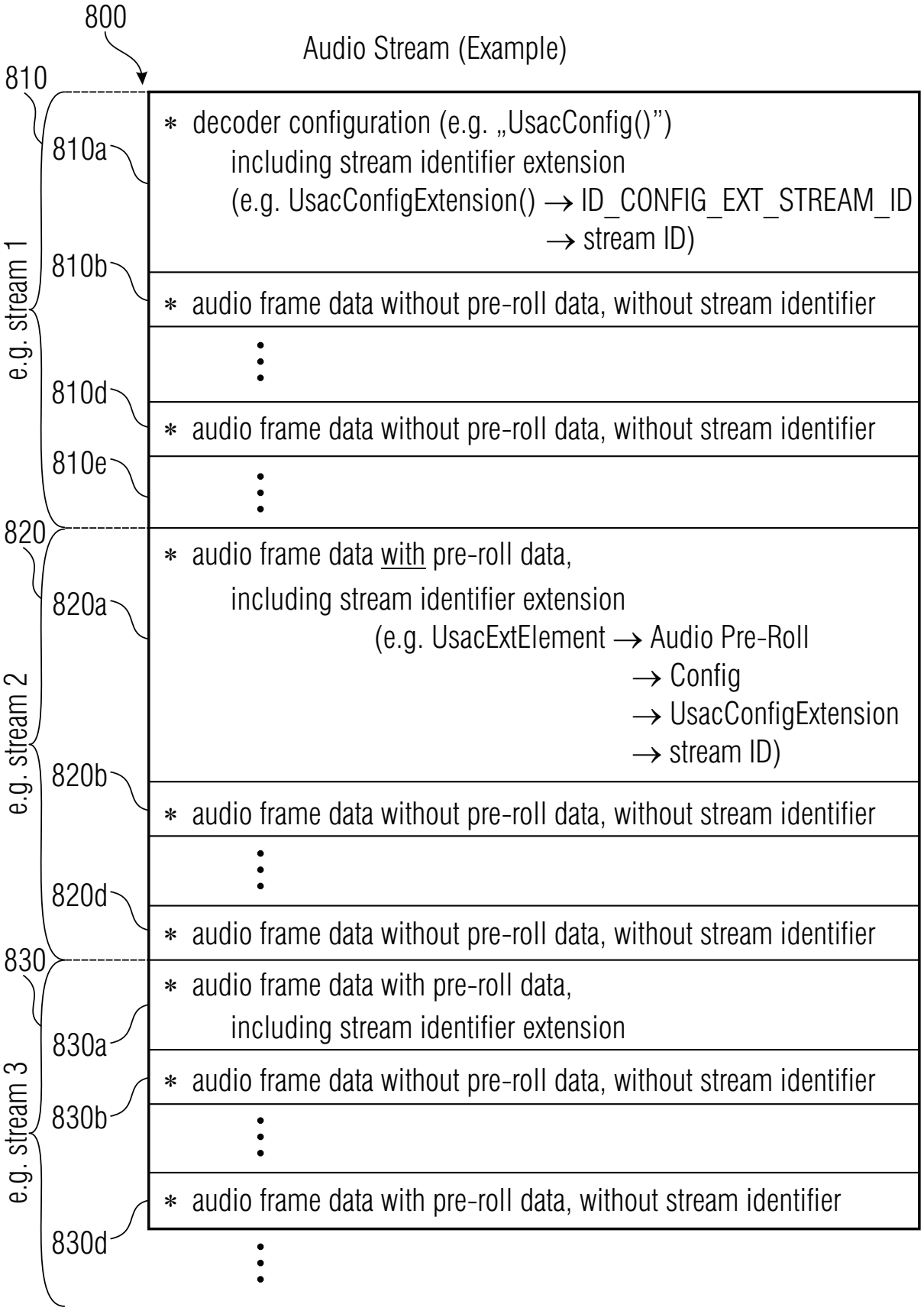

800
Audio Stream (Example)
810
810a
* decoder configuration (e.g. „UsacConfig()")
including stream identifier extension
(e.g. UsacConfigExtension() → ID_CONFIG_EXT_STREAM_ID
→ stream ID)
810b
* audio frame data without pre-roll data, without stream identifier
810d
* audio frame data without pre-roll data, without stream identifier
810e
e.g. stream 1
820
820a
* audio frame data with pre-roll data,
including stream identifier extension
(e.g. UsacExtElement → Audio Pre-Roll
→ Config
→ UsacConfigExtension
→ stream ID)
820b
* audio frame data without pre-roll data, without stream identifier
820d
* audio frame data without pre-roll data, without stream identifier
e.g. stream 2
830
830a
* audio frame data with pre-roll data,
including stream identifier extension
830b
* audio frame data without pre-roll data, without stream identifier
830d
* audio frame data with pre-roll data, without stream identifier
e.g. stream 3

Fig. 8

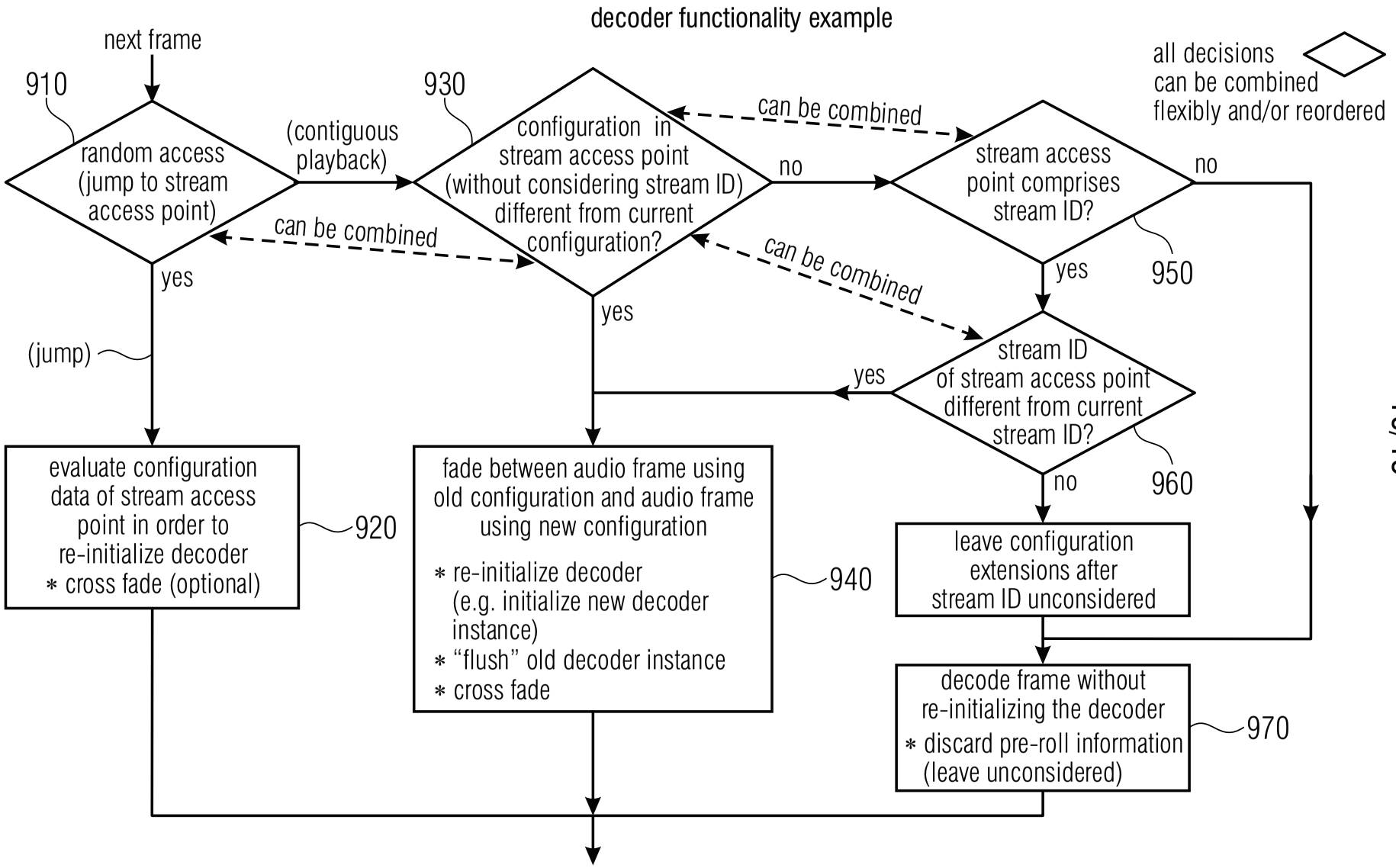

decoder functionality example
next frame
910
random access (jump to stream access point)
(contiguous playback)
930
configuration in stream access point (without considering stream ID) different from current configuration?
can be combined
no
stream access point comprises stream ID?
no
950
yes
can be combined
can be combined
yes
yes
(jump)
stream ID of stream access point different from current stream ID?
yes
960
no
evaluate configuration data of stream access point in order to re-initialize decoder
* cross fade (optional)
920
fade between audio frame using old configuration and audio frame using new configuration
* re-initialize decoder (e.g. initialize new decoder instance)
* "flush" old decoder instance
* cross fade
940
leave configuration extensions after stream ID unconsidered
decode frame without re-initializing the decoder
* discard pre-roll information (leave unconsidered)
970
all decisions can be combined flexibly and/or reordered

Fig. 9

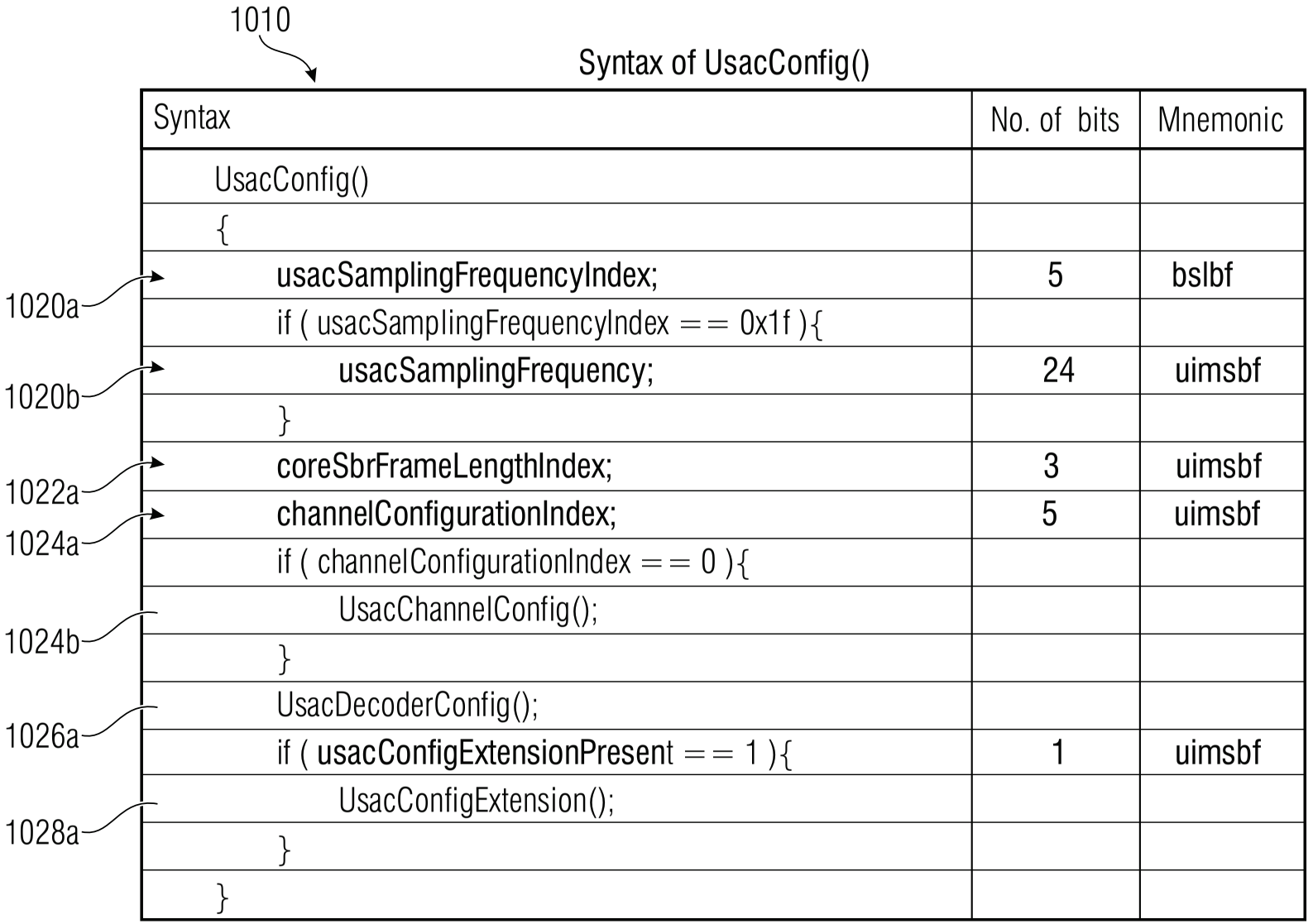

1010
Syntax of UsacConfig()
Syntax
No. of bits
Mnemonic
UsacConfig()
{
usacSamplingFrequencyIndex;
5
bslbf
if (usacSamplingFrequencyIndex == 0x1f){
usacSamplingFrequency;
24
uimsbf
}
coreSbrFrameLengthIndex;
3
uimsbf
channelConfigurationIndex;
5
uimsbf
if (channelConfigurationIndex == 0){
UsacChannelConfig();
}
UsacDecoderConfig();
if (usacConfigExtensionPresent == 1){
1
uimsbf
UsacConfigExtension();
}
}
1020a
1020b
1022a
1024a
1024b
1026a
1028a
can be modified

Fig. 10a

1030

Syntax of UsacConfigExtension()

Syntax	No. of bits	Mne-monic	
UsacConfigExtension()			
{			
1040a numConfigExtensions = escapedValue(2,4,8) + 1;			different encoding possible
1042a for (confExtIdx=0; confExtIdx<numConfigExtensions; confExtIdx++) {			
usacConfigExtType[confExtIdx] =			different encoding possible
1044a escapedValue(4,8,16);			
usacConfigExtLength[confExtIdx] =			
escapedValue(4,8,16);			
switch (usacConfigExtType[confExtIdx]) {			
1046a case ID_CONFIG_EXT_FILL:			
while (usacConfigExtLength[confExtIdx]--) {			optional
fill_byte[i]; /* should be '10100101' */	8	uimsbf	
}			
break;			
1048a case ID_CONFIG_EXT_LOUDNESS_INFO:			optional
loudnessInfoSet()			
break;			
1050a case ID_CONFIG_EXT_STREAM_ID:			
streamId();			
break;			
default:			
while (usacConfigExtLength[confExtIdx]--) {			optional
tmp;	8	uimsbf	
}			
break;			
}			
}			
}			

Fig. 10b

Syntax of StreamId()

Syntax	No. of bits	Mnemonic
StreamId()		
{		
streamIdentifier	16	Uimsbf
}		

Fig. 10c

usacConfigExtType	Value
ID_CONFIG_EXT_FILL	0
/* reserved for ISO use */	1
ID_CONFIG_EXT_LOUDNESS_INFO	2
/* reserved for ISO use */	3...6
ID_CONFIG_EXT_STREAM_ID	7
/* reserved for ISO use */	8-127
/* reserved for use outside of ISO scope */	128 and higher

Fig. 10d

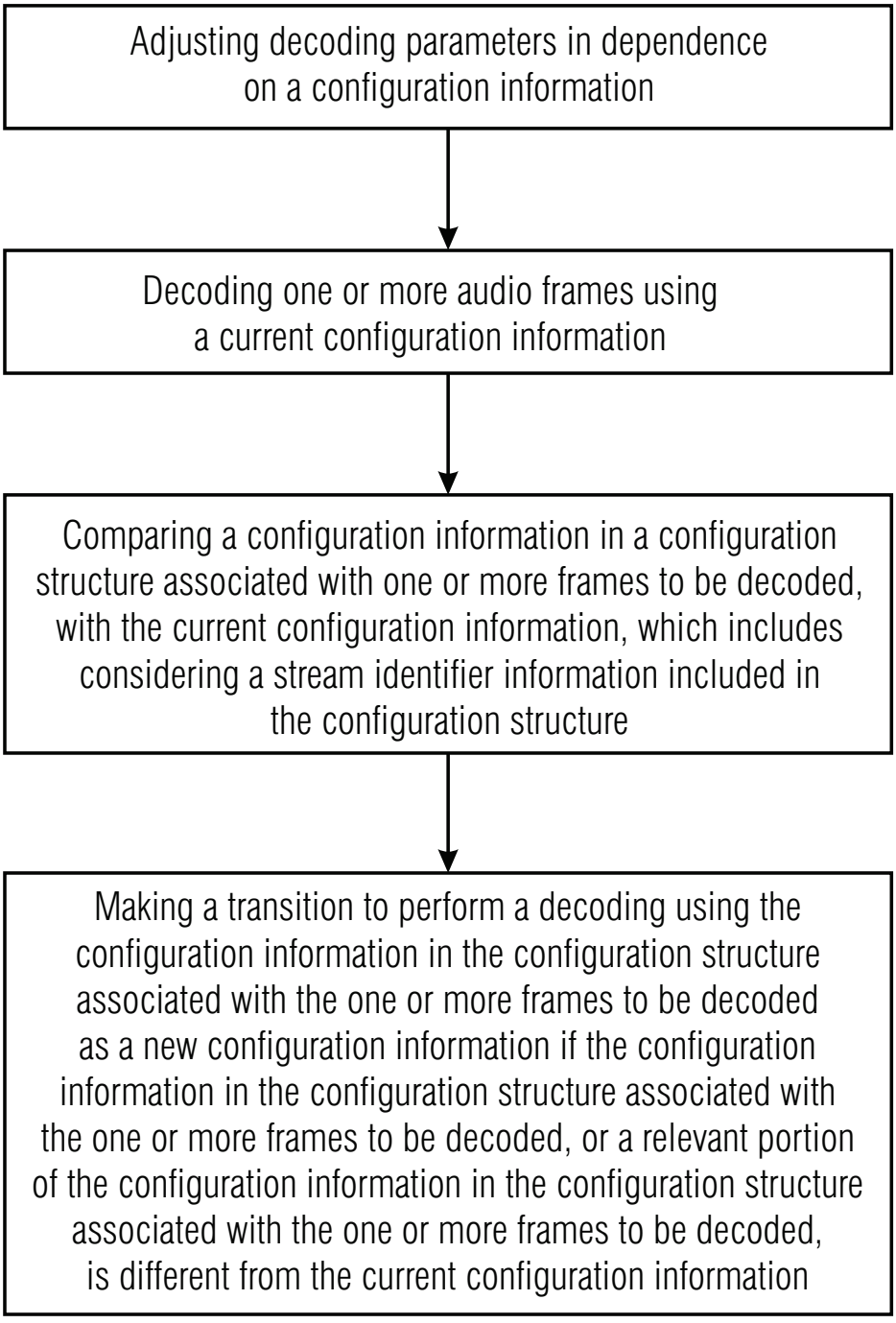

Adjusting decoding parameters in dependence on a configuration information
Decoding one or more audio frames using a current configuration information
Comparing a configuration information in a configuration structure associated with one or more frames to be decoded, with the current configuration information, which includes considering a stream identifier information included in the configuration structure
Making a transition to perform a decoding using the configuration information in the configuration structure associated with the one or more frames to be decoded as a new configuration information if the configuration information in the configuration structure associated with the one or more frames to be decoded, or a relevant portion of the configuration information in the configuration structure associated with the one or more frames to be decoded, is different from the current configuration information

Fig. 11a

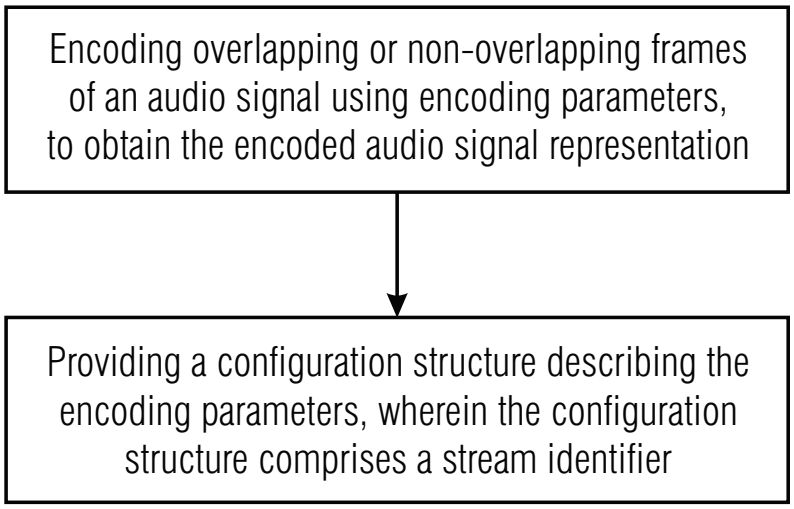


Fig. 11b

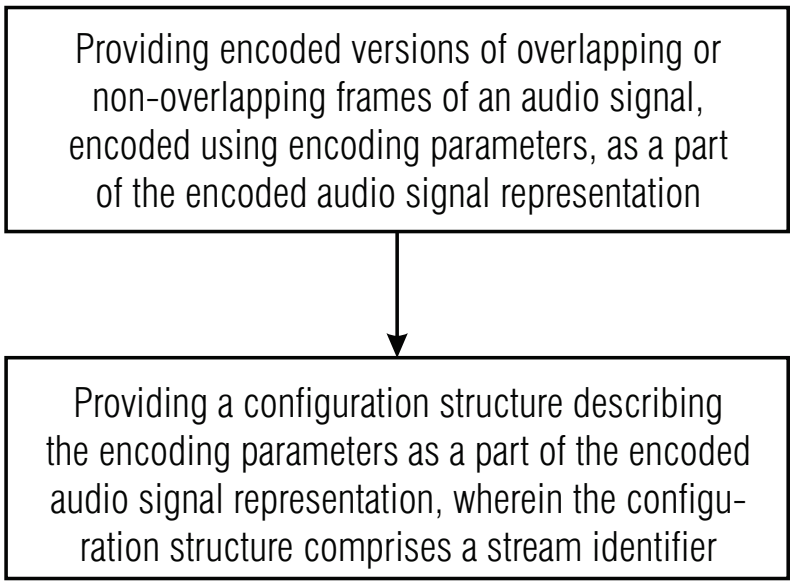


Fig. 11c