User's Manual

Fraunhofer IIS
Institut
Integrierte Schaltungen

RENESAS

MP3 Decode Middleware

User's Manual

Rev.1.01 August, 2014

Renesas Electronics
www.renesas.com

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The recommended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

   “Standard”: Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots etc.

   “High Quality”: Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; and safety equipment etc.

   Renesas Electronics products are neither intended nor authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems, surgical implantations etc.), or may cause serious property damages (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

How to Use This Manual

1. Purpose and Target Readers

This manual is intended to give users of the middleware an understanding of the decoder functionality, performance, and usage of the middleware. It is targeted at people who wish to design application systems which use the middleware. It assumes readers hold general knowledge in the fields of audio technology, programming languages, and microcontrollers.

This manual is broadly divided into the following sections:

-Product overview
-Middleware specifications
-Library function specifications
-Usage precautions

Use this middleware after carefully reading the precautions. The precautions are stated in the main text of each section, at the end of each section, and in the usage precaution section.

The revision history summarizes major corrections and additions to the previous version. It does not cover all the changes. For details, refer to this manual.

2. Use of This Product

To use this product, you need to enter into a software license agreement with Renesas Electronics.

MPEG Layer III audio decoding technology licensed from Fraunhofer IIS and Thomson.

Licensing and patent issues on MPEG Layer III

The various patents claimed to cover MPEG Layer III by different patent-holders have many different expiration dates, Thomson S.A, Fraunhofer IIS, or any other companies claims to control the licensing of MPEG Layer III patents in many countries. If you require our MP3 Decode Middleware for your systems, you are responsible for obtaining this technology license from them. For information about Fraunhofer IIS and Thomson's patent portfolio and licensing terms, see their website. (http://www.mp3licensing.com) And please contact Thomson S.A for licensing guidance.

3. Related Documents

MPEG 1-related Documents

Specifications Numbers and Titles	Date of issuance
Audio specifications	
ISO/IEC 11172-3:1993/Cor.1:1996 Information technology -- Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s - Part 3: Audio	April 15, 1996
JIS X 4323:1996 Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s - Part 3: Audio	November 20, 1996
Test specifications	
ISO/IEC 11172-4:1995/Cor.1:2007 Information technology -- Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s -- Part 4: Compliance testing	January 10, 2007
Software specifications	
ISO/IEC TR 11172-5:1998/Cor.1:2007 Information technology -- Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s -- Part 5: Software simulation	September 25, 2007

MPEG 2-related Documents

Specifications Numbers and Titles	Date of issuance
Audio specifications	
ISO/IEC 13818-3:1998 Information technology -- Generic coding of moving pictures and associated audio information - Part 3: Audio	April 15, 1998
JIS X 4327:1998 Generic coding of moving pictures and associated audio information - Part 3: Audio	January 20, 1998
Test specifications	
ISO/IEC 13818-4:2004/Cor.2:2011 Information technology -- Generic coding of moving pictures and associated audio information -- Part4: Compliance testing	May 11, 2011
Software specifications	
ISO/IEC TR 13818-5:2005 Information technology -- Generic coding of moving pictures and associated audio information -- Part5: Software simulation	October 17, 2005

MPEG 2.5-related Documents

Titles	Date of issuance
Proprietary MPEG-2.5 specifications document	
MPEG-Layer3 Bitstream Syntax and Decoding (Fraunhofer IIS)	April 24, 2002
Software	
MPEG Layer-3 Decoder Library for Fixed-point Processors - Core Design Kit	April 17, 2007

Processor-related Documents

Refer to attached product manual.

4. List of Abbreviations and Acronyms

Abbreviation	Full Form
ANSI-C	American National Standards Institute - C
bps	bits per second
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DAC	digital to analog converter
FPU	Floating Point Unit
IEC	International Electrotechnical Commission
I/O	Input / Output
ISO	International Organization for Standardization
JIS	Japanese Industrial Standards
LSB	Least Significant Bit
LSF	Low Sampling Frequency
MPEG	Moving Picture Experts Group
MSB	Most Significant Bit
OS	Operating System
PCM	Pulse Code Modulation
RAM	Random Access Memory
RIFF	Resource Interchange File Format
ROM	Read Only Memory
TR	Technical Report

Table of Contents

1. Overview

This section provides an overview of the MP3 decoder.

1.1 Specifications Outline

MPEG Audio is an audio codec for coding or decoding an audio signal, and is a general term of MPEG-1 Audio Layer-1/2/3, MPEG-2 Audio Layer-1/2/3 LSF, and MPEG2.5 Audio Layer-3. This middleware, which meets these decoding systems, decodes compressed input data and outputs the decoding results. For these specifications, refer to Table 1.1. For basic specifications and performance, refer to attached product manual.

Table 1.1 Supported MP3 Specifications

Item	Description
Input data format	[1] MPEG-1 Audio Layer-1/2/3 (ISO/IEC 11172-3:1993) [2] MPEG-2 Audio Layer-1/2/3 (ISO/IEC 13818-3:1998/2nd) [3] MPEG2.5 Audio Layer-3 (developed by Fraunhofer Institute)
Output data format	16-bit linear PCM
Sampling frequency (Hz) supported	[1] 48000 / 44100 / 32000 [2] 24000 / 22050 / 16000 [3] 12000 / 11025 / 8000
Number of channels supported	1 channel (single channel) 2 channels (stereo / joint stereo / dual channel)
Bit rate (kbps) supported	Bit rate complies with each standard for input data. [1] Layer-1: 32 / 64 / 96 / 128 / 160 / 192 / 224 / 256 / 288 / 320 / 352 / 384 / 416 / 448 Layer-2: 32 / 48 / 56 / 64 / 80 / 96 / 112 / 128 / 160 / 192 / 224 / 256 / 320 / 384 Layer-3: 32 / 40 / 48 / 56 / 64 / 80 / 96 / 112 / 128 / 160 / 192 / 224 / 256 / 320 [2] Layer-1: 32 / 48 / 56 / 64 / 80 / 96 / 112 / 128 / 144 / 160 / 176 / 192 / 224 / 256 Layer-2/3: 8 / 16 / 24 / 32 / 40 / 48 / 56 / 64 / 80 / 96 / 112 / 128 / 144 / 160 [3] Layer-3: 8 / 16 / 24 / 32 / 40 / 48 / 56 / 64 / 80 / 96 / 112 / 128 / 144 / 160 [Note] Free format is not supported. [Note] MPEG-2.5 Layer -3 supports the same range of bit rates as MPEG-2 Layer-3.
CRC	Supported
Reentrant	Supported
Restrictions	This middleware does not support the following functionality: - Decode data in 3 or more channels. - Decode data in free format. - Parse ID3 tags, RIFF chunks, and MPEG-4 containers. - Decode frames including the "Xing" / "Info" character string. - Decode ancillary data. (This middleware skips the ancillary data.) - Implement de-emphasis. (This middleware returns the emphasis bit.)

Table 1.2 Memory Size Requirements

Memory type	Location	Memory area name		Size (in bytes)	
Instruction	ROM	Instruction area			
Data		Constant table area			---
		Other area(Depended on the compiler)			
	RAM	Middleware work area			23,712
		Area breakdown	Static area	Size breakdown	21,652
			Scratch area		2,060
		User work area			6,452
		Area breakdown	Input buffer	Size breakdown	1,728
			Output buffer		4,608
			Structure		116
		Stack area			4,096
		Other area(Depended on the compiler)			---

[Note] Area whose location is shown as ROM in the location column can be included in RAM or ROM.

[Note] Area whose location is shown as RAM in the location column can be included in RAM only.

[Note] For Instruction area, Constant table area, and Other area refer to attached product manual.

1.2 Configuration

Figure 1.1 shows an example of the decode system configuration which uses this middleware.

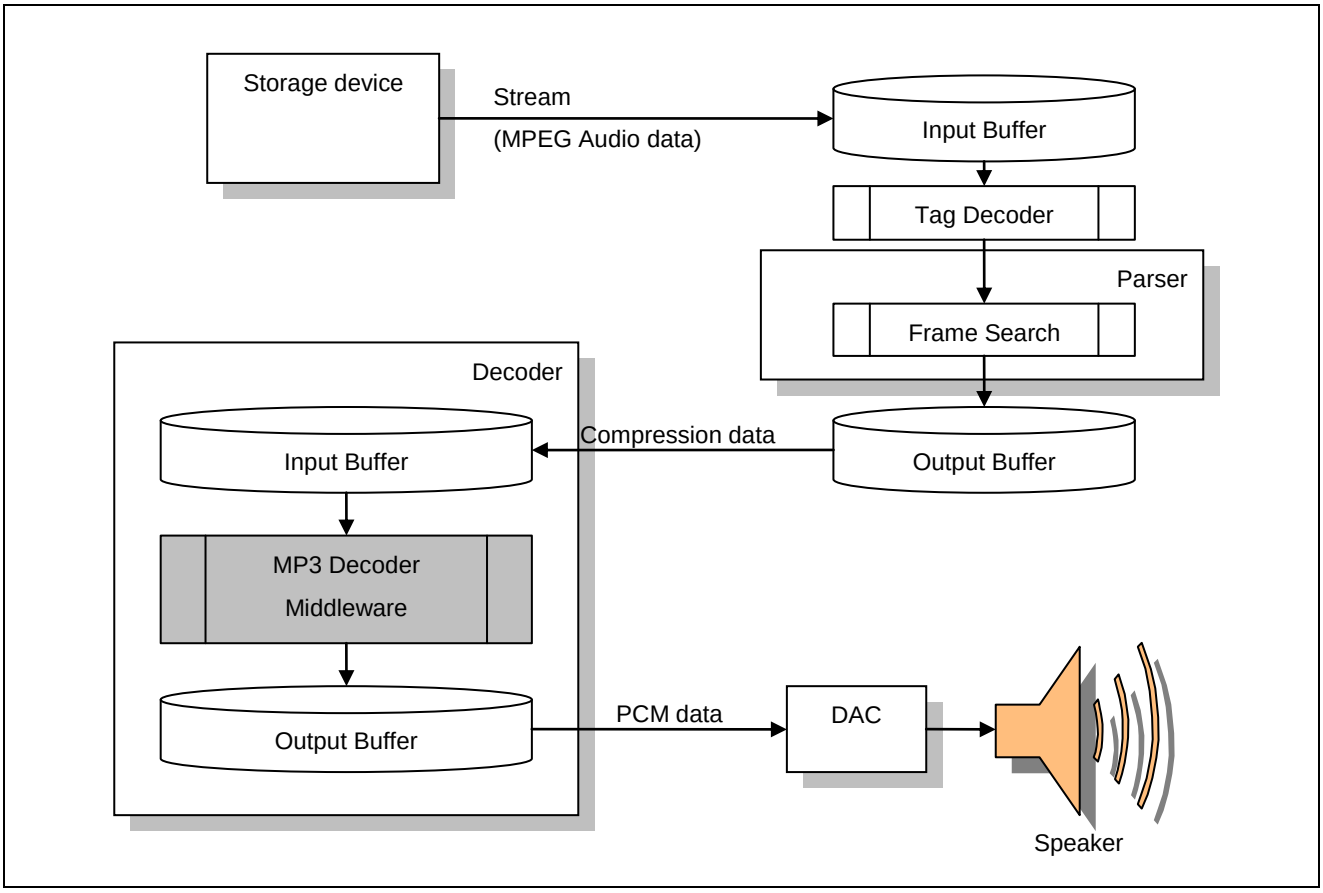

Figure 1.1 Example of the Decode System Configuration

1. MPEG Audio data

   MPEG Audio data is a linear PCM data sample compressed according to the MPEG Audio specifications. For these specifications, refer to Table 1.1.

2. Parser

   The parser outputs the MPEG Audio data in units of frames. The user should design the parser to suit the target system.

3. Compression data

   MPEG Audio bit stream data in frames.

4. Decoder

   This middleware processes the data stored in the input buffer and outputs the processing results to the output buffer.

5. PCM data

   16-bit linear PCM data which is a decoding result generated by this middleware.

6. DAC

   The DAC converts 16-bit linear PCM data into an analog signal.

2. Middleware Specifications

2.1 Library Functions

Table 2.1 lists the functions offered by this middleware. For detailed specifications of these functions, refer to Section 3.1.

Table 2.1 Functions

Function name	Outline
mp3d_GetMemorySize	Calculates the required memory size.
mp3d_Init	Initializes the MP3 decoder.
mp3d_Decode	Decodes MP3 frame data.
mp3d_GetErrorFactor	Obtains an error factor.
mp3d_GetVersion	Obtains version information.

2.2 Structures

Table 2.2 lists the structures for this middleware. The user should reserve areas required for these structures. For detailed specifications of these structures, refer to Section 3.2.

Table 2.2 Structures

Structure name	Outline	I/O
Memory size acquisition settings structure	Stores the parameters necessary for memory size acquisition.	I
Memory size acquisition results structure	Stores the acquired memory sizes.	O
Work memory information structure	Stores the parameters related to work memory.	I
Initialization settings structure	Stores the parameters necessary for initialization.	I
Decode settings structure	Stores the parameters necessary for decoding.	I
Decode results structure	Stores the decoding results.	O
Buffer memory settings structure	Stores the parameters related to the input/output buffer.	I
Buffer memory results structure	Stores the processing results related to the input/output buffer.	O

2.3 Macro Definitions

2.3.1 Type Definitions

Table 2.3 lists the type definitions available in this middleware.

Table 2.3 Type Definitions

Type	Size in bytes	Description	
ACMW_INT8	1	8-bit signed integer	-128 to 127
ACMW_INT16	2	16-bit signed integer	-32768 to 32767
ACMW_INT32	4	32-bit signed integer	-2147483648 to 2147483647
ACMW_UINT8	1	8-bit unsigned integer	0 to 255
ACMW_UINT16	2	16-bit unsigned integer	0 to 65535
ACMW_UINT32	4	32-bit unsigned integer	0 to 4294967295
ACMW_BOOL	2	Boolean value (16-bit signed integer)	Zero (false)/Non-zero (true)

[Note] All the pointers have the same size (4 bytes).

2.3.2 Common Symbols

Table 2.4 lists the symbol definitions available in this middleware.

Table 2.4 Common Symbols

Common symbol	Definition	Description
MP3D_RESULT_OK	0x00000000	Processing results are normal.
MP3D_RESULT_NG	0x00000001	Processing results are abnormal.
MP3D_RESULT_WARNING	0x00000002	Abnormality has occurred, which does not prevent the process from continuing.
MP3D_RESULT_FATAL	0x00000003	Abnormality has occurred, which prevents the process from continuing.

2.4 Reserved Words

Table 2.5 lists the naming rules for the symbols available in this middleware.

When you use this middleware together with other applications, be careful to avoid the duplication of symbol names.

Table 2.5 Naming Rules for Symbols

Classification	Outline
Function names	mp3d_XXXX
Structure names	mp3d_XXXX
Return values from functions	MP3D_RESULT_XXXX [Note] XXXX consists only of upper-case letters.
Error factor names	MP3D_ERR_XXXX [Note] XXXX consists only of upper-case letters.
Basic type prefix names	ACMW_XXXX [Note] XXXX consists only of upper-case letters.
Other prefix names	MP3D_XXXX [Note] XXXX consists only of upper-case letters.

[Note] XXXX can be any alphanumeric string.

2.5 Processing Flow

Figure 2.1 shows a flow diagram of processing performed by an application which uses this middleware.

The steps executed by the functions of this middleware are shaded. The steps defined by the user are white. Design the process to suit the target system.

Figure 2.2 show a flow diagram of processing sample program and parser.

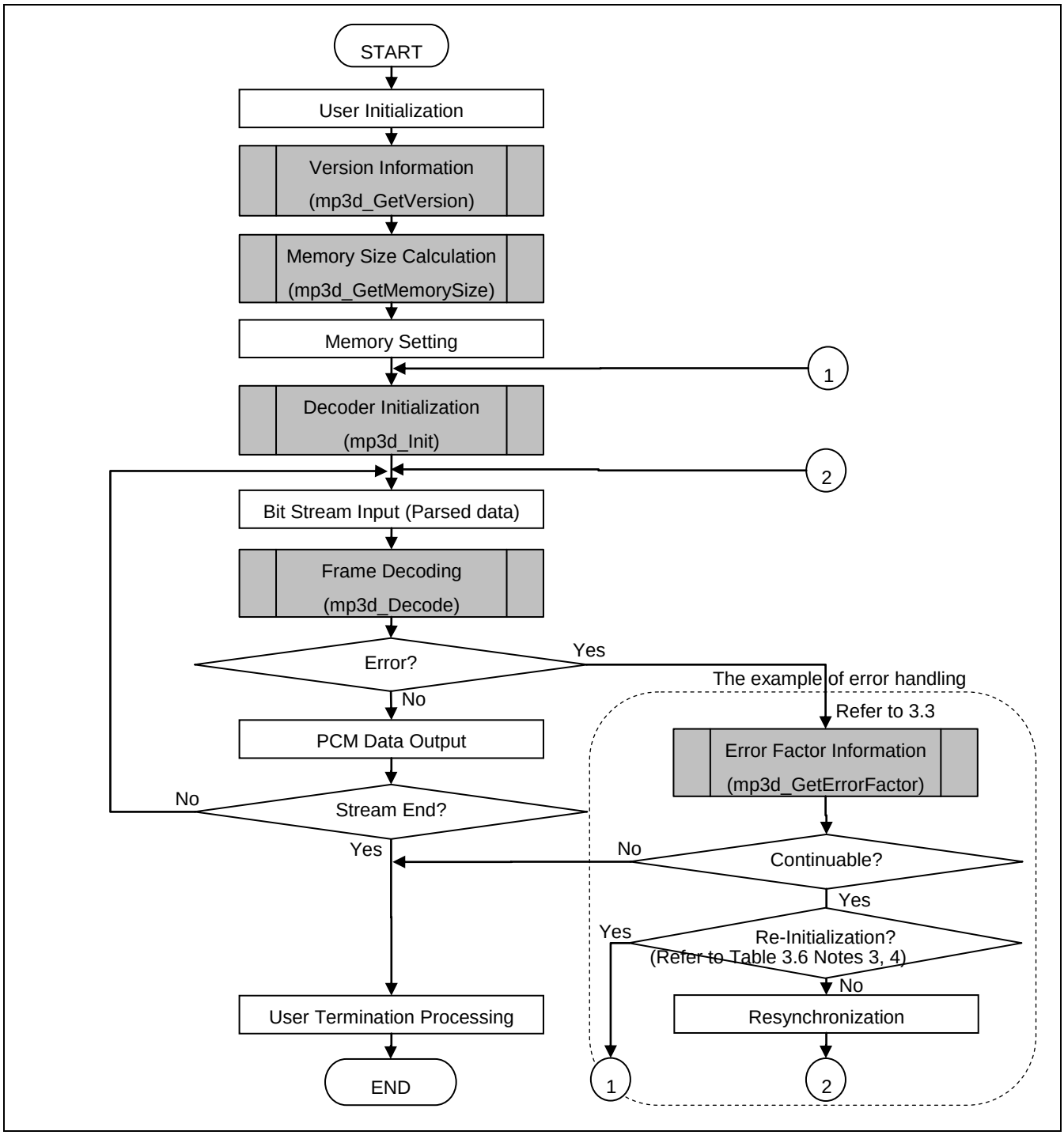

Figure 2.1 Example of the Application Processing Flow

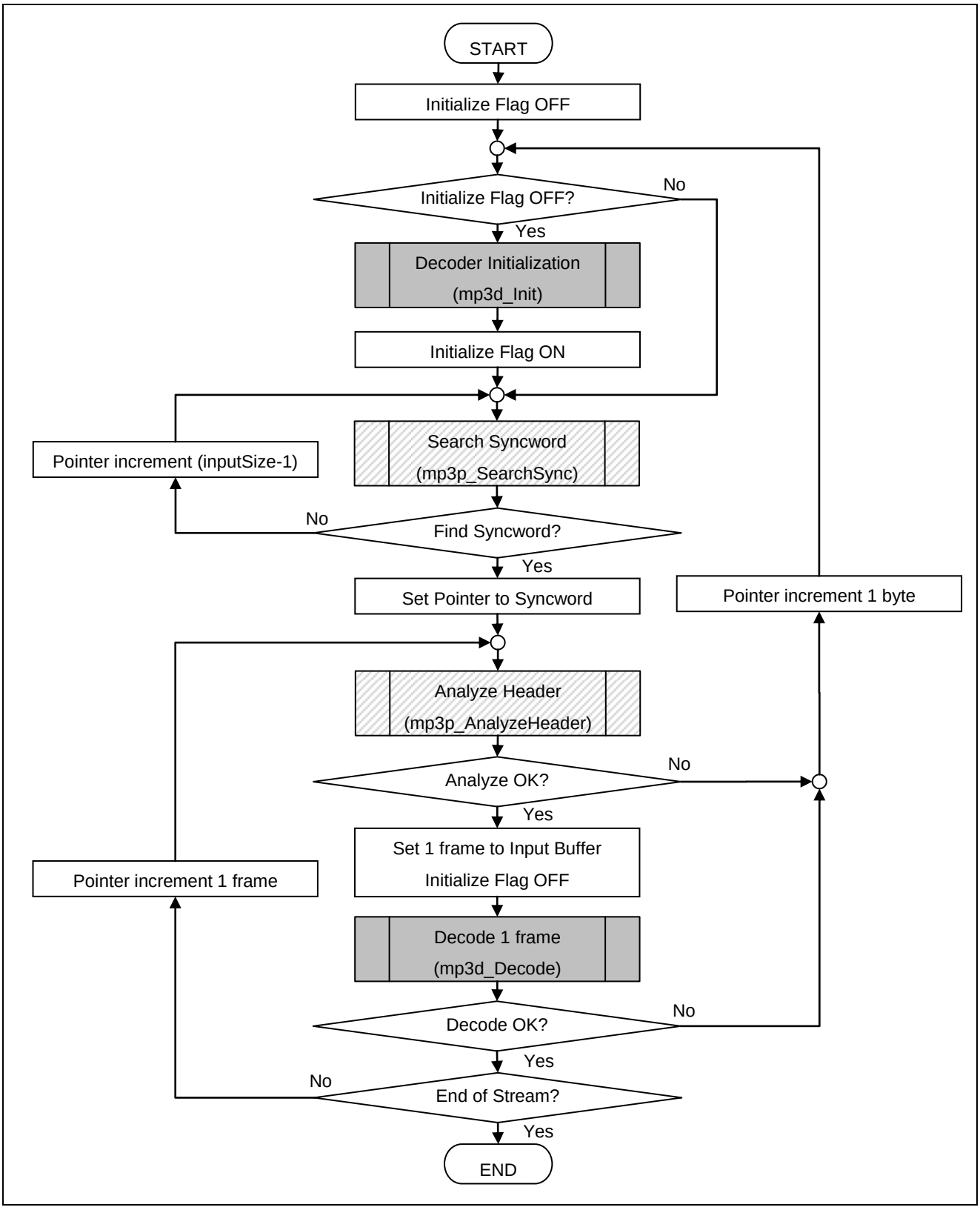

Figure 2.2 Example of the sample program and parser flow

3. Library Function Specifications

3.1 Function Specifications

The next sections describe this middleware's functions by using the description format below.

Synopsis	Outlines the function.		
Syntax	Describes the syntax for calling the function.		
Function	Describes what the function does.		
Arguments		I/O	Describes the arguments for the function.
Return value	Type name		Describes the return values from the function.
Description	Provides information such as precautions in using the function.		

[Note] This syntax format complies with ANSI-C. It does not use to standard C libraries of functions with C language standard.

3.1.1 mp3d_GetMemorySize Function

Synopsis	Calculates the necessary memory size.		
Syntax	ACMW_INT32 mp3d_GetMemorySize(const mp3d_getMemorySizeConfigInfo* const pGetMemorySizeConfigInfo, mp3d_getMemorySizeStatusInfo* const pGetMemorySizeStatusInfo);		
Function	This function calculates the necessary memory size for the static area, scratch area, and input/output buffer which are used by this middleware. Then, it stores the calculation results into the memory size acquisition results structure.		
Arguments		I/O	Meaning
mp3d_getMemorySizeConfigInfo *pGetMemorySizeConfigInfo		I	Memory size acquisition settings structure
mp3d_getMemorySizeStatusInfo *pGetMemorySizeStatusInfo		O	Memory size acquisition results structure
Return value	ACMW_INT32 errorCode		Error code For details, refer to Table 3.4
Description	Execute this function before mp3d_Init function and then reserve the required amount of memory space. You can execute this function at any time because it does not need to be initialized.		

3.1.2 mp3d_Init Function

Synopsis	Initializes the MP3 decoder.		
Syntax	ACMW_INT32 mp3d_Init(const mp3d_workMemoryInfo* const pWorkMemInfo, const mp3d_initConfigInfo* const pInitConfigInfo);		
Function	This function initializes the static and scratch areas which are used by this middleware. Also, it sets various parameters.		
Arguments		I/O	Meaning
mp3d_workMemoryInfo *pWorkMemInfo		I	Work memory information structure
mp3d_initConfigInfo *pInitConfigInfo		I	Initialization settings structure
Return value	ACMW_INT32 errorCode		Error code For details, refer to Table 3.4
Description	Execute this function only once before starting a series of decode process steps. They make up a process from the start to the end of decoding a certain stream.		

3.1.3 mp3d_Decode Function

Synopsis	Decodes MP3 frame data.		
Syntax	ACMW_INT32 mp3d_Decode(const mp3d_workMemoryInfo* const pWorkMemInfo, const mp3d_decConfigInfo* const pDecConfigInfo, const mp3d_ioBufferConfigInfo* const pBuffConfigInfo, mp3d_decStatusInfo* const pDecStatusInfo, mp3d_ioBufferStatusInfo* const pBuffStatusInfo);		
Function	This function decodes MPEG Audio frame data.		
Arguments		I/O	Meaning
mp3d_workMemoryInfo *pWorkMemInfo		I	Work memory information structure
mp3d_decConfigInfo *pDecConfigInfo		I	Decode settings structure
mp3d_ioBufferConfigInfo *pBuffConfigInfo		I	Buffer memory settings structure
mp3d_decStatusInfo *pDecStatusInfo		O	Decode results structure
mp3d_ioBufferStatusInfo *pBuffStatusInfo		O	Buffer memory results structure
Return value	ACMW_INT32 errorCode		Error code For details, refer to Table 3.4
Description	Execute this function when you decode one frame of stream data. [Note] If an error occurs, the members of the decode results structure are indefinite. Do not reference these members. [Note] If CRC is forcibly set to 0, the error code "MP3D_RESULT_WARNING" (severity: negligible) is returned upon error occurrence. The error factor is MP3D_ERR_CRC_ZERO.		

3.1.4 mp3d_GetErrorFactor Function

Synopsis	Obtains error factors.		
Syntax	ACMW_UINT32 mp3d_GetErrorFactor(const mp3d_workMemoryInfo* const pWorkMemInfo);		
Function	This function returns any error factors related to the most recently executed mp3d_Init, mp3d_Decode functions.		
Arguments		I/O	Meaning
mp3d_workMemoryInfo *pWorkMemInfo		I	Work memory information structure
Return value	ACMW_UINT32 errorFactor		Value indicating an error factor For details, refer to Table 3.6.
Description	This function returns any error factors related to the most recently executed mp3d_Init, mp3d_Decode functions. It cannot return error factors related to any other functions. Nor can it return error factors for any errors that have occurred before the mp3d_Init function initializes the static area. The error factors are overwritten next time you execute the mp3d_Init, mp3d_Decode functions. So, if you need the error factors, execute this function before reexecuting the functions above.		

3.1.5 mp3d_GetVersion Function

Synopsis	Obtains version information.		
Syntax	ACMW_UINT32 mp3d_GetVersion(void);		
Function	This function returns the version number of this middleware.		
Arguments		I/O	Meaning
None		-	-
Return value	ACMW_UINT32 versionCode		Version information Example: If the return code is 0x00000123, the version number is 1.23. For details, refer to Table 3.1.
Description	This function obtains the version number of this middleware. You can execute this function at any time.		

Table 3.1 versionCode Settings

Setting	Value	Description
Customer ID (8bit)	0x00	Standard version
	Others	Reserved
Release ID (8bit)	0x00	Authorized version
	0xA0 to 0xAF	alpha version (restricted in functionality) 0xA1 : alpha 1 ... 0xA9 : alpha 9 Other : Reserved
	0xB0 to 0xBF	beta version (not restricted in functionality, but not completely tested) 0xB1 : beta 1 ... 0xB9 : beta 9 Other : Reserved
	Others	Reserved
Major ID (8bit)	0xXY	Version XY.xy (major number) When X=0 to 9 and Y=0 to 9: 0x00 : Version 0.xy ... 0x10 : Version10.xy ... 0x99 : Version99.xy Other : Reserved
Minor ID (8bit)	0xXY	Version xy.XY (minor number) When X=0 to 9 and Y=0 to 9: 0x00 : Version xy.00 ... 0x10 : Version xy.10 ... 0x99 : Version xy.99 Other : Reserved

3.2 Structure Specifications

The next sections describe this middleware's structures by using the description format below.

[Structure name]	Name of the structure
[Function]	Describes what the structure does.
[Prototype]	Prototype of the structure
[Member description]	Describes the members of the structure.
[Remarks]	Provides information such as precautions in using the structure.

3.2.1 Memory Size Acquisition Settings Structure

[Structure name] mp3d_getMemorySizeConfigInfo

[Function] This structure specifies the conditions for calculating the necessary memory size when mp3d_GetMemorySize function obtains that size.

[Prototype]
```
typedef struct {
    ACMW_UINT32    Reserved;
} mp3d_getMemorySizeConfigInfo;
```

[Member description]

Member Variable Name	Description
Reserved	Reserved value

[Remarks] The user should reserve the necessary areas. The user should reserve the areas and set the values before calling the mp3d_GetMemorySize function.

3.2.2 Memory Size Acquisition Results Structure

[Structure name] mp3d_getMemorySizeStatusInfo

[Function] This structure stores the necessary-memory-size calculation results by using the necessary-memory-size calculation process (mp3d_GetMemorySize function).

[Prototype]
```
typedef struct {
    ACMW_UINT32     nStaticSize;
    ACMW_UINT32     nScratchSize;
    ACMW_UINT32     nInputBufferSize;
    ACMW_UINT32     nOutputBufferSize;
    ACMW_UINT32     nStackSize;
} mp3d_getMemorySizeStatusInfo;
```

[Member description]

Member Variable Name	Description
nStaticSize	Necessary memory size (in bytes) of the static area
nScratchSize	Necessary memory size (in bytes) of the scratch area
nInputBufferSize	Necessary memory size (in bytes) of the input buffer area
nOutputBufferSize	Necessary memory size (in bytes) of the output buffer area *1
nStackSize	Necessary memory size (in bytes) of the software stack area

[Remarks] The user should reserve the necessary areas before calling the mp3d_GetMemorySize function.

[Remarks2] *1: The size of the output buffer area is equal to that for each channel.

3.2.3 Work Memory Information Structure

[Structure name] mp3d_workMemoryInfo

[Function] This structure specifies the addresses in the work memory used by this middleware.

[Prototype]
```
typedef struct {
    void *      pStatic;
    void *      pScratch;
} mp3d_workMemoryInfo;
```

[Member description]

Member Variable Name	Description
pStatic	Pointer to the beginning of the static area.
pScratch	Pointer to the beginning of the scratch area.

[Remarks] The user should reserve the necessary areas. The user should reserve the areas and set the values before calling the library function which requires this structure as the arguments.

[Remarks2] You can obtain the sizes of the static and scratch areas by using the necessary-memory-size calculation process (mp3d_GetMemorySize function).

3.2.4 Initialization Settings Structure

[Structure name] mp3d_initConfigInfo

[Function] This structure specifies the decoding conditions for the initialization process (mp3d_Init function).

[Prototype]
```
typedef struct {
    ACMW_UINT32    Reserved;
} mp3d_initConfigInfo;
```

[Member description]

Member Variable Name	Description
Reserved	Reserved value

[Remarks] The user should reserve the necessary areas. The user should reserve the areas and set the values before calling the mp3d_Init function.

3.2.5 Decode Settings Structure

[Structure name] mp3d_decConfigInfo

[Function] This structure specifies the processing conditions when the mp3d_Decode function decodes data.

[Prototype]
```
typedef struct {
    ACMW_UINT32    Reserved;
} mp3d_decConfigInfo;
```

[Member description]

Member Variable Name	Description
Reserved	Reserved value

[Remarks] The user should reserve the necessary areas. The user should reserve the areas and set the values before calling the mp3d_Decode function.

3.2.6 Decode Results Structure

[Structure name] mp3d_decStatusInfo

[Function] This structure stores the results of decoding performed by the mp3d_Decode function.

[Prototype]

```
typedef struct {
    ACMW_UINT32     nSamplingRate;
    ACMW_UINT16     nChannelNum;
    ACMW_UINT16     nFrameBitrate;
    mp3d_headerInfo sHeader;
    mp3d_frameInfo  sFrame;
    ACMW_UINT16     nSamplesPerFrame;
} mp3d_decStatusInfo;
```

[Member description]

Member Variable Name	Description
nSamplingRate	Sampling frequency [Hz]
nChannelNum	Number of channels
nFrameBitrate	Bit rate [kbps]
sHeader	Header information structure (see 3.2.9)
sFrame	Frame information structure (see 3.2.10)
nSamplesPerFrame	Number of samples per channel per frame

[Remarks] The user should reserve the necessary areas. The user should reserve the areas and set the values before calling the mp3d_Decode function.

[Remarks2] The members of this structure are automatically aligned in C language.

3.2.7 Buffer Memory Settings Structure

[Structure name] mp3d_ioBufferConfigInfo

[Function] This structure specifies the parameters for the input buffer used by the mp3d_Decode function.

[Prototype]
```
typedef struct {
    ACMW_UINT8 *    pInBuffStart;
    ACMW_UINT32     nInBuffSetDataSize;
    ACMW_INT16 **   pOutBuffStart;
    ACMW_UINT32     nOutBuffSize
} mp3d_ioBufferConfigInfo;
```

[Member description]

Member Variable Name	Description
pInBuffStart	Start address of input data
nInBuffSetDataSize	Size (in bytes) of input data
pOutBuffStart	Start address of output data per channel
nOutBuffSize	Size (in bytes) of each output buffer*1

[Remarks] The user should reserve the necessary areas. The user should reserve the areas and set the values before calling the mp3d_Decode function.

[Remarks2] *1: The output buffer size is the same for all channels.

3.2.8 Buffer Memory Results Structure

[Structure name] mp3d_ioBufferStatusInfo

[Function] This structure stores the results of processing information about buffer memory used by the mp3d_Decode function.

[Prototype]
```
typedef struct {
    ACMW_UINT8 *     pInBuffLast;
    ACMW_UINT32      nInBuffUsedDataSize;
    ACMW_INT16 **    pOutBuffLast;
    ACMW_UINT32      nOutBuffUsedDataSize;
} mp3d_ioBufferStatusInfo;
```

[Member description]

Member Variable Name	Description
pInBuffLast	Post-read address of an input buffer*1
nInBuffUsedDataSize	Consumed data size (in bytes) of an input buffer
pOutBuffLast	Post-write address of output data for each channel*1
nOutBuffUsedDataSize	Consumed data size (in bytes) of an output buffer for each channel*2

[Remarks] The user should reserve the necessary areas. The user should reserve the areas and set the values before calling the mp3d_Decode function.

[Remarks2] *1: This address indicates the next start address.
*2: The consumed data size of an output buffer is the same for all channels.

3.2.9 Header Information Structure

[Structure name] mp3d_headerInfo

[Function] This structure stores header information.

[Prototype]
```
typedef struct {
    ACMW_UINT16    nID;
    ACMW_UINT16    nLayer;
    ACMW_UINT16    nProtectionBit;
    ACMW_UINT16    nBitrateIndex;
    ACMW_UINT16    nSamplingIndex;
    ACMW_UINT16    nPaddingBit;
    ACMW_UINT16    nPrivateBit;
    ACMW_UINT16    nMode;
    ACMW_UINT16    nModeExtension;
    ACMW_UINT16    nCopyright;
    ACMW_UINT16    nOrgCopy;
    ACMW_UINT16    nEmphasis;
} mp3d_headerInfo;
```

[Member description]

Member Variable Name	Description
nID	Standard IDex+ID (identification between MPEG-1, MPEG-2, and MPEG2.5) 0x0000 : MPEG2.5 Audio Layer-3 0x0002 : MPEG-2 Audio Layer-1/2/3 0x0003 : MPEG-1 Audio Layer-1/2/3 Other : Reserved
nLayer	Layer 0x0001 : Layer-3 0x0002 : Layer-2 0x0003 : Layer-1 Other : Reserved
nProtectionBit	Presence/absence of CRC 0x0000 : CRC present 0x0001 : CRC absent Other : Reserved
nBitrateIndex	Bit rate index (see Table 3.2)
nSamplingIndex	Sampling frequency index (see Table 3.3)
nPaddingBit	Padding bit 0x0000 : Padding bit not added to the frame 0x0001 : 1 byte (4 bytes for Layer -1) added to the frame Other : Reserved
nPrivateBit	Private bit
nMode	Mode 0x0000 : stereo 0x0001 : joint stereo 0x0002 : dual channel 0x0003 : single channel Other : Reserved

Member Variable Name	Description
nModeExtension	Mode extension (is: intensity stereo, ms: MS stereo) *1 0x0000 : is_off, ms_off 0x0001 : is_on, ms_off 0x0002 : is_off, ms_on 0x0003 : is_on, ms_on Other : Reserved
nCopyright	Copyright 0x0000 : No copyright 0x0001 : Copyright protected Other : Reserved
nOrgCopy	Original/copy 0x0000 : Copy 0x0001 : Original Other : Reserved
nEmphasis	Emphasis 0x0000 : No emphasis 0x0001 : 50/15us 0x0002 : Reserved 0x0003 : CCITT J.17 Other : Reserved

[Remarks] The members of this structure are the same as those of the decode results structure shown in Section 3.2.6. Thus, the header information structure is reserved when the decode results structure is reserved. The user should reserve the decode results structure before calling the mp3d_Decode function.

[Remarks 2] *1: This mode extension is always 0x0000 when the mode is not joint stereo. The values for Layer-3 are shown above. In the case of Layer-1/2, subbands in intensity stereo are shown.

Table 3.2 Bit Rate Indexes

Value	MPEG-1			MPEG-2		MPEG2.5
	Layer-1 [kbps]	Layer-2 [kbps]	Layer-3 [kbps]	Layer-1 [kbps]	Layer-2/3 [kbps]	Layer-3 [kbps]
0x0000	Free format (not supported by this middleware)					
0x0001	32	32	32	32	8	8
0x0002	64	48	40	48	16	16
0x0003	96	56	48	56	24	24
0x0004	128	64	56	64	32	32
0x0005	160	80	64	80	40	40
0x0006	192	96	80	96	48	48
0x0007	224	112	96	112	56	56
0x0008	256	128	112	128	64	64
0x0009	288	160	128	144	80	80*1
0x000a	320	192	160	160	96	96*1
0x000b	352	224	192	176	112	112*1
0x000c	384	256	224	192	128	128*1
0x000d	416	320	256	224	144	144*1
0x000e	448	384	320	256	160	160*1
0x000f	Forbidden (not supported by this middleware)					
Other	Reserved					

[Note] *1: Bit rates of 80 kbps and above, which are not defined for MPEG-2.5 Layer-3, are treated the same as the bit rates for MPEG-2 Layer-3.

Table 3.3 Sampling Frequency Indexes

Value	MPEG-1 Audio Layer-1/2/3 [Hz]	MPEG-2 Audio Layer-1/2/3 [Hz]	MPEG2.5 Audio Layer-3 [Hz]
0x0000	44100	22050	11025
0x0001	48000	24000	12000
0x0002	32000	16000	8000
Other	Reserved		

3.2.10 Frame Information Structure

[Structure name] mp3d_frameInfo

[Function] This structure stores frame information.

[Prototype]
```
typedef struct {
    ACMW_UINT8 *    pAncillaryPtr;
    ACMW_UINT16     nAncillarySize;
    ACMW_UINT16     nFrameSize;
} mp3d_frameInfo;
```

[Member description]

Member Variable Name	Description
pAncillaryPtr	Start position of ancillary data *1
nAncillarySize	Size (in bits) of ancillary data *1
nFrameSize	Frame size (in bytes)

[Remarks] The members of this structure are the same as those of the decode results structure shown in Section 3.2.6. Thus, the frame information structure is reserved when the decode results structure is reserved. The user should reserve the decode results structure before calling the mp3d_Decode function.

[Remarks2] *1: Values for Layer-1/2 only are set.

3.3 Error Processing

This middleware's functions return the error codes listed in Table 3.4. To obtain details about the cause of an error, execute the mp3d_GetErrorFactor function.

3.3.1 Error codes

Below are the error codes for this middleware.

Table 3.4 Error Codes

Error code (32bit)	Value	Description	Reinitialization
[1] MP3D_RESULT_OK	0x00000000	The processing results are normal. The process has terminated normally.	Unnecessary
[2] MP3D_RESULT_NG	0x00000001	The processing results are abnormal. An invalid parameter is specified in the structure. Or else, the program execution is incorrect. PCM data is not output. Specify the valid parameter in the structure or reexecute the program by using the correct procedure.	Unnecessary
[3] MP3D_RESULT_WARNING	0x00000002	Abnormality has occurred, which does not disable process continuation. The decoder detected an error, but PCM data was output. At this time, the error concealment or MUTE signal (all 0's) might have been output. Check the error by using the error factor acquisition process (mp3d_GetErrorFactor function).	Unnecessary
[4] MP3D_RESULT_FATAL	0x00000003	Abnormality has occurred, which disables process continuation. The process cannot continue. PCM data is not output. Reinitialization the program. An error factor cannot be obtained through the mp3d_GetErrorFactor function.	Necessary
[5] Others	Other than the above	Reserved	-

Table 3.5 Error Codes Used by the Library Functions

Function / Error code	mp3d_GetMemorySize	mp3d_Init	mp3d_Decode	mp3d_GetErrorFactor*1	mp3d_GetVersion*2
MP3D_RESULT_OK	o	o	o	-	-
MP3D_RESULT_NG	-	o	o	-	-
MP3D_RESULT_WARNING	-	-	o	-	-
MP3D_RESULT_FATAL	o	o	o	o	-

[Note] o: Error code might be output. -: Error code is not used.

[Note] *1: Returns an error factor.

*2: Returns version information.

3.3.2 Error Factors

An error factor provides details about an error upon its occurrence. The mp3d_GetErrorFactor function lets you obtain error factors related to errors except for the MP3D_RESULT_FATAL error. Table 3.6 lists the error factors.

Table 3.6 Error Factors

errorFactor(32bit)	Value	Description	Table 3.4	PCM
MP3D_ERR_NONE	0x00000000	The program has normally terminated. No error factor is available.	[1]	Normal*1
MP3D_ERR_POINTER	0x00000010	Invalid pointer value	[2]	Unavailable
MP3D_ERR_PARAMETER	0x00000020	Invalid parameter	[2]	Unavailable
MP3D_ERR_SEQUENCE	0x00000040	The library functions were executed in invalid order.*3	[2]	Unavailable
MP3D_ERR_HDR_SYNCWORD	0x00010000	The header's syncword was not detected.	[2]	Unavailable
MP3D_ERR_HDR_FS_CHANGE	0x00020000	The sampling frequency is different from the one for a previous frame.*4	[2]	Unavailable
MP3D_ERR_HDR_BAD_PARAMETER	0x00040000	The header's parameter is abnormal.	[2]	Unavailable
MP3D_ERR_HDR_LAYER_CHANGE	0x00080000	The layer is different from the one for a previous frame.*4	[2]	Unavailable
MP3D_ERR_1ST2ND_FRAME	0x00100000	The first two frames have been output.	[3]	Available
MP3D_ERR_CRC	0x00200000	A CRC error has occurred.	[3]	Available
MP3D_ERR_CRC_ZERO	0x00400000	The CRC word was 0.	[3]	Available
MP3D_ERR_DECODE	0x01000000	An error has occurred during decode processing.	[3]	Available*2
Others	Other than the above	Reserved	-	-

[Note] *1: Error determination may be impossible depending on which part of data is damaged. In this case, the error factor is assumed to be MP3D_ERR_NONE, the decode processing continues, and the decoding results are output.

*2: PCM data generation may be prevented during the first half of the decode processing depending on which part of data is damaged.

*3: Reinitialization is necessary because the execution sequence is invalid.

*4: Reinitialization is necessary if new frame information is used for playback.

Table 3.7 Relationship of the Library Functions to the Error Factors

Function / Error factor	mp3d_GetMemorySize	mp3d_Init	mp3d_Decode	mp3d_GetErrorFactor*1	mp3d_GetVersion
MP3D_ERR_NONE	-	o	o	-	-
MP3D_ERR_POINTER	-	o	o	-	-
MP3D_ERR_PARAMETER	-	-	o	-	-
MP3D_ERR_SEQUENCE	-	-	o	o	-
MP3D_ERR_HDR_SYNCWORD	-	-	o	-	-
MP3D_ERR_HDR_FS_CHANGE	-	-	o	-	-
MP3D_ERR_HDR_BAD_PARAMETER	-	-	o	-	-
MP3D_ERR_HDR_LAYER_CHANGE	-	-	o	-	-
MP3D_ERR_1ST2ND_FRAME	-	-	o	-	-
MP3D_ERR_CRC	-	-	o	-	-
MP3D_ERR_CRC_ZERO	-	-	o	-	-
MP3D_ERR_DECODE	-	-	o	-	-

[Note] o: Error factor might be output. -: Error factor is not used.

[Note] *1: Returns an error factor.

In the case of could not returns an error factor, returns MP3D_RESULT_FATAL as error code.

3.4 Memory Specifications

This section describes the memory areas used by this middleware.

3.4.1 Scratch Area

Table 3.8 Scratch Area Description

Item	Area which temporarily contains values when this middleware is used. If the user manipulates this area for interrupt or other processing while a function in this middleware is being called, the correct execution of this middleware cannot be ensured. The user can freely use this area after decoding one frame.
Symbol name	- (freely defined by the user)
Size	Obtain the actually required size with mp3d_GetMemorySize.
Area reservation	The user should reserve this area. The user can freely use this area after returning from a function in this middleware. Note that if the user calls a function in this middleware after using this area, the value stored in this area is overwritten.
Allocation	This area is included in RAM.
Alignment	Align this area on a 4-byte boundary.

3.4.2 Static Area

Table 3.9 Static Area Description

Item	Area which always holds values when this middleware is used. If the user manipulates this area after initialization, the correct execution of this middleware is not ensured.
Symbol name	- (freely defined by the user)
Size	Obtain the actually required size with mp3d_GetMemorySize.
Area reservation	The user should reserve this area.
Allocation	This area is included in RAM.
Alignment	Align this area on a 4-byte boundary.

3.4.3 Software Stack Area

Table 3.10 Software Stack Area Description

Item	Stack area used by this middleware
Symbol name	- (freely defined by the user)
Size	Obtain the actually required size with mp3d_GetMemorySize.
Area reservation	The user should reserve this area. To use this middleware, reserve a software stack area which exceeds the size above.
Allocation	This area is included in RAM.
Alignment	-

3.4.4 Heap Area

This middleware does not use a heap area.

3.4.5 Input Buffer

Table 3.11 Input Buffer Description

Item	Area which stores inputs to this middleware. The input buffer contains stream data (MPEG Audio compressed data) If the user manipulates this area during decode processing, the normal execution of the program cannot be ensured. [Note] This middleware does not support an input buffer which is a ring buffer.
Symbol name	- (freely defined by the user)
Size	Be sure to reserve an amount of memory equal to or greater than the necessary memory size obtained by the mp3d_GetMemorySize function.
Area reservation	The user should reserve this area. The user can freely use this area after decoding one frame.
Allocation	This area is included in RAM.
Alignment	Alignment is not restricted.

When executing the mp3d_Decode function, specify the parameters in the data input settings structure. This stores the processing results into the buffer memory results structure. Figure 3.1 shows the relationship between the input buffer and structure members.

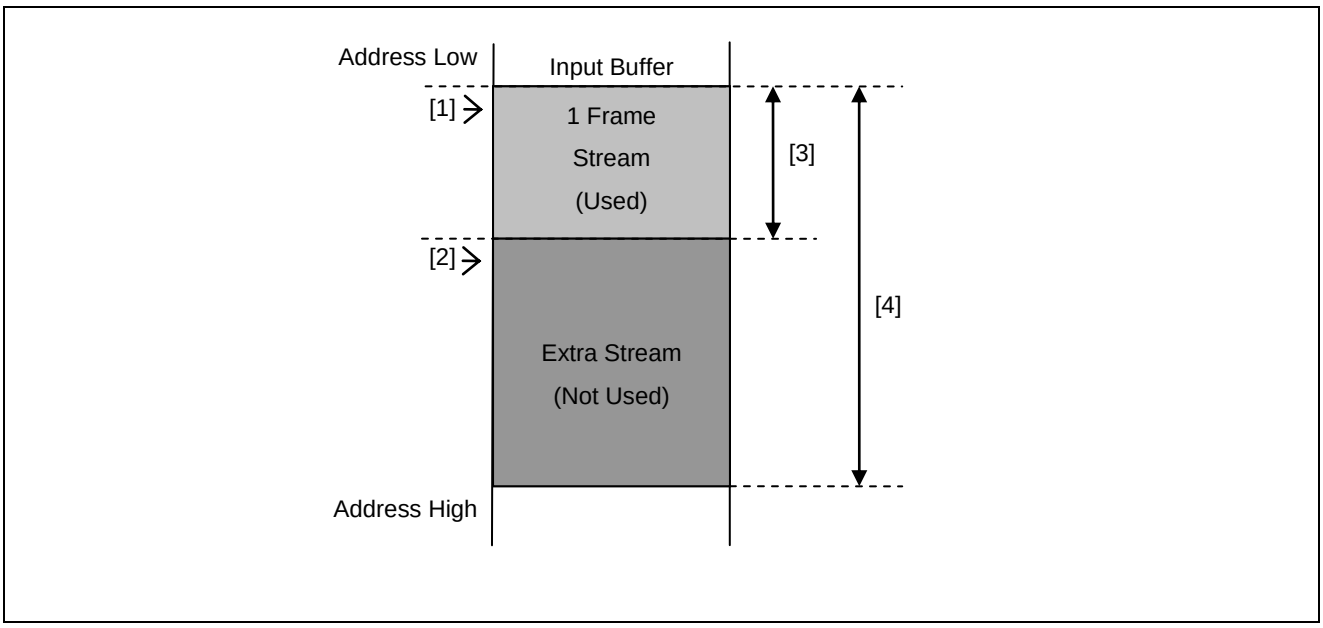

Figure 3.1 Structure Members in the Input Buffer

[Note] Data exceeding the size of one frame is placed into the input buffer for decode processing.

Table 3.12 Structure Members in the Input Buffer

[1]	pInBuffStart	(Buffer memory settings structure)	Input data start address
[2]	pInBuffLast	(Buffer memory results structure)	Input buffer post-read address
[3]	nInBuffUsedDataSize	(Buffer memory results structure)	Input buffer consumed-data size
[4]	nInBuffSetDataSize	(Buffer memory settings structure)	Input data size

[Note] Items [1] and [2] indicate addresses. Items [3] and [4] indicate sizes.

(1) Input data storage method

Figure 3.2 shows how input data is stored. Store data given in bytes into the input buffer (memory).

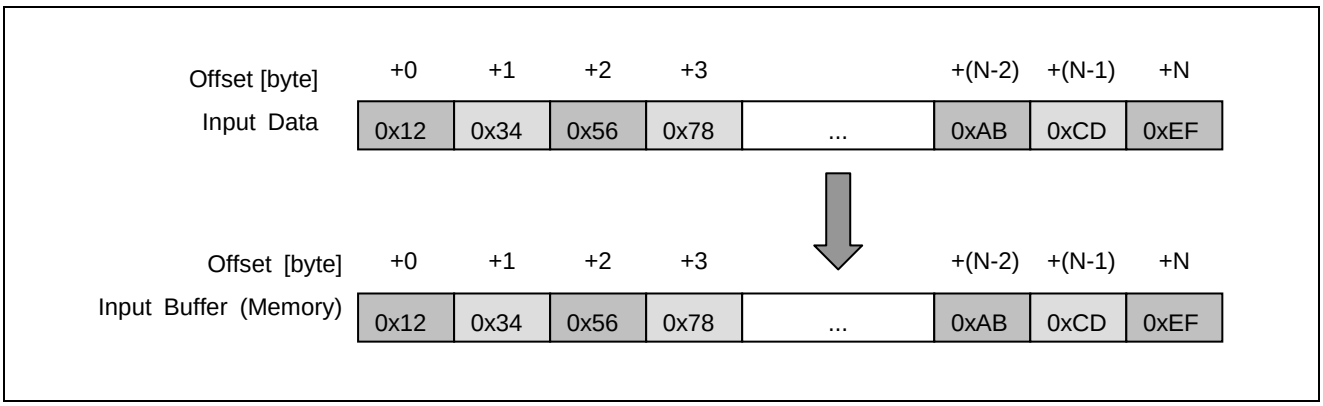

Figure 3.2 Input Data Storage Method

3.4.6 Output Buffer

Table 3.13 Output Buffer Description

Item	Area which stores outputs from this middleware. The output buffer contains 16-bit linear PCM data (hereinafter called PCM data). If the user manipulates this area during decode processing, the normal execution of the program cannot be ensured.
Symbol name	- (freely defined by the user)
Size	Be sure to reserve an amount of memory equal to or greater than the necessary memory size obtained by the mp3d_GetMemorySize function. [Note] The function above obtains a size per channel. Thus, you need to reserve memory for two channels.
Area reservation	The user should reserve this area. The user can freely use this area after decoding one frame.
Allocation	This area is included in RAM.
Alignment	Align this area on a 2-byte boundary.

When executing the mp3d_Decode function, specify the parameters in the buffer memory settings structure. This stores the processing results into the buffer memory results structure. Figure 3.3 shows the relationship between the output buffer and the structure members.

The second and subsequent channels are managed just like the first channel. It does not matter whether the output buffers for the first and second channels are consecutive or not. The number of samples output to the output buffer varies depending on the input data and decode conditions. Refer to the description of nSamplesPerFrame of the decode results structure.

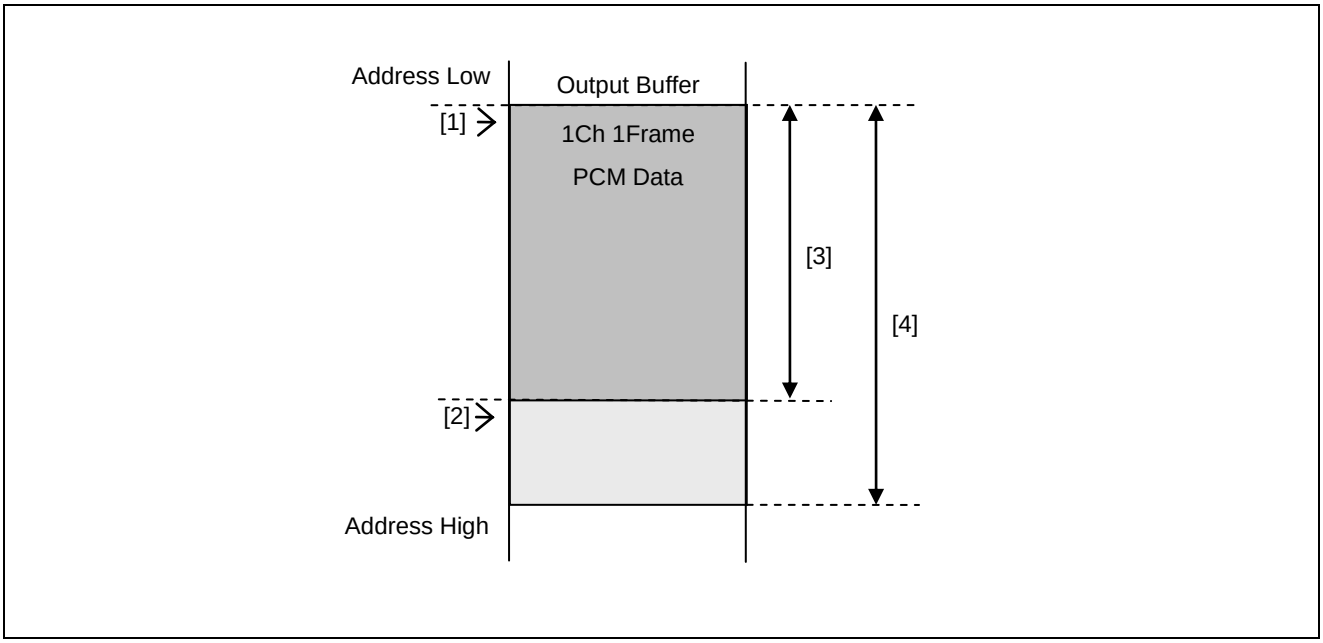

Figure 3.3 Structure Members in the Output Buffer

[Note] Output results for one frame are placed into the output buffer for decode processing.

Table 3.14 Structure Members in the Output Buffer

[1]	pOutBuffStart[0]	(Buffer memory settings structure)	Output data start address for the first channel
[2]	pOutBuffLast[0]	(Buffer memory results structure)	Post-write address for the first channel
[3]	nOutBuffUsedDataSize	(Buffer memory results structure)	Consumed data size per channel
[4]	nOutBuffSize	(Buffer memory settings structure)	Output buffer size per channel

[Note] Items [1] and [2] indicate addresses. Items [3] and [4] indicate sizes.

(1) Output data storage method

Data is output in the formats which vary depending on the decode conditions as shown in Figure 3.5 through Figure 3.7 (consecutive buffers are specified for the channels). The output buffer (memory) stores data in 2-byte (16-bit) units. The byte order for accessing the buffer is little endian (see Figure 3.4).

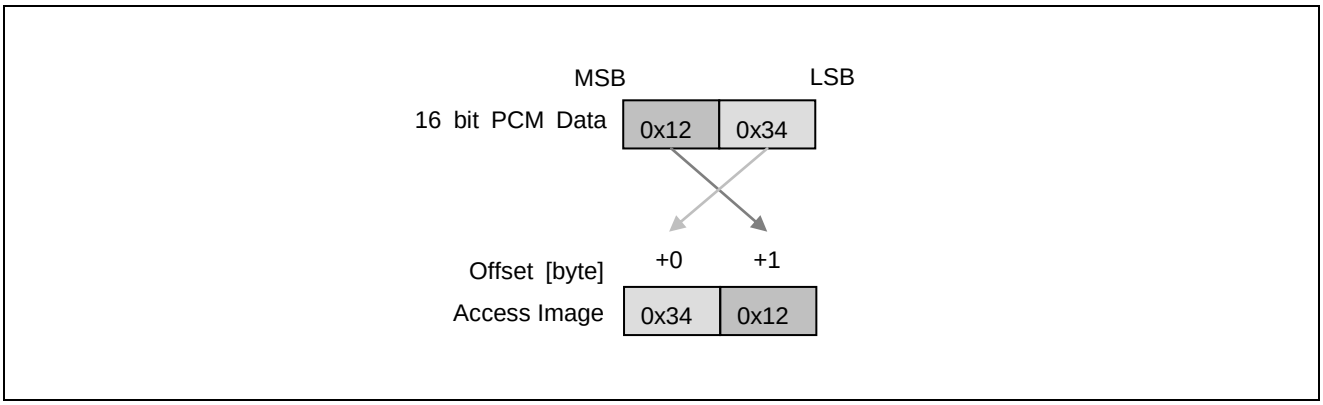

Figure 3.4 PCM Data Access (Little Endian Mode)

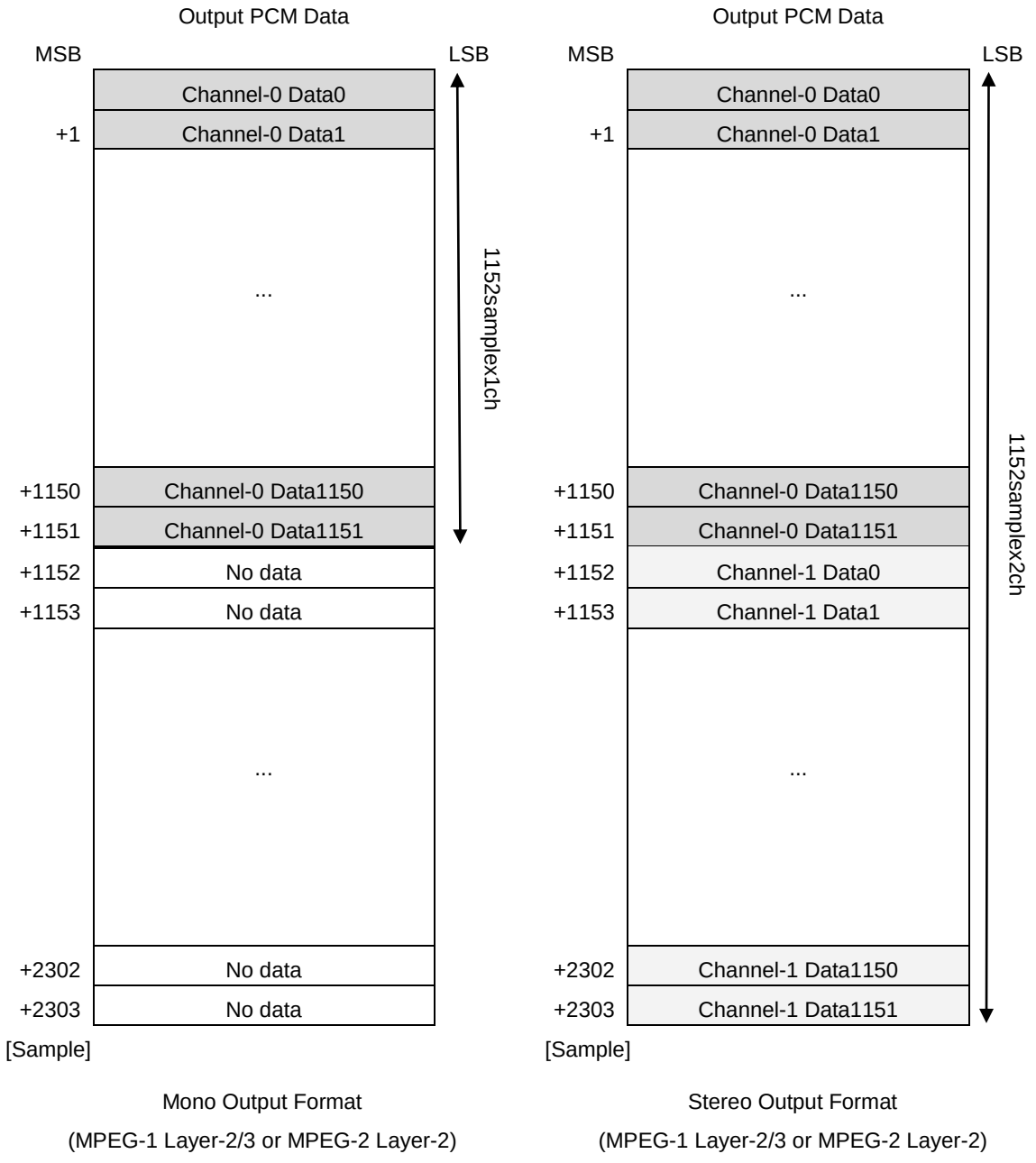

Figure 3.5 MPEG-1 Layer-2/3 or MPEG-2 Layer-2 Output Formats

[Note] Consecutive buffers are specified for the channels. In fact, there is no problem if the start position of Channel-1 is not +1152. (Nonconsecutive buffers can be specified for the buffers.)

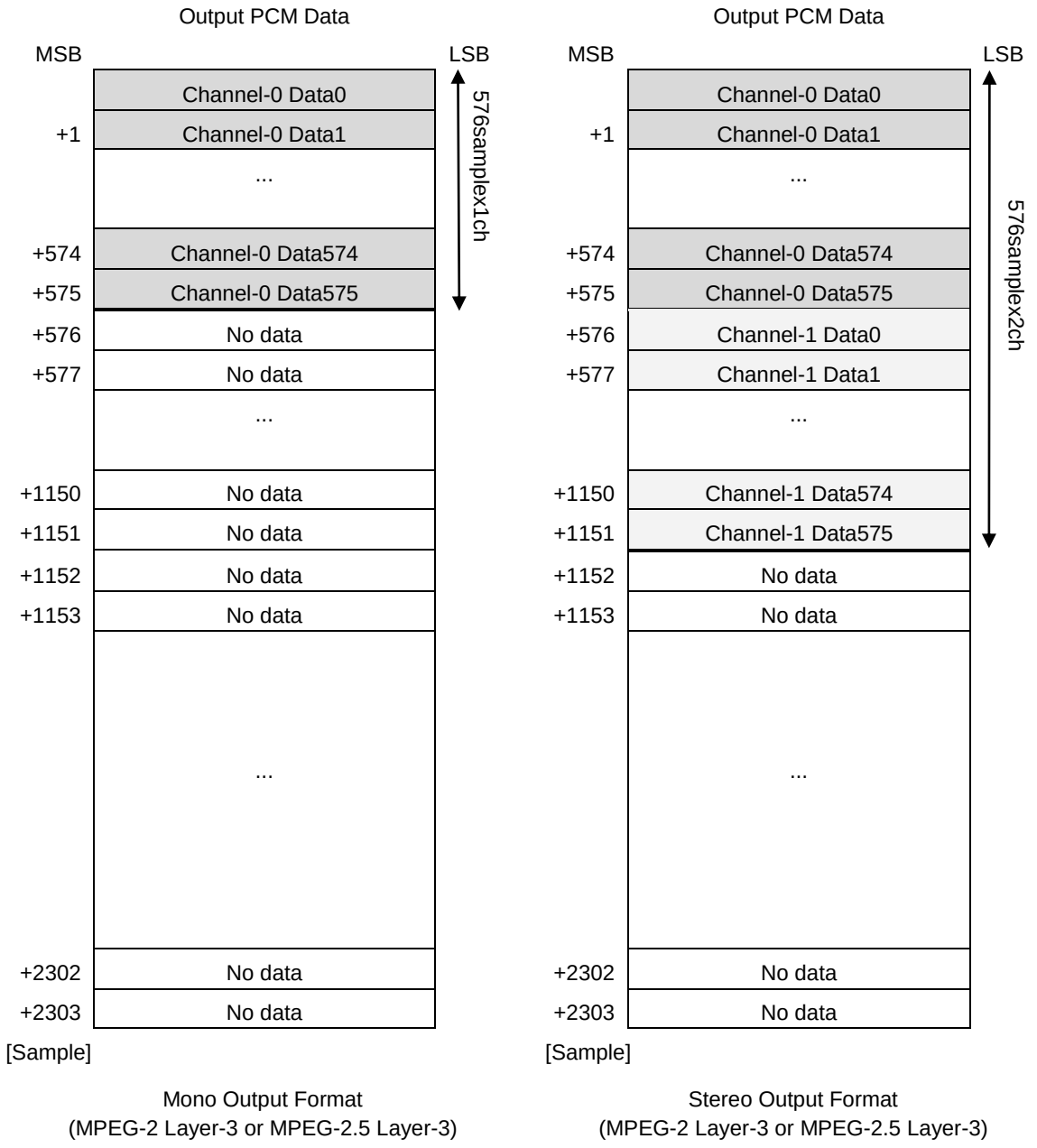

Figure 3.6 MPEG-2 Layer-3 or MPEG-2.5 Layer-3 Output Formats

[Note] Consecutive buffers are specified for the channels. In fact, there is no problem if the start position of Channel-1 is not +576. (Nonconsecutive buffers can be specified for the buffers.)

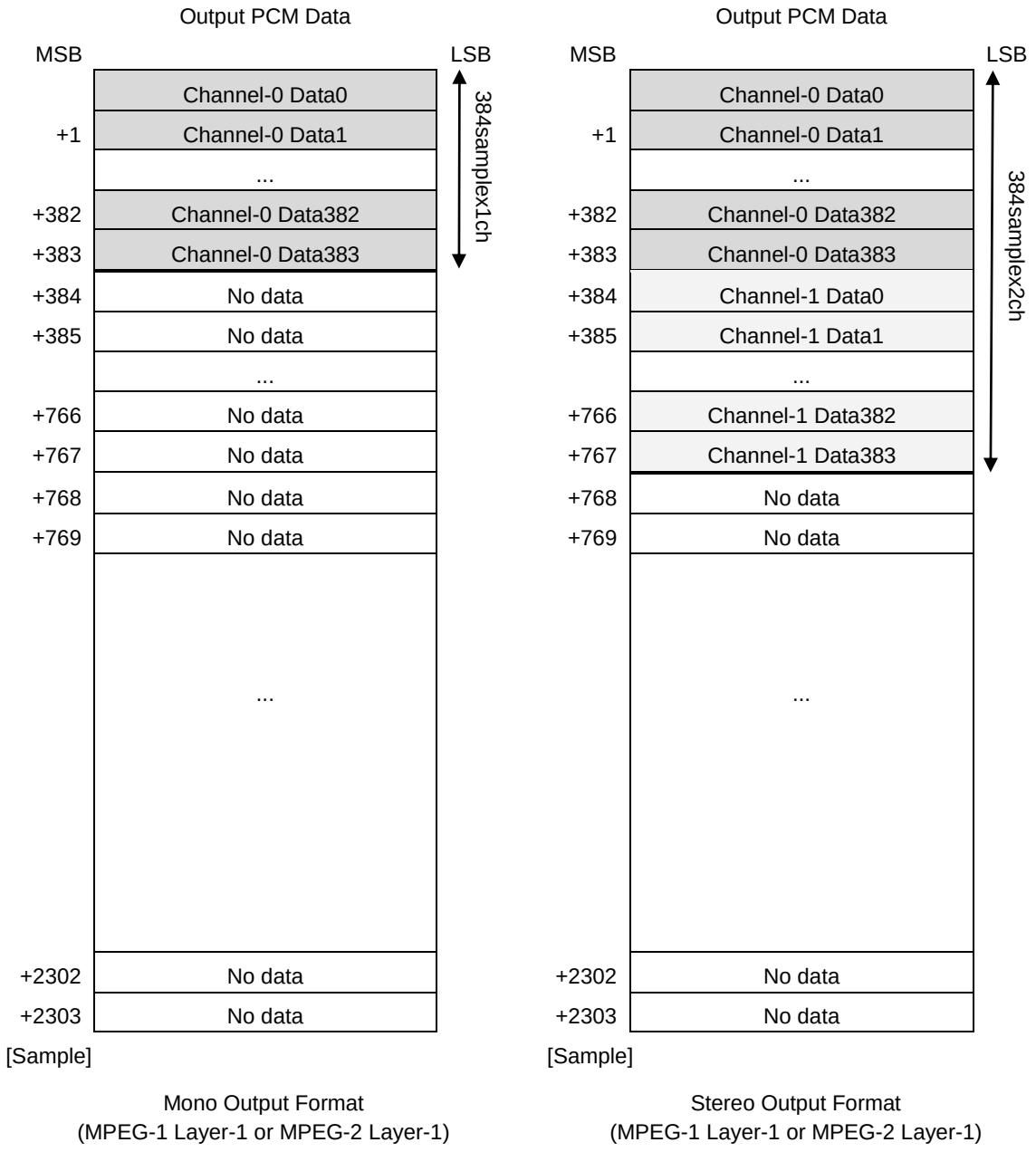

Figure 3.7 MPEG-1 Layer-1 or MPEG-2 Layer-1 Output Formats

[Note] Consecutive buffers are specified for the channels. In fact, there is no problem if the start position of Channel-1 is not +384. (Nonconsecutive buffers can be specified for the buffers.)

3.5 Input Data

You can input MPEG Audio compressed data to this middleware. For details about the compressed-data format, refer to the specifications. This product complies with the specifications except for the fact that it does not support the free format for the bitrate_idex '0000', nor does it support three or more channels. Compressed data consists of multiple bit streams. Each bit stream is called a frame.

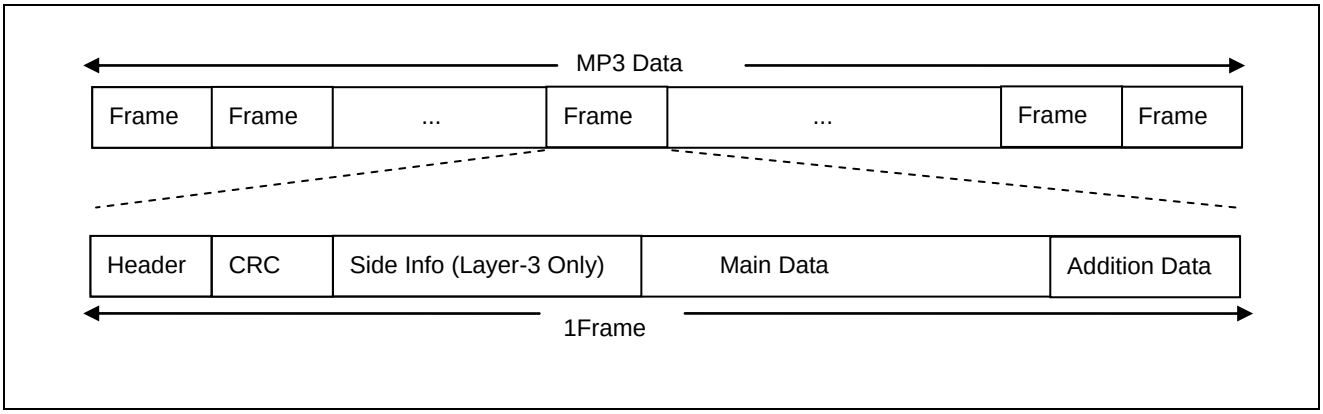

Figure 3.8 Compressed Data Format

3.5.1 Header

A header contains not only a sampling frequency for synchronization, but also other information such as a bit rate, mode, etc. Only the headers for MPGE-1 Layer-2 audio are restricted of Table 3.18.

Table 3.15 Header Details

Information	No. of bits	Value
Frame synchronization word	11	'1111 1111 111': Fixed value
IDex	1	'1': MPEG-1, MPEG-2 '0': MPEG2.5
ID	1	'1': MPEG-1 '0': MPEG-2, MPEG2.5
Layer	2	'11': Layer-1 '10': Layer-2 '01': Layer-3
Protection bit	1	'0': CRC present '1': CRC absent
Bit rate	4	See Table 3.16.
Sampling frequency	2	See Table 3.17.
Padding bit	1	'1': Bits added to frames for adjustment of fractional amounts '0': Bits not added to frames for adjustment of fractional amounts [Note] If the sampling frequency is 44.1, 22.05, or 11.025 kHz, 1 byte (or 4 bytes for Layer-1) is added to frames for adjustment of fractional amounts.
Private bit	1	Not used
Mode	2	'00': stereo '01': joint_stereo '10': dual_channel '11': single_channel
Mode extension	2	'00': is_off, ms_off '01': is_on, ms_off '10': is_off, ms_on '11': is_on, ms_on [Note] "is" stands for intensity stereo. "ms" stands for MS stereo. [Note] In the case of Layer-1/2, subbands in intensity stereo are shown.
Copyright	1	'0': No copyright '1': Copyright protected
Original/copy	1	'0': Copy '1': Original
Emphasis	2	'00': No emphasis '01': 50/15us '10': Reserved '11': CCITT J.17

Table 3.16 Bit Rates

Value	MPEG-1			MPEG-2		MPEG2.5
	Layer-1 [kbps]	Layer-2 [kbps]	Layer-3 [kbps]	Layer-1 [kbps]	Layer-2/3 [kbps]	Layer-3 [kbps]
'0000'	Free format (not supported by this middleware)					
'0001'	32	32	32	32	8	8
'0010'	64	48	40	48	16	16
'0011'	96	56	48	56	24	24
'0100'	128	64	56	64	32	32
'0101'	160	80	64	80	40	40
'0110'	192	96	80	96	48	48
'0111'	224	112	96	112	56	56
'1000'	256	128	112	128	64	64
'1001'	288	160	128	144	80	Forbidden*1
'1010'	320	192	160	160	96	Forbidden*1
'1011'	352	224	192	176	112	Forbidden*1
'1100'	384	256	224	192	128	Forbidden*1
'1101'	416	320	256	224	144	Forbidden*1
'1110'	448	384	320	256	160	Forbidden*1
'1111'	Forbidden (setting prohibited)*2					

[Note] *1: These bit rates for this product are treated the same as those for MPEG-2 Layer-3 as shown in Table 3.2.

*2: Not supported by this product.

Table 3.17 Sampling Frequencies

Value	MPEG-1 Audio Layer-1/2/3 [Hz]	MPEG-2 Audio Layer-1/2/3 [Hz]	MPEG2.5 Audio Layer-3 [Hz]
'00'	44100	22050	11025
'01'	48000	24000	12000
'10'	32000	16000	8000
'11'	Reserved (setting prohibited) *1		

[Note] *1: Not supported by this product.

Table 3.18 Relation between a bit rate and the mode(MPEG-1 Layer-2 only)

bit rate (kbits/s)	Allowed modes
free format*1	all modes
32	single_channel
48	single_channel
56	single_channel
64	all modes
80	single_channel
96	all modes
112	all modes
128	all modes
160	all modes
192	all modes
224	stereo, intensity stereo, dual channel
256	stereo, intensity stereo, dual channel
320	stereo, intensity stereo, dual channel
384	stereo, intensity stereo, dual channel

[Note] *1: Not supported by this product.

3.5.2 CRC

If the protection bit for a header indicates CRC is present, then two-byte CRC information follows the header. If that bit indicates CRC is absent, two-byte CRC information does not follow the header.

3.5.3 Side Information

Information required for MPEG Audio Layer-3 data decoding specifies not only the start position of audio data in the frame, but also the decoding method, etc. Table 3.19 shows the sizes of side information.

Table 3.19 Sizes of Side Information

	MPEG-1 Audio Layer-3 [byte]	MPEG-2 Audio Layer-3 [byte]	MPEG2.5 Audio Layer-3 [byte]
Monaural	17	9	9
Stereo	32	17	17

3.5.4 Main Data

Main data is related to audio samples. Main data for MPEG Audio Layer-1/2 follows the header or CRC information for each frame. Main data for MPEG Audio Layer-3 starts at the position specified by side information. Main data may start in the frame which includes side information indicating the main-data position, or it may start in previous frames. Main data does not always start in the immediately preceding frame. It may start in the second or subsequent preceding frame from the current position. For details about main data, refer to the specifications.

3.5.5 Ancillary Data

Ancillary data is user-defined data. It may or may not reside in a frame. It may include extended data for multi-channel processing. Note that this middleware skips ancillary data without processing it.

3.6 Output Data

16-bit linear PCM data is output. The number of samples varies depending on the standard parameters (MPEG-1/MPEG-2/MPEG2.5 and Layer-1/2/3).

Table 3.20 Number of Output Data Samples (per Channel)

mp3d_headerInfo::nID		mp3d_headerInfo::nLayer		
		0x0003 Layer-1	0x0002 Layer-2	0x0001 Layer-3
0x0001	MPEG-1	384	1152	1152
0x0000	MPEG-2	384	1152	576
0x0002	MPEG2.5	-	-	576

4. Precautions

This section provides precautions in creating an application.

4.1 Precautions in Calling a Function

The user program which calls a function in this middleware should follow the calling rules for the compiler used.

4.1.1 Function Execution Timing

The following describes the timing of executing functions in this middleware.

(1) mp3d_GetMemorySize function

You can execute this function at any time before executing the mp3d_Init function. Execute the mp3d_GetMemorySize function to reserve the necessary amount of memory.

(2) mp3d_Init function

Execute this function only once before starting a series of decode process steps. They make up a process from the start to the end of decoding a certain stream.

(3) mp3d_Decode function

Execute this function when you decode one frame of stream data.

(4) mp3d_GetErrorFactor function

You can execute this function at any time after executing the mp3d_Init function.

(5) mp3d_GetVersion function

You can execute this function at any time.

4.2 Other Precautions

4.2.1 Reserving and Allocating Memory Areas

Before calling a function in this middleware, reserve a static area, a scratch area, an input/output buffer area, and areas for structures which should hold the arguments of functions.

4.2.2 Access Outside a Memory Range

This middleware does not access memory space outside the reserved areas.

4.2.3 Combination with Other Applications

If you use this middleware together with other applications, be careful to avoid the duplication of symbol names.

4.2.4 Monitoring on the Performance

The products embedding this middleware shall observe performance of the middleware periodically with Watch Dog timer or such functions in order not to damage system performance.

Revision History	MP3 Decode Middleware User's Manual

Rev.	Date	Description	
		Page	Summary
1.00	July 31, 2014	-	First Edition issued
1.01	Aug. 29, 2014	3	Changed the format of "Table 1.2 Memory Size Requirements"
		52	Added Monitoring on the Performance in Section 4.2.4

MP3 Decode Middleware User's Manual

Publication Date: August 29, 2014 Rev.1.01

Published by: Renesas Electronics Corporation

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2880 Scott Boulevard Santa Clara, CA 95050-2554, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
1101 Nicholson Road, Newmarket, Ontario L3Y 9C3, Canada
Tel: +1-905-898-5441, Fax: +1-905-898-3220

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700, Fax: +44-1628-651-804

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-65030, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
7th Floor, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100083, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 LanGao Rd., Putuo District, Shanghai, China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1613, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2886-9318, Fax: +852 2886-9022/9044

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 2.0

MP3 Decode Middleware

Renesas Electronics Corporation