User's Manual

JPEG Decoder

User's Manual

Renesas Micro Computer Middleware

Renesas Electronics
www.renesas.com

Rev.1.03 2015.03

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.

Introduction

JPEG Decoder is a software library which compand a JPEG file. Sequential DCT-based is adopted.

This manual provides the information for making application programs by using the JPEG Decoder.

1. References

The JPEG Decoder is the software that is made based on the specification indicated by the following written standard. Refer to it together with this manual.

■ISO/IEC 10918-1 Information technology - Digital compression and coding of continuous-tone still images

■Code process and digital compression of JIS X 4301-1995 continuous-tone still image

2. Symbols and Terminologies

In this manual, it explains by using the symbols and terms shown below as there is no explanation specifically.

Table 1. Symbols

Notations	Description
Numeric	Described in decimal number as long as no explanations on this manual.

Table 2. Terminologies

Notation	Description
Original image	Original image expressed synthesizing brilliance (Y) and chrominance difference (Cb,Cr).
Component	Each data assembly showing brilliance (Y) and chrominance difference (Cb,Cr).
JPEG file	JPEG file which JPEG Decoder Library generates.
JPEG header	SOI marker of the JPEG files, the data of APPn substring, DQT substring, SOF0 substring, DHT substring, DRI substring, and SOS substring.
Image Data	Image data following SOS in a JPEG file.
Block	Unit of target data for processing when image is compressed or is expanded according to the JPEG Decoder library function.
marker	A two-byte code in which the first byte is hexadecimal FF and the second byte is a value between 1 and hexadecimal FE.
Substring	Shows the marker and the argument that continues to it.
DCT	Discrete Cosine Transform
IDCT	Inverse Discrete Cosine Transform

Notation	Description
DCT coefficient	Coefficient got by DCT (No quantization)
Quantization DCT coefficient	Coefficient got by DCT and quantization
MCU ; minimum coded unit	The smallest group of data units that is coded.
restart interval	The integer number of MCUsprocessed as an independent sequence within a scan.
restart marker ; RSTm	The marker that separates two restart intervals in a scan.
Huffman encoding	An entropy encoding procedure which assigns a variable length code to each input symbol.
Huffman decoding	An entropy decoding procedure which recovers the symbol from each variable length code produced by the Huffman encoder.
Huffman table	The set of variable length codes required in a Huffman encoding and Huffman decoding.
zig-zag sequence	A specific sequential ordering of the DCT cofficients from (approximately) lowest spatial frequency to highest.

3. Composition of Manual

■Chapter 1, “JPEG Decoder Overview”
Describes the overview of the JPEG Decoder.

■Chapter 2, “JPEG File Expand Library”
Explains the JPEG File Expand Library

■Chapter 3, “JPEG Decode Library”
Explains about JPEG Decode Library which performs basic operations, such as inverse quantization required for extension, reverse DCT, etc. of a JPEG picture.

Supplementary explanation

JPEG Decoder is preparing "the introductory guide " other than this manual for every correspondence microcomputer. There is a data in which notes for every ROM/RAM size and processing performance of a program, and correspondence microcomputer etc. were summarized. Please refer to it with this manual.

Table of Contents

List of Figures

List of Tables

1. JPEG Decoder Overview

This chapter describes the features and the overview of development procedure of JPEG Decoder.

1.1. Features of JPEG Decoder

1.1.1. JPEG Decoder Algorithm

JPEG Decoder is the software library based on the specification of “ISO/IEC10918-1” and “JIS X 4301”. The features of this library are as follows.

■Algorithm of sequential DCT-based

■Entropy coding using by Huffman code table

■The image expansion to three elements of 8bit accuracy are possible.

1.1.2. Function Overview

JPEG Decoder consists of two, JPEG File Expand Library which elongates a JPEG file and JPEG Decode Library which performs basic operation.

■JPEG File Expand Library
JPEG File Expand Library is a library for elongating a JPEG file to a bit-mapped image. It is used in combination with JPEG Decode Library. To control from JPEG File Expand Library to JPEG Decode Library, user only uses the library function of JPEG File Expand Library, and can get a bit-mapped image. Please refer to Chapter 2, “JPEG File Expand Library” for the details of JPEG File Expand Library.

■JPEG Decode Library
JPEG Decode Library performs the Huffman decryption,inverse quantization, and reverse DCT to compression image data. Please use it in combination with JPEG File Expand Library, or use it by an application program for mounting the extension portion of a JPEG file. Please refer to Chapter 3, “JPEG Decode Library” for the details of JPEG Decode Library.

Figure 1.1, “ Image Data Compression and Expansion ” shows the data flow.

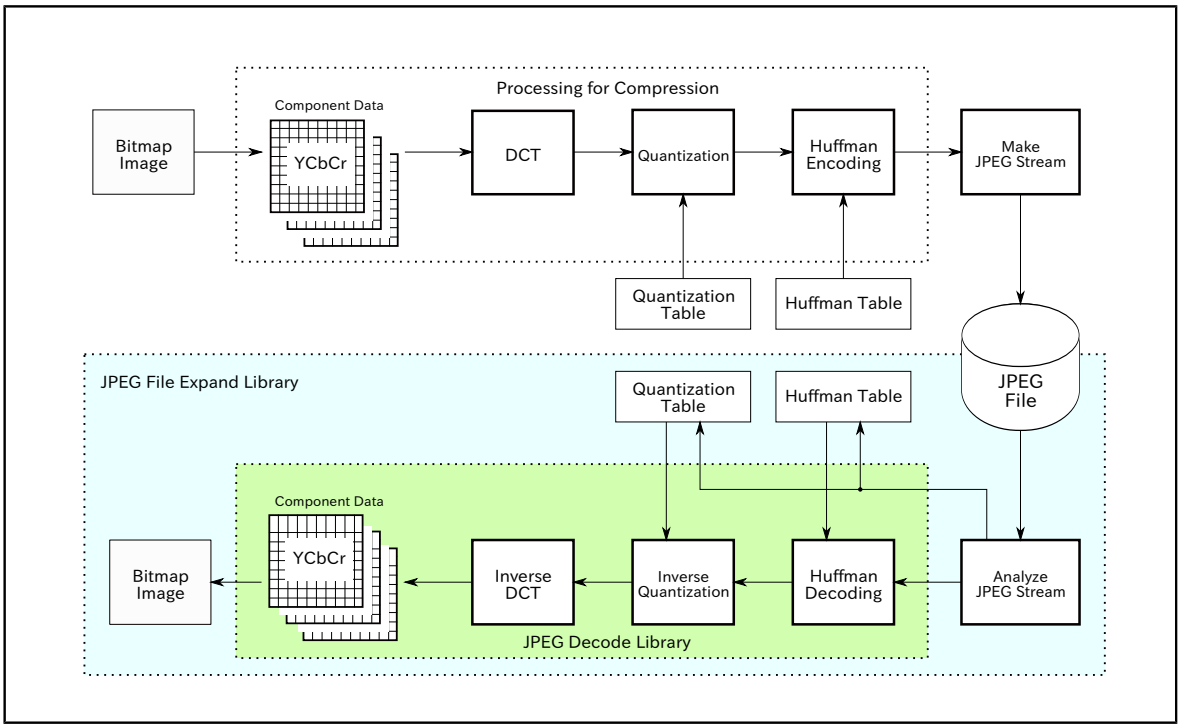

Figure 1.1. Image Data Compression and Expansion

Table 1.1, “Specification of JPEG File Expand Library” shows specification.

Table 1.1. Specification of JPEG File Expand Library

Item	Specification
Correspondence Format	Confirms JFIF Ver.1.1
Color element	Y,Cb,Cr
Sampling ratios	4:4:4 (1x1,1x1,1x1) 4:2:2 (2x1,1x1,1x1) 4:2:2 virtical (1x2,1x1,1x1) 4:2:0 (2x2,1x1,1x1)
Progressive	No support
Exif	No support
Output format	RGB565 (16bit color)
Clipping	No support (expand all)

As for JPEG File Expand Library is attached with source code, thereby a user can change specification.

Please refer to Section 2.6, “Source code information” about source code.

1.2. Program Development Procedure

Figure 1.2, “Development Flow of Application Program” shows the development flow.

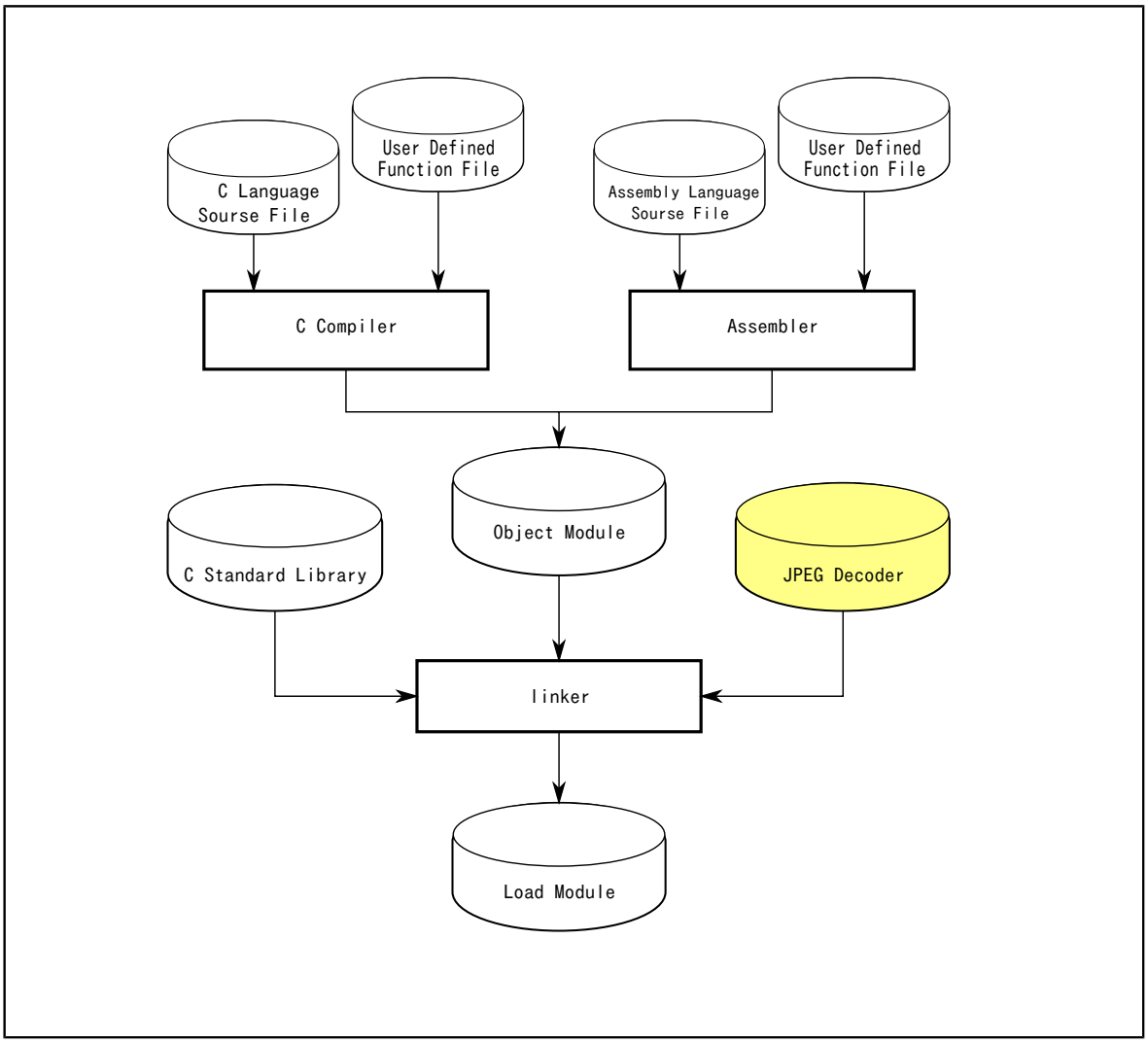

Figure 1.2. Development Flow of Application Program

2. JPEG File Expand Library

In this chapter, it describes about JPEG File Expand Library.

2.1. Overview

JPEG File Expand Library is a library for elongating a JPEG file to a bit-mapped image.

The library function supported by JPEG File Expand Library to Table 2.1, “Library Function List” .

Table 2.1. Library Function List

Function Name	Function Overview
R_init_jpeg	Initialization of JPEG Decoder
R_expand_jpeg	Expand JPEG file
R_get_info_jpeg	Get information of JPEG file

2.2. Composition

Figure 2.1, “Composition of JPEG File Expand Library” show the composition.

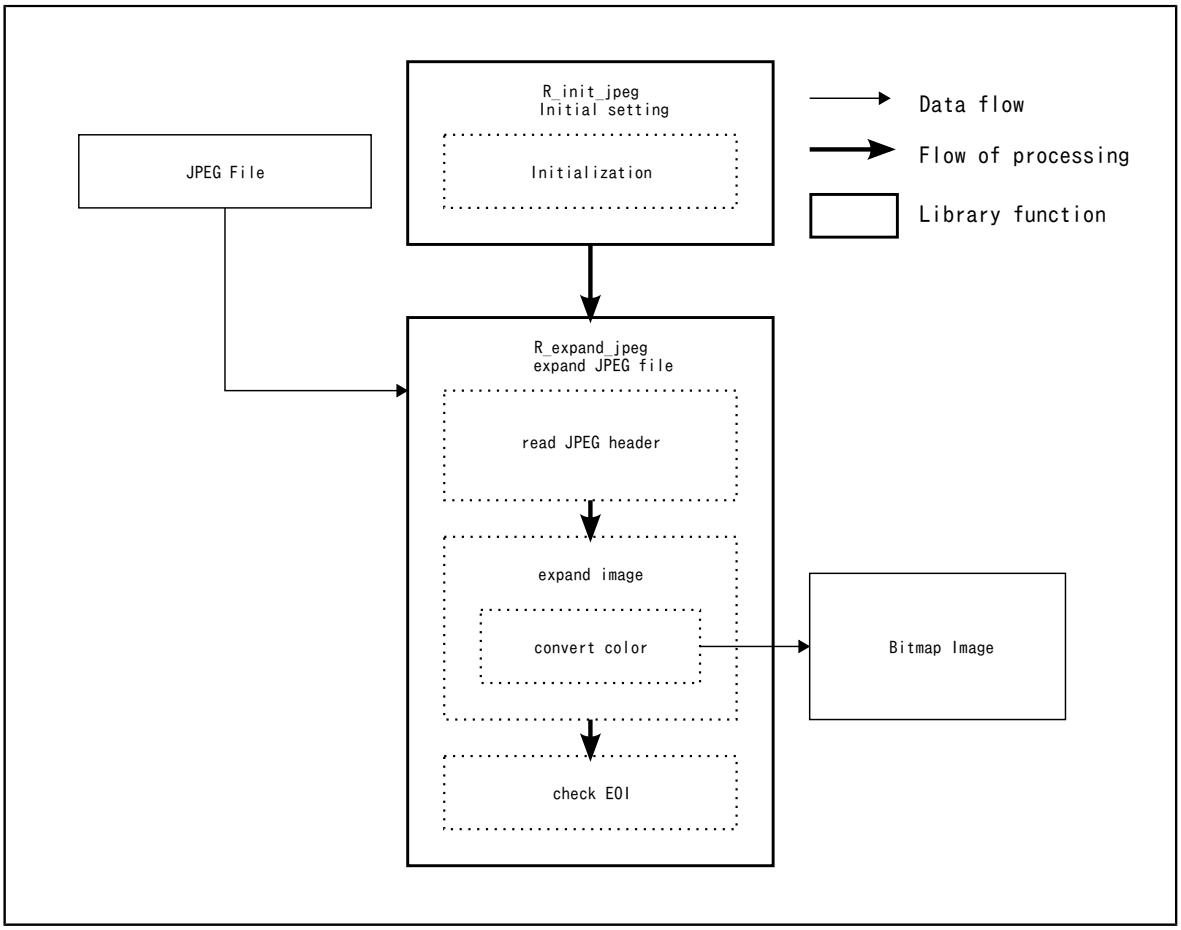

Figure 2.1. Composition of JPEG File Expand Library

2.3. Data structure

It explains the data structure which defined JPEG File Expand Library.

2.3.1. Library structure

JPEG File Expand Library uses the library structure defined by JPEG Decode Library. JPEG File Expand Library has secured all variables required for decoding of JPEG file. Please refer to Section 2.6, “Source code information” for secured variable.

2.3.2. Data input method

The data to input is specified by the argument of function R_expand_jpeg of JPEG File Expand Library. The size of JPEG data is specified for the beginning address of a domain which stores JPEG data as input by fsize.

These information is set up as an initial value of the input buffer of JPEG Decode Library and be managed. Please refer to Section 3.3.2, “Data Input Method” for an input buffer.

2.3.3. Macro Definition

This section describes the macro definition in the header file r_expand_jpegd.h.

Table 2.2. Error Code Definition

Definition	Value	Contents
EXPAND_JPEGD_OK	0	Normal end
EXPAND_JPEGD_ERROR_HEADER	-1	Header analysis error
EXPAND_JPEGD_ERROR_DECODE	-2	Expansion error
EXPAND_JPEGD_NOT_SUPPORT	-3	Without support
EXPAND_JPEGD_ERROR_RST	-4	RST detection error
EXPAND_JPEGD_ERROR_SOI	-5	SOI detection error
EXPAND_JPEGD_ERROR_EOI	-6	EOI detection error

2.4. Library Function Details

This section shows the details of each function of the JPEG File Expand Library. The way to description of each function is as follows.

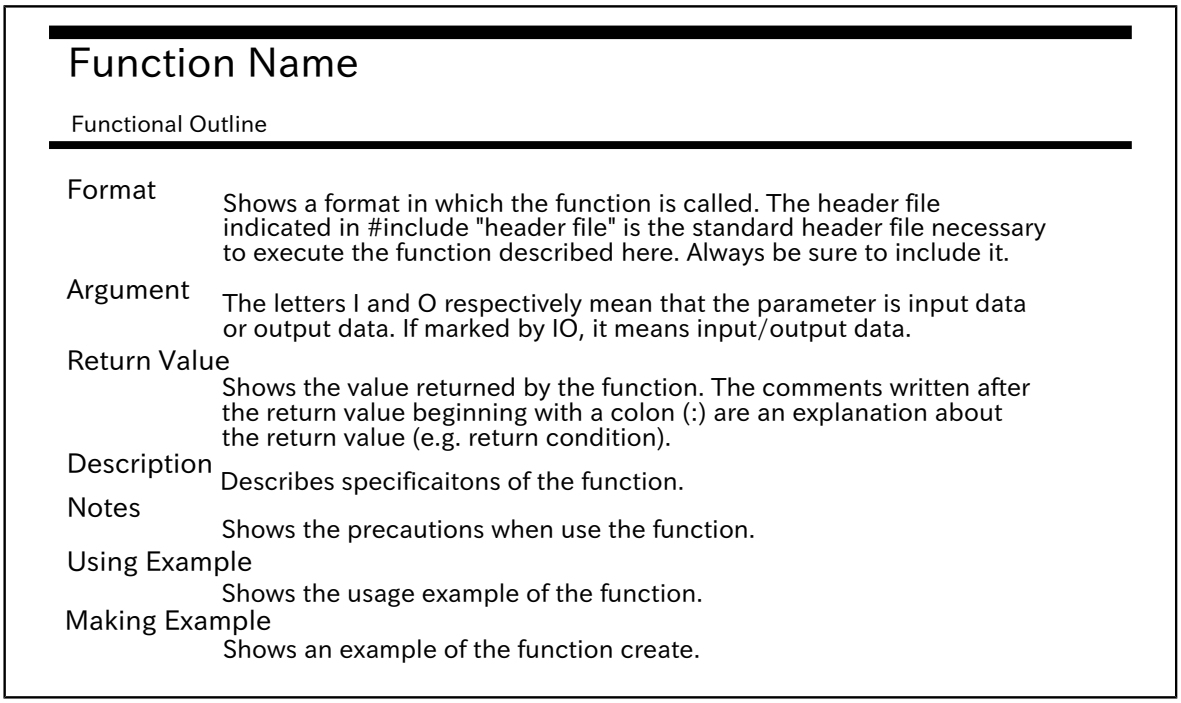

Figure 2.2. Description of Library Function Details

R_init_jpeg

Library Function

— Initialization of JPEG File Expand Library

Format

```
#include "r_expand_jpegd.h"

void R_init_jpeg (
                void  );
```

Argument

None

Return value

None

Description

This function initialize JPEG File Expand Library.

R_expand_jpeg

Library Function

— Expand JPEG file

Format

```
#include "r_expand_jpegd.h"

int16_t R_expand_jpeg (
                uint8_t *input ,
                uint32_t fsize ,
                uint16_t *outptr ,
                uint32_t offset );
```

Argument

argument name	I/O	explanation
input	I	Pointer to head of input data
fsize	I	Size of input data
outptr	O	Pointer to head of output data
offset	I	Number of pixels by 1 line in output area

Return value

Return value	explanation
0	Normal termination
Except 0	Error

Description

This function expands JPEG file specified argument "input", and stores data formatted RGB565 (1 pixel = 2 bytes) to area specified argument "output".

User specifies input data size to the argument "fsize". Input JPEG data needs continuous memory area.

User specified number of picture cells by 1 line in output area. For example, user specified value to argument "offset" 320, when the frame buffer has 320x240 (horizontal x vertical) pixels. Figure 2.3, “Output image of Bitmap” show the output image.

When the error occurred, this function would cancel processing and return with error code.

Notes

Output data area needs more size than expanded bitmap data. The image size after extension is acquirable by using R_get_info_jpeg.

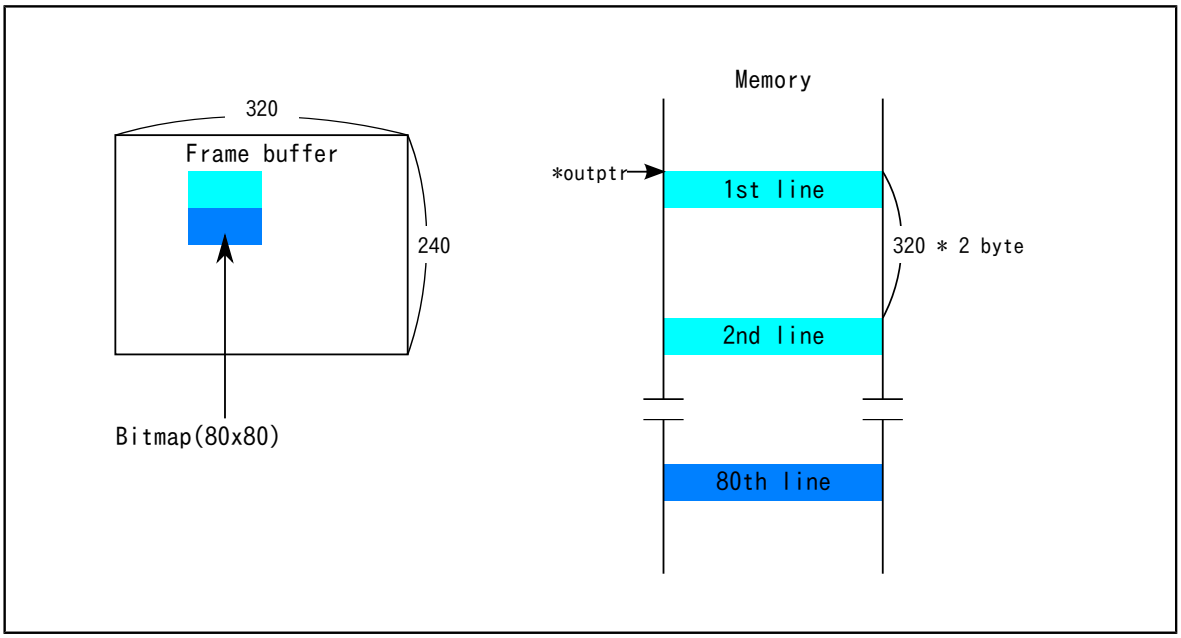

Figure 2.3. Output image of Bitmap

R_get_info_jpeg

Library Function

— Get information of JPEG file

Format

```
#include "r_expand_jpegd.h"

int16_t R_get_info_jpeg (
                uint8_t *input ,
                uint32_t fsize ,
                uint16_t *w ,
                uint16_t *h );
```

Argument

argument name	I/O	explanation
input	I	Pointer to head of input data
fsize	I	Size of input data
w	O	Pointer to data of picture width (number of pixel)
h	O	Pointer to data of picture height (number of pixel)

Return value

Return value	explanation
0	Normal termination
Except 0	Error

Description

This function judges specified picture file is JPEG file or not. If specified picture file is JPEG file, this function outputs picture size to argument "w", and "h".

If specified picture file is not JPEG file or has format error, this function would cancel processing and return with error code.

2.5. User Defined Function

A user-defined function is not in JPEG File Expand Library.

2.6. Source code information

The information on the source code of JPEG File Expand Library is shown.

Table 2.3. File List of JPEG File Expand Library

File Name	Overview
r_C_read_headers.c	Header analysis
r_expand.c	Extension processing of JPEG file
r_expand_jpegd_version.c	Version information
r_jpeg_read_input.c	The supplement of an input buffer (dummy function)
r_ycc2rgb.c	YCC->RGB conversion
r_expand_jpegd_version.c	Version Infomation
r_jpegd.h	The headre file of JPEG Decode Library
r_jpeg_maker.h	Marker definition
r_rgb2short.h	The macro definition of RGB565 conversion
r_expand_jpegd.h	The header file of JPEG File Expand Library
r_stdint.h	Data type header file
r_mw_version.h	Version data header file

Table 2.4. Function List of JPEG File Expand Library

Function Name	Overview
R_init_jpeg	Initialization of JPEG Decoder
R_expand_jpeg	Expand JPEG file
R_get_info_jpeg	Get information of JPEG file
decode444	Decode processing of YCbCr 4:4:4
decode422	Decode processing of YCbCr 4:2:2
decode422v	Decode processing of YCbCr 4:2:2 virtical
decode420	Decode processing of YCbCr 4:2:0
init_ycc444_outptr	Initialization of the output pointer of each component of YCbCr 4:4:4
init_ycc4xx_outptr	Initialization of the output pointer of each ingredient except YCbCr 4:4:4
init_last_outptr	Initialization of the output pointer for the last picture
_restart_marker	The judgement of a restart marker
_jpeg_open	Entry of JPEG file
_jpeg_read_header	Header analysis
_jpeg_readMarkers	A marker is detected and each processing is called

Function Name	Overview
_jpeg_readSOF0	SOF0 processing
_jpeg_readSOS	SOS processing
_jpeg_readAPP0	APP0 processing
_jpeg_readAPP14	APP14 processing, without real processing
_jpeg_readDHT	DHT processing
_jpeg_readDQT	DQT processing
_jpeg_skipMarkerSegment	Skip in reading of the data in a marker. Heder analysis is continuing
_jpeg_noSupportMarkers	Marker detection outside of support. Heder analysis is error finish
_jpeg_readEOI	EOI processing
_jpeg_readDRI	DRI processing
_jpeg_checkSOI	SOI check
_jpeg_checkEOI	EOI check
_jpeg_checkTableConsistency	Consistency check of table
ycc444_422v_rgb565	Output 1 line by RGB565, for YCbCr 4:4:4 and 4:2:2 virtical.
ycc422_420_rgb565	Output 1 line by RGB565, for YCbCr 4:2:2 and 4:2:0.
ycc2rgb	YCbCr is changed into RGB. 1 pixel
R_jpeg_read_input	The dummy of a user-defined function

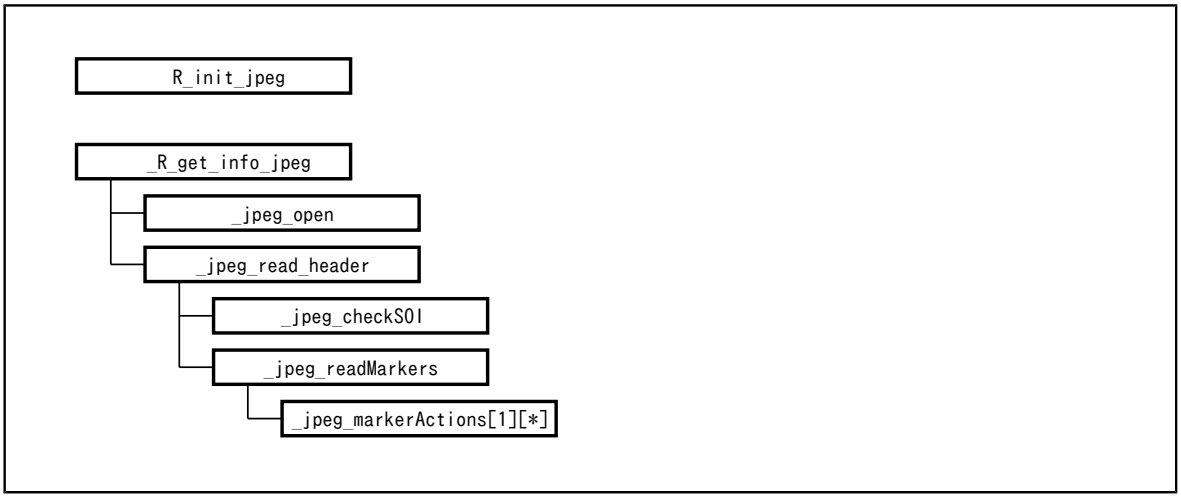

Figure 2.4. Function Tree (1/2)

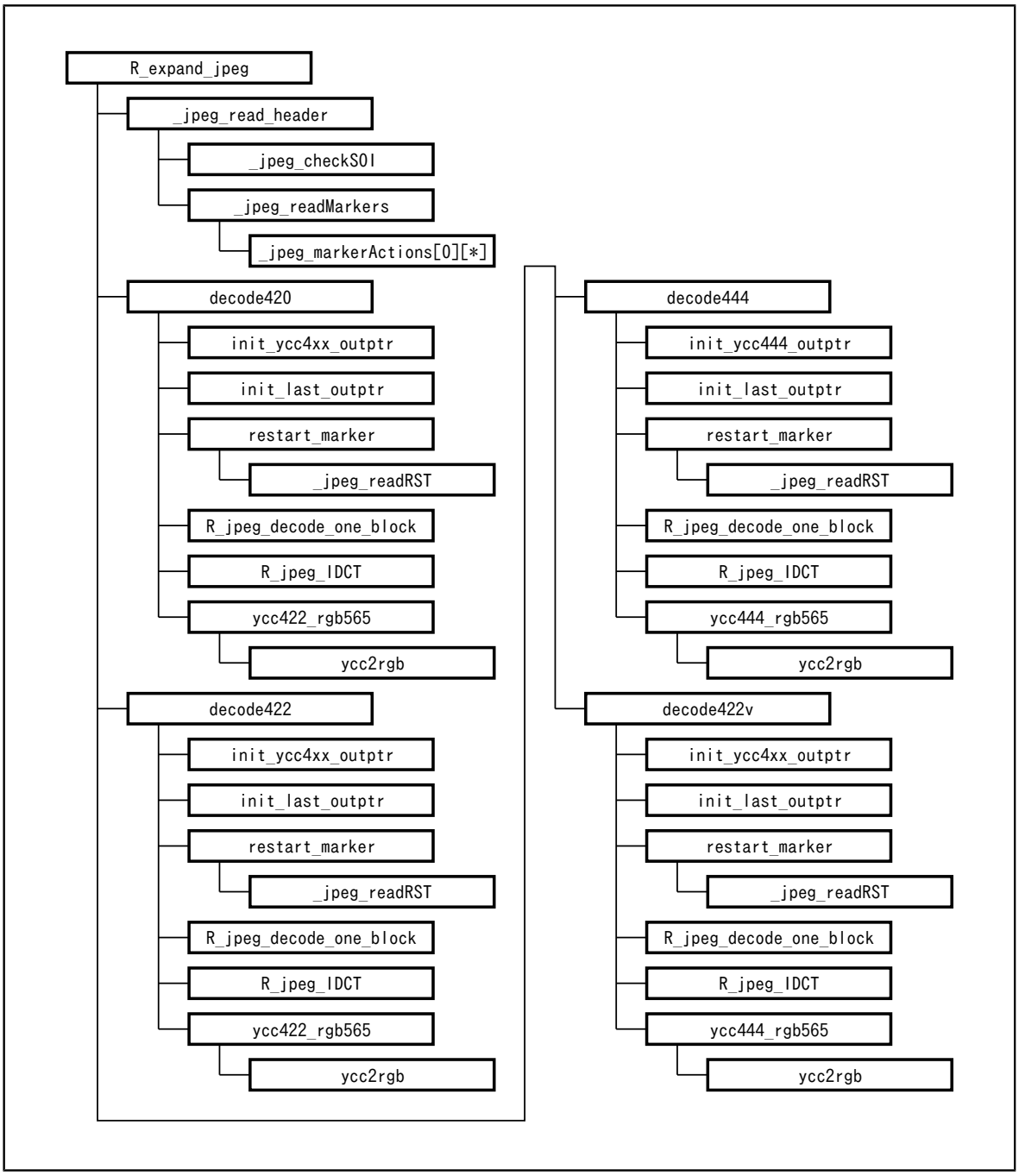

Figure 2.5. Function Tree (2/2)

Table 2.5. Global Variable

Variable Name	Use
working	Struct _jpeg_working of JPEG Decode Library
FMB	Struct _jpeg_dec_FMB of JPEG Decode Library
SMB	Struct _jpeg_dec_SMB of JPEG Decode Library

Variable Name	Use
align.mem.dtc_work[] (DTC_WORK[])	Work area for IDCT (8x8+2 word)
align.mem.Y[]	Decoding result of Y component ((8x8)x4 byte)
align.mem.Cb[]	Decoding result of Cb component (8x8 byte)
align.mem.Cr[]	Decoding result of Cr component (8x8 byte)
Y_last_dc_val	DC value of the last of Y component
Cb_last_dc_val	DC value of the last of Cb component
Cr_last_dc_val	DC value of the last of Cr component
*Y_outptr[]	Output place management of Y component (16 line)
*Cb_outptr[]	Output place management of Cb component (8 line)
*Cr_outptr[]	Output place management of Cr component (8 line)
*RGB_outptr[]	Output place management of a bit map (16+1 line)
MCU_count	MCU count for RST processing
next_restart_num	Next, the number of RST which should be detected (0-7)

3. JPEG Decode Library

In this chapter, it describes about JPEG Decode Library which performs basic operation.

3.1. Overview

JPEG Decode Library is a library which performs the Huffman decryption required for extension, inverse quantization and reverse DCT of a JPEG file.

The library function supported by JPEG Decode Library is shown as Table 3.1, “Library Function List” and a user-defined function is shown as Table 3.2, “User Defined Function List” .

Table 3.1. Library Function List

Function Name	Function Overview
R_jpeg_make_huff_table	Registration of Huffman table
R_jpeg_add_iquant_table	Registration of quantization table
R_jpeg_decode_one_block	Huffman decoding
R_jpeg_IDCT	Inverse quantization and Inverse DCT
R_jpeg_readRST	Detection of a restart marker

Table 3.2. User Defined Function List

User Defined Function Name (*)	Function Overview
R_jpeg_read_input	Input of JPEG file

*The function names can be freely changed.

3.2. Configuration

Figure 3.1, “Configuration of JPEG Decode Library” shows the example that Y component, Cb component and Cr component of 8x8 pixels are recovered repeatedly by using JPEG Decode Library from a JPEG file.

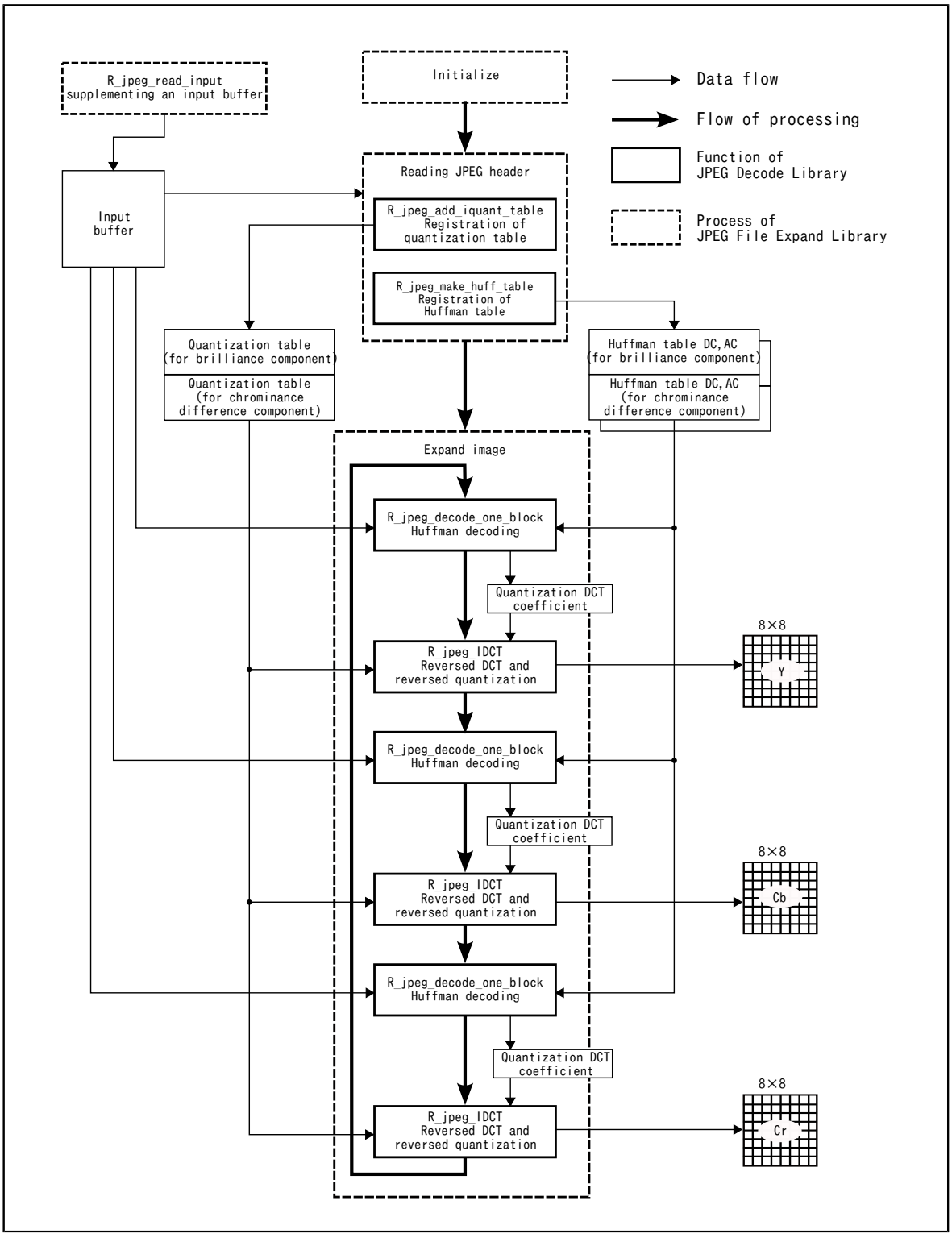

Figure 3.1. Configuration of JPEG Decode Library

3.3. Data Structure

It explains that the data structure which defined by JPEG Decode Library. The substance of a variable is not secured by JPEG Decode Library.

3.3.1. Library structure

3.3.1.1. JPEG Decode Library Environment Variable

In the JPEG Decoder, the data used in library is managed en bloc by one structure for easy access for the data used in library. In this manual, this structure name is called JPEG Decode Library environment variable structure.

Figure 3.2, “JPEG Decode Library Environment Variable _jpeg_working” shows the contents of structure.

```
struct  _jpeg_working{
    /* reserved */
    void *encFMB;
    void *encFMC;
    void *encSMB;
    void *encSMC;
    void (*enc_dump_func)(struct _jpeg_working *);

    /* decode */
    struct _jpeg_dec_FMB *decFMB;
    struct _jpeg_dec_FMC *decFMC;
    struct _jpeg_dec_SMB *decSMB;
    void (*dec_read_input)(struct _jpeg_working *);
};
```

Figure 3.2. JPEG Decode Library Environment Variable _jpeg_working

Please secure the domain for JPEG Decode Library Environment Variable by an application program. About a member given in "reserved", there is no necessity for reference by an application program and a setup. When using JPEG Decode Library together, JPEG Decode Library performs reservation of this domain, reference, and a set up.

3.3.1.2. JPEG Decode Library High Speed Memory Variable Group

The variable with high use frequency of the JPEG Decode Library is defined in structure _jpeg_dec_FMB. Figure 3.3, “JPEG Decode Library High Speed Memory Variable Structure _jpeg_dec_FMB” shows the contents of structure.

```
#define _JPEG_HUFFVAL_SIZE              256

struct _jpeg_dec_FMB
{
    uint8_t fmb1[512];                  /* reserved */

    uint16_t _jpeg_work[_JPEG_HUFFVAL_SIZE+1];    /* reserved */
    int16_t _jpeg_restart_interval;
    uint8_t *_jpeg_next_read_byte;
    int32_t _jpeg_d_free_in_buffer;
    uint32_t _jpeg_cur_read_buffer;
    uint32_t _jpeg_cur_bits_offset;

    uint8_t fmb2[3632];                 /* reserved */
};
```

Figure 3.3. JPEG Decode Library High Speed Memory Variable Structure _jpeg_dec_FMB

Table 3.3, “Member of Structure _jpeg_dec_FMB” lists the description of each member.

Table 3.3. Member of Structure _jpeg_dec_FMB

Structure Member Name	Sign	Function Overview
fmb1[]	-	reserved
_jpeg_work[]	-	reserved
_jpeg_restart_interval	Ri	Restart interval
*_jpeg_next_read_byte	-	Pointer indicating the encoded data for next read
_jpeg_d_free_in_buffer	-	Encoded data read in input buffer (Byte count)
_jpeg_cur_read_buffer	-	reserved
_jpeg_cur_bits_offset	-	reserved
fmb2[]	-	reserved

The sign on the table means the parameter symbol defined by bibliography.

3.3.1.3. JPEG Decode Library High Speed Memory Constant Group

The constant with high use frequency of the JPEG Decode Library is defined in the structure _jpeg_dec_FMC. Figure 3.4, “JPEG Decode Library High Speed Memory Constant Structure _jpeg_dec_FMC” shows the contents of structure.

```
/* for RX */
struct _jpeg_dec_FMC
{
    uint8_t fmc[1284];                  /* reserved */
};


/* for SH */
struct _jpeg_dec_FMC
{
    uint8_t fmc[1240];                  /* reserved */
};
```

Figure 3.4. JPEG Decode Library High Speed Memory Constant Structure _jpeg_dec_FMC

The constant group of _jpeg_dec_FMC is defined inside the JPEG Decode Library and can be referred by the symbol of _top_of_jpeg_dec_FMC from application program. Figure 3.5, “Initialization of Structure _jpeg_dec_FMC” shows the registration method to JPEG Decode Library environment variable structure.

```
#include ”r_jpegd.h”

sample(void)
{
                    …
    working.decFMC = (struct _jpeg_dec_FMC *)_top_of_jpeg_dec_FMC;
                    …
}
```

Figure 3.5. Initialization of Structure _jpeg_dec_FMC

Table 3.4, “Member of Structure _jpeg_dec_FMC” lists the description of each member.

Table 3.4. Member of Structure _jpeg_dec_FMC

Structure Member Name	Sign	Function Overview
fmc[]	-	reserved

The sign on the table means the parameter symbol defined by bibliography.

3.3.1.4. JPEG Decode Library Low Speed Memory Variable Group

The variable with low use frequency of the JPEG Decode Library is defined in the structure _jpeg_dec_SMB. Figure 3.6, “JPEG Decode Library Low Speed Memory Variable Structure _jpeg_dec_SMB” shows the contents of structure.

```
#define _JPEG_COMPONENT_NUM             3   /* YCbCr */

struct component_info
{
    uint8_t component_id[_JPEG_COMPONENT_NUM+1];    /* +1 for alignment */
    uint8_t hsample_ratio[_JPEG_COMPONENT_NUM+1];   /* +1 for alignment */
    uint8_t vsample_ratio[_JPEG_COMPONENT_NUM+1];   /* +1 for alignment */
    uint8_t quant_tbl_no[_JPEG_COMPONENT_NUM+1];    /* +1 for alignment */
};

struct frame_component_info
{
    uint8_t component_id[_JPEG_COMPONENT_NUM+1];    /* +1 for alignment */
    uint8_t dc_tbl_no[_JPEG_COMPONENT_NUM+1];       /* +1 for alignment */
    uint8_t ac_tbl_no[_JPEG_COMPONENT_NUM+1];       /* +1 for alignment */
};

struct _jpeg_dec_SMB
{
    int32_t _jpeg_d_num_QTBL;
    int32_t _jpeg_thinning_mode;
    uint16_t _jpeg_d_number_of_lines;
    uint16_t _jpeg_d_line_length;

    int16_t _jpeg_X_density;
    int16_t _jpeg_Y_density;
    int32_t _jpeg_d_frame_num_of_components;
    struct frame_component_info frame_component_info;
    int32_t _jpeg_error_stat;
    uint8_t _jpeg_d_precision;
    int32_t _jpeg_d_num_of_components;
    struct component_info component_info;
    int32_t flagStreamHeader[3];
};
```

Figure 3.6. JPEG Decode Library Low Speed Memory Variable Structure _jpeg_dec_SMB

The area of _jpeg_dec_SMB should be retained by application program.

Table 3.5, “Member of Structure _jpeg_dec_SMB” lists the description of each member.

Table 3.5. Member of Structure _jpeg_dec_SMB

Structure Member Name	Sign	Function Overview
_jpeg_num_QTBL	-	Number of quantization table
_jpeg_thinning_mode	-	reserved
_jpeg_d_number_of_lines	Y	Line count of original image
_jpeg_d_line_length	X	Pixel count per one line

Structure Member Name	Sign	Function Overview
_jpeg_X_density	-	reserved
_jpeg_Y_density	-	reserved
_jpeg_d_frame_num_of_components	-	Number of components inside the scan
struct frame_component_info { uint8_t component_id[]; uint8_t dc_tbl_no[]; uint8_t ac_tbl_no[]; } frame_component_info	-	Information about flame Members of structure are described as follows.
component_id[]	Csj	Identifier of component
dc_tbl_no[]	Tdj	Arrangement storing DC table number (0,1) of Huffman table that component uses
ac_tbl_no[]	Taj	Arrangement storing AC table number (0,1) of Huffman table that component uses
_jpeg_error_stat	-	Variable storing error code
_jpeg_d_precision	-	reserved
_jpeg_d_num_of_components	Ns	Arrangement storing the number of components inside the scan
struct component_info { uint8_t component_id[]; uint8_t hsample_ratio[]; uint8_t vsample_ratio[]; uint8_t quant_tbl_no[]; } component_info	-	Information about color component Members of structure are described as follows
component_id[]	Ci	Identifier of component
hsample_ratio[]	Hi	Horizontal extraction rate of component
vsample_ratio[]	Vi	Vertical extraction rate of component
quant_tbl_no[]	Tqi	Quantization table number that component uses
flagStreamHeader[]	-	Flag for checking a JPEG header

The sign on the table means the parameter symbol defined by bibliography.

3.3.2. Data Input Method

It is necessary that the data to input is stored in the RAM domain set up as an input buffer of JPEG Decode Library or the ROM domain where the data to input is stored is set up as an input buffer of JPEG Decode Library. An input buffer is a domain on the address space where some or all of data of JPEG file is stored.

■How to Manage the Output Buffer
Input buffer is controlled by two members of structure _jpeg_dec_FMB.

```
uint8_t *_jpeg_next_read_byte;   // Position for reading the next data
int32_t _jpeg_d_free_in_buffer;  // Number of enabled data (byte count)
```

One is added as for _jpeg_next_read_byte, and one is subtracted as for _jpeg_free_in_buffer every time the data of one byte is read from input buffer.

■Reservation of an input buffer
Please ensure the domain of an input buffer by an application program. 4 bytes is a reservation domain of the JPEG decoding library from the head of the input buffer. It is necessary to initialize these 4 bytes by 0xFF. Please store the data of a JPEG file from the 5th byte. The 5th byte or subsequent ones becomes effective data size.

■Processing when effective data in an input buffer is lost (buffering function)
When there is no effective data in an input buffer at the time of data read-out, a user-defined function is called. The division input of the JPEG file can be carried out by supplementing an input buffer with the data of a continuation within this function. The user-defined function can register arbitrary functions for every work domain management in R_init_jpeg.

When all the data of a JPEG file is arranged at the address on memory space, such as ROM, It can consider that a head address with a JPEG file is an input buffer, and carry out a mass entry. At this time, 0xFF domain of 4 bytes of head is unnecessary.

Supplementary explanation

The buffering function of JPEG Decode Library is not used in JPEG File Expand Library.

3.3.3. Macro Definition

This section describes the macro definition in the header file r_jpegd.h.

Table 3.6. Error code definition

Define	Value	Contents
_JPEGD_OK	0	Normal end
_JPEGD_ERROR	-1	Error close

Table 3.7. Constant definition

Define	Value	Contents
_JPEG_DCTSIZE	8	DCT size
_JPEG_DCTSIZE2	64	Squaring of DCT size
_JPEG_COMPONENT_NUM	3	Number of components
_JPEG_HUFFVAL_SIZE	256	Number of Huffman code
_JPEG_BITS_SIZE	17	Number of Huffman bit

Table 3.8. Macro Variable Definition (High Speed Memory Variable Group for Expansion Library _jpeg_dec_FMB)

Define	Contents
_jpeg_work(base)	((base)->_jpeg_work)

Define	Contents
_jpeg_next_read_byte(base)	((base)->_jpeg_next_read_byte)
_jpeg_d_free_in_buffer(base)	((base)->_jpeg_d_free_in_buffer)
_jpeg_cur_read_buffer(base)	((base)->_jpeg_cur_read_buffer)
_jpeg_cur_bits_offset(base)	((base)->_jpeg_cur_bits_offset)
_jpeg_restart_interval(base)	((base)->_jpeg_restart_interval)

Table 3.9. Macro Variable Definition (Low Speed Memory Variable Group for Expansion Library _jpeg_dec_SMB)

Define	Contents
_jpeg_d_num_QTBL(base)	((base)->_jpeg_d_num_QTBL)
_jpeg_d_number_of_lines(base)	((base)->_jpeg_d_number_of_lines)
_jpeg_d_line_length(base)	((base)->_jpeg_d_line_length)
_jpeg_d_precision(base)	((base)->_jpeg_d_precision)
_jpeg_X_density(base)	((base)->_jpeg_X_density)
_jpeg_Y_density(base)	((base)->_jpeg_Y_density)
_jpeg_d_num_of_components(base)	((base)->_jpeg_d_num_of_components)
_jpeg_d_frame_num_of _components(base)	((base)->_jpeg_d_frame_num_of_components)
_jpeg_error_stat(base)	((base)->_jpeg_error_stat)
component_info(base)	((base)->component_info)
frame_component_info(base)	((base)->frame_component_info)

Table 3.10. Macro function of data reading

Define	Contents
CHECK_BUFF(env, fmb)	Check Input buffer
INPUT_BYTE(var, env, fmb)	Reading of 1-byte length data
INPUT_2BYTES(var, env, fmb)	Reading of 2-byte length data
READ_NBYTE(p, n, env, fmb)	n-byte reading
SKIP_BYTES(n, env, fmb)	n-byte skipped reading
READ_LENGTH(len, env, fmb)	Reading of length

When you read data from an input buffer, please be sure to use these macroscopic functions.

Table 3.11. The macro definition for stream check

Define	Contents
STREAM_HEADER_FLAG_CLEAR (env)	The flag clearance for stream check
STREAM_HEADER_FLAG_SET(env, n)	The flag set for stream check
STREAM_HEADER_DQT_SET(env, n)	The DQT flag set for stream check

Define	Contents
DQT_BITS	4: Number of Secured Bits for DQT
DHT_INDEX2N(n)	Index of DHT(n) is changed into the bit information on a flag.
STREAM_HEADER_DHT_SET(env, n)	The DHT(n) flag set for stream check
STREAM_HEADER_CHECK_ QUNAT_TABLE(env, n)	The compatibility check of quantization table(n)
STREAM_HEADER_CHECK_ DC_TABLE(env, n)	The compatibility check of DC Huffman table(n)
STREAM_HEADER_CHECK_ AC_TABLE(env, n)	The compatibility check of AC Huffman table(n)
STREAM_HEADER_MUST_SET0	The definition 1 of an indispensable header
STREAM_HEADER_MUST_SET1	The definition 2 of an indispensable header
STREAM_HEADER_CHECK(f,v)	The existence check of header(v)
STREAM_HEADER_CHECK_SOF0 (flag)	The existence check of SOF0
STREAM_HEADER_CHECK_SOS (flag)	The existence check of SOS
STREAM_HEADER_FLAG_CHECK (env)	The existence check of an indispensable header

3.4. Library Function Details

This section shows the details of each function of the JPEG File Expand Library. The way to description of each function is as follows.

Function Name

Functional Outline

Format
Shows a format in which the function is called. The header file indicated in #include "header file" is the standard header file necessary to execute the function described here. Always be sure to include it.

Argument
The letters I and O respectively mean that the parameter is input data or output data. If marked by IO, it means input/output data.

Return Value
Shows the value returned by the function. The comments written after the return value beginning with a colon (:) are an explanation about the return value (e.g. return condition).

Description
Describes specificaitons of the function.

Notes
Shows the precautions when use the function.

Using Example
Shows the usage example of the function.

Making Example
Shows an example of the function create.

Figure 3.7. Description of Library Function Details

R_jpeg_make_huff_table

Library Function

— Registration of Huffman table

Format

```
#include "r_jpegd.h"

int16_t R_jpeg_make_huff_table (
                uint8_t index ,
                uint8_t *huffval ,
                uint8_t *bits ,
                int16_t count ,
                struct _jpeg_working *wenv );
```

Argument

argument name	I/O	explanation
index	I	Huffman table number (Tc/Th)
huffval	I	Pointer to the area in which Value associated with each Huffman code (Vij) is stored
bits	I	Pointer to the area in which Length of Huffman code (Li) is stored
count	I	The number of data to register
wenv	I/O	Pointer to JPEG Decode Library environment variable structure

Return value

Return value	explanation
0	Normal termination
Except 0	Error

Description

In the JPEG Decode Library environment specified with wenv, this function registers the data registered in the Define Huffman table (DHT) of JPEG file.

In index, the table class is specified by high four bits and the Huffman table destination identifier is specified by low four bits. 0 (DC table) and 1 (AC table) are specified by the table class. The Huffman table destination identifier specifies the table number (0, 1) of Huffman table. The operation when the value except above is specified to index is indeterminate.

Pointer to the area in which Value associated with each Huffman code (Vij) is stored is specified to huffval.

Pointer to the area in which Length of Huffman code (Li) is stored is specified to bits

The number of data to register is specified to count

R_jpeg_add_iquant_table

Library Function

— Registration of quantization table

Format

```
#include "r_jpegd.h"

int16_t R_jpeg_add_iquant_table (
                int16_t qtbl_no ,
                uint16_t *qtbl ,
                struct _jpeg_working *wenv );
```

Argument

argument name	I/O	explanation
qtbl_no	I	Quantization table destination identifier (Tq)
qtbl	I	Pointer to the area in which Quantization table elemant (Qk) is stored
wenv	I/O	Pointer to JPEG Decode Library environment variable structure

Return value

Return value	explanation
0	Normal termination
Except 0	Error

Description

In the JPEG Decode Library environment specified with wenv, this function registers the data defined in the Define quantization table (DQT) of a JPEG file.

Quantization table destination identifier (Tq) are specified to qtbl_no. The operation which the value except 0, 1, and 2 is specified to qtbl_no is indeterminate.

The pointer to the area (uint16_t type, array of 8x8) in which Quantization table elemant (Qk) registering in this function is stored is specified to qtbl. Pass the quantization table to this function like the order of the zigzag sequence.

Limitations (For SH-2A)

Please arrange the address of qtbl to become a multiple of 4.

R_jpeg_decode_one_block

Library Function

— Huffman decoding

Format

```
#include "r_jpegd.h"

int16_t R_jpeg_decode_one_block (
                int16_t last_dc_val ,
                int16_t dc_tbl_no ,
                int16_t ac_tbl_no ,
                int16_t *block ,
                struct _jpeg_working *wenv );
```

Argument

argument name	I/O	explanation
last_dc_val	I	DC coefficient of front block
dc_tbl_no	I	DC table number of Huffman table
ac_tbl_no	I	AC table number of Huffman table
block	O	Pointer to storage area of quantization DTC coefficient in Huffman decoded
wenv	I/O	Pointer to JPEG Decode Library environment variable structure

Return value

Return value	explanation
0	Normal termination
Except 0	Error

Description

In the JPEG Decode Library environment specified with wenv, this function reads a JPEG file from input buffer and stores the Huffman decoding data (quantization DCT coefficient) in the area specified with block.

DC coefficient of front block is specified to last_dc_val.

DC table numbers of Huffman table (0 and 1) are specified to dc_tbl_no. Specify 0 when the target of processing is Y (brilliance). Specify 1 when the target of processing is Cb or Cr.

The operation when the value except 0 and 1 is specified to dc_tbl_no is indeterminate.

AC table numbers of Huffman table (0 and 1) are specified to ac_tbl_no. Specify 0 when the target of processing is Y (brilliance). Specify 1 when the target of processing is Cb or Cr.

The operation when the value except 0 and 1 is specified to ac_tbl_no is indeterminate.

The area (64+2 bytes) in which the Huffman decoding data (quantization DCT coefficient) output from this function is stored is specified to block. Clear 0 of the area specified by block before this function is executed.

Note

When input buffer is lost while this function is being executed, the User Defined Function R_jpeg_read_input is called and input buffer is initialized. After the termination of execution of the _v_read_input function, execution of this function is continued. Refer to "User Defined Function" on R_jpeg_read_input for the specification of _jpeg_read_input function. Before this function is executed, the Huffman code table needs to be registered by the R_jpeg_make_huff_table function.

Limitations (For SH-2A)

Please arrange the address of block to become a multiple of 4.

Using Example

```
#include "r_jpegd.h"

#define DCT_WORK    align.mem.dct_work

static union {
  uint32_t dummy;
  struct {
    int16_t dct_work[_JPEG_DCTSIZE2 + 2];
    uint8_t  Y[_JPEG_DCTSIZE2 * 4];
    uint8_t Cb[_JPEG_DCTSIZE2];
    uint8_t Cr[_JPEG_DCTSIZE2];
  } mem;
} align;
static uint8_t *Y_outptr[_JPEG_DCTSIZE*2];
static uint8_t *Cb_outptr[_JPEG_DCTSIZE];
static uint8_t *Cr_outptr[_JPEG_DCTSIZE];
static int16_t Y_last_dc_val, Cb_last_dc_val, Cr_last_dc_val;

void
sample(void)
{
      ...
  for (i = 0; i < lines; ++i)
  {
    for (j = 0; j < width; ++j)
    {
      //  Y0
      ret = R_jpeg_decode_one_block(Y_last_dc_val, Y_dc_tbl_no, Y_ac_tbl_no, DCT_WORK, wenv);
      if(ret != _JPEGD_OK)
      {
        return EXPAND_JPEGD_ERROR_DECODE;
      }
      Y_last_dc_val = DCT_WORK[0];
      R_jpeg_IDCT(DCT_WORK, Y_outptr, 0, Y_q_tbl_no, wenv);
      //  Y1
      ret = R_jpeg_decode_one_block(Y_last_dc_val, Y_dc_tbl_no, Y_ac_tbl_no, DCT_WORK, wenv);
      if(ret != _JPEGD_OK)
      {
        return EXPAND_JPEGD_ERROR_DECODE;
      }
```

```
        Y_last_dc_val = DCT_WORK[0];
        R_jpeg_IDCT(DCT_WORK, Y_outptr, _JPEG_DCTSIZE, Y_q_tbl_no, wenv);
        //  Cb
        ret = R_jpeg_decode_one_block(Cb_last_dc_val, Cb_dc_tbl_no, Cb_ac_tbl_no, DCT_WORK, wenv);
        if(ret != _JPEGD_OK)
        {
          return EXPAND_JPEGD_ERROR_DECODE;
        }
        Cb_last_dc_val = DCT_WORK[0];
        R_jpeg_IDCT(DCT_WORK, Cb_outptr, 0, Cb_q_tbl_no, wenv);
        //  Cr
        ret = R_jpeg_decode_one_block(Cr_last_dc_val, Cr_dc_tbl_no, Cr_ac_tbl_no, DCT_WORK, wenv);
        if(ret != _JPEGD_OK)
        {
          return EXPAND_JPEGD_ERROR_DECODE;
        }
        Cr_last_dc_val = DCT_WORK[0];
        R_jpeg_IDCT(DCT_WORK, Cr_outptr, 0, Cr_q_tbl_no, wenv);
                    ...
      }
    }
  }
```

R_jpeg_IDCT

Library Function

— Inverse DCT and Inverse quantization

Format

```
#include "r_jpegd.h"

void R_jpeg_IDCT (
                int16_t *block ,
                uint8_t *outptr[] ,
                int16_t start_col ,
                int16_t qtbl_no ,
                struct _jpeg_working *wenv );
```

Argument

argument name	I/O	explanation
block	I	Pointer to the storage area of quantization DCT coefficient in Huffman decoded
outptr	I	Array in which the first address of line storing the processing result is stored
start_col	I	Offset value to the processing target block from the top of line
qtbl_no	I	Quantization table number
wenv	I/O	Pointer to JPEG Decode Library environment variable structure

Return value

None

Description

In the JPEG Decode Library environment specified with wenv, this function inputs the Huffman decoding data (quantization DCT coefficient) stored in the area specified with block and executes the inverse quantization and inverse DCT, and stores the data of Y, Cb, and Cr components in the area specified with outptr and start_col to.

The pointer to the area in which the Huffman decoding data (quantization DCT coefficient) of processing target is stored is specified to block.

The pointer to the pointer array to the line storing the component data is specified to outptr. In other words, the data of component is stored in the line to which outptr[0], outptr[1] and so forth separately indicate. The number of elements that the array of outptr needs is 8.

The offset from the top of line specified with outptr to the position in which output data is written is specified to start_col.

The quantization table numbers (0 and 1) are specified to qtbl_no. The operation when the value except 0 and 1 is specified to qtbl_no is indeterminate.

Note

Before this function is executed, the registration of quantization table by the R_jpeg_add_iquant_table function is necessary.

Limitations (For SH-2A)

Please arrange the address of block to become a multiple of 4.

Please arrange the line which stores the processing result specified by outptr so that a beginning address serves as a multiple of 4. Moreover, please specify the value of start_col by the multiple of 4.

Using Example

Figure 3.8, “Relationship Between Argument outptr and start_col of the R_jpeg_IDCT Function” shows the examples when the inverse quantization and inverse DCT are executed.

When it is described as the example 1, the component played is stored in block 1. When it is described as the example 2, the component played is stored in block 6.

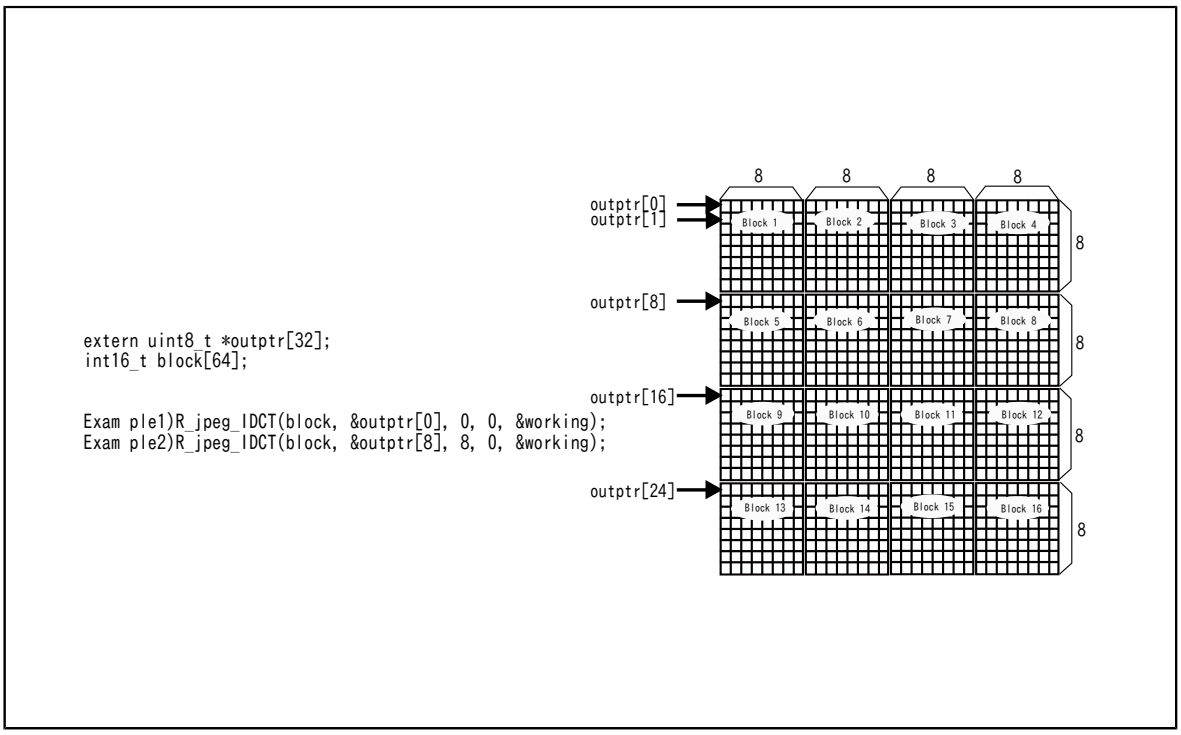

Figure 3.8. Relationship Between Argument outptr and start_col of the R_jpeg_IDCT Function

R_jpeg_readRST

Library Function

— Detection of a restart marker

Format

```
#include "r_jpegd.h"

int16_t R_jpeg_readRST (
                struct _jpeg_working *wenv );
```

Argument

argument name	I/O	explanation
wenv	I/O	Pointer to JPEG Decode Library environment variable structure

Return value

Return value	explanation
-1	Error
0 ~ 7	Normal termination

Description

In the JPEG Decode Library environment specified with wenv, this function reads the restart marker (RSTm) and returns the number of restart marker (m) as the return value.

As for the number of return value, the value of modulo-8 sequentially repeated to 0 ~ 7 is returned. The error of data can be detected by comparing the return value with the expected value.

3.5. User Defined Function

When using JPEG Decode Library, it needs user defined function, R_jpeg_read_input.

R_jpeg_read_input

User Defined Function

— JPEG File Input

Format

```
#include "r_jpegd.h"

void R_jpeg_read_input (
                struct _jpeg_working *wenv );
```

Argument

argument name	I/O	explanation
wenv	I/O	Address pointing to JPEG Decode Library environment variable structure

Return value

None

Description

This function is called when enabled data of input buffer is lost, the data of JPEG file is refilled to input buffer by application.

In this function, Please supply data of a JPEG file to input buffer, and update arameter of input buffer, use _jpeg_d_free_in_buffer() and _jpeg_next_read_byte(). If data carry out a mass entry, There is no processing with this function.

Notes

The first 4-byte of input buffer is used by the library. It is necessary to initialize it with 0xFF. Please store the data to input from the 5th byte of input buffer.

Making Example

```
void _jpeg_read_input(struct _jpeg_working *wenv)
{
    /* The number of data (byte) read in input buffer is set */
    _jpeg_d_free_in_buffer(wenv->decFMB) = JPG_BUFSIZE;

    /* The position of data read next from input buffer is set (The first 4-byte is
                used by the library) */
    _jpeg_next_read_byte(wenv->decFMB)   = _input_buf + 4;

    /* The new data is read in input buffer */
    transmit_data(src_address, (int32_t)_input_buf+4, JPG_BUFSIZE);

    /* Source address to transfer is updated */
    src_address += JPG_BUFSIZE;
}
```

JPEG Decoder
User's Manual

Publication Date 17 Mar 2015 Rev.1.03
17 Mar 2015 Rev.1.03
Publisher Renesas Electronics Corporation
1753 Shimonumabe, Nakahara-ku, Kawasaki,
Kanagawa 211-8668, Japan

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

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

California Eastern Laboratories, Inc.
4590 Patrick Henry Drive, Santa Clara, California 95054-1817, U.S.A.
Tel: +1-408-919-2500, Fax: +1-408-988-0279

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, 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 Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, 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 1207, 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 India Pvt. Ltd.
No.777C, 100 Feet Road, HALII Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

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 4.0

JPEG Decoder
User's Manual

Renesas Electronics Corporation

R20UW0075EJ0103