User's Manual

32

VSP Manager for Linux

User's Manual: Software

R-Car H2/M2 Series

Renesas Electronics
www.renesas.com

Rev.1.00 May. 2014

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 designed to provide the user with an understanding the functions of this software to management VSP and 2DDMAC H/W resource and for the reference manual to develop systems implementing image extraction function. This manual is written for engineers who use this VSP management functions with VSP and 2DDMAC.

Particular attention should be paid to the precautionary notes when using the manual. These notes occur within the body of the text, at the end of each section, and in the Usage Notes section.

The revision history summarizes the locations of revisions and additions. It does not list all revisions. Refer to the text of the manual for details.

Please refer to documents of software and hardware for a target system implementing this VSP Manager as necessary.
The following documents are related documents. Make sure to refer to the latest versions of these documents. The newest versions of the documents listed may be obtained from the Renesas Electronics Web site.

Document Type	Description	Document Title	Notes
User's manual for Hardware	Hardware specifications (pin assignments, memory maps, peripheral function specifications, electrical characteristics, timing charts) and operation description Note: Refer to the application notes for details on using peripheral functions.	R-Car {H/M}2 User's Manual: Hardware	
User's manual for Software	Description of VSP manager	VSP Manager User's Manual	This manual

2. Notation of Numbers and Symbols

This manual use following nation.

Binary	0bXXXXXXXX	(X = 0 or 1)
Decimal	XXX	(X = 0 to 9)
Hex	0xXXXXXXXX	(X = 0 to 9, A to F)

3. List of Abbreviations and Acronyms

Abbreviation	Full Form
VSP	Video Signal Processor
2DDMAC	2D Direct Memory Access Controller
RPF	Read Pixel Formatter
WPF	Write Pixel Formatter
SRU	Super Resolution Unit
UDS	Up Down Scaler
LUT	Look Up Table
CLU	Cubic Look Up table
HST	Hue Saturation value Transform
HSI	Hue Saturation value Inverse transform
HGO	Histogram Generator-One dimension
HGT	Histogram Generator-Two dimension
BRU	Blend ROP Unit
ROP	Raster OPration

Table of Contents

1. Overview

1.1. Overview of the Software

This document describes how to use of VSP manager.
VSP manager is software with the management of VSP and 2DDMAC resources so that more than one application can use VSP and 2DDMAC at the same time.

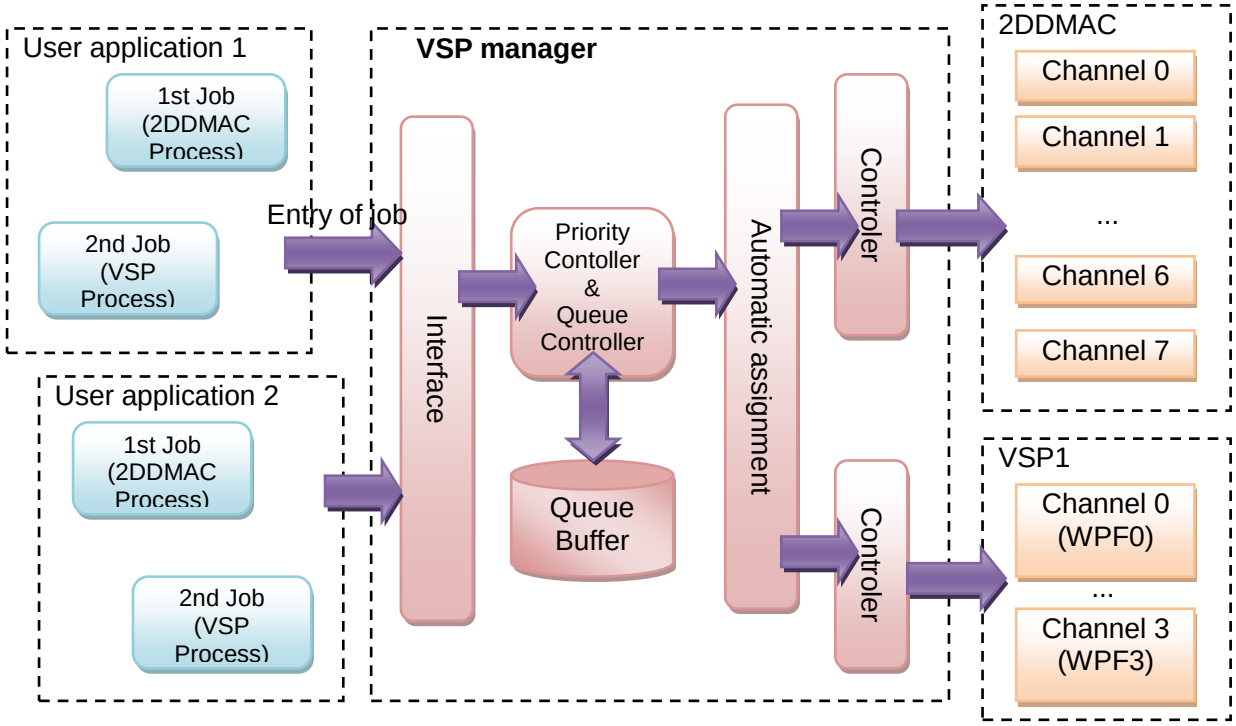


Figure 1-1 Overview of this software

The following is the functional overview of the VSP manager.

- Controls the VSP and 2DDMAC.
- Automatic assignment of free channels. (RPF, UDS and WPF channel of VSP, output channel of 2DDMAC)
- The VSP manager has 32 queue buffers controlled by priority. It's possible to buffer the entry jobs of 32. Therefore maximum 32 applications can use at a time.

The following is the functional overview of the VSP.

- SRU
  The SRU is a module which executes the super resolution processing. It can be specified in 6 levels.
- UDS
  The UDS is a module which up-scales or down-scales the image size. It can be specified in 1/16 to 16 times.
- BRU
  The BRU is a module which executes the image blending processing and Raster Opration (ROP).
- HST
  The HST is a module which converts the RGB color space into the HSV color space.
- HSI
  The HSI is a module which converts the HSV color space into the RGB color space.
- LUT
  This is a 1D-LUT that converts each of three color components by using a lookup table.
- CLU
  This is a three-dimensional LUT (3D-LUT) that converts the input three-color-component data into desired three

color components by using a lookup table.
- HGO
  The HGO generates the one-dimensional histogram for the dynamic gamma correction.
- HGT
  The HGT generates the two-dimensional histogram for the dynamic color correction.
- RPF
  The RPF reads image data from the external memory, unpacks data according to the specified format, convers the color space, converts the number of colors, and executes color keying, ROP operation.
- WPF
  The WPF is an output module that receives image data, converts the color space, number of colors, and format of the data, and outputs the results of VSP image processing to external memory.

Table 1-1 shows supporting modules and channel number.

Table 1-1 Supported module each device

	R-Car H2 (VSPS)	R-Car H2 (VSPR) *1	R-Car M2 (VSPS)	R-Car H2/M2 (VSPD0/VSPD1) *2
RPF (CLUT)	5 (2)	5 (1)	5 (1)	4 (1)
SRU	1	1	1	0
UDS	3	1	1	1
LUT	1	0	1	1
CLU	1	0	1	0
HST	1	1	1	1
HSI	1	1	1	1
BRU	1	1	1	1
HGO	1	0	1	1
HGT	1	0	1	0
WPF	4	4	4	1

*1 Available by enabling the compile switch in makefile. Assignment will decide by the VSP manager.
*2 Reference information.

The following is the functional overview of the 2DDMAC.

- Image extraction
  The 2DDMAC extracts an image in the rectangular area from a point shifting from the source image data origin to a point in the frame memory, and then writes the extracted image data to another frame memory.
- Image rotation / inversion
  Vertical/horizontal inversion and the 90, 180, and 270 degree rotation can be performed.
- Simple enlargement
  When writing a destination image, it can simply be enlarged twice in the X and Y directions.
- Format conversion
  RGB formats can be converted to each other.
  YCbCr formats (YCbCr4:2:0 and YCbCr4:2:2) can be converted to each other.
  No format conversion is possible between RGB and YCbCr.
  The format conversion method is equivalent to that of VSP.

1.2. Configuration of Software

This software consists of the following resources.

- Documents
- Release source files
- Sample source code
- Make file

Table 1-2 and Figure 1-2 show the configurations of the released software.

To use this software, the following additional software which is not included in this software is required.
Details of this additional software are shown below.

- Kernel module source code

This software is distributed based on Dual MIT/GPLv2 licenses. Figure 1-3 shows the lists of these source files.

Table 1-2 Configuration of Document File

No	Name
1	R-Car H2/M2 VSP Manager for Linux User's Manual (this document)

```
vspm
   |-- vspm-module
   |     |-- docs
   |     |     |-- RCH2M2_MMP_VSPM_Linux_UME_(Revision).pdf
   |     |
   |     |-- files
   |           |-- vspm
   |                 |-- if
   |                 |     |-- Makefile
   |                 |     |-- vspm_api.c
   |                 |
   |                 |-- include
   |                       |-- tddmac_drv.h
   |                       |-- vsp_drv.h
   |                       |-- vspm_public.h
   |
   |-- vspm-to-user
         |-- files
               |-- vspm
                     |-- Makefile
                     |-- vspm_tp.c
```

Figure 1-2 Configuration of this software

```
vspm
   |-- vspm-module
         |-- files
               |-- vspm
                     |-- drv
                     |     |-- manager
                     |     |     |-- GPL-COPYING
                     |     |     |-- MIT-COPYING
                     |     |     |-- vspm_common.h
                     |     |     |-- vspm_control.c
                     |     |     |-- vspm_drv_2ddmac.c
                     |     |     |-- vspm_drv_vsp.c
                     |     |     |-- vspm_exec_manager.c
                     |     |     |-- vspm_ip_ctrl.h
                     |     |     |-- vspm_job_manager.c
                     |     |     |-- vspm_lib.c
                     |     |     |-- vspm_sort_queue.c
                     |     |     |-- vspm_task.c
                     |     |     |-- vspm_task_private.h
                     |     |
                     |     |-- tddmac
                     |     |     |-- GPL-COPYING
                     |     |     |-- MIT-COPYING
                     |     |     |-- tddmac_drv.c
                     |     |     |-- tddmac_drv_local.h
                     |     |     |-- tddmac_drv_table.c
                     |     |
                     |     |-- vsp
                     |     |     |-- GPL-COPYING
                     |     |     |-- MIT-COPYING
                     |     |     |-- vsp_drv.c
                     |     |     |-- vsp_drv_local.h
                     |     |     |-- vsp_drv_par.c
                     |     |     |-- vsp_drv_phy.c
                     |     |
                     |     |-- frame.c
                     |     |-- frame.h
                     |     |-- GPL-COPYING
                     |     |-- Makefile
                     |     |-- MIT-COPYING
                     |     |-- tddmac_drv_public.h
                     |     |-- vsp_drv_public.h
                     |     |-- vspm_ioctl.c
                     |     |-- vspm_log.h
                     |     |-- vspm_main.c
                     |     |-- vspm_main.h
                     |     |-- vspm_private.h
                     |     |-- vspm_sub.c
                     |
                     |-- include
                           |-- GPL-COPYING
                           |-- MIT-COPYING
                           |-- tddmac_drv.h
                           |-- vsp_drv.h
                           |-- vspm_public.h
```

Figure 1-3 Configuration of kernel Source Code

1.3. Development Environments

This section describes the development environments for this software.

1.3.1. Hardware Development Environment

Table 1-3 shows the hardware environment for development of systems using this software.

Table 1-3 Hardware Development Environment

Hardware Name		Remarks
Platform	RTP0RC7790SEB00010S (LAGER) RTP0RC7791SEB00011S (KOELSCH)	-
Device	R-Car H2 / M2	-
Using IP	VSP1, 2DDMAC	-

1.3.2. Software Development Environment

Table 1-4 shows the software environment for development of systems using this software.

Table 1-4 Software Development Environment

Software Name	Version / Revision	Remarks
R-Car H2 Linux BSP	-	-
Memory manager	-	Sample application uses.

2. Installation Procedures

2.1. Building the Kernel Modules

The following is the procedure for building the kernel modules that are included in this software.

(1) Setting environment variables
Set the following environment variables. Define $WORK is work directory. CROSS_COMPILE path setting is an example in case the cross compiler extracting directory is $WORK.

```
$ export PATH=$PATH:$WORK/gcc-linaro-arm-linux-gnueabihf-xxx_linux/bin
$ export ARCH=arm
$ CROSS_COMPILE=$WORK/gcc-linaro-arm-linux-gnueabihf-xxx_linux/bin/
arm-linux-gnueabihf-
```

Note: the 'xxx' is version number. Please follow the instructions of the BSP.

The kernel module has special variables.
If you will use the R-CarH2, please set the following environment variables.

```
$ export VSPM_CONFIG = H2CONFIG
```

Other than those above.

```
$ export VSPM_CONFIG = M2CONFIG
```

(2) Bulding
Execute "make" in the build directory.

```
$ cd vspm/vspm-module/files/vspm/drv
$ make
```

(3) Verifying the kernel module
Make sure that the following kernel modules are built under "vspm/vspm-module/files/vspm/drv".

```
vspm.ko
```

2.2. Building the shared library

The following is the procedure for building the release source files that are included in this software.

(1) Setting environment variables
Same as building the release source files. Please refer to section 2.1.

(2) Bulding
Execute "make" in the build directory

```
$ cd vspm/vspm-module/files/vspm/if
$ make
```

(3) Verifyling the binary module
Make sure that the following binary modules are built under "vspm/vspm-module/files/vspm/if".

```
libvspm.so.x.x.x
libvspm.so.x (symbolic link)
libvspm.so   (symbolic link)
```

Note) The symbolic link files referred when you build your application.

2.3. Binary Inclusion Procedure

The following is the procedure for including the kernel and binary modules that are built according to the procedure described in section 2.1 and 2.2.

(1) Storing the kernel modules
Copy ‘vspm.ko’ to BSP user land. Define $NFS is root directory on BSP.
$ sudo cp vspm.ko $NFS/home/root/workspace
(2) Storing the binary module
Copy ‘libvspm.so.x.x.x’ to BSP user land. The ‘x’ number will be changed by release version.
Example: Please execute on PC.
$ sudo cp libvspm.so.x.x.x $NFS/usr/local/lib
$ sudo cp –d libvspm.so.x $NFS/usr/local/lib
$ sudo cp –d libvspm.so $NFS/usr/local/lib
(3) Setting environment variable on lagar board.
Set the LD_LIBRARY_PATH environment variable if ‘/usr/local/lib’ is not included in the path.
$ export LD_LIBRARY_PATH=/usr/local/lib

2.4. Sample program executing procedure

The following is the procedure for building the sample source codes that are included in this software.
This sample source uses memory manager. About memory manager, Please refer to the memory manager users manual.

(1) Modification makefile
Adapt makefile to the circumstances of your environment.
Change of the include path and library path.
(2) Building
Execute “make” in the build directory
$ cd vspm/vspm-tp-user/files/vspm
$ make
(3) Verifyling the executing object
Make sure that the following executing object is built under “vspm/vspm-tp-user/files/vspm”.
vspm_tp
(4) Executing on lagar board.
Copy ‘vspm_tp’ to BSP user land. Executing and enjoying.
$ ./vspm_tp

3. Processing Specifications

3.1. Module Configuration

Figure 3-1 shows the module configuration of this system.

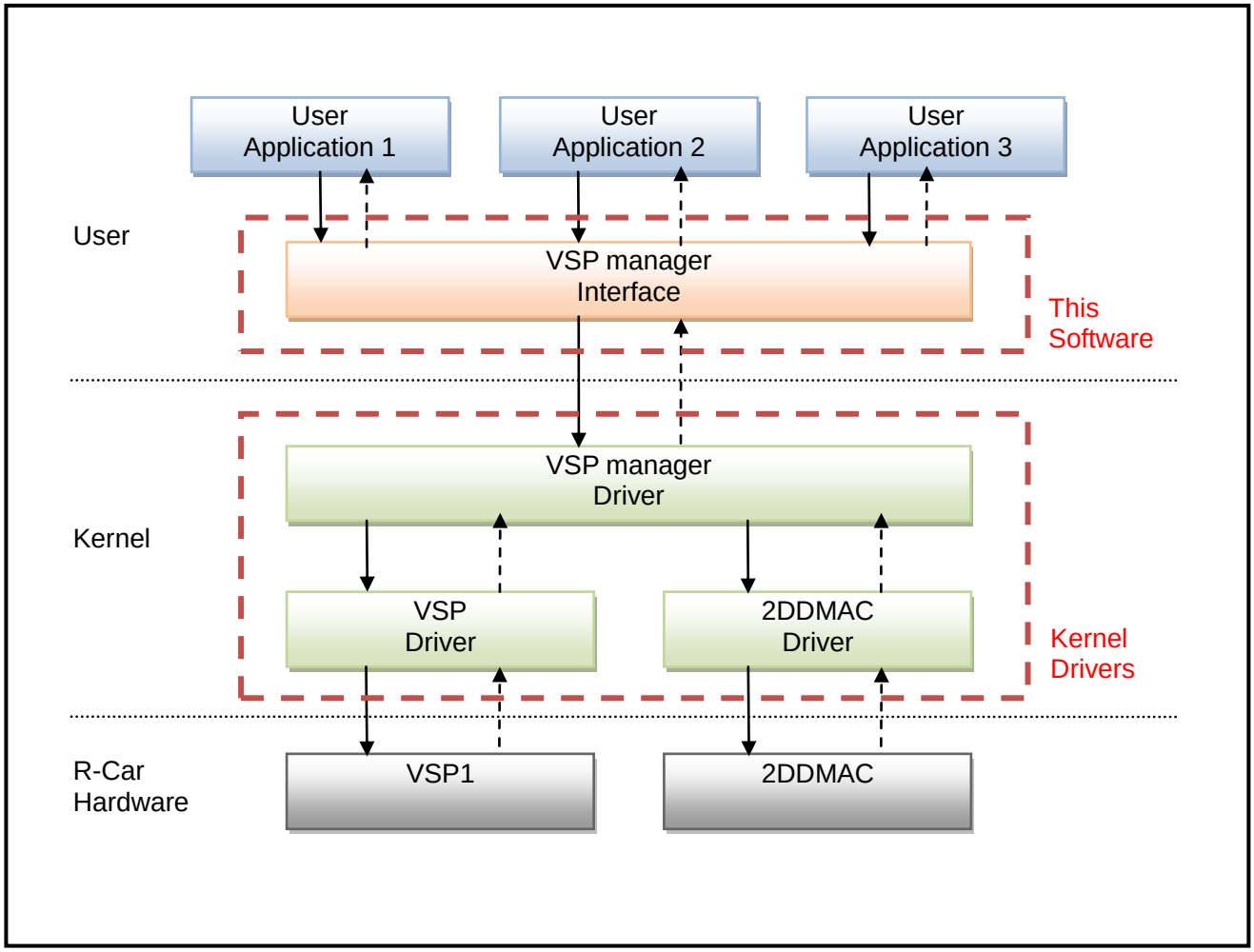


Figure 3-1 Configuration of Module

3.2. Processing Procedure

Figure 3-2 shows the basic processing procedure of VSP manager I/F.
This figure is described that VSP manager I/F is called by two applications. In this figure, the processing procedures between VSP manager I/F and VSP manager driver are drawn briefly. Initialize *1 executes only once. In this figure, after user application 1 executes initial processing, user application 2 does the same initial processing. The initial *1 is carried out at the time application 1 executes the initial processing.
In the same way, finalize *2 executes only once. In this figure, after user application 1 executes finalize processing, user application 2 does the same finalize processing. The finalize *2 is carried out at the time application 2 executes the finalize processing even when initial and finalize processing are not necessary.

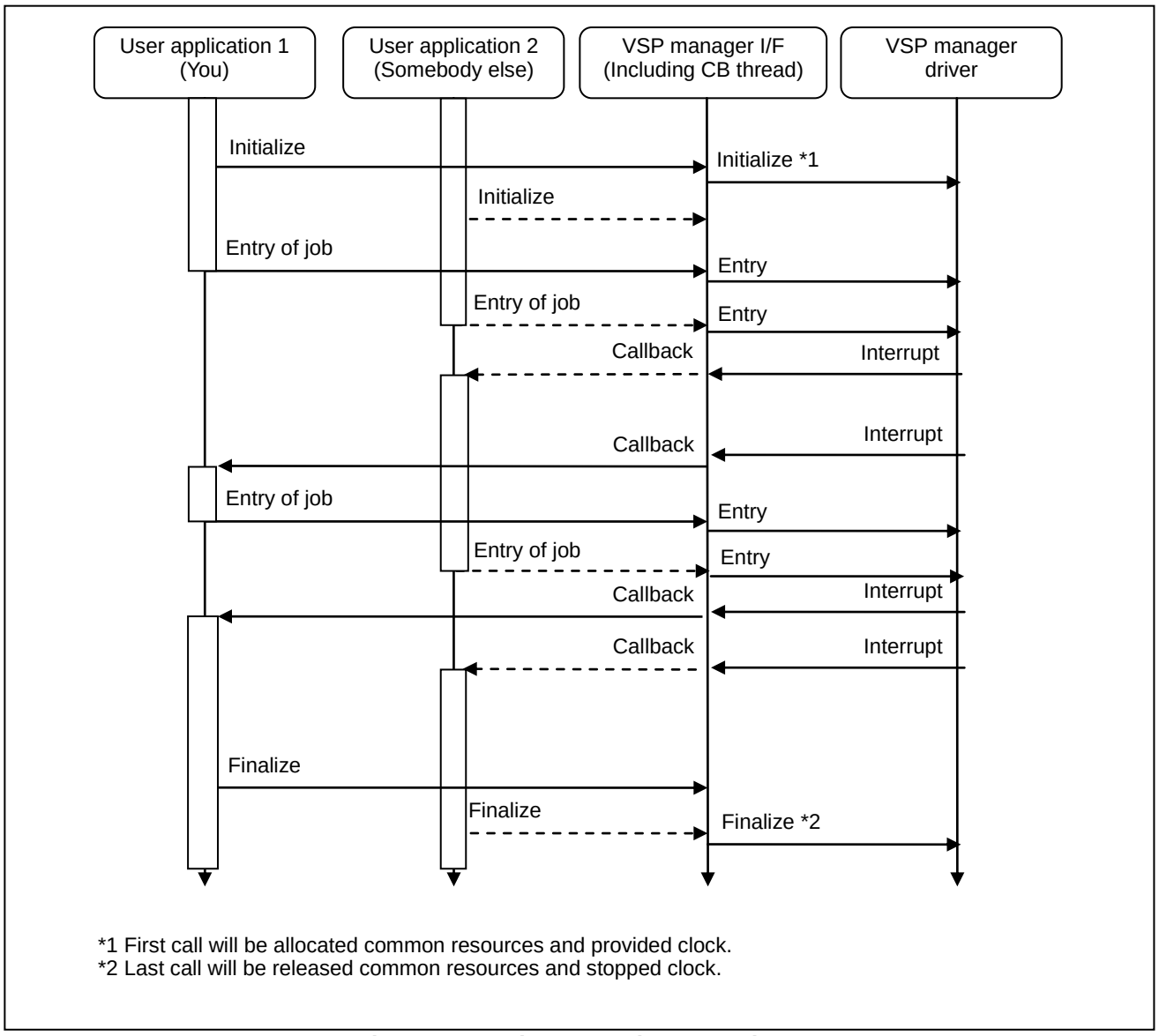


Figure 3-2 Basic Processing Procedure

Figure 3-3 shows VSP manager I/F more detail edly than Figure 3-1.
In this figure, callback thread of user function is described. If you need to avoid from using a polling loop, you have to call sleep-thread at end of Entry-of-job and call wakeup-thread at end of callback thread.

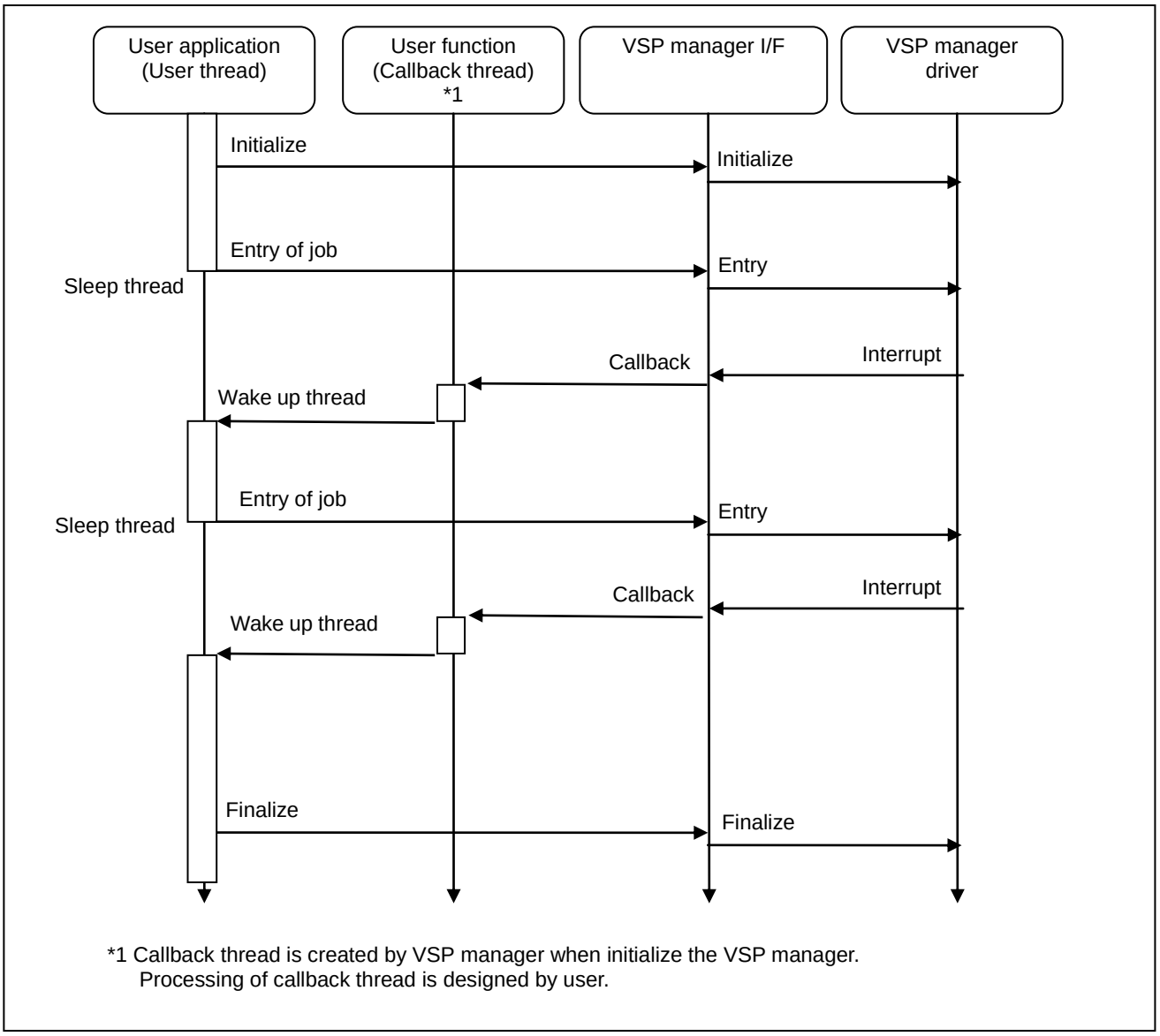


Figure 3-3 Callback Processing Procedure

If Entry-job (a) from application to VSP manager I/F are not related with the result of Entry-job (a) can be execused before Entry-job (b) ends.

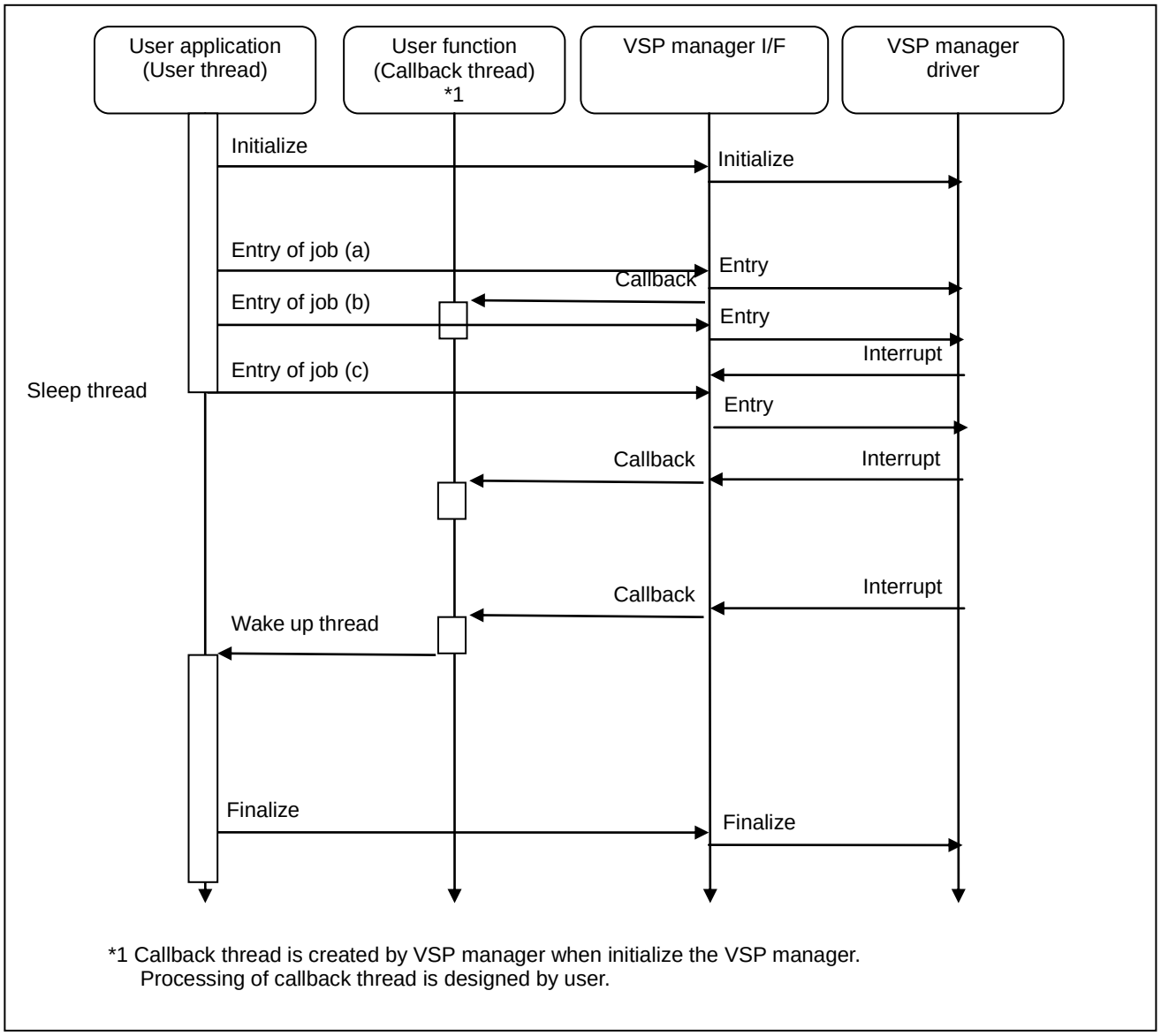


Figure 3-4 Continuous Processing Procedure

3.3. Timing chart

Figure 3-5 shows timing chart until callback from job entry. This figure shows execution from 2 applications. It will understand execution at the same time.
The colored parts of the bars show execution state. The white color shows sleep state. Same color spans two blocks, because assigned function is different. The callback function is executed by callback thread, it is prepared by user, and two colors are mixed.

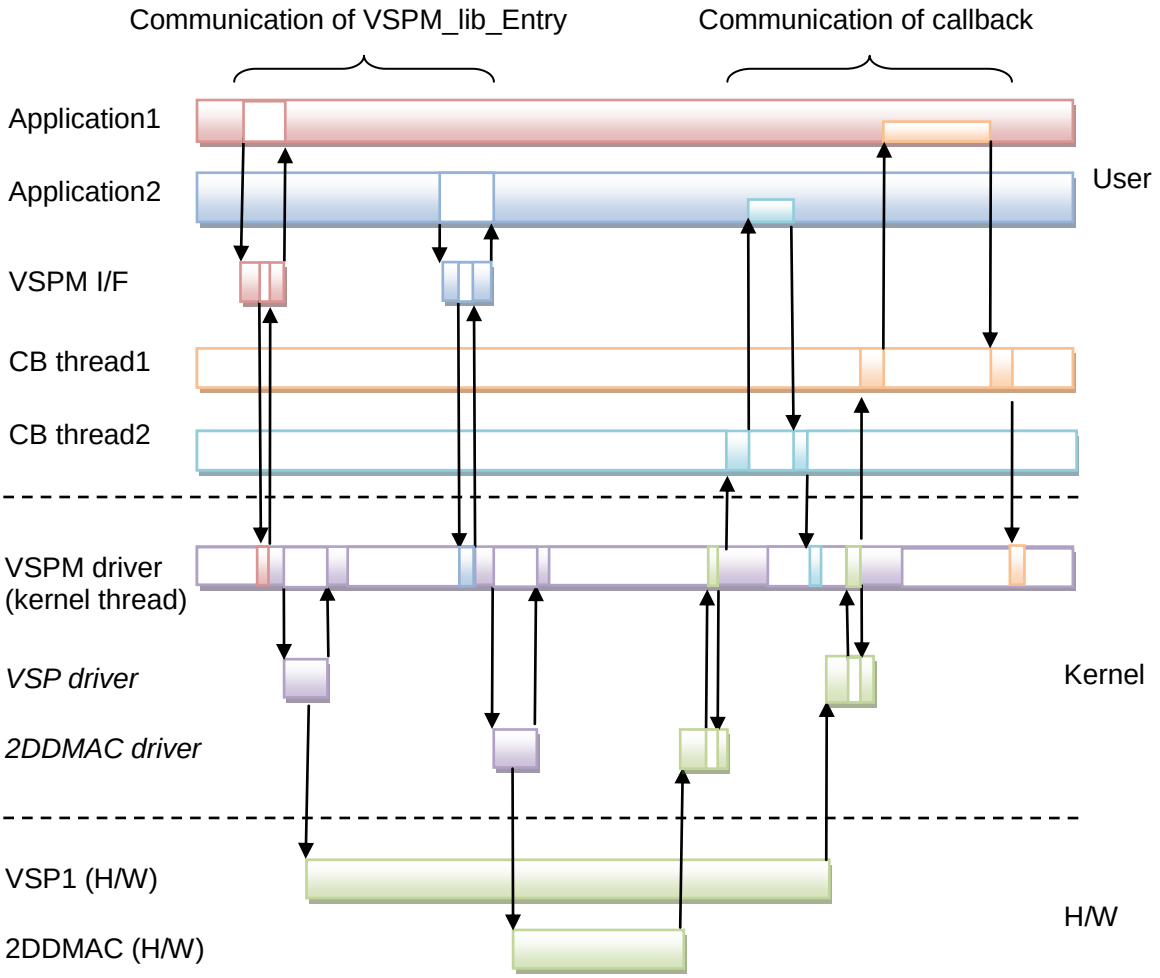


Figure 3-5 Timing chart (Until callback from job entry)

3.4. Control jobs

Registration to the queue of job is carried out by executing the VSPM_lib_Entry (). When a queued job becomes runnable, the VSP manager will start the hardware. Also it delete job from queue. Queue use linked list.

- Sorting jobs

When a job is entered, the VSP manager performs a sort according as priority of jobs in the queue. Follow the steps below, the VSP manager sort jobs.

(1) The VSP manager compares the priority from high priority job (top of list).
(2) If the priority of the entered job is high, to insert the job.
(3) If the same priority, executing priority to jobs who are registered in the destination.

- Priorities of executing

Follow the steps below, the VSP manager execute jobs.

(1) The VSP manager processes job from high priority job (the top of list) in which it is enqueued.
(2) If there is a high priority job waiting for execution, is not executed even if there is a usable a low priority job later.
(3) Remove from the queue after processing complete.

4. List of API

Table 4-1 shows the list of API.

Table 4-1 List of API

No.	Name	Function
1	VSPM_lib_DriverInitialize()	Initializing VSP manager
2	VSPM_lib_DriverQuit()	Finalizing VSP manager
3	VSPM_lib_Entry()	Entry of job.
4	VSPM_lib_Cancel()	Cancel of job
5	PFN_VSPM_COMPLETE_CALLBACK()	Callback functions of finished processing.

4.1. Initializing VSP manager

Name

VSPM_lib_DriverInitialize -- Initializing VSP manager.

Synopsis

```
#include “vspm_public.h”
long VSPM_lib_DriverInitialize (
        unsigned long *pHandle                    (output)
)
```

Arguments

unsigned long *pHandle: Pointer to a handle

Return value

R_VSPM_OK: Successful.
R_VSPM_NG: Failure.

Description

- This API allocates common resource, creates thread and provides clock for IP.
- If successful, this API will return handle value.
- This API is supported multi-calls from user’s applications. First call will be allocated common resources and provide clock.

Notes

- User’s application can not execute from signal handler.
- If user’s application uses the VSP manager’s function, it executes this function at first. When user’s application executes VSPM_lib_DriverQuit (), it can not execute the VSP manager’s functions.
- The handle of parameter used until executing VSPM_lib_DriverQuit () by user calling this function.
- If this API returned other than R_VSPM_OK, please check hardware configuration, memory resource and etc.

See Also

VSPM_lib_DriverQuit ()

4.2. Finalizing VSP manager

Name

VSPM_lib_DriverQuit -- Finalizing VSP manager.

Synopsis

```
#include “vspm_public.h”
long VSPM_lib_DriverQuit (
        unsigned long handle                    (input)
)
```

Arguments

unsigned long *handle*: handle value.

Return value

R_VSPM_OK: Successful.
R_VSPM_NG: Failure.

Description

- This API releases common resource, deletes thread and stops clock for IP. It cancels all jobs (including executing).
- This API is supported multi-calls from user’s applications. Last call will be released common resources and stopped clock.

Notes

- User’s application can not execute from signal handler.
- The VSPM_lib_DriverInitialize () and VSPM_lib_DriverQuit () are supported multi-call. In case of you executing repeat this APIs, this API doesn’t return error (Except in case of failed allocation resource).
- If this API returned other than R_VSPM_OK, please checks handle value. When handle value is true, please check hardware configuration, memory resource and etc.

See Also

VSPM_lib_DriverInitialize ()

4.3. Entry of job

Name

VSPM_lib_Entry -- Entry of job.

Synopsis

```
#include “vspm_public.h”
long VSPM_lib_Entry (
        unsigned long handle,                                   (input)
        unsigned long *puwJobId,                                (output)
        char bJobPriority,                                      (input)
        VSPM_IP_PAR *pIpParam,                                  (input)
        unsigned long uwUserData,                               (input)
        PFN_VSPM_COMPLETE_CALLBACK pfnNotifyComplete            (input)
)
```

Arguments

unsigned long *handle*: handle value.
unsigned long **puwJobId*: Pointer to a job ID.
char *bjobPriority*: Priority of job. 1 (VSPM_PRI_MIN) to 126 (VSPM_PRI_MAX)
VSPM_IP_PAR **pIpParam*: Pointer to a processing parameter.
unsigned long *uwUserData*: Data set by user.
PFN_VSPM_COMPLETE_CALLBACK *pfnNotifyComplete*: Function pointer of callback function.

Struct

```
typedef struct {
        unsigned short uhType;
        union {
                VSPM_VSP_PAR *ptVsp;
                VSPM_2DDMAC_PAR *pt2dDmac;
        } unionIpParam;
} VSPM_IP_PAR;
```

unsigned short *uhType*: Processing type.
VSPM_TYPE_VSP_AUTO: Auto channel assignment.
(Assignment order: ch3 > ch2 > ch1 > ch0)
VSPM_TYPE_2DDMAC_AUTO: Auto channel assignment.
(Assignment order: ch7 > ch3 > ch6 > ch2 > ch5 > ch1 > ch4 > ch0)

```
typedef struct {
        T_TDDMAC_MODE *ptTdDmacMode;
        T_TDDMAC_REQUEST *ptTdDmacRequest;
} VSPM_2DDMAC_PAR;
```

The VSPM_VSP_PAR is redefinition of type from T_VSP_START. Please refer to section 6.1. The T_TDDMAC_MODE and T_TDDMAC_REQUEST are 2DDMAC driver’s parameter. Please refer to section 7.1 and 7.2.

Return value

R_VSPM_OK: Successful.
R_VSPM_NG: Failure.
R_VSPM_PARAERR: Invalid parameter.
R_VSPM_QUE_FULL: Overflow queue.

Description

- This API requests image processing.
- Request unit is 1 channel. Also entry can not process VSP and 2DDMAC at a time.
- Be set to *unionIpParam* the structure of the type specified in *uhType*.
- Process does not end at the time of the completion of the entry. Since the completion callback function that is set to *pfnNotifyComplete* of argument is called, please judge at that time.
- Completion callback is possible to specify the same function. It has a user's data and job ID. Job ID can get this API. It's possible to judge whether the callback of any request using these parametes.
- If there is no correlation in the buffer, you can run the entry without waiting for the completion callback.
- Priority is effective when stacked in the queue. Processing request will be set queue in order of decreasing prioriry. For the same priority is the FIFO.

Notes

- User's application can not execute from signal handler.
- The buffer of specified to the *ptIpParam* of argument should not release until processing finished.
- The *pfnNotifyComplete* of argument should not set null pointer.
- About detail of the VSPM_VSP_PAR and VSPM_2DDMAC_PAR, refer to section 6 and section 7.
- If return value is other than R_VSPM_OK, the VSPM manager is rejecting entry. Therefore you no need to cancel.

See Also

VSPM_lib_Entry ()

4.4. Cancel of job

Name

VSPM_lib_Cancel -- Cancel of job.

Synopsis

```
#include “vspm_public.h”
long VSPM_lib_Cancel (
        unsigned long handle,                    (input)
        unsigned long uwJobId                    (input)
)
```

Arguments

unsigned long *handle*: handle value.
unsigned long *uwJobId*: Job ID.

Return value

R_VSPM_OK: Successful.
R_VSPM_NG: Failure.
R_VSPM_PARAERR: Invalid parameter.
VSPM_STATUS_ACTIVE: Failure (Job is executing)
VSPM_STATUS_NO_ENTRY: Failure (Job is not entry)

Description

- This API cancels job. When job is standby, cancels entry and calls finished call-back function.
- When job is executing, continue executing and this API will return VSPM_STATUS_ACTIVE.
- When already finished job or not found job, this API will return VSPM_STATUS_NO_ENTRY.

Notes

- In case of hardware failure, rather than this API, please re-initialization. Because, this API can not cancel executing job.

See Also

VSPM_lib_Entry ()

4.5. Callback functions of finished processing

Name

(PFN_VSPM_COMPLETE_CALLBACK) – Callback functions of finished processing.

Synopsis

```
#include “vspm_public.h”
void (*PFN_VSPM_COMPLETE_CALLBACK) (
    unsigned long uwJobId,              (output)
    long wResult,                       (output)
    unsigned long uwUserData            (output)
)
```

Arguments

unsigned long *uwJobId*: Job ID.
Long *wResult*: Processing has been done.
R_VSPM_OK: Processing successful.
R_VSPM_NG: Failure.
R_VSPM_CANCEL: Cancel has been done.
R_VSPM_DRIVER_ERR: Fatal error of VSP and 2DDMAC driver.
Other: Minor error of VSP and 2DDMAC driver.
Unsigned long *uwUserData*: Data set by the entry of job.

Return value

None.

Description

- When finish image processing or detect abnormal, the VSP manager execute this API.
- The *uwJobId* and *uwUserData* of argument are set by VSPM_lib_Entry ().
- When the *wResult* is other than R_VSPM_OK, R_VSPM_NG, R_VSPM_CANCEL and R_VSPM_DRIVER_ERR, the wResult is set detail error code of VSP or 2DDMAC. In case of using VSP, refer to section 6.4. In case of using 2DDMAC, refer to section 7.4.

Notes

- User’s application must judge by this API. If *wResult* of argument is other than R_VSPM_OK, image processing is failure.
- Don’t call the VSPM manager’s function within the callback context.
- When the VSPM_lib_Entry () processing is delayed, in some case, before entry processing, completion callback is called.
- If the *wResult* of argument is other than R_VSPM_OK, you can retry entry. Because, the VSP manager initialize register every time. When the VSP manager can not be recovery, must re-initialize system.

See Also

VSPM_lib_Entry ()

5. VSP manager parameters

Table 5-1 Configuration parameter lists

Define Name	Value	Note
VSPM_TYPE_VSP_AUTO	0x0600	Automation assignment channel of VSP
VSPM_TYPE_2DDMAC_AUTO	0x0400	Automation assignment channel of 2DDMAC
VSPM_PRI_MAX	126	Maximum priority
VSPM_PRI_MIN	1	Minimum priority
VSPM_STATUS_ACTIVE	2	
VSPM_STATUS_NO_ENTRY	3	

Table 5-2 Error code of VSP manager

Define Name	Value	Note
R_VSPM_OK	0	Result OK
R_VSPM_NG	-1	Result NG
R_VSPM_PARAERR	-2	Parameter error
R_VSPM_SEQERR	-3	Sequence error
R_VSPM_QUE_FULL	-4	Overflow of queue
R_VSPM_CANCEL	-5	Cancel of job
R_VSPM_DRIVER_ERR	-10	Driver’s error
R_VSPM_HARDWARE_ERR	-11	Hardware’s error
R_VSPM_START_ERR	-12	Staring error

6. VSP driver parameters

6.1. T_VSP_START

The following is described about the member of T_VSP_START structure.

```
Typedef struct{
    unsigned char       rpf_num;
    unsigned long       rpf_order;
    unsigned long       use_module;
    T_VSP_IN            *src1_par ;
    T_VSP_IN            *src2_par ;
    T_VSP_IN            *src3_par ;
    T_VSP_IN            *src4_par ;
    T_VSP_OUT           *dst_par ;
     T_VSP_CTRL         *ctrl_par ;
} T_VSP_START;
```

Member	Direction	Contents
unsigned char *rpf_num*	Input	Input source number (0 to 4) If you set 0 to *rpf_num*, you must set virtual input on BRU. If you set 1 to *rpf_num*, you must set *src1_par*. If you set 2 to *rpf_num*, you must set *src1_par* and *src2_par*. If you set 3 to *rpf_num*, you must set *src1_par*, *src2_par* and *src3_par*. If you set 4 to *rpf_num*, you must set *src1_par*, *src2_par*, *src3_par* and *src4_par*.
Unsigned long *rpf_order*	Input	Not used. The specified value will be ignored.
Unsigned long *use_module*	Input	Processing module setting If you use more than one module, you specify the logical disjunction. VSP_SRU_USE (0x0001) : Super-resolution VSP_UDS_USE (0x0002) : Up down scaler VSP_UDS1_USE (0x0004) : Up down scaler VSP_UDS2_USE (0x0008) : Up down scaler VSP_LUT_USE (0x0010) : Look up table VSP_CLU_USE (0x0020) : Cubic-Look up table VSP_HST_USE (0x0040) : Hue saturation value transform VSP_HSI_USE (0x0080) : Hue saturation value transform inverse VSP_BRU_USE (0x0100) : Blend ROP VSP_HGO_USE (0x0200) : Histgram generator-one VSP_HGT_USE (0x0400) : Histgram generator-two
T_VSP_IN **src1_par*	Input	Pointer to a 1st source configuration image structure. If you set 1 or more to *rpf_num*, can't set NULL pointer.
T_VSP_IN **src2_par*	Input	Pointer to a 2nd source configuration image structure. If you set 2 or more to *rpf_num*, can't set NULL pointer.
T_VSP_IN **src3_par*	Input	Pointer to a 3rd source configuration image structure. If you set 3 or more to *rpf_num*, can't set NULL pointer.

T_VSP_IN **src4_par*	Input	Pointer to a 4th source configuration image structure. If you set 4 to *rpf_num*, can't set NULL pointer.
T_VSP_OUT **dst_par*	Input	Pointer to a destination configuration image structure. Can not set NULL pointer to *dst_par*.
T_VSP_CTRL **ctrl_par*	Input	Pointer to a module configuration structure. Can not set NULL pointer to *ctrl_par*.

Figure 6-1 shows input parameter and connection modules. The *rpf_num* is number of input image source. The *use_module* is for specify to use modules. You must set configuration parameter for using module. About coupling between modules, specify to the *connect* of each module parameter.

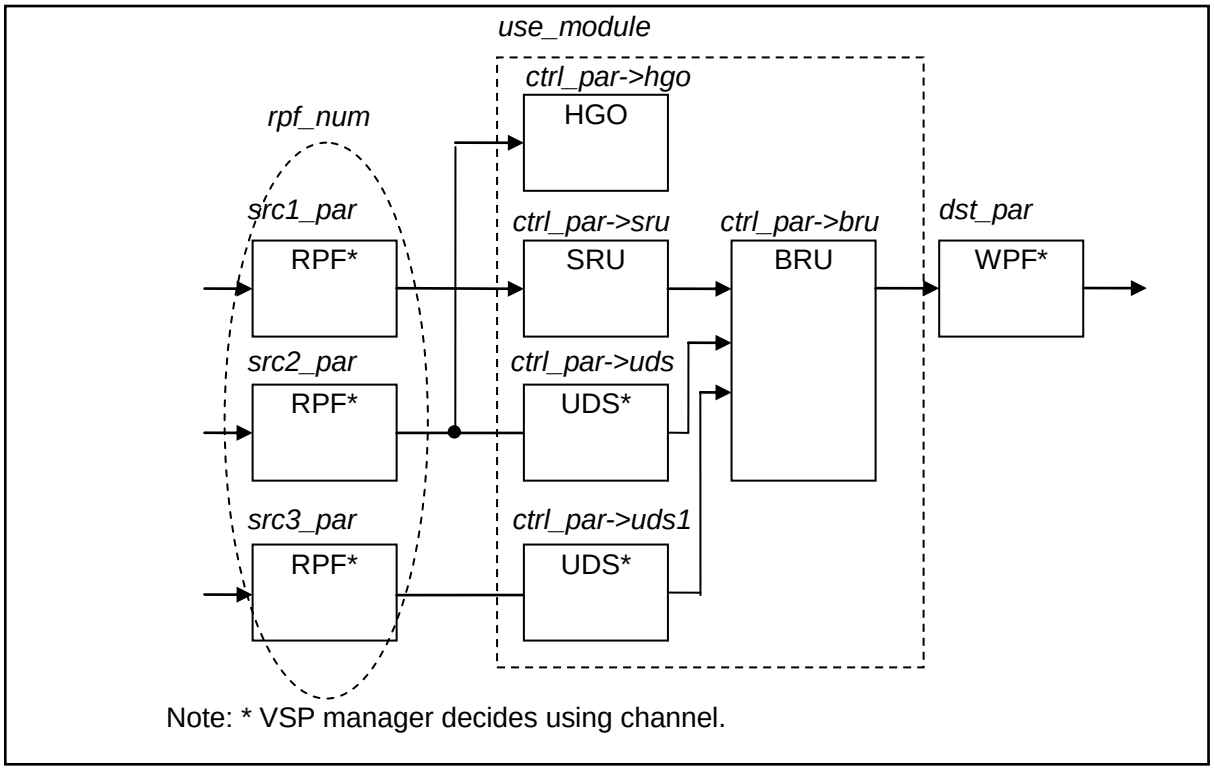


Figure 6-1 Basic module connection association chart

6.1.1. T_VSP_IN

The following is described about the member of T_VSP_IN structure.

```
Typedef struct{
    void                *addr;
    void                *addr_c0;
    void                *addr_c1;
    unsigned short      stride;
    unsigned short      stride_c;
    unsigned short      width;
    unsigned short      height;
    unsigned short      width_ex;
    unsigned short      height_ex;
    unsigned short      x_offset;
    unsigned short      y_offset ;
    unsigned short      format ;
    unsigned char       swap;
    unsigned short      x_position;
    unsigned short      y_position ;
    unsigned char       pwd ;
    unsigned char       cipm;
    unsigned char       cext;
    unsigned char       csc;
    unsigned char       iturbt;
    unsigned char       clrcng;
    unsigned char       vir;
    unsigned long       vircolor ;
    T_VSP_OSDLUT        *osd_lut ;
    T_VSP_ALPHA         *alpha_blend ;
    T_VSP_CLRCNV        *clrcnv ;
    unsigned long       connect ;
} T_VSP_IN;
```

Member	Direction	Contents
void *addr	Input	Pointer to a top buffer address of Y or RGB. Specify continuous physical address.
Void *addr_c0	Input	Pointer to a top buffer address of C When select Semi-Planar of YUV, specify top buffer address of Cb/Cr mixing plane. When select the Planar of YUV, specify top address of Cb plane. Specify continuous physical address.
Void *addr_c1	Input	Pointer to a top buffer address of C When select the Planar of YUV, specify top buffer address of Cr plane. Specify continuous physical address.
Unsigned short stride	Input	Stride of Y/RGB plane buffer. [byte] Specify stride size of Y/RGB plane buffer. When select the Semi Planar or Interleaved of YUV, specify size including Cb/Cr.

Unsigned short *stride_c*	Input	Stride of C plane buffer. [byte] Specify stride size of C plane buffer. When select the Interleaved, C plane isn't used. Therefore this parameter is invalid.
Unsigned short *width*	Input	Image horizontal size. [pixel] Specify horizontal size of input image. Input and output limited size is shown Table 6-5 and Table 6-6. When input format is YUV422 or YUV420. Specify a multiple of 2.
Unsigned short *height*	Input	Image vertical size. [line] Specify vertical size of input image. Input and output limited size is shown Table 6-5 and Table 6-6. When input format is YUV420. Specify a multiple of 2.
Unsigned short *width_ex*	Input	Extended horizontal read size. [pixel] (0 to 8190) Specify the horizontal size of extended read area. Specify *width* of parameter or more. When specify 0, extended read is not used. When input format is YUV422 or YUV420, specify a multiple of 2.
Unsigned short *height_ex*	Input	Extended vertical read size. [line] (0 to 8190) Specify the vertical size of extended read area. Specify *height* of parameter or more. When specify 0, extended read is not used. When input format is YUV420, specify a multiple of 2.
Unsigned short *x_offset*	Input	Horizontal offset. [pixel] Specify horizontal offset. When input format is YUV422 or YUV420, specify a multiple of 2. When use 1bit per pixel alpha plane, specify a multiple of 8.
Unsigned short *y_offset*	Input	Vertical offset. [line] Specify vertical offset. When input format is YUV420, specify a multiple of 2.
Unsigned short *format*	Input	Input format setting. Specify define of "6.3.1 Input format". Note: When use virtual input, specify VSP_IN_ARGB8888 (RGB) or VSP_IN_YUV444_SEMI_PLANAR (YUV).
Unsigned char *swap*	Input	Swap setting. VSP_SWAP_NO (0x00): no swap VSP_SWAP_B (0x01): Byte unit VSP_SWAP_W (0x02): Word unit VSP_SWAP_L (0x04): Long word unit VSP_SWAP_LL (0x08): Long long word unit Example: When specify byte and word swap, set (VSP_SWAP_B|VSP_SWAP_W).
unsigned short *x_position*	Input	Horizontal coordinate of sublayer display location on master layer. A value from 0 to 8189 can be specified. When specify VSP_LAYER_PARENT to *pwd* or don't use BRU, specify 0.
Unsigned short *y_position*	Input	Vertical coordinate of sublayer display location on master layer. A value from 0 to 8189 can be specified. When specify VSP_LAYER_PARENT to *pwd* or don't use BRU, specify 0.

<table>
<tr><td>Unsigned char
pwd</td><td>Input</td><td>Layer setting.
When specify sub layer, put to x_position and y_position are specified position. Also, don’t protrude from the master layer. Specify master layer one out of input image all.

VSP_LAYER_PARENT (0x02): master layer
VSP_LAYER_CHILD (0x01): sub layer</td></tr>
<tr><td>unsigned char
cipm</td><td>Input</td><td>Horizontal chrominance interpolation method setting.
Image data is processed in the YUV444 format inside VSP in case of YUV color space. When the chrominance format of the input image is YUV422 or YUV420, data is upsampled for internal processing. This parameter specifies the method of upsampling for this purpose.

VSP_CIPM_0_HOLD (0x00): The nearest-neighbor method
VSP_CIPM_BI_LINEAR (0x01): The bilinear method.</td></tr>
<tr><td>Unsigned char
cext</td><td>Input</td><td>Lower-bit color data extension method setting.

VSP_CEXT_EXPAN (0x00): extended with 0
VSP_CEXT_COPY (0x01): copied to the lower-order bits
VSP_CEXT_EXPAN_MAX (0x02): extended with 0.
The maximum value is limited to 0xFF.</td></tr>
<tr><td>Unsigned char
csc</td><td>Input</td><td>Color space conversions enable setting.
Enables of disables color space conversion between YUV and RGB to be executed in RPF. The characteristics of color space conversion are determined by iturbt and clrcng.
Note1: When using the BRU, unify input color space on BRU.
Note2: When using the virtual input (vir = VSP_VIR), specify VSP_CSC_OFF.

VSP_CSC_OFF (0x00): Disable
VSP_CSC_ON (0x01): Enable</td></tr>
<tr><td>unsigned char
iturbt</td><td>Input</td><td>CSC conversion expression setting (1).

VSP_ITURBT_601 (0x00): ITU-R BT601 compliant
VSP_ITURBT_709 (0x01): ITU-R BT709 compliant</td></tr>
<tr><td>unsigned char
clrcng</td><td>Input</td><td>CSC conversion expression setting (2).

VSP_ITU_COLOR (0x00): ITU-R rule conversion
VSP_FULL_COLOR (0x01): Full scale conversion

<table>
<tr><th>iturbt</th><th>clrcng</th><th></th></tr>
<tr><td>VSP_ITURBT_601</td><td>VSP_ITU_COLOR</td><td>YUV[16,235/240]
<-> RGB[0,255]</td></tr>
<tr><td>VSP_ITURBT_601</td><td>VSP_FULL_COLOR</td><td>YUV[0,255]
<-> RGB[0,255]</td></tr>
<tr><td>VSP_ITURBT_709</td><td>VSP_ITU_COLOR</td><td>YUV[16,235/240]
<-> RGB[0,255]</td></tr>
<tr><td>VSP_ITURBT_709</td><td>VSP_FULL_COLOR</td><td>YUV[16,235/240]
<-> RGB[16,235]</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>unsigned char
vir</td><td>Input</td><td>Virtual input enable setting.
Enables or Disables the virtual input function. The image to be processed by the RPF is usually read from the external memory. Instead of this input, the virtual input function generates a single-color image within the RPF and sends it to the modules in VSP.
When the virtual input function is enabled, the fixed value specified in the vircolor is used as the input to the RPF.
Note: When the virtual input function is enabled, transparent color and color conversion are invalid. Also, the x_offset and y_offset are invalid.

VSP_NO_VIR (0x00): Disable. (Don’t use)
VSP_VIR (0x01): Enable. (Use)</td></tr>
<tr><td>unsigned long
vircolor</td><td>Input</td><td>Image color setting of virtual input.
Specify RGB or YUV color data of virtual input when specify VSP_VIR to vir of parameter.

RGB format
MSB LSB
alpha(8bit) | R(8bit) | G(8bit) | B(8bit)
31 0
YcbCr format
MSB LSB
alpha(8bit) | Cr(8bit) | Y(8bit) | Cb(8bit)
31 0</td></tr>
<tr><td>T_VSP_OSDLUT
*osd_lut</td><td>Input</td><td>Pointer to a structure of RPF clut setting.
When input format is VSP_IN_RGB_CLUT_DATA or VSP_IN_YUV_CLUT_DATA, this parameter will be valid. Specify color lookup table pointer.</td></tr>
<tr><td>T_VSP_ALPHA
*alpha_blend</td><td>Input</td><td>Pointer to a structure of alpha blend setting
Can not specify null pointer.</td></tr>
<tr><td>T_VSP_CLRCNV
*clrcnv</td><td>Input</td><td>Pointer to a structure of color conversion setting
When specify color conversion setting pointer, color conversion function will be valid. When virtual input or transparent color setting is valid, this parameter is invalid.</td></tr>
<tr><td>Unsigned long
connect</td><td>Input</td><td>Processing connection setting.

Specify the module to be executed next to the RPF. If connect to WPF from RPF, you set 0.

VSP_SRU_USE (0x0001) : Super-resolution
VSP_UDS_USE (0x0002) : Up down scaler
VSP_UDS1_USE (0x0004) : Up down scaler
VSP_UDS2_USE (0x0008) : Up down scaler
VSP_LUT_USE (0x0010) : Look up table
VSP_CLU_USE (0x0020) : Cubic-Look up table
VSP_HST_USE (0x0040) : Hue saturation value transform
VSP_BRU_USE (0x0100) : Blend ROP</td></tr>
</table>

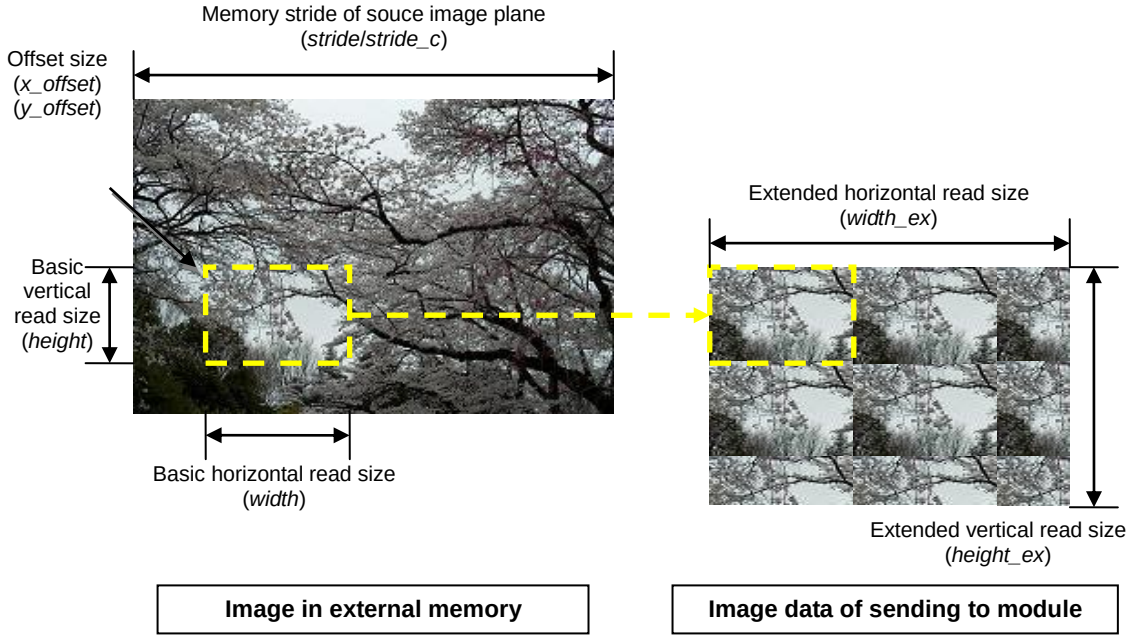


Figure 6-2 Extend reading size association chart

Figure 6-2 is shown input image and extended reading size association chart.
When extended read function is valid, reads repeated until the size specified by the *width_ex* and *height_ex* from an area of the specified size in *width* and *height*, and sends it to the modules in VSP.

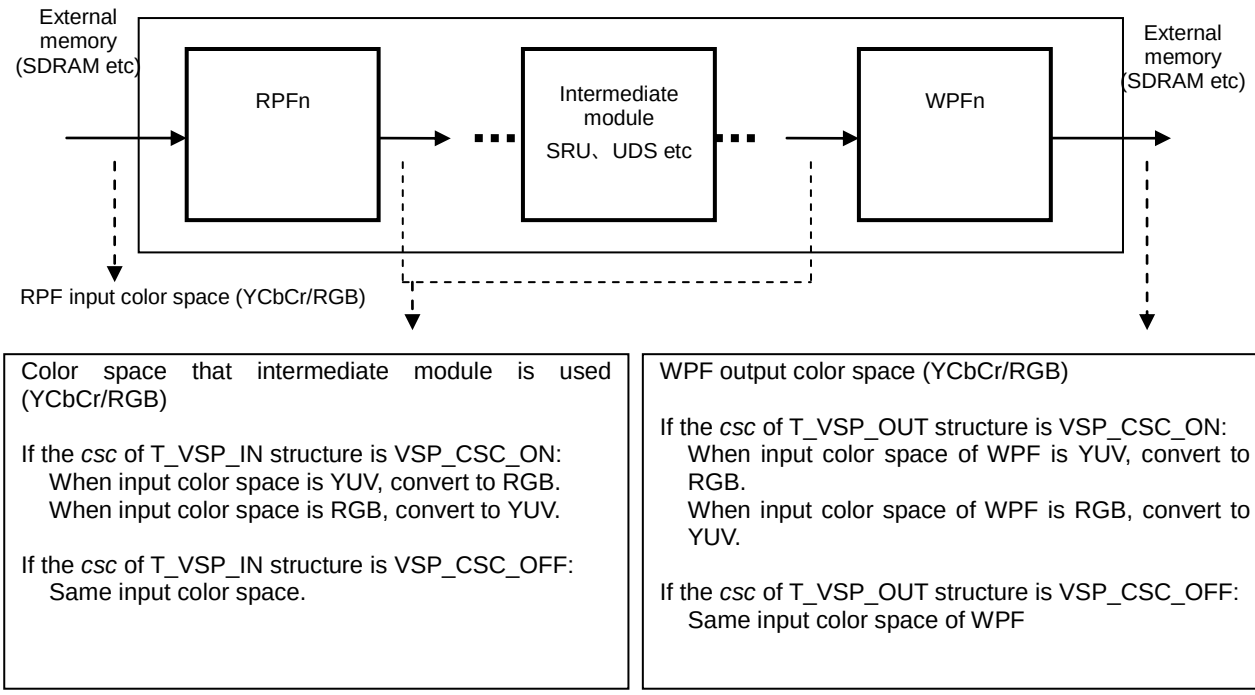


Figure 6-3 Input/Output format and color space

Figure 6-3 is shown input/output format and color space association chart.
Color space that intermediate module uses is decided by specified color space of input format and the *csc* of T_VSP_IN structure. When using BRU, unify input color space on BRU.

6.1.1.1. T_VSP_OSDLUT

The following is described about the member of T_VSP_OSDLUT structure.

```
Typedef struct{
    unsigned long       *clut;
    short               size;
} T_VSP_OSDLUT;
```

<table>
<tr><th>Member</th><th>Direction</th><th>Contents</th></tr>
<tr><td>unsigned long
*clut</td><td>Input</td><td>Pointe to the CLUT/RPF.
The CLUT/RPF color format depends on the format in T_VSP_IN structure.
Each component has 8bit range. About alpha value, 0 is transparency and 255 is opacity.

<table>
<tr><td></td><td>MSB</td><td></td><td></td><td>LSB</td></tr>
<tr><td>RGB format</td><td>alpha(8bit)</td><td>R(8bit)</td><td>G(8bit)</td><td>B(8bit)</td></tr>
<tr><td></td><td>31</td><td></td><td></td><td>0</td></tr>
<tr><td></td><td>MSB</td><td></td><td></td><td>LSB</td></tr>
<tr><td>YUV format</td><td>alpha(8bit)</td><td>Cr(8bit)</td><td>Y(8bit)</td><td>Cb(8bit)</td></tr>
<tr><td></td><td>31</td><td></td><td></td><td>0</td></tr>
</table>
</td></tr>
<tr><td>short
size</td><td>Input</td><td>CLUT/RPF table size.
The setting range is 1 to 256.

Note: When the size specified fewer than 256, the VSP manager offers no guarantee off value you don’t set.</td></tr>
</table>

6.1.1.2. T_VSP_ALPHA

The following is described about the member of T_VSP_ALPHA structure.

```
Typedef struct{
    void            *addr_a;
    unsigned char   alphan;
    unsigned long   alpha1;
    unsigned long   alpha2;
    unsigned short  astride;
    unsigned char   aswap;
    unsigned char   asel;
    unsigned char   aext;
    unsigned char   anum0;
    unsigned char   anum1;
    unsigned char   esou;
    unsigned char   irop;
    unsigned char   msken;
    unsigned char   bsel;
    unsigned long   mgcolor;
    unsigned long   mscolor0;
    unsigned long   mscolor1;
} T_VSP_ALPHA;
```

Member	Direction	Contents
void *addr_a	Input	Pointer to a top buffer address of alpha plane. When using alpha plane, specify. Specify continuous physical address.
Unsigned char *alphan*	Input	Transparent color setting. This parameter specify logical sum. Example, when enables *alpha1* and *alpha2*, specify (VSP_ALPHA_AL1|VSP_ALPHA_AL2). VSP_ALPHA_NO (0x00): Disable transparent color VSP_ALPHA_AL1 (0x01): compare to *alpha1* VSP_ALPHA_AL2 (0x02): compare to *alpha2* Note: When virtual input is valid, this parameter will be VSP_ALPHA_NO.
unsigned long *alpha1*	Input	Alpha value and transparent color setting 1. When the *alphan* is set to VSP_ALPHA_AL1, this member is valid. Specify the color data to compare and the alpha value to replace if they match. According to the setting of *cext*, specify the value of the extension after. RGB format: MSB alpha(8bit) R(8bit) G(8bit) B(8bit) LSB; 31 0 YUV foamat: MSB alpha(8bit) - Y(8bit) - LSB; 31 0
unsigned long *alpha2*	Input	Alpha value and transparent color setting 2. When the *alphan* is set to VSP_ALPHA_AL2, this member is valid. Refer to the *alpha1*.
Unsigned short	Input	Stride of alpha plane. [byte]

astride		
unsigned char *aswap*	Input	Swap setting of alpha plane. VSP_SWAP_NO (0x00): no swap VSP_SWAP_B (0x01): byte unit VSP_SWAP_W (0x02): word unit VSP_SWAP_L (0x04): long word unit VSP_SWAP_LL (0x08): long long word unit Example: When specify byte and word swap, set (VSP_SWAP_B|VSP_SWAP_W).
unsigned char *asel*	Input	Alpha format and processing method. This member selects how to handle the alpha value to be used. When a 1bit alpha value is used. VSP assumes that the 1bpp alpha value for each pixel is stored in the order from MSB to LSB in each byte (big endian). When specify VSP_ALPHA_NUM1 or VSP_ALPHA_NUM3 to *asel*, must specify the pack format that alpha is present in the input image format always. Also when virtual input is valid, specify VSP_ALPHA_NUM. About detail refer to Table 6-2. VSP_ALPHA_NUM1 (0x00): 1/4/8bit packed alpha + plane plane The alpha bit field in 1, 4 or 8bit packed alpha is handled as transparency information. Be sure to specify the packed format that includes alpha. When the *msken* is VSP_MSKEN_ALPHA and the *irop* is not 0, 5, 10 or 15, the alpha plane should be read as mask information. VSP_ALPHA_NUM2 (0x01): 8bit alpha plane The 8bit alpha plane is read from external RAM as transparency information. When the packed RGB format has a bit field for alpha, the information in the alpha bit field is discarded. VSP_ALPHA_NUM3 (0x02): 1bit packed alpha + alpha plane The 1bit packed alpha input is converted by the 8bit transparent alpha generator shown in Figure 6-4 according to the *anum0/1* setting into the 8bit alpha value as transparency information. Select the packed input format that includes a 1bit alpha field. VSP_ALPHA_NUM4 (0x03): 1bit alpha plane + 8bit-transparent generator. The 1bit alpha plane is read from external RAM and converted by the 8bit transparent alpha generator shown in Figure 6-4 according to the *anum0/1* setting into the 8bit alpha value as transparency information. VSP_ALPHA_NUM5 (0x04)：Fixed alpha value

unsigned char *aext*	Input	Lower-bit alpha data extension method setting. When specified VSP_ALPHA_NUM1 to the *asel*, this parameter is valid. VSP_AEXT_EXPAN (0x00): extended with 0 VSP_AEXT_COPY (0x01): copied to the lower-order bits VSP_AEXT_EXPAN_MAX (0x02): extended with 0. The maximum value is limited to 0xFF.
Unsigned char *anum0*	Input	8bit value output when 1bit alpha value is 0. This member specifies the 8bit alpha value to be output when 1bit alpha data is input and the alpha value input the 8bit transparent alpha generator shown in Figure 6-4 is 0. This setting is valid when the *asel* is set to VSP_ALPHA_NUM3 or VSP_ALPHA_NUM4.
Unsigned char *anum1*	Input	8bit value output when 1bit alpha value is 1. This member specifies the 8bit alpha value to be output when 1bit alpha data is input and the alpha value input the 8bit transparent alpha generator shown in Figure 6-4 is 1. This setting is valid when the *asel* is set to VSP_ALPHA_NUM3 or VSP_ALPHA_NUM4.
Unsigned char *afix*	Input	Fixed alpha value. This member specifies the fixed alpha value. This setting is valid when the *asel* is set to VSP_ALPHA_NUM5.
Unsigned char *irop*	Input	IROP operation setting. The source (SRC) for the IROP operation is the pixel data and alpha data specified in the *mgcolor0* or *mgcolor1* IROP input value, which is selected according to the value (0 or 1) generated by the 1bit-mask generator. The destination (DST) is the image data (RGB/YUV) and 8bit alpha data output from the unpack/CLUT processor. IROP operation is applied both for the image data and alpha data between the source and destination data. Speficy define of Table 6-1. About available, refer to Table 6-3.
Unsigned char *msken*	Input	Mask generation specification. Specifies the method of alpha value generation in the 1bit mask alpha generator shown Figure 6-4. VSP_MSKEN_ALPHA (0x00): A 1bit mask value is generated according to the input alpha plane value. When the input alpha is in the 1bit format (*bsel* = VSP_ALPHA_1BIT), the 1bit mask value is output without change. When the input alpha is in the 8bit format (*bsel* = VSP_ALPHA_8BIT), the 1bit mask value is 0 if the alpha value is 0x00; otherwise, the 1bit mask value is 1. VSP_MSKEN_COLOR (0x01): The R/Cr, G/Y, and B/Cb components of the image input to the destination side of the IROP operation unit are compared with the value specified in the *mgcolor* member, respectively. When value match, 1 is output as the 1bit mask value, and in other cases, 0 is output. When the generated 1bit mask data is not used, set *irop* to VSP_IROP_NOP.

<table>
<tr><td>Unsigned char
bsel</td><td>Input</td><td>Alpha bit count conversion selection for 1bit-mask generator.
Specifies the number of bits in the alpha plane to be read as mask information from the external RAM. The alpha value in mask information is used for the source (SRC) in IROP unit. When alpha plane data is 8bit, it is converted to 1bit through the 1bit-mask generator shown in Figure 6-4.

VSP_ALPHA_8BIT(0x00):
8bit alpha is converted to 1bit alpha through the 1bit-mask generator. When the 8bit alpha value input to the RPF is not 0, it is converted to 1; when the value is 0, it is converted to 0.
VSP_ALPHA_1BIT(0x01):
Alpha value goes through the 1bit-mask generator. The 1bit alpha value input to the RPF is output through the 1bit-mask generator without change.

Note: This member setting is valid when the asel is set to VSP_ALPHA_NUM1 or VSP_ALPHA_NUM3 and the msken is set to VSP_MSKEN_ALPHA. In other cases, this member setting has no effect.</td></tr>
<tr><td>Unsigned long
mgcolor</td><td>Input</td><td>Comparison value for 1bit alpha generation
This member specifies the value to be compared for 1bit alpha generation by using the pixel data on the destination side. This setting is ignored when the msken member is set VSP_MSKEN_ALPHA.

<table>
<tr><td></td><td>MSB</td><td></td><td></td><td>LSB</td></tr>
<tr><td>RGB format</td><td>-</td><td>R(8bit)</td><td>G(8bit)</td><td>B(8bit)</td></tr>
<tr><td></td><td>31</td><td></td><td></td><td>0</td></tr>
<tr><td></td><td>MSB</td><td></td><td></td><td>LSB</td></tr>
<tr><td>YUV format</td><td>-</td><td>Cr(8bit)</td><td>Y(8bit)</td><td>Cb(8bit)</td></tr>
<tr><td></td><td>31</td><td></td><td></td><td>0</td></tr>
</table>
</td></tr>
<tr><td>unsigned long
mscolor0</td><td>Input</td><td>IROP source input value when 1bit alpha is 0.
This member specifies the value to be input as the source to the IROP operation unit when the internal 1bit alpha value generated through the 1bit-mask generator is 0. (Figure 6-4)

<table>
<tr><td></td><td>MSB</td><td></td><td></td><td>LSB</td></tr>
<tr><td>RGB format</td><td>alpha(8bit)</td><td>R(8bit)</td><td>G(8bit)</td><td>B(8bit)</td></tr>
<tr><td></td><td>31</td><td></td><td></td><td>0</td></tr>
<tr><td></td><td>MSB</td><td></td><td></td><td>LSB</td></tr>
<tr><td>YUV format</td><td>alpha(8bit)</td><td>Cr(8bit)</td><td>Y(8bit)</td><td>Cb(8bit)</td></tr>
<tr><td></td><td>31</td><td></td><td></td><td>0</td></tr>
</table>
</td></tr>
</table>

unsigned long *mscolor1*	Input	IROP source input value when 1bit alpha is 1. This member specifies the value to be input as the source to the IROP operation unit when the internal 1bit alpha value generated through the 1bit-mask generator is 1. (Figure 6-4)

RGB format

MSB			LSB
alpha(8bit)	R(8bit)	G(8bit)	B(8bit)
31			0

YUV format

MSB			LSB
alpha(8bit)	Cr(8bit)	Y(8bit)	Cb(8bit)
31			0

Figure 6-4 shows configuration diagram of alpha plane.

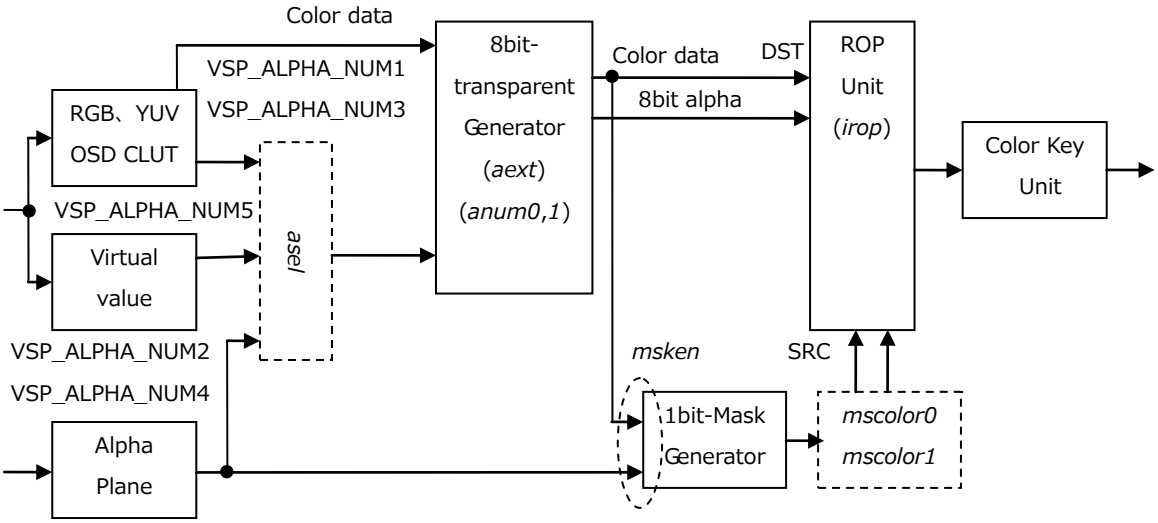


Figure 6-4 Configuration diagram of alpha plane

Decide alpha format and processing method by specify the *asel* of T_VSP_ALPHA member.

In 8bit-transparent Generator, less than 8bit bit field is converted 8bit. If already 8bit, pass through.

1bit-mask generator can select input data by the *msken*. When specify VSP_MSKEN_ALPHA to the *msken*, use alpha plane that select 1bit or 8bit. When specify VSP_ALPHA_NUM2 or VSP_ALPHA_NUM4 to the *asel*, alpha plane data will be used in 8bit-transparent Generator. If you want to use 1bit-mask generator, specify VSP_MSKEN_COLOR to the *msken*. In this case, 1bit-mask generator can use color data of 8bit-transparent Generator.

In ROP operation unit, when the internal 1bit alpha value generated through the 1bit-mask generator is 0, use *mscolor0*. When 1bit alpha value is 1, use *mscolor1*. When don‘t use mask information, specify VSP_IROP_NOP to *irop*. Likewise, When 1bit-Mask generator is invalid or the *asel* is set to VSP_ALPHA_NUM5, set to VSP_IROP_NOP.

Table 6-1 Define of Raster opration

Define	Value	Contents
VSP_IROP_NOP	0x00	NOP(D)
VSP_IROP_AND	0x01	AND(S & D)
VSP_IROP_AND_REVERSE	0x02	AND_REVERSE(S & ~D)
VSP_IROP_COPY	0x03	COPY(S)
VSP_IROP_AND_INVERTED	0x04	AND_INVERTED(~S & D)
VSP_IROP_CLEAR	0x05	CLEAR(0)
VSP_IROP_XOR	0x06	XOR(S ^ D)
VSP_IROP_OR	0x07	OR(S | D)
VSP_IROP_NOR	0x08	NOR(~(S | D))
VSP_IROP_EQUIV	0x09	EQUIV(~(S ^ D))
VSP_IROP_INVERT	0x0A	INVERT(~D)
VSP_IROP_OR_REVERSE	0x0B	OR_REVERSE(S | ~D)
VSP_IROP_COPY_INVERTED	0x0C	COPY_INVERTED(~S)
VSP_IROP_OR_INVERTED	0x0D	OR_INVERTED(~S | D)
VSP_IROP_NAND	0x0E	NAND(~(S & D))
VSP_IROP_SET	0x0F	SET(all 1)

Note: S is source of Blend/ROP unit. D is destination.

Table 6-2 Select alpha value by asel and input format

asel	Input format		
	RGB	YcbCr	RPF(CLUT)
VSP_ALPHA_NUM1	1/4/8bit-alpha	0xFF*	alpha value in CLUT
VSP_ALPHA_NUM2	8bit-alpha plane	8bit-alpha plane	8bit-alpha plane
VSP_ALPHA_NUM3	*anum0* or *anum1* setting	0xFF*	0xFF
VSP_ALPHA_NUM4	*anum0* or *anum1* setting	*anum0* or *anum1* setting	*anum0* or *anum1* setting
VSP_ALPHA_NUM5	*esou* setting	*esou* setting	*esou* setting

Note: Fixed value 0xFF is output because packed alpha is not included in YcbCr.

Table 6-3 Select raster opration enable/disable by asel and msken

asel	*msken*	
	VSP_MSKEN_ALPHA	VSP_MSKEN_COLOR
VSP_ALPHA_NUM1	Valid (alpha plane input)	Valid
VSP_ALPHA_NUM2	Invalid (IROP operation is not available)	Valid
VSP_ALPHA_NUM3	Valid (alpha plane input)	Valid
VSP_ALPHA_NUM4	Invalid (IROP operation is not available)	Valid
VSP_ALPHA_NUM5	Invalid (IROP operation is not available, fixed alpha is output to the subsequent modules behind RPF)	

Note: When invalid (IROP operation is not available), specify VSP_IROP_NOP to *irop*.

6.1.1.3. T_VSP_CLRCNV

The following is described about the member of T_VSP_CLRCNV structure.

```
Typedef struct{
    unsigned long       color1;
    unsigned long       color2;
} T_VSP_CLRCNV;
```

<table>
<tr><th>Member</th><th>Direction</th><th>Contents</th></tr>
<tr><td>unsigned long
color1</td><td>Input</td><td>Color conversion source data setting.
Specify color data (RGB or Y) to compare.
<table>
<tr><td></td><td>MSB</td><td></td><td></td><td>LSB</td></tr>
<tr><td>RGB :</td><td>-</td><td>R(8bit)</td><td>G(8bit)</td><td>B(8bit)</td></tr>
<tr><td></td><td>31</td><td></td><td></td><td>0</td></tr>
<tr><td></td><td>MSB</td><td></td><td></td><td>LSB</td></tr>
<tr><td>YUV :</td><td>-</td><td>-</td><td>Y(8bit)</td><td>-</td></tr>
<tr><td></td><td>31</td><td></td><td></td><td>0</td></tr>
</table>
According to the setting of ‘cext’, specify the value of the extension after.</td></tr>
<tr><td>Unsigned long
color2</td><td>Input</td><td>Color conversion destination data setting.
When compared with color1 and matched, specify color data (RGB or Y) to replace.
<table>
<tr><td></td><td>MSB</td><td></td><td></td><td>LSB</td></tr>
<tr><td>RGB :</td><td>alpha(8bit)</td><td>R(8bit)</td><td>G(8bit)</td><td>B(8bit)</td></tr>
<tr><td></td><td>31</td><td></td><td></td><td>0</td></tr>
<tr><td></td><td>MSB</td><td></td><td></td><td>LSB</td></tr>
<tr><td>YUV :</td><td>alpha(8bit)</td><td>-</td><td>Y(8bit)</td><td>-</td></tr>
<tr><td></td><td>31</td><td></td><td></td><td>0</td></tr>
</table>
According to the setting of cext, specify the value of the extension after.</td></tr>
</table>

Note: These parameters are valid when virtual input or transparent color are invalid.

6.1.2. T_VSP_OUT

The following is described about the member of T_VSP_OUT structure.

```
Typedef struct{
    void            *addr;
    void            *addr_c0;
    void            *addr_c1;
    unsigned short  stride;
    unsigned short  stride_c;
    unsigned short  width;
    unsigned short  height;
    unsigned short  x_offset;
    unsigned short  y_offset ;
    unsigned short  format ;
    unsigned char   swap;
    unsigned char   pxa;
    unsigned char   pad;
    unsigned short  x_coffset;
    unsigned short  y_coffset;
    unsigned char   csc;
    unsigned char   iturbt;
    unsigned char   clrcng;
    unsigned char   cbrm;
    unsigned char   abrm;
    unsigned char   athres;
    unsigned char   clmd;
    unsigned char   ln16;
    unsigned char   dith;
    unsigned char   rotation;
    unsigned char   mirror;
} T_VSP_OUT;
```

Member	Direction	Contents
void *addr	Input	Pointer to a top buffer address of Y or RGB. Specify continuous physical address.
Void *addr_c0	Input	Pointer to a top buffer address of C When select Semi-Planar of YUV, specify top buffer address of Cb/Cr mixing plane. When select the Planar of YUV, specify top address of Cb plane. Specify continuous physical address.
Void *addr_c1	Input	Pointer to a top buffer address of C When select the Planar of YUV, specify top buffer address of Cr plane. Specify continuous physical address.
Unsigned short *stride*	Input	Stride of Y/RGB plane buffer. [byte] Specify stride size of Y/RGB plane buffer. When select the Semi Planar or Interleaved of YUV, specify size including Cb/Cr.
Unsigned short *stride_c*	Input	Stride of C plane buffer. [byte] Specify stride size of C plane buffer. When select the Interleaved, C plane isn't used. Therefore this parameter is invalid.
Unsigned short *width*	Input	Image horizontal size. [pixel] Specify horizontal image size. Input and output limited size is shown Table 6-5 and Table 6-6. When input format is YUV422 or YUV420. Specify a multiple of 2.

Unsigned short *height*	Input	Image vertical size. [line] Specify vertical image size. Input and output limited size is shown Table 6-5 and Table 6-6. When input format is YUV420. Specify a multiple of 2.
Unsigned short *x_offset*	Input	Horizontal offset. [pixel] Specify horizontal offset. When input format is YUV422 or YUV420, specify a multiple of 2.
Unsigned short *y_offset*	Input	Vertical offset. [line] Specify vertical offset. When input forma is YUV420, specify a multiple of 2.
Unsigned short *format*	Input	Output format setting. Specify define of “6.3.2 Output format”.
Unsigned char *swap*	Input	Swap setting. VSP_SWAP_NO (0x00): no swap VSP_SWAP_B (0x01): byte unit VSP_SWAP_W (0x02): word unit VSP_SWAP_L (0x04): long word unit VSP_SWAP_LL (0x08): long long word unit Example: When specify byte and word swap, set (VSP_SWAP_B|VSP_SWAP_W).
unsigned char *pxa*	Input	PAD data select. Select the value to be strored in the bit field indicated as PAD or P in the packed RGB output formats shown in section 6.3.2.1. Both the value specified in the *pad* and the alpha data input from the DPR to WPF are 8bits, but some of the PAD and P bit fields shown section 6.3.2.1 are 4bits or 1bit. When the target bit field is not 8bits, the number of bits in the *pad* value and the alpha data input from the DPR to WPF is reduced according to the *abrm*. VSP_PAD_P (0x00): The value specified in the *pad*. VSP_PAD_IN (0x01): The alpha value output from DPR.
Unsigned char *pad*	Input	PAD value in output packed data. This member specifies the value to be stored in the bit field indicated as PAD or P in the output formats shown in section 6.3.2.1. Specify VSP_PAD_P in the *pxa* member.
Unsigned short *x_coffset*	Input	Horizontal size clipping offset value setting. [pixel] This member specifies the offset size (pixel) from the left end of the image in horizontal size clipping. The left side of the image input to the WPF is cut off for the size specified in this member. A value from 0 to 255 can be specified. (*x_coffset* + *width*) should not exceed the horizontal size of the WPF input.
Unsigned short *y_coffset*	Input	Vertical size clipping offset value setting. [line] This member specifies the offset size (line) from the top end of the image in vertical size clipping. The top side of the image input to the WPF is cut off for the size specified in this member. A value from 0 to 255 can be specified. (*y_coffset* + *height*) should not exceed the vertical size of the WPF input.
Unsigned char *csc*	Input	Color space conversions enable setting. Enables of disables color space conversion between YUV and RGB to be executed in WPF. The characteristics of color space conversion are determined by iturbt and clrcng. VSP_CSC_OFF (0x00): Disable VSP_CSC_ON (0x01): Enable

<table>
<tr><td>unsigned char
iturbt</td><td>Input</td><td>CSC conversion expression setting (1).

VSP_ITURBT_601 (0x00): ITU-R BT601 compliant
VSP_ITURBT_709 (0x01): ITU-R BT709 compliant</td></tr>
<tr><td>unsigned char
clrcng</td><td>Input</td><td>CSC conversion expression setting (2).

VSP_ITU_COLOR (0x00): ITU-R rule conversion
VSP_FULL_COLOR (0x01): Full scale conversion

<table>
<tr><th>iturbt</th><th>clrcng</th><th></th></tr>
<tr><td>VSP_ITURBT_601</td><td>VSP_ITU_COLOR</td><td>YUV[16,235/240]
<-> RGB[0,255]</td></tr>
<tr><td>VSP_ITURBT_601</td><td>VSP_FULL_COLOR</td><td>YUV[0,255]
<-> RGB[0,255]</td></tr>
<tr><td>VSP_ITURBT_709</td><td>VSP_ITU_COLOR</td><td>YUV[16,235/240]
<-> RGB[0,255]</td></tr>
<tr><td>VSP_ITURBT_709</td><td>VSP_FULL_COLOR</td><td>YUV[16,235/240]
<-> RGB[16,235]</td></tr>
</table>
</td></tr>
<tr><td>unsigned char
cbrm</td><td>Input</td><td>Bit count reduction method selection for data storage in packed RGB.
This member specifies the method for reducing when data is stored in the bit fields indicated as R, G and B in section 6.3.2.1 and the target bit fields are not 8 bits.

VSP_CSC_ROUND_DOWN (0x00):
The lower-order bits are truncated.
VSP_CSC_ROUND_OFF (0x01):
Rounding (rounding off).</td></tr>
<tr><td>Unsigned char
abrm</td><td>Input</td><td>Bit count reduction method selection for data storage in PAD.
This member specifies the method for reducing when the data selected through the pxa is stored in the bit fields indicated as PAD or P in section 6.3.2.1 and the target bit field is 4 bits or 1 bit.
VSP_CONVERSION_THRESHOLD can be specified only when the packed RGB format includes a 1bit P field. In this case, when the data selected through the pxa is greater then the athres, 1 is stored in the P field. When the selected data is not greater then the athres, 0 is stored.

VSP_CONVERSION_ROUNDDOWN (0x00):
The lower-order bits are truncated
VSP_CONVERSION_ROUNDING (0x01):
Rounding (rounding off)
VSP_CONVERSION_THRESHOLD (0x02):
Comparison with the threshold value. (this setting is allowed only when the storage field is 1bit)</td></tr>
<tr><td>unsigned char
athres</td><td>Input</td><td>Threshold for conversion to 1bit alpha data.
This member specifies the threshold value used for conversion from 8bit alpha data to 1bit when the abrm is set to VSP_CONVERSION_THRESHOLD.
When the 8bit alpha value before bit count reduction is equal to or smaller than the athres, 0 is stored as the reduced 1bit alpha data. In other cases, 1 is stored as the 1bit alpha data.</td></tr>
</table>

Unsigned char *clmd*	Input	Color data clipping setting. This member specifies the method for clipping the YUV color data output from the WPF. When RGB color data is output from the WPF, specify VSP_CLMD_NO in this member. VSP_CLMD_NO (0x00): Not clipped. (0-255) VSP_CLMD_MODE1 (0x01): YUV mode 1. (16-235(Y),16-240(Cb/Cr)) VSP_CLMD_MODE2 (0x02): YUV mode 2. (1-254)
unsigned char *ln16*	Input	Not used. The specified value will be ignored.
Unsigned char *dith*	Input	Dithering enable setting. When the output format specified RGB with 18 bpp (262144 colors) or less, the color reduction processing is applied to match the number of colors. VSP_NO_DITHER (0x00): Disable VSP_DITHER (0x03): Enable
unsigned char *rotation*	Input	Not used. The specified value will be ignored.
Unsigned char *mirror*	Input	Not used. The specified value will be ignored.

6.1.3. T_VSP_CTRL

The following is described about the member of T_VSP_CTRL structure.

```
Typedef struct{
    T_VSP_SRU       *sru ;
    T_VSP_UDS       *uds ;
    T_VSP_UDS       *uds1 ;
    T_VSP_UDS       *uds2;
    T_VSP_LUT       *lut ;
    T_VSP_CLU       *clu ;
    T_VSP_HST       *hst ;
    T_VSP_HSI       *hsi ;
    T_VSP_BRU       *bru ;
    T_VSP_HGO       *hgo ;
    T_VSP_HGT       *hgt ;
} T_VSP_CTRL ;
```

Member	Direction	Contents
T_VSP_SRU *sru	Input	Pointer to a super-resolution setting structure. If you set VSP_USE_SRU to *connect*, *sru* is referred.
T_VSP_UDS *uds *uds1 *uds2	Input	Pointer to an up-down scaler setting structure. If you set VSP_USE_UDS to *connect*, *uds* is referred. If you set VSP_USE_UDS1 to *connect*, *uds1* is referred. If you set VSP_USE_UDS2 to *connect*, *uds2* is referred.
T_VSP_LUT *lut	Input	Pointer to a look-up table setting structure. If you set VSP_USE_LUT to *connect*, *lut* is referred.
T_VSP_CLU *clu	Input	Pointer to a cubic look-up table setting structure. If you set VSP_USE_CLU to *connect*, *clu* is referred.
T_VSP_HST *hst	Input	Pointer to a hue saturation value transforming setting structure. If you set VSP_USE_HST to *connect*, *hst* is referred.
T_VSP_HSI *hsi	Input	Pointer to a hue saturation value transforming inverse setting structure. If you set VSP_USE_HSI to *connect*, *his* is referred.
T_VSP_BRU *bru	Input	Pointer to a blend/ROP setting structure. If you set VSP_USE_BRU to *connect*, *bru* is referred.
T_VSP_HGO *hgo	Input	Pointer to a histogram generator-one setting structure. If you set VSP_USE_HGO to *use_module*, *hgo* is referred.
T_VSP_HGT *hgt	Input	Pointer to a histogram generator-two setting structure. If you set VSP_USE_HGT to *use_module*, *hgt* is referred.

Note: The *connect* is member of each module's structure. The *use_module* is member of T_VSP_IN's structure.

6.1.3.1. T_VSP_SRU

The following is described about the member of T_VSP_SRU structure.

```
Typedef struct{
    unsigned char       mode;
    unsigned char       param;
    unsigned short      enscl;
    unsigned char       fxa ;
    unsigned long       connect ;
} T_VSP_SRU;
```

Member	Direction	Contents
unsigned char *mode*	Input	Super resolution mode setting VSP_SRU_MODE1 (0x00) : Super resolution without scaling VSP_SRU_MODE2 (0x40) : Super resolution with double scale-up
unsigned char *param*	Input	Apply super-resolution to image This parameter setting depends on the color space of the image input to the SRU. You can set to each color component. Be set logical disjunction. Recommendation setting is RGB format: VSP_SRU_RCR | VSP_SRU_GY | VSP_SRU_BCB YUV format: VSP_SRU_GY VSP_SRU_RCR (0x08) : apply to R/Cr component VSP_SRU_GY (0x04) : apply to G/Y component VSP_SRU_BCB (0x02) : apply to B/Cb component
unsigned short *enscl*	Input	Suprt resolution intensity setting. VSP_SCL_LEVEL1 (0) : Level 1 (Weak) VSP_SCL_LEVEL2 (1) : Level 2 VSP_SCL_LEVEL3 (2) : Level 3 VSP_SCL_LEVEL4 (3) : Level 4 VSP_SCL_LEVEL5 (4) : Level 5 VSP_SCL_LEVEL6 (5) : Level 6 (strong)
unsigned char *fxa*	Input	Fixed alpha output value setting. The SRU does not support input/output of the alpha value. The alpha value input to the SRU is discarded, and the fixed alpha value specified in this param is always output from the SRU.

Unsigned long *connect*	Input	Processing connection setting. Specify the module to be executed next to the SRU. If connect to WPF from SRU, you set 0. VSP_UDS_USE (0x0002) : Up down scaler VSP_UDS1_USE (0x0004) : Up down scaler VSP_UDS2_USE (0x0008) : Up down scaler VSP_LUT_USE (0x0010) : Look up table VSP_CLU_USE (0x0020) : Cubic-Look up table VSP_HST_USE (0x0040) : Hue saturation value transform VSP_BRU_USE (0x0100) : Blend ROP

6.1.3.2. T_VSP_UDS

The following is described about the member of T_VSP_UDS structure.

```
Typedef struct{
    unsigned char       amd;
    unsigned char       fmd;
    unsigned long       filcolor;
    unsigned char       clip;
    unsigned char       alpha;
    unsigned char       complement;
    unsigned char       athres0;
    unsigned char       athres1;
    unsigned char       anum0;
    unsigned char       anum1;
    unsigned char       anum2;
    unsigned short      x_ratio ;
    unsigned short      y_ratio ;
    unsigned short      x_stp;
    unsigned short      x_edp;
    unsigned short      y_stp ;
    unsigned short      y_edp ;
    unsigned short      in_cwidth;
    unsigned short      in_cheight;
    unsigned short      x_coffset;
    unsigned short      y_coffset;
    unsigned short      out_cwidth;
    unsigned short      out_cheight;
    unsigned long       connect ;
} T_VSP_UDS ;
```

Member	Direction	Contents
unsigned char *amd*	Input	Pixel count at scale-up. Specifies the number of pixels generated through scale-up in the UDS. This bit setting is ignored for scale-down. VSP_AMD_NO (0x00) : Pixel count after scale-up is 1 + ((n-1) * scale-up factor) VSP_AMD (0x01) : Pixel count after scale-up is (n * scale-up factor)
unsigned char *fmd*	Input	Padding for insufficient clipping size When the scaling filter outputs an image that is smaller than the clipping size, pixels are interpolated to match the clipping size. This parameter specifies the pixel filling method. VSP_FMD_NO (0x00) : Pixels are filled by copying pixels at the right edge and the bottom edge. VSP_FMD (0x01) : Pixels are filled with the color specified by *filcolor*.

<table>
<tr><td>Unsigned long
filcolor</td><td>Input</td><td>Filling color setting
When the scaling filter outputs an image that is smaller than the clipping size, pixels are interpolated to match the clipping size. This parameter specifies the pixel filling color.

The alpha value is decided by fmd.
When fmd is VSP_FMD_NO, the pixel value at the right edge (bottom edge) of the image is repeated as the alpha value.
When fmd is VSP_FMD, the alpha value is 0.

<table>
<tr><td></td><td>MSB</td><td></td><td></td><td>LSB</td></tr>
<tr><td>RGB format</td><td>-</td><td>R(8bit)</td><td>G(8bit)</td><td>B(8bit)</td></tr>
<tr><td></td><td>31</td><td></td><td></td><td>0</td></tr>
<tr><td></td><td>MSB</td><td></td><td></td><td>LSB</td></tr>
<tr><td>YUV format</td><td>-</td><td>Cr(8bit)</td><td>Y(8bit)</td><td>Cb(8bit)</td></tr>
<tr><td></td><td>31</td><td></td><td></td><td>0</td></tr>
</table>
</td></tr>
<tr><td>unsigned char
clip</td><td>Input
athon</td><td>Alpha output data threshold comparison enable/disable.
Enables or disables comparison with the alpha output data threshold. When this member is VSP_CLIP_ON, the output alpha value is replaced according the the athres0-1 and anum0-2 value.
When you specify VSP_ALPHA_OFF, this member will be invalid.

VSP_CLIP_OFF (0x00): Disable
VSP_CLIP_ON (0x01): Enable</td></tr>
<tr><td>unsigned char
alpha</td><td>Input
aon</td><td>Scale-up/down of alpha plane.
This member specifies whether to enable or disable scale-up/down of the alpha plane when scaling up/down in the RGB format. When the alpha is set VSP_ALPHA_OFF, the UDS outputs the value of the anum0.

VSP_ALPHA_OFF (0x00) : alpha scale-up/-down is not performed
VSP_ALPHA_ON (0x01) : alpha scale-up/-down is performed</td></tr>
<tr><td>unsigned char
complement</td><td>Input</td><td>Interpolation method.

VSP_COMPLEMENT_BIL (0x00) : Bilinear method
VSP_COMPLEMENT_NN (0x01) : Nearest neighbor method *1
VSP_COMPLEMENT_BC (0x02) : multi-tap method *2

*1 This method can be used only when the scale-up/-down factor is 1/1 to 1/4.
*2 When you specify VSP_COMPLEMENT_BC to complement can not specify VSP_ALPHA_ON to alpha.</td></tr>
<tr><td>Unsigned char
athres0</td><td>Input</td><td>Alpha data threshold setting 0.
When the alpha value is equal to or smaller than the value of the athres0, the alpha value is replaced with that of anum0.
When you specify VSP_ALPHA_OFF to alpha, the member will be invalid.</td></tr>
<tr><td>Unsigned char
athres1</td><td>Input</td><td>Alpha data threshold setting 1.
When the alpha value is equal to or greater than value of the athres1, the alpha value is replaced with that of anum2.
When you specify VSP_ALPHA_OFF to alpha, the member will be invalid.</td></tr>
</table>

Unsigned char *anum0*	Input	Replacing alpha value setting after clipping 0. This member set a value that replaces the alpha value when it is equal to or smaller than the value of the *athres0*. When you specify VSP_ALPHA_OFF to *alpha*, this member will be output as alpha value.
Unsigned char *anum1*	Input	Replacing alpha value setting after clipping 1. This member set a value that replaces the alpha value when it is greater than the value of the *athres0* and also smaller than that of the *athres1*. When you specify VSP_ALPHA_OFF to *alpha*, this member will be invalid.
Unsigned char *anum2*	Input	Replacing alpha value setting after clipping 2. This member set a value that replaces the alpha value when it is equal to or greater than the value of the *athres1*. When you specify VSP_ALPHA_OFF to *alpha*, this member will be invalid.
Unsigned short *x_ratio*	Input	Horizontal scaling factor. The horizontal scaling factor has integral part (MANT, 4bit) and fractional part (FRAC, 12bit). Scale factor is the following formula: scale factor = 4096 / ((4096 * MANT) + FRAC) When specify same size, MANT=1 and FRAC=0. X_ratio = 0x1000.
Unsigned short *y_ratio*	Input	Vertical scaling factor. Same as specified in the horizontal.
Unsigned short *x_stp*	Input	Not used. The specified value will be ignored.
Unsigned short *x_edp*	Input	Not used. The specified value will be ignored.
Unsigned short *y_stp*	Input	Not used. The specified value will be ignored.
Unsigned short *y_edp*	Input	Not used. The specified value will be ignored.
Unsigned short *in_cwidth*	Input	Not used. The specified value will be ignored.
Unsigned short *in_cheight*	Input	Not used. The specified value will be ignored.
Unsigned short *x_coffset*	Input	Not used. The specified value will be ignored.
Unsigned short *y_coffset*	Input	Not used. The specified value will be ignored.
Unsigned short *out_cwidth*	Input	Clipping size of horizontal pixel count after scale-up/down. [pixel] The horizontal width of an image output from the scaling filter is adjusted (clipped or padded) to match the pixel count set in the *out_cwidth*. The setting range is 4 to 2048 in a scale-down operation and 4 to 8190 in a scale-up operation.

		This parameter always has to be set when using the UDS, regardless of the scale-up, scale-down or no-scaling setting. When the input color format is YUV422 or YUV420, specify the size in 2-pixel units.
Unsigned short *out_cheight*	Input	Clipping size of vertical pixel count after scale-up/down. [line] The vertical height of an image output from the scaling filter is adjusted (clipped or padded) to match the pixel count set in the *out_cheight*. The setting range is 4 to 2048 in a scale-down operation and 4 to 8190 in a scale-up operation. This parameter always has to be set when using the UDS, regardless of the scale-up, scale-down or no-scaling setting. When the input color format is YUV420, specify the size in 2-pixel units.
Unsigned long *connect*	Input	Processing connection setting. Specify the module to be executed next to the UDS. If connect to WPF from UDS, you set 0. VSP_SRU_USE (0x0001) : Super-resolution VSP_LUT_USE (0x0010) : Look up table VSP_CLU_USE (0x0020) : Cubic-Look up table VSP_HST_USE (0x0040) : Hue saturation value transform VSP_BRU_USE (0x0100) : Blend ROP

6.1.3.3. T_VSP_LUT

The following is described about the member of T_VSP_LUT structure.

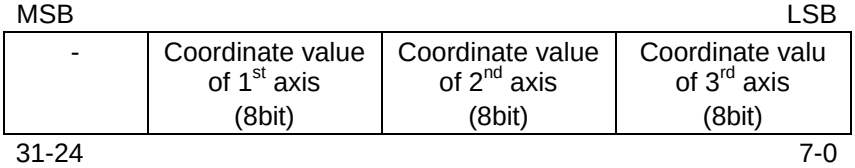
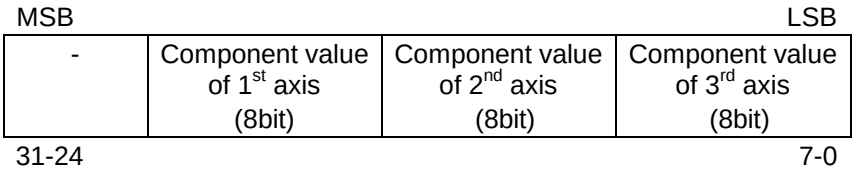
```
Typedef struct{
    unsigned long       *lut;
    short               size;
    unsigned char       fxa ;
    unsigned long       connect ;
} T_VSP_LUT ;
```

Member	Direction	Contents
unsigned long **lut*	Input	Pointer to the look up table. The LUT color format depends on the color space of the image input to the LUT. RGB format MSB LSB Don't Care | R(8bit) | G(8bit) | B(8bit) 31 0 YUV fomat MSB LSB Don't Care | Cr(8bit) | Y(8bit) | Cb(8bit) 31 0 HSV format MSB LSB Don't Care | H(8bit) | S(8bit) | V(8bit) 31 0
short *size*	Input	LUT table size. The setting range is 1 to 256. Note: When the *size* specified fewer than 256, the VSP manager offers no guarantee off value you don't set.
Unsigned char *fxa*	Input	Fixed alpha output value setting. The LUT does not support input/output of the alpha value. The alpha value input to the LUT is discarded, and the fixed alpha value specified in this param is always output from the LUT.
Unsigned long *connect*	Input	Processing connection setting. Specify the module to be executed next to the LUT. If connect to WPF from LUT, you set 0. VSP_SRU_USE (0x0001) : Super-resolution VSP_UDS_USE (0x0002) : Up down scaler VSP_UDS1_USE (0x0004) : Up down scaler VSP_UDS2_USE (0x0008) : Up down scaler VSP_CLU_USE (0x0020) : Cubic-Look up table VSP_HST_USE (0x0040) : Hue saturation value transform VSP_HSI_USE (0x0080) : Hue saturation value transform inverse VSP_BRU_USE (0x0100) : Blend ROP

6.1.3.4. T_VSP_CLU

The following is described about the member of T_VSP_CLU structure.

```
Typedef struct{
     unsigned char      mode;
    unsigned long       *clu_addr;
    unsigned long       *clu_data;
    short               size;
    unsigned char       fxa;
    unsigned long       connect;
} T_VSP_CLU;
```

Member	Direction	Contents
unsigned char *mode*	Input	LUT dimension number Specifies the number of LUT dimensions. 2D mode can be used only when the CLU input color space is YcbCr. VSP_CLU_MODE_3D (0x00) : Operates in 3D mode VSP_CLU_MODE_2D (0x01) : Operates in 2D mode VSP_CLU_MODE_3D_AUTO (0x80) : Operates in 3D mode with automatic table address increment. VSP_CLU_MODE_2D_AUTO (0x81) : Opeartes in 2D mode with automatic table address increment.
unsigned long **clu_addr*	Input	Pointer to a coordinate value. The setting range of each coordinate is 0 to 16. If you specify automatic table address increment, this argument is not refered. Also starting address is 0. Please refer to table 6-4 (2).  When operates in 2D mode, coordinate value of 3rd axis must be set to 0.
Unsigned long **clu_data*	Input	Pointer to a component value. The setting range of each component is 0 to 255.  When operates in 3D mode, 1st axis is R/Cr/H component, 2nd axis is G/Y/S component and 3rd axis is B/Cb/V component. When operates in 2D mode, 2nd axis is Y component. Components of 1st and 3rd axis must be set to 0. Because pass through output.
Short *size*	Input	CLU table size. The setting range is 1 to 4913 in 3D mode and 1 to 289 in 2D mode. Note: The VSP manager offers no guarantee off value you don't set.

Unsigned char *fxa*	Input	Fixed alpha output value setting. The CLU does not support input/output of the alpha value. The alpha value input to the CLU is discarded, and the fixed alpha value specified in this param is always output from the CLU.
Unsigned long *connect*	Input	Processing connection setting. Specify the module to be executed next to the CLU. If connect to WPF from CLU, you set 0. VSP_SRU_USE (0x0001) : Super-resolution VSP_UDS_USE (0x0002) : Up down scaler VSP_UDS1_USE (0x0004) : Up down scaler VSP_UDS2_USE (0x0008) : Up down scaler VSP_LUT_USE (0x0010) : Look up table VSP_HST_USE (0x0040) : Hue saturation value transform VSP_HSI_USE (0x0080) : Hue saturation value transform inverse VSP_BRU_USE (0x0100) : Blend ROP

Table 6-4 shows the relationship between a coordinate and a component. A coordinate and a component are same buffer array.

Table 6-4 storage method of coordinate and component value

(1) VSP_CLU_MODE_3D/VSP_CLU_MODE_2D

Offset	Coordinate[size]				Component[size]			
0	-	1st axis	2nd axis	3rd axis	-	1st axis	2nd axis	3rd axis
1	-	1st axis	2nd axis	3rd axis	-	1st axis	2nd axis	3rd axis
...	-	...	...	...	-	...	...	...
size-2	-	1st axis	2nd axis	3rd axis	-	1st axis	2nd axis	3rd axis
size-1	-	1st axis	2nd axis	3rd axis	-	1st axis	2nd axis	3rd axis

(2) VSP_CLU_MODE_3D_AUTO/VSP_CLU_MODE_2D_AUTO

Offset	Coordinate[size]				Component[size]			
0	-	0	0	0	-	1st axis	2nd axis	3rd axis
1	-	1	0	0	-	1st axis	2nd axis	3rd axis
...	-	...	...	...	-	...	...	...
15	-	15	0	0	-	1st axis	2nd axis	3rd axis
16	-	16	0	0	-	1st axis	2nd axis	3rd axis
17	-	0	1	0	-	1st axis	2nd axis	3rd axis
18	-	1	1	0	-	1st axis	2nd axis	3rd axis
...	-	...	...	...	-	...	...	...
287	-	15	16	0	-	1st axis	2nd axis	3rd axis
288	-	16	16	0	-	1st axis	2nd axis	3rd axis
289	-	0	0	1	-	1st axis	2nd axis	3rd axis
290	-	1	0	1	-	1st axis	2nd axis	3rd axis
...	-	...	...	...	-	...	...	...
4911	-	15	16	16	-	1st axis	2nd axis	3rd axis
4912	-	16	16	16	-	1st axis	2nd axis	3rd axis

Note: 2D mode range is 0 to 288. 3D mode rage is 0 to 4912.

6.1.3.5. T_VSP_HST

The following is described about the member of T_VSP_HST structure.

```
Typedef struct{
    unsigned char       fxa;
    unsigned long       connect;
} T_VSP_HST;
```

Member	Direction	Contents
unsigned char *fxa*	Input	Fixed alpha output value setting. The HST does not support input/output of the alpha value. The alpha value input to the HST is discarded, and the fixed alpha value specified in this param is always output from the HST.
Unsigned long *connect*	Input	Processing connection setting. Specify the module to be executed next to the HST. If connect to WPF from HST, you set 0. VSP_LUT_USE (0x0010) : Look up table VSP_CLU_USE (0x0020) : Cubic-Look up table VSP_HSI_USE (0x0080) : Hue saturation value transform inverse

6.1.3.6. T_VSP_HSI

The following is described about the member of T_VSP_HSI structure.

```
Typedef struct{
    unsigned char       fxa;
    unsigned long       connect;
} T_VSP_HSI;
```

Member	Direction	Contents
unsigned char *fxa*	Input	Fixed alpha output value setting. The HIS does not support input/output of the alpha value. The alpha value input to the HIS is discarded, and the fixed alpha value specified in this param is always output from the HIS.
Unsigned long *connect*	Input	Processing connection setting. Specify the module to be executed next to the HIS. If connect to WPF from HIS, you set 0. VSP_SRU_USE (0x0001) : Super-resolution VSP_UDS_USE (0x0002) : Up down scaler VSP_UDS1_USE (0x0004) : Up down scaler VSP_UDS2_USE (0x0008) : Up down scaler VSP_LUT_USE (0x0010) : Look up table VSP_CLU_USE (0x0020) : Cubic-Look up table VSP_HST_USE (0x0040) : Hue saturation value transform VSP_BRU_USE (0x0100) : Blend ROP

6.1.3.7. T_VSP_BRU

The following is described about the member of T_VSP_BRU structure.

```
Typedef struct{
    unsigned long           lay_order;
    unsigned char           adiv;
    unsigned char           qnt[4];
    unsigned char           dith[4];
    T_VSP_BLEND_VIRTUAL     *blend_virtual;
    T_VSP_BLEND_CONTROL     *blend_control_a;
    T_VSP_BLEND_CONTROL     *blend_control_b;
    T_VSP_BLEND_CONTROL     *blend_control_c;
    T_VSP_BLEND_CONTROL     *blend_control_d;
    T_VSP_BLEND_ROP         *blend_rop;
    unsigned long           connect;
} T_VSP_BRU;
```

<table>
<tr><th>Member</th><th>Direction</th><th>Contents</th></tr>
<tr><td>unsigned long
lay_order</td><td>Input</td><td>Layer order setting of input image.
Specify layer number you want put. You can specify 5 layers including virtual input. You must specify valid layer to lowest back (DST_A).

VSP_LAY_NO (0x00): no input
VSP_LAY_1 (0x01): input image 1 (correspond to the src1_par)
VSP_LAY_2 (0x02): input image 2 (correspond to the src2_par)
VSP_LAY_3 (0x03): input image 3 (correspond to the src3_par)
VSP_LAY_4 (0x04): input image 4 (correspond to the src4_par)
VSP_LAY_VIRTUAL (0x05): virtual input

MSB LSB
<table>
<tr><td>-</td><td>4th from lowest back</td><td>3rd from lowest back</td><td>2nd from lowest back</td><td>1st from lowest back</td><td>Lowest back</td></tr>
<tr><td></td><td>SRC_D</td><td>SRC_R/ SRC_C</td><td>DST_R</td><td>SRC_A</td><td>DST_A</td></tr>
<tr><td>31-20</td><td>19-16</td><td>15-12</td><td>11-8</td><td>7-4</td><td>3-0</td></tr>
</table>
</td></tr>
<tr><td>unsigned char
adiv</td><td>Input</td><td>Color data normalization
Enables or disables division by the alpha value of the color data in BRU blending operation.
This is used when converting the RGB color data format to which the alpha value is multiplied (premultiplied color) into the RGB color data format to which the alpha value is not multiplied (non premultiplied color). DO not use this for the YUV format.

VSP_DIVISION_OFF (0x00):
Divider does not divide the color value by alpha.
VSP_DIVISION_ON (0x01):
Divider divides the color value by alpha.</td></tr>
</table>

Unsigned char *qnt*[4]	Input	Dithering (color reduction) enable setting. Enables or disables dithering (color reduction). The *qnt[0]* corresponds to the input image 1. The *qnt[1]* corresponds to the input image 2. The *qnt[2]* corresponds to the input image 3. The *qnt[3]* corresponds to the input image4. VSP_QNT_OFF (0x00): Disable VSP_QNT_ON (0x01): Enable
unsigned char *dith*[4]	Input	Number of color for pixels after dithering setting. Specify the number of colors for pixels after dithering (color reduction). When you specify VSP_QNT_ON to *qnt*, this parameter will be valid. The *dith[0]* corresponds to the input image 1. The *dith[1]* corresponds to the input image 2. The *dith[2]* corresponds to the input image 3. The *dith[3]* corresponds to the input image4. VSP_DITH_OFF (0x00): Disable VSP_DITH_18BPP (0x01): 18bpp (RGB666:260000 colors) VSP_DITH_16BPP (0x02): 16bpp (RGB565:65535 colors) VSP_DITH_15BPP (0x03): 15bpp (RGB555:32768 colors) VSP_DITH_12BPP (0x04): 12bpp (RGB666:4096 colors) VSP_DITH_8BPP (0x05): 8bpp (RGB666:256 colors)
T_VSP_BLEND_VIRTUAL **blend_virtual*	Input	Pounter to a structure virtual input setting. When you specify the VSP_LAY_VIRTUAL to *lay_order*, this member will be refered.
T_VSP_BLEND_CONTROL **blend_control_a*	Input	Pointer to a structure of Blend/ROP Unit A. When you specify null pointer, the blend/ROP unit through to the DST_A. Note: can not specify VSP_LAYER_NO to DST_A.
T_VSP_BLEND_CONTROL **blend_control_b*	Input	Pointer to a structure of Blend/ROP Unit B. When you specify VSP_LAY_NO to DST_R or null pointer to this member, the Blend/ROP unit through to the DST_B.
T_VSP_BLEND_CONTROL **blend_control_c*	Input	Pointer to a structure of Blend/ROP Unit C. When you specify VSP_LAY_NO to SRC_C (SRC_R) or null pointer to this member, the Blend/ROP unit through to the DST_C.
T_VSP_BLEND_CONTROL **blend_control_d*	Input	Pointer to a structure of Blend/ROP Unit D. When you specify VSP_LAY_NO to SRC_D or null pointer to this member, the Blend/ROP unit through to the DST_D.
T_VSP_BLEND_ROP **blend_rop*	Input	Pointer to a structure of ROP Unit. When you specify VSP_LAY_NO to SRC_C (SRC_R) or null pointer to this member, the Blend/ROP unit through to the DST_D. Also when you specify VSP_LAY_NO to DST_R, ROP unit will be invalid. In that case, The Blend/ROP Unit B through to the DST_B.
unsigned long *connect*	Input	Processing connection setting. Specify the module to be executed next to the BRU. If connect to WPF from BRU, you set 0. VSP_SRU_USE (0x0001) : Super-resolution VSP_UDS_USE (0x0002) : Up down scaler VSP_UDS1_USE (0x0004) : Up down scaler VSP_UDS2_USE (0x0008) : Up down scaler VSP_LUT_USE (0x0010) : Look up table VSP_CLU_USE (0x0020) : Cubic-Look up table VSP_HST_USE (0x0040) : Hue saturation value transform

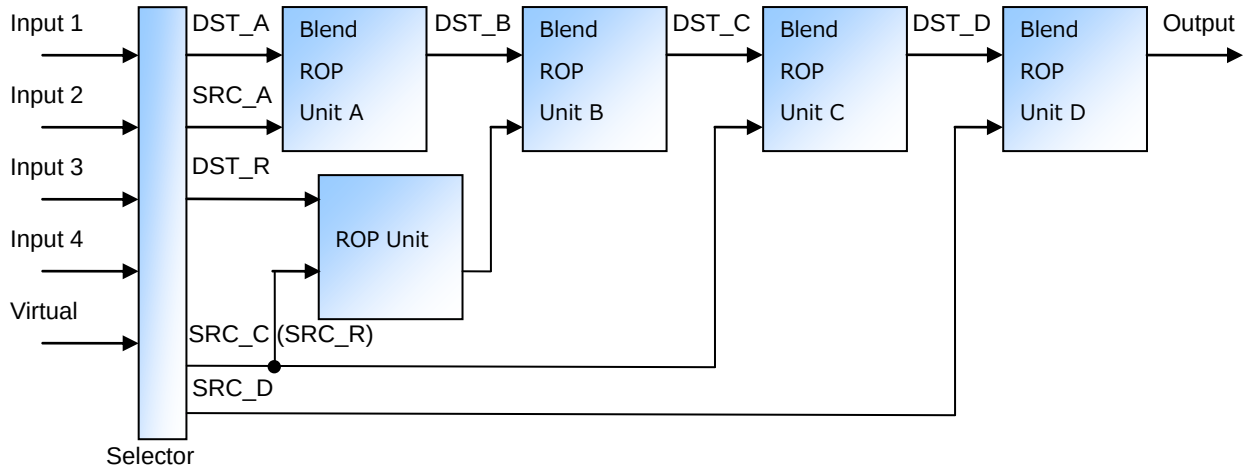


Figure 6-5 Configuration BLEND/ROP unit

Figure 6-5 shows configuration Blend/ROP unit. The Blend/ROP unit is composed of 4 multifunction units and a ROP unit. Source (SRC) and destination (DST) of The Blend/ROP unit is specified the *lay_order* of T_VSP_BRU. You can specify 5 parameters of DST_A, SRC_A, DST_R, SRC_C (SRC_R) and SRC_D. The DST of DST_A, SRC_A, DST_R and SRC_C (SRC_R) are output of each Blend/ROP unit A, B and C. Also the SRC of Blend/ROP unit B is output of ROP unit.

If any of the following conditions is satisfied, the Blend/ROP unit through the DST.
- When specify null pointer to *blend_control_a*, *blend_control_b*, *blend_control_c*, *blend_control_d* and *blend_rop*.
- When specify invalid input to SRC. (VSP_LAY_NO etc)
- About the Blend/ROP Unit B, When the ROP Unit has no output.

Layer that you specify for the *lay_order*, you must match the input image information that you specify for the *src_par* of T_VSP_START.

Example1:
when *rpf_num* = 1 (*src1_par* is valid), can specify VSP_LAY_1/VSP_LAY_VIRTUAL.
Example2:
when *rpf_num* = 2 (*src1_par* and *src2_par* are valid), when specify VSP_LAY_2 only, this is NG. Must be set VSP_LAY_1.

(a) T_VSP_BLEND_VIRTUAL

The following is described about the member of T_VSP_BLEND_VIRTUAL structure.

```
Typedef struct{
    unsigned short      width;
    unsigned short      height;
    unsigned short      x_position;
    unsigned short      y_position ;
    unsigned char       pwd ;
    unsigned long       color;
} T_VSP_BLEND_VIRTUAL;
```

Member	Direction	Contents
unsigned short *width*	Input	Horizontal size of virtual input. [pixel] (1 to 8190)
unsigned short *height*	Input	Vertical size of virtual input. [line] (1 to 8190)
unsigned short *x_position*	Input	Horizontal coordinate of sublayer display location on master layer. A value from 0 to 8189 can be specified. When specify VSP_LAYER_PARENT to *pwd*, specify 0.
Unsigned short *y_position*	Input	Vertical coordinate of sublayer display location on master layer. A value from 0 to 8189 can be specified. When specify VSP_LAYER_PARENT to *pwd*, specify 0.
Unsigned char *pwd*	Input	Layer setting. When specify sub layer, put to *x_position* and *y_position* are specified position. Also, don't protrude from the master layer. Specify master layer one out of input image all. VSP_LAYER_PARENT (0x02): master layer VSP_LAYER_CHILD (0x01): sub layer
unsigned long *color*	Input	Image color setting of virtual input. Specify RGB or YUV color data of virtual input when specify VSP_VIR to *vir* of parameter. RGB: MSB α(8bit) R(8bit) G(8bit) B(8bit) LSB; 31 0 YUV: MSB α(8bit) Cr(8bit) Y(8bit) Cb(8bit) LSB; 31 0

(b) T_VSP_BLEND_CONTROL

The following is described about the member of T_VSP_BLEND_CONTROL structure.

```
Typedef struct{
    unsigned char       rbc ;
    unsigned char       crop ;
    unsigned char       arop ;
    unsigned char       blend_formula ;
    unsigned char       blend_coefx;
    unsigned char       blend_coefy;
    unsigned char       aformula;
    unsigned char       acoefx;
    unsigned char       acoefy;
    unsigned char       acoefx_fix;
    unsigned char       acoefy_fix ;
} T_VSP_BLEND_CONTROL;
```

Member	Direction	Contents
unsigned char *rbc*	Input	Operation type of blending / ROP unit. VSP_RBC_ROP (0x00) : Raster operation VSP_RBC_BLEND (0x01) : Blending operation
unsigned char *crop*	Input	Raster operation setting of color data. Can specify the defined “Table 6-1 Define of Raster opration”.
Unsigned char *arop*	Input	Raster operation setting of alpha value. Can specify the defined “Table 6-1 Define of Raster opration”.
Unsigned char *blend_formula*	Input	Blending expression selection Selects the blending expression of the color data in the BRU. Blending coefficients are specified by the *blend_coefx* and *blend_coefy*. If set to VSP_RBC_BLEND the *rbc*, can be used. VSP_FORM_BLEND0 (0x00) : coefficient x * (DST color data) + coefficient y * (SRC color data) VSP_FORM_BLEND1 (0x01) : coefficient x * (DST color data) – coefficient y * (SRC color data)
unsigned char *blend_coefx*	Input	Blending coefficient X selection VSP_COEFFICIENT_BLENDX1 (0x00) : (DST alpha data) VSP_COEFFICIENT_BLENDX2 (0x01) : 255-(DST alpha data) VSP_COEFFICIENT_BLENDX3 (0x02) : (SRC alpha data) VSP_COEFFICIENT_BLENDX4 (0x03) : 255-(SRC alpha data) VSP_COEFFICIENT_BLENDX5 (0x04) : (*acoefx_fix*)
unsigned char *blend_coefy*	Input	Blending coefficient Y selection VSP_COEFFICIENT_BLENDY1 (0x00) : (DST alpha data) VSP_COEFFICIENT_BLENDY2 (0x01) : 255-(DST alpha data) VSP_COEFFICIENT_BLENDY3 (0x02) : (SRC alpha data) VSP_COEFFICIENT_BLENDY4 (0x03) : 255-(SRC alpha data) VSP_COEFFICIENT_BLENDY5 (0x04) : (*acoefy_fix*)

unsigned char *aformula*	Input	Blending alpha creation expression Specifies the expression for creating alpha data after blending by blend / ROP unit. Alpha creation coefficients are specified by the *acoefx* and *acoefy*. VSP_FORM_ALPHA0 (0x00) : coefficient x * (DST alpha data) + coefficient y * (SRC alpha data) VSP_FORM_ALPHA1 (0x01) : coefficient x * (DST alpha data) – coefficient y * (SRC alpha data)
unsigned char *acoefx*	Input	Alpha creation coefficient X. VSP_COEFFICIENT_ALPHAX1 (0x00) : (DST alpha data) VSP_COEFFICIENT_ALPHAX2 (0x01) : 255-(DST alpha data) VSP_COEFFICIENT_ALPHAX3 (0x02) : (SRC alpha data) VSP_COEFFICIENT_ALPHAX4 (0x03) : 255-(SRC alpha data) VSP_COEFFICIENT_ALPHAX5 (0x04) : (*acoefx_fix*)
unsigned char *acoefy*	Input	Alpha creation coefficient Y. VSP_COEFFICIENT_ALPHAY1 (0x00) : (DST alpha data) VSP_COEFFICIENT_ALPHAY2 (0x01) : 255-(DST alpha data) VSP_COEFFICIENT_ALPHAY3 (0x02) : (SRC alpha data) VSP_COEFFICIENT_ALPHAY4 (0x03) : 255-(SRC alpha data) VSP_COEFFICIENT_ALPHAY5 (0x04) : (*acoefy_fix*)
unsigned char *acoefx_fix*	Input	Fixed alpha value 1. (0 to 255) This parameter specify fixed alpha value 1 used when the *acoefx* is set to VSP_COEFFICIENT_ALPHAX5 or *blend_coefx* is set to VSP_COEFFICIENT_BLENDX5.
Unsigned char *acoefy_fix*	Input	Fixed alpha value 2. (0 to 255) This parameter specify fixed alpha value 1 used when the *acoefy* is set to VSP_COEFFICIENT_ALPHAY5 or *blend_coefy* is set to VSP_COEFFICIENT_BLENDY5.

(c) T_VSP_BLEND_ROP

The following is described about the member of T_VSP_BLEND_ROP structure.

```
Typedef struct{
    unsigned char       crop;
    unsigned char       arop;
} T_VSP_BLEND_ROP;
```

Member	Direction	Contents
unsigned char *crop*	Input	Raster operation setting of color data. Can specify the defined “Table 6-1 Define of Raster opration”.
Unsigned char *arop*	Input	Raster operation setting of alpha value. Can specify the defined “Table 6-1 Define of Raster opration”.

6.1.3.8. T_VSP_HGO

The following is described about the member of T_VSP_HGO structure.

```
Typedef struct{
    void            *addr;
    unsigned short  width;
    unsigned short  height;
    unsigned short  x_offset;
    unsigned short  y_offset;
    unsigned char   binary_mode ;
    unsigned char   maxrgb_mode ;
    unsigned short  x_skip ;
    unsigned short  y_skip ;
    unsigned long   sampling ;
} T_VSP_HGO ;
```

Member	Direction	Contents
void *addr	Output	Pointer to a histogram result. 4 byte alignment is required. Also, specify the logocal address.
Unsigned short *width*	Input	Horizontal size of histogram detection window. (1 to 8190) [pixel unit]
unsigned short *height*	Input	Vertical size of histogram detection window. (1 to 8190) [line]
unsigned short *x_offset*	Input	Horizontal offset of histogram detection window. (0 to 8189) [pixel unit] If '*width* + *x_offset*' is greater than 8190, VSP will return error.
Unsigned short *y_offset*	Input	Vertical size of histogram detection window. (0 to 8189) [line] If '*height* + *y_offset*' is greater than 8190, VSP will return error.
Unsigned char *binary_mode*	Input	Offset binary mode setting. In offset binary mode, values are converted to absolute values before they are used to detect the maximum value, minimum value, sum, and black band. Note that values without conversion are always used for histogram creation regardless of this mode setting. VSP_STRAIGHT_BINARY (0x00) : straight binary mode VSP_OFFSET_BINARY (0x50) : offset binary mode Note: VSP_OFFSET_BINARY is available only YUV. When color space of target is RGB, recommend to set VSP_STRAIGHT_BINARY.
Unsigned char *maxrgb_mode*	Input	Histgram source component setting. VSP_MAXRGB_OFF (0x00): 3 color components independently. VSP_MAXRGB_ON (0x80): the maximum value of RGB data. Note: VSP_MAXRGB_ON is available only RGB. When color space of target is other than RGB, must set VSP_MAXRGB_OFF.

Unsigned short *x_skip*	Input	Horizontal pixel skipping mode setting VSP_SKIP_OFF (0x00) : No skipping. VSP_SKIO_1_2 (0x01) : Horizonta 1/2 skipping. One pixel is discarded from every two pixels before a histogram is created. VSP_SKIP_1_4 (0x02) : Horizontal 1/4 skipping. Three pixels are discarded from every four pixels before a histogram is created.
Unsigned short *y_skip*	Input	Vertical pixel skipping mode setting. Refer to *x_skip* parameter.
Unsigned long *sampling*	Input	Detection module setting. You can specify from the following modules to be detected. If you specify a module you don't use, returns the parameter error. VSP_SMPPT_SRC1 (0) : 1st input source VSP_SMPPT_SRC2 (1) : 2nd input source VSP_SMPPT_SRC3 (2) : 3rd input source VSP_SMPPT_SRC4 (3) : 4th input source VSP_SMPPT_SRU (16) : Super-resolution VSP_SMPPT_UDS (17) : Up down scaler VSP_SMPPT_UDS1 (18) : Up down scaler VSP_SMPPT_UDS2 (19) : Up down scaler VSP_SMPPT_LUT (22) : Look up table VSP_SMPPT_BRU (27) : Blend ROP VSP_SMPPT_CLU (29) : Cubic-Look up table VSP_SMPPT_HST (30) : Hue saturation value transform VSP_SMPPT_HSI (31) : Hue saturation value transform inverse

The HGO uses 768 (32bit * 64 * 3) bytes. Be allocating memory over 768 bytes. Also VSP manager write to buffer by 4 byte unit. Be careful endian when you read buffer by byte unit.

Offset	Component	Bit[31:0]
+0	VSP_MAXRGB_OFF: R/Cr/H	HISTGRAM_0[21:0]
+1		HISTGRAM_1[21:0]
...		...
+62	VSP_MAXRGB_ON: n/a *1	HISTGRAM_62[21:0]
+63		HISTGRAM_63[21:0]
+64	VSP_MAXRGB_OFF: G/Y/S	HISTGRAM_0[21:0]
+65		HISTGRAM_1[21:0]
...		...
+126	VSP_MAXRGB_ON: max(R, G, B) *2	HISTGRAM_62[21:0]
+127		HISTGRAM_63[21:0]
+128	VSP_MAXRGB_OFF: B/Cb/V	HISTGRAM_0[21:0]
+129		HISTGRAM_1[21:0]
...		...
+190	VSP_MAXRGB_ON: n/a *1	HISTGRAM_62[21:0]
+191		HISTGRAM_63[21:0]

Note1: When specify VSP_MAXRGB_ON, not ensured.
Note2: max(R, G, B) indicates maximum value of input R, G, and B data.

6.1.3.9. T_VSP_HGT

The following is described about the member of T_VSP_HGT structure.

```
Typedef struct{
    void                *addr;
    unsigned short      width;
    unsigned short      height;
    unsigned short      x_offset;
    unsigned short      y_offset;
    unsigned short      x_skip ;
    unsigned short      y_skip ;
    T_VSP_HUE_AREA      area[6] ;
    unsigned long       sampling ;
} T_VSP_HGT ;
```

Member	Direction	Contents
void *addr	Output	Pointer to a histogram result. 4 byte alignment is required. Also, specify the logocal address.
Unsigned short *width*	Input	Horizontal size of histogram detection window. (1 to 8190) [pixel unit]
unsigned short *height*	Input	Vertical size of histogram detection window. (1 to 8190) [line]
unsigned short *x_offset*	Input	Horizontal offset of histogram detection window. (0 to 8189) [pixel unit] If '*width* + *x_offset*' is greater than 8190, VSP will return error.
Unsigned short *y_offset*	Input	Vertical size of histogram detection window. (0 to 8189) [line] If '*height* + *y_offset*' is greater than 8190, VSP will return error.
Unsigned short *x_skip*	Input	Horizontal pixel skipping mode setting VSP_SKIP_OFF (0x00) : No skipping. VSP_SKIO_1_2 (0x01) : Horizonta 1/2 skipping. One pixel is discarded from every two pixels before a histogram is created. VSP_SKIP_1_4 (0x02) : Horizontal 1/4 skipping. Three pixels are discarded from every four pixels before a histogram is created.
Unsigned short *y_skip*	Input	Vertical pixel skipping mode setting. Refer to *x_skip* parameter.
T_VSP_HUE_AREA *area[6]*	Input	HUE area structure. Please refer to the T_VSP_HUE_AREA structure.
Unsigned long *sampling*	Input	Detection module setting. You can specify from the following modules to be detected. If you specify a module you don't use, returns the parameter error. VSP_SMPPT_SRC1 (0) : 1st input source VSP_SMPPT_SRC2 (1) : 2nd input source VSP_SMPPT_SRC3 (2) : 3rd input source VSP_SMPPT_SRC4 (3) : 4th input source VSP_SMPPT_SRU (16) : Super-resolution VSP_SMPPT_UDS (17) : Up down scaler VSP_SMPPT_UDS1 (18) : Up down scaler VSP_SMPPT_UDS2 (19) : Up down scaler

		VSP_SMPPT_LUT	(22) : Look up table
		VSP_SMPPT_BRU	(27) : Blend ROP
		VSP_SMPPT_CLU	(29) : Cubic-Look up table
		VSP_SMPPT_HST	(30) : Hue saturation value transform
		VSP_SMPPT_HSI	(31) : Hue saturation value transform inverse

(a) T_VSP_HUE_AREA

The following is described about the member of T_VSP_HUE_AREA structure.

```
Typedef struct{
    unsigned char       lower;
    unsigned char       upper;
} T_VSP_HUE_AREA;
```

Member	Direction	Contents
unsigned char *lower*	Input	Lower boundary value for hue area. (0 to 255)
unsigned char *upper*	Input	Upper boundary value for hue area. (0 to 255)

Set the HUE Area as shown in Figure 6-6.

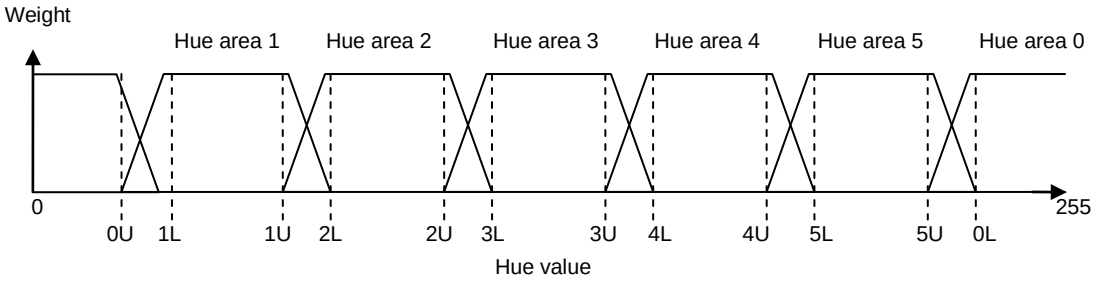


Figure 6-6 Weighting Histgram Using Hue

0L = *area[0].lower* 0U = *area[0].upper*
1L = *area[1].lower* 1U = *area[1].upper*
...
5L = *area[5].lowe* 5U = *area[5].upper*

0L <= 0U <= 1L <= 1U <= 2L <= 2U <= 3L <= 3U <= 4L <= 4U <= 5L <= 5U
0U <= 1L <= 1U <= 2L <= 2U <= 3L <= 3U <= 4L <= 4U <= 5L <= 5U <= 0L

The HGT uses 768 (32bit * 64 * 3) bytes. Be allocating memory over 768 bytes. Also VSP manager write to buffer by 4 byte unit. Be careful endian when you read buffer by byte unit.

Offset	Hue area	Bit[31:0]
+0	m = 0	HISTGRAM_0[21:0]
+1		HISTGRAM_1[21:0]
...		...
+30		HISTGRAM_30[21:0]
+31		HISTGRAM_31[21:0]
+32	m = 1	HISTGRAM_0[21:0]
+33		HISTGRAM_1[21:0]
...		...
+62		HISTGRAM_30[21:0]
+63		HISTGRAM_31[21:0]
…		
+160	m = 5	HISTGRAM_0[21:0]
+161		HISTGRAM_1[21:0]
...		...
+190		HISTGRAM_30[21:0]
+191		HISTGRAM_31[21:0]

6.2. Input/Output image limited size

Table 6-5 and Table 6-6 show usable input and output size in each module. If you use module of limited input and output, it's necessary to consider the size of the output module connected to earlier. Example, if input size of RPF is greater than 2048, can not use the SRU and UDS.

Table 6-5 Minimum size of input/output image

Processing module		Input [pixel]		Ouput [pixel]	
		width	height	width	height
RPF		1	1	1	1
SRU	Normal size	4	4	4	4
	Double size	4	4	4	4
UDS	Scale-down	4	4	4	4
	Scale-up	4	4	4	4
LUT		1	1	1	1
CLU		1	1	1	1
HST		1	1	1	1
HIS		1	1	1	1
BRU		1	1	1	1
HGO		1	1	1	1
HGT		1	1	1	1
WPF		1	1	1	1

Table 6-6 Maximum size of input/output image

Processing module		Input [pixel]		Ouput [pixel]	
		width	height	width	height
RPF		8190	8190	8190	8190
SRU	Normal size	2048	8190	2048	8190
	Double size	1024	4095	2048	8190
UDS	Scale-down	8190	8190	2048	2048
	Scale-up	2048	8190	2048	2048
LUT		8190	8190	8190	8190
CLU		8190	8190	8190	8190
HST		8190	8190	8190	8190
HIS		8190	8190	8190	8190
BRU		8190	8190	8190	8190
HGO		8190	8190	8190	8190
HGT		8190	8190	8190	8190
WPF		2048	2048	2048	2048

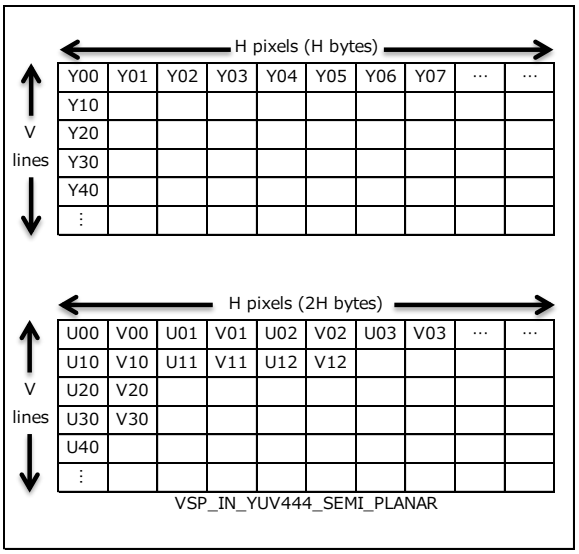
6.3. Format

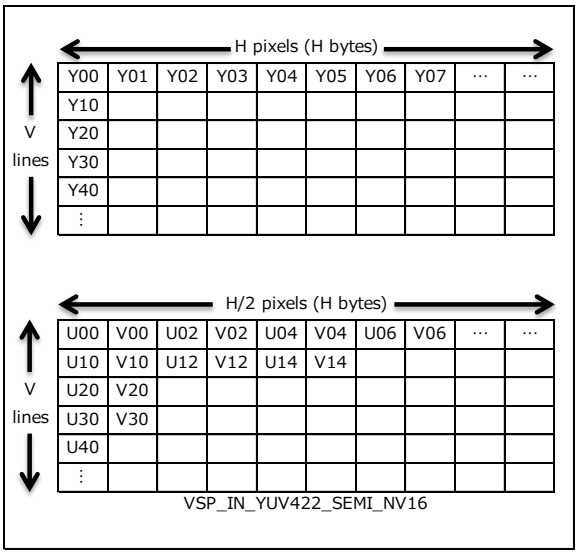
6.3.1. Input format

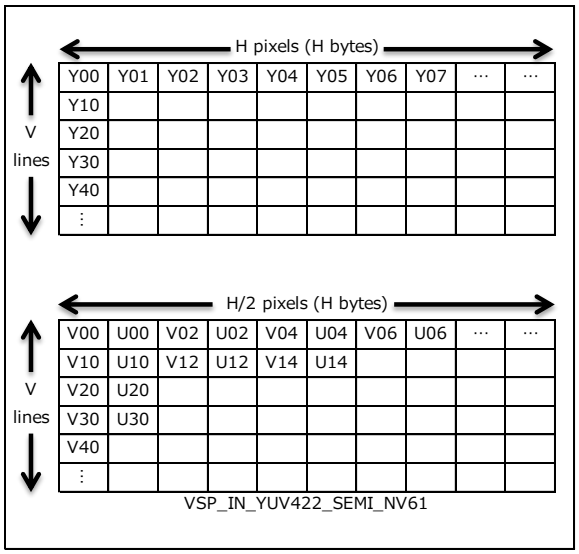
6.3.1.1. RGB format

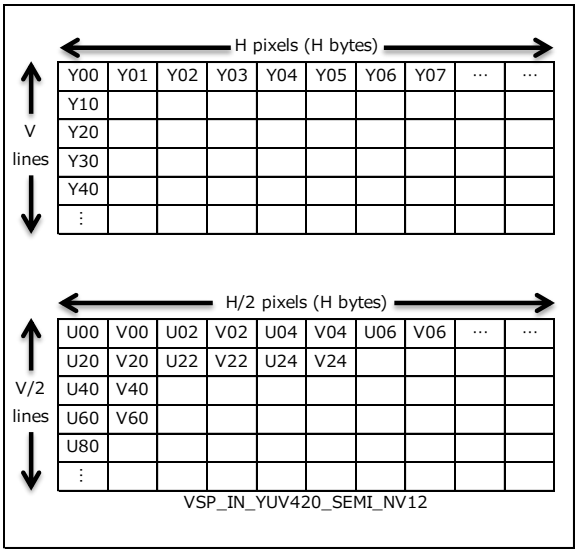
format	byte	phase	bit																											
			31 to 24								23 to 16								15 to 8								7 to 0			
VSP_IN_RGB332	1		R0	R0	R0	G0	G0	G0	B0	B0	R1	R1	R1	G1	G1	G1	B1	B1	R2	R2	R2	G2	G2	G2	B2	B2	R3	R3	R3	G3	G3	G3	B3	B3
VSP_IN_XRGB4444	2						R0	R0	R0	R0	G0	G0	G0	G0	B0	B0	B0	B0				R1	R1	R1	R1	G1	G1	G1	G1	B1	B1	B1	B1
VSP_IN_RGBX4444	2		R0	R0	R0	R0	G0	G0	G0	G0	B0	B0	B0	B0					R1	R1	R1	R1	G1	G1	G1	G1	B1	B1	B1	B1				
VSP_IN_XRGB1555	2			R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0		R1	R1	R1	R1	R1	G1	G1	G1	G1	B1	B1	B1	B1	B1
VSP_IN_RGBX5551	2		R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	R1	R1	R1	R1	R1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1	
VSP_IN_RGB565	2		R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	R1	R1	R1	R1	R1	G1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1
VSP_IN_AXRGB86666	4		A0	A0	A0	A0	A0	A0	A0	A0							R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0
VSP_IN_RGBXA66668	4		R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0							A0	A0	A0	A0	A0	A0	A0	A0
VSP_IN_XRGBA66668	4								R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	A0	A0	A0	A0	A0	A0	A0	A0
VSP_IN_ARGBX86666	4		A0	A0	A0	A0	A0	A0	A0	A0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0						
VSP_IN_AXXXRGB82666	4		A0	A0	A0	A0	A0	A0	A0	A0			R0	R0	R0	R0	R0	R0			G0	G0	G0	G0	G0	G0			B0	B0	B0	B0	B0	B0
VSP_IN_XXXRGBA26668	4				R0	R0	R0	R0	R0	R0			G0	G0	G0	G0	G0	G0			B0	B0	B0	B0	B0	B0	A0	A0	A0	A0	A0	A0	A0	A0
VSP_IN_ARGBXXX86662	4		A0	A0	A0	A0	A0	A0	A0	A0	R0	R0	R0	R0	R0	R0			G0	G0	G0	G0	G0	G0			B0	B0	B0	B0	B0	B0		
VSP_IN_RGBXXXA66628	4		R0	R0	R0	R0	R0	R0			G0	G0	G0	G0	G0	G0			B0	B0	B0	B0	B0	B0			A0	A0	A0	A0	A0	A0	A0	A0
VSP_IN_XRGB6666	3	0							R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0						R1	R1
		1	R1	R1	R1	R1	G1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1	B1							R2	R2	R2	R2	R2	R2	G2	G2	G2	G2
		2	G2	G2	B2	B2	B2	B2	B2	B2							R3	R3	R3	R3	R3	R3	G3	G3	G3	G3	G3	G3	B3	B3	B3	B3	B3	B3
VSP_IN_RGBX6666	3	0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0							R1	R1	R1	R1	R1	R1	G1	G1
		1	G1	G1	G1	G1	B1	B1	B1	B1	B1	B1							R2	R2	R2	R2	R2	R2	G2	G2	G2	G2	G2	G2	B2	B2	B2	B2
		2	B2	B2							R3	R3	R3	R3	R3	R3	G3	G3	G3	G3	G3	G3	B3	B3	B3	B3	B3	B3						
VSP_IN_XXXRGB2666	3	0			R0	R0	R0	R0	R0	R0			G0	G0	G0	G0	G0	G0			B0	B0	B0	B0	B0	B0			R1	R1	R1	R1	R1	R1
		1			G1	G1	G1	G1	G1	G1			B1	B1	B1	B1	B1	B1			R2	R2	R2	R2	R2	R2			G2	G2	G2	G2	G2	G2
		2			B2	B2	B2	B2	B2	B2			R3	R3	R3	R3	R3	R3			G3	G3	G3	G3	G3	G3			B3	B3	B3	B3	B3	B3
VSP_IN_RGBXXX6662	3	0	R0	R0	R0	R0	R0	R0			G0	G0	G0	G0	G0	G0			B0	B0	B0	B0	B0	B0			R1	R1	R1	R1	R1	R1		
		1	G1	G1	G1	G1	G1	G1			B1	B1	B1	B1	B1	B1			R2	R2	R2	R2	R2	R2			G2	G2	G2	G2	G2	G2		
		2	B2	B2	B2	B2	B2	B2			R3	R3	R3	R3	R3	R3			G3	G3	G3	G3	G3	G3			B3	B3	B3	B3	B3	B3		
VSP_IN_ARGB8888	4		A0	A0	A0	A0	A0	A0	A0	A0	R0	R0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	B0	B0
VSP_IN_RGBA8888	4		R0	R0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	B0	B0	A0	A0	A0	A0	A0	A0	A0	A0
VSP_IN_RGB888	3	0	R0	R0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	B0	B0	R1	R1	R1	R1	R1	R1	R1	R1
		1	G1	G1	G1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1	B1	B1	B1	R2	R2	R2	R2	R2	R2	R2	R2	G2	G2	G2	G2	G2	G2	G2	G2
		2	B2	B2	B2	B2	B2	B2	B2	B2	R3	R3	R3	R3	R3	R3	R3	R3	G3	G3	G3	G3	G3	G3	G3	G3	B3	B3	B3	B3	B3	B3	B3	B3
VSP_IN_XXRGB7666	4								R0	R0	R0	R0	R0	R0	G0	G0	G0	G0							G0	G0	G0	G0	B0	B0	B0	B0	B0	B0
VSP_IN_XRGB14666	4																R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0
VSP_IN_BGR888	3	0	B0	B0	B0	B0	B0	B0	B0	B0	G0	G0	G0	G0	G0	G0	G0	G0	R0	R0	R0	R0	R0	R0	R0	R0	B1	B1	B1	B1	B1	B1	B1	B1
		1	G1	G1	G1	G1	G1	G1	G1	G1	R1	R1	R1	R1	R1	R1	R1	R1	B2	B2	B2	B2	B2	B2	B2	B2	G2	G2	G2	G2	G2	G2	G2	G2
		2	R2	R2	R2	R2	R2	R2	R2	R2	B3	B3	B3	B3	B3	B3	B3	B3	G3	G3	G3	G3	G3	G3	G3	G3	R3	R3	R3	R3	R3	R3	R3	R3
VSP_IN_ARGB4444	2		A0	A0	A0	A0	R0	R0	R0	R0	G0	G0	G0	G0	B0	B0	B0	B0	A1	A1	A1	A1	R1	R1	R1	R1	G1	G1	G1	G1	B1	B1	B1	B1
VSP_IN_RGBA4444	2		R0	R0	R0	R0	G0	G0	G0	G0	B0	B0	B0	B0	A0	A0	A0	A0	R1	R1	R1	R1	G1	G1	G1	G1	B1	B1	B1	B1	A1	A1	A1	A1
VSP_IN_ARGB1555	2		A0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	A1	R1	R1	R1	R1	R1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1
VSP_IN_RGBA5551	2		R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	A0	R1	R1	R1	R1	R1	G1	G1	G1	G1	G1	B1	B1	B1	B1	A1
VSP_IN_ABGR4444	2		A0	A0	A0	A0	B0	B0	B0	B0	G0	G0	G0	G0	R0	R0	R0	R0	A1	A1	A1	A1	B1	B1	B1	B1	G1	G1	G1	G1	R1	R1	R1	R1
VSP_IN_BGRA4444	2		B0	B0	B0	B0	G0	G0	G0	G0	R0	R0	R0	R0	A0	A0	A0	A0	B1	B1	B1	B1	G1	G1	G1	G1	R1	R1	R1	R1	A1	A1	A1	A1
VSP_IN_ABGR1555	2		A0	B0	B0	B0	B0	B0	G0	G0	G0	G0	G0	R0	R0	R0	R0	R0	A1	B1	B1	B1	B1	B1	G1	G1	G1	G1	G1	R1	R1	R1	R1	R1
VSP_IN_BGRA5551	2		B0	B0	B0	B0	B0	G0	G0	G0	G0	G0	R0	R0	R0	R0	R0	A0	B1	B1	B1	B1	B1	G1	G1	G1	G1	G1	R1	R1	R1	R1	R1	A1
VSP_IN_XXXBGR2666	3	0			B0	B0	B0	B0	B0	B0			G0	G0	G0	G0	G0	G0			R0	R0	R0	R0	R0	R0			B1	B1	B1	B1	B1	B1
		1			G1	G1	G1	G1	G1	G1			R1	R1	R1	R1	R1	R1			B2	B2	B2	B2	B2	B2			G2	G2	G2	G2	G2	G2
		2			R2	R2	R2	R2	R2	R2			B3	B3	B3	B3	B3	B3			G3	G3	G3	G3	G3	G3			R3	R3	R3	R3	R3	R3
VSP_IN_ABGR8888	4		A0	A0	A0	A0	A0	A0	A0	A0	B0	B0	B0	B0	B0	B0	B0	B0	G0	G0	G0	G0	G0	G0	G0	G0	R0	R0	R0	R0	R0	R0	R0	R0
VSP_IN_XRGB16565	4																		R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0
VSP_IN_RGB_CLUT_DATA			RGB_CLUT_DATA0								RGB_CLUT_DATA1								RGB_CLUT_DATA2								RGB_CLUT_DATA3							

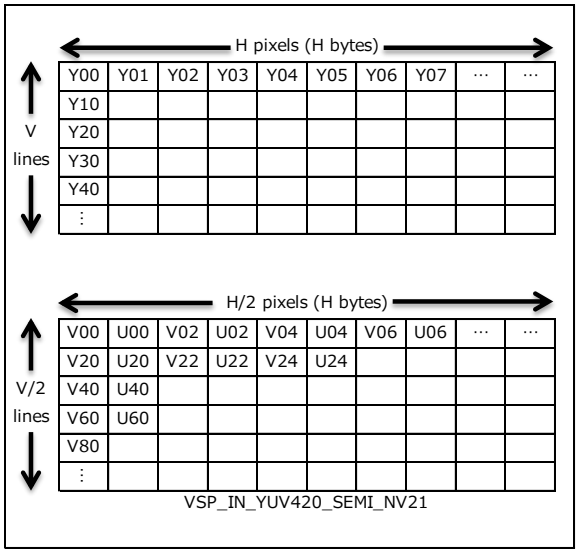
6.3.1.2. YcbCr (Semi planar) format



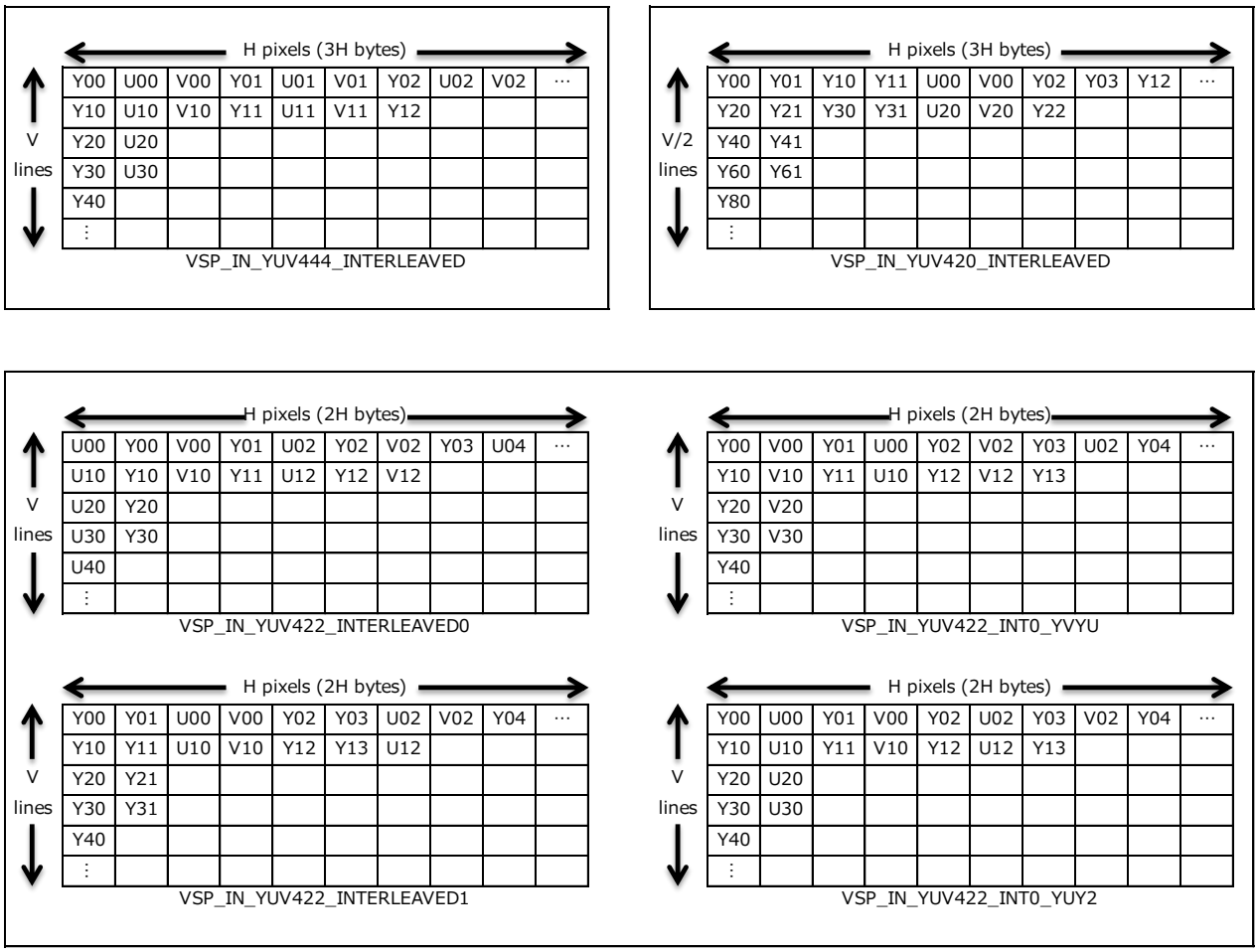




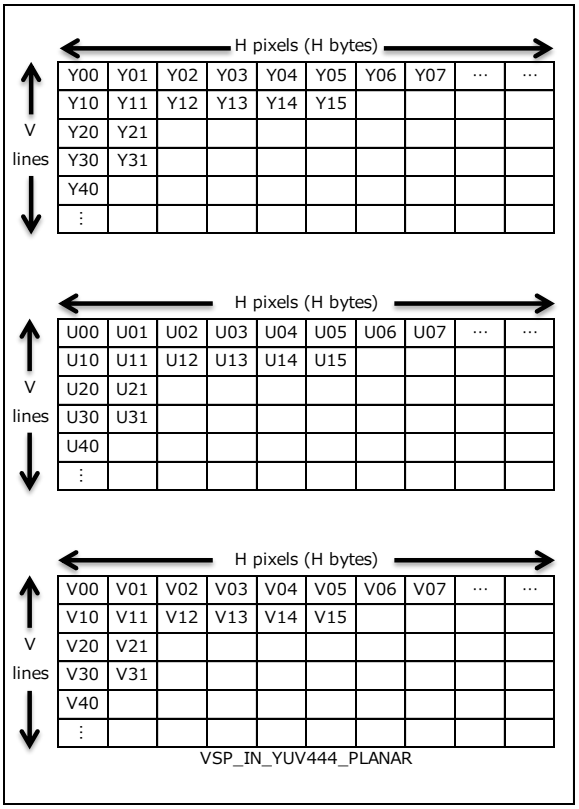


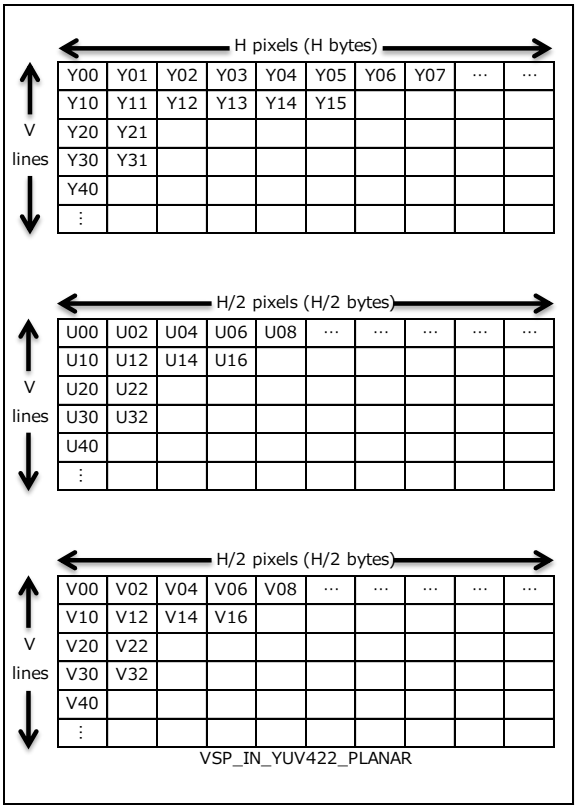


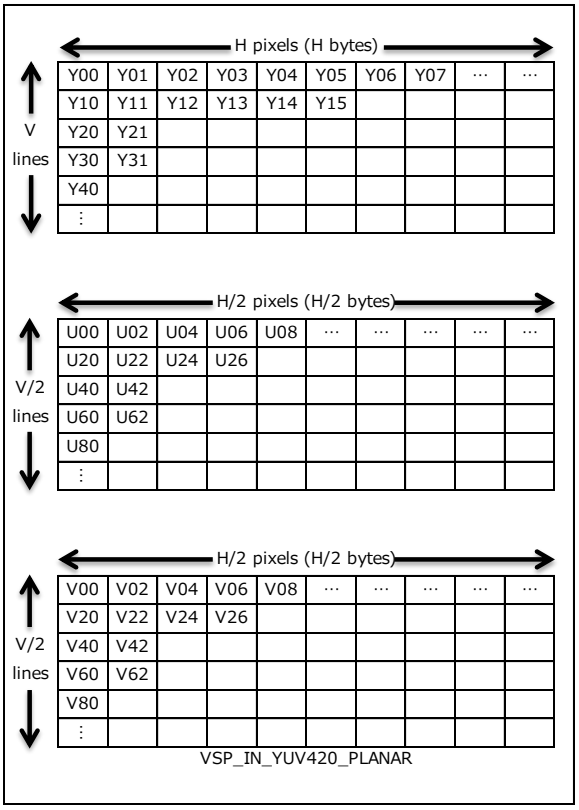
6.3.1.3. YcbCr (Interleaved) format



6.3.1.4. YcbCr (Planar) format





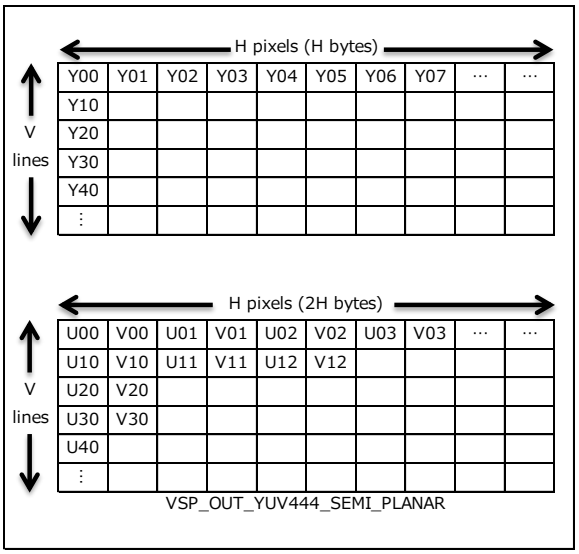


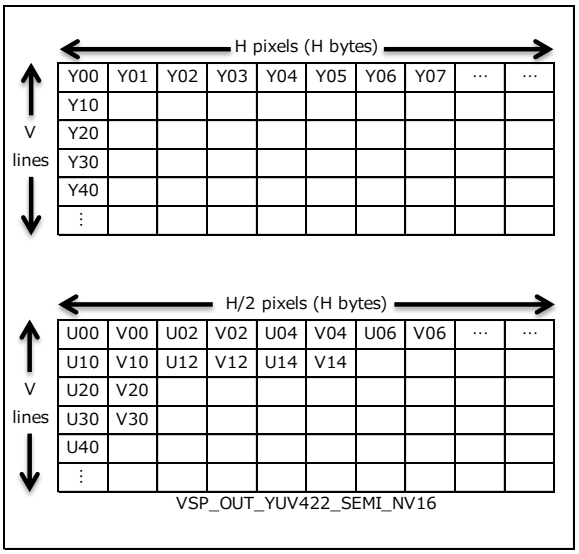
6.3.2. Output format

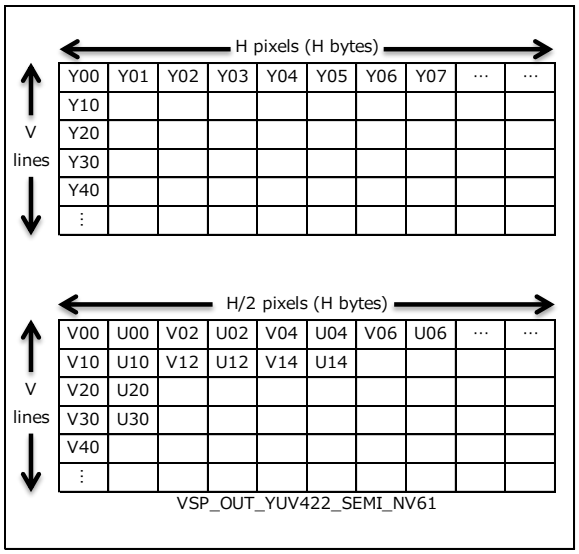
6.3.2.1. RGB format

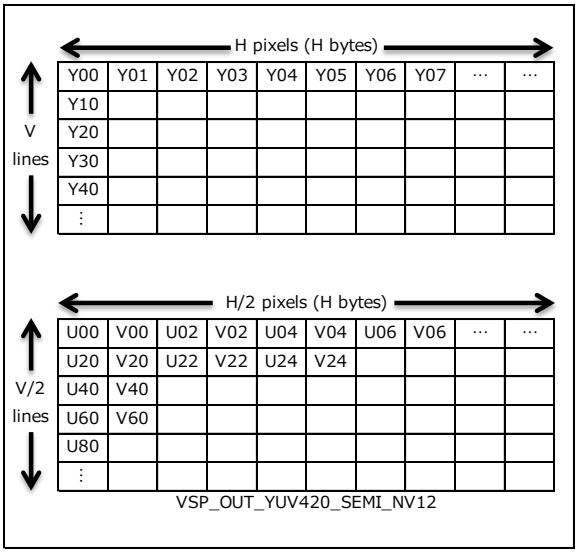
format	byte	phase	bit																											
			31 to 24								23 to 16								15 to 8								7 to 0			
VSP_OUT_RGB332	1		R0	R0	R0	G0	G0	G0	B0	B0	R1	R1	R1	G1	G1	G1	B1	B1	R2	R2	R2	G2	G2	G2	B2	B2	R3	R3	R3	G3	G3	G3	B3	B3
VSP_OUT_XRGB4444	2						R0	R0	R0	R0	G0	G0	G0	G0	B0	B0	B0	B0					R1	R1	R1	R1	G1	G1	G1	G1	B1	B1	B1	B1
VSP_OUT_RGBX4444	2		R0	R0	R0	R0	G0	G0	G0	G0	B0	B0	B0	B0					R1	R1	R1	R1	G1	G1	G1	G1	B1	B1	B1	B1				
VSP_OUT_XRGB1555	2			R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0		R1	R1	R1	R1	R1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1
VSP_OUT_RGBX5551	2		R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0		R1	R1	R1	R1	R1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1	
VSP_OUT_RGB565	2		R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	R1	R1	R1	R1	R1	G1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1
VSP_OUT_PXRGB86666	4		P0	P0	P0	P0	P0	P0	P0	P0							R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0
VSP_OUT_RGBXP66668	4		R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0							P0	P0	P0	P0	P0	P0	P0	P0
VSP_OUT_XRGBP66668	4								R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	P0	P0	P0	P0	P0	P0	P0	P0
VSP_OUT_PRGBX86666	4		P0	P0	P0	P0	P0	P0	P0	P0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0						
VSP_OUT_PXXXRGB82666	4		P0	P0	P0	P0	P0	P0	P0	P0			R0	R0	R0	R0	R0	R0			G0	G0	G0	G0	G0	G0			B0	B0	B0	B0	B0	B0
VSP_OUT_XXXRGBP26668	4				R0	R0	R0	R0	R0	R0			G0	G0	G0	G0	G0	G0			B0	B0	B0	B0	B0	B0	P0	P0	P0	P0	P0	P0	P0	P0
VSP_OUT_PRGBXXX86662	4		P0	P0	P0	P0	P0	P0	P0	P0	R0	R0	R0	R0	R0	R0			G0	G0	G0	G0	G0	G0			B0	B0	B0	B0	B0	B0		
VSP_OUT_RGBXXXP66628	4		R0	R0	R0	R0	R0	R0			G0	G0	G0	G0	G0	G0			B0	B0	B0	B0	B0	B0			P0	P0	P0	P0	P0	P0	P0	P0
VSP_OUT_XRGB6666	3	0							R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0						R1	R1	
		1	R1	R1	R1	R1	G1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1	B1							R2	R2	R2	R2	R2	R2	G2	G2	G2	G2
		2	G2	G2	B2	B2	B2	B2	B2	B2							R3	R3	R3	R3	R3	R3	G3	G3	G3	G3	G3	G3	B3	B3	B3	B3	B3	B3
VSP_OUT_RGBX6666	3	0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0							R1	R1	R1	R1	R1	R1	G1	G1
		1	G1	G1	G1	G1	B1	B1	B1	B1	B1	B1							R2	R2	R2	R2	R2	R2	G2	G2	G2	G2	G2	G2	B2	B2	B2	B2
		2	B2	B2							R3	R3	R3	R3	R3	R3	G3	G3	G3	G3	G3	G3	B3	B3	B3	B3	B3	B3						
VSP_OUT_XXXRGB2666	3	0			R0	R0	R0	R0	R0	R0			G0	G0	G0	G0	G0	G0			B0	B0	B0	B0	B0	B0			R1	R1	R1	R1	R1	R1
		1			G1	G1	G1	G1	G1	G1			B1	B1	B1	B1	B1	B1			R2	R2	R2	R2	R2	R2			G2	G2	G2	G2	G2	G2
		2			B2	B2	B2	B2	B2	B2			R3	R3	R3	R3	R3	R3			G3	G3	G3	G3	G3	G3			B3	B3	B3	B3	B3	B3
VSP_OUT_RGBXXX6662	3	0	R0	R0	R0	R0	R0	R0			G0	G0	G0	G0	G0	G0			B0	B0	B0	B0	B0	B0			R1	R1	R1	R1	R1	R1		
		1	G1	G1	G1	G1	G1	G1			B1	B1	B1	B1	B1	B1			R2	R2	R2	R2	R2	R2			G2	G2	G2	G2	G2	G2		
		2	B2	B2	B2	B2	B2	B2			R3	R3	R3	R3	R3	R3			G3	G3	G3	G3	G3	G3			B3	B3	B3	B3	B3	B3		
VSP_OUT_PRGB8888	4		P0	P0	P0	P0	P0	P0	P0	P0	R0	R0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	B0	B0
VSP_OUT_RGBP8888	4		R0	R0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	B0	B0	P0	P0	P0	P0	P0	P0	P0	P0
VSP_OUT_RGB888	3	0	R0	R0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	B0	B0	R1	R1	R1	R1	R1	R1	R1	R1
		1	G1	G1	G1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1	B1	B1	B1	R2	R2	R2	R2	R2	R2	R2	R2	G2	G2	G2	G2	G2	G2	G2	G2
		2	B2	B2	B2	B2	B2	B2	B2	B2	R3	R3	R3	R3	R3	R3	R3	R3	G3	G3	G3	G3	G3	G3	G3	G3	B3	B3	B3	B3	B3	B3	B3	B3
VSP_OUT_XXRGB7666	4								R0	R0	R0	R0	R0	R0	G0	G0	G0								G0	G0	G0	B0	B0	B0	B0	B0	B0	
VSP_OUT_XRGB14666	4																R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0
VSP_OUT_BGR888	3	0	B0	B0	B0	B0	B0	B0	B0	B0	G0	G0	G0	G0	G0	G0	G0	G0	R0	R0	R0	R0	R0	R0	R0	R0	B1	B1	B1	B1	B1	B1	B1	B1
		1	G1	G1	G1	G1	G1	G1	G1	G1	R1	R1	R1	R1	R1	R1	R1	R1	B2	B2	B2	B2	B2	B2	B2	B2	G2	G2	G2	G2	G2	G2	G2	G2
		2	R2	R2	R2	R2	R2	R2	R2	R2	B3	B3	B3	B3	B3	B3	B3	B3	G3	G3	G3	G3	G3	G3	G3	G3	R3	R3	R3	R3	R3	R3	R3	R3
VSP_OUT_PRGB4444	2		P0	P0	P0	P0	R0	R0	R0	R0	G0	G0	G0	G0	B0	B0	B0	B0	P1	P1	P1	P1	R1	R1	R1	R1	G1	G1	G1	G1	B1	B1	B1	B1
VSP_OUT_RGBP4444	2		R0	R0	R0	R0	G0	G0	G0	G0	B0	B0	B0	B0	P0	P0	P0	P0	R1	R1	R1	R1	G1	G1	G1	G1	B1	B1	B1	B1	P1	P1	P1	P1
VSP_OUT_PRGB1555	2		P0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	P1	R1	R1	R1	R1	R1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1
VSP_OUT_RGBP5551	2		R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	P0	R1	R1	R1	R1	R1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1	P1
VSP_OUT_PBGR4444	2		P0	P0	P0	P0	B0	B0	B0	B0	G0	G0	G0	G0	R0	R0	R0	R0	P1	P1	P1	P1	B1	B1	B1	B1	G1	G1	G1	G1	R1	R1	R1	R1
VSP_OUT_BGRP4444	2		B0	B0	B0	B0	G0	G0	G0	G0	R0	R0	R0	R0	P0	P0	P0	P0	B1	B1	B1	B1	G1	G1	G1	G1	R1	R1	R1	R1	P1	P1	P1	P1
VSP_OUT_PBGR1555	2		P0	B0	B0	B0	B0	B0	G0	G0	G0	G0	G0	R0	R0	R0	R0	R0	P1	B1	B1	B1	B1	B1	G1	G1	G1	G1	G1	R1	R1	R1	R1	R1
VSP_OUT_BGRP5551	2		B0	B0	B0	B0	B0	G0	G0	G0	G0	G0	G0	R0	R0	R0	R0	R0	P0	B1	B1	B1	B1	B1	G1	G1	G1	G1	G1	G1	R1	R1	R1	R1	R1	P1
VSP_OUT_XXXBGR2666	3	0			B0	B0	B0	B0	B0	B0			G0	G0	G0	G0	G0	G0	G0	G0			R0	R0	R0	R0	R0	R0			B1	B1	B1	B1	B1	B1
		1			G1	G1	G1	G1	G1	G1			R1	R1	R1	R1	R1	R1	R1	R1			B2	B2	B2	B2	B2	B2			G2	G2	G2	G2	G2	G2
		2			R2	R2	R2	R2	R2	R2			B3	B3	B3	B3	B3	B3	B3	B3			G3	G3	G3	G3	G3	G3			R3	R3	R3	R3	R3	R3
VSP_OUT_PBGR8888	4		P0	P0	P0	P0	P0	P0	P0	P0	B0	B0	B0	B0	B0	B0	B0	B0	G0	G0	G0	G0	G0	G0	G0	G0	R0	R0	R0	R0	R0	R0	R0	R0
VSP_OUT_XRGB16565	4																		R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0

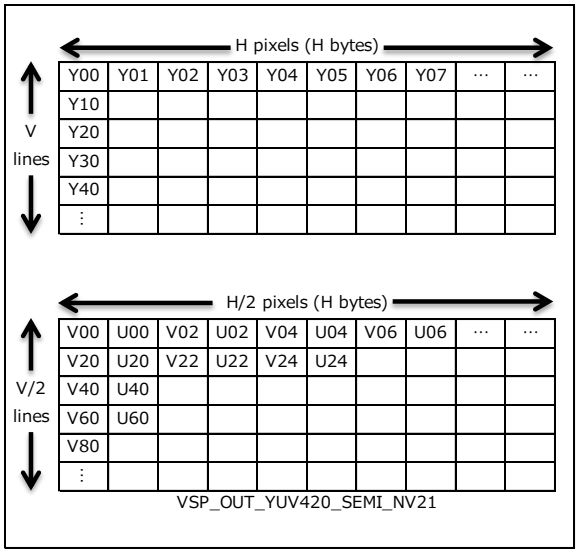
6.3.2.2. YcbCr (Semi planar) format



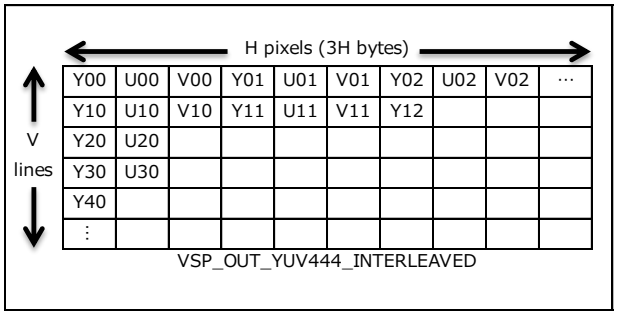


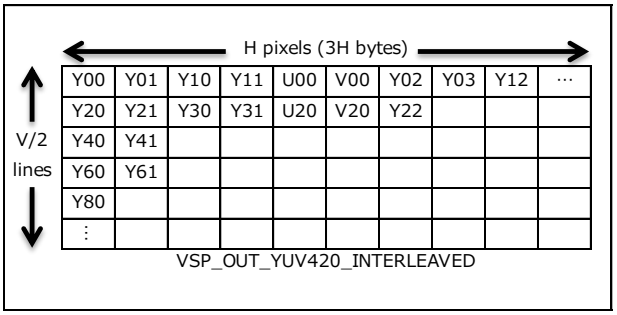


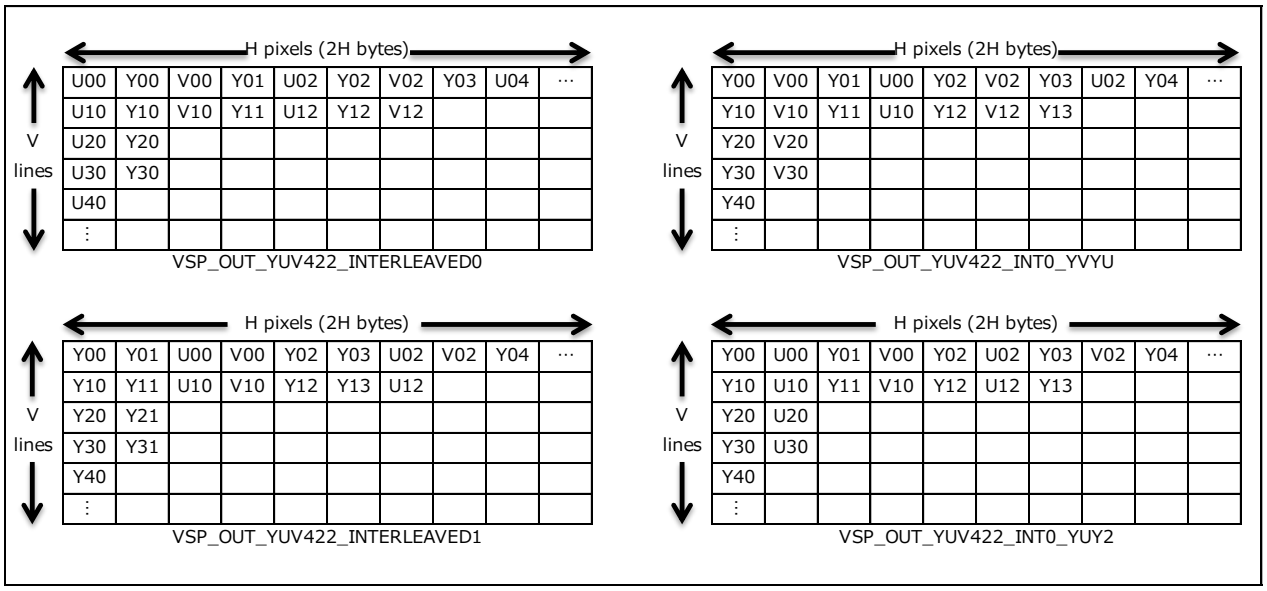




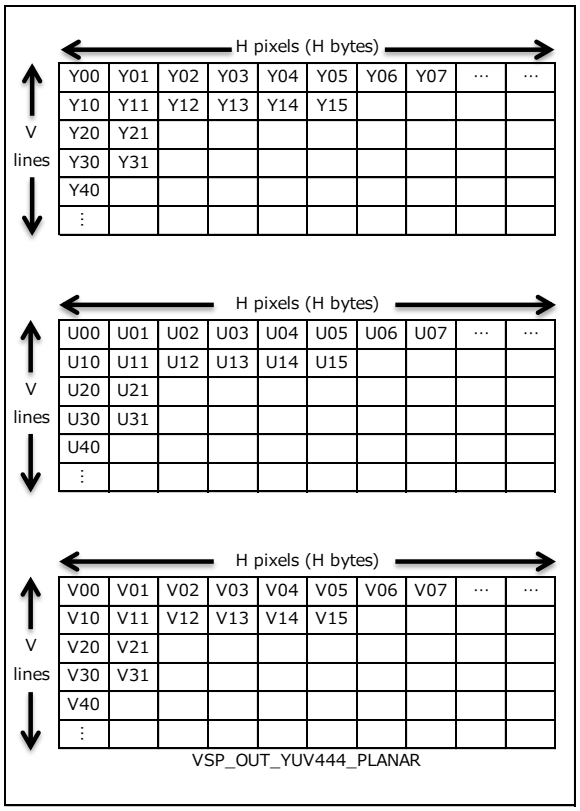
6.3.2.3. YcbCr (Interleaved) format

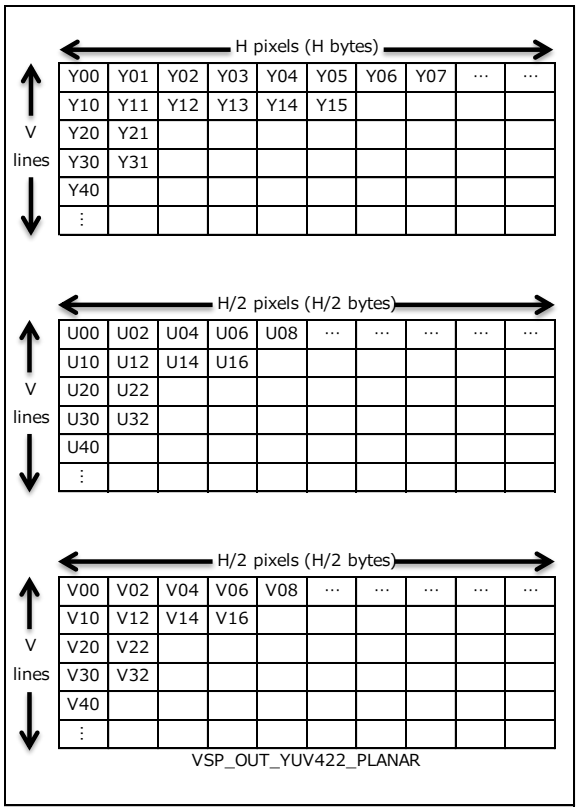


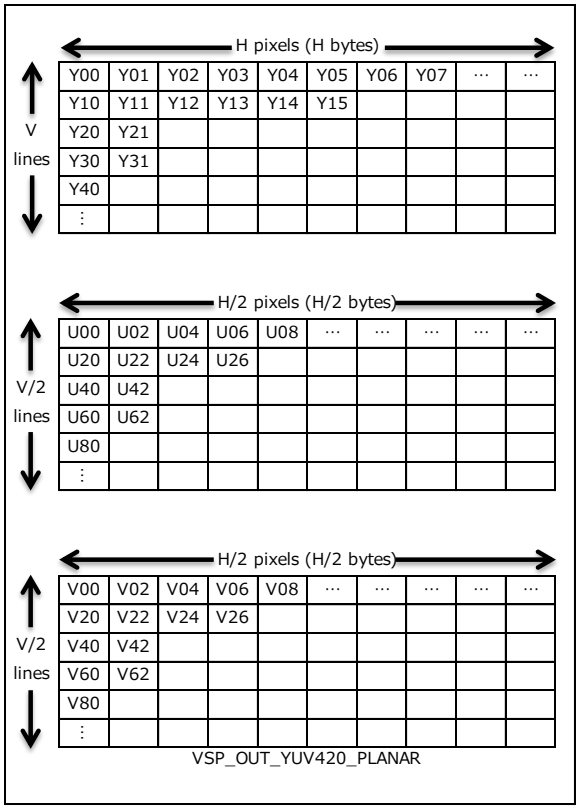




6.3.2.4. YcbCr (Planar) format







6.4. Error code

Table 6-7 shows the detail error code of VSP. Accoding to error code, please check argument.

Table 6-7 Detail of error code

Define name	Error code	Contains
E_VSP_PARA_USEMODULE	-212	Module specified in each connects and use_module don't match.
E_VSP_PARA_OUTPAR	-213	The *dst_par* of T_VSP_START was null pointer.
E_VSP_PARA_CTRLPAR	-214	The *ctrl_par* of T_VSP_START was null pointer.
E_VSP_PARA_CONNECT	-216	Connecting modules were abnormal.
E_VSP_PARA_NOPARENT	-217	All source images (include virtual input) have no VSP_LAYER_PARENT.
E_VSP_PARA_NOINPUT	-218	Not found source image.
E_VSP_PARA_IN_ADR	-220	The *addr* of T_VSP_IN was null pointer. Note: When 'vir' was VSP_NO_VIR.
E_VSP_PARA_IN_ADRC0	-221	Then *addr_c0* of T_VSP_IN was null pointer when source fomat was YUV (semi planar or planar). Note: When *vir* was VSP_NO_VIR.
E_VSP_PARA_IN_ADRC1	-222	The *addr_c1* of T_VSP_IN was null pointer when source format was YUV (planar). Note: When *vir* was VSP_NO_VIR.
E_VSP_PARA_IN_WIDTH	-223	The *width* of T_VSP_IN was out of range 1-8190.
		Then *width* wasn't a multiple of 2 when source format YUV. Note: When *vir* was VSP_NO_VIR.
E_VSP_PARA_IN_HEIGHT	-224	The *height* of T_VSP_IN was out of range 1-8190.
		Then *height* wasn't a multiple of 2 when source format YUV420. Note: When *vir* was VSP_NO_VIR.
E_VSP_PARA_IN_WIDTHEX	-225	When the *width_ex* of T_VSP_IN was other than 0, it was less than *width*.
		The *width_ex* wasn't a multiple of 2 when source format was YUV. Note: When *vir* was VSP_NO_VIR.
E_VSP_PARA_IN_HEIGHTEX	-226	The *height_ex* of T_VSP_IN was other than 0, it was less than *height*.
		The *height_ex* wasn't a multiple of 2 when source format was YUV420. Note: When *vir* was VSP_NO_VIR.
E_VSP_PARA_IN_XOFFSET	-227	The *x_offset* wasn't a multiple of 2 when source format was YUV. Note: When *vir* was VSP_NO_VIR.
E_VSP_PARA_IN_YOFFSET	-228	The *y_offset* wasn't a multiple of 2 when source format was YUV420. Note: When *vir* was VSP_NO_VIR.
E_VSP_PARA_IN_FORMAT	-229	When *vir* was VSP_NO_VIR, the *format* of T_VSP_IN was out of specification.
		When *vir* was VSP_VIR, the *format* of T_VSP_IN was other than VSP_IN_ARGB8888 and VSP_IN_YUV444_SEMI_PLANAR.
E_VSP_PARA_IN_XPOSI	-231	When *pwd* was VSP_LAYER_CHILD, caluculating value of the *x_position* + *width* was greather than input image size.
E_VSP_PARA_IN_YPOSI	-232	When *pwd* was VSP_LAYER_CHILD, caluculating value of the *y_position* + *height* was greather than input image size.
E_VSP_PARA_IN_CIPM	-233	The *cipm* of T_VSP_IN was out of specification.
E_VSP_PARA_IN_CEXT	-234	The *cext* of T_VSP_IN was out of specification.

E_VSP_PARA_IN_CSC	-235	When *vir* was VSP_NO_VIR, the *csc* of T_VSP_IN was out of specification.
		When *vir* was VSP_VIR, the *csc* of T_VSP_IN was other than VSP_CSC_OFF.
E_VSP_PARA_IN_ITURBT	-236	The *iturbt* of T_VSP_IN was out of specification.
E_VSP_PARA_IN_CLRCNG	-237	The *clrcng* of T_VSP_IN was out of specification.
E_VSP_PARA_IN_VIR	-238	The *vir* of T_VSP_IN was out of specification.
E_VSP_PARA_IN_ALPHA	-239	The *alpha_blend* of T_VSP_IN was null pointer.
E_VSP_PARA_IN_CONNECT	-240	The *connect* of T_VSP_IN was out of specification.
E_VSP_PARA_IN_PWD	-241	The *pwd* of T_VSP_IN was out of specification.
E_VSP_PARA_OSD_CLUT	-250	The *clut* of T_VSP_OSDLUT was null pointer.
E_VSP_PARA_OSD_SIZE	-251	The *size* of T_VSP_OSDLUT was out of range 1-256.
E_VSP_PARA_ALPHA_ADR	-260	The *addr_a* of T_VSP_ALPHA was null pointer. Note: When use alpha plane.
E_VSP_PARA_ALPHA_ALPHAN	-261	The *alphan* of T_VSP_ALPHA was specified invalid parameter.
E_VSP_PARA_ALPHA_ASEL	-263	When enable virtual input, the *asel* of T_VSP_ALPHA was other than VSP_ALPHA_NUM5.
		When disable virtual input, the *asel* of T_VSP_ALPHA was out of specification.
E_VSP_PARA_ALPHA_AEXT	-264	The *aext* of T_VSP_ALPHA was out of specification. Note: When the *asel* was VSP_ALPHA_NUM1
E_VSP_PARA_ALPHA_IROP	-265	The *irop* of T_VSP_ALPHA was out of specification. Note: When the *asel* was other than VSP_ALPHA_NUM5
		The *irop* of T_VSP_ALPHA was other than VSP_IROP_NOP Note: When the *asel* was VSP_ALPHA_NUM5
E_VSP_PARA_ALPHA_MSKEN	-266	The *msken* of T_VSP_ALPHA was out of specification.
E_VSP_PARA_ALPHA_BSEL	-267	The *bsel* of T_VSP_ALPHA was out of specification. Note: When the *asel* was VSP_ALPHA_NUM1 or VSP_ALPHA_NUM3, and the *masken* was VSP_MSKEN_ALPHA.
E_VSP_PARA_OUT_ADR	-270	The *addr* of T_VSP_OUT was null pointer.
E_VSP_PARA_OUT_ADRC0	-271	The *addr_c0* of T_VSP_OUT was null pointer when destination format was YUV (semi planar or planar).
E_VSP_PARA_OUT_ADRC1	-272	The *addr_c1* of T_VSP_OUT was null pointer when destination format was YUV (planar).
E_VSP_PARA_OUT_WIDTH	-273	The *width* of T_VSP_OUT was 0.
		The *width* wasn't a multiple of 2 when destination format was YUV.
E_VSP_PARA_OUT_HEIGHT	-274	The *height* of T_VSP_OUT was 0.
		The *height* wasn't a multiple of 2 when destination format was YUV420.
E_VSP_PARA_OUT_XOFFSET	-275	The *x_offset* wasn't a multiple of 2 when destination format was YUV.
E_VSP_PARA_OUT_YOFFSET	-276	The *y_offset* wasn't a multiple of 2 when destination format was YUV420.
E_VSP_PARA_OUT_XCLIP	-277	Caluculating value of the *x_coffse* + *width* was greather than input horizontal size.
E_VSP_PARA_OUT_YCLIP	-278	Caluculating value of the *y_coffset* + *height* was greather than input vertical size.
E_VSP_PARA_OUT_FORMAT	-279	The *format* of T_VSP_OUT was out of specification.
E_VSP_PARA_OUT_PXA	-281	The *pxa* of T_VSP_OUT was out of specification.
E_VSP_PARA_OUT_XCOFFSET	-282	The *x_coffset* of T_VSP_OUT was greater than 255.
E_VSP_PARA_OUT_YCOFFSET	-283	The *y_coffset* of T_VSP_OUT was greater than 255
E_VSP_PARA_OUT_CSC	-284	The *csc* of T_VSP_OUT was out of specification.
E_VSP_PARA_OUT_ITURBT	-285	The *iturbt* of T_VSP_OUT was out of specification.

E_VSP_PARA_OUT_CLRCNG	-286	The *clrcng* of T_VSP_OUT was out of specification.
E_VSP_PARA_OUT_CBRM	-287	The *cbrm* of T_VSP_OUT was out of specification.
E_VSP_PARA_OUT_ABRM	-288	The *abrm* of T_VSP_OUT was out of specification.
E_VSP_PARA_OUT_CLMD	-289	The *clmd* of T_VSP_OUT was out of specification.
E_VSP_PARA_OUT_DITH	-291	The *dith* of T_VSP_OUT was out of specification.
E_VSP_PARA_OUT_INHSV	-292	Color space for input to the WPF was the HSV.
E_VSP_PARA_OUT_INWIDTH	-293	Image horizontal size for input to the WPF was out of range.
E_VSP_PARA_OUT_INHEIGHT	-294	Image vertical size for input to the WPF was out of range.
E_VSP_PARA_OUT_NOTCOLOR	-295	Color space for input and the *format* were mismatched. Note: When The RPF is one or more inputs.
E_VSP_PARA_BRU_LAYORDER	-300	The *lay_order* was specified value over source image number.
		The top back (DSP_A) of *lay_order* was specified VSP_LAY_NO.
E_VSP_PARA_BRU_ADIV	-301	The *adiv* of T_VSP_BRU was out of specification.
E_VSP_PARA_BRU_QNT	-302	The *qnt* of T_VSP_BRU was out of specification.
E_VSP_PARA_BRU_DITH	-303	The *dith* of T_VSP_BRU was out of specification.
E_VSP_PARA_BRU_CONNECT	-304	The *connect* of T_VSP_BRU was out of specification.
E_VSP_PARA_BRU_INHSV	-305	Color space for input to the BRU was the HSV.
E_VSP_PARA_VIR_ADR	-310	The *blend_virtual* of T_VSP_BRU was null pointer. Note: The *lay_order* was specified VSP_LAY_VIRTUAL.
E_VSP_PARA_VIR_WIDTH	-311	The *width* of T_VSP_BLEND_VIRTUAL was out of range 1-8190. Note: The *lay_order* was specified VSP_LAY_VIRTUAL.
E_VSP_PARA_VIR_HEIGHT	-312	The *height* of T_VSP_BLEND_VIRTUAL was out of range 1-8190. Note: The *lay_order* was specified VSP_LAY_VIRTUAL.
E_VSP_PARA_VIR_XPOSI	-313	When *pwd* was VSP_LAYER_CHILD, caluculating value of the *x_position* + *width* was greather than input image size. Note: The *lay_order* was specified VSP_LAY_VIRTUAL.
E_VSP_PARA_VIR_YPOSI	-314	When *pwd* was VSP_LAYER_CHILD, caluculating value of the *y_position* + *height* was greather than input image size. Note: The *lay_order* was specified VSP_LAY_VIRTUAL.
E_VSP_PARA_VIR_PWD	-315	The *pwd* of T_VSP_BLEND_VIRTUAL was out of specification. Note: The *lay_order* was specified VSP_LAY_VIRTUAL.
E_VSP_PARA_BLEND_RBC	-320	The *rbc* of T_VSP_BLEND_CONTROL was out of specification. Note: The blend_control was not null pointer.
E_VSP_PARA_BLEND_CROP	-321	The *crop* of T_VSP_BLEND_CONTROL was out of specification. Note: The *blend_control* was not null pointer.
E_VSP_PARA_BLEND_AROP	-322	The *arop* of T_VSP_BLEND_CONTROL was out of specification. Note: The *blend_control* was not null pointer.
E_VSP_PARA_BLEND_FORM	-323	The *blend_formula* of T_VSP_BLEND_CONTROL was out of specification. Note: The *blend_control* was not null pointer.
E_VSP_PARA_BLEND_COEFX	-324	The *blend_coefx* of T_VSP_BLEND_CONTROL was out of specification. Note: The *blend_control* was not null pointer.
E_VSP_PARA_BLEND_COEFY	-325	The *blend_corfy* of T_VSP_BLEND_CONTROL was out of specification. Note: The *blend_control* was not null pointer.
E_VSP_PARA_BLEND_AFORM	-326	The *aformula* of T_VSP_BLEND_CONTROL was out of specification. Note: The *blend_control* was not null pointer.

E_VSP_PARA_BLEND_ACOEFX	-327	The *acoefx* of T_VSP_BLEND_CONTROL was out of specification. Note: The *blend_control* was not null pointer.
E_VSP_PARA_BLEND_ACOEFY	-328	The *acoefy* of T_VSP_BLEND_CONTROL was out of specification. Note: The *blend_control* was not null pointer.
E_VSP_PARA_ROP_CROP	-330	The *crop* of T_VSP_BLEND_ROP was out of specification. Note: When *blend_rop* was not null pointer.
E_VSP_PARA_ROP_AROP	-331	The *arop* of T_VSP_BLEND_ROP was out of specification. Note: When *blend_rop* was not null pointer.
E_VSP_PARA_SRU_MODE	-340	The *mode* of T_VSP_SRU was out of specification.
E_VSP_PARA_SRU_PARAM	-341	The *param* of T_VSP_SRU was specified invalid parameter.
E_VSP_PARA_SRU_ENSCL	-342	The *enscl* of T_VSP_SRU was out of specification.
E_VSP_PARA_SRU_CONNECT	-343	The *connect* of T_VSP_SRU was out of specification.
E_VSP_PARA_SRU_WIDTH	-344	Image horizontal size for input to the SRU was out of range.
E_VSP_PARA_SRU_HEIGHT	-345	Image vertical size for input to the SRU was out of range.
E_VSP_PARA_SRU_INHSV	-346	Color space for input to the SRU was not the HSV.
E_VSP_PARA_UDS_AMD	-350	The *amd* of T_VSP_UDS was out of specification.
E_VSP_PARA_UDS_FMD	-351	The *fmd* of T_VSP_UDS was out of specification.
E_VSP_PARA_UDS_CLIP	-352	The *clip* of T_VSP_UDS was out of specification. Note: When *alpha* is VSP_ALPHA_ON.
E_VSP_PARA_UDS_ALPHA	-353	The *alpha* of T_VSP_UDS was out of specification.
E_VSP_PARA_UDS_COMP	-354	The *complement* of T_VSP_UDS was out of specification.
		When *complement* was VSP_COMPLEMENT_NN, the *x_ratio* was over 0x4000 or the *y_ratio* was over 0x4000.
		When *complement* was VSP_COMPLEMENT_BC, The *alpha* was VSP_ALPHA_ON,
E_VSP_PARA_UDS_CONNECT	-355	The *connect* of T_VSP_UDS was out of specification.
E_VSP_PARA_UDS_XRATIO	-356	The *x_ratio* of T_VSP_UDS was less than 0x100.
E_VSP_PARA_UDS_YRATIO	-357	The *y_ratio* of T_VSP_UDS was less than 0x100.
E_VSP_PARA_UDS_OUTCWIDTH	-358	Image horizontal size for output to the UDS was out of range.
E_VSP_PARA_UDS_OUTCHEIGHT	-359	Image vertical size for output to the UDS was out of range.
E_VSP_PARA_UDS_INWIDTH	-360	Image horizontal size for input to the UDS was out of range.
E_VSP_PARA_UDS_INHEIGHT	-361	Image vertical size for input to the UDS was out of range.
E_VSP_PARA_LUT	-600	The *lut* of T_VSP_LUT was null pointer.
E_VSP_PARA_LUT_SIZE	-601	The *size* of T_VSP_LUT was out of range 1-256.
E_VSP_PARA_LUT_CONNECT	-602	The *connect* of T_VSP_LUT was out of specification.
E_VSP_PARA_CLU_MODE	-610	The *mode* of T_VSP_CLU was out of specification.
E_VSP_PARA_CLU_ADR	-611	The *clu_addr* of T_VSP_CLU was null pointer.
		When the *mode* was VSP_CLU_MODE_3D, the bit [23:16], [15:8] and [7:0] were other than 0-16, and the bit [31:24] was other than 0.
		When the *mode* was VSP_CLU_MODE_2D, the bit [23:16] and [15:8] were other than 0-16, and the bit [31:24] and [7:0] were other than 0.
E_VSP_PARA_CLU_DATA	-612	The *clu_data* of T_VSP_CLU was null pointer.
		When *mode* was VSP_CLU_MODE_3D, the bit [31:24] of *clu_data* was other than 0.
		When *mode* was VSP_CLU_MODE_2D, the bit [31:24] and bit [7:0] of *clu_data* were than 0.
E_VSP_PARA_CLU_SIZE	-613	When *mode* was VSP_CLU_MODE_3D, the *size* of T_VSP_CLU was out of range 1-4913.
		When *mode* was VSP_CLU_MODE_2D, the *size* of T_VSP_CLU was out of range 1-289.
E_VSP_PARA_CLU_CONNECT	-614	The *connect* of T_VSP_CLU was out of specification.

E_VSP_PARA_HST_NOTRGB	-630	Color space for input to the HST was not the RGB.
E_VSP_PARA_HST_CONNECT	-631	The *connect* of T_VSP_HST was out of specification.
E_VSP_PARA_HSI_NOTHSV	-640	Color space for input to the HIS was not the HSV.
E_VSP_PARA_HSI_CONNECT	-641	The *connect* of T_VSP_HSI was out of specification.
E_VSP_PARA_HGO_ADR	-660	The *addr* of T_VSP_HGO was null pointer.
E_VSP_PARA_HGO_WIDTH	-661	The *width* of T_VSP_HGO was out of 1-8190.
E_VSP_PARA_HGO_HEIGHT	-662	The *height* of T_VSP_HGO was out of 1-8190.
E_VSP_PARA_HGO_XOFFSET	-663	Caluculating value of the *width* + *x_offset* was greather than 8190.
E_VSP_PARA_HGO_YOFFSET	-664	Caluculating value of the *height* + *y_offset* was greather than 8190.
E_VSP_PARA_HGO_BINMODE	-665	The *binary_mode* of T_VSP_HGO was out of specification.
E_VSP_PARA_HGO_MAXRGB	-669	The *maxrgb_mode* of T_VSP_HGO was out of specification.
E_VSP_PARA_HGO_XSKIP	-666	The *x_skip* of T_VSP_HGO was out of specification.
E_VSP_PARA_HGO_YSKIP	-667	The *y_skip* of T_VSP_HGO was out of specification.
E_VSP_PARA_HGO_SMMPT	-668	The *sampling* of T_VSP_HGO was out of specification.
E_VSP_PARA_HGT_ADR	-670	The *addr* of T_VSP_HGT was null pointer.
E_VSP_PARA_HGT_WIDTH	-671	The *width* of T_VSP_HGT was out of range 1-8190.
E_VSP_PARA_HGT_HEIGHT	-672	The *height* of T_VSP_HGT was out of range 1-8190.
E_VSP_PARA_HGT_XOFFSET	-673	Caluculating value of the *width* + *x_offset* was greather than 8190.
E_VSP_PARA_HGT_YOFFSET	-674	Caluculating value of the *height* + *y_offset* was greather than 8190.
E_VSP_PARA_HGT_AREA	-675	The *area* of T_VSP_HGT was out of specification.
E_VSP_PARA_HGT_XSKIP	-676	The *x_skip* of T_VSP_HGT was out of specification.
E_VSP_PARA_HGT_YSKIP	-677	The *y_skip* of T_VSP_HGT was out of specification.
E_VSP_PARA_HGT_SMMPT	-678	The *sampling* of T_VSP_HGT was out of specification.
E_VSP_PARA_NOSRU	-650	The *sru* of T_VSP_CTRL was null pointer.
E_VSP_PARA_NOUDS	-651	The *uds* of T_VSP_CTRL was null pointer.
E_VSP_PARA_NOLUT	-652	The *lut* of T_VSP_CTRL was null pointer.
E_VSP_PARA_NOCLU	-653	The *clu* of T_VSP_CTRL was null pointer.
E_VSP_PARA_NOHST	-654	The *hst* of T_VSP_CTRL was null pointer.
E_VSP_PARA_NOHSI	-655	The *his* of T_VSP_CTRL was null pointer.
E_VSP_PARA_NOBRU	-656	The *bru* of T_VSP_CTRL was null pointer.
E_VSP_PARA_NOHGO	-657	The *hgo* of T_VSP_CTRL was null pointer.
E_VSP_PARA_NOHGT	-658	The *hgt* of T_VSP_CTRL was null pointer.
E_VSP_PARA_BRU_INCOLOR	-660	Image format for input to the BRU were not unified.
E_VSP_PARA_UDS_SERIAL	-661	The UDS were connected in series.

7. 2DDMAC driver parameters

7.1. T_TDDMAC_MODE

The following is described about the member of T_TDDMAC_MODE structure.

```
Typedef struct{
    unsigned char          renewal;
    unsigned char          resource;
    T_TDDMAC_EXTEND        *p_extend ;
} T_TDDMAC_MODE;
```

Member	Direction	Contents
unsigned char *renewal*	Input	DMA configuration renewal adoption. TDDMAC_RNEW_NORMAL: Normal mode If you specify a value other than the above, VSP manager returns a parameter error.
Unsigned char *resource*	Input	DMA request source adoption. TDDMAC_RES_AUTO : Auto request mode If you specify a value other than the above, VSP manager returns a parameter error.
T_TDDMAC_EXTEND* *p_extend*	Input	Not used. You must set NULL pointer.

7.2. T_TDDMAC_REQUEST

The following is described about the member of T_TDDMAC_REQUEST structure.

```
Typedef struct{
    void            *src_adr;
    unsigned short  src_stride;
    unsigned short  src_x_offset;
    unsigned short  src_y_offset;
    unsigned char   src_format;
    unsigned char   ratio;
    void            *dst_adr;
    unsigned char   alpha_ena;
    unsigned char   alpha;
    unsigned char   dst_format;
    unsigned short  dst_stride;
    unsigned short  dst_x_offset;
    unsigned short  dst_y_offset;
    unsigned short  dst_width;
    unsigned short  dst_height;
    void            *cb_finished;
    void            *userdata;
    unsigned long   swap;
    unsigned char   mirror;
    unsigned char   rotation;
} T_TDDMAC_REQUEST;
```

Member	Direction	Contents
void* *src_adr*	Input	Pointer to a source buffer address. Must specity the physical top address of consecutive buffer.
Unsigned short *src_stride*	Input	Source image horizontal byte size. (1 to 65535) Specify the one-line width of the source image in units of bytes. This value should be a multiple of the pack size. (Ex: If RGB565, set of a multiple of 2) Note: If use rotation or reversal, must adjust image stride with multiple of 16.
Unsigned short *src_x_offset*	Input	Source image clipping horizontal offset. [pixel]
unsigned short *src_y_offset*	Input	Source image clipping vertical offset. [line] Note: If source image format is YUV420, set multiple of 2.

Unsigned char *src_format*	Input	Source image format TDDMAC_FORMAT_Y : Y format TDDMAC_FORMAT_C420 : CbCr format(YcbCr4:2:0) TDDMAC_FORMAT_C422 : CbCr format(YcbCr4:2:2) TDDMAC_FORMAT_ARGB8888 : ARGB8888 TDDMAC_FORMAT_RGBA8888 : RGBA8888 TDDMAC_FORMAT_RGB888 : RGB888 TDDMAC_FORMAT_RGB565 : RGB565 TDDMAC_FORMAT_RGB332 : RGB332 TDDMAC_FORMAT_pRGB14_666 : pRGB14-666 TDDMAC_FORMAT_pRGB4_444 : pRGB4-444 TDDMAC_FORMAT_RGB666 : RGB666 TDDMAC_FORMAT_BGR666 : BGR666 TDDMAC_FORMAT_BGR888 : BGR888 TDDMAC_FORMAT_ABGR8888 : ABGR8888 TDDMAC_FORMAT_RGB0565 : RGB0565 Note: About detail of image format, refer to “7.3 format”.
Unsigned char *ratio*	Input	Method of scale-up TDDMAC_RATIO_1_1 : no scale-up TDDMAC_X_RATIO_2_1 : double up horizontal TDDMAC_Y_RATIO_2_1 : double up vertical TDDMAC_XY_RATIO_2_1 : double up horizontal and vertical Note) Can not set reversal and rotation at a time.If you would like to use scale-up, please set TDDMAC_MRR_OFF to *mirror* and TDDMAC_ROT_OFF to *rotation*.
Void* *dst_adr*	Input	Pointer to a destination buffer address. Must specity the physical top address of consecutive buffer. Note: Set a multiple of 16.
Unsigned char *alpha_ena*	Input	Select of alpha value. TDDMAC_SRCALPHA_DISABLE : Use alpha value of *alpha* parameter (Disable souce image alpha value) TDDMAC_SRCAPLHA_ENABLE : Use alpha value of souce image Note: If source image doesn’t have alpha value, Be equal TDDMAC_SRCALPHA_DISABLE.
Unsigned char *alpha*	Input	Set distination alpha value. (0 to 255) If you set TDDMAC_SRCALPHA_DISABLE to *alpha_ena*, or source image doesn’t have alpha value, this parameter value is used.

unsigned char *dst_format*	Input	Destination image format setting TDDMAC_FORMAT_Y : Y format TDDMAC_FORMAT_C420 : CbCr format(YcbCr4:2:0) TDDMAC_FORMAT_C422 : CbCr format(YcbCr4:2:2) TDDMAC_FORMAT_ARGB8888 : ARGB8888 TDDMAC_FORMAT_RGBA8888 : RGBA8888 TDDMAC_FORMAT_RGB888 : RGB888 TDDMAC_FORMAT_RGB565 : RGB565 TDDMAC_FORMAT_RGB332 : RGB332 TDDMAC_FORMAT_pRGB14_666 : pRGB14-666 TDDMAC_FORMAT_pRGB4_444 : pRGB4-444 TDDMAC_FORMAT_RGB666 : RGB666 TDDMAC_FORMAT_BGR666 : BGR666 TDDMAC_FORMAT_BGR888 : BGR888 TDDMAC_FORMAT_ABGR8888 : ABGR8888 TDDMAC_FORMAT_RGB0565 : RGB0565 Note1: The image format can not convert different type. Example: When source format is RGB. Destination format must be RGB. Note2: About detail of image format, refer to “7.3 format”.
Unsigned short *dst_stride*	Input	Destination image horizontal byte size. (16 to 65520) Specify the one-line width of the Destination image in units of bytes. Note: Set multiple of 16.
Unsigned short *dst_width*	Input	Destination image horizontal pixel size. (1 to 65535) Specify the horizontal pixel size of the destionation image. Note1: When use scale-up, set a value before magnification. Note2: When destination image format is YUV422 or YUV420, set a multiple of 2. Note3: When specify following condition, set of multiple a 16. Condition: 90 degree rotation, 180 degree rotation, horizontal reversal, horizontal and vertical reversal, 90 degree rotation and vertical reversal, 180 degree rotation and vertical reversal, 270 degree rotation and horizontal reversal, 270 degree rotation and horizontal and vertical reversal.
Unsigned short *dst_height*	Input	Destination image vertical pixel size. (1 to 65535) Specify the vertical pixel size of the destionation image. Note1: When use scale-up, set a value before magnification. Note2: When destination image format is YUV420, set a multiple of 2. Note3: When change format YUV420 to YUV422 and use reversal, set a multiple of 2.
Unsigned short *dst_x_offset*	Input	Destination image horizontal offset. [pixel] Note: Set to be a multiple of 16 after calculating from offset. (Example: If RGB565, you can specify 0, 8 and 16. And can not specify 1 to 7, 8 to 15.)
unsigned short *dst_y_offset*	Input	Destination image vertical offset. [line] Note: If source image format is YUV420, set multiple of 2.
Void* *cb_finished*	-	VSP manager is using this member, so you set NULL pointer.
Void* userdata	-	VSP manager is using this member, so you set NULL pointer.

Unsigned long swap	Input	Select of input and ouput swap Swapping is able to specify with logical disjunction combination. TDDMAC_SWAP_OFF : no swap TDDMAC_SWAP_OLS : long word swap of destination TDDMAC_SWAP_OWS : word swap of destination TDDMAC_SWAP_OBS : byte swap of destination TDDMAC_SWAP_ILS : long word swap of source TDDMAC_SWAP_IWS : word swap of source TDDMAC_SWAP_IBS : byte swap of source
unsigned char mirror	Input	Method of reversal TDDMAC_MRR_OFF : no reversal TDDMAC_MRR_H : horizontal reversal TDDMAC_MRR_V : vertical reversal TDDMAC_MRR_HV : horizontal and vertical revesal (= 180 degree rotation) Note: Can not set reversal and scale-up at a time. If you would like to use revesal, please set TDDMAC_RATIO_1_1 to *ratio*.
Unsigned char rotation	Input	Method of rotation TDDMAC_ROT_OFF : no rotation TDDMAC_ROT_90 : 90 degree rotation TDDMAC_ROT_180 : 180 degree rotation TDDMAC_ROT_270 : 270 degree rotation Note) Can not set rotation and scale-up at a time. If you would like to use rotation, please set TDDMAC_RATIO_1_1 to *ratio*.

Figure 7-1 shows a source and destination image association chart.

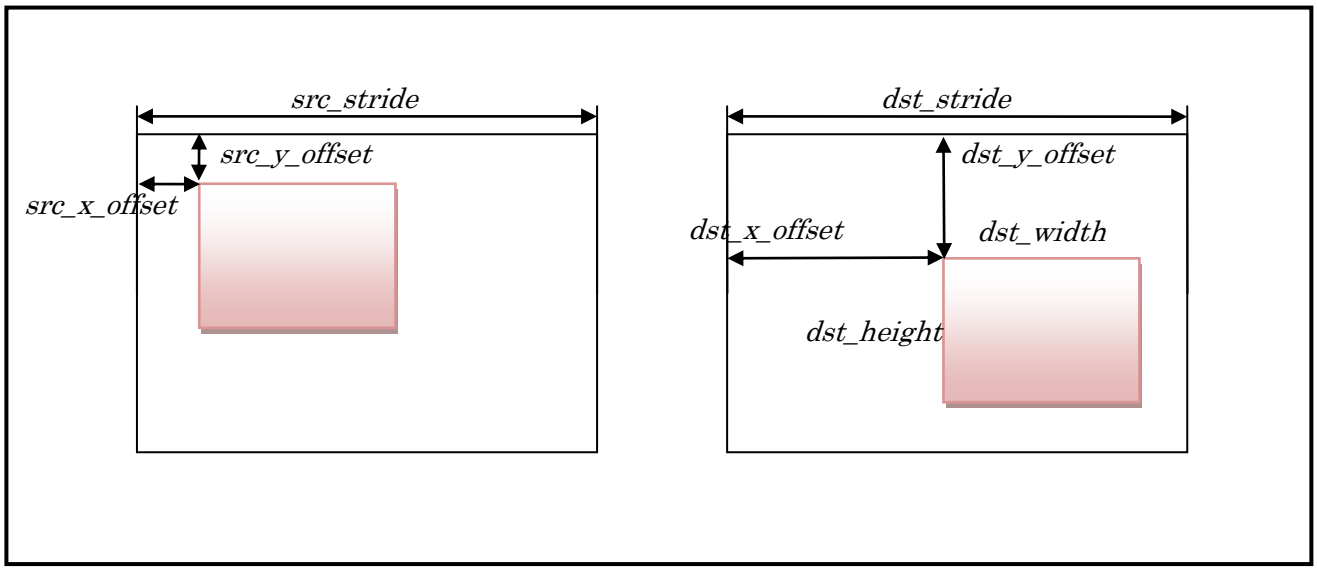


Figure 7-1 Source and destination image association chart

7.3. Format

7.3.1. Y format

Figure 7-2 shows Y format.

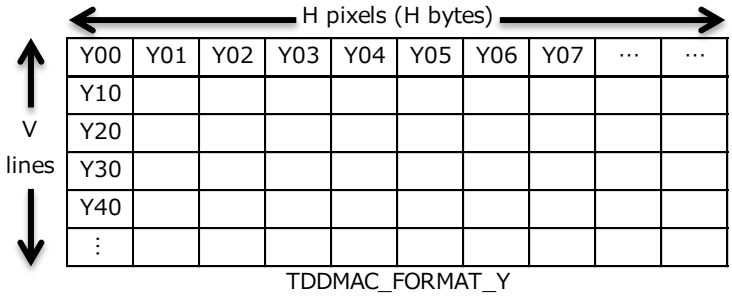


Figure 7-2 Y format

7.3.2. CbCr format

Figure 7-3 shows CbCr format. The CbCr format is supported semi planar only. The 2DDMAC can convert same array (Ex: NV16 to NV12). If you would like different array, adjust endian by *swap* parameter.

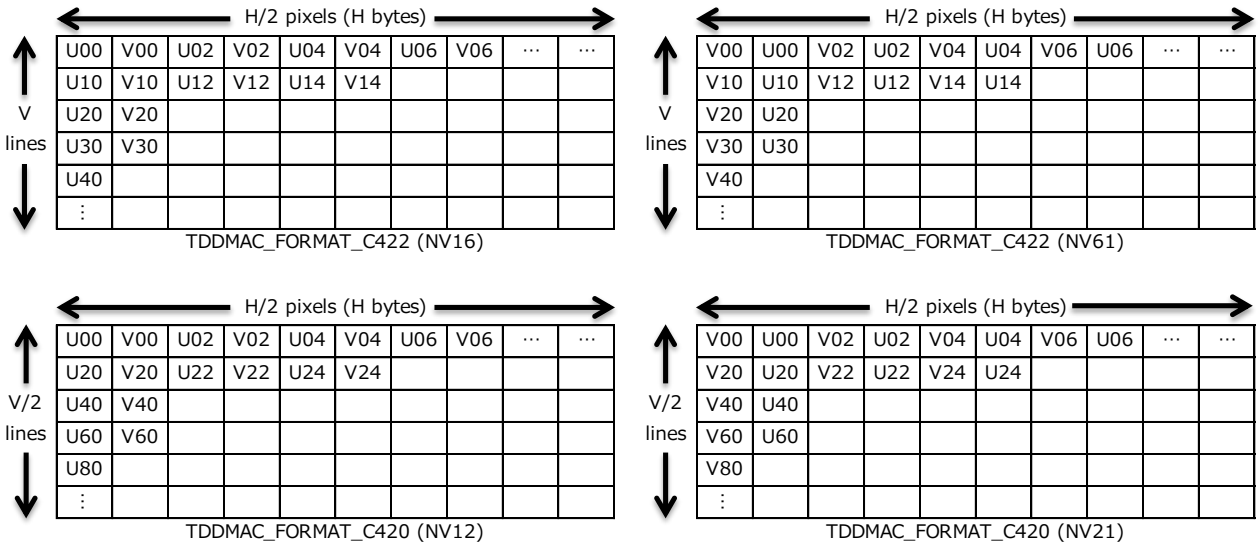


Figure 7-3 CbCr format

7.3.3. RGB format

Figure 7-4 shows RGB format. To specify stride and source offset, refer to following. Example, when format is ARGB8888, It uses 4 bytes per pixel. Therefore must set a multiple of 4 bytes.

format	byte	phase	bit																													
			31 to 24								23 to 16								15 to 8								7 to 0							
TDDMAC_FORMAT_ARGB8888	4		a	a	a	a	a	a	a	a	R0	R0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	B0	B0
TDDMAC_FORMAT_RGBA8888	4		R0	R0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	B0	B0	a	a	a	a	a	a	a	a
TDDMAC_FORMAT_RGB888	3	0	R0	R0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	B0	B0	R1	R1	R1	R1	R1	R1	R1	R1
		1	G1	G1	G1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1	B1	B1	B1	R2	R2	R2	R2	R2	R2	R2	R2	G2	G2	G2	G2	G2	G2	G2	G2
		2	B2	B2	B2	B2	B2	B2	B2	B2	R3	R3	R3	R3	R3	R3	R3	R3	G3	G3	G3	G3	G3	G3	G3	G3	B3	B3	B3	B3	B3	B3	B3	B3
TDDMAC_FORMAT_RGB565	2																		R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0
TDDMAC_FORMAT_RGB332	1																										R0	R0	R0	G0	G0	G0	B0	B0
TDDMAC_FORMAT_pRGB14_666	4		0	0	0	0	0	0	0	0	0	0	0	0	0	0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0
TDDMAC_FORMAT_pRGB4_444	2																		0	0	0	0	R0	R0	R0	R0	G0	G0	G0	G0	B0	B0	B0	B0
TDDMAC_FORMAT_RGB666	3	0	0	0	R0	R0	R0	R0	R0	R0	0	0	G0	G0	G0	G0	G0	G0	0	0	B0	B0	B0	B0	B0	B0	0	0	R1	R1	R1	R1	R1	R1
		1	0	0	G1	G1	G1	G1	G1	G1	0	0	B1	B1	B1	B1	B1	B1	0	0	R2	R2	R2	R2	R2	R2	0	0	G2	G2	G2	G2	G2	G2
		2	0	0	B2	B2	B2	B2	B2	B2	0	0	R3	R3	R3	R3	R3	R3	0	0	G3	G3	G3	G3	G3	G3	0	0	B3	B3	B3	B3	B3	B3
TDDMAC_FORMAT_BGR666	3	0	0	0	B0	B0	B0	B0	B0	B0	0	0	G0	G0	G0	G0	G0	G0	0	0	R0	R0	R0	R0	R0	R0	0	0	B1	B1	B1	B1	B1	B1
		1	0	0	G1	G1	G1	G1	G1	G1	0	0	R1	R1	R1	R1	R1	R1	0	0	B2	B2	B2	B2	B2	B2	0	0	G2	G2	G2	G2	G2	G2
		2	0	0	R2	R2	R2	R2	R2	R2	0	0	B3	B3	B3	B3	B3	B3	0	0	G3	G3	G3	G3	G3	G3	0	0	R3	R3	R3	R3	R3	R3
TDDMAC_FORMAT_BGR888	3	0	B0	B0	B0	B0	B0	B0	B0	B0	G0	G0	G0	G0	G0	G0	G0	G0	R0	R0	R0	R0	R0	R0	R0	R0	B1	B1	B1	B1	B1	B1	B1	B1
		1	G1	G1	G1	G1	G1	G1	G1	G1	R1	R1	R1	R1	R1	R1	R1	R1	B2	B2	B2	B2	B2	B2	B2	B2	G2	G2	G2	G2	G2	G2	G2	G2
		2	R2	R2	R2	R2	R2	R2	R2	R2	B3	B3	B3	B3	B3	B3	B3	B3	G3	G3	G3	G3	G3	G3	G3	G3	R3	R3	R3	R3	R3	R3	R3	R3
TDDMAC_FORMAT_ABGR8888	4		a	a	a	a	a	a	a	a	B0	B0	B0	B0	B0	B0	B0	B0	G0	G0	G0	G0	G0	G0	G0	G0	R0	R0	R0	R0	R0	R0	R0	R0
TDDMAC_FORMAT_RGB0565	4		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0

Figure 7-4 RGB format

7.4. Error code

Table 7-1 shows the detail error code of 2DDMAC. Accoding to error code, please check argument.

Table 7-1 Detail of error code

Define name	Error code	Contains
E_TDDMAC_PARA_RENEWAL	-111	An invalid parameter was specified to *renewal*.
E_TDDMAC_PARA_RESOURCE	-112	An invalid parameter was specified to *esource*.
E_TDDMAC_PARA_SRC_ADR	-120	A null pointer was specified to *src_adr*.
E_TDDMAC_PARA_DST_ADR	-121	The *src_adr* was null pointer.
		The *src_adr* wasn't a multiple of 16.
E_TDDMAC_PARA_SRC_Y_OFFSET	-133	The *src_y_offset* wasn't a multiple of 2 when source image format was YUV420.
E_TDDMAC_PARA_SRC_FORMAT	-134	An invalid parameter was specified to *src_format*.
E_TDDMAC_PARA_RATIO	-135	The *ratio* wasn't TDDMAC_RATIO_1_1 when any of *mirror* or *rotation* was enabled.
		An invalid parameter was specified to *ratio* when both of *mirror* and *rotation* was disabled.
E_TDDMAC_PARA_ALPHA_ENA	-137	An invalid parameter was specified to *alpha_ena*.
E_TDDMAC_PARA_DST_FORMAT	-139	An invalid parameter was specified to *dst_format*.
		A souce and destination image fomat are different format. Ex: Source format is RGB, but destination format is Y.
E_TDDMAC_PARA_DST_WIDTH	-140	The *dst_width* was 0.
		The *dst_width* wasn't a multiple of 2 when destination format was YUV422 or YUV420.
		The *dst_width* wasn't a multiple of 16 when following conditions. Condition: 90 degree rotation, 180 degree rotation, horizontal reversal, horizontal and vertical reversal, 90 degree rotation and vertical reversal, 180 degree rotation and vertical reversal, 270 degree rotation and horizontal reversal, 270 degree rotation and horizontal and vertical reversal.
E_TDDMAC_PARA_DST_HEIGHT	-141	The *dst_height* was 0.
		The *dst_height* wasn't a multiple of 2 when souce format was YUV420 or destination format was YUV420.
E_TDDMAC_PARA_SRC_STRIDE	-143	The *src_stride* was 0.
		The *src_stride* wasn't a multiple of 16 when any of *mirror* or *rotation* was enabled.
		The *src_stride* wasn't a multiple of byte per pixel when both of *mirror* and *rotation* was disabled.
E_TDDMAC_PARA_DST_STRIDE	-144	The *dst_stride* was 0.
		The *dst_stride* wasn't a multiple of 16.
E_TDDMAC_PARA_DST_X_OFFSET	-146	Calculated address wasn't a multiple of 16.

E_TDDMAC_PARA_DST_Y_OFFSET	-147	The *dst_y_offset* wasn't a multiple of 2 when destination format was YUV420.
E_TDDMAC_PARA_MIRROR	-148	An invalid parameter was specified to *mirror*.
E_TDDMAC_PARA_ROTATION	-149	An invalid parameter was specified to *rotation*.

8. Restrictions and Notes

This section describes the restrictions on the use of this software.

8.1. VSP's Restrictions

- If use the BRU, all inputting color space must be same color space.
- The UDS can not connect serial. Parallel is ok.
- There are limited numbers of modules that have the VSP. If you use the same module, you must execute divided into multiple entries.
- You must specify different source and destination buffer area. If memory buffer is overlapping, but it's not guaranteed to work.
- Support a progressive only.
- HST module has calculation error. If you convert to the HSV from RGB, and convert to the RGB, It is not equal original.

8.2. 2DDMAC's Restrictions

- 2DDMAC can not scale-up at the time as rotation and reversal. 2DDMAC can execute rotation and reversal.
- If you specify rotation and inversion at a time, rotation processing takes precedence.
- 2DDMAC can not convert different format. (ex: RGB to Y)
- CbCr format is compatible with semi planar. Interleave or Planar is not supported.
- If you would like to transform YcbCr format, you must separate Y format and CbCr format. Can not entry at a time.
- When the calculating source address*1 not equal 16n and cutting out effective pixel bytes number*2 is 16 or less, line finality access equal address where line terminal address is rounded up to 16n+16byte. Address where souce image final pixel address*3 of cutting out object is rounded up to 16n+16.

Note1: calculating source address = *src_adr* + *src_stride* * *src_y_offset* + *src_x_offset* * (bpp / 8)
Note2: cutting out effective pixel bytes number = *dst_width* * (bpp / 8)
Note3: souce image final pixel address = *src_adr* + *src_stride* * (*src_y_offset* + *dst_height* – 1) + (*src_x_offset* + *dst_width*) * (bpp / 8)

- When 2DDMAC transforms YUV420 to YUV422, Request a double size memory. You must allocate double size memory stored buffer of destination.

REVISION HISTORY	VSP Manager User's Manual: Software

Rev.	Date	Description	
		Page	Summary
1.00	May. 2014	-	First Edition Issued.

VSP Manager User's Manual: Software

Publication Date: Rev.1.00 May. 2014

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 204, 205, AZIA Center, No.1233 Lujiazui Ring Rd., Pudong District, Shanghai 200120, China
Tel: +86-21-5877-1818, Fax: +86-21-6887-7858 / -7898

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.
11F., Samik Lavied' or Bldg., 720-2 Yeoksam-Dong, Kangnam-Ku, Seoul 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 1.3

VSP Manager
User's Manual: Software

Renesas Electronics Corporation