User's Manual

CONFIDENTIAL

VSP Manager for Linux

User's Manual: Software

R-Car H3/M3/M3N/E3 Series

Renesas Electronics
www.renesas.com

Rev.3.00 Dec. 2021

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 or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving 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, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. 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 shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended 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; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or 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 damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, 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 and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall 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 shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

Trademark

- Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.
- Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.
- Other company names and product names mentioned herein are registered trademarks or trademarks of their respective owners.
- Registered trademark and trademark symbols (® and ™) are omitted in this document

How to Use This Manual

- **[Readers]**
  This manual is intended for engineers who develop products which use the R-Car H3 / M3 / M3N / E3 processor.

- **[Purpose]**
  This manual is intended to give users an understanding of the functions of the R-Car H3 / M3 / M3N / E3 processor device driver and to serve as a reference for developing hardware and software for systems that use this driver.

- **[How to Read This Manual]**
  It is assumed that the readers of this manual have general knowledge in the fields of electrical
  — engineering, logic circuits, microcontrollers, and Linux.
  → Read this manual in the order of the CONTENTS.
  — To understand the functions of a multimedia processor for R-Car H3 / M3 / M3N / E3
  → See the R-Car H3 / M3 / M3N / E3 User's Manual.
  — To know the electrical specifications of the multimedia processor for R-Car H3 / M3 / M3N / E3
  → See the R-Car H3 / M3 / M3N / E3 Data Sheet.

- **[Conventions]**
  The following symbols are used in this manual.
  Data significance: Higher digits on the left and lower digits on the right
  Note: Footnote for item marked with Note in the text
  Caution: Information requiring particular attention
  Remark: Supplementary information
  Numeric representation: Binary ... ××××, 0b××××, or ××××B
  Decimal ... ××××
  Hexadecimal ... 0x×××× or ××××H
  Data type: Double word ... 64 bits
  Word … 32 bits
  Half word ... 16 bits
  Byte ... 8 bits

Table of Contents

1. Overview

This manual is designed to provide the user with an understanding the functions of this software to management VSP and FDP 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 FDP.

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 FDP resources so that more than one application can use VSP and FDP at the same time.

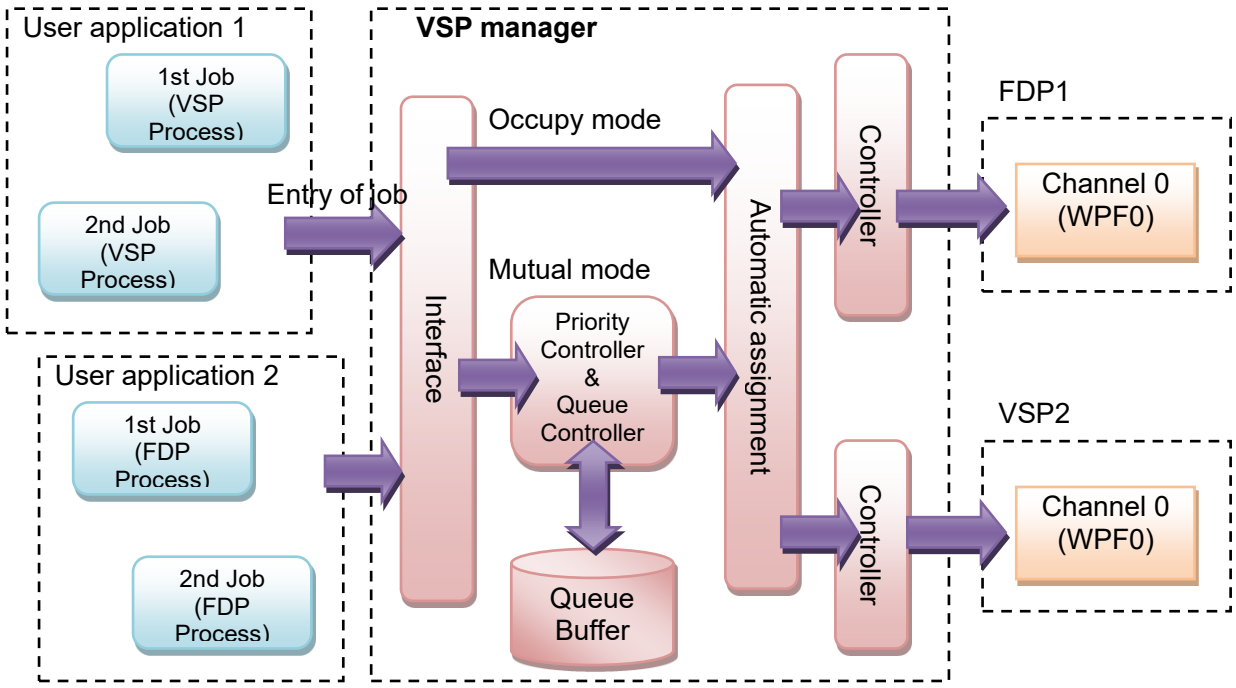


Figure 1-1 Overview of this software

The following is the functional overview of the VSP manager.

- Controls the VSP, FDP and FCP.
- Automatic assignment of free RPF channels.
- The VSP manager supports 2 modes. (Mutual mode or occupy mode).
- Mutual mode supports queue buffer control. 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. Queued jobs determines the device (VSPB or VSPI) by the module to be used, find available devices and process them in order. Occupied devices (device is set occupy mode) are not used in Mutual mode.
- Occupy mode supports low latency access. User application can low latency access because one application occupies one channel.

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 Operation (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.
- SHP
  The SHP is a module which executes the sharpen or un-sharpen the image.
- 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 H3 (VSPI)	R-Car H3 (VSPBD)	R-Car H3 (VSPBC)	R-Car M3/M3N/E3 (VSPI)	R-Car M3/M3N/E3 (VSPB)
RPF (CLUT)	1 (1)	5 (2)	5 (2)	1 (1)	5 (2)
SRU	1	0	0	1	0
UDS	1	0	0	1	0
LUT	1	0	1	1	1
CLU	1	0	1	1	1
HST	1	0	0	1	0
HSI	1	0	0	1	0
BRU	0	1	1	0	1
HGO	1	0	1	1	1
HGT	1	0	0	1	0
SHP	1	0	0	1	0

The following is the functional overview of the FDP.

- High image quality motion adaptive de-interlacing algorithm (basing on luma component only)
  (a) Combines the best aspects of both Bob (2D) and Weave (3D)
  (b) 2D and 3D comparisons are performed to decide whether or not an individual pixel has motion.
  (c) Diagonal interpolation is supported in 2D compensation.
- Support 3840 x 2160 resolution

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 and Figure 1-4 show the lists of these source files.

Table 1-2 Configuration of Document File

No	Name
1	R-Car H3/M3/M3N/E3 VSP Manager for Linux User's Manual (this document)

```
git://github.com/renesas-rcar/vspmif_lib.git
   |-- vspm_if-module
   |     |-- files
   |           |-- vspm_if
   |                 |-- if
   |                 |     |-- Makefile
   |                 |     |-- vspm_api.c
   |                 |     |-- vspm_api_fdp.c
   |                 |     |-- vspm_api_vsp.c
   |                 |
   |                 |-- include
   |                       |-- fdp_drv.h *1
   |                       |-- fdpm_api.h
   |                       |-- vsp_drv.h *1
   |                       |-- vspm_cmn.h
   |                       |-- vspm_public.h
   |
   |-- vspm_if-tp-user
   |     |-- files
   |           |-- vspm_if
   |                 |-- Makefile
   |                 |-- vspm_tp_lossy.c
   |                 |-- fdpm_tp.c
   |
   |-- MIT-COPYING
   |-- README

*1 Not included in this software. Please copy from kernel modules.
```

Figure 1-2 Configuration of this software

```
git://github.com/renesas-rcar/vspmif_drv.git
   |-- vspm_if-module
   |     |-- files
   |           |-- vspm_if
   |                |-- drv
   |                |     |-- Makefile
   |                |     |-- vspm_if_local.h
   |                |     |-- vspm_if_main.c
   |                |     |-- vspm_if_sub.c
   |                |
   |                |-- include
   |                      |-- vspm_if.h
   |
   |-- GPL-COPYING
   |-- MIT-COPYING
   |-- README
```

Figure 1-3 Configuration of interface for user land

```
git://github.com/renesas-rcar/vspm_drv.git
    |-- vspm-module
    |       |-- files
    |               |-- vspm
    |                   |-- drv
    |                   |     |-- manager
    |                   |     |     |-- vspm_common.h
    |                   |     |     |-- vspm_control.c
    |                   |     |     |-- vspm_drv_fdp.c
    |                   |     |     |-- vspm_drv_vsp.c
    |                   |     |     |-- vspm_exec_manager.c
    |                   |     |     |-- vspm_job_manager.c
    |                   |     |     |-- vspm_lib.c
    |                   |     |     |-- vspm_sort_queue.c
    |                   |     |     |-- vspm_task.c
    |                   |     |     |-- vspm_task_private.h
    |                   |     |
    |                   |     |-- fdp
    |                   |     |     |-- fdp_drv.c
    |                   |     |     |-- fdp_drv_hw.h
    |                   |     |     |-- fdp_drv_l.c
    |                   |     |     |-- fdp_drv_local.h
    |                   |     |     |-- fdp_drv_tbl.h
    |                   |     |
    |                   |     |-- vsp
    |                   |     |     |-- vsp_drv.c
    |                   |     |     |-- vsp_drv_local.h
    |                   |     |     |-- vsp_drv_par.c
    |                   |     |     |-- vsp_drv_phy.c
    |                   |     |
    |                   |     |-- fdp_drv_public.h
    |                   |     |-- frame.c
    |                   |     |-- frame.h
    |                   |     |-- Makefile
    |                   |     |-- vsp_drv_public.h
    |                   |     |-- vspm_ip_ctrl.h
    |                   |     |-- vspm_lib_public.h
    |                   |     |-- vspm_log.h
    |                   |     |-- vspm_main.c
    |                   |     |-- vspm_main.h
    |                   |     |-- vspm_sub.c
    |                   |
    |                   |-- include
    |                         |-- fdp_drv.h
    |                         |-- vsp_drv.h
    |                         |-- vspm_cmn.h
    |                         |-- vspm_public.h
    |
    |-- GPL-COPYING
    |-- MIT-COPYING
    |-- README
```

Figure 1-4 Configuration of the VSP manager

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	R-CarH3-SiP System Evaluation Board (Salvator-X) R-CarM3-SiP System Evaluation Board (Salvator-X) R-CarH3-SiP/M3-SiP/M3N-SiP System Evaluation Board (Salvator-XS) R-CarE3 System Evaluation Board (Ebisu) R-CarE3 System Evaluation Board (Ebisu-4D)	-
Device	R-Car H3 / M3 / M3N / E3	-
Using IP	VSP2, FDP, FCP	-

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 H3/M3/M3N/E3 Linux BSP	-	-
Memory manager	-	Use legacy interface and sample code.

1.4. Related Document

The related document to this module is as follows.

Table 1.1 Related document (R-Car H3 / M3 / M3N / E3)

Number	Issue	Title	Edition	Date
-	Renesas Electronics	R-Car Series, 3rd Generation User's Manual: Hardware	Rev.2.20	Jun. 30, 2020
-	Renesas Electronics	R-CarH3-SiP System Evaluation Board Salvator-X Hardware Manual RTP0RC7795SIPB0011S	Rev.1.09	May. 11, 2017
-	Renesas Electronics	R-CarM3-SiP System Evaluation Board Salvator-X Hardware Manual RTP0RC7796SIPB0011S	Rev.0.04	Oct. 3, 2016
-	Renesas Electronics	R-CarH3-SiP/M3-SiP/M3N-SiP System Evaluation Board Salvator-XS Hardware Manual	Rev.2.04	Jul. 17, 2018
-	Renesas Electronics	R-CarE3 System Evaluation Board Ebisu Hardware Manual RTP0RC77990SEB0010S	Rev.0.03	Apr. 11, 2018
-	Renesas Electronics	R-CarE3 System Evaluation Board Ebisu-4D (E3 board 4xDRAM) Hardware Manual	Rev.1.01	Jul. 19, 2018

1.5. Notice

- When using HGO refer to section 32.3.7.8 of the H/W manual.

2. Terminology

Table 2.1 Terminology

Abbreviation	Full Form
VSP	Video Signal Processor
FDP	Fine Display Processor
FCP	Frame Compression Processor
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
SHP	Sharpness
ROP	Raster OPration

3. Installation Procedures

3.1. 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.
```
$ source /opt/poky/1.8/environment-setup-aarch64-poky-linux
$ undef LDFLAGS
$ export KERNELSRC = (kernel source directory).
$ export INCSHARED = (include directory of SDK).
```

(2) Download source code of kernel modules
```
$ cd $WORK
$ git clone git://github.com/renesas-rcar/vspm_drv.git
$ git clone git://github.com/renesas-rcar/vspmif_drv.git
```

(3) Building of the VSP Manager
Execute “make” in the build directory.
```
$ cd vspm_drv/vspm-module/files/vspm/drv
$ make
```

(4) Copy header file to $(KERNELSRC)/include and $(INCSHARED).
Execute “make install” in the build directory. If you copy manually, this process is not required.
```
$ make install
```

(5) Verifying the VSP Manager module
Make sure that the following kernel module is built under “vspm_drv/vspm-module/files/vspm/drv”.
```
vspm.ko
```

(6) Building of the interface for user land.
Execute “make” in the build directory.
```
$ cd vspmif_drv/vspm_if-module/files/vspm_if/drv
$ make
```

(7) Copy header file to $(KERNELSRC)/include and $(INCSHARED).
Execute “make install” in the build directory. If you copy manually, this process is not required.
```
$ make install
```

(8) Verifying the interface for user land module.
Make sure that the following kernel module is built under “vspmif_drv/vspm_if-module/files/vspm_if/drv”.
```
vspm_if.ko
```

3.2. 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 3.1.

If you will use the legacy I/F, please set the following environment variables.
$ export VSPM_LEGACY_IF=0 : Disable
$ export VSPM_LEGACY_IF=1 : Enable

(2) Download source code of shared library
```
$ cd $WORK
$ git clone git://github.com/renesas-rcar/vspmif_lib.git
```

(3) Building
Execute “make” in the build directory
```
$ cd vspmif_lib/vspm_if-module/files/vspm_if/if
$ make
```

(4) Copy header file to $(INCSHARED).
Execute “make install” in the build directory. If you copy manually, this process is not required.
```
$ make install
```

(5) Verifying the binary module
Make sure that the following binary modules are built under “vspmif_lib/vspm_if-module/files/vspm_if/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.

3.3. 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 3.1 and 3.2.

(1) Storing the kernel modules
Copy ‘vspm.ko’ and ‘vspm_if.ko’ to BSP user land. Define $NFS is root directory on BSP.
```
$ sudo cp vspm.ko $NFS/home/root/workspace
$ sudo cp vspm_if.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 lager 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
```

3.4. 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 user's 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 vspmif_lib/vspm_if-tp-user/files/vspm_if
$ make
(3) Verifying the executing object
Make sure that the following executing objects are built under "vspmif_lib/vspm_if-tp-user/files/vspm_if".
vspm_tp
fdpm_tp
(4) Executing on lager board.
Copy 'vspm_tp' and 'fdpm_tp' to BSP user land. Executing and enjoying.
$ ./vspm_tp
$ ./fdpm_tp

3.5. Device tree configuration

The following is the procedure for recognize the VSP and FDP.

(1) common

name	contents	offset
reg	Physical base address and size of the 2 registers area.	0: VSP or FDP register 1: FCP register
compatible	Should contain one of: Note: Use this when renesas VSP is used. * "renesas,vspm" Note: Use this when renesas FDP is used. * "renesas,fdpm"	
interrupts	Interrupt specifier. Refer to interrupt bindings.	
clocks	Input 2 clocks specifier. Refer to common clock bindings.	0: VSP or FDP module clock 1: FCP module clock
clock-names	Input 2 clocks name. Refer to below. Note: Use this when renesas VSP is used. * “vsp”, “fcp” Note: Use this when renesas FDP is used. * “fdp”, “fcp”	
power-domains	Power domains specifier. Refer to power domains bindings.	

(2) VSP

name	contents	value
renesas,#ch = <num>	Assignment channel of VSP manager.	0 to 4
renesas,#rpf = <bit>	Specify a valid RPF bits.	Set decimal. example: valid ch0 to ch4. Set bit = 31.
renesas,#rpf_clut = <bit>	Specify a valid clut of RPF bits.	Set decimal.
renesas,#wpf_rot = <bit>	Specify a valid rotation of WPF bits.	Set decimal.
renesas,has-sru	SRU on board	
renesas,has-uds	UDS on board	
renesas,has-lut	LUT on board	
renesas,has-clu	CLU on board	
renesas,has-hst	HST on board	
renesas,has-hsi	HSI on board	
renesas,has-hgo	HGO on board	
renesas,has-hgt	HGT on board	
renesas,has-bru	BRU on board	
renesas,has-shp	SHP on board	
renesas,#read_outstanding = <num>	Specify read outstanding value.	0: VSPB series 2: VSPI series
renesas,#start_reservation = <num>	Specify start reservation mode.	0: Not used. 1: Use start reservation with double buffer of histogram. 2: Use start reservation with H/W transfer of histogram.
renesas,#burst_access = <num>	Specify burst access size of RPF.	0: 256 pixels. 1: 512 pixels.

(3) FDP

name	contents	value
renesas,#ch = <num>	Assignment channel	0 to 2
renesas,#lut_table_index = <num>	Select LUT table index. “lut_table_index = 0” which keeps the balance ratio of 2D and 3D interpolated images on the output image, is recommended as default setting. When it’s needed to strengthen the 3D interpolated image, use 1 or 2 setting.	0 to 2

Example: Registries VSPI0 to 0 channel of VSP manager.

```
vspi0: vspm@fe9a0000 {
        compatible = “renesas,vspm”;
        reg = <0 0xfe9a0000 0 0x8000>, <0 0xfe9af000 0 0x200>;
        interrupts = <GIC_SPI 444 IRQ_TYPE_LEVEL_HIGH>;
        clocks = <&cpg CPG_MOD 631>, <&cpg CPG_MOD 611>;
        clock-names = “vsp”, “fcp”;
        power-domains = <&pd_a3vp>;
        renesas,#ch = <0>;
        renesas,#rpf = <1>;
        renesas,#rpf_clut = <1>;
        renesas,#wpf_rot = <1>;
        renesas,has-sru;
        renesas,has-uds;
        renesas,has-lut;
        renesas,has-clu;
        renesas,has-hst;
        renesas,has-hsi;
        renesas,has-hgo;
        renesas,has-hgt;
        renesas,has-shp;
        renesas,#read_outstanding = <2>;
        renesas,#start_reservation = <0>;
};
```

Default channel assignment

channel	R-Car H3		R-Car M3 / M3N / E3			
	VSP	FDP	VSP	FDP		
0	VSPI0	FDP0	VSPI0	FDP0		
1	VSPI1	FDP1	-	-		
2	-	-	-	-		
3	VSPBD	-	VSPB	-		
4	VSPBC	-	-	-		

4. Processing Specifications

4.1. Module Configuration

Figure 4-1 shows the module configuration of this software.

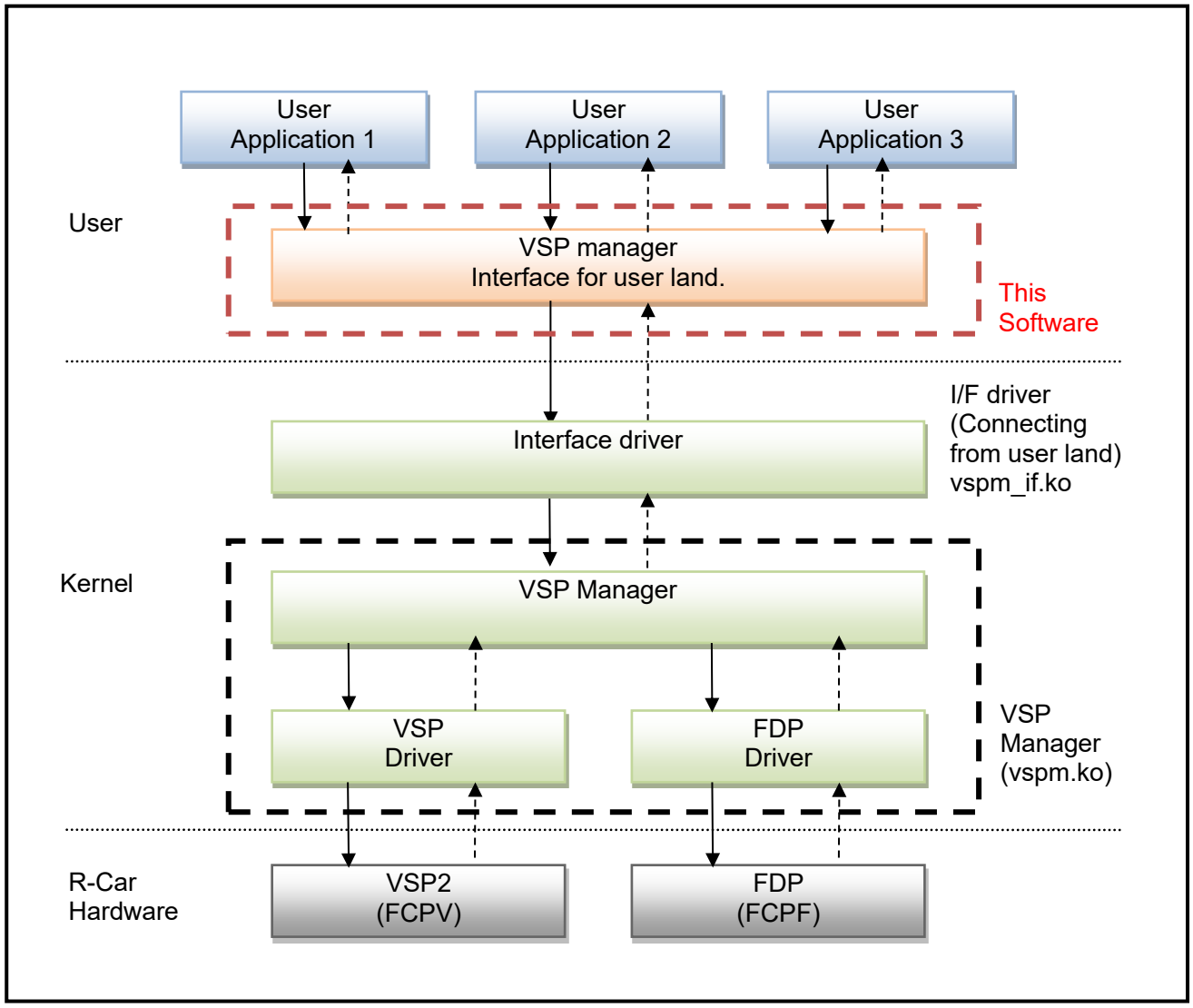


Figure 4-1 Configuration of Module

4.2. Processing Procedure

Figure 4-2 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.

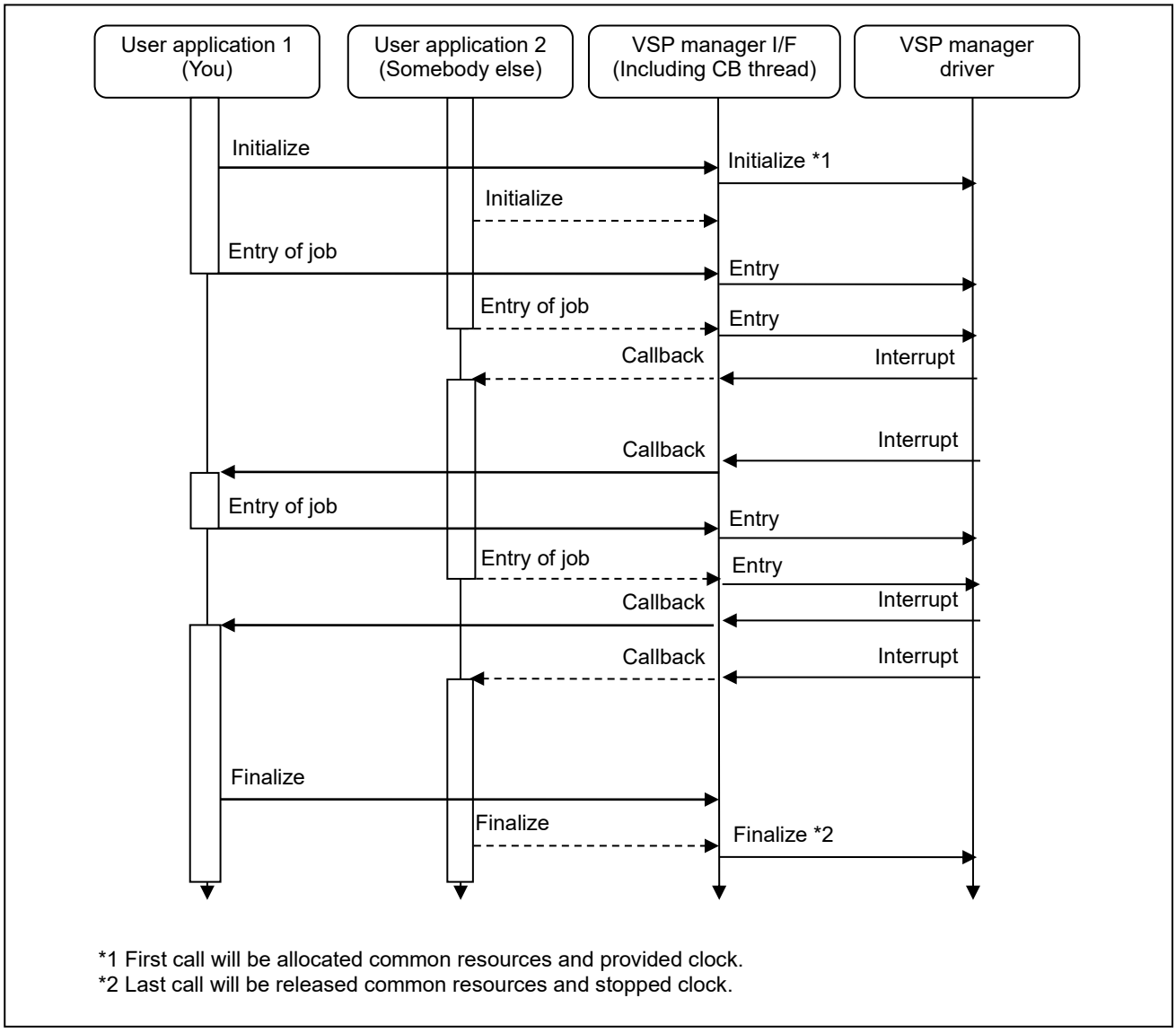


Figure 4-2 Basic Processing Procedure

Figure 4-3 shows VSP manager I/F more detailedly than Figure 4-2.
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.

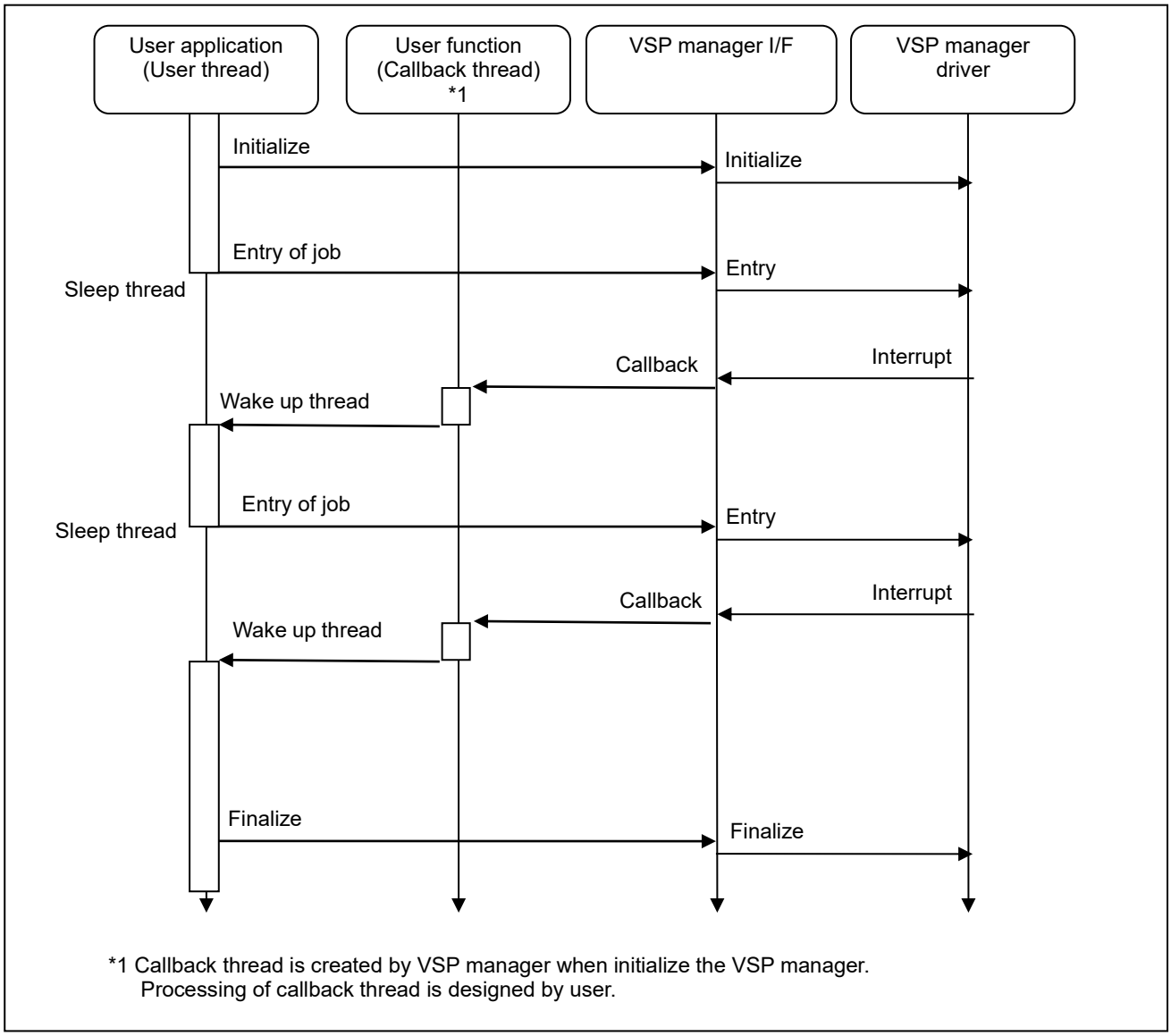


Figure 4-3 Callback Processing Procedure

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

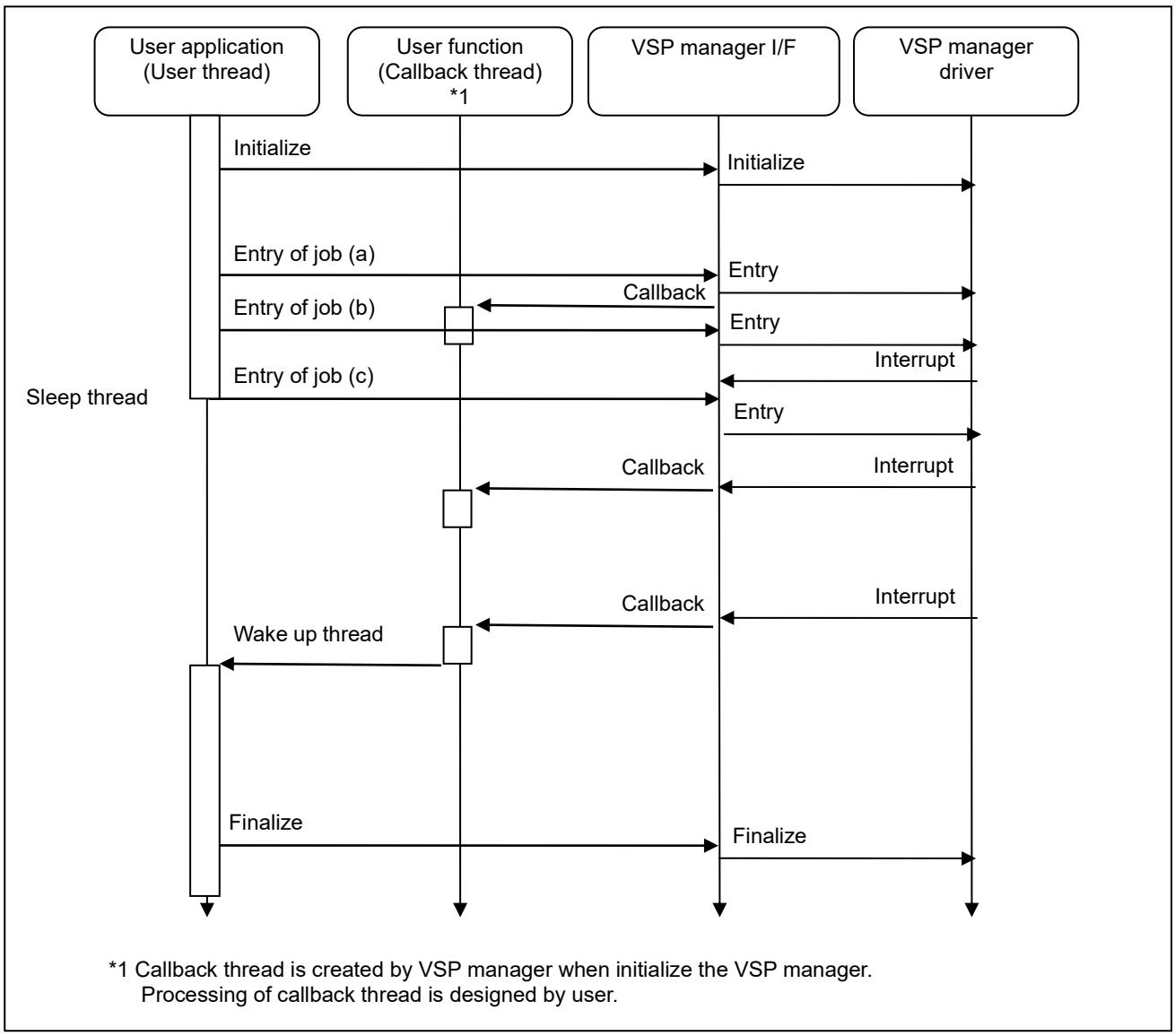


Figure 4-4 Continuous Processing Procedure

4.3. Timing chart

Figure 4-5 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.

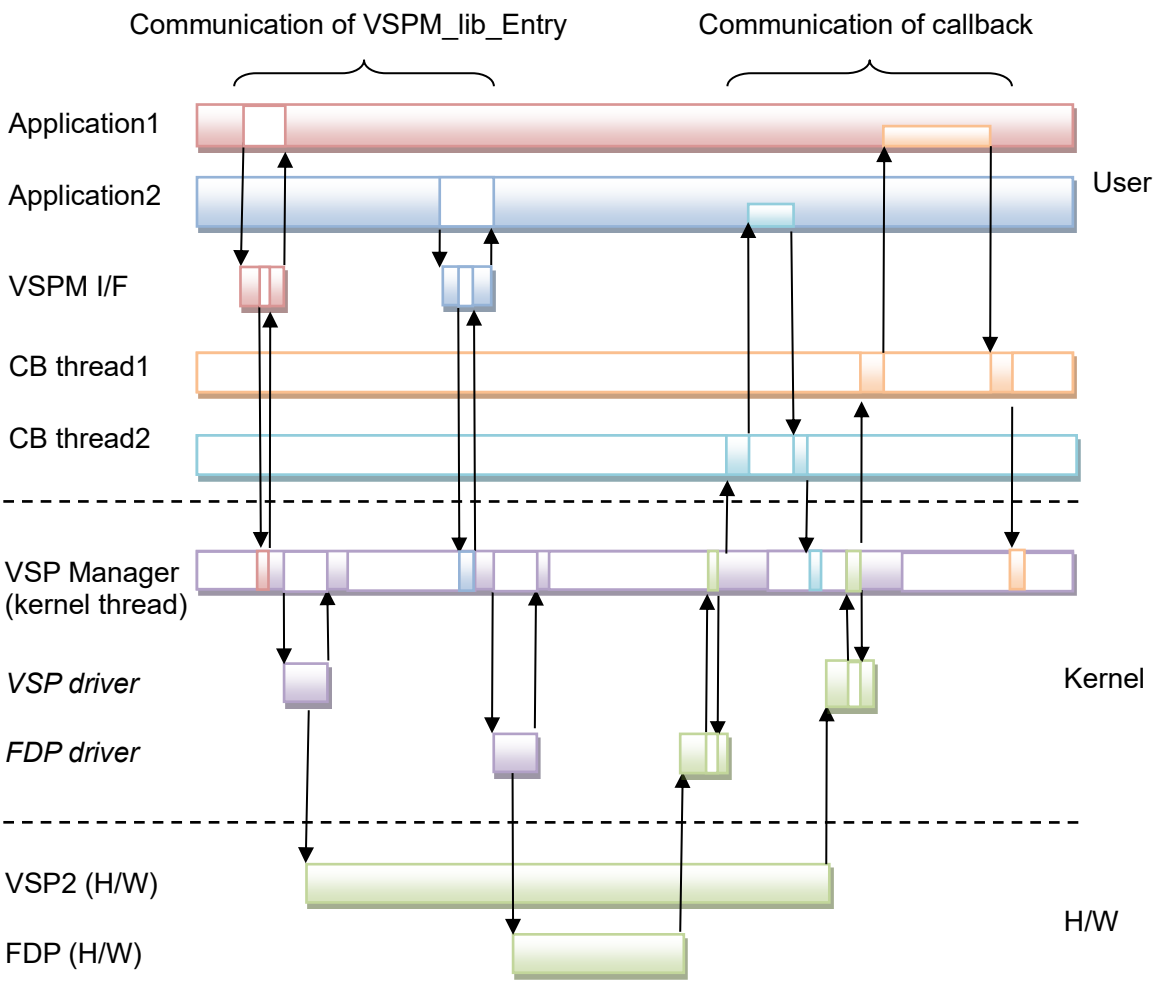


Figure 4-5 Timing chart (Until callback from job entry)

4.4. Control jobs

Registration to the queue of job is carried out by executing the vspm_job_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) Remove from the queue after processing complete.

5. List of API

Table 5-1 shows the list of API.

Table 5-1 List of API

No.	Name	Function
1	vspm_init_driver()	Initializing VSP manager
2	vspm_quit_driver()	Finalizing VSP manager
3	vspm_entry_job()	Entry of job.
4	vspm_cancel_job()	Cancel of job
5	vspm_get_status()	Get status
6	PFN_VSPM_COMPLETE_CALLBACK()	Callback functions of finished processing.

5.1. Initializing VSP manager

Name

vspm_init_driver -- Initializing VSP manager.

Synopsis

```
#include “vspm_public.h”
long vspm_init_driver (
        void **handle                          (output)
        struct vspm_init_t *param              (input)
)
```

Arguments

void ***handle*:	Pointer to a handle
struct vspm_init_t **param:*	Pointer to an initialize parameter.

Struct

```
struct vspm_init_t {
    unsigned int use_ch;
    unsigned short mode;
    unsigned short type;
    union {
        struct vspm_init_vsp_t *vsp;
        struct vspm_init_fdp_t *fdp;
    } par;
};
```

unsigned int *use_ch:* Using channel.

VSPM_EMPTY_CH:	use empty channel. The VSP manager searches empty channel.
VSPM_USE_CH0:	use channel 0.
VSPM_USE_CH1:	use channel 1.
VSPM_USE_CH2:	use channel 2.
VSPM_USE_CH3:	use channel 3.
VSPM_USE_CH4:	use channel 4.

unsigned short *mode*: Processing mode.

VSPM_MODE_MUTUAL:	Mutual mode. Share the IP.
VSPM_MODE_OCCUPY:	Occupy mode. Occupy the IP.

unsigned short *type*: Processing IP type.

VSPM_TYPE_VSP_AUTO:
Select VSP. Specify null pointer to *vsp*.

VSPM_TYPE_FDP_AUTO:
Select FDP. When use 3D-IPC mode, do not specify null pointer to *fdp*.

```
struct vspm_init_fdp_t {
    unsigned int hard_addr[2];
};
```

unsigned int *hard_addr[2]*:

Work buffer for still mask. When use 3D-IPC mode, Specify work buffer 2 area ([0] is top field, [1] is bottom field). Calculate method of work buffer size per field. Horizontal and vertical size are maximum of processing.

(2 * ((horizontal size + 7) / 8) * vertical size)

Return value

R_VSPM_OK:	Successful.
R_VSPM_NG:	Failure.
R_VSPM_PARAERR:	Invalid parameter.
R_VSPM_ALREADY_USED:	Specify channel already used.

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_quit_driver (), it can not execute the VSP manager's functions.
- The handle of parameter used until executing vspm_quit_driver () by user calling this function.
- If this API returned other than R_VSPM_OK, please check hardware configuration, memory resource and etc.
- The default channel assignment refers to section 2.5. When it is changed by the environment, check the configuration of the device tree (DT).

See Also

vspm_quit_driver ()

5.2. Finalizing VSP manager

Name

vspm_quit_driver -- Finalizing VSP manager.

Synopsis

```
#include “vspm_public.h”
long vspm_quit_driver (
        void *handle                                    (input)
)
```

Arguments

void **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_init_driver () and vspm_quit_driver () 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_init_driver ()

5.3. Entry of job

Name

vspm_entry_job -- Entry of job.

Synopsis

```
#include “vspm_public.h”
long vspm_entry_job (
        void *handle,                                                   (input)
        unsigned long *job_id,                                          (output)
        char job_priority,                                              (input)
        struct vspm_job_t *ip_param,                                    (input)
        void *user_data,                                                (input)
        PFN_VSPM_COMPLETE_CALLBACK cb_func                              (input)
)
```

Arguments

void *handle:	handle value.
unsigned long *job_id:	Pointer to a job ID.
char job_priority:	Priority of job. 1 (VSPM_PRI_MIN) to 126 (VSPM_PRI_MAX)
struct vspm_job_t *ip_param:	Pointer to a processing parameter.
void *user_data:	Data set by user.
PFN_VSPM_COMPLETE_CALLBACK cb_func:	Function pointer of callback function.

Struct

```
struct vspm_job_t {
        unsigned short type;
        union {
                struct vsp_start_t *vsp;
                struct fdp_start_t *fdp;
        } par;
};
```

unsigned short type:	Processing type. Specify same value of initialize parameter.
struct vsp_start_t *vsp:	This member is VSP driver’s parameter. Refer to section 7.1.
struct fdp_start_t *fdp:	This member is FDP driver’s parameter. Refer to section 8.1.

Return value

R_VSPM_OK:	Successful.
R_VSPM_NG:	Failure.
R_VSPM_PARAERR:	Invalid parameter.
R_VSPM_QUE_FULL:	Overflow queue.
Driver’s error:	When occupy mode, return driver’s error code.

Description

- This API requests image processing.
- Request unit is 1 channel. Also entry can not process VSP and FDP at a time.
- Be set to *par* the structure of the type specified in *type*.
- Process does not end at the time of the completion of the entry. Since the completion callback function that is set to *cb_func* 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 parameters. Job ID is invalid when occupy mode. It returns 0 every time.
- 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 priority. For the same priority is the FIFO. Priority is invalid when occupy mode.

Notes

- User's application can not execute from signal handler.
- The buffer of specified to the *ip_param* of argument should not release until processing finished.
- The *cb_func* of argument should not set null pointer.
- About detail of the vsp_start_t and fdp_start_t, refer to section 7 and section 8.
- If return value is other than R_VSPM_OK, the VSP manager is rejecting entry. Therefore you no need to cancel.

See Also

vspm_cancel_job ()

5.4. Cancel of job

Name

vspm_cancel_job -- Cancel of job.

Synopsis

```
#include “vspm_public.h”
long vspm_cancel_job (
        void *handle,                    (input)
        unsigned long job_id             (input)
)
```

Arguments

void **handle*:	handle value.
unsigned long *job_id*:	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_entry_job ()

5.5. Get status

Name

vspm_get_status -- Get status information.

Synopsis

```
#include “vspm_public.h”
long vspm_get_status (
        void *handle,                          (input)
        struct vspm_status_t *status           (output)
)
```

Arguments

void *handle:	handle value.
struct vspm_status_t *status:	Pointer to a status information.

Struct

```
struct vspm_status_t {
    struct fdp_status_t *fdp;    Pointer to a status parameter of FDP. Refer to section 8.3
};
```

Return value

R_VSPM_OK:	Successful.
R_VSPM_NG:	Failure.
R_VSPM_PARAERR:	Invalid parameter.

Description

- This API gets a status.

Notes

- This API can execute from receiving callback to entry next job. If you execute other condition, you can not get correct information.

5.6. 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 job_id,                   (output)
        long result,                            (output)
        void *user_data                         (output)
)
```

Arguments

unsigned long *job_id*: Job ID.
long *result*: 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 FDP driver.
Other:	Minor error of VSP and FDP driver.

void **user_data*: 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 *job_id* and *user_data* of argument are set by vspm_entry_job ().
- When the *result* is other than R_VSPM_OK, R_VSPM_NG, R_VSPM_CANCEL and R_VSPM_DRIVER_ERR, the *result* is set detail error code of VSP or FDP. In case of using VSP, refer to section 7.4. In case of using FDP, refer to section 8.6.

Notes

- User’s application must judge by this API. If *result* of argument is other than R_VSPM_OK, image processing is failure.
- Don’t call the VSP manager’s function within the callback context.
- When the vspm_job_entry () processing is delayed, in some case, before entry processing, completion callback is called.
- If the *result* 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_entry_job ()

6. VSP manager parameters

Table 6-1 Configuration parameter lists

Define Name	Value	Note
VSPM_TYPE_VSP_AUTO	0	Automation assignment channel of VSP
VSPM_TYPE_FDP_AUTO	1	Automation assignment channel of FDP
VSPM_MODE_MUTUAL	0	Mutual mode
VSPM_MODE_OCCUPY	1	Occupy mode
VSPM_PRI_MAX	126	Maximum priority
VSPM_PRI_MIN	1	Minimum priority
VSPM_PRI_LOW	32	Low priority
VSPM_PRI_STD	64	Standard priority
VSPM_PRI_HIGH	96	High priority
VSPM_STATUS_ACTIVE	2	
VSPM_STATUS_NO_ENTRY	3	
VSPM_EMPTY_CH	0xFFFFFFFF	Select all channels
VSPM_USE_CH0	0x00000001	Select channel 0
VSPM_USE_CH1	0x00000002	Select channel 1
VSPM_USE_CH2	0x00000004	Select channel 2
VSPM_USE_CH3	0x00000008	Select channel 3
VSPM_USE_CH4	0x00000010	Select channel 4

Table 6-2 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_ALREADY_USED	-6	Already used all channel
R_VSPM_OCCUPY_CH	-7	Occupy channel
R_VSPM_DRIVER_ERR	-10	Driver’s error

7. VSP driver parameters

7.1. vsp_start_t

The following is described about the member of vsp_start_t structure.

```
struct vsp_start_t {
    unsigned char       rpf_num;
    unsigned long       rpf_order;
    unsigned long       use_module;
    struct vsp_src_t    *src_par[5] ;
    struct vsp_dst_t    *dst_par ;
    struct vsp_ctrl_t   *ctrl_par ;
    struct vsp_dl_t     dl_par ;
};
```

Member	Direction	Contents
unsigned char *rpf_num*	Input	Input source number (0 to 5) If you set 0 to *rpf_num*, you must set virtual input on BRU. If you set 1 or more to *rpf_num*, you must set source configuration image structure.
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_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) : Histogram generator-one VSP_HGT_USE (0x0400) : Histogram generator-two VSP_SHP_USE (0x0800) : Sharpness
struct vsp_src_t **src_par[5]*	Input	Pointer to a source configuration image structure. If you set 1 or more to *rpf_num*, can't set null pointer to *src_par [0]*. If you set 2 or more to *rpf_num*, can't set null pointer to *src_par [1]*. If you set 3 or more to *rpf_num*, can't set null pointer to *src_par [2]*. If you set 4 or more to *rpf_num*, can't set null pointer to *src_par [3]*. If you set 5 to *rpf_num*, can't set null pointer to *src_par [4]*.
struct vsp_dst_t **dst_par*	Input	Pointer to a destination configuration image structure. Can not set null pointer to *dst_par*.
struct vsp_ctrl_t **ctrl_par*	Input	Pointer to a module configuration structure. Can not set null pointer to *ctrl_par*.

struct vsp_dl_t *dl_par*	Input	Work buffer parameter for display list. Specify 192 to *tbl_num* of vsp_dl_t member. If you use the following function, set according to the calculation method. Functions: Up-down scaler (UDS) Super-resolution (SRU) Rotation (except VSP_ROT_OFF and VSP_ROT_V_FLIP) Calculation method: Division number: div_num = ROUNDUP(destination horizontal size / 256) *tbl_num* = 192 + 64 * (div_num - 1)

Figure 7-1 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.

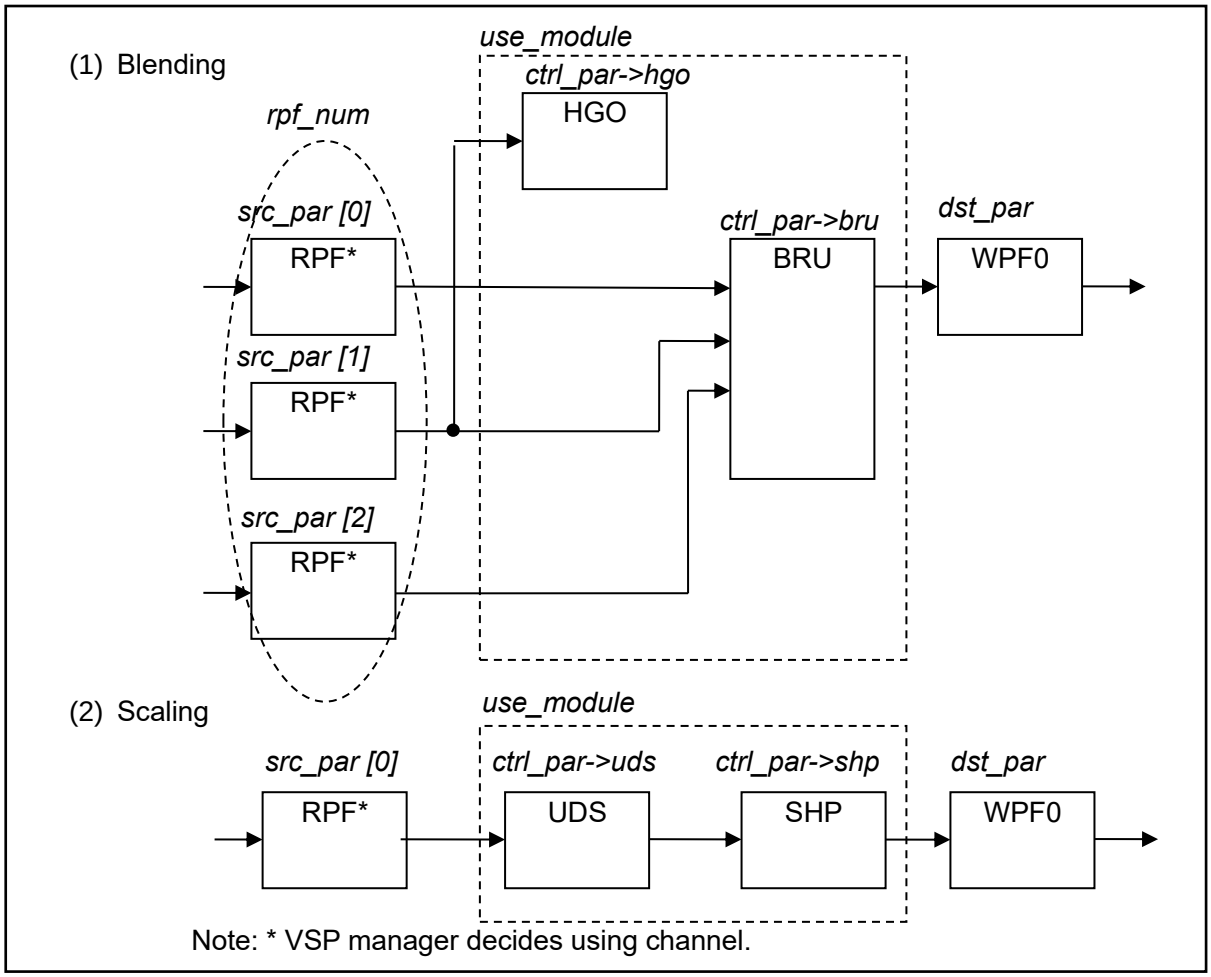


Figure 7-1 Basic module connection association chart

7.1.1. vsp_src_t

The following is described about the member of vsp_src_t structure.

```
struct vsp_src_t {
    unsigned int            addr;
    unsigned int            addr_c0;
    unsigned int            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 ;
    struct vsp_dl_t         *clut ;
    struct vsp_alpha_unit_t *alpha ;
    unsigned long           connect ;
};
```

Member	Direction	Contents
unsigned int *addr*	Input	Starting buffer address of Y or RGB. Specify continuous physical address.
unsigned int *addr_c0*	Input	Starting 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.
unsigned int *addr_c1*	Input	Starting 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 7-9 and Table 7-10. 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 7-9 and Table 7-10. 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. Note: If you use the following functions, set to 0. Up-down scaler (UDS) Super-resolution (SRU) Rotation (except VSP_ROT_OFF and VSP_ROT_V_FLIP)
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. Note: If you use the following functions, set to 0. Up-down scaler (UDS) Super-resolution (SRU) Rotation (except VSP_ROT_OFF and VSP_ROT_V_FLIP)
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 “7.3.1 Input format”. If you use a virtual input (*vir* = VSP_VIR), the following formats are available. VSP_IN_ARGB8888 (RGB) 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: If data array is big-endian, specify, (VSP_SWAP_B|VSP_SWAP_W|VSP_SWAP_L|VSP_SWAP_LL) to this member.
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.

<table>
<tr><td>unsigned short
y_position</td><td>Input</td><td>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.</td></tr>
<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 or 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 conversion (input format depth = output format depth)

<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.
<table>
<tr><td></td><td>MSB</td><td></td><td></td><td></td><td></td><td>LSB</td></tr>
<tr><td>RGB format</td><td>-</td><td>A (8bit)</td><td>R (8bit)</td><td>G (8bit)</td><td>B (8bit)</td><td></td></tr>
<tr><td></td><td>63 32</td><td>31</td><td></td><td></td><td>0</td><td></td></tr>
<tr><td></td><td>MSB</td><td></td><td></td><td></td><td></td><td>LSB</td></tr>
<tr><td>YUV format</td><td>-</td><td>A (8bit)</td><td>Cr (8bit)</td><td>Y (8bit)</td><td>Cb (8bit)</td><td></td></tr>
<tr><td></td><td>63 32</td><td>31</td><td></td><td></td><td>0</td><td></td></tr>
</table></td></tr>
<tr><td>struct vsp_dl_t *clut</td><td>Input</td><td>Pointer to a structure of 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. Refer to Table 7-1.

The setting range of tbl_num is 1 to 256. When the size specified fewer than 256, areas not setting does not guarantee.</td></tr>
<tr><td>struct vsp_alpha_unit_t *alpha</td><td>Input</td><td>Pointer to a structure of alpha blend setting
Can not specify null pointer.</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_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
VSP_SHP_USE (0x0800) : Sharpness</td></tr>
</table>

Table 7-1 storage method of CLUT.

table number	MSB (31)			LSB (0)
0	0			
	A (8bit)	R/Cr (8bit)	G/Y (8bit)	B/Cr (8bit)
1	0			
	A (8bit)	R/Cr (8bit)	G/Y (8bit)	B/Cr (8bit)
2	0			
	A (8bit)	R/Cr (8bit)	G/Y (8bit)	B/Cr (8bit)
…	…			
254	0			
	A (8bit)	R/Cr (8bit)	G/Y (8bit)	B/Cr (8bit)
255	0			
	A (8bit)	R/Cr (8bit)	G/Y (8bit)	B/Cr (8bit)

Note: When format is VSP_IN_RGB_CLUT_DATA, set to R, G and B.
When format is VSP_IN_YUV_CLUT_DATA, set to Cb, Y and Cr.

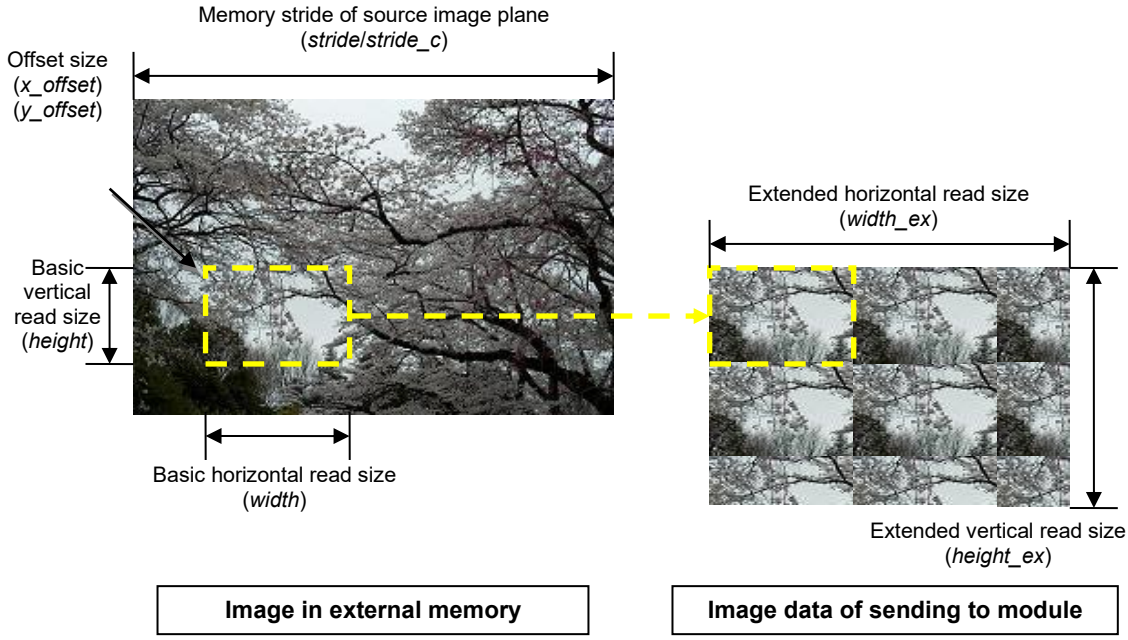


Figure 7-2 Extend reading size association chart

Figure 7-2 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.
If you use the following functions, extended reading is invalid.
Up-down scaler (UDS), super-resolution (SRU), rotation (except VSP_ROP_OFF and VSP_ROP_V_FLIP).

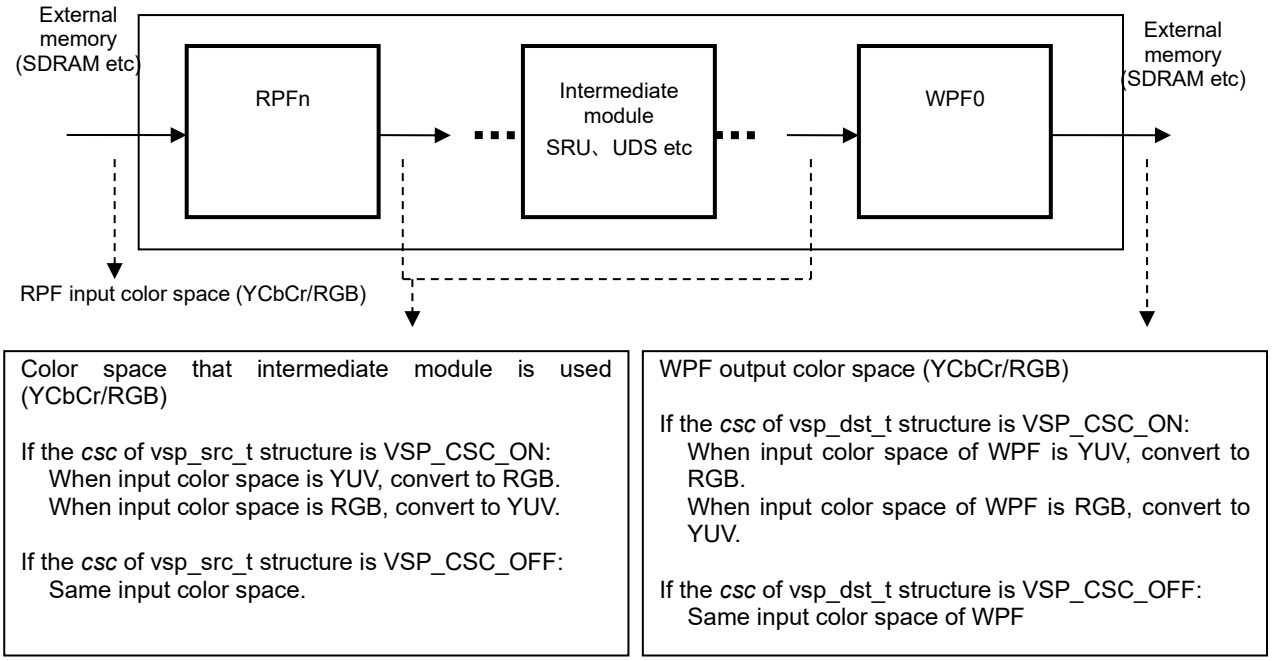


Figure 7-3 Input/Output format and color space

Figure 7-3 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 vsp_src_t structure. When using BRU, unify input color space on BRU.

7.1.1.1. vsp_alpha_unit_t

The following is described about the member of struct vsp_alpha_unit_t structure.

```
struct vsp_alpha_unit_t {
    unsigned int        addr_a;
    unsigned short      stride_a;
    unsigned char       swap;
    unsigned char       asel;
    unsigned char       aext;
    unsigned char       anum0;
    unsigned char       anum1;
    unsigned char       afix;
    struct vsp_irop_unit_t *irop;
    struct vsp_ckey_unit_t *ckey;
    struct vsp_mult_unit_t *mult;
};
```

Member	Direction	Contents
unsigned int *addr_a*	Input	Starting buffer address of alpha plane. When using alpha plane, specify. Specify continuous physical address.
unsigned short *stride_a*	Input	Stride of alpha plane. [byte]
unsigned char *swap*	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: If data array is big-endian, specify (VSP_SWAP_B|VSP_SWAP_W|VSP_SWAP_L|VSP_SWAP_LL) to this member.
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 7-3. VSP_ALPHA_NUM1 (0x00): 1/4/8bit packed alpha + plane alpha. 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 *ref_sel* of IROP unit is VSP_MSKEN_ALPHA and the *op_mode* of IROP unit 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 + plane alpha The 1bit packed alpha input is converted by the 8bit transparent alpha generator shown in Figure 7-4 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 7-4 according to the *anum0/1* setting into the 8bit alpha value as transparency information. VSP_ALPHA_NUM5 (0x04) : Fixed alpha value Note: If you use the following functions, can not set VSP_ALPHA_NUM4. Up-down scaler (UDS) Super-resolution (SRU) Rotation (except VSP_ROP_OFF and VSP_ROP_V_FLIP).
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 7-4 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 7-4 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.
struct vsp_irop_unit_t *irop	Input	Pointer to a 1 bit mask generator and IROP unit setting structure. When specify null pointer, alpha and image data go through IROP unit.
struct vsp_ckey_unit_t *ckey	Input	Pointer to a color keying setting structure. When specify null pointer, alpha and image data go through color keying unit. When a virtual input is valid (*vir* = VSP_VIR), does not refer this member.
struct vsp_mult_unit_t	Input	Pointer to a multiple setting structure. When specify null pointer, alpha and image data go through multiple unit.

7.1.1.2. vsp_irop_unit_t

The following is described about the member of vsp_irop_unit_t structure.

```
struct vsp_irop_unit_t {
    unsigned char       op_mode;
    unsigned char       ref_sel;
    unsigned char       bit_sel;
    unsigned long       comp_color;
    unsigned long       irop_color0;
    unsigned long       irop_color1;
};
```

Member	Direction	Contents
unsigned char *op_mode*	Input	IROP operation setting. The source (SRC) for the IROP operation is the pixel data and alpha data specified in the *irop_color0* or *irop_color1* 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. Specify define of Table 7-2. About available, refer to Table 7-4.
unsigned char *ref_sel*	Input	Reference source setting. Specifies the method of alpha value generation in the 1bit mask alpha generator shown Figure 7-4. 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 (*bit_sel* = VSP_ALPHA_1BIT), the 1bit mask value is output without change. When the input alpha is in the 8bit format (*bit_sel* = 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 *comp_color* 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 *op_mode* to VSP_IROP_NOP.
unsigned char *bit_sel*	Input	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 7-4. 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 *ref_sel* is set to

<table>
<tr><td></td><td></td><td>VSP_MSKEN_ALPHA. In other cases, this member setting has no effect.</td></tr>
<tr><td>unsigned long
comp_color</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 ref_sel member is set VSP_MSKEN_ALPHA.

RGB format
<table><tr><td></td><td>MSB</td><td></td><td></td><td></td><td></td><td>LSB</td></tr><tr><td></td><td>-</td><td>-</td><td>R (8bit)</td><td>G (8bit)</td><td>B (8bit)</td></tr><tr><td></td><td>63 32</td><td>31</td><td></td><td></td><td></td><td>0</td></tr></table>YUV format
<table><tr><td></td><td>MSB</td><td></td><td></td><td></td><td></td><td>LSB</td></tr><tr><td></td><td>-</td><td>-</td><td>Cr (8bit)</td><td>Y (8bit)</td><td>Cb (8bit)</td></tr><tr><td></td><td>63 32</td><td>31</td><td></td><td></td><td></td><td>0</td></tr></table></td></tr>
<tr><td>unsigned long
irop_color0</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 7-4)

RGB format
<table><tr><td></td><td>MSB</td><td></td><td></td><td></td><td></td><td>LSB</td></tr><tr><td></td><td>-</td><td>A (8bit)</td><td>R (8bit)</td><td>G (8bit)</td><td>B (8bit)</td></tr><tr><td></td><td>63 32</td><td>31</td><td></td><td></td><td></td><td>0</td></tr></table>YUV format
<table><tr><td></td><td>MSB</td><td></td><td></td><td></td><td></td><td>LSB</td></tr><tr><td></td><td>-</td><td>A (8bit)</td><td>Cr (8bit)</td><td>Y (8bit)</td><td>Cb (8bit)</td></tr><tr><td></td><td>63 32</td><td>31</td><td></td><td></td><td></td><td>0</td></tr></table></td></tr>
<tr><td>unsigned long
irop_color1</td><td>Input</td><td>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 7-4)

RGB format
<table><tr><td></td><td>MSB</td><td></td><td></td><td></td><td></td><td>LSB</td></tr><tr><td></td><td>-</td><td>A (8bit)</td><td>R (8bit)</td><td>G (8bit)</td><td>B (8bit)</td></tr><tr><td></td><td>63 32</td><td>31</td><td></td><td></td><td></td><td>0</td></tr></table>YUV format
<table><tr><td></td><td>MSB</td><td></td><td></td><td></td><td></td><td>LSB</td></tr><tr><td></td><td>-</td><td>A (8bit)</td><td>Cb (8bir)</td><td>Y (8bit)</td><td>Cr (8bit)</td></tr><tr><td></td><td>63 32</td><td>31</td><td></td><td></td><td></td><td>0</td></tr></table></td></tr>
</table>

Figure 7-4 shows configuration diagram of alpha plane.

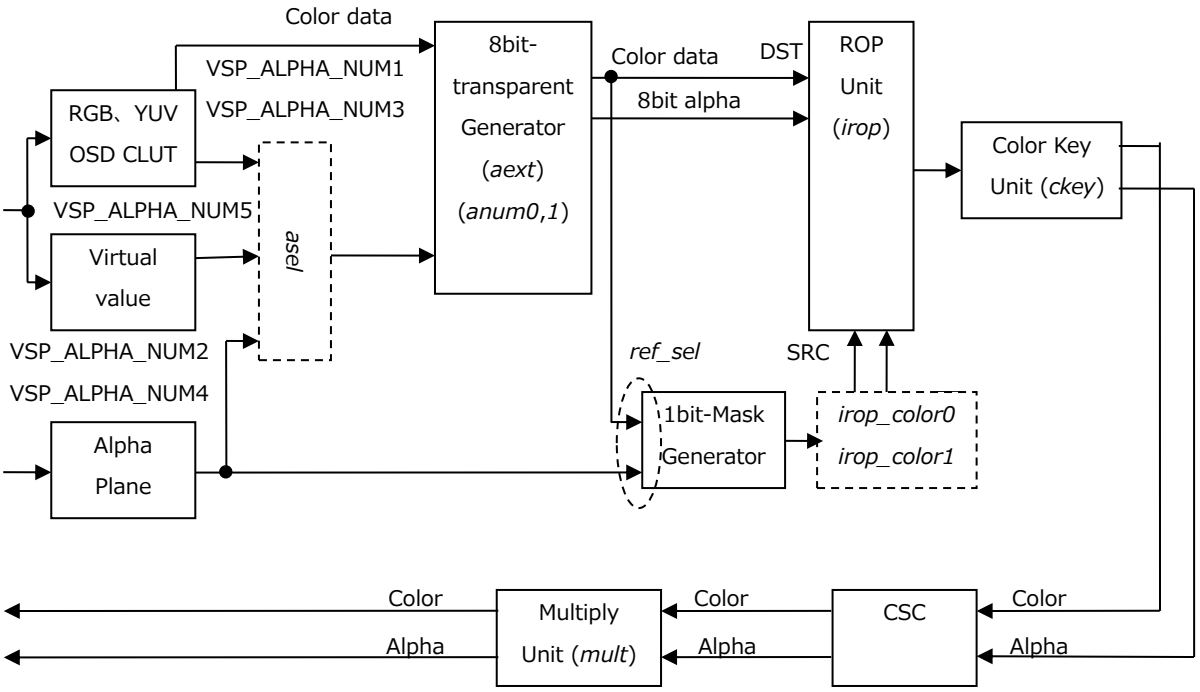


Figure 7-4 Configuration diagram of alpha plane

Decide alpha format and processing method by specify the *asel* of vsp_alpha_unit_t 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 *ref_sel*. When specify VSP_MSKEN_ALPHA to the *rel_sel*, 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 *ref_sel*. 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 *irop_color0*. When 1bit alpha value is 1, use *irop_color1*. When don't use mask information, specify VSP_IROP_NOP to *op_mode*. Likewise, When 1bit-Mask generator is invalid or the *asel* is set to VSP_ALPHA_NUM5, set to VSP_IROP_NOP.

Table 7-2 Define of Raster operation

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 7-3 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	*42afix* setting	*42afix* setting	*42afix* setting

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

Table 7-4 Select raster operation 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 *op_mode* of IROP unit.

7.1.1.3. vsp_ckey_unit_t

The following is described about the member of vsp_ckey_unit_t structure.

```
struct vsp_ckey_unit_t {
    unsigned char       mode;
    unsigned long       color1;
    unsigned long       color2;
};
```

Member	Direction	Contents
unsigned char *mode*	Input	Color keying setting. VSP_CKEY_THROUGH (0x00): Alpha and image data go through. VSP_CKEY_TRANS_COLOR1 (0x01): Transparent color mode (1 color). VSP_CKEY_TRANS_COLOR2 (0x02): Transparent color mode (2 colors). VSP_CKEY_MATCHED_COLOR (0x03): Matched color mode. VSP_CKEY_LUMA_THRESHOLD (0x04): Color-luma threshold mode.
unsigned long *color1*	Input	(1) Transparent color mode. Specify the color data (RGB or Y) to compare and the alpha value (A) to replace if they match. According to the setting of cext, specify the value of the extension after. RGB format: MSB LSB - (63–32) | A (8bit) (31–24) | R (8bit) (23–16) | G (8bit) (15–8) | B (8bit) (7–0) YUV format: MSB LSB - (63–32) | A (8bit) (31–24) | - (23–16) | Y (8bit) (15–8) | - (7–0) (2) Matched color mode. Specify the color data (RGB or Y) to compare. RGB format: MSB LSB - (63–32) | - (31–24) | R (8bit) (23–16) | G (8bit) (15–8) | B (8bit) (7–0) YUV format: MSB LSB - (63–32) | - (31–24) | - (23–16) | Y (8bit) (15–8) | - (7–0) According to the setting of cext, specify the value of the extension after. (3) Color-luma threshold mode. When the input data is in YUV format, and if Y value is equal to or smaller than Y value field of this member, the input alpha value is replaced with the value specified in alpha field of this member.

<table>
<tr>
<td>unsigned long
color2</td>
<td>Input</td>
<td>(1) Transparent color mode.
When the mode is set to VSP_CKEY_TRANS_COLOR2, this member is valid. Refer to the color1.

(2) Matched color mode.
When compared with color1 and matched, specify alpha and color data to replace.
<table>
<tr><td></td><td>MSB</td><td></td><td></td><td></td><td>LSB</td></tr>
<tr><td>RGB format</td><td>-</td><td>A (8bit)</td><td>R (8bit)</td><td>G (8bit)</td><td>B (8bit)</td></tr>
<tr><td></td><td>63 32</td><td>31</td><td></td><td></td><td>0</td></tr>
<tr><td></td><td>MSB</td><td></td><td></td><td></td><td>LSB</td></tr>
<tr><td>YUV format</td><td>-</td><td>A (8bit)</td><td>Cr (8bit)</td><td>Y (8bit)</td><td>Cb (8bit)</td></tr>
<tr><td></td><td>63 32</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.

(3) Color-luma threshold mode.
Color-luma threshold mode does not refer this member.</td>
</tr>
</table>

Note: When use color key to transparent color, please set the rbc of struct vsp_bld_ctrl_t to VSP_RBC_BLEND.
When an image is output from the RPF, the alpha value is changed according to the setting of the struct vsp_ckey_unit_t.
Thereafter, to blending using the set α value, setting the rbc of struct vsp_bld_ctrl_t to VSP_RBC_BLEND.
Composite color is calculated by the value of other parameters of struct vsp_bld_ctrl_t.

For example, if you want to make the color set with struct vsp_ckey_unit_t transparent, you can set the following values to parameter of struct vsp_bld_ctrl_t.

rbc = VSP_RBC_BLEND
blend_formula = VSP_FORM_BLEND0
blend_coefx = VSP_COEFFICIENT_BLENDX4
blend_coefy = VSP_COEFFICIENT_BLENDY3
aformula = VSP_FORM_ALPHA0
acoefx = VSP_COEFFICIENT_ALPHAX5
acoefy = VSP_COEFFICIENT_ALPHAY5
*acoefx*_fix = 1
*acoefy*_fix = 0

7.1.1.4. vsp_mult_unit_t

The following is described about the member of vsp_mult_unit_t structure.

```
struct vsp_mult_unit_t {
    unsigned char       a_mmd;
    unsigned char       p_mmd;
    unsigned char       ratio;
};
```

Member	Direction	Contents
unsigned char *a_mmd*	Input	Alpha data mode setting VSP_MULT_THROUGH (0x00): Alpha data go through. VSP_MULT_RATIO (0x01): Multiple unit multiplies alpha data by ratio. Note: When output format from csc unit is YUV, set VSP_MULT_THROUGH to *a_mmd*.
unsigned char *p_mmd*	Input	Image data mode setting VSP_MULT_THROUGH (0x00): Image data go through. VSP_MULT_RATIO (0x01): Multiple unit multiplies image data by *ratio*. VSP_MULT_ALPHA (0x02): Multiple unit multiplies image data by alpha data. VSP_MULT_RATIO_ALPHA (0x03): Multiple unit multiplies image data by *ratio* and alpha data. Note: When output format from csc unit is YUV, set VSP_MULT_THROUGH to *p_mmd*.
unsigned char *ratio*	Input	Multiple alpha value. Note: When specify VSP_MULT_RATIO or VSP_MULT_RATIO_ALPHA, this member is valid.

Table 7-5 Expression of output alpha data from multiple unit

a_mmd	*ratio*	
VSP_MULT_THROUGH	Don't care	Aout = Ain
VSP_MULT_RATIO	not 255	Aout = Ain * *ratio* / 256
	255	Aout = Ain

Table 7-6 Expression of output alpha data from multiple unit

p_mmd	*ratio*	Ain	
VSP_MULT_THROUGH	Don't care	Don't care	Dout = Din
VSP_MULT_RATIO	Not 255	Don't care	Dout = Din * *ratio* / 256
	255	Don't care	Dout = Din
VSP_MULT_ALPHA	Don't care	Not 255	Dout = Din * Ain / 256
	Don't care	255	Dout = Din
VSP_MULT_RATIO_ALPHA	Not 255	Not 255	Dout = Din * Ain * *ratio* / 256 / 256
	255	Not 255	Dout = Din * Ain / 256
	Not 255	255	Dout = Din * *ratio* / 256
	255	255	Dout = Din

7.1.2. vsp_dst_t

The following is described about the member of vsp_dst_t structure.

```
struct vsp_dst_t {
    unsigned int        addr;
    unsigned int        addr_c0;
    unsigned int        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       dith;
    unsigned char       rotation;
    struct fcp_info_t   *fcp;
};
```

Member	Direction	Contents
unsigned int *addr*	Input	Starting buffer address of Y or RGB. Specify continuous physical address. Note: If you use a FCNL compression (*fcnl* = FCP_FCNL_ENABLE), specify a multiple of 256.
unsigned int *addr_c0*	Input	Starting 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. Note: If you use a FCNL compression (*fcnl* = FCP_FCNL_ENABLE), specify a multiple of 256.
unsigned int *addr_c1*	Input	Starting buffer address of C When select the Planar of YUV, specify top buffer address of Cr plane. Specify continuous physical address. Note: If you use a FCNL compression (*fcnl* = FCP_FCNL_ENABLE), specify a multiple of 256.

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. Note: If you use a FCNL compression (*fcnl* = FCP_FCNL_ENABLE), specify a multiple of 256.
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. Note: If you use a FCNL compression (*fcnl* = FCP_FCNL_ENABLE), specify a multiple of 256.
unsigned short *width*	Input	Image horizontal size. [pixel] Specify horizontal image size. Input and output limited size is shown Table 7-9 and Table 7-10. When input format is YUV422 or YUV420. Specify a multiple of 2. When uses 90 or 270 degree rotation, specifies after rotation.
unsigned short *height*	Input	Image vertical size. [line] Specify vertical image size. Input and output limited size is shown Table 7-9 and Table 7-10. When input format is YUV420. Specify a multiple of 2. When uses 90 or 270 degree rotation, specifies after rotation.
unsigned short *x_offset*	Input	Horizontal offset. [pixel] Specify horizontal offset. When input format is YUV422 or YUV420, specify a multiple of 2. Note: If you use a FCNL compression (*fcnl* = FCP_FCNL_ENABLE), address after calculation must be a multiple of 256.
unsigned short *y_offset*	Input	Vertical offset. [line] Specify vertical offset. When input forma is YUV420, specify a multiple of 2. Note: If you use a FCNL compression (*fcnl* = FCP_FCNL_ENABLE), address after calculation must be a multiple of 256.
unsigned short *format*	Input	Output format setting. Specify define of "7.3.2 Output format". Note: If you use a FCNL compression (*fcnl* = FCP_FCNL_ENABLE), the following formats are available. VSP_OUT_PRGB8888 (RGB) VSP_OUT_YUV422_INT0_YUY2 (YUV) VSP_OUT_YUV444_PLANAR (YUV) VSP_OUT_YUV422_PLANAR (YUV) VSP_OUT_YUV420_PLANAR (YUV) Further if you use a 90 or 270 degree rotation, only VSP_OUT_PRGB8888 is available.

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 Note: If you use a FCNL compression (*fcnl* = FCP_FCNL_ENABLE), set VSP_SWAP_LL only.
unsigned char *pxa*	Input	PAD data select. Select the value to be stored in the bit field indicated as PAD or P in the packed RGB output formats shown in section 7.3.2.1. 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 7.3.2.1 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 7.3.2.1. 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. Note: If you use the following functions, set to 0. Up-down scaler (UDS) Super-resolution (SRU) Rotation (except VSP_ROT_OFF and VSP_ROT_V_FLIP)
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. Note: If you use the following functions, set to 0. Up-down scaler (UDS) Super-resolution (SRU) Rotation (except VSP_ROT_OFF and VSP_ROT_V_FLIP)
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 conversion (input format depth = output format depth)

<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 7.3.2.1 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 7.3.2.1 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 than the athres, 1 is stored in the P field. When the selected data is not greater than 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 *dith*	Input	Dithering 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_DITH_OFF (0x00): disable VSP_DITH_COLOR_REDUCTION (0x01): color reduction dither mode VSP_DITH_ORDERED_DITHER (0x02): ordered dither mode Note1: Color reduction dither is available for RGB format. Note2: Ordered dither is available only for 18bpp of RGB format. So when specify VSP_DITH_ORDERED_DITHER, set format at 18bpp of RGB. Note3: When you specify VSP_CSC_ON to *csc* of parameter, dither function is invalid. Specify VSP_DITH_OFF.
unsigned char *rotation*	Input	Rotation setting. VSP_ROT_OFF (0): no rotation and flipping VSP_ROT_V_FLIP (1): vertical flipping VSP_ROT_H_FLIP (2): horizontal flipping VSP_ROT_180 (3): 180 degree rotation VSP_ROT_90 (4): 90 degree rotation VSP_ROT_90_V_FLIP (5): 90 degree rotation and vertical flipping VSP_ROT_90_H_FLIP (6): 90 degree rotation and horizontal flipping VSP_ROT_270 (7): 270 degree rotation
struct fcp_info_t **fcp*	Input	Frame compression setting. Pointer to a frame compression setting structure.

7.1.2.1. fcp_info_t

The following is described about the member of fcp_info_t structure.

```
struct fcp_info_t {
    unsigned char       fcnl ;
    unsigned char       tlen ;
    unsigned short      pos_y ;
    unsigned short      pos_c ;
    unsigned short      stride_div16 ;
    unsigned int        ba_anc_prev_y ;
    unsigned int        ba_anc_cur_y ;
    unsigned int        ba_anc_next_y ;
    unsigned int        ba_anc_cur_c ;
    unsigned int        ba_ref_prev_y ;
    unsigned int        ba_ref_cur_y ;
    unsigned int        ba_ref_next_y ;
    unsigned int        ba_ref_cur_c ;
};
```

Member	Direction	Contents
unsigned char *fcnl*	Input	Renesas near-lossless compression setting. FCP_FCNL_DISABLE (0): Disable FCP_FCNL_ENABLE (1): Enable Note: Renesas near-lossless decompression is executed by DBSC4, DDR3/4 memory controller. When FCNL is enable, specify decompression area to destination buffer.
unsigned char *tlen*	Input	Not used.
unsigned short *pos_y*	Input	Not used.
unsigned short *pos_c*	Input	Not used.
unsigned short *stride_div16*	Input	Not used.
unsigned int *ba_anc_prev_y*	Input	Not used.
unsigned int *ba_anc_cur_y*	Input	Not used.
unsigned int *ba_anc_next_y*	Input	Not used.
unsigned int *ba_anc_cur_c*	Input	Not used.
unsigned int *ba_ref_prev_y*	Input	Not used.
unsigned int *ba_ref_cur_y*	Input	Not used.
unsigned int *ba_ref_next_y*	Input	Not used.
unsigned int *ba_ref_cur_c*	Input	Not used.

7.1.3. vsp_ctrl_t

The following is described about the member of vsp_ctrl_t structure.

```
struct vsp_ctrl_t {
    struct vsp_sru_t    *sru ;
    struct vsp_uds_t    *uds ;
    struct vsp_lut_t    *lut ;
    struct vsp_clu_t    *clu ;
    struct vsp_hst_t    *hst ;
    struct vsp_hsi_t    *hsi ;
    struct vsp_bru_t    *bru ;
    struct vsp_brs_t    *brs ;
    struct vsp_hgo_t    *hgo ;
    struct vsp_hgt_t    *hgt ;
    struct vsp_shp_t    *shp ;
} ;
```

Member	Direction	Contents
struct vsp_sru_t *sru	Input	Pointer to a super-resolution setting structure. If you set VSP_USE_SRU to *connect*, *sru* is referred.
struct vsp_uds_t *uds	Input	Pointer to an up-down scaler setting structure. If you set VSP_USE_UDS to *connect*, *uds* is referred.
struct vsp_lut_t *lut	Input	Pointer to a look-up table setting structure. If you set VSP_USE_LUT to *connect*, *lut* is referred.
struct vsp_clu_t *clu	Input	Pointer to a cubic look-up table setting structure. If you set VSP_USE_CLU to *connect*, *clu* is referred.
struct vsp_hst_t *hst	Input	Pointer to a hue saturation value transforming setting structure. If you set VSP_USE_HST to *connect*, *hst* is referred.
struct vsp_hsi_t *hsi	Input	Pointer to a hue saturation value transforming inverse setting structure. If you set VSP_USE_HSI to *connect*, *hsi* is referred.
struct vsp_bru_t *bru	Input	Pointer to a blend/ROP setting structure. If you set VSP_USE_BRU to *connect*, *bru* is referred.
struct vsp_brs_t *brs	-	unused.
struct vsp_hgo_t *hgo	Input	Pointer to a histogram generator-one setting structure. If you set VSP_USE_HGO to *use_module*, *hgo* is referred.
struct vsp_hgt_t *hgt	Input	Pointer to a histogram generator-two setting structure. If you set VSP_USE_HGT to *use_module*, *hgt* is referred.
struct vsp_shp_t *shp	Input	Pointer to a sharpness setting structure. If you set VSP_USE_SHP to *use_module*, *shp* is referred.

Note: The *connect* is member of each module's structure.
The *use_module* is member of vsp_start_t's structure.
Set NULL to the member of unused modules in struct vsp_ctrl_t.

7.1.3.1. vsp_sru_t

The following is described about the member of vsp_sru_t structure.

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

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	Super 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.
usigned 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_LUT_USE (0x0010) : Look up table VSP_CLU_USE (0x0020) : Cubic-Look up table VSP_HST_USE (0x0040) : Hue saturation value transform VSP_SHP_USE (0x0800) : Sharpness

7.1.3.2. vsp_uds_t

The following is described about the member of vsp_uds_t structure.

```
struct vsp_uds_t {
    unsigned char       amd;
    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 long       connect ;
} ;
```

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 *clip*	Input	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 *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
unsigned char *alpha*	Input	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
unsigned char *complement*	Input	Interpolation method. Specifies the interpolation method. Recommending method is multi-tap. 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*.

unsigned char *athres0*	Input	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.
unsigned char *athres1*	Input	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.
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 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_SHP_USE (0x0800) : Sharpness

7.1.3.3. vsp_lut_t

The following is described about the member of vsp_lut_t structure.

```
struct vsp_lut_t {
    struct vsp_dl_t     lut;
    unsigned char       fxa ;
    unsigned long       connect ;
} ;
```

Member	Direction	Contents
struct vsp_dl_t *lut*	Input	Look up table. Specify color lookup table. The setting range of *tbl_num* is 1 to 256. When the *size* specified fewer than 256, areas not setting does not guarantee.
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_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_SHP_USE (0x0800) : Sharpness

Table 7-7 storage method of lut buffer.

offset	MSB (31)			LSB (0)
0	0x00007000			
	Don't care	R/Cr/H (8bit)	G/Y/S (8bit)	B/Cb/V (8bit)
1	0x00007004			
	Don't care	R/Cr/H (8bit)	G/Y/S (8bit)	B/Cb/V (8bit)
2	0x00007008			
	Don't care	R/Cr/H (8bit)	G/Y/S (8bit)	B/Cb/V (8bit)
…	…			
n	0x00007000 + n * 4			
	Don't care	R/Cr/H (8bit)	G/Y/S (8bit)	B/Cb/V (8bit)
…	…			
254	0x000073F8			
	Don't care	R/Cr/H (8bit)	G/Y/S (8bit)	B/Cb/V (8bit)
255	0x000073FC			
	Don't care	R/Cr/H (8bit)	G/Y/S (8bit)	B/Cb/V (8bit)

Note: When color format is RGB, set to R, G, and B.
When color format is YUV, set to Cr, Y and Cb.
When color format is HSV, set to H, S and V.

7.1.3.4. vsp_clu_t

The following is described about the member of vsp_clu_t structure.

```
struct vsp_clu_t {
    unsigned char       mode;
    struct vsp_dl_t     clu;
    unsigned char       fxa;
    unsigned long       connect;
};
```

<table>
<tr><th>Member</th><th>Direction</th><th>Contents</th></tr>
<tr><td>unsigned char
mode</td><td>Input</td><td>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) :
Operates in 2D mode with automatic table address increment.</td></tr>
<tr><td>struct
vsp_dl_t clu</td><td>Input</td><td>Cubic look-up table
Specify color lookup table.
When you automatic table address increment, the setting range of tbl_num is 1 to 4913 in 3D mode and 1 to 289 in 2D mode.
When you use normal operation mode, the setting range of tbl_num is 2 to 9826 in 3D mode and 2 to 578 in 2D mode. Specify multiple of 2.

When operates in 3D mode.
<table>
<tr><th></th><th>1st axis</th><th>2nd axis</th><th>3rd axis</th></tr>
<tr><td>coordinate</td><td>valid</td><td>valid</td><td>valid</td></tr>
<tr><td>component</td><td>R/Cr/H (8bit)</td><td>G/Y/S (8bit)</td><td>B/Cb/V (8bit)</td></tr>
</table>
When operates in 2D mode.
<table>
<tr><th></th><th>1st axis</th><th>2nd axis</th><th>3rd axis</th></tr>
<tr><td>coordinate</td><td>valid</td><td>valid</td><td>invalid</td></tr>
<tr><td>component</td><td>0</td><td>Y (8bit)</td><td>0</td></tr>
</table></td></tr>
<tr><td>unsigned char
fxa</td><td>Input</td><td>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.</td></tr>
</table>

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_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 VSP_SHP_USE (0x0800) : Sharpness

Table 7-8 shows the relationship between a coordinate and a component. A coordinate and a component are same buffer array.

Table 7-8 storage method of coordinate and component value

(1) VSP_CLU_MODE_3D/VSP_CLU_MODE_2D

offset	element	MSB (31)			LSB (0)
0	Coordinate [31:0]	0x00007400			
		-	1st axis	2nd axis	3rd axis
1	Component [31:0]	0x00007404			
		-	1st axis	2nd axis	3rd axis
2	Coordinate [31:0]	0x00007400			
		-	1st axis	2nd axis	3rd axis
3	Component [31:0]	0x00007404			
		-	1st axis	2nd axis	3rd axis
…	…	…			
9824	Coordinate [31:0]	0x00007400			
		-	1st axis	2nd axis	3rd axis
9825	Component [31:0]	0x00007404			
		-	1st axis	2nd axis	3rd axis

Note: 2D mode range is 0 to 577. 3D mode range is 0 to 9825.

(2) VSP_CLU_MODE_3D_AUTO/VSP_CLU_MODE_2D_AUTO

offset	Coordinate (automatic increment)				Component [31 :0]			
0	-	0	0	0	0x00007404			
					-	1st axis	2nd axis	3rd axis
1	-	1	0	0	0x00007404			
					-	1st axis	2nd axis	3rd axis
…	…	…	…	…	…	…	…	…
15	-	15	0	0	0x00007404			
					-	1st axis	2nd axis	3rd axis
16	-	16	0	0	0x00007404			
					-	1st axis	2nd axis	3rd axis
17	-	0	1	0	0x00007404			
					-	1st axis	2nd axis	3rd axis
18	-	1	1	0	0x00007404			
					-	1st axis	2nd axis	3rd axis
…	…	…	…	…	…	…	…	…
287	-	15	16	0	0x00007404			
					-	1st axis	2nd axis	3rd axis
288	-	16	16	0	0x00007404			
					-	1st axis	2nd axis	3rd axis
289	-	0	0	1	0x00007404			
					-	1st axis	2nd axis	3rd axis
290	-	1	0	1	0x00007404			
					-	1st axis	2nd axis	3rd axis
…	…	…	…	…	…	…	…	…
4911	-	15	16	16	0x00007404			
					-	1st axis	2nd axis	3rd axis
4912	-	16	16	16	0x00007404			
					-	1st axis	2nd axis	3rd axis

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

7.1.3.5. vsp_hst_t

The following is described about the member of vsp_hst_t structure.

```
struct vsp_hst_t {
    unsigned char       fxa;
    unsigned long       connect;
};
```

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

7.1.3.6. vsp_hsi_t

The following is described about the member of vsp_hsi_t structure.

```
struct vsp_hsi_t {
    unsigned char       fxa;
    unsigned long       connect;
};
```

Member	Direction	Contents
unsigned char *fxa*	Input	Fixed alpha output value setting. The HSI does not support input/output of the alpha value. The alpha value input to the HSI is discarded, and the fixed alpha value specified in this param is always output from the HSI.
unsigned long *connect*	Input	Processing connection setting. Specify the module to be executed next to the HSI. If connect to WPF from HSI, you set 0. VSP_SRU_USE (0x0001) : Super-resolution VSP_UDS_USE (0x0002) : 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_SHP_USE (0x0800) : Sharpness

7.1.3.7. vsp_bru_t

The following is described about the member of vsp_bru_t structure.

```
struct vsp_bru_t {
    unsigned long               lay_order;
    unsigned char               adiv;
    struct vsp_bld_dither_t     *dither_unit[5];
    struct vsp_bld_vir_t        *blend_virtual;
    struct vsp_bld_ctrl_t       *blend_unit_a;
    struct vsp_bld_ctrl_t       *blend_unit_b;
    struct vsp_bld_ctrl_t       *blend_unit_c;
    struct vsp_bld_ctrl_t       *blend_unit_d;
    struct vsp_bld_ctrl_t       *blend_unit_e;
    struct vsp_bld_rop_t        *rop_unit;
    unsigned long               connect;
};
```

<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 src_par[0])
VSP_LAY_2 (0x02): input image 2 (correspond to the src_par[1])
VSP_LAY_3 (0x03): input image 3 (correspond to the src_par[2])
VSP_LAY_4 (0x04): input image 4 (correspond to the src_par[3])
VSP_LAY_VIRTUAL (0x05): virtual input
VSP_LAY_5 (0x06): input image 5 (correspond to the src_par[4])

MSB LSB
<table>
<tr><td>-</td><td>5th from lowest back</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_E</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>63-24</td><td>23-20</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>

struct vsp_bld_dither_t *dither_unit[5]	Input	Dither unit setting When specify null pointer, dithering will be disable. The *dither_unit[0]* corresponds to the input image 1. The *dither_unit[1]* corresponds to the input image 2. The other is also similar.
struct vsp_bld_vir_t *blend_virtual	Input	Pointer to a structure virtual input setting. When you specify the VSP_LAY_VIRTUAL to *lay_order*, this member will be referred.
struct vsp_bld_ctrl_t *blend_unit_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.
struct vsp_bld_ctrl_t *blend_unit_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.
struct vsp_bld_ctrl_t *blend_unit_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.
struct vsp_bld_ctrl_t *blend_unit_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.
struct vsp_bld_ctrl_t *blend_unit_e	Input	Pointer to a structure of Blend/ROP Unit E. When you specify VSP_LAY_NO to SRC_E or null pointer to the member, the Blend/ROP unit through to the DST_E.
struct vsp_bld_rop_t *rop_unit	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. Note: When setting *rop_unit*, be sure to set *blend_unit_c*. It is prohibited to set null pointer to *blend_unit_c* when *rop_unit* is set.
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_LUT_USE (0x0010) : Look up table VSP_CLU_USE (0x0020) : Cubic-Look up table

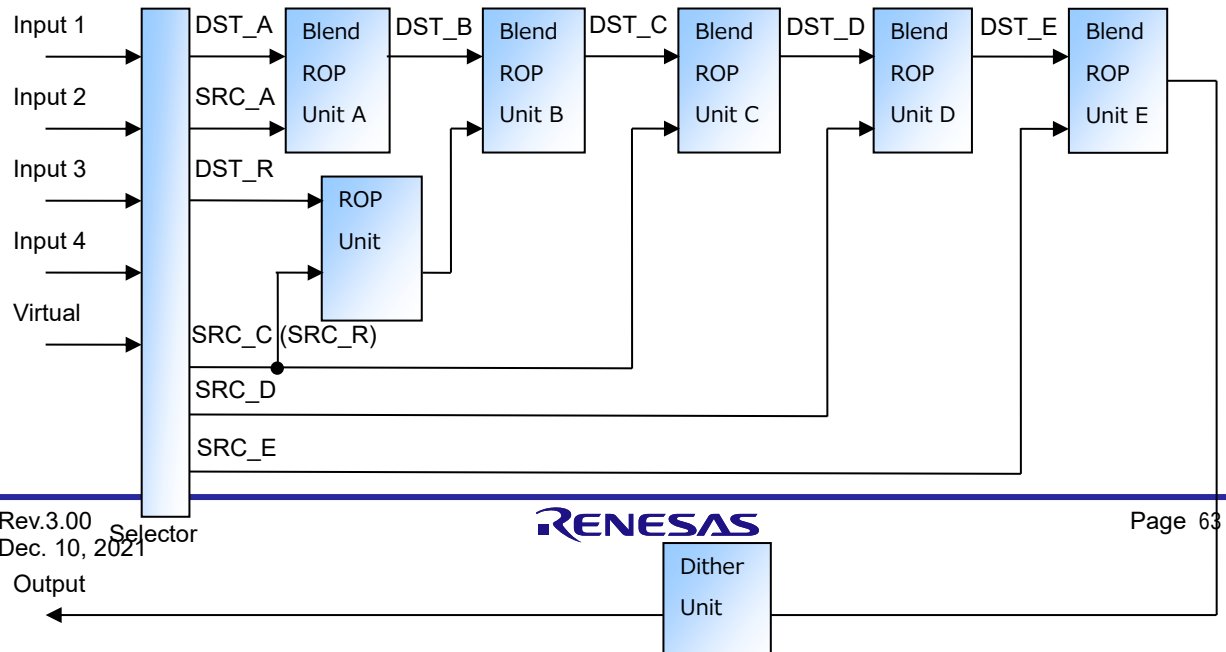


Figure 7-5 Configuration BLEND/ROP unit

Figure 7-5 shows configuration Blend/ROP unit. The Blend/ROP unit is composed of 5 multifunction units and a ROP unit. Source (SRC) and destination (DST) of The Blend/ROP unit is specified the *lay_order* of vsp_bru_t. You can specify 6 parameters of DST_A, SRC_A, DST_R, SRC_C (SRC_R) SRC_D and SRC_E. 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_unit_a*, *blend_unit_b*, *blend_unit_c*, *blend_unit_d, blend_unit_e* and and *rop_unit*.
- 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 vsp_start_t.

Example1:
when *rpf_num* = 1 (*src_par[0]* is valid), can specify VSP_LAY_1/VSP_LAY_VIRTUAL.
Example2:
when *rpf_num* = 2 (*src_par[0]* and *src_par[1]* are valid), when specify VSP_LAY_2 only, this is parameter error. Must be set VSP_LAY_1.

(a) vsp_bld_dither_t

The following is described about the member of vsp_bld_dither_t structure.

```
struct vsp_bld_dither_t {
    unsigned char       mode;
    unsigned char       bpp;
};
```

Member	Direction	Contents
unsigned char *mode*	Input	Dither unit setting. Select dithering function. VSP_DITH_COLOR_REDUCTION (0x01): color reduction dither mode VSP_DITH_ORDERED_DITHER (0x02): ordered dither mode
unsigned char *bpp*	Input	Number of color for pixels after dithering setting. Specify the number of colors for pixels after dithering. When specify VSP_DITH_ORDERED_DITHER to *mode*, specify VSP_DITH_18BPP to *bpp*. 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)

(b) vsp_bld_vir_t

The following is described about the member of vsp_bld_vir_t structure.

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

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. RGB format: MSB LSB - (63 to 32) | A (8bit) | R (8bit) | G (8bit) | B (8bit) (31 to 0) YUV format: MSB LSB - (63 to 32) | A (8bit) | Cr (8bit) | Y (8bit) | Cb (8bit) (31 to 0)

(c) vsp_bld_ctrl_t

The following is described about the member of vsp_bld_ctrl_t structure.

```
struct vsp_bld_ctrl_t {
    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 ;
};
```

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 7-2 Define of Raster operation”.
unsigned char *arop*	Input	Raster operation setting of alpha value. Can specify the defined “Table 7-2 Define of Raster operation”.
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.

(d) vsp_bld_rop_t

The following is described about the member of vsp_bld_rop_t structure.

```
struct vsp_bld_rop_t {
    unsigned char       crop;
    unsigned char       arop;
};
```

Member	Direction	Contents
unsigned char *crop*	Input	Raster operation setting of color data. Can specify the defined “Table 7-2 Define of Raster operation”.
unsigned char *arop*	Input	Raster operation setting of alpha value. Can specify the defined “Table 7-2 Define of Raster operation”.

7.1.3.8. vsp_hgo_t

The following is described about the member of vsp_hgo_t structure.

```
struct vsp_hgo_t {
    unsigned int        hard_addr;
    void                *virt_addr;
    void                *mem_par;
    unsigned short      width;
    unsigned short      height;
    unsigned short      x_offset;
    unsigned short      y_offset;
    unsigned char       binary_mode ;
    unsigned char       maxrgb_mode ;
    unsigned char       step_mode ;
    unsigned short      x_skip ;
    unsigned short      y_skip ;
    unsigned long       sampling ;
} ;
```

Member	Direction	Contents
unsigned int *hard_addr*	Input	Histogram buffer address. 256 byte alignment is required. Also, specify the physical address. Buffer size request 1088 bytes or more.
void **virt_addr*	Input	Pointer to a histogram buffer address. 256 byte alignment is required. Also, specify the virttual address. Buffer size request 1088 bytes or more.
void **mem_par*	-	Not used.
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	Histogram 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 char *step_mode*	Input	Histogram step of Y or maximum RGB setting. VSP_STEP_64 (0x00): 64 step mode. VSP_STEP_256 (0x01): 256 step mode. Note: VSP_STEP_256 is available Y component or maximum value of RGB (VSP_MAXRGB_ON).
unsigned short *x_skip*	Input	Horizontal pixel skipping mode setting VSP_SKIP_OFF (0x00) : No skipping. VSP_SKIP_1_2 (0x01) : Horizontal 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_SRC5 (4) : 5th input source VSP_SMPPT_SRU (16) : Super-resolution VSP_SMPPT_UDS (17) : 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 VSP_SMPPT_SHP (46) : Sharpness

The HGO uses 1088 bytes. Be allocating memory over 1088 bytes.

(1) 64 step mode & maxRGB disable

Offset	Component	Bit [31:0]
+0	R/Cr/H	R_HISTOGRAM_0[23:0]
+1		R_HISTOGRAM_1[23:0]
...		...
+62		R_HISTOGRAM_62[23:0]
+63		R_HISTOGRAM_63[23:0]
+64	G/Y/S	G_HISTOGRAM_0[23:0]
+65		G_HISTOGRAM_1[23:0]
...		...
+126		G_HISTOGRAM_62[23:0]
+127		G_HISTOGRAM_63[23:0]
+128	B/Cb/V	B_HISTOGRAM_0[23:0]
+129		B_HISTOGRAM_1[23:0]
...		...
+190		B_HISTOGRAM_62[23:0]
+191		B_HISTOGRAM_63[23:0]
+192	N.A	Reserved
	...	
+271	N.A	Reserved

(2) 64 step mode & maxRGB enable

Offset	Component	Bit [31:0]
+0	N.A	Reserved
		...
+63		Reserved
+64	max(R, G, B)	HISTOGRAM_0 [23:0]
+65		HISTOGRAM_1 [23:0]
...		...
+126		HISTOGRAM_62 [23:0]
+127		HISTOGRAM_63 [23:0]
+128	N.A	Reserved
		...
+271		Reserved

(3) 256 step mode

Offset	Component	Bit [31:0]
+0	max(R, G, B)	HISTOGRAM_0 [23:0]
+1		HISTOGRAM_1 [23:0]
...		...
+254		HISTOGRAM_254 [23:0]
+255		HISTOGRAM_255 [23:0]
+256	N.A	Reserved
		...
+271		Reserved

7.1.3.9. vsp_hgt_t

The following is described about the member of vsp_hgt_t structure.

```
struct vsp_hgt_t {
    unsigned int          hard_addr;
    void                  *virt_addr;
    void                  *mem_par;
    unsigned short        width;
    unsigned short        height;
    unsigned short        x_offset;
    unsigned short        y_offset;
    unsigned short        x_skip ;
    unsigned short        y_skip ;
    struct vsp_hue_area_t area[6] ;
    unsigned long         sampling ;
} ;
```

Member	Direction	Contents
unsigned int *hard_addr*	Input	Histogram buffer address. 256 byte alignment is required. Also, specify the physical address. Buffer size request 800 bytes or more.
void **virt_addr*	Input	Pointer to a histogram buffer address. 256 byte alignment is required. Also, specify the virttual address. Buffer size request 800 bytes or more.
void **mem_par*	-	Not used.
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_SKIP_1_2 (0x01) : Horizontal 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.
struct vsp_hue_area_t *area[6]*	Input	HUE area structure. Please refer to the vsp_hue_area_t 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_SRC5	(4) : 5th input source
		VSP_SMPPT_SRU	(16) : Super-resolution
		VSP_SMPPT_UDS	(17) : 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
		VSP_SMPPT_SHP	(46) : Sharpness

(a) vsp_hue_area_t

The following is described about the member of vsp_hue_area_t structure.

```
struct vsp_hue_area_t {
    unsigned char       lower;
    unsigned char       upper;
};
```

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 7-6.

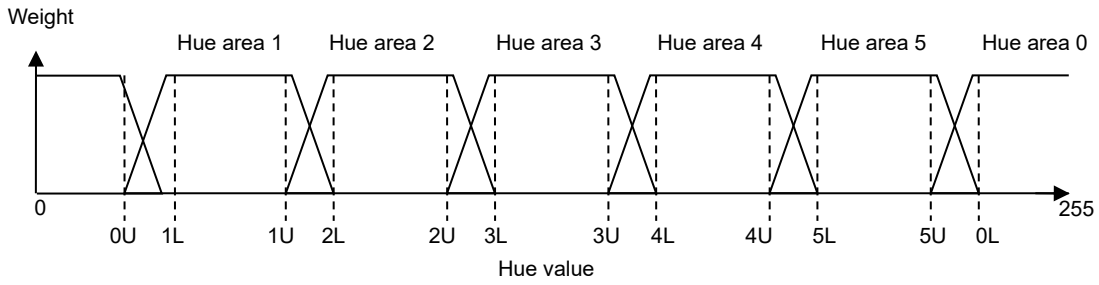


Figure 7-6 Weighting Histogram Using Hue

0L = *area[0].lower* 0U = *area[0].upper*
1L = *area[1].lower* 1U = *area[1].upper*
…
5L = *area[5].lower* 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 800 bytes. Be allocating memory over 800 bytes.

Offset	Hue area	Bit [31:0]
+0	Hue Area 0	HISTOGRAM_0 [21:0]
+1		HISTOGRAM_1 [21:0]
...		...
+30		HISTOGRAM_30 [21:0]
+31		HISTOGRAM_31 [21:0]
+32	Hue Area 1	HISTOGRAM_0 [21:0]
+33		HISTOGRAM_1 [21:0]
...		...
+62		HISTOGRAM_30 [21:0]
+63		HISTOGRAM_31 [21:0]
…		
+160	Hue Area 5	HISTOGRAM_0 [21:0]
+161		HISTOGRAM_1 [21:0]
...		...
+190		HISTOGRAM_30 [21:0]
+191		HISTOGRAM_31 [21:0]
+192	N.A	Reserved.
…		
+199	N.A	Reserved.

7.1.3.10. vsp_shp_t

The following is described about the member of vsp_shp_t structure.

```
struct vsp_shp_t {
    unsigned char       mode;
    unsigned char       gain0;
    unsigned char       limit0;
    unsigned char       gain10;
    unsigned char       limit10;
    unsigned char       gain11;
    unsigned char       limit11;
    unsigned char       gain20;
    unsigned char       limit20;
    unsigned char       gain21;
    unsigned char       limit21;
    unsigned char       fxa;
    unsigned long       connect;
};
```

Member	Direction	Contents
unsigned char *mode*	Input	Sharpness or Blurring processing setting. VSP_SHP_SHARP (0x00): sharpness select VSP_SHP_UNSHARP (0x02): blurring select Refer to the H/W manual for setting following parameter. Sharpness setting as shown in Table 32.41 and blurring setting as shown in Table 32.42. There are 24 types of setting value to apply sharpness and 24 types of setting value to apply blurring. Specify an appropriate value from among its combination.
unsigned char *gain0*	Input	Sharpness parameter Gain0
unsigned char *limit0*	Input	Sharpness parameter Limit0
unsigned char *gain10*	Input	Sharpness parameter Gain10
unsigned char *limit10*	Input	Sharpness parameter Limit10
unsigned char *gain11*	Input	Sharpness parameter Gain11
unsigned char *limit11*	Input	Sharpness parameter Limit11
unsigned char *gain20*	Input	Sharpness parameter Gain20
unsigned char *limit20*	Input	Sharpness parameter Limit20
unsigned char *gain21*	Input	Sharpness parameter Gain21
unsigned char *limit21*	Input	Sharpness parameter Limit21

unsigned char *fxa*	Input	Fixed alpha output value setting. The SHP does not support input/output of the alpha value. The alpha value input to the SHP is discarded, and the fixed alpha value specified in this param is always output from the SHP.
unsigned long *connect*	Input	Processing connection setting. Specify the module to be executed next to the SHP. If connect to WPF from SHP, you set 0. VSP_SRU_USE (0x0001) : Super-resolution VSP_UDS_USE (0x0002) : 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

7.1.4. vsp_dl_t

The following is described about the member of vsp_dl_t structure.

```
struct vsp_lut_t {
    unsigned int        hard_addr;
    void                *virt_addr ;
    unsigned short      tbl_num ;
    void                *mem_par ;
} ;
```

Member	Direction	Contents
unsigned int *hard_addr*	Input	Display list buffer address that H/W IP can access. Allocate memory size is *tbl_num* * 8 bytes. Specify the same area as CPU can access.
void *virt_addr	Input	Display list buffer address that CPU can access. Allocate memory size is *tbl_num* * 8 bytes. Specify the same area as H/W IP can access.
unsigned short *tbl_num*	Input	Set table number. (1 to 16383) Value to set to *tbl_num* is refer to each member.
void *mem_par	-	Not used.

7.2. Input/Output image limited size

Table 7-9 and Table 7-10 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.

Table 7-9 Minimum size of input/output image

Processing module		Input [pixel]		Output [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
HSI		1	1	1	1
BRU		1	1	1	1
HGO		1	1	1	1
HGT		1	1	1	1
SHP		4	4	4	4
WPF		1	1	1	1

Table 7-10 Maximum size of input/output image

Processing module		Input [pixel]		Output [pixel]	
		width	height	width	height
RPF		8190	8190	8190	8190
SRU	Normal size	8190	8190	8190	8190
	Double size	8190	4095	8190	8190
UDS	Scale-down	8190	8190	8190	8190
	Scale-up	8190	8190	8190	8190
LUT		8190	8190	8190	8190
CLU		8190	8190	8190	8190
HST		8190	8190	8190	8190
HSI		8190	8190	8190	8190
BRU		8190	8190	8190	8190
HGO		8190	8190	8190	8190
HGT		8190	8190	8190	8190
SHP		8190	8190	8190	8190
WPF		8190	8190	8190	8190

7.3. Format

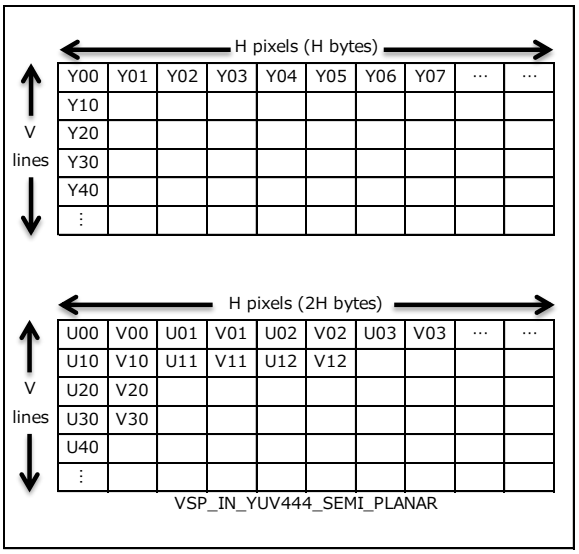
7.3.1. Input format

7.3.1.1. RGB format

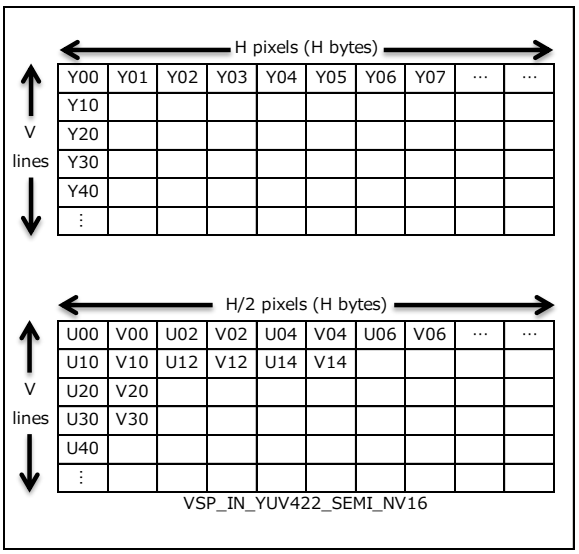
format	byte	phase	address Note) n	n+1	n+2	n+3
VSP_IN_RGB332	1		R0R0R0G0G0G0B0B0	R1R1R1G1G1G1B1B1	R2R2R2G2G2G2B2B2	R3R3R3G3G3G3B3B3
VSP_IN_XRGB4444	2		R0R0R0R0G0G0G0G0	B0B0B0B0R1R1R1R1	G1G1G1G1B1B1B1B1	
VSP_IN_RGBX4444	2		R0R0R0R0G0G0G0G0	B0B0B0B0R1R1R1R1	G1G1G1G1B1B1B1B1	
VSP_IN_XRGB1555	2		R0R0R0R0R0G0G0G0	G0B0B0B0B0R1R1R1	R1R1G1G1G1G1B1B1	B1B1B1B1
VSP_IN_RGBX5551	2		R0R0R0R0R0G0G0G0	G0G0B0B0B0B0R1R1	R1G1G1G1G1G1B1B1	B1B1B1B1
VSP_IN_RGB565	2		R0R0R0R0R0G0G0G0	G0G0G0B0B0B0B0R1	R1R1G1G1G1G1G1B1	B1B1B1B1
VSP_IN_AXRGB86666	4		A0A0A0A0A0A0A0A0		R0R0R0R0R0R0G0G0	G0G0G0G0G0G0B0B0
VSP_IN_RGBXA66668	4		R0R0R0R0R0R0G0G0	G0G0G0G0B0B0B0B0	B0B0	A0A0A0A0A0A0A0A0
VSP_IN_XRGBA66668	4		R0R0R0R0R0R0G0G0	G0G0G0G0B0B0B0B0	B0B0B0B0B0B0A0A0	A0A0A0A0A0A0A0A0
VSP_IN_ARGBX86666	4		A0A0A0A0A0A0A0A0	R0R0R0R0R0R0G0G0	G0G0G0G0B0B0B0B0	B0B0
VSP_IN_AXXXRGB82666	4		A0A0A0A0A0A0A0A0	R0R0R0R0R0R0	G0G0G0G0G0G0	B0B0B0B0B0B0
VSP_IN_XXXRGBA26668	4		R0R0R0R0R0R0R0R0	G0G0G0G0G0G0	B0B0B0B0B0B0A0A0	A0A0A0A0A0A0A0A0
VSP_IN_ARGBXXX86662	4		A0A0A0A0A0A0A0A0	R0R0R0R0R0R0	G0G0G0G0G0G0	B0B0B0B0B0B0
VSP_IN_RGBXXXA66628	4		R0R0R0R0R0R0	G0G0G0G0G0G0	B0B0B0B0B0B0	A0A0A0A0A0A0A0A0
VSP_IN_XRGB6666	3	0	R0R0R0R0R0R0G0G0	G0G0G0G0B0B0B0B0	B0B0	R1R1
		1	R1R1R1R1G1G1G1G1	G1G1B1B1B1B1B1B1	R2R2R2R2R2R2G2G2	G2G2
		2	G2G2B2B2B2B2B2B2	R3R3R3R3R3R3G3G3	G3G3G3G3B3B3B3B3	B3B3
VSP_IN_RGBX6666	3	0	R0R0R0R0R0R0G0G0	G0G0G0G0B0B0B0B0	B0B0	R1R1R1R1R1R1G1G1
		1	G1G1G1G1B1B1B1B1	B1B1	R2R2R2R2R2R2G2G2	G2G2G2G2B2B2B2B2
		2	B2B2	R3R3R3R3R3R3G3G3	G3G3G3G3B3B3B3B3	B3B3
VSP_IN_XXXRGB2666	3	0	R0R0R0R0R0R0G0G0	G0G0G0G0B0B0B0B0	B0B0	R1R1R1R1R1R1
		1	G1G1G1G1G1G1B1B1	B1B1B1B1R2R2R2R2	R2R2G2G2G2G2G2G2	
		2	B2B2B2B2B2B2R3R3	R3R3R3R3G3G3G3G3	G3G3B3B3B3B3B3B3	
VSP_IN_RGBXXX6662	3	0	R0R0R0R0R0R0G0G0	G0G0G0G0B0B0B0B0	B0B0	R1R1R1R1R1R1
		1	G1G1G1G1G1G1B1B1	B1B1B1B1R2R2R2R2	R2R2G2G2G2G2G2G2	
		2	B2B2B2B2B2B2R3R3	R3R3R3R3G3G3G3G3	G3G3B3B3B3B3B3B3	
VSP_IN_ARGB8888	4		A0A0A0A0A0A0A0A0	R0R0R0R0R0R0R0R0	G0G0G0G0G0G0G0G0	B0B0B0B0B0B0B0B0
VSP_IN_RGBA8888	4		R0R0R0R0R0R0R0R0	G0G0G0G0G0G0G0G0	B0B0B0B0B0B0B0B0	A0A0A0A0A0A0A0A0
VSP_IN_RGB888	3	0	R0R0R0R0R0R0R0R0	G0G0G0G0G0G0G0G0	B0B0B0B0B0B0B0B0	R1R1R1R1R1R1R1R1
		1	G1G1G1G1G1G1G1G1	B1B1B1B1B1B1B1B1	R2R2R2R2R2R2R2R2	G2G2G2G2G2G2G2G2
		2	B2B2B2B2B2B2B2B2	R3R3R3R3R3R3R3R3	G3G3G3G3G3G3G3G3	B3B3B3B3B3B3B3B3
VSP_IN_XXRGB7666	4		R0R0R0R0R0R0G0G0	G0G0G0B0B0B0B0B0	B0B0	
VSP_IN_XRGB14666	4		R0R0R0R0R0R0G0G0	G0G0G0G0B0B0B0B0	B0B0	
VSP_IN_BGR888	3	0	B0B0B0B0B0B0B0B0	G0G0G0G0G0G0G0G0	R0R0R0R0R0R0R0R0	B1B1B1B1B1B1B1B1
		1	G1G1G1G1G1G1G1G1	R1R1R1R1R1R1R1R1	B2B2B2B2B2B2B2B2	G2G2G2G2G2G2G2G2
		2	R2R2R2R2R2R2R2R2	B3B3B3B3B3B3B3B3	G3G3G3G3G3G3G3G3	R3R3R3R3R3R3R3R3
VSP_IN_ARGB4444	2		A0A0A0A0R0R0R0R0	G0G0G0G0B0B0B0B0	A1A1A1A1R1R1R1R1	G1G1G1G1B1B1B1B1
VSP_IN_RGBA4444	2		R0R0R0R0G0G0G0G0	B0B0B0B0A0A0A0A0	R1R1R1R1G1G1G1G1	B1B1B1B1A1A1A1A1
VSP_IN_ARGB1555	2		A0R0R0R0R0R0G0G0	G0G0G0B0B0B0B0B0	A1R1R1R1R1R1G1G1	G1G1G1B1B1B1B1B1
VSP_IN_RGBA5551	2		R0R0R0R0R0G0G0G0	G0G0B0B0B0B0B0A0	R1R1R1R1R1G1G1G1	G1G1B1B1B1B1B1A1
VSP_IN_ABGR4444	2		A0A0A0A0B0B0B0B0	G0G0G0G0R0R0R0R0	A1A1A1A1B1B1B1B1	G1G1G1G1R1R1R1R1
VSP_IN_BGRA4444	2		B0B0B0B0G0G0G0G0	R0R0R0R0A0A0A0A0	B1B1B1B1G1G1G1G1	R1R1R1R1A1A1A1A1
VSP_IN_ABGR1555	2		A0B0B0B0B0B0G0G0	G0G0G0G0R0R0R0R0R0	A1B1B1B1B1B1G1G1	G1G1G1G1R1R1R1R1R1
VSP_IN_BGRA5551	2		B0B0B0B0B0G0G0G0	G0G0G0R0R0R0R0R0	A0B1B1B1B1B1G1G1	G1G1G1R1R1R1R1R1A1
VSP_IN_XXXBGR2666	3	0	B0B0B0B0B0B0G0G0	G0G0G0G0R0R0R0R0	R0R0	B1B1B1B1B1B1
		1	G1G1G1G1G1G1R1R1	R1R1R1R1B2B2B2B2	B2B2G2G2G2G2G2G2	
		2	R2R2R2R2R2R2B3B3	B3B3B3B3G3G3G3G3	G3G3R3R3R3R3R3R3	
VSP_IN_ABGR8888	4		A0A0A0A0A0A0A0A0	B0B0B0B0B0B0B0B0	G0G0G0G0G0G0G0G0	R0R0R0R0R0R0R0R0
VSP_IN_XRGB16565	4		R0R0R0R0R0G0G0G0	G0G0G0B0B0B0B0B0		
VSP_IN_RGB_CLUT_DATA			RGB_CLUT_DATA0	RGB_CLUT_DATA1	RGB_CLUT_DATA2	RGB_CLUT_DATA3

Note) In this case, swap setting is VSP_SWAP_B|VSP_SWAP_W|VSP_SWAP_L|VSP_SWAP_LL.

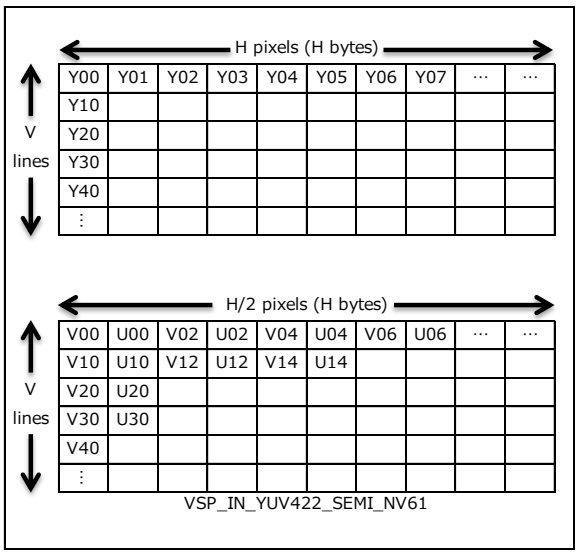
7.3.1.2. YCbCr (Semi planar) format



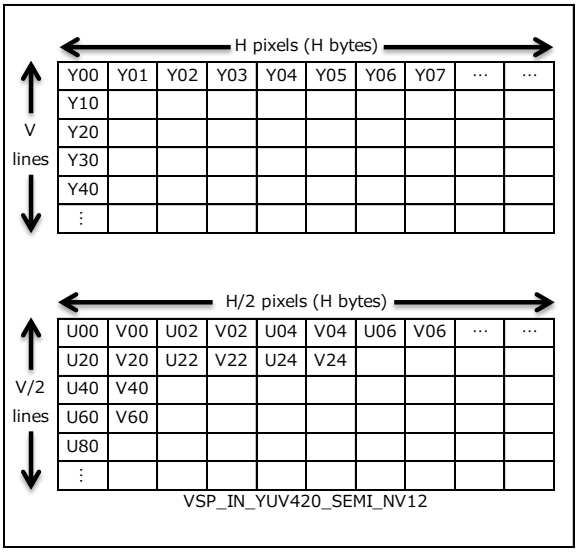
VSP_IN_YUV444_SEMI_PLANAR



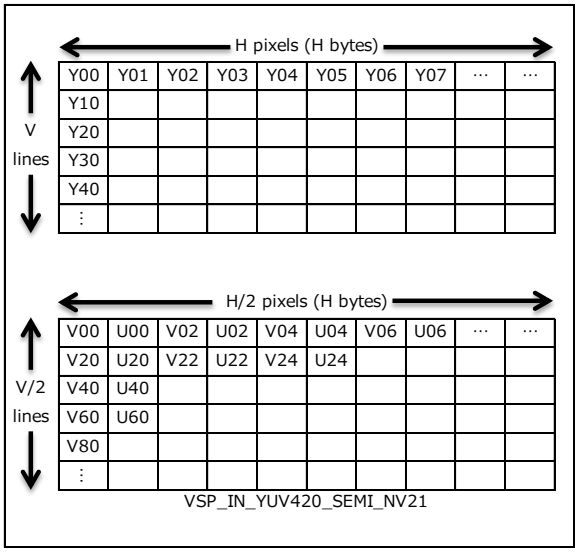
VSP_IN_YUV422_SEMI_NV16



VSP_IN_YUV422_SEMI_NV61

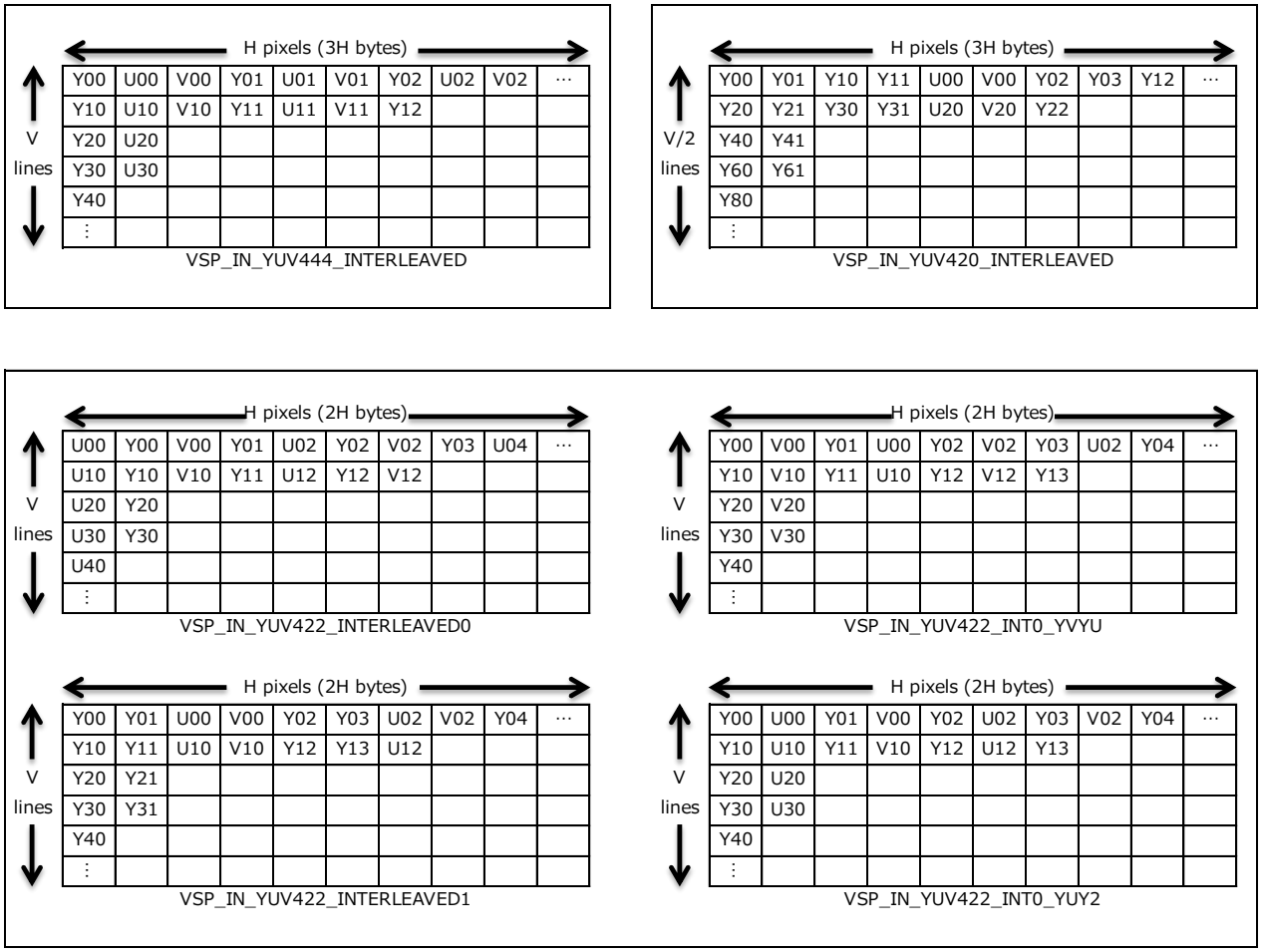


VSP_IN_YUV420_SEMI_NV12

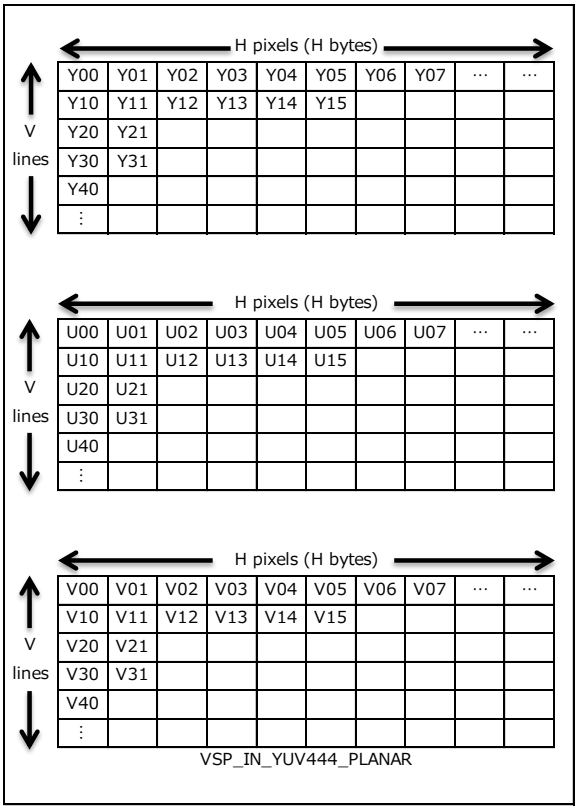


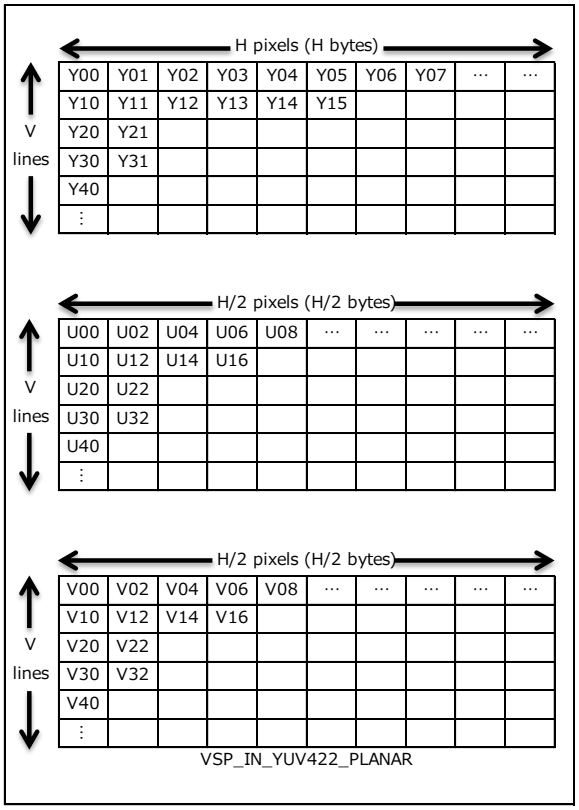
VSP_IN_YUV420_SEMI_NV21

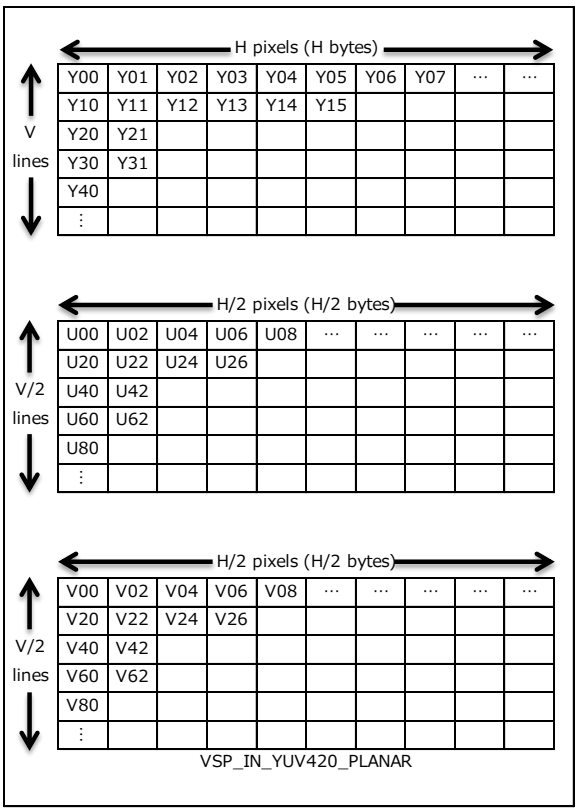
7.3.1.3. YCbCr (Interleaved) format



7.3.1.4. YCbCr (Planar) format







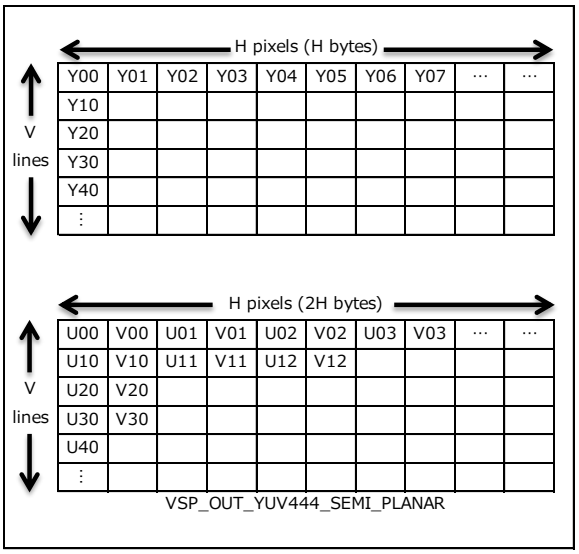
7.3.2. Output format

7.3.2.1. RGB format

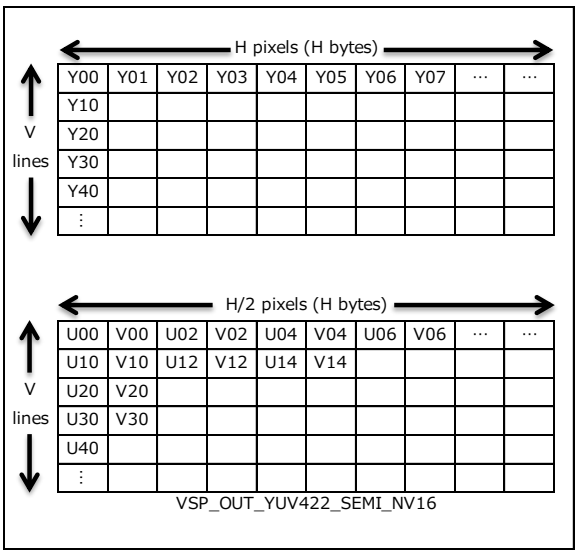
format	byte	phase	address Note)																													
			n								n+1								n+2								n+3							
VSP_OUT_RGB332	1		R0	R0	R0	G0	G0	G0	B0	B0	R1	R1	R1	G1	G1	G1	B1	B1	R2	R2	R2	G2	G2	G2	B2	B2	R3	R3	R3	G3	G3	G3	B3	B3
VSP_OUT_XRGB4444	2		0	0	0	0	R0	R0	R0	R0	G0	G0	G0	G0	B0	B0	B0	B0	0	0	0	0	R1	R1	R1	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	0	0	0	0	R1	R1	R1	R1	G1	G1	G1	G1	B1	B1	B1	B1	0	0	0	0
VSP_OUT_XRGB1555	2		0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	0	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	G0	B0	B0	B0	B0	B0	B0	0	R1	R1	R1	R1	R1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1	0
VSP_OUT_RGB565	2		R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	G0	B0	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	0	0	0	0	0	0	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	0	0	0	0	0	0	P0	P0	P0	P0	P0	P0	P0	P0
VSP_OUT_XRGBP66668	4		0	0	0	0	0	0	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	0	0	0	0	0	0
VSP_OUT_PXXXRGB82666	4		P0	P0	P0	P0	P0	P0	P0	P0	0	0	R0	R0	R0	R0	R0	R0	0	0	G0	G0	G0	G0	G0	G0	0	0	B0	B0	B0	B0	B0	B0
VSP_OUT_XXXRGBP26668	4		0	0	R0	R0	R0	R0	R0	R0	0	0	G0	G0	G0	G0	G0	G0	0	0	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	0	0	G0	G0	G0	G0	G0	G0	0	0	B0	B0	B0	B0	B0	B0	0	0
VSP_OUT_RGBXXXP66628	4		R0	R0	R0	R0	R0	R0	0	0	G0	G0	G0	G0	G0	G0	0	0	B0	B0	B0	B0	B0	B0	0	0	P0	P0	P0	P0	P0	P0	P0	P0
VSP_OUT_XRGB6666	3	0	0	0	0	0	0	R0	R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	B0	B0	B0	B0	B0	0	0	0	0	0	0	R1	R1	
		1	R1	R1	R1	R1	G1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1	B1	0	0	0	0	0	0	R2	R2	R2	R2	R2	R2	G2	G2	G2	G2
		2	G2	G2	B2	B2	B2	B2	B2	B2	0	0	0	0	0	0	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	0	0	0	0	0	0	R1	R1	R1	R1	R1	R1	G1	G1
		1	G1	G1	G1	G1	B1	B1	B1	B1	B1	B1	0	0	0	0	0	0	R2	R2	R2	R2	R2	R2	G2	G2	G2	G2	G2	G2	B2	B2	B2	B2
		2	B2	B2	0	0	0	0	0	0	R3	R3	R3	R3	R3	R3	G3	G3	G3	G3	G3	G3	B3	B3	B3	B3	B3	B3	0	0	0	0	0	0
VSP_OUT_XXXRGB2666	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
VSP_OUT_RGBXXX6662	3	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	0	0
		1	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	0	0
		2	B2	B2	B2	B2	B2	B2	0	0	R3	R3	R3	R3	R3	R3	G3	G3	G3	G3	G3	G3	B3	B3	B3	B3	B3	B3	0	0	0	0	0	0
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		0	0	0	0	0	0	0	R0	R0	R0	R0	R0	R0	R0	G0	G0	G0	0	0	0	0	0	0	0	G0	G0	G0	B0	B0	B0	B0	B0	B0	B0
VSP_OUT_XRGB14666	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
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	G0	B0	B0	B0	B0	B0	P1	R1	R1	R1	R1	R1	G1	G1	G1	G1	G1	B1	B1	B1	B1	B1	B1
VSP_OUT_RGBP5551	2		R0	R0	R0	R0	R0	G0	G0	G0	G0	G0	G0	B0	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	G0	R0	R0	R0	R0	R0	P1	B1	B1	B1	B1	B1	G1	G1	G1	G1	G1	R1	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	R0	P0	B1	B1	B1	B1	B1	G1	G1	G1	G1	G1	R1	R1	R1	R1	R1	P1
VSP_OUT_XXXBGR2666	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
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		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

Note) In this case, swap setting is VSP_SWAP_B|VSP_SWAP_W|VSP_SWAP_L|VSP_SWAP_LL.

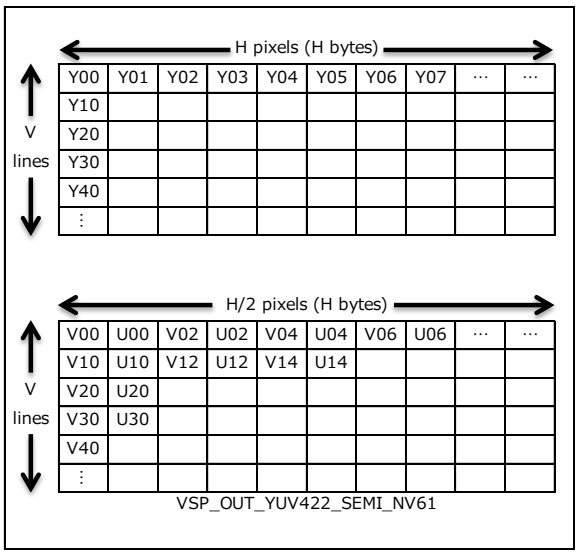
7.3.2.2. YCbCr (Semi planar) format



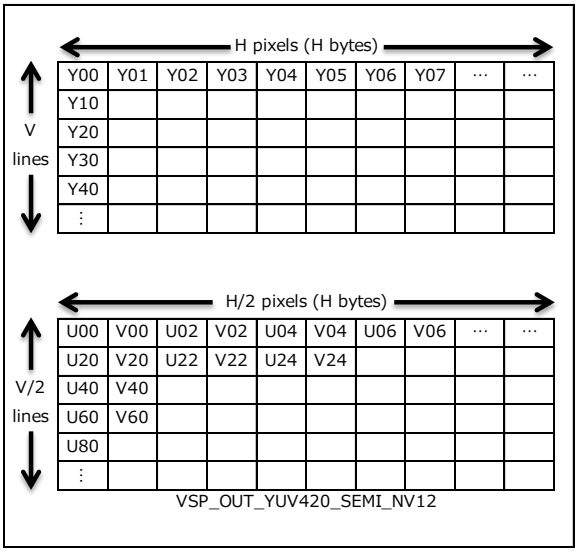
VSP_OUT_YUV444_SEMI_PLANAR



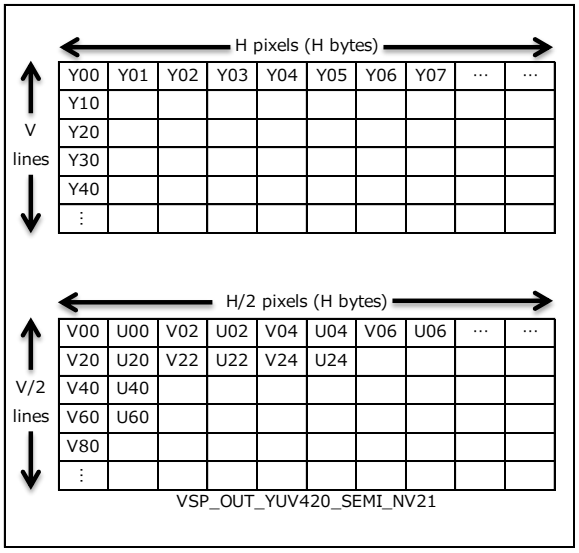
VSP_OUT_YUV422_SEMI_NV16



VSP_OUT_YUV422_SEMI_NV61

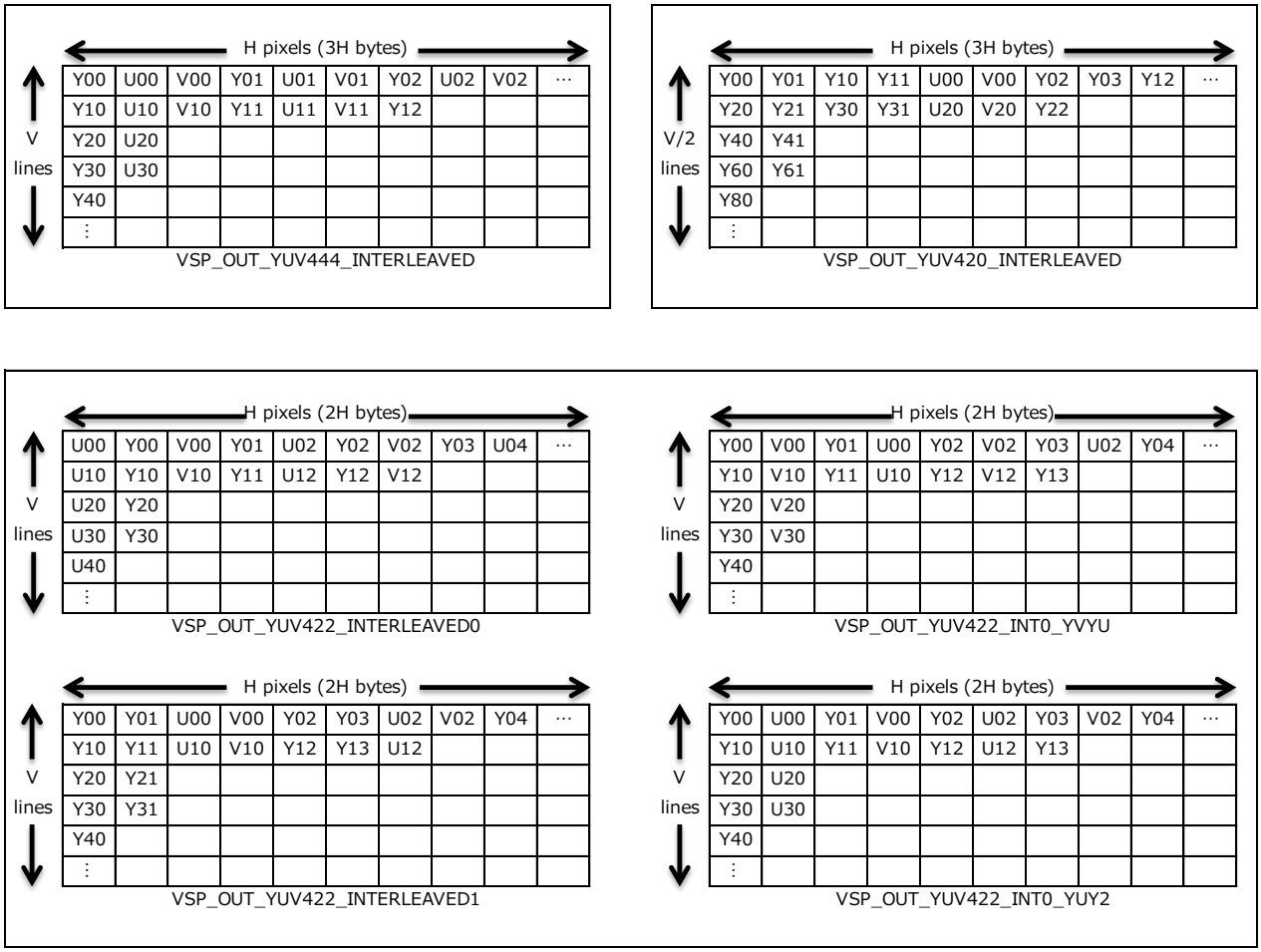


VSP_OUT_YUV420_SEMI_NV12

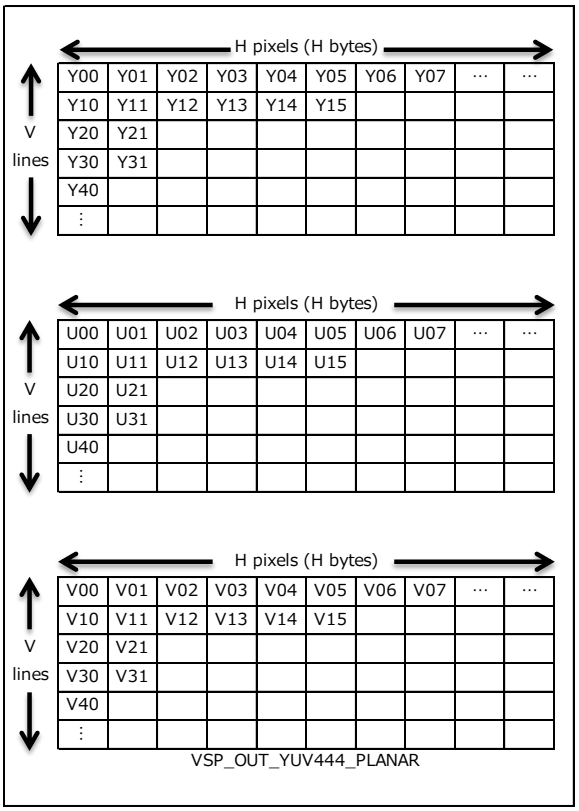


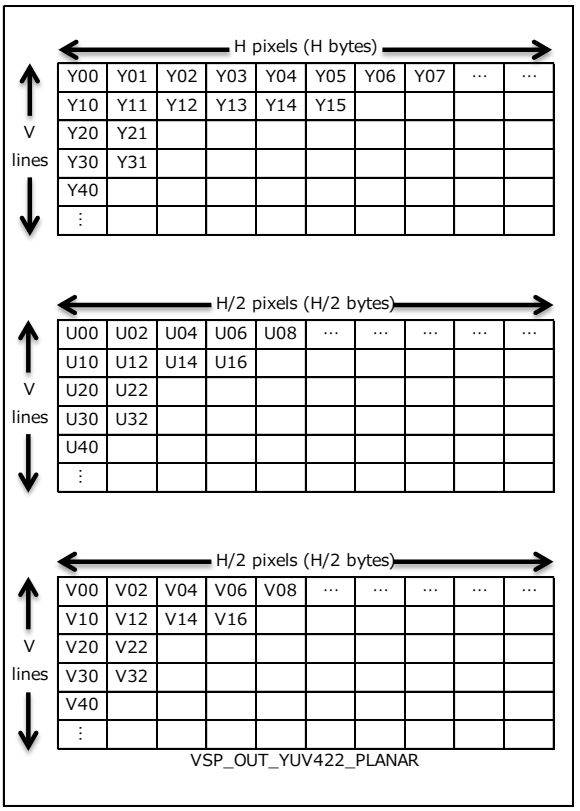
VSP_OUT_YUV420_SEMI_NV21

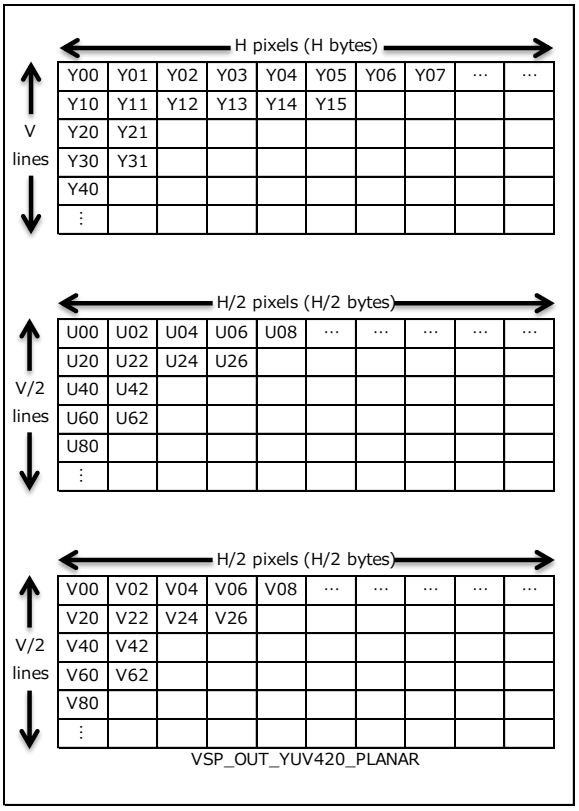
7.3.2.3. YCbCr (Interleaved) format



7.3.2.4. YCbCr (Planar) format







7.4. Error code

Table 7-11 shows the detail error code of VSP. According to error code, please check argument.

Table 7-11 Detail of error code

Define name	Error code	Contains
E_VSP_INVALID_STATE	-105	When you specify occupy mode, VSP is working.
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 vsp_start_t was null pointer.
E_VSP_PARA_CTRLPAR	-214	The *ctrl_par* of vsp_start_t was null pointer.
E_VSP_PARA_NOIN	-215	The *src_par* of vsp_start_t 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 vsp_src_t was null pointer. Note: When 'vir' was VSP_NO_VIR.
E_VSP_PARA_IN_ADRC0	-221	Then *addr_c0* of vsp_src_t was null pointer when source format was YUV (semi planar or planar). Note: When *vir* was VSP_NO_VIR.
E_VSP_PARA_IN_ADRC1	-222	The *addr_c1* of vsp_src_t 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 vsp_src_t 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 vsp_src_t 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 vsp_src_t 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.
		The *width_ex* wasn't 0 when use SRU, UDS or rotation (except VSP_ROT_OFF and VSP_ROT_V_FLIP).
E_VSP_PARA_IN_HEIGHTEX	-226	The *height_ex* of vsp_src_t 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.
		The *height_ex* wasn't 0 when use SRU, UDS or rotation (except VSP_ROT_OFF and VSP_ROT_V_FLIP).
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 vsp_src_t was out of specification.
		When *vir* was VSP_VIR, the *format* of vsp_src_t was other than VSP_IN_ARGB8888 and VSP_IN_YUV444_SEMI_PLANAR.

E_VSP_PARA_IN_XPOSI	-231	When *pwd* was VSP_LAYER_CHILD, calculating value of the *x_position* + *width* was greater than input image size.
E_VSP_PARA_IN_YPOSI	-232	When *pwd* was VSP_LAYER_CHILD, calculating value of the *y_position* + *height* was greater than input image size.
E_VSP_PARA_IN_CIPM	-233	The *cipm* of vsp_src_t was out of specification.
E_VSP_PARA_IN_CEXT	-234	The *cext* of vsp_src_t was out of specification.
E_VSP_PARA_IN_CSC	-235	When *vir* was VSP_NO_VIR, the *csc* of vsp_src_t was out of specification.
		When *vir* was VSP_VIR, the *csc* of vsp_src_t was other than VSP_CSC_OFF.
E_VSP_PARA_IN_ITURBT	-236	The *iturbt* of vsp_src_t was out of specification.
E_VSP_PARA_IN_CLRCNG	-237	The *clrcng* of vsp_src_t was out of specification.
E_VSP_PARA_IN_VIR	-238	The *vir* of vsp_src_t was out of specification.
E_VSP_PARA_IN_ALPHA	-239	The *alpha* of vsp_src_t was null pointer.
E_VSP_PARA_IN_CONNECT	-240	The *connect* of vsp_src_t was out of specification.
E_VSP_PARA_IN_PWD	-241	The *pwd* of vsp_src_t was out of specification.
E_VSP_PARA_OSD_CLUT	-250	The *hard_addr* or *virt_addr* of vsp_dl_t was null pointer.
E_VSP_PARA_OSD_SIZE	-251	The *tbl_num* of vsp_dl_t was out of range 1-256.
E_VSP_PARA_ALPHA_ADR	-260	The *addr_a* of vsp_alpha_unit_t was null pointer. Note: When use alpha plane.
E_VSP_PARA_ALPHA_CKEY	-261	The *mode* of vsp_ckey_unit_t was out of specification.
E_VSP_PARA_ALPHA_ASEL	-263	When enable virtual input, the *asel* of vsp_alpha_unit_t was other than VSP_ALPHA_NUM5.
		When disable virtual input, the *asel* of vsp_alpha_unit_t was out of specification.
		When use SRU, UDS or rotation (except VSP_ROT_OFF and VSP_ROT_V_FLIP), the *asel* of vsp_alpha_unit_t was VSP_ALPHA_NUM4.
E_VSP_PARA_ALPHA_AEXT	-264	The *aext* of vsp_alpha_unit_t was out of specification. Note: When the *asel* was VSP_ALPHA_NUM1
E_VSP_PARA_ALPHA_IROP	-265	The *op_mode* of vsp_irop_unit_t was out of specification. Note: When the *asel* was other than VSP_ALPHA_NUM5
		The *op_mode* of vsp_irop_unit_t was other than VSP_IROP_NOP Note: When the *asel* was VSP_ALPHA_NUM5
E_VSP_PARA_ALPHA_MSKEN	-266	The *ref_sel* of vsp_irop_unit_t was out of specification.
E_VSP_PARA_ALPHA_BSEL	-267	The *bit_sel* of vsp_irop_unit_t was out of specification. Note: When the *asel* was VSP_ALPHA_NUM1 or VSP_ALPHA_NUM3, and the *ref_sel* was VSP_MSKEN_ALPHA.
E_VSP_PARA_ALPHA_MULT	-268	The *a_mmd* or *p_mmd* were out of specification.
		Color space of multiple unit was other than YUV.
E_VSP_PARA_OUT_ADR	-270	The *addr* of vsp_dst_t was null pointer.
		The *addr* wasn't a multiple of 256 when fcnl enable.
E_VSP_PARA_OUT_ADRC0	-271	The *addr_c0* of vsp_dst_t was null pointer when destination format was YUV (semi planar or planar).
		The *addr_c0* wasn't a multiple of 256 when fcnl enable.
E_VSP_PARA_OUT_ADRC1	-272	The *addr_c1* of vsp_dst_t was null pointer when destination format was YUV (planar).
		The *addr_c1* wasn't a multiple of 256 when fcnl enable.
E_VSP_PARA_OUT_WIDTH	-273	The *width* of vsp_dst_t was 0.
		The *width* wasn't a multiple of 2 when destination format was YUV.

E_VSP_PARA_OUT_HEIGHT	-274	The *height* of vsp_dst_t 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	Calculating value of the *x_coffse* + *width* was greater than input horizontal size.
E_VSP_PARA_OUT_YCLIP	-278	Calculating value of the *y_coffset* + *height* was greater than input vertical size.
E_VSP_PARA_OUT_FORMAT	-279	The *format* of vsp_dst_t was out of specification.
E_VSP_PARA_OUT_SWAP	-280	The *swap* of vsp_dst_t was out of specification.
E_VSP_PARA_OUT_PXA	-281	The *pxa* of vsp_dst_t was out of specification.
E_VSP_PARA_OUT_XCOFFSET	-282	The *x_coffset* of vsp_dst_t was greater than 255.
		When use SRU, UDS or rotation (except VSP_ROT_OFF and VSP_ROT_V_FLIP), the *x_coffset* of vsp_dst_t was other than 0.
E_VSP_PARA_OUT_YCOFFSET	-283	The *y_coffset* of vsp_dst_t was greater than 255
		When use SRU, UDS or rotation (except VSP_ROT_OFF and VSP_ROT_V_FLIP), the *y_coffset* of vsp_dst_t was other than 0.
E_VSP_PARA_OUT_CSC	-284	The *csc* of vsp_dst_t was out of specification.
E_VSP_PARA_OUT_ITURBT	-285	The *iturbt* of vsp_dst_t was out of specification.
E_VSP_PARA_OUT_CLRCNG	-286	The *clrcng* of vsp_dst_t was out of specification.
E_VSP_PARA_OUT_CBRM	-287	The *cbrm* of vsp_dst_t was out of specification.
E_VSP_PARA_OUT_ABRM	-288	The *abrm* of vsp_dst_t was out of specification.
E_VSP_PARA_OUT_CLMD	-289	The *clmd* of vsp_dst_t was out of specification.
E_VSP_PARA_OUT_ROTATION	-290	The *rotation* of vsp_dst_t was out of specification.
E_VSP_PARA_OUT_DITH	-291	The *dith* of vsp_dst_t was out of specification.
E_VSP_PARA_OUT_INHSV	-292	Color space for input to the WPF was the HSV.
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_OUT_STRIDE_Y	-296	The *stride* of vsp_dst_t wasn't a multiple of 256 when fcnl enable.
E_VSP_PARA_OUT_STRIDE_C	-297	The *stride_c* of vsp_dst_t wasn't a multiple of 256 when fcnl enable.
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 vsp_bru_t was out of specification.
E_VSP_PARA_BRU_DITH_MODE	-302	The *mode* of vsp_bld_dither_t was out of specification.
E_VSP_PARA_BRU_DITH_BPP	-303	The *bpp* of vsp_bld_dither_t was out of specification.
E_VSP_PARA_BRU_CONNECT	-304	The *connect* of vsp_bru_t was out of specification.
E_VSP_PARA_BRU_INHSV	-305	Color space for input to the BRU was the HSV.
E_VSP_PARA_BRU_INCOLOR	-306	Image format for input to the BRU were not unified.
E_VSP_PARA_VIR_ADR	-310	The *blend_virtual* of vsp_bru_t was null pointer. Note: The *lay_order* was specified VSP_LAY_VIRTUAL.
E_VSP_PARA_VIR_WIDTH	-311	The *width* of vsp_bld_vir_t was out of range 1-8190. Note: The *lay_order* was specified VSP_LAY_VIRTUAL.
E_VSP_PARA_VIR_HEIGHT	-312	The *height* of vsp_bld_vir_t 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, calculating value of the *x_position* + *width* was greater 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, calculating value of the *y_position* + *height* was greater than input image size. Note: The *lay_order* was specified VSP_LAY_VIRTUAL.
E_VSP_PARA_VIR_PWD	-315	The *pwd* of vsp_bld_vir_t was out of specification. Note: The *lay_order* was specified VSP_LAY_VIRTUAL.
E_VSP_PARA_BLEND_RBC	-320	The *rbc* of vsp_vld_ctrl_t was out of specification. Note: The *blend_unit* was not null pointer.
E_VSP_PARA_BLEND_CROP	-321	The *crop* of vsp_bld_ctrl_t was out of specification. Note: The *blend_unit* was not null pointer.
E_VSP_PARA_BLEND_AROP	-322	The *arop* of vsp_bld_ctrl_t was out of specification. Note: The *blend_unit* was not null pointer.
E_VSP_PARA_BLEND_FORM	-323	The *blend_formula* of vsp_bld_ctrl_t was out of specification. Note: The *blend_unit* was not null pointer.
E_VSP_PARA_BLEND_COEFX	-324	The *blend_coefx* of vsp_bld_ctrl_t was out of specification. Note: The *blend_unit* was not null pointer.
E_VSP_PARA_BLEND_COEFY	-325	The *blend_corfy* of vsp_bld_ctrl_t was out of specification. Note: The *blend_unit* was not null pointer.
E_VSP_PARA_BLEND_AFORM	-326	The *aformula* of vsp_bld_ctrl_t was out of specification. Note: The *blend_unit* was not null pointer.
E_VSP_PARA_BLEND_ACOEFX	-327	The *acoefx* of vsp_bld_ctrl_t was out of specification. Note: The *blend_unit* was not null pointer.
E_VSP_PARA_BLEND_ACOEFY	-328	The *acoefy* of vsp_bld_ctrl_t was out of specification. Note: The *blend_unit* was not null pointer.
E_VSP_PARA_ROP_CROP	-330	The *crop* of vsp_bld_rop_t was out of specification. Note: When *rop_unit* was not null pointer.
E_VSP_PARA_ROP_AROP	-331	The *arop* of vsp_bld_rop_t was out of specification. Note: When *rop_unit* was not null pointer.
E_VSP_PARA_SRU_MODE	-340	The *mode* of vsp_sru_t was out of specification.
E_VSP_PARA_SRU_PARAM	-341	The *param* of vsp_sru_t was specified invalid parameter.
E_VSP_PARA_SRU_ENSCL	-342	The *enscl* of vsp_sru_t was out of specification.
E_VSP_PARA_SRU_CONNECT	-343	The *connect* of vsp_sru_t 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 the HSV.
E_VSP_PARA_UDS_AMD	-350	The *amd* of vsp_uds_t was out of specification.
E_VSP_PARA_UDS_CLIP	-352	The *clip* of vsp_uds_t was out of specification. Note: When *alpha* is VSP_ALPHA_ON.
E_VSP_PARA_UDS_ALPHA	-353	The *alpha* of vsp_uds_t was out of specification.
E_VSP_PARA_UDS_COMP	-354	The *complement* of vsp_uds_t 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 vsp_uds_t was out of specification.
E_VSP_PARA_UDS_XRATIO	-356	The *x_ratio* of vsp_uds_t was less than 0x100.
E_VSP_PARA_UDS_YRATIO	-357	The *y_ratio* of vsp_uds_t was less than 0x100.
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_ADR	-600	The *hard_addr* of vsp_lut_t was null pointer.
E_VSP_PARA_LUT_SIZE	-601	The *tbl_num* of vsp_lut_t was out of range 1-256.
E_VSP_PARA_LUT_CONNECT	-602	The *connect* of vsp_lut_t was out of specification.
E_VSP_PARA_CLU_MODE	-610	The *mode* of vsp_clu_t was out of specification.
E_VSP_PARA_CLU_ADR	-611	The *hard_addr* of vsp_clu_t was null pointer.
E_VSP_PARA_CLU_SIZE	-613	The *tbl_num* of vsp_clu_t was out of range.

E_VSP_PARA_CLU_CONNECT	-614	The *connect* of vsp_clu_t 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 vsp_hst_t was out of specification.
E_VSP_PARA_HSI_NOTHSV	-640	Color space for input to the HSI was not the HSV.
E_VSP_PARA_HSI_CONNECT	-641	The *connect* of vsp_hsi_t was out of specification.
E_VSP_PARA_HGO_ADR	-660	The *hard_addr* or *virt_addr* of vsp_hgo_t was null pointer.
		The *hard_addr* or *virt_addr* weren't a multiple of 256.
E_VSP_PARA_HGO_WIDTH	-661	The *width* of vsp_hgo_t was out of 1-8190.
E_VSP_PARA_HGO_HEIGHT	-662	The *height* of vsp_hgo_t was out of 1-8190.
E_VSP_PARA_HGO_XOFFSET	-663	Calculating value of the *width* + *x_offset* was greater than 8190.
E_VSP_PARA_HGO_YOFFSET	-664	Calculating value of the *height* + *y_offset* was greater than 8190.
E_VSP_PARA_HGO_BINMODE	-665	The *binary_mode* of vsp_hgo_t was out of specification.
E_VSP_PARA_HGO_MAXRGB	-669	The *maxrgb_mode* of vsp_hgo_t was out of specification.
E_VSP_PARA_HGO_XSKIP	-666	The *x_skip* of vsp_hgo_t was out of specification.
E_VSP_PARA_HGO_YSKIP	-667	The *y_skip* of vsp_hgo_t was out of specification.
E_VSP_PARA_HGO_SMMPT	-668	The *sampling* of vsp_hgo_t was out of specification.
E_VSP_PARA_HGO_STEP	-730	The *step* of vsp_hgo_t was out of specification.
E_VSP_PARA_HGT_ADR	-670	The *hard_addr* or *virt_addr* of vsp_hgt_t was null pointer.
		The *hard_addr* or *virt_addr* weren't a multiple of 256.
E_VSP_PARA_HGT_WIDTH	-671	The *width* of vsp_hgt_t was out of range 1-8190.
E_VSP_PARA_HGT_HEIGHT	-672	The *height* of vsp_hgt_t was out of range 1-8190.
E_VSP_PARA_HGT_XOFFSET	-673	Calculating value of the *width* + *x_offset* was greater than 8190.
E_VSP_PARA_HGT_YOFFSET	-674	Calculating value of the *height* + *y_offset* was greater than 8190.
E_VSP_PARA_HGT_AREA	-675	The *area* of vsp_hgt_t was out of specification.
E_VSP_PARA_HGT_XSKIP	-676	The *x_skip* of vsp_hgt_t was out of specification.
E_VSP_PARA_HGT_YSKIP	-677	The *y_skip* of vsp_hgt_t was out of specification.
E_VSP_PARA_HGT_SMMPT	-678	The *sampling* of vsp_hgt_t was out of specification.
E_VSP_PARA_SHP_INYUV	-690	Color space for input to the SHP was not the YUV.
E_VSP_PARA_SHP_WIDTH	-691	Image horizontal size for input to the SHP was out of range.
E_VSP_PARA_SHP_HEIGHT	-692	Image vertical size for input to the SHP was out of range.
E_VSP_PARA_SHP_MODE	-693	The *mode* of vsp_shp_t was out of specification.
E_VSP_PARA_SHP_CONNECT	-694	The *connect* of vsp_shp_t was out of specification.
E_VSP_PARA_NOSRU	-650	The *sru* of vsp_ctrl_t was null pointer.
E_VSP_PARA_NOUDS	-651	The *uds* of vsp_ctrl_t was null pointer.
E_VSP_PARA_NOLUT	-652	The *lut* of vsp_ctrl_t was null pointer.
E_VSP_PARA_NOCLU	-653	The *clu* of vsp_ctrl_t was null pointer.
E_VSP_PARA_NOHST	-654	The *hst* of vsp_ctrl_t was null pointer.
E_VSP_PARA_NOHSI	-655	The *hsi* of vsp_ctrl_t was null pointer.
E_VSP_PARA_NOBRU	-656	The *bru* of vsp_ctrl_t was null pointer.
E_VSP_PARA_NOHGO	-657	The *hgo* of vsp_ctrl_t was null pointer.
E_VSP_PARA_NOHGT	-658	The *hgt* of vsp_ctrl_t was null pointer.
E_VSP_PARA_NOSHP	-659	The *shp* of vsp_ctrl_t was null pointer.
E_VSP_PARA_DL_ADR	-680	The *hard_addr* or *virt_addr* of vsp_dl_t was null pointer.
E_VSP_PARA_DL_SIZE	-681	The *tbl_num* was out of range.

8. FDP driver parameters

8.1. fdp_start_t

The following is described about the member of fdp_star_t structure.

```
struct fdp_start_t {
      unsigned char              fdpgo;
      struct fdp_fproc_t         *fproc_par ;
};
```

Member	Direction	Contents
unsigned char *fdpgo*	Input	Frame processing request When using half-rate, switch FDP_NOGO and FDPGO. Example: n*2 sequence specify FDP_GO, n*2+1 sequence specify FDP_NOGO. When using full-rate, FDP_NOGO is not used. FDP_NOGO (0): do not request. (update internal sequence) FDP_GO (1): request frame processing.
struct fdp_fproc_t **fproc_par*	Input	Pointer to a frame processing parameter. This parameter valid in case of *fdpgo* = “FDP_GO”. In case of *fdpgo* = “FDP_GO”, do not specify null pointer.

8.2. fdp_fproc_t

The following is described about the member of fdp_fproc_t structure.

```
struct fdp_fproc_t {
    struct fdp_seq_t        *seq_par;
    struct fdp_pic_t        *in_pic ;
    unsigned char           last_seq_indicator ;
    unsigned char           current_field ;
    unsigned char           interpolated_line ;
    unsigned char           out_format ;
    struct fdp_imgbuf_t     *out_buf ;
    struct fdp_refbuf_t     *ref_buf ;
    struct fcp_info_t       *fcp_par ;
    struct fdp_ipc_t        *ipc_par ;
};
```

Member	Direction	Contents
struct fdp_seq_t *seq_par	Input	Pointer to a struct of sequence parameter. If specify null pointer, the FDP uses previous setting. If specify not null pointer, the FDP recognizes new sequence start.
struct fdp_pic_t *in_pic	Input	Pointer to a struct of input picture. Do not specify null pointer.
unsigned char *last_seq_indicator*	Input	Last sequence indication. Not used. Note: This is legacy member. It was setting of changing to forced 2D-IPC processing. In the new design argument, when you want changing to forced 2D-IPC processing, specify FDP_SEQ_INTER_2D to *seq_mode*.
unsigned char *current_field*	Input	Current field parity indication Set current field parity in case of interlace mode. In case of progressive mode, ignore this member. FDP_CF_TOP (0): Top field. FDP_CF_BOTTOM (1): Bottom field
unsigned char *interpolated_line*	Input	Select of interpolated lines. When detecting film mode of *telecine_mode* is selected, specify interpolated lines. When specify other than detecting film mode, ignore this member. FDP_DIM_PREV (3): Select previous field for interpolated lines. FDP_DIM_NEXT (4): Select next field for interpolated lines.

unsigned char *out_format*	Input	Output format. FDP_YUV420 (0): YUV420 semi-planar(NV12) FDP_YUV420_YU12 / FDP_YUV420_YV12 (1): YUV420 planar(YU12, YV12) *1 FDP_YUV420_NV21 (2): YUV420 semi-planar(NV21) FDP_YUV422_NV16 (3): YUV422 semi-planar(NV16) FDP_YUV422_YUY2 (4): YUV422 packed(YUY2) FDP_YUV422_UYVY (5): YUV422 packed(UYVY) FDP_YUV422_PLANAR (6): YUV422 planar(YV16) *1 FDP_YUV444_PLANAR (7): YUV444 planar *1 Note: Output format of FCNL compression support is planar format.
struct fdp_imgbuf_t **out_buf*	Input	Pointer to a struct of output buffer Do not specify null pointer. Note: If you use a FCNL compression (*fcnl* = FCP_FCNL_ENABLE), a stride and buffer address set multiple of 256.
struct fdp_refbuf_t **ref_buf*	Input	Pointer to a struct of reference buffer. Do not specify null pointer.
struct fcp_info_t **fcp_par*	Input	Frame compression setting Pointer to a frame compression setting structure.
struct fdp_ipc_t **ipc_par*	Input	De-interlace setting. Pointer to a de-interlace setting structure. Progressive or 2D-IPC is invalid. If specify null pointer, the FDP uses default setting. About Detail default setting, refer to struct fdp_ipc_t member.

8.2.1. fdp_seq_t

The following is described about the member of fdp_seq_t structure.

```
struct fdp_seq_t {
    unsigned char       seq_mode ;
    unsigned char       telecine_mode ;
    unsigned short      in_width ;
    unsigned short      in_height ;
};
```

Member	Direction	Contents
unsigned char *seq_mode*	Input	Sequence mode. FDP_SEQ_PROG (0): progressive mode. FDP_SEQ_INTER (1): Adaptive 2D/3D interlace mode. FDP_SEQ_INTER_2D (3): Fixed 2D interlace mode.
unsigned char *telecine_mode*	Input	Telecine mode. FDP_TC_OFF (0): Disable telecine detect mode. Normal 2D/3D-IPC mode. FDP_TC_FORCED_PULL_DOWN (2): Forced 2-3 pull down mode. FDP_TC_INTERPOLATED_LINE (3): Interpolated line mode.
unsigned short *in_width*	Input	Input picture horizontal size. 32-8190 pixel even number only permit.
unsigned short *in_height*	Input	Input picture vertical size. In interlace sequence, 16-4095 line permit. In progressive sequence, 32-8190 line permit.

8.2.2. fdp_pic_t

The following is described about the member of fdp_pic_t structure.

```
struct fdp_pic_t {
    unsigned long       picid;
    unsigned char       chroma_format;
    unsigned short      width;
    unsigned short      height;
    unsigned char       progressive_sequence;
    unsigned char       progressive_frame;
    unsigned char       picture_structure;
    unsigned char       repeat_first_field;
    unsigned char       top_field_first;
};
```

Member	Direction	Contents
unsigned long *picid*	Input	Picture ID. Set optional value. This value use identification each frame for user. This value reflects to *picid* which are members of struct fdp_status_t.
unsigned char *chroma_format*	Input	Input format. FDP_YUV420 (0): YUV420 semi-planar(NV12) *1 FDP_YUV420_YU12 / FDP_YUV420_YV12 (1): YUV420 planar(YU12, YV12) FDP_YUV420_NV21 (2): YUV420 semi-planar(NV21) *1 FDP_YUV422_NV16 (3): YUV422 semi-planar(NV16) FDP_YUV422_YUY2 (4): YUV422 packed(YUY2) FDP_YUV422_UYVY (5): YUV422 packed(UYVY) FDP_YUV422_PLANAR (6): YUV422 planar(YV16) FDP_YUV444_PLANAR (7): YUV444 planar Note: Input format of TL conversion supports FDP_YUV420 or FDP_YUV420_NV21.
unsigned short *width*	Input	Horizontal size When specify FDP_TC_FORCED_PULL_DOWN to *telecine_mode* (Forced 2-3 pull-down mode), this member is valid. Note: Need consistency with *in_width* of fdp_seq_t.
unsigned short *height*	Input	Vertical size When specify FDP_TC_FORCED_PULL_DOWN to *telecine_mode* (Forced 2-3 pull-down mode), this member is valid. Note: Need consistency with *in_height* of fdp_seq_t.
unsigned char *progressive_sequence*	Input	Decode information progressive_sequence. When specify FDP_TC_FORCED_PULL_DOWN to *telecine_mode* (Forced 2-3 pull-down mode), this member is valid. Note: Need consistency with *seq_mode* of fdp_seq_t. (When *seq_mode* = “FDP_SEQ_PROG”, *progressive_sequence* = 1, other case *progressive_sequence* = 0).

unsigned char *progressive_frame*	Input	Decode information progressive_frame. When specify FDP_TC_FORCED_PULL_DOWN to *telecine_mode* (Forced 2-3 pull-down mode), this member is valid.
unsigned char *picture_structure*	Input	Decode information picture_structure. When specify FDP_TC_FORCED_PULL_DOWN to *telecine_mode* (Forced 2-3 pull-down mode), this member is valid.
unsigned char *repeat_first_field*	Input	Decode information repeat_first_field. When specify FDP_TC_FORCED_PULL_DOWN to *telecine_mode* (Forced 2-3 pull-down mode), this member is valid.
unsigned char *top_field_first*	Input	Decode information top_field_first. When specify FDP_TC_FORCED_PULL_DOWN to *telecine_mode* (Forced 2-3 pull-down mode), this member is valid.

8.2.3. fdp_imgbuf_t

The following is described about the member of fdp_imgbuf_t structure.

```
struct fdp_imgbuf_t {
    unsigned int        addr ;
    unsigned int        addr_c0 ;
    unsigned int        addr_c1 ;
    unsigned short      stride ;
    unsigned short      stride_c ;
};
```

Member	Direction	Contents
unsigned int *addr*	Input	Y buffer address Set physical address. Do not specify null pointer YCbCr/Planar, Semi-Planar format: Y plane address. YCbCr/Packed format: Y/Cb/Cr plane address.
unsigned int *addr_c0*	Input	C0 buffer address Set physical address. When planar or semi-planar format, do not specify null pointer YCbCr/Planar format: Cb plane address. YCbCr/Semi-planar format: Cb/Cr plane address.
unsigned int *addr_c1*	Input	C1 buffer address Set physical address. When planar, do not specify null pointer YCbCr/Planar format: Cr plane address.
unsigned short *stride*	Input	Buffer width (Y buffer) Specify Y buffer horizontal size by 1pixel unit. Set greater than input picture horizontal size.
unsigned short *stride_c*	Input	Buffer width (C buffer) Specify C buffer horizontal size.

8.2.4. fdp_refbuf_t

The following is described about the member of fdp_refbuf_t structure.

```
struct fdp_regbuf_t {
        struct fdp_imgbuf_t             *next_buf;
        struct fdp_imgbuf_t             *cur_buf;
        struct fdp_imgbuf_t             *prev_buf;
};
```

Member	Direction	Contents
struct fdp_imgbuf_t **next_buf*	Input	Pointer to next read buffer. Specify next field information. When you set FDP_SEQ_INTER to *seq_mode* of struct fdp_seq_t, this member will be valid. In other case, ignore this member. The timing of loading is different by the *telecine_mode. About detail,* refer to section 8.5.1
struct fdp_imgbuf_t **cur_buf*	Input	Pointer to current read buffer. Specify current field or frame information. Do not specify null pointer.
struct fdp_imgbuf_t **prev_buf*	Input	Pointer to previous read buffer. Specify previous field information. When you set FDP_SEQ_INTER to *seq_mode* of struct fdp_seq_t, this member will be valid. In other case, ignore this member. The timing of loading is different by the *telecine_mode. About detail,* refer to section 8.5.1

8.2.5. fcp_info_t

The following is described about the member of fcp_info_t structure.

```
struct fcp_info_t {
    unsigned char       fcnl ;
    unsigned char       tlen ;
    unsigned short      pos_y ;
    unsigned short      pos_c ;
    unsigned short      stride_div16 ;
    unsigned int        ba_anc_prev_y ;
    unsigned int        ba_anc_cur_y ;
    unsigned int        ba_anc_next_y ;
    unsigned int        ba_anc_cur_c ;
    unsigned int        ba_ref_prev_y ;
    unsigned int        ba_ref_cur_y ;
    unsigned int        ba_ref_next_y ;
    unsigned int        ba_ref_cur_c ;
};
```

Member	Direction	Contents
unsigned char *fcnl*	Input	Renesas near-lossless compression setting. FCP_FCNL_DISABLE (0): Disable FCP_FCNL_ENABLE (1): Enable Note: Renesas near-lossless decompression is executed by DBSC4, DDR3/4 memory controller. When FCNL is enable, specify decompression area to destination buffer.
unsigned char *tlen*	Input	Tile/Linear conversion setting. FCP_TL_DISABLE (0): Disable FCP_TL_ENABLE (1): Enable
unsigned short *pos_y*	Input	Vertical position of tile address luma plane which FDP start read. (0 to 8189)
unsigned short *pos_c*	Input	Vertical position of tile address chroma plane which FDP start read. (0 to 4094)
unsigned short *stride_div16*	Input	Memory stride of the tile addressing image. The *stride_div16* specifies the picture of the tile address image. It must be specified so that *stride_div16* * 16 is exponent of 2 and larger than or equal to 128.
unsigned int *ba_anc_prev_y*	Input	The base address of the ancillary information of the plane. This member is corresponding to the luma plane of previous frame (field). It must be specified in 128 byte unit.
unsigned int *ba_anc_cur_y*	Input	The base address of the ancillary information of the plane. This member is corresponding to the luma plane of current frame (field). It must be specified in 128 byte unit.

unsigned int *ba_anc_next_y*	Input	The base address of the ancillary information of the plane. This member is corresponding to the luma plane of next frame (field). It must be specified in 128 byte unit.
unsigned int *ba_anc_cur_c*	Input	The base address of the ancillary information of the plane. This member is corresponding to the chroma plane of current frame (field). It must be specified in 128 byte unit.
unsigned int *ba_ref_prev_y*	Input	The base address of the tile addressing image of the plane. This member is corresponding to the luma plane of previous frame (field). It must be specified in 16384 byte unit.
unsigned int *ba_ref_cur_y*	Input	The base address of the tile addressing image of the plane. This member is corresponding to the luma plane of current frame (field). It must be specified in 16384 byte unit.
unsigned int *ba_ref_next_y*	Input	The base address of the tile addressing image of the plane. This member is corresponding to the luma plane of next frame (field). It must be specified in 16384 byte unit.
unsigned int *ba_ref_cur_c*	Input	The base address of the tile addressing image of the plane. This member is corresponding to the chroma plane of current frame (field). It must be specified in 16384 byte unit.

8.2.6. fdp_ipc_t

The following is described about the member of fdp_ipc_t structure.

```
struct fdp_ipc_t {
    unsigned char       cmb_ofst ;
    unsigned char       cmb_max ;
    unsigned char       cmb_gard ;
};
```

Member	Direction	Contents
unsigned char *cmb_ofst*	Input	Comb detection parameter setting 1. Default setting is 0x20. Refer to the H/W manual for setting other than the above.
unsigned char *cmb_max*	Input	Comb detection parameter setting 2. Default setting is 0x00. Refer to the H/W manual for setting other than the above.
unsigned char *cmb_gard*	Input	Comb detection parameter setting 3. Default setting is 0x40. Refer to the H/W manual for setting other than the above.

8.3. fdp_status_t

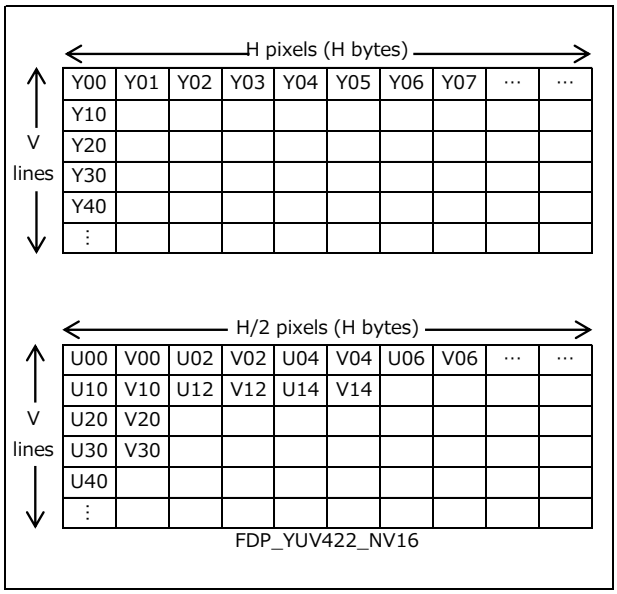
The following is described about the member of fdp_status_t structure.

```
struct fdp_status_t {
        unsigned long           picid;
        unsigned int            vcycle;
        unsigned int            sensor[18];
};
```

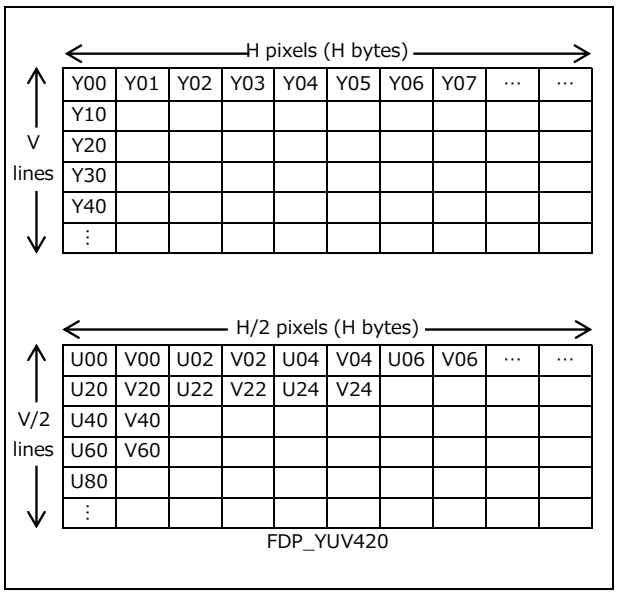
Member	Direction	Contents
unsigned long *picid*	Output	Picture ID of output picture. Return of value that specify *picid* of fdp_pic_t
unsigned int *vcycle*	Output	Number of cycle of the previous frame processing. This member returns value from getting FD1_CTL_VCYCLE_STAT register. Refer to H/W manual.
unsigned int *sensor[18]*	Output	Sensor information This member returns value from getting FD1_SENSOR_m register. (*sensor[m]* equal FD1_SENSOR_m. m = 0, 1, 2 … 17.) Refer to H/W manual.

8.4. Format

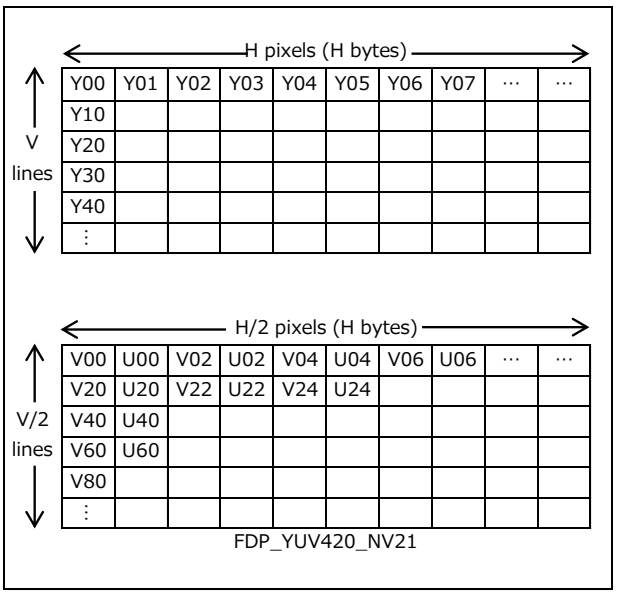
8.4.1. YCbCr (Semi planar) format



FDP_YUV422_NV16

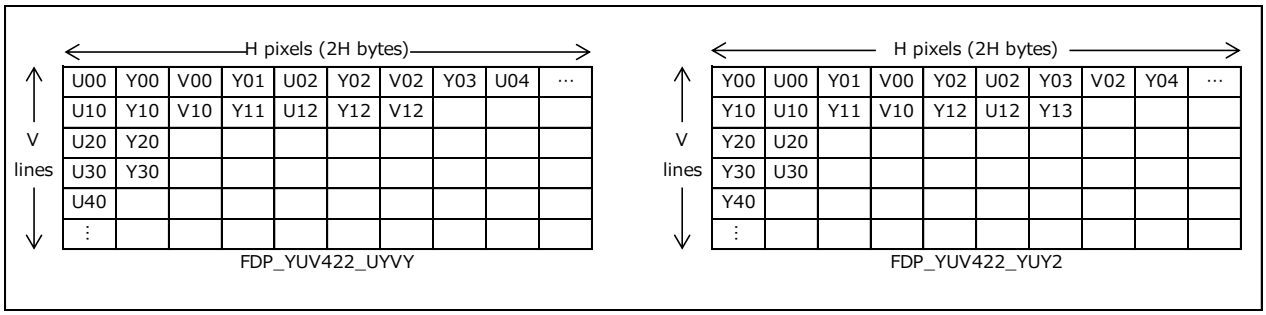


FDP_YUV420

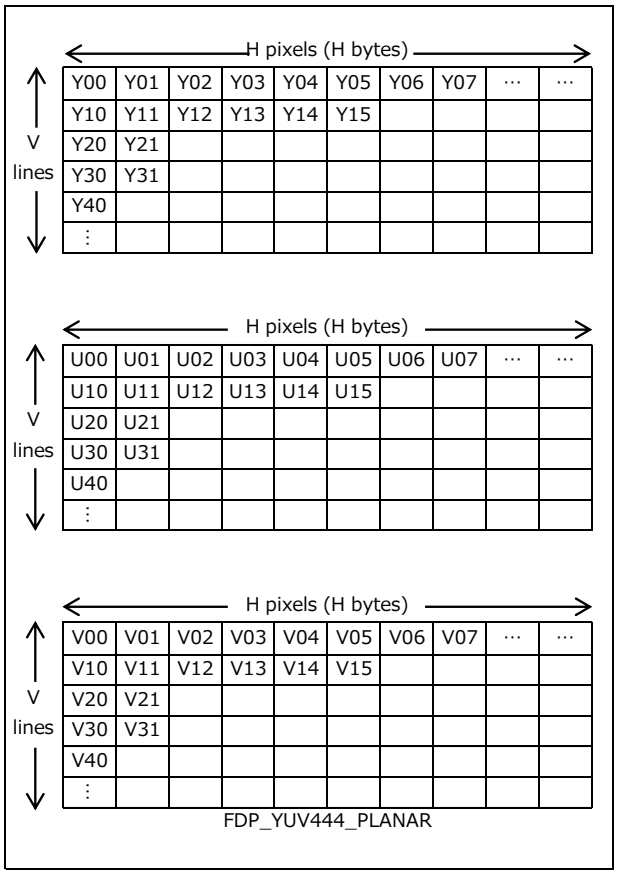


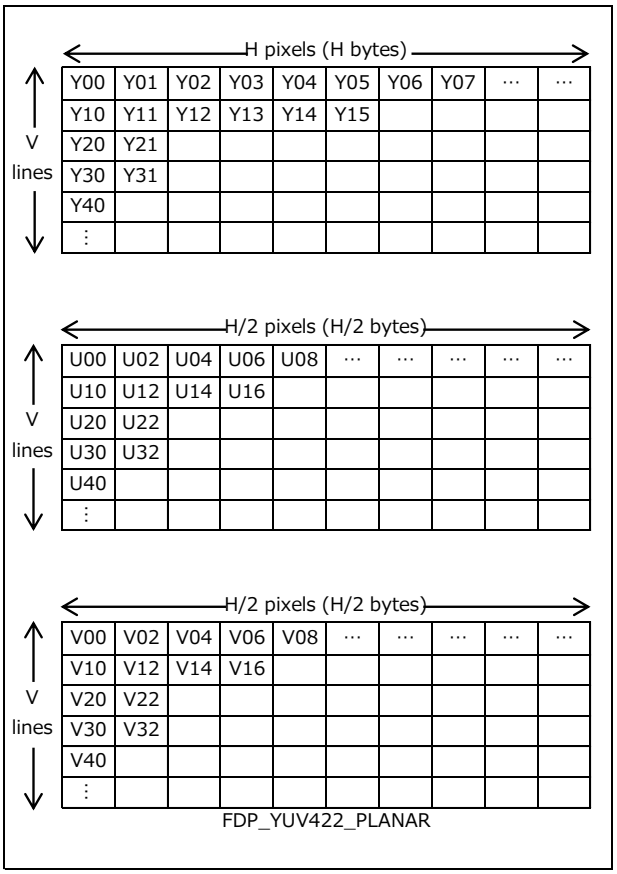
FDP_YUV420_NV21

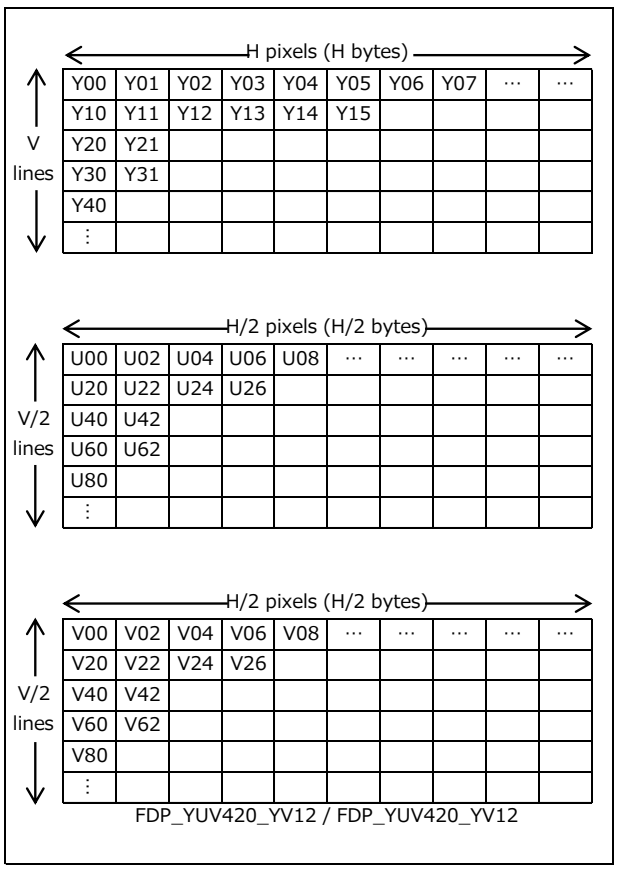
8.4.2. YCbCr (Packed) format



8.4.3. YCbCr (Planar) format







8.5. Sequence

8.5.1. Interlace (3D-IPC)

8.5.1.1. Normal

16.67ms (60fps)

			frame no.									
(a) input			1	2	3	4	5	6	7	8	9	10
fdp_go			FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO
fproc_par	*seq_par*	*seq_mode*	FDP_SEQ_INTER	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
		telechine_mode	FDP_TC_OFF	-	-	-	-	-	-	-	-	-
		in_pic progressive_sequence	-	-	-	-	-	-	-	-	-	-
		progressive_frame	-	-	-	-	-	-	-	-	-	-
		picture_structure	-	-	-	-	-	-	-	-	-	-
		repeat_first_field	-	-	-	-	-	-	-	-	-	-
		top_field_first	-	-	-	-	-	-	-	-	-	-
		last_seq_indicator	0	0	0	0	0	0	0	0	0	1
		current_field	FDP_CF_TOP	FDP_CF_BOTTOM	FDP_CF_TOP	FDP_CF_BOTTOM	FDP_CF_TOP	FDP_CF_BOTTOM	FDP_CF_TOP	FDP_CF_BOTTOM	FDP_CF_TOP	FDP_CF_BOTTOM
	ref_buf	*prev_buf* Note)	bottom-1	top0	bottom0	top1	bottom1	top2	bottom2	top3	bottom3	top4
		cur_buf	top0	bottom0	top1	bottom1	top2	bottom2	top3	bottom3	top4	bottom4
		next_buf Note)	bottom0	top1	bottom1	top2	bottom2	top3	bottom3	top4	bottom4	top5
(b) output	*out_buf*		frame0	frame1	frame2	frame3	frame4	frame5	frame6	frame7	frame8	frame9

Note) you can specify null pointer.
If you specify null pointer to prev_buff or next_buf, FDP Manager will change 2D-IPC.

8.5.1.2. Half-rate (60fps vsync)

16.67ms (60fps)

			frame no.								
(a) input			1	2	3	4	5	6	7	8	9
fdp_go			FDP_GO	FDP_NOGO	FDP_GO	FDP_NOGO	FDP_GO	FDP_NOGO	FDP_GO	FDP_NOGO	FDP_GO
fproc_par	*seq_par*	*seq_mode*	FDP_SEQ_INTER	-	NULL	-	NULL	-	NULL	-	NULL
		telechine_mode	FDP_TC_OFF	-	-	-	-	-	-	-	-
		in_pic progressive_sequence	-	-	-	-	-	-	-	-	-
		progressive_frame	-	-	-	-	-	-	-	-	-
		picture_structure	-	-	-	-	-	-	-	-	-
		repeat_first_field	-	-	-	-	-	-	-	-	-
		top_field_first	-	-	-	-	-	-	-	-	-
		last_seq_indicator	0	-	0	-	0	-	0	-	1
		current_field	FDP_CF_TOP	-	FDP_CF_TOP	-	FDP_CF_TOP	-	FDP_CF_TOP	-	FDP_CF_TOP
	ref_buf	*prev_buf* Note)	bottom-1	-	bottom0	-	bottom1	-	bottom2	-	bottom3
		cur_buf	top0	-	top1	-	top2	-	top3	-	top4
		next_buf Note)	bottom0	-	bottom1	-	bottom2	-	bottom3	-	bottom4
(b) output	*out_buf*		frame0	-	frame2	-	frame4	-	frame6	-	frame8

Note) you can specify null pointer.
If you specify null pointer to prev_buff or next_buf, FDP Manager will change 2D-IPC.

8.5.1.3. Half-rate (30fps vsync)

33.33ms (30fps)

			frame no.				
(a) input			1	2	3	4	5
fdp_go			FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO
fproc_par	*seq_par*	*seq_mode*	FDP_SEQ_INTER	NULL	NULL	NULL	NULL
		telechine_mode	FDP_TC_OFF	-	-	-	-
		in_pic progressive_sequence	-	-	-	-	-
		progressive_frame	-	-	-	-	-
		picture_structure	-	-	-	-	-
		repeat_first_field	-	-	-	-	-
		top_field_first	-	-	-	-	-
		last_seq_indicator	0	0	0	0	1
		current_field	FDP_CF_TOP	FDP_CF_TOP	FDP_CF_TOP	FDP_CF_TOP	FDP_CF_TOP
	ref_buf	*prev_buf* Note)	bottom-1	bottom0	bottom1	bottom2	bottom3
		cur_buf	top0	top1	top2	top3	top4
		next_buf Note)	bottom0	bottom1	bottom2	bottom3	bottom4
(b) output	*out_buf*		frame0	frame2	frame4	frame6	frame8

Note) you can specify null pointer.
If you specify null pointer to prev_buff or next_buf, FDP Manager will change 2D-IPC.

8.5.1.4. 2:3 pull-down

			16.67ms (60fps)									
			frame no.									
(a) input			1	2	3	4	5	6	7	8	9	10
fdp_go			FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO
fproc_par	*seq_par*	*seq_mode*	FDP_SEQ_INTER	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
		telechine_mode	P_TC_FORCED_PULL_DOV	-	-	-	-	-	-	-	-	-
		in_pic progressive_sequence	0	0	0	0	0	0	0	0	0	0
		progressive_frame	1	1	1	1	1	1	1	1	1	1
		picture_structure	3	3	3	3	3	3	3	3	3	3
		repeat_first_field	1	1	1	0	0	1	1	1	0	0
		top_field_first	1	1	1	0	0	0	0	0	1	1
	last_seq_indicator		0	0	0	0	0	0	0	0	0	1
	current_field		FDP_CF_TOP	FDP_CF_BOTTOM	FDP_CF_TOP	FDP_CF_BOTTOM	FDP_CF_TOP	FDP_CF_BOTTOM	FDP_CF_TOP	FDP_CF_BOTTOM	FDP_CF_TOP	FDP_CF_BOTTOM
	ref_buf	*prev_buf*	bottom0 Note)	top0	bottom0	top1 Note)	bottom1	top2 Note)	bottom2	top2	bottom3 Note)	top3
		cur_buf	top0	bottom0	top0	bottom1	top1	bottom2	top2	bottom2	top3	bottom3
		next_buf	bottom0	top0 Note)	bottom0 Note)	top1	bottom1 Note)	top2	bottom2 Note)	top2 Note)	bottom3	top3 Note)
(b) output	*out_buf*		frame0	frame0	frame0	frame1	frame1	frame2	frame2	frame2	frame3	frame3

Note) you can specify top (or bottom) field to this member.
In this case, you can specify null pointer to other member that specify same field.

8.5.1.5. 2:3 pull-down half-rate (60fps vsync)

			16.67ms (60fps)								
			frame no.								
(a) input			1	2	3	4	5	6	7	8	9
fdp_go			FDP_GO	FDP_NOGO	FDP_GO	FDP_NOGO	FDP_GO	FDP_NOGO	FDP_GO	FDP_NOGO	FDP_GO
fproc_par	*seq_par*	*seq_mode*	FDP_SEQ_INTER	-	NULL	-	NULL	-	NULL	-	NULL
		telechine_mode	P_TC_FORCED_PULL_DOV	-	-	-	-	-	-	-	-
		in_pic progressive_sequence	0	-	0	-	0	-	0	-	0
		progressive_frame	1	-	1	-	1	-	1	-	1
		picture_structure	3	-	3	-	3	-	3	-	3
		repeat_first_field	1	-	1	-	0	-	1	-	0
		top_field_first	1	-	1	-	0	-	0	-	1
	last_seq_indicator		0	-	0	-	0	-	0	-	1
	current_field		FDP_CF_TOP	-	FDP_CF_TOP	-	FDP_CF_TOP	-	FDP_CF_TOP	-	FDP_CF_TOP
	ref_buf	*prev_buf*	bottom0 Note)	-	bottom0	-	bottom1	-	bottom2	-	bottom3 Note)
		cur_buf	top0	-	top0	-	top1	-	top2	-	top3
		next_buf	bottom0	-	bottom0 Note)	-	bottom1 Note)	-	bottom2 Note)	-	bottom3
(b) output	*out_buf*		frame0	-	frame0	-	frame1	-	frame2	-	frame3

Note) you can specify top (or bottom) field to this member.
In this case, you can specify null pointer to other member that specify same field.

8.5.1.6. 2:3 pull-down half-rate (30fps vsync)

			33.33ms (30fps)				
			frame no.				
(a) input			1	2	3	4	5
fdp_go			FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO
fproc_par	*seq_par*	*seq_mode*	FDP_SEQ_INTER	NULL	NULL	NULL	NULL
		telechine_mode	FDP_TC_FORCED_PULL_DOWN	-	-	-	-
		in_pic progressive_sequence	0	0	0	0	0
		progressive_frame	1	1	1	1	1
		picture_structure	3	3	3	3	3
		repeat_first_field	1	1	0	1	0
		top_field_first	1	1	0	0	1
	last_seq_indicator		0	0	0	0	1
	current_field		FDP_CF_TOP	FDP_CF_TOP	FDP_CF_TOP	FDP_CF_TOP	FDP_CF_TOP
	ref_buf	*prev_buf*	bottom0 Note)	bottom0	bottom1	bottom2	bottom3
		cur_buf	top0	top0	top1	top2	top3
		next_buf	bottom0	bottom0 Note)	bottom1 Note)	bottom2 Note)	bottom3 Note)
(b) output	*out_buf*		frame0	frame0	frame1	frame2	frame2

Note) you can specify top (or bottom) field to this member.
In this case, you can specify null pointer to other member that specify same field.

8.5.2. Interlace (2D-IPC)

8.5.2.1. Normal

16.67ms (60fps)

			frame no.	1	2	3	4	5	6	7	8	9	10
(a) input													
fdp_go				FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO
fproc_par	*seq_par*	*seq_mode*		FDP_SEQ_INTER_2D	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
		telechine_mode		FDP_TC_OFF	-	-	-	-	-	-	-	-	-
		in_pic progressive_sequence		-	-	-	-	-	-	-	-	-	-
		progressive_frame		-	-	-	-	-	-	-	-	-	-
		picture_structure		-	-	-	-	-	-	-	-	-	-
		repeat_first_field		-	-	-	-	-	-	-	-	-	-
		top_field_first		-	-	-	-	-	-	-	-	-	-
		last_seq_indicator		0	0	0	0	0	0	0	0	0	1
		current_field		FDP_CF_TOP	FDP_CF_BOTTOM	FDP_CF_TOP	FDP_CF_BOTTOM	FDP_CF_TOP	FDP_CF_BOTTOM	FDP_CF_TOP	FDP_CF_BOTTOM	FDP_CF_TOP	FDP_CF_BOTTOM
		ref_buf	*prev_buf*	-	-	-	-	-	-	-	-	-	-
			cur_buf	top0	bottom0	top1	bottom1	top2	bottom2	top3	bottom3	top4	bottom4
			next_buf	-	-	-	-	-	-	-	-	-	-
(b) output	*out_buf*			frame0	frame1	frame2	frame3	frame4	frame5	frame6	frame7	frame8	frame9

8.5.3. Progressive

8.5.3.1. Normal

16.67ms (60fps)

			frame no.	1	2	3	4	5	6	7	8	9	10
(a) input													
fdp_go				FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO
fproc_par	*seq_par*	*seq_mode*		FDP_SEQ_PROG	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
		telechine_mode		FDP_TC_OFF	-	-	-	-	-	-	-	-	-
		in_pic progressive_sequence		-	-	-	-	-	-	-	-	-	-
		progressive_frame		-	-	-	-	-	-	-	-	-	-
		picture_structure		-	-	-	-	-	-	-	-	-	-
		repeat_first_field		-	-	-	-	-	-	-	-	-	-
		top_field_first		-	-	-	-	-	-	-	-	-	-
		last_seq_indicator		0	0	0	0	0	0	0	0	0	1
		current_field		-	-	-	-	-	-	-	-	-	-
		ref_buf	*prev_buf*	-	-	-	-	-	-	-	-	-	-
			cur_buf	frame0	frame1	frame2	frame3	frame4	frame5	frame6	frame7	frame8	frame9
			next_buf	-	-	-	-	-	-	-	-	-	-
(b) output	*out_buf*			frame0	frame1	frame2	frame3	frame4	frame5	frame6	frame7	frame8	frame9

8.5.3.2. 2:3 pull-down

16.6ms (60fps)

			frame no.	1	2	3	4	5	6	7	8	9	10
(a) input													
fdp_go				FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO	FDP_GO
fproc_par	*seq_par*	*seq_mode*		FDP_SEQ_PROG	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
		telechine_mode		P_TC_FORCED_PULL_DOV	-	-	-	-	-	-	-	-	-
		in_pic progressive_sequence		1	1	1	1	1	1	1	1	1	1
		progressive_frame		-	-	-	-	-	-	-	-	-	-
		picture_structure		3	3	3	3	3	3	3	3	3	3
		repeat_first_field		1	1	1	1	1	1	1	1	1	1
		top_field_first		0	0	1	1	1	0	0	1	1	1
		last_seq_indicator		0	0	0	0	0	0	0	0	0	1
		current_field		-	-	-	-	-	-	-	-	-	-
		ref_buf	*prev_buf*	-	-	-	-	-	-	-	-	-	-
			cur_buf	frame0	frame0	frame1	frame1	frame1	frame0	frame0	frame1	frame1	frame1
			next_buf	-	-	-	-	-	-	-	-	-	-
(b) output	*out_buf*			frame0	frame0	frame1	frame1	frame1	frame0	frame0	frame1	frame1	frame1

8.6. Error code

Table 8-1 shows the detail error code of FDP. According to error code, please check argument.

Table 8-1 Detail of error code

Define name	Error code	Contains
E_FDP_INVALID_STATE	-105	When you specify occupy mode, FDP is working.
E_FDP_PARA_REFBUF	-253	The *ref_buf* of fdp_fproc_t was null pointer.
E_FDP_PARA_FDPGO	-301	The *fdpgo* of fdp_start_t was out of specification.
E_FDP_PARA_FPROCPAR	-302	The *fproc_par* of fdp_start_t was null pointer, when fdpgo was FDP_GO.
E_FDP_PARA_SEQPAR	-303	The *seq_par* of fdp_fproc_t was null pointer, immediately after the initialize.
E_FDP_PARA_INPIC	-305	The *in_pic* of fdp_fproc_t was null pointer.
E_FDP_PARA_OUTBUF	-306	The *out_buf* of fdp_fproc_t was null pointer.
E_FDP_PARA_SEQMODE	-307	The *seq_mode* of fdp_seq_t was out of specification.
E_FDP_PARA_TELECINEMODE	-308	The *telecine_mode* of fdp_seq_t was out of specification.
E_FDP_PARA_INWIDTH	-309	The *in_width* of fdp_seq_t was out of range.
		The *in_width* wasn't a multiple of 2.
E_FDP_PARA_INHEIGHT	-310	The *in_height* of fdp_seq_t was out of range.
E_FDP_PARA_PICWIDTH	-314	The *width* of fdp_pic_t was not equal *in_width* of fdp_seq_t.
E_FDP_PARA_PICHEIGHT	-315	The *height* of fdp_pic_t was not equal *in_height* of fdp_seq_t.
E_FDP_PARA_CHROMA	-316	The *chroma_format* of fdp_pic_t was out of specification.
E_FDP_PARA_PROGSEQ	-317	The *progressive_sequence* of fdp_pic_t was not equal mode specified *seq_mode* of fdp_seq_t.
E_FDP_PARA_PICSTRUCT	-318	The *picture_structure* of fdp_pic_t was out of standard.
E_FDP_PARA_REPEATTOP	-319	The *repeat_first_field* and *top_field_first* of fdp_pic_t were out of standard.
E_FDP_PARA_BUFREFRD0	-321	The *pref_buf* of fdp_refbuf_t was null pointer.
E_FDP_PARA_BUFREFRD1	-322	The *cur_buf* of fdp_refbuf_t was null pointer.
E_FDP_PARA_BUFREFRD2	-323	The *next_buf* of fdp_refbuf_t was null pointer.
E_FDP_PARA_LASTSTART	-329	The *last_start_indicator* of fdp_fproc_t was out of specification.
E_FDP_PARA_CF	-330	The *current_field* of fdp_fproc_t was out of specification.
E_FDP_PARA_INTERPOLATED	-331	The *interpolated_line* of fdp_fproc_t was out of specification when *telecine_mode* was FDP_TC_INTERPOLATED_LINE.
E_FDP_PARA_OUTFORMAT	-332	The *out_format* of fdp_fproc_t was out of specification.
E_FDP_PARA_SRC_ADDR	-350	The *addr* of reference buffer was null pointer.
E_FDP_PARA_SRC_ADDR_C0	-351	The *addr_c0* of reference buffer was null pointer.
E_FDP_PARA_SRC_ADDR_C1	-352	The *addr_c1* of reference buffer was null pointer.
E_FDP_PARA_SRC_STRIDE	-353	The *stride* of reference buffer was less than *in_width*.
E_FDP_PARA_SRC_STRIDE_C	-354	The *stride_c* of reference buffer was less than *in_width*.

E_FDP_PARA_DST_ADDR	-355	The *addr* of destination buffer was null pointer.
		The *addr* wasn't a multiple of 256 when fcnl enable.
E_FDP_PARA_DST_ADDR_C0	-356	The *addr_c0* of destination buffer was null pointer.
		The *addr_c0* wasn't a multiple of 256 when fcnl enable.
E_FDP_PARA_DST_ADDR_C1	-357	The *addr_c1* of destination buffer was null pointer.
		The *addr_c1* wasn't a multiple of 256 when fcnl enable.
E_FDP_PARA_DST_STRIDE	-358	The *stride* of destination buffer was less than *in_width*.
		The *stride* wasn't a multiple of 256 when fcnl enable.
E_FDP_PARA_DST_STRIDE_C	-359	The *stride_c* of destination buffer was less than *in_width*.
		The *stride_c* wasn't a multiple of 256 when fcnl enable.
E_FDP_PARA_STLMSK_ADDR	-360	Work buffer memory of specified initialize was null pointer.
E_FDP_PARA_FCNL	-400	The *fcnl* of fcp_info_t was out of specification.
E_FDP_PARA_TLEN	-401	The *tlen* of fcp_info_t was out of specification.
E_FDP_PARA_FCP_POS	-402	The *pos_c* or *pos_y* were out of range.
E_FDP_PARA_FCP_STRIDE	-403	The *fcp_stride* of fcp_info_t was out of specification.
E_FDP_PARA_BA_ANC	-404	The *ba_anc_*** of fcp_info_t was out of specification.
E_FDP_PARA_BA_REF	-405	The *ba_ref_*** of fcp_info_t was out of specification.

9. Restrictions and Notes

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

9.1. VSP’s Restrictions

- The VSP Manager takes over restrictions of hardware. Refer to H/W manual.
- The VSP Manager does not accept downscaling after super-resolution. Upscaling (include same size) after super-resolution or super-resolution after downscaling are possible.
- Legacy interface does not is guaranteed all of the functions.

9.2. FDP’s Restrictions

- The VSP Manager takes over restrictions of hardware. Refer to H/W manual.
- Legacy interface does not is guaranteed all of the functions.
- The VSP Manager does not check ancillary data. So when inputs invalid ancillary data, it will freeze FDP.

REVISION HISTORY	VSP Manager User's Manual: Software

Rev.	Date	Description	
		Page	Summary
1.50	Jan. 29, 2018	-	First Edition Issued.
1.51	Mar 28, 2018	-	Add R-Car E3 device support
		1	Add description about Mutual mode
1.52	Apr 11, 2018	62	Add note when rop_unit is used
1.53	Oct 29, 2018	-	How to Use This Manual - 1.Purpose and Target Readers Update reference document
2.00	Dec 25, 2018	-	Update AddressList
		6	Table 1-3 Update Platform name
2.01	Apr 17, 2019	-	Update AddressList
		-	How to Use This Manual - 1.Purpose and Target Readers Update reference documents
2.50	Apr. 21, 2021	-	Add Kernel 5.10 support
3.00	Dec.10, 2021	33,49	Fix VSP_FULL_COLOR setting expression
		-	Add Kernel v5.10.41 support

VSP Manager User's Manual: Software

Publication Date: Rev.1.50 Jan. 29, 2018
Rev.3.00 Dec.10, 2021

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 Corporation
TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

Renesas Electronics America Inc. Milpitas Campus
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics America Inc. San Jose Campus
6024 Silver Creek Valley Road, San Jose, CA 95138, USA
Tel: +1-408-284-8200, Fax: +1-408-284-2775

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

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

Renesas Electronics (China) Co., Ltd.
Room 101-T01, Floor 1, Building 7, Yard No. 7, 8th Street, Shangdi, Haidian District, Beijing 100085, China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai 200333, China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

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

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

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

Renesas Electronics Malaysia Sdn.Bhd.
Unit No 3A-1 Level 3A Tower 8 UOA Business Park, No 1 Jalan Pengaturcara U1/51A, Seksyen U1, 40150 Shah Alam, Selangor, Malaysia
Tel: +60-3-5022-1288, Fax: +60-3-5022-1290

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 7.0

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口

http://www.renesas.com

※営業お問合せ窓口の住所は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス株式会社 〒135-0061 東京都江東区豊洲3-2-24（豊洲フォレシア）

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：https://www.renesas.com/contact/

Colophon 3.1

VSP Manager
User's Manual: Software

Renesas Electronics Corporation